Appendix A

Determining Dynamic Cutoff
Priority CAC Threshold
A.1 Dynamic cutoff priority

This section presents analytical results for dynamic cutoff priority threshold, described as $m_{c}^{\text{tol}}$ (figure A.19a) for 1-Dimention (cell having left and right cells as neighbours as in figure A.19b) and 2-Dimentional (Cell ‘N’ having left, top, right and bottom cells as neighbours as in figure A.19b). The model is explained in section 4.1.2.3. The analytical results which are presented here are used for simulation for estimating cutoff thresholds for the current cell for ongoing calls in the neighbouring cells.

If $n$ denotes calls in the current cell $C$, then $r$, $l$ are the number of calls in neighbouring cells (right, left) respectively. For 2-D model the number of calls would be top($t$), right ($r$), bottom ($b$), left($l$). Let’s assume $P_s$ is the probability that the call remains in the same cell. Let $P_m/2$ and $P_m/4$ be the probability that call will be handed off to neighbouring cell during time $T$ for 1-D and 2-D model respectively. Also assume that the call can be handed over only once. Let $\lambda$ denote the new call arrival rate to any cell, and $\mu$ denotes the call departure rate. Let a new call admission request is made in addition to the state of its adjacent cells $T$ units of time after the call admission.

Figure A.1: (a) Traffic Model: Single Class, (b) 1-D Cluster, (c) 2-D Cluster
A.1.1 1-D Cluster for D-CAC

Let N be total number of channels available with each cell. Let n denotes calls in the current cell C, then r, l are the number of calls in neighbouring cells (right, left) respectively. Let's assume $P_s$ is the probability that the call remains in the same cell. Let $P_m / 2$ be the probability that call will be handed off to neighbouring cell during time T. Also assume that the call can be handed over only once. Let $\lambda$ denote the new call arrival rate to any cell, and $\mu$ denotes the call departure rate. Let a new call admission request is made in addition to the state of its adjacent cells T units of time after the call admission.

The simulation model of 5.1 is used with parameters for D-CAC as:

An average handoff time of 100s, and an estimation T period 20 s, $\mu = 1/500$, $\lambda / \mu = 0.5$, $1/\mu = 1/100$ and average call per neighboring cells $E(n) = 10$

$$p_s = e^{-\left(\mu + h\right) T} = 0.7866 \quad \text{and} \quad p_m = 1 - e^{-hT} = 0.1813$$

Where $a$ is tunable and equals 2.1 for N=20.

The figure A.2 and A.3 show analytical results for D-CAC for different number of channel for 1-D D-CAC.

Figure A.2: Dynamic Cutoff Threshold for N=20 and N=50
Analysis 1-D D-CAC:

It was observed that all the channels are available for new call assignment in the current/centre cell ‘C’, if there are no ongoing calls in the neighbourhood \((r = 0, l = 0)\). Effectively the D-CAC threshold is at \(N\). As number of ongoing calls in the neighboring cells increase, some channels are reserved for handling handoff calls from these cells, thus decreasing the threshold i.e. less number of channels are available for new calls. Thus after a certain level of neighboring calls are reached this threshold falls drastically thus increasing new call blocking probability and eventually causing quality degradation termed as congestion.

A.1.2 2-D Cluster for D-CAC

Figure A.1 (c) presents the 2-D model with 4 neighbors of cell ‘C’. The results presented here are analytical results for D-CAC for different number of channel for 2-D D-CAC. The traffic model is as in figure A.1 (a) simulation model followed is that of section 5.1 and other simulation parameters are presented in section A.1.1
Let $X$ be a 4-D array holding value $m_c^{tot}$ for calls being generated in each neighboring cell as well in current cell and satisfying eq. 4.13 i.e.

$$m_c^{tot} = \min(n_c, n_t, n_r, n_b, n_i)$$.

The resultant $X$ could not be plotted because of its complex 4-D nature.

The resultant $X$, in Table A.1 (a), for (N=20) shows that the maximum number of calls in cell C moves from 20 to approximately 14 as the adjacent calls moves from 1 to 18(max), no. of calls dropped to 3 and then 0, once the adjacent calls increase further.

For N=30, N=40 and N=50 the analytical values of maximum numbers of calls allowed were found. For N=40 and a= 3.092, the drop in number of allowed calls in cell C occurs at either of r, t, b or l reaching 37. For N=50 and a=3.48, the drop in number of allowed calls in cell C occurs at either of r, t, b or l reaching 47 as shown in Table A.1(b).

The tables represent following property of resultant $X$.

Let set $S$ represent array $X$.
Let $S = \{C_1, C_2, ..., C_4\}$  $1 \leq i \leq 4$
Let $A \subseteq S$ such that $|A| = 3$
Let $B \subseteq S$ such that $B = \overline{A}$
$Z_i = \{K_{i1}, K_{i2}, ..., K_{ij}\}$  $1 \leq j \leq 20$
if $K_{ij} = j$ for $\forall i$ where $Z_i \in A$
Let $Y \in B$  $1 \leq Y \leq 20$ Then $n_c^{tot} = F(j,Y)$

**Analysis 2-D D-CAC:**

Analytical results of table A.1 present that as the cell capacity increases the admission surface moves away from the origin and beyond a cutoff level of number of calls in the adjacent cells, the number of calls in the current cell C drops dramatically. Also as the probability to handoff is taken equal in all 4 directions, as any of the cell’s
number of calls reach cutoff threshold, the number of calls which could be admitted in the current cell C decreases. 'a' increases as cell capacity increases.

Table A.1: (a) 2-D D-CAC: N=20

<table>
<thead>
<tr>
<th>j</th>
<th>( n^\text{tot}_c = F(j,Y) ) For ( 1 \leq Y \leq 20 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20.000, 19.921, 19.843, 19.764, \ldots, 18.751, 18.674, 7.807, 0</td>
</tr>
<tr>
<td>2</td>
<td>19.764, 19.686, 19.608, 19.529, \ldots, 18.519, 18.442, 7.807, 0</td>
</tr>
<tr>
<td>3</td>
<td>19.529, 19.451, 19.373, 19.295, \ldots, 18.287, 18.210, 7.807, 0</td>
</tr>
<tr>
<td>4</td>
<td>19.295, 19.217, 19.139, 19.062, \ldots, 18.056, 17.979, 7.807, 0</td>
</tr>
<tr>
<td>5</td>
<td>19.062, 18.984, 18.906, 18.829, \ldots, 17.826, 17.749, 7.807, 0</td>
</tr>
<tr>
<td>6</td>
<td>18.829, 18.751, 18.674, 18.596, \ldots, 17.596, 17.519, 7.807, 0</td>
</tr>
<tr>
<td>7</td>
<td>18.596, 18.519, 18.442, 18.364, \ldots, 17.366, 17.290, 7.807, 0</td>
</tr>
<tr>
<td>8</td>
<td>18.364, 18.287, 18.210, 18.133, \ldots, 17.137, 17.061, 7.807, 0</td>
</tr>
<tr>
<td>9</td>
<td>18.133, 18.056, 17.979, 17.902, \ldots, 16.909, 16.832, 7.807, 0</td>
</tr>
<tr>
<td>10</td>
<td>17.902, 17.826, 17.749, 17.672, \ldots, 16.680, 16.604, 7.807, 0</td>
</tr>
<tr>
<td>11</td>
<td>17.672, 17.596, 17.519, 17.443, \ldots, 16.453, 16.377, 7.807, 0</td>
</tr>
<tr>
<td>12</td>
<td>17.443, 17.366, 17.290, 17.213, \ldots, 16.226, 16.150, 7.807, 0</td>
</tr>
<tr>
<td>13</td>
<td>17.213, 17.137, 17.061, 16.985, \ldots, 15.999, 15.923, 7.807, 0</td>
</tr>
<tr>
<td>14</td>
<td>16.985, 16.909, 16.832, 16.756, \ldots, 15.773, 15.697, 7.807, 0</td>
</tr>
<tr>
<td>15</td>
<td>16.756, 16.680, 16.604, 16.529, \ldots, 15.547, 15.471, 7.807, 0</td>
</tr>
<tr>
<td>16</td>
<td>16.529, 16.453, 16.377, 16.301, \ldots, 15.321, 15.246, 7.807, 0</td>
</tr>
<tr>
<td>17</td>
<td>16.301, 16.226, 16.150, 16.074, \ldots, 15.096, 15.021, 7.807, 0</td>
</tr>
<tr>
<td>18</td>
<td>16.074, 15.999, 15.923, 15.848, \ldots, 14.871, 14.796, 7.807, 0</td>
</tr>
<tr>
<td>19</td>
<td>7.807, 7.807, 7.807, \ldots, 7.807, 7.807, 7.807, 0</td>
</tr>
<tr>
<td>20</td>
<td>0, 0, 0, \ldots, 0, 0, 0</td>
</tr>
</tbody>
</table>
Table A.2: (b) 2-D D-CAC: N=50

<table>
<thead>
<tr>
<th>j</th>
<th>$n_c^{tot} = F(j,Y)$ For $1 \leq Y \leq 50$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>49.998, 49.918, 49.839, 49.759, ......, 46.44, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td>2</td>
<td>49.759, 49.680, 49.600, 49.521, ......, 46.21, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td>3</td>
<td>49.521, 49.442, 49.362, 49.28, ......, 45.97, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td></td>
<td>.................................................................................................</td>
</tr>
<tr>
<td>15</td>
<td>46.682, 46.604, 46.526, 46.447, ......, 43.18, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td>16</td>
<td>46.447, 46.369, 46.291, 46.213, ......, 42.95, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td></td>
<td>.................................................................................................</td>
</tr>
<tr>
<td>23</td>
<td>44.809, 44.732, 44.654, 44.576, ......, 41.33, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td>24</td>
<td>44.576, 44.499, 44.421, 44.344, ......, 41.10, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td></td>
<td>.................................................................................................</td>
</tr>
<tr>
<td>35</td>
<td>42.027, 41.950, 41.873, 41.796, ......, 38.58, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td>36</td>
<td>41.796, 41.719, 41.643, 41.566, ......, 38.36, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td></td>
<td>.................................................................................................</td>
</tr>
<tr>
<td>46</td>
<td>39.501, 39.425, 39.349, 39.272, ......, 36.09, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td>47</td>
<td>35.548, 35.548, 35.548, 35.548, ......, 35.548, 22.881, 10.276, 0</td>
</tr>
<tr>
<td>48</td>
<td>22.881, 22.881, 22.881, 22.881, ......, 22.881, 22.881, 10.276, 0</td>
</tr>
<tr>
<td>49</td>
<td>10.276, 10.276, 10.276, 10.276, ......, 10.276, 10.276, 10.276, 0</td>
</tr>
<tr>
<td>50</td>
<td>0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0</td>
</tr>
</tbody>
</table>
Simulation: Extended
B.1 Single Class Traffic

B.1.1 Single Class Traffic: C_NA-S-CAC

The results in figure B.1 establishes and validates that the analytical and simulated results for S-CAC (non agent and multi agents) are identical. It illustrates the analytical and simulated results for the blocking probabilities for static cutoff thresholds typically m=27. It is observed that as the cutoff threshold increased, the new call blocking probability decreased. This was due to more number of channels being available for new calls where as the handoff blocking probability increased as less number of channels were reserved for handoff calls. Also it was noted that the new call block probability moved from 0.2 to 0.85 for m=27.
These results are validated with the analytical model presented in section 4.2.

**B.1.2 Single Class traffic: C\_NA-S-DAC and C\_NA-D-CAC**

The figure B.2 shows the comparison of non agent based S-CACs \((m=21, m=25\) and \(m=27\)) and the D-CAC (where dynamic cutoff threshold is being calculated.)

It clearly shows that the handoff blocking probability is less as compared to that of S-CAC and new call blocking probability is significantly more. Also for D-CAC the new call blocking is less at lower traffic load.

![Figure B.2: Comparing S-CACs and D-CAC (\(P_{nb}\) & \(P_{hb}\))](image-url)
It is seen that the handoff dropping probability is less in D-CAC as compared to S-CAC.

B.1.2 Single Class traffic: MBCR & D-CAC

Figure B.3 and B.4 presents the simulated results of new call blocking and handoff call blocking probabilities for multi agent and non agent based MBCR schemes respectively according to the analytical model presented in section 4.1.2.4. It shows that the Fractional MBCR fairs well and blocking probability is lower as compared to its integral counterpart. This is because of the rounding effect. These results are used for verification of MAS-MBCR schemes.

![Call Blocking Probability for Multi Agent Based Distributed Architecture](image)

Figure B.3: Multi Agent Based (D-CAC & MBCR): (P_{nb} & P_{hb})
Figure B.4: Non Agent Based (D-CAC & MBCR): (P_{nb} & P_{hb})
B.2 Multi Class Traffic: D-CAC

The results in figures B.5-B.8 present the effect of queue size on call level parameters of S-CAC (M=27 and 21) also results are verified with its non agent based simulations.

Figure B.5: NA-S-CAC, MA-S-CAC (M=27&K=0, 1, and 2): (P_{nb})
Figure B.6: NA-S-CAC, MA-S-CAC (M=27 & K=0, 1, and 2): (P_{hb})
Figure B.7: NA-S-CAC, MA-S-CAC (M=21 & K=0, 1, and 2): (P_{nb})
The Studies of Multi-Agent System in Mobile Computing for Mobile Application

Figure B.8: NA-S-CAC, MA-S-CAC (M=21 & K=0, 1, and 2): ($P_{bh}$)
**Validation: Non Priority Based Multi Class S-CAC**

Figures B.9 and B.10 validate the results of multi agent based simulation of S-CAC for multi class, non priority based CAC with that of analytical results for new call blocking probability and handoff call blocking probability respectively.

![New Call Blocking Probability(Pnb) for Voice and Data for K=0](image)

**Figure B.9: Analytical and Multi Agent (MA) Non-Priority S-CAC: \( P_{\text{nb}} \)**
Figure B.10: Analytical and Multi Agent (MA) Non-Priority S-CAC: $P_{hb}$
Verification: Non Priority Based Multi Class S-CAC

Figures B.11 and B.12 verify the results of multi agent based simulations of S-CAC for multi class, non priority based CAC with that of the results of non agent based S-CAC, for new call blocking probability and handoff call blocking probability respectively.

![Figure B.11: Non Agent and Multi Agent Non-Priority S-CAC: \( P_{nb} \)](image-url)
Handoff Call Blocking Probability ($P_{bh}$) for voice and data for $K=0$

Figure B.12: Non Agent and Multi Agent Non-Priority S-CAC: ($P_{bh}$)
Appendix C

Research Publications
Performance Evaluation of Multi Agent Based Call Admission in Cellular Networks

Nupur Giri¹ and Shrikant Bodhe²
¹VESIT, Computer Engineering Department, Mumbai, India
Email: nupur.giri@gmail.com
²B VCOE, Electronics and Telecommunication Engineering Department, Pune, India
Email: s_k_bodhe@yahoo.com

Abstract—A Multi-agent System can be defined as a set of agents that interact with each other and with the environment to solve a particular problem in a coordinated (behaviorally coherent) manner. Our paper models Call Admission Control in Cellular networks as Multi Agent resource provisioning problem to provide Service Level Agreement (SLA) in wireless cellular network. For which we modify SHUFFLE model to introduce Network Provider Cell Agent which implements three Call admission strategies namely Static Cut Off Priority, Dynamic Call Admission and Mobility Based Channel Reservations strategies for single traffic class using Multi Agent Environment using Java Agent Development Framework.

The Paper evaluates the performance of call level parameters of CAC strategies along with performance parameters in Non Agent and Multi Agent System.

Index Terms—Multi Agent, Call Admission Control, JADE

I. INTRODUCTION

Multi Agent System extends the intelligent agent paradigm to improve upon the conceptual model of designing problems through agent interaction. This interaction could be of the form of collaboration, cooperation or even negotiation through Agent Communication Languages [11].

Service Level Agreement (SLA) is a pact between service providers (SP) and network providers (NP) or between service providers and customers specifying service guarantee in terms of service parameters, acceptable / unacceptable service levels, and action to be taken in special cases and penalties incurred by service provider in case the guarantee is not met. Call admission control (CAC) [10, 7, 6] is a fundamental mechanism used for quality of service (QoS) provisioning in a network by limiting the number of call connections into the networks in order guarantee Connection level QoS parameters such as new call blocking probability and handoff blocking probability, packet level QoS parameters such as delay, jitter, packet loss for various classes of traffic and mobility QoS parameters such as velocity, distance and direction of the movement of mobile terminal.

Although various call admission control schemes [3,4, 6] have been researched thoroughly. The most direct previous work is that of IST Project SHUFFLE [2]. It presents agent based layered architecture for Network management. It divides agents according to their responsibility assigned in Negotiation Plane and Resource Plane to meet the SLA. The work offers hypothesis that agents can be used for SLA control without offering implementation or performance analysis.

Our work is motivated by this SHUFFLE project, and intends in refining the model/architecture by introducing Network Provider Cell Agent (NPCA) as shown in fig. 1. By doing so two Multi agent System based Service Architectures: Centralized Network Provider Resource Agent (NPRA-based) and Distributed (NPCA-based) are proposed in our paper [8].

In this paper we present the performance of the CAC schemes implemented by these architectures and the analysis and evaluation of the same.

Section 2 details of the three CAC schemes: Static Cut Off Priority [5], Dynamic Cut Off Priority using D-CAC [7,10] and Mobility Based Channel Reservation based CAC [9]. Section 3 presents the MAS based model in JADE along with the FIPA performatives used for interaction amongst agents. Section 4 presents the simulations and results presenting connection level parameters along with comparative time of simulation required and message overhead involved in using Non agent and Multi agent environment along with analyzing these results.

II. CALL ADMISSION CONTROL

The two architectures can use following schemes for call admission.
A. Static Cutoff priority (S-CAC)

The Static Cutoff priority scheme [5] admits the calls in the cell if number of on-going calls in a cell is less than the threshold when new call arrives else the new call will be blocked. The handoff call is rejected only when all channels are used up.

If \( \lambda \) and \( \lambda_s \) is the arrival rate of new calls and handoff call respectively and \( 1/\mu \) and \( 1/\mu_s \) are average channel holding time for new calls and handoff calls, also if \( m \) is the chosen cutoff threshold for \( j \) busy channels then the new call will be blocked. The handoff call is rejected only when all \( j \) channels are used up.

If, \( \lambda \) is the rate of new calls and handoff calls then the following stationary distribution for the approximate the chosen cutoff threshold for \( j \) busy channels then the new call respectively and \( 1/n \) and \( 1/u \) are average channel holding time for new calls and handoff calls, also if \( m \) is the chosen cutoff threshold for \( j \) busy channels then the new call will be blocked. The handoff call is rejected only when all \( j \) channels are used up.

\[
\rho_j = \frac{\sum_{i} (\rho + \rho_s) \rho_s^i}{\sum_{i} (\rho + \rho_s) \rho_s^i + \sum_{i} (\rho + \rho_s) \rho_s^i}
\]

\[
\rho_j = \frac{\sum_{i} (\rho + \rho_s) \rho_s^i}{\sum_{i} (\rho + \rho_s) \rho_s^i + \sum_{i} (\rho + \rho_s) \rho_s^i + (\rho + \rho_s) \rho_s^i}
\]

Where \( \rho = \lambda/\mu \) and \( \rho_s = \lambda_s/\mu_s \)

B. Dynamic Cutoff priority (D-CAC)

The Dynamic Cutoff Priority scheme [7, 10] requires the threshold of call admission of a new call to be calculated based on the "Ongoing Calls" rather than Fixed threshold.

We assume Manhattan, micro-cell network model. Also the mobile terminal can move in only 4 direction (top, right, bottom, left). The channel allocation scheme is fixed. Let a new call admission request be made in addition to the state of its adjacent cells \( T \) units of time after the call admission. Let \( N \) denote the number of calls which a cell can support. If \( n \) denotes ongoing calls in the current cell \( C \), then \( t, r, b \) and \( l \) are the number of ongoing calls in neighboring cells (top, right, bottom and left), respectively. Let a new call admission request is made in addition to the state of its adjacent cells \( T \) units of time after the call admission. To calculate this threshold following conditions should be checked:

1) whether by admitting a new call at \( t \), the current handoff probability of cell \( C \) is less than the Threshold of call admission.

2) and whether the handoff probability of cell \( C \) affected by handoffs from cell \( C \) or the cell to the top, right, bottom and left of \( C \) is less than the Threshold of call admission.

\[
n_c = \frac{\left[ d(1-p_s) + 2N \right] - \frac{2 \rho_s \rho_s^2}{4}}{2 \rho_s}
\]

C. Mobility Based Channel Reservation (MBCR-CAC)

These schemes [9] require that elapsed time of a call in a cell and the velocity class (high mobility/ low mobility user) of calls can characterize the extent influence a cell has on its neighboring cell. The more influence a call exerts on its neighboring cells ((top, right, bottom and left), the more likely a channel should be reserved in the neighboring cells to maintain the QoS requirement of this call. If cell dwell times for both classes of users have negative exponential distributions. Then

\[
f_s(t) = \mu_s e^{-\mu_s t}, \quad f_h(t) = \mu_h e^{-\mu_h t}
\]

where \( 1/\mu_h \) and \( 1/\mu_s \) are the average cell dwell times for high-speed users and low-speed users, respectively.

Let \( \alpha_i,j \) (\( j \in N_i \)) be the directional factor,

\[
R_{i,j} = BI_{i,j}
\]

Hence at time \( T \), cell \( j \) needs to reserve channels for possible handoff calls from its neighboring cells. This scheme requires that neighboring cells (top, right, bottom and left) exchange information with each other: cell \( i \) should report \( R_{i,j} \) to all its neighbors. It is assumed that users are moving in a random movement pattern. Now from above equations

\[
R_{j} = \frac{B}{4} \sum_{k \in S_j} (1 - e^{-\rho_s(t-s)}) + \sum_{k \in S_j} (1 - e^{-\rho_s(t-s)})
\]

The Integral mobility based CAC calculates the call admission probability as:

\[
P_{new} = \begin{cases} 1 - B_{used} & C - R_j \leq B_{new} \\ 0 & C - R_j > B_{new} \end{cases}
\]

Where \( B_{used} \) and \( B_{new} \) are the number of used channels and the number of channels required by the incoming new call, respectively. Note that \( R_j \) may not be an integer. In this scheme, \( R_j \) is rounded off to the nearest integer \( R_j \) and use \( R_j \) as the final target number of reserved channels.

In this scheme, because of rounding of reservation request some information carried by the fractional part.
may be lost during the rounding; The Fractional Mobility scheme is modification of the above scheme which eliminates this problem. Thus in this paper we compare performances of Centralized (C) and Distributed (D) architecture based S-DAC, D-CAC and Integral MBCR-CAC (IMBCR-CAC) and Fractional MBCR-CAC (FMBCR-CAC) schemes using Non agent (NA) and Multi Agent (MA) models. Fig. 2 lists these 10 schemes.

III. MULTI AGENT SYSTEM IN JADE

The implementation of Non Agent CAC schemes are in Matlab 7 and Multi Agent-CAC is in Open Source JADE 3.1. JADE is FIPA complaint and runs on variety of operating systems including Windows and Linux. JADE offers support for Multi Agent interaction coordination competition through ACL messages using FIPA Performatives.

The Static scheme requires NPRA-NPCA (fig. 3) interaction for each incoming call where as Dynamic scheme calls for cluster of NPCA-NPCA interaction catering to real time traffic.

We define one NPRA agent in main container, NPCA agents can be invoked in different containers. JADE offers 12 message performatives out of which five have been implemented as shown in Table 1.

The process of starting NPCA agent consists of "registration" with corresponding NPRA. One NPRA agent can at most cater to 25 NPCAs. The agents implement the "Cyclic Behavior" of JADE to simulate the discrete event generation of call admission. The Controller agent is used to maintain synchronization amongst NPRA and NPCAs.

Fig. 4 NPCA-NPCA (Distributed Architecture) interaction amongst the agents. The NPCA of cell 'N' (on which the call arrives) 'Proposes' to perform call admission for cell 'N' by interacting with all its neighbors (NPCAs), thus performing MA-CAC interactions. It accepts an 'Inform' from each neighbouring NPCAs as a response of this 'Propose'. These responses contain the information about ongoing calls/mobility pattern in each of these NPCAs. The NPCA now calculates the threshold based on these responses and 'Accept/Reject' an incoming new call. In case of handoff calls the call is admitted if the channel is available.

JADE supports the Sniffer Agent to sniff the ACL messages exchanged between agents.

<table>
<thead>
<tr>
<th>TABLE 1</th>
<th>PERFORMATIVES USED FOR INTERACTION FOR CAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performatives</td>
<td>Meanings</td>
</tr>
<tr>
<td>Propose</td>
<td>Issued by the NPRA to propose performing Call admission control to neighboring cells</td>
</tr>
<tr>
<td>Inform</td>
<td>Issued as response to 'Propose'. The message carries 'Ongoing Calls', 'percentile of high/low velocity traffic' and also the 'class of traffic' information</td>
</tr>
<tr>
<td>Request</td>
<td>Requests NPRA to perform Congestion Control.</td>
</tr>
<tr>
<td>Accept</td>
<td>Confirms acceptance of call (for centralized)</td>
</tr>
<tr>
<td>Reject</td>
<td>Rejects the call(for centralized)</td>
</tr>
</tbody>
</table>

© 2009 ACADEMY PUBLISHER
The system is simulated for single (voice) traffic class. The calls are admitted in NPRA of 25 NPCAs. Manhattan model of 9 NPCA cluster having cell area of 4 square kilometer is considered. Each cell is given 30 channels. The new call arrival rate \((\lambda)\) which forms a Poisson process is taken as 20. The traffic load is varied from 10 to 100 Erlang. Handoffs are performed automatically based on RSS value or parameterized handoff threshold. Emulation of forced call termination on insufficient signal strength is implemented. The fig. 5 verifies the blocking probabilities of Centralized S-CAC schemes which are plotted for Non Agent as well as Multi Agent Environment, for threshold \(m=21\).

In fig. 6 comparison of call level parameters of Centralised S-CAC and D-CAC are plotted. We see that the new call blocking probability \((P_{nb})\) of D-CAC is less in case of less load but is high as compared to S-CAC \(m=25\) for higher traffic. It remains in between S-CAC \(P_{nb}\) \(m=27\) to S-CAC \(P_{nb}\) \(m=21\). The handoff call blocking probability \((P_{hb})\) is very less as compared to the that of S-CAC. This is because unlike in S-CAC the cutoff threshold changes dynamically in D-CAC and as the traffic load increases, more call are available in the neighboring cells so less calls are admitted in current cell threshold \(P_{hb}\) and thus implicitly increasing the channels for handoff calls thus reducing the handoff call dropping probability.

In D-CAC the user velocity and thus the dwell time of a call is constant. Mobility based schemes reserve channels for handoff calls implicitly defining threshold for new calls. The probabilities vary depending on the real time traffic as the thresholds depend on influence parameters. The call level parameters of Multi Agent based and Non Agent based schemes for Distributed architectures D-D-CAC and D-MBCRs are compared in fig. 7 and fig. 8. We see that the new call blocking probability of D-CAC is higher as compared to in MBCR schemes. Whereas its handoff blocking probability is in between that of the two mobility based schemes. The new call blocking of both mobility based schemes are almost same where as there is significant reduction in case of Fractional scheme. This is because of the rounding effect as explained in earlier section. MBCR-CACs fair well as compared to D-CAC.

Although message overhead in table I is higher in Distributed architecture as compared to Centralized architecture but again it is less by about 20 % in D-CAC scheme as compared to S-CAC scheme for Distributed architecture.
Tables III - IV give us an estimate of the reactiveness of MA-Based CAC over non agent based CACs for Centralized as well as Distributed SLA architectures. We see that by using Agent based schemes there is a significant reduction in simulation time, approximately of 22-23% in case of Centralised D-CAC, 11-12% in case of Distributed MBCR-CACs also as shown in tables V and IV the Distributed architecture scheme (D-MA-D-CAC) using Multi Agent schemes is faster as compared to the Centralized Non Agent based (C_NA-D-CAC) as well as Centralized Multi Agent based schemes (C_MA-D-CAC).

The reason for difference between reduction in D-CAC and MBCR is due to the time requirement for probabilistic estimation of Cutoff threshold in case of D-CAC. The above schemes are simulated for single traffic class. The role of Multi Agent system needs to be exploited for congestion control for multi class traffic. Also various interaction models would bring better reactivity and balance of resources in the cellular system.

References

Analytical Model for Multi Agent - Call Admission in Wireless Networks

Nupur Giri and Shrikant Bodhe
Computer Engineering Department, VESIT, Sindhi Society, Chembur, Mumbai, India.
Electronics and Telecommunication Department, BVPCOE, Pune, Mumbai, India.
nupur.giri@gmail.com, s_k_bodhe@yahoo.co.in

Abstract

Resource allocation and its planning are major issues in wireless networks. Distributed Call Admission Control requires calls to be admitted by considering local as well as global view of the network so as to reduce overall congestion. The paper proposes MA-CAC (Multi Agent-CAC) scheme which models a D-CAC problem using Multi Agent System (MAS) where each cell comprises of agents. These agents interact, communicate, coordinate or negotiate with each other to determine the optimal utility function based on maximum tolerable handoff dropping probability for a cell. It also presents the analytical model for calculating dependency of handoff dropping probability of a cell on the number of calls in the adjoining cells.

1. Introduction

The proliferation of mobile devices and use of mobile network for various integrated services such as voice, data and multimedia by the mobile user, places heavy demands of QoS requirements over wireless infrastructure. Provision of resources need to be done optimally, in real time as well as keeping future demands in purview.

Call admission control (CAC) is a fundamental mechanism used for QoS provisioning in a network. CAC limits the number of call connections into the networks in order to reduce the network congestion. Connection level QoS parameters such as new call blocking probability and handoff blocking probability, packet level QoS parameters such as delay, jitter, packet loss for various classes of traffic and mobility QoS parameters such as velocity, distance and direction of the movement of mobile terminal, all need to be considered together to come up with a complete solution for CAC. Thus providing QoS guarantee through CAC is increasingly becoming difficult.

Agents are software components which find their use in resource management due to their autonomous, social, reactive, proactive, self learning properties. A Multi-agent System (MAS) can be defined as a set of agents that interact with each other and with the environment to solve a particular problem in a coordinated (behaviorally coherent) manner. A multi agent System involves interaction, communication with system resources, collaboration, co-operation through Agent communication Languages (ACLs) [1].

The paper attempts to model CAC as Distributed Multi Agent resource provisioning problem, where agents of each cell interacts with other neighboring cell agents to get information of their number of ongoing call status. This interaction helps the cell agent derive the impact a neighboring cell’s calls has on its own call admission.

Section 2 reviews the related work in the field of CAC. Section 3 presents the MA-CAC strategy. Section 4 presents the analytical model for MA-CAC. Finally section 5 presents the results of this analytical model.

2. Related Work

Channel allocation schemes [2],[4] and Call admission control schemes such as guarded channel (GC), cutoff priority, fractional guard channel, rigid division, Queuing Priority (QP) [4], [5] are single service schemes and are local strategies. These algorithms reduce new call blocking probability and provide solution for service providers for increasing their subscriber base. But they do not address subscriber’s point of view for reducing handoff dropping probability (probability that the call is dropped after handoff). Thus guarantee of resources on the availability of wireless resources during their connection, in addition to a low new-call blocking probability needs to be evaluated.
In [3], a system that supports multiple traffic types using cost based reserve channels and dual threshold bandwidth reservation scheme is presented, which is also local admission control scheme. CAC scheme in [2] balances the blocking probability of micro-cell by redirecting the overflow traffic to macro cell if it exceeds a certain threshold for two service classes.

Distributed schemes [7] and [10] model overload probability using ongoing neighboring calls but do not consider multi traffic class or mobility parameters. Mobility parameters based schemes such as MBCR [8] and shadow cluster [9] are advanced CAC schemes. MBCR determines influence curve, which is the impact an ongoing call exerts on the adjacent cells, according to which the channel reservation can be adjusted. [9] suggests determining a set of neighboring cells called shadow cluster by using probabilistic information on the future position of mobile terminals with active calls, and then predicting resource demands. [12] uses agents to dynamically adapt to changes in the network load in order to maintain a target call dropping probability.

Our MA-CAC strategy on the other hand aims to guarantee user satisfaction level (utility function) by maintaining the maximum threshold of handoff dropping probability, and in doing so also calculates the impact of the number of ongoing calls in the neighboring cells on the current cell’s QoS parameters.

3. Multi Agent- Call admission Control

3.1 Problem Formalization

In terms of economics, utility functions describe satisfaction level with the perceived Quality of service, the higher the utility, the more satisfied the customer. To guarantee a certain level of performance for a particular cell we define a utility function \( U_s \) for it.

We assume that the utility function \( U_s \) of the cell is differentiable and monotonically decreasing concave function of the QoS parameter \( P_{hd} \), where \( P_{hd} \) is the handoff dropping probability of a cell. Therefore \( U_s = F(P_{hd}) \)

where \( F(P_{hd}) \) has following properties.

- \( F(P_{hd}) \geq 0 \)
- \( F'(P_{hd}) < 0 \)
- \( F''(P_{hd}) < 0 \)

\( U_s \) is maximum at \( P_{hd} = 0 \), which means that if the handoff dropping probability is zero for a cell, then user using the cell has the highest level of satisfaction. \( U_s(P_{hd} = 0) = (U_s)_{\text{max}} \)

Also there exists a \( P_{hd} = \text{max} \) called \( (P_{hd})_{\text{max}} \) such that \( U_s(P_{hd} = \text{max}) = 0 \)

Now there exists a \( (P_{hd})_{\text{tol}} \) such that

\[ 0 \leq (P_{hd})_{\text{tol}} \leq (P_{hd})_{\text{max}} \]

which can be tolerated so that the QoS and utility \( (U_s)_{\text{opt}} \) is considered acceptable. Based on the above definitions and assumptions we need to find \( (P_{hd})_{\text{tol}} \).

We already know that as number of calls in a particular cell increases the \( P_{hd} \) will also increase.

Thus now we need to find \( (n_c)_{\text{tol}} \) where \( n_c \) is the number of ongoing calls in any cell \( C \), which maintains the handoff dropping probability within the tolerable limit and maintains utility function as optimal. We further assume that number of on going calls in the neighborhood of cell \( C \) affect the maximum number of on going call in the cell \( C \), as few of those calls could be handed off to the cell with some probability also few on going calls from cell \( C \) might be handed off to neighboring cells. Thus the problem now is to find dependency of \( (n_c)_{\text{tol}} \) to the neighboring cell such that \( P_{hd} \) is acceptable i.e. \( (P_{hd})_{\text{tol}} \).

3.2. MA-CAC Cell-Agent Model

The Wireless cell network can be modeled as MAS having 4-cell neighbor (top, right, bottom, left) cluster shown in figure 1, to solve the above problem. The MA-CAC strategy is a Multi agent strategy which uses probability to estimate the impact of neighboring cells on the current cell and its handoff dropping probability and interacts with the neighboring/adjacent cell’s agent.

Figure 1. MA-CAC Cell-Agent Model
Each cell comprises of a proactive and a reactive agent. The proactive agent first probabilistically estimates the maximum tolerable load of the current cell by using equal probability of handover to the neighboring cell and derives analytical values.

When new/handoff call is to be admitted, the reactive agents of cells (neighboring) interact with each other to exchange their (cells) ongoing call status. If the actual number of calls do not cross analytical threshold then the call is admitted or else it is rejected.

4. MA-CAC Analytical model

Here we present the analytical model for MA-CAC strategy for Manhattan, micro-cell network, we assume that the mobile terminal can move in only 4 directions (top, right, bottom, left). The channel allocation scheme is fixed.

If \( n \) denotes calls in the current cell \( C \), then \( t, r, b \) and \( l \) are the number of calls in neighboring cells (top, right, bottom, left) respectively. Let us assume \( p_s \) is the probability that the call remains in the same cell. Let \( p_s / 4 \) be the probability that call will be handed off to neighboring cell during time \( T \). Also assume that the call can be handed over only once \( (m=1) \). \( \lambda \) denotes the new call arrival rate to any cell, and \( \mu \) denotes the call departure rate. Let a new call admission request be made in addition to new calls admitted to cell \( C \) during the time \( T \) units of time after the call admission is made.

Let \( N \) denote the number of calls which a cell can support.

Let \( (P_{hd})_{\text{tol}} \) be the maximum tolerable handoff dropping probability of that call. Then at the time of admitting a call following conditions should be checked

1) whether by admitting a new call at \( t_0+T \), the handoff probability of cell \( C_n \), due to incoming handoff calls from \( C_t, C_r, C_b \) and \( C_l \) and also outgoing handoffs from \( C_e \) is less than the \( (P_{hd})_{\text{tol}} \).

2) and whether the handoff probability of cell \( C_t(C_r/C_b/C_l) \) affected by handoffs from cell \( C_n \) or the cell to the top, right, bottom and left of \( C_t \) to cell \( C_t \) and including handoffs from cell \( C_e \) to any other cell, in addition to new calls admitted to cell \( C_t \), during \( T \), must be less than \( (P_{hd})_{\text{tol}} \).

Using classic D-CAC[7] 1-D model where number of calls in cell \( C_t \) at \( t_0+T \) can be expressed as Gaussian distribution we derive values for Manhattan model having the handover probability in each cell as \( p_s / 4 \) and sum of the neighboring calls as \( SUM = (t+r+b+l) \).

\[
P_{n_{i0+T}}(k) = G \left( \frac{n_p + SUM p_m / 4}{\sqrt{(np_s(1-p_s)+SUM p_m(1-p_m/4))/4}} \right)
\]

For a given \( (P_{hd})_{\text{tol}} \) there exists a value \( a \) such that \( (P_{hd})_{\text{tol}} = Q(a) \)

The relation now becomes

\[
N - np_s - SUM p_m/4 = a \sqrt{(np_s(1-p_s) + SUM p_m(1-p_m/4))/4}
\]

where \( Q(.) \) is the integral over the tail of a Gaussian distribution. Solving for admission threshold \( n_t \) which satisfies the first condition by putting eq. 5 in quadratic form, we get

\[
n_t = \frac{\alpha^2 p_s(1-p_s) + 2Np_s - SUM p_m}{2} \pm \sqrt{4p_s^2 + 2N^2 + SUM^2 p_m^2 / 16 - \alpha^2 SUM p_m^2 / 4 + \alpha^2 SUM p_m^2 / 16}
\]

Solving further we get eq. (7) as

\[
n_t = \frac{\alpha^2 p_s(1-p_s) + 2N - SUM p_m/2}{2} \pm \sqrt{4p_s^2 + 2N^2 + SUM^2 p_m^2 / 16 - \alpha^2 SUM p_m^2 / 4 + \alpha^2 SUM p_m^2 / 16}
\]

Eq. 7 gives number of calls in the current cell which satisfy the first condition.

For the second condition to be satisfied let \( E(n) \) be the average calls per cell in the neighboring cells of \( C \) (\( C_t/C_r/C_b/C_l) \).

Then the handoff dropping probability for the cell to right of \( C \) at \( t_0+T \) can be presented as Gaussian distribution such that

\[
P_{r_{T0+T}}(k) = G \left( \frac{rp_s + (E(n)+n)p_m/4 + \lambda \Gamma}{\sqrt{rp_s(1-p_s) + (E(n)+n)p_m(1-p_m/4)/4 + \lambda \Gamma}} \right)
\]

Similarly, the maximum number of calls or the admission threshold that can be admitted to \( C_t \) such that the second admission condition for cell \( C_t(C_r/C_b/C_l) \) is satisfied, which is calculated by

\[
N - rp_s - SUM p_m/4 \Gamma = a \sqrt{(rp_s(1-p_s) + (E(n)+n)p_m(1-p_m/4))/4 + \lambda \Gamma}
\]

Putting eq. 9 in the quadratic form for \( n_t \), where \( n_t \) the admission threshold that can be admitted to \( C_t \) such
that the second admission condition for cell \( C_r \) is satisfied.

\[
\begin{align*}
\alpha & = \frac{\sqrt{\left(4-p_m\right)^2 + 64N + 16p_m(\pi^2 - N) - 64p_m^2}}{2p_m} \\
\end{align*}
\]

Similarly admission thresholds \( n_c, n_l \) and \( n_b \) for admitting calls in \( C_a \) for satisfying second condition for \( C_c, C_b, C_s \) is also calculated. The final proactive estimate for admission threshold for a new call which \( \text{admitting calls in } C \) for satisfying second condition \( r = 10 \), based on:

\[
\begin{align*}
\alpha & = \frac{\sqrt{\left(4-p_m\right)^2 + 64N + 16p_m(\pi^2 - N) - 64p_m^2}}{2p_m} \\
\text{cwl} & = n \text{e}^{V} \\
\text{nl} & = n \text{e}^{V} \\
\text{nb} & = n \text{e}^{V} \\
\end{align*}
\]

The proactive agent uses equation (6), (10) and (11) for analytically deriving the \( n_c \).

5. Analytical results

For analytically results of the MA-CAC strategy, we have chosen an average handoff as time of 100s, and an estimation \( T \) period 20 s, \( \lambda = 1/500 \), \( \mu = 0.5 \), \( 1/h = 1/100 \) and average call per neighboring cell \( E(n) = 10 \), based on :

\[
\begin{align*}
p_s &= e^{-\mu h} \\
p_m &= 1 - e^{-\mu h} \\
\end{align*}
\]

\( p_m \) and \( p_s \) are calculated as 0.1813 and respectively 0.7866, \( a \) equals 2.1 for \( N=20 \). The number of calls for top, right, bottom and left cells range from 1 to 20.

Let \( X \) be an 4-D array holding value \( n_c \) for calls being generated in each neighboring cell as well in current cell and satisfying equation 11. The resultant \( X \) could not be plotted because of its complex 4-D nature.

The resultant \( X \) for \( N=20 \) in Table 1, shows that the maximum number of calls allowed in cell \( C \) moved from 20 to approximately 14 as the adjacent number of calls moved from 1 to 18(max), no, of calls dropped to 7 and then 0, once the adjacent calls increase further.

For \( N=30 \), \( N=40 \) and \( N=50 \) the analytical values of maximum numbers of calls allowed were found.

For \( N=40 \) and \( a=3.092 \), the drop in number of allowed calls in cell \( C \) occurs at either of \( r, t, b \) or \( l \) reaching 37.

For \( N=50 \) and \( a=3.48 \), the drop in number of allowed calls in cell \( C \) occurs at either of \( r, t, b \) or \( l \) reaching 47 as shown in Table 2.

As we see from the analytical results, as the cell capacity increases the admission surface moves away from the origin and beyond a cutoff level of number of calls in the adjacent cells, the number of calls in the current cell \( C \) drops dramatically. Also as the probability to handoff is taken equal in all 4 directions, as any of the cell’s number of calls reach cutoff threshold, the number of calls which could be admitted in the current cell \( C \) decreases. \( a \) increases as cell capacity increases.

The tables represent following property of resultant \( X \).

Let set \( S \) represent array \( X \).

\( S = \{C_1,C_2,...,C_l\} \quad 1 \leq l \leq 4 \)

Let \( A \subseteq S \) such that \( |A| = 3 \)

Let \( B \subseteq S \) such that \( B = A \)

\( Z_i = \{K_1, K_2, ..., K_j\} \quad 1 \leq j \leq 20 \)

Let \( Y \subseteq B \) where \( 1 \leq Y \leq 20 \)

Then \( n_c = F(j,Y) \)

\[
\begin{align*}
t & = F(j,Y) \\
\end{align*}
\]

\[
\begin{align*}
\text{TABLE 1.} \\
\text{Analytical results for } N=20: \quad n_c = F(j,Y) \\
\end{align*}
\]

\[
\begin{align*}
\begin{array}{|c|c|c|}
\hline
\text{i} & \text{n} & \text{F} & \text{Y} \\
\hline
1 & 20.000,19.921,19.843,19.764,18.751,18.674,7.8070 & 1 & 1 \\
5 & 19.062,18.984,18.906,18.829,17.826,17.749,7.8070 & 1 & 5 \\
6 & 18.829,18.751,18.674,18.596,17.596,17.519,7.8070 & 1 & 6 \\
7 & 18.596,18.519,18.442,18.364,17.366,17.290,7.8070 & 1 & 7 \\
8 & 18.364,18.287,18.210,18.133,17.137,17.061,7.8070 & 1 & 8 \\
9 & 18.133,18.056,17.979,17.902,17.902,16.909,16.832,7.8070 & 1 & 9 \\
10 & 17.902,17.826,17.749,17.672,16.680,16.604,2.0707 & 1 & 10 \\
11 & 17.672,17.596,17.519,17.443,16.453,16.377,7.8070 & 1 & 11 \\
12 & 17.443,17.366,17.290,17.213,16.226,16.150,7.8070 & 1 & 12 \\
13 & 17.213,17.137,17.061,16.985,15.999,15.923,7.8070 & 1 & 13 \\
17 & 16.301,16.226,16.150,16.074,15.096,15.021,7.8070 & 1 & 17 \\
18 & 16.074,15.999,15.923,15.848,14.871,14.796,7.8070 & 1 & 18 \\
19 & 7.807,7.807,7.807,7.807,7.807,7.807,7.807 & 1 & 19 \\
20 & 0,0,0,0,0,0,0,0 & 1 & 20 \\
\hline
\end{array}
\end{align*}
\]
TABLE 2.
Analytical results for N=50: $n_c^{\text{tot}} = F(j, Y)$

<table>
<thead>
<tr>
<th>$j$</th>
<th>$n_c^{\text{tot}} = F(j, Y)$ for $1 \leq Y \leq 50$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$49.998, 49.918, 49.839, \ldots, 46.44, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>1</td>
<td>$49.759, 49.680, 49.600, \ldots, 46.21, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>2</td>
<td>$49.521, 49.442, 49.363, \ldots, 45.97, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>3</td>
<td>$46.682, 46.604, 46.526, \ldots, 43.18, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>4</td>
<td>$46.447, 46.369, 46.291, \ldots, 42.95, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>5</td>
<td>$44.809, 44.732, 44.654, \ldots, 41.33, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>6</td>
<td>$44.576, 44.499, 44.421, \ldots, 41.10, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>7</td>
<td>$42.027, 41.950, 41.873, \ldots, 38.58, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>8</td>
<td>$41.796, 41.719, 41.643, \ldots, 38.36, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>9</td>
<td>$39.501, 39.425, 39.349, \ldots, 36.09, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>10</td>
<td>$35.548, 35.548, 35.548, \ldots, 35.548, 35.548, 35.548, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>11</td>
<td>$22.881, 22.881, 22.881, \ldots, 22.881, 22.881, 22.881, 10.276, 0$</td>
</tr>
<tr>
<td>12</td>
<td>$10.276, 10.276, 10.276, \ldots, 10.276, 10.276, 10.276, 10.276, 0$</td>
</tr>
<tr>
<td>13</td>
<td>$0.0, 0.0, 0.0, \ldots, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0$</td>
</tr>
</tbody>
</table>

6. Future work

The future work includes building simulation model and taking velocity, direction of mobile terminal into consideration while assigning the probability of handoff instead of assuming equal probability as $P_m/4$. Thus assigning a percentile weight called $w_{\alpha, \beta, \gamma, \phi}$ to the probability that the call can be handed off in that direction. Thus

$$\alpha + \beta + \gamma + \phi = 1$$

(13)

where $\alpha, \beta, \gamma, \phi$ are percentile weight handoff probability multipliers for top, right, bottom and left cells and are calculated depending on the position and the velocity of the mobile device with respect to the cell boundaries. Thus $n_c^{\text{tot}}$ varies according to mobility parameters.

Also once the utility function of each cell is found, load balancing of utility function for the entire cluster using agents negotiation can be achieved.

7. References


[10] Shengming Jiang, Danny H. K. Tsang and Bo li: An Enhanced distributed Call Admission for wireless system


ABSTRACT
Multi Agent System extends the intelligent agent paradigm to improve upon the conceptual model of designing problems through agent interaction. This interaction could be of the form of collaboration, cooperation or even negotiation through Agent Communication Languages. Our paper models Call Admission Control in Cellular networks as Multi Agent resource provisioning problem, where agents interact with each other to find the optimal threshold for call admission using Distributed Service Architectures. It evaluates connection level performance characteristics of Dynamic and Mobility based call admission control schemes using agents. The Multi Agent architecture is designed and simulated using open source Java Agent DEvelopment Framework (JADE). Finally the paper compares performance of the CAC schemes in Non Agent based and Multi agent based environments.

Keywords
Call Admission control, JADE, Mobile Computing Multi Agent System, Service Level Agreement

1. INTRODUCTION
Agents are software components which find their use in resource management due to their autonomous, social, reactive, proactive, self learning properties. Multi Agent Systems (MAS) defines the behavior of such autonomous agents that work together to solve a given problems that are beyond the individual capabilities or knowledge of each problem solver agent [12] [14].

The true MAS have following characteristics:

• Each agent has incomplete information/capabilities for solving the problem.
• Absence of global system control.
• Decentralized data
• Asynchronous computation

Call admission control (CAC) [6], [7], is a fundamental mechanism used for quality of service provisioning in a network. CAC limits the number of call connections into the networks in order to reduce the network congestion. Connection level QoS parameters such as new call blocking probability and handoff blocking probability, packet level QoS parameters such as delay, jitter, packet loss for various classes of traffic and mobility QoS parameters such as velocity, distance and direction of the movement of mobile terminal, all need to be considered together to come up with a complete solution for CAC.

This paper models Dynamic CAC [9] [11] schemes and Mobility based Channel Reservation CAC schemes which are suitable for modeling real time traffic, using Multi Agent System. In these each cell agent is required to interact with other neighbouring cell agents to get information of number of ongoing calls, traffic pattern (high/low speed) etc. in the cell. This interaction helps the current cell agent to derive the impact/influence that a neighbouring cell’s calls has on its own call admission.

We have use MatLab 7 for simulating Non Agent based CAC, whereas Java Agent DEvelopment Framework (JADE) an open source platform is chosen for implementation of MAS [1].

Section 2 reviews the agent and multi agent based work in the field of CAC. Section 3 presents the extended SHUFFLE model for Distributed service architectures for SLA. Section 4 gives details of Multi Agent-CAC (MA-CAC) schemes. Section 5 discusses modeling the schemes in JADE. Section 6 gives the
simulation results and finally section 7 analyses these results presenting the comparison of three MA-CAC schemes.

2. REVIEW OF PREVIOUS WORK
Although various call admission control schemes have been researched thoroughly, here we review only agent based work in the domain of network management and resource allocation. In papers [2], [3] greater autonomy is given to the Base station resulted in a distributed resource allocation scheme for first generation mobile networks using intelligent agents that offered an efficient solution for resource allocation under moderate and heavy loads. The use of intelligent agents in Cheikhrouhou [4] brings Network Management awareness by splitting objectives to different layers to handle sub goals. The most direct previous work is that of IST Project SHUFFLE [5]. In this work of Bodanese he only offered the hypothesis that the agents could control SLAs, giving no details of implementation, or any results on SLA management. Gibney [8] proposed market based call routing based on self interested networks whereas work done by Chantaraskul [3] is on case based reasoning for congestion control.

Our work is motivated by the SHUFFLE project, and intends in refining the model/architecture by introducing Network Provider Cell Agent (NPCA). Giri [10] presents this extended model and proposes two Multi agent System based service architectures, Centralized (NPRA-based) and Distributed (NPCA-based). It also compares Static or Dynamic Call Admission schemes for Agent and Non agent based architectures.

This paper further presents the comparison new call blocking probability and handoff blocking probability of 3 MA-CAC schemes: Dynamic Cut Off priority which catering to real time traffic, Mobility Based Integral and Fractional which caters to mobile users.

3. DISTRIBUTED MULTI AGENT SERVICE ARCHITECTURE
The SHUFFLE Model divides the agents according to their responsibility assigned in Negotiation Plane and Resource Plane to meet the SLA. This is as shown in figure 1.

3.1 User Agent (UA)
UA interacts with the service provider negotiation agent and acts on behalf of the user QoS to maintain SLAs with all service providers.

3.2 Service Provider Agent and Network Provider Agent (SPA & NPA)
They act on behalf of the Service Provider (SP) or Network Provider (NP) to control the overall policies. They coordinate the activities between different provider agents.

3.3 Service Provider Negotiation Agent & Network Provider Negotiation Agent (SPNA & NPNA)
SPNA is responsible for negotiating SLAs with the UAs and NPNA. The NPNA acts on behalf of each NP to negotiate SLAs with SPs in order to manage the contract as actual traffic conditions vary.

3.4 The Network Provider Resource Agent (NPRA)
It acts on behalf of the NP to implement the policies of the NPA and to manage the radio resource of the NP.

3.5 The Network Provider Cell Agent (NPCA)
NPCA does local call admission and flow control in the cell. Along with peer NPCAs, it can participate in Dynamic call admission to meet SLA requirement given by NPRA.

The centralized architecture is controlled by NPRA where distributed architecture requires NPCA-NPCA participation using Dynamic Multi Agent - Call Admission Control schemes.

<table>
<thead>
<tr>
<th>UA : User Agent</th>
<th>NPA : Network Provider Agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPA : Service Provider Agent</td>
<td>NPNA : Network Provider Negotiation Agent</td>
</tr>
<tr>
<td>SPNA : Service Provider Negotiation Agent</td>
<td>NPPA : Network Provider resource Agent</td>
</tr>
<tr>
<td>NPCA : Network Provider Cell Agent</td>
<td></td>
</tr>
</tbody>
</table>
The Distributed architecture uses NPCA along with MA-CAC strategies (section 3) to admit calls according to ongoing calls in neighboring cells and thus considering the neighboring cell’s load. This NPCA-NPCA cooperation (neighboring cell) can be termed as true Multi Agent system for SLA. It requires NPCA to be proactive as well be reactive and also thus actively participate in CAC and handle exceptions. The decision making of call admission is more granular and it lies with the cell itself (NPCA) rather than NPRA. The role of NPRA is reduced to overall management and changes in basic policies for congestion control, etc. whereas actual call admission and flow control needs to be now handled by NPCA. The advantage this architecture is reduction in bottleneck at NPRA, which results in increasing scalability and faster decision making and less message overhead.

4. MULTI AGENT CALL ADMISSION CONTROL (MA-CAC)

Incoming call in a cell could be either a new call or a handoff call. Each cell requires, the neighboring cell to reserve channel to handle these calls to maintain the SLA as handoff calls need to be given higher priority over new call admission. In this section we discuss few schemes which try to accommodate these neighboring handoff calls and perform call admission control taking real time traffic into consideration. To explain the strategies we consider the Manhattan model of cells and a cluster of 5 cells with each cell having top (Co), right (C1), bottom (C2), and left (C3) cell as their neighboring cells.

4.1 Dynamic Cutoff priority based (D-CAC)

The Dynamic Cutoff Priority scheme [9] requires the threshold of call admission of a new call to be calculated based on the "Ongoing Calls" rather than Fixed threshold.

Let a new call admission request be made in addition to the state (j=0,1,...C). Thus this dynamically calculated threshold remains in the same cell. Let \( P_{j} \) be the probability and \( s \) is tunable. This \( n_{c}^{	ext{tol}} \) is substituted in equations as ‘\( m \)’ in (4) and (5) for D-CAC probabilities.

This dynamic threshold affects the connection level parameters, the new call blocking probability and handoff call blocking probability in the current cell. For capacity ‘\( C \)’ of a cell Let \( P_{i} \) denote the probability that the call that remains in the same cell. Let \( P_{c}/4 \) be the probability and \( s \) is tunable. This \( n_{c}^{	ext{tol}} \) is substituted in equations as ‘\( m \)’ in (4) and (5) for D-CAC probabilities.

If \( \lambda, \lambda_{h} \) is the arrival rate of new calls and handoff call respectively and \( 1/\mu_{b} \) and \( 1/\mu_{h} \) are average channel holding time for new calls and handoff calls. Then

\[
P_{c}^{\text{tol}} = \min(n_{c}, n_{1}, n_{r}, n_{b}, n_{l})
\]

where \( E(n) \) be the average calls per cell in the neighboring cells of \( C \). Let \( P_{j} \) denote the probability that there are \( j \) busy channels in steady state \( \lambda=0,1,...,C \). Thus this dynamically calculated threshold gives the New call Blocking Probability and Hand Off Call Blocking Probability as:

4.2 Mobility Based MA-CAC Schemes

These schemes in paper [12] propose that elapsed time of a call in a cell and the velocity class (high mobility / low mobility user) of calls can characterize the extent influence a cell has on its neighboring cell. The more influence a call exerts on its neighboring cell, the more likely a channel should be reserved in the neighboring cells to maintain the QoS requirement of this call.

In order to understand the concept of this scheme the following points must be noticed

The handoff probability is a function of the time elapsed after a call enters a cell;

After dwelling in a cell for the same length of time, a high-speed user is more likely to request a handoff than a low-speed user,
which implies that the handoff probability is also related to the speed class of a user. Because of cell handoffs, traffics among cells are no longer independent. An ongoing call in the current cell exerts some influence on the channel assignment in the neighboring cells.

The Integral mobility based CAC calculates the call admission probability as

\[ P_{\text{new}} = \begin{cases} 1 - B_{\text{used}} & \text{if } C - R_{j} \leq B_{\text{new}} \\ 0 & \text{if } C - R_{j} > B_{\text{new}} \end{cases} \]  

(6)

Where \( P_{\text{new}} \) is the admission probability for new calls, \( B_{\text{used}} \) and \( B_{\text{new}} \) are the number of used channels and the number of channels

<table>
<thead>
<tr>
<th>TABLE 1. PERFORMATIVES USED FOR INTERACTION FOR CAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performatives</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>Propose</td>
</tr>
<tr>
<td>Inform</td>
</tr>
<tr>
<td>Request</td>
</tr>
<tr>
<td>Accept</td>
</tr>
<tr>
<td>Reject</td>
</tr>
</tbody>
</table>

required by the incoming new call, respectively. Note that \( R_{j} \) may not be an integer. In this scheme, \( R_{j} \) is rounded off to the nearest integer \( R_{j}^{'} \), and use \( R_{j}^{'} \) as the final target number of reserved channels. In this scheme, because of rounding of reservation request, some information carried by the fractional part may be lost during the rounding; The Fractional Mobility scheme is modification of the above scheme which eliminates this problem.

5. MODELING MULTI AGENT SYSTEMS IN JADE

The implementation of Distributed architecture for all three MA-CAC is using Open Source JADE 3.1.

The reason for selection of JADE as compared to other [13] is that it is FIPA complaint and runs on variety of operating systems including Windows and Linux. JADE offers support for multi agent interaction coordination competition through ACL messages using FIPA Performatives

This matches our requirements of high number of agents and message exchange. A Distributed scheme calls for cluster of NPCA-NPCA interaction. The process of starting NPCA agent consists of “registration” with corresponding NPRA.

One NPRA agent can at most cater to 25 NPCAs. The agents implement the “Cyclic Behavior” of JADE to simulate the discrete event generation of call admission. The Controller agent is used to maintain synchronization among NPRA and NPCAs. We define one NPRA agent in main container. NPCA agents can be invoked in different containers.

JADE offers 12 message Performatives out of which five have been implemented as shown in Table 1. The interaction amongst these agents could be of the form of: cooperation (agents share a common goal); coordination (scheduling and synchronization) for mutual benefit and negotiation (coming to an agreement for mutual beneficial solution). Here we use coordination as the interaction model and ensure that the agents synchronize with each other and communicate the number of ongoing calls and mobility information. The NPCA of cell ‘N’ (on which the call arrives) ‘Proposes’ to perform call admission for cell ‘N’ by interacting with all its neighbors, thus performing MA-CAC interactions. It accepts an ‘Inform’ from each neighboring NPCAs as a response of this ‘Propose’. These responses contain the information about ongoing call/mobility pattern in each of these NPCAs. The NPCA now calculates the threshold based on these responses and ‘Accept/Reject’ an incoming new call. In case of handoff calls the call is admitted if the channel is available.

JADE supports the Sniffer Agent to sniff the ACL messages exchanged between agents. This interaction is shown in figure 2. The Congestion Control could be achieved either by reallocation of channels to cells (NPCAs), involving NRPA-NPRA negotiations or could be by local flow control (adjusting the new call blocking thresholds), which could be handled by NPCAs. As handoff calls are given priority over new calls, handoff calls could be queued.
6. SIMULATIONS AND RESULTS

The system is simulated for single (voice) traffic class. The calls are admitted in NPRA of 25 NPCAs. Manhattan model of 9 NPCA cluster having cell area of 4 square kilometer is considered. Each cell is given 30 channels. The new call arrival rate ($\lambda$) which forms a Poissons process is taken as 20. The traffic load is varied from 10 to 100 Erlang. Handoffs are performed automatically based on RSS value or parameterized handoff threshold. Simulation of forced call termination on insufficient signal strength is implemented. The system developed in the simulation model is a combination between discrete and continuous systems as some state variables change continuously with respect to time and some change at discrete points in time.

The MA-CAC schemes give preference to the handoff calls. The mobility based schemes reserve channels for handoff calls implicitly defining threshold for new calls. The probabilities vary depending on the real time traffic as the thresholds are calculated dynamically. The Figures 3 and 4 present the simulated results of new call blocking and Handoff call blocking probabilities for MA-CAC schemes. We see that the New Call blocking probability of D-CAC is higher as compared to in Mobility based schemes. Whereas its handoff blocking probability is between the two mobility based schemes (Integral and fractional). The New Call blocking of both mobility based schemes are almost same where as there is significant reduction in case of Fractional scheme. This is because of the rounding effect as explained in earlier section. Mobility based schemes fair well as compared to Dynamic Cut Off schemes in terms of call level parameters.

For Dynamic Cutoff Priority MA-CAC the user velocity and thus the dwell time of a call is constant. Mobility Based schemes are simulated for 20%, 30% and 40% (by changing the dwell time).

The results show that as percentage of high speed users increase, the new call blocking probability decreases and handoff blocking probability increases, as the a slow speed user dwells in the cell for a longer time thus occupying a channel for a longer time thus increasing the new call blocking probability where as a high speed user will definitely require a handover thus increasing the hand off blocking probability.

Figure 2. NPCA-NPCA Interaction amongst the cell agents of a Cluster

Figure 3. MA-CAC scheme comparison: New Call Blocking Probability

Figure 4. MA-CAC scheme comparison: Hand Off Call Blocking Probability

Figure 5. Mobility Based Integral CAC for varying mobile user call velocity
The two MA-CAC schemes (D-CAC and Integral Mobility Based) are implemented and compared using agent based (JADE) and non agent based environments in terms of call level parameters (in figure 6) as well as elapsed time (Table 2). This gives us an estimate of the reactivity of MA-Based CAC over Non Agent based CACs. We see that by using agent Based schemes there is a significant reduction in simulation time, approximately 41-42% in case of D-CAC and 11-12% in case of mobility Based. The reason for difference between reduction in D-CAC and mobility based being due to the time requirement for probabilistic estimation of Cut off Threshold in case of D-CAC.

Multi Agent systems definitely have lot of scope in resource management in cellular system as the MA-CAC schemes provide more reactivity and also Fractional Mobility Based MA-CAC offers better connection Call level parameters as over D-CAC or even its Integral counterpart.

The above results are for single traffic class (voice) and uniform mobility pattern of the user. The results very clearly point out the benefits of Multi Agent based over Non agent Based Systems.

The MA-CAC schemes are being expanded to accommodate multi class traffic (voice, data e-mail-Fax & video conferencing) along with different mobility patterns.

Also congestion control schemes need to be implemented to explore the full benefit of multi agent system for cellular networks for mobile computing for which Queuing Based models are being considered. Various interactions models, coordination, co-operation and negotiation can be explored to understand the advantage of one over other.

7. REFERENCES


Intelligent Agent as Middleware for Mobile Computing: Special Reference to Mobile Agents

Vikram S. Patil1, Nupur Giri2, S. K. Bodhe3

1 Assistant Professor, I.T. dept, UMIT SNDT Women’s University, Mumbai, India. Tel: 022-8682318, vsp_res@rediffmail.com
2 Assistant Professor, Computer Engg. dept, VESIT, Sindh Society, Chembur, Mumbai, India. Tel: 022-55768718, nupur.giri@gmail.com
3 Prof & HOD ETC, Shahu College of Engineering, Pune, India, Tel: 022-9890942022, skbodhe@hotmail.com

Abstract

Intelligent Agents (IA) paradigm can be used to develop middleware for mobile computing to support mobility. Mobile Agents (MA) is a special case of software agents, which requires different environmental support. This paper provides a review of the two main agent middleware standards, provided by FIPA and OMG (MASIF) and also discusses various platforms which have been developed using the specifications. The choice of platform, amongst many, depends on specific application domain and devices used for application deployment.

Keywords:

Mobile agent architecture, MA Designing, FIPA Abstract architecture, MASIF, Middleware’s, Mobile Agent platforms.

Introduction

Wireless communications and the Internet are converging towards an integrated scenario where both traditional and novel services should be ubiquitously accessible, independently of the mobility of users, terminals, resources and service components. Mobility-enabled service provisioning introduces several issues to address: from client/server location change at provision time, to wide heterogeneity of access terminals, and to unpredictable modifications in accessible resources. In this complex scenario, two main guidelines are recently emerging: First one is, the need for novel middleware solutions to support service development and deployment, second is, the necessity of full visibility of the context, intended as the logical set of accessible resources depending on client location, access terminal capabilities, and system/service management policies, to adapt service provisioning to specific runtime conditions. Middleware are facilitating communication and coordination of distributed components. Systems like CORBA, RMI are developed as middleware and they are having strong advantages. However both theses systems are having difficulties in managing code mobility, whereas Mobile Agents can migrate very easily over even heterogeneous networks.

Mobile agents can be classified based on the degree of complexity, known as granularity. Simple agents with little or no intelligence regarding their behaviour are known as reactive agents. Agents capable of knowing their environment and able to reason about their goals are cognitive or deliberative agents. These agents are more complex compared to earlier agents. [1] Cooperation, autonomy and learning are the three basic attributes of agents and from them three types of agents can be identified: collaborative agents, interface agents and collaborative learning agents. Smart agents emphasize on the three attributes equally. In this paper life cycle of agent, FIPA abstract architecture, concrete realization of agent architecture and design issues are discussed.

Agent life Cycle Models

All The different execution states of a mobile agent and the events that cause the movement from one state to another constitute the life cycle of Mobile agent. Different life cycle models are derived. First one, which is powerful and flexible, is known as Persistent process based agent life cycle model. It starts with a ‘start’ state, proceeds to a ‘running’ state where a persistent process is executed, and hopefully enters the ‘frozen’ state where the process is terminated. When a mobile agent is transported from one node to another, the process in the running state is check-pointed and the agent enters a ‘death’ state where the process is terminated. Next, its context is delivered to the destination node where the process is resumed and re-enters the ‘running’ state at the point it left off. Second model, which is known as task based life cycle model, has lesser flexibility. It starts in the ‘start’ state.

Then depending on a set of conditions it proceeds through a network of tasks. Each task having its own state. This is shown in figure 1. However, when the agent moves to a new node, the context of the currently executing task is lost. Before moving the agent must indicate the first task to be served when it re-materialises on the destination node.

MA structure consists of the computational model at the core. This has significant impact on the other models. It defines how we address other agents, hosts and resources, which is important to the security model. The type of life cycle model adopted is dependent on the facilities of the computational model. Both the security and life cycle models are structurally very close to the core. Security issues permeate every aspect of a mobile agent and therefore must be provided for at the most basic level. The
life cycle model defines the valid states for an agent. The outer layer contains the communication, navigation and agent models. The agent model defines the 'intelligent agent' aspects of a mobile agent such as learning and collaboration functions. The communication model is heavily dependent on the security model so that other agents or hosts do not corrupt agents. Finally the navigation model is also dependent on the security model as it hands itself over to the host to be transported to another node. [2]

Figure 1. Mobile Agent life cycle

Infrastructure Requirements

Agents must interact with their hosts to make use of their services and negotiate with other agents about services. Mobile agents must also be able to move about in a heterogeneous computer network. The mechanics that allow them to do so are known as the agent infrastructure and the entity providing support for agents on a particular host is an agent server. [2] This environment must have elements like, distributed computing platforms, messaging platforms, security services, directory services, intermittent connectivity technologies.

Agent Architecture

All the properties that have been described that an agent must or may have are embedded inside its architecture. Different agent architectures can be used to realize an agent.

Deliberative Architecture

A deliberative agent architecture possesses an explicitly represented, symbolic model, in which decisions are made via logical reasoning, based on pattern matching and symbolic manipulation. This model is also known as Beliefs, Desires and Intentions (BDI) models as is based on. This architecture is shown in figure 2. The basic idea of BDI approach is to describe the internal processing state of an agent by means of a set of mental categories, and to define a control architecture by which the agent rationally selects its course of action based on their representation. The mental categories are belief, desire, and intentions are supplemented by goals and plans. [3] These concepts are described as follows:

Beliefs of an agent express its expectations about the current state of the world and about the likelihood of a course of action achieving certain effects. Beliefs are modeled using possible worlds semantics, where a set of possible world is associated with each situation, denoting the worlds the agent belief is possible.

Desire is an abstract notion that specifies preferences over future world states or courses of action. An important feature of desire is that an agent is allowed to have inconsistent desire, and that it does not have to believe that its desires are achievable.

Intention: Since an agent is resource-bounded, it cannot pursue all its goals or options at once. It is often necessary to select a certain goal to commit, even if the set off goals is consistent.

Goals: The weak definition of desire mentioned above enforces an additional step of selecting a consistent subset of desires that an agent might pursue. However, there is not yet any commitment to specific courses of action. The notion of commitment describes the transition from goals to intentions. Further it is required that the agent believes its goals to be achievable.

Plans: Plans are very important for a pragmatic implementation of intentions. Intentions are partial plans of actions that the agent is committed to execute in order achieve its goals.

Figure 2. BDI Architecture

This architecture has high capability of general intelligent action towards a goal and agents' behaviour could be optimal and highly adaptable to different contingencies. However it has disadvantages like difficult in logical representation of belief, desire, time, and reasoning is hardly tractable.

Reactive Architecture

To overcome the problems of BDI agents, reactive architecture is evolved. In this architecture agent have very limited amount of information and their real-time decisions are based on sensory input and simple situation action rules. These agents are aimed for faster response and robust behavior instead of optimal behavior. This architecture has
limited scope and they are not able to execute a complex task that depends on long-term goals or co-operation.

Hybrid Architecture

Drawbacks of deliberative and reactive agents can be overcome by combining both these architectures. These Hybrid architectures are designed to respond rapidly to changes in the environment and to provide means to achieve long-term goals. Different functionalities and goals are arranged in different layers that interact in a well-defined control interface. This architecture is discussed in figure 3.

![Figure 3: Mobile Agent Layered Model](image)

Mobile Agent Development Standards

Intelligent agents and typically mobile agent paradigm is targeting adaptive and flexible co-operation, particularly for interoperability between or within distributed systems in a dynamically changing and mobile environment. This section discusses the efforts and the differences between the two groups, Foundations for Intelligent Physical Agents (FIPA)[4] and Object Management Group (OMG)[5] for developing specifications for agent interaction and mobility.

FIPAs intelligent agents take less time and transport capacity to migrate due to interoperability provided by agent communication language which has richer set of semantically standardised interactions between static software systems than the mobile agent paradigm. Agent communication paradigm and its languages can be more easily associated to a formal theory for agent interactions. Such theory enables the formal analysis and verification of the global distributed systems and can further increase the reliability of agent-based applications. Also it is easier to analyse the behaviour of a intelligent agent message. Therefore, the receiving intelligent agent can check the messages for subtle security and contract violations. Intelligent agents are therefore safer than mobile agents.[6]

And finally Mobile agents require homogeneous platforms for interoperability, while the intelligent agent paradigm supports the interoperability among heterogeneous environments. Thus FIPA specifications target at efficiency in terms of transport time and capacity, syntactical and semantic interoperability, richness of interaction protocols, security and reliability.

Whereas OMGs started working on a standard called Mobile Agent Facility (MAF), in order to promote interoperability among agent platforms. Later the standards name was changed to to Mobile Agent System Interoperability Facility[26](MASIF). It is based on agent platforms and it enables agents to migrate from one platform to another. The MASIF identifies a Distributed Agent Environment (DAE) and a Distributed Processing Environment (DPE). In a DAE, there are the following elements:

- **Place**: A place is a context in which an agent can execute, so a place is an execution environment.
- **Agency**: An agency is an agent system. An agency can have several places. An agent system represents a platform that can create, interpret, execute, and transfer agents.
- **Region**: A region is a group of agencies that belong to a single authority.

Two interfaces represent the core of the MASIF standard. The MAFAgentSystem is associated with every agency and provides operations for the management and transfer of agents. The MAFFinder: It is associated with a region. It supports localization of agents, agencies, and places in the scope of a region.

The following agent functionalities are covered by MASIF:

- **Agent management**: This comprises the creation, termination, suspension, and resumption of agents. The MAFAgentSystem provides several methods for this purpose.
- **Agent tracking**: Agencies, places and agents are registered in a region registration component via MAFFinder.
- **Agent transport**: MAFAgentSystem offers two methods to support agent migration.
- **Agent and agency naming**: Standardized syntax and semantics of agent and agency names enable agents and agencies to identify each other and allow dents to identify agents and agencies.
- **Agent type and location syntax**: Agency types provide information about important aspects of specific agencies, such as the used implementation language. The location is standardized in order to enable to locate each other.

MASIF relies on CORBA to handle agent security. [6] MASIF does not address the agent communication aspect. MASIF adopts a mobile agent paradigm, which is more appropriate in situations where dynamic and autonomous swapping, replacement, modification, and updating of application components are required.
Platforms

FIPA Complaint public available Platforms

• Agent Development Kit[7](Tryllian BV): The ADK is a mobile component based development platform using lightweight runtime environment based on Java that allows to build reliable and scalable industrial strength applications.

• Agentcities Forum [8]: Agentcities is an initiative that was first conceived in January 2000 to create a next generation Internet that is based upon a worldwide network of services that use the metaphor of a real or a virtual city to cluster services. These services, ranging from eCommerce to integrating business processes into a virtual organization, can be accessed across the Internet, and have an explicit representation of the capabilities that they offer. The ultimate aim is to enable the dynamic, intelligent and autonomous composition of services to achieve user and business goals, thereby creating compound services to address changing needs. Since its inception the testbed network has grown rapidly to support a wide range of small prototype and small test systems to large demonstrators involving over 170 deployed agent-based services.

April Agent Platform [8] (Jonathan Dale and Johnny Knottenbelt): It is FIPA-compliant agent platform that is designed to be a lightweight and gives powerful solution for developing agent based systems. It is written using the April programming language and the InterAgent Communication System (IMC).

• Comtec Agent Platform [9]: Comtec Agent Platform is an open-source, Java based, free implementation of FIPA 97-98 agent communication, agent management, agent message transport and some of the applications. Unique feature of it is the implementation of FIPA Ontology Service and Agent/Software Integration in SL2 as the content language.

• FIPA-OS[10](Emporpha): FIPA-OS was the first Open Source implementation of the FIPA standard and has already recorded thousands of downloads. FIPA-OS now supports most of the FIPA experimental specifications currently under development. FIPA-OS is a component-based toolkit implemented in 100% pure Java. One of the most significant contributions recently is a small-footprint version of FIPA-OS (FIPA-OS), aimed at PDAs and smart mobile phones.

• Grasshopper[11](Germany): Grasshopper is an open 100% Java-based mobile intelligent agent platform, which is compliant to both available international agent standards, namely the OMG MASIF and FIPA specifications. Grasshopper includes two optional open source extensions providing the OMG MASIF and FIPA standard interfaces for agent/platform interoperability.

• Jack[12](The Agent Oriented Software Group): JACK Intelligent Agents, is an environment for building, running and integrating commercial-grade multi-agent systems using a component-based approach. JACK is based upon the company’s Research and Development work on software agent technologies. The JACK Agent Language is a programming language that extends Java with agent-oriented concepts.

• JADE[13](TILAB, formerly CSELT): JADE simplifies the development of multi-agent applications, which comply with the latest FIPA 2000 specifications. While appearing as a single entity to the outside world, a JADE agent platform can be distributed over several hosts. Agents can also migrate or clone themselves to other hosts of the platform, regardless of the OS. The communication architecture tries to offer (agent transparent) flexible and efficient messaging by choosing, on an as-needed-basis, the best of the FIPA-compliant Message Transport Protocols (MTP) that are activated at platform run time. JADE is implemented in version 1.2 of JAVA and has no further dependency on third-party software.

• Java Agent Services[14] :Fujitsu, Sun, IBM, HP, Spawar, InterX, Institute of Human and Machine Cognition, Comtec, Verizon) The Java Agent Services (JAS) project defines an industry standard specification and API for the deployment of agent platform-service infrastructures. It is an implementation of the FIPA Abstract Architecture within the Java Community. Process [www.jcp.org] initiative and is intended to form the basis for creating commercial grade applications based on FIPA specifications. The API provides interfaces for message creation, message encoding, message transport, directory and naming. This design is intended to ensure that a JAS based system deployment remains transparent to shifts in the underlying technology without causing interruption to service delivery and therefore the business process.

• LEAP [15](Fr): LEAP (Lightweight Extensible Agent Platform) is a development and run-time environment for Intelligent Agents, is the precursor of the second generation of FIPA compliant platforms. It represents a major technical challenge - it aims to become the first integrated agent development environment capable of generating agent applications in the ZEUS environment and executing them on run-time environments derived from JADE, implemented over a large family of devices (computers, PDA and mobile phones) and communication mechanisms (TCP/IP, WAP). In this way LEAP benefits from the advanced design-time features of Zeus and the lightweight and extensible properties of JADE.

• ZEUS[16](BT Labs): ZEUS is an Open Source agent system entirely implemented in Java, developed by BT Labs and can be considered a toolkit for constructing collaborative multi-agent applications. Zeus provides support for generic agent functionality and has sophisticated support for the planning and scheduling of an agent’s actions. Moreover, Zeus provides facilities for supporting agent communication using FIPA ACL as the message transport and TCP/IP sockets as the delivery mechanism. Zeus provides facilities for building agents in a visual environment and support for redirecting agent behavior, too.

OMG MASIF Complaint Platforms

• AGLET[17]: IBM’s mobile agent platform is implemented in Java. An Aglets is actually a Java object
that can move from one host on the Internet to another. That is, an aglet that executes on one host can suddenly halt execution, dispatch itself to a remote host, and resume execution there. When the aglet moves, it takes along its program code as well as its data. The hosts need a ATP server to receive aglets.

- **Concordia[18]**: This is Mitsubishi Electric's Mobile Agent Environment. Concordia is a full-featured framework for the development and management of network-efficient mobile agent applications which extend to any device supporting Java. Concordia is written in Java and is portable to any platform running Java.

- **Grasshopper[11]** (Germany): Grasshopper is an open 100% Java-based mobile intelligent agent platform, which is compliant to both available international agent standards, namely the OMG MASIF and FIPA specifications. Grasshopper includes two optional open source extensions providing the OMG MASIF and FIPA standard interfaces for agent/platform interoperability.

- **Kafka[19]**: Kafka is an agent library designed for constructing multi-agent based distributed applications from Fujitsu. Kafka is a flexible, extendable, and easy-to-use Java class library for programmers who are familiar with distributed programming. It is based on Java's RMI. Kafka is now integrated together with Pathwalker, a process-oriented programming library.

**Conclusion**

The concept of an Intelligent Agent to work on behalf of the user and satisfy the program goals is now well established. The software entities or agents are fed with a certain goal and a set of capabilities (BDI agent). Using knowledge gained from its environment the agent works its way towards its goals under certain constraints imposed by its work environment. This concept of intelligent agents has opened a new vista in the world of computing. The use of agents in distributed systems is now gaining ground as the options available to the developer in such a system is numerous. Plenty of agent development frameworks have been proposed that enable creation, maintenance and destruction of agents. These platforms allow developers to empower the agents with intelligence that helps improve performance.

**References**


PAPER : Multi Agent System based Service Architectures for Service Level Agreement in Cellular Networks

Nupur Giri  
Asst Prof. Computer Engg. Dept.  
VESIT, Mumbai, India  
00919820793793  
nupur.giri@gmail.com

Siddharth Mundle  
Computer Engg. Dept.  
VESIT, Mumbai, India  
00919819319117  
siddharthmundle@gmail.com

Arpita Ray  
Computer Engg. Dept.  
VESIT, Mumbai, India  
009833182197  
arpiray@yahoo.com

Shrikant Bodhe  
Professor, E&T Dept.,  
BVCOE, Pune, India  
0091980094022  
s_k_bodhe@yahoo.co.in

ABSTRACT
To provide Service Level Agreement (SLA) in wireless cellular network the network provider needs to do resource management through call admission and channel allocation strategies. Also if the SLA is unsatisfactory it has to react rapidly to reinstate the promised quality of service (QoS), provide alternatives or do congestion control.

Our Paper proposes two Multi Agent System based, Centralized and Distributed, Service Architectures for SLA management. We extend the SHUFFLE model by introducing Network Provider Cell Agent (NPCA) which works in collaboration with Network Resource Provider Agent. Dynamic Call Admission is implemented using NPCA – NPCA interaction to accommodate real time traffic. The Agent are implemented using open source Java Agent DDevelopment Framework (JADE). The results of Multi agent and Non agent based connection level parameters are compared to understand the overhead characteristics of the systems.

Keywords
Call Admission Control, JADE, Multi Agent System, Service Level Agreement

1. INTRODUCTION
Service Level Agreement (SLA) is a pact between service providers (SP) and network providers (NP) or between service providers and customers specifying service guarantee in terms of service parameters, acceptable / unacceptable service levels, and action to be taken in special cases and penalties incurred by service provider in case the guarantee is not met. Resource management is very crucial aspect of any telecommunications system as it aims to maximize the utilization while maintaining SLAs.

Wireless networks present further challenges, as the users are moving and the traffic pattern changes rapidly. Therefore, the critical issue is the allocation and use of the bandwidth in the radio cells in order to avoid local congestion or degradation of the QoS. Call admission control (CAC) [6, 7, 10] is a fundamental mechanism used for quality of service (QoS) provisioning in a network. CAC limits the number of call connections into the networks in order to reduce the network congestion. Connection level QoS parameters such as new call blocking probability and handoff blocking probability, packet level QoS parameters such as delay, jitter, packet loss for various classes of traffic and mobility QoS parameters such as velocity, distance and direction of the movement of mobile terminal, all need to be considered together to come up with a complete solution for CAC.

Agents are software components, which find their use in resource management due to their autonomous, social, reactive, proactive, self learning properties. A Multi-agent System (MAS) [13] can be defined as a set of agents that interact with each other and with the environment to solve a particular problem in a coordinated (behaviorally coherent) manner. A MAS involves interaction, communication with system resources, collaborate, cooperate with other agents through Agent communication Languages (ACLs).

The paper attempts to model CAC as Multi Agent resource provisioning problem, where agents interact, cooperate with each other to find the optimal threshold for call admission using Centralized or Distributed architectures.

The SHUFFLE model [5] presents agent based layered architecture for Network management. We extend this model to introduce Connection plane and propose Network Provider Cell Agent (NPCA), which interacts with the Network Resource Provider Agent (NPRA) of Resource plane.
To take real time traffic into consideration Dynamic Call Admission technique is implemented which requires each cell agent to interact with other neighboring cell agents to get information of their number of ongoing call. This interaction helps the cell agent to derive the impact that a neighboring cell’s calls has on its own call admission. Java Agent DEvelopment Framework (JADE) an open source platform is chosen for implementation of co-operative MAS [12,1].

Section 2 reviews the agent and multi agent based work in the field of CAC. Section 3 presents the extended SHUFFLE model to accommodate Centralized and Distributed, the two proposed service architectures for SLA. Section 4 details of the Static and Dynamic CAC schemes being used in the architectures. Section 5 presents the two MAS based architectures along with their agent interaction model and finally section 7 analyses the simulation results presented in section 6.

2. REVIEW OF PREVIOUS WORK
Various non agent based CAC schemes [6, 7, 11] are proposed and implemented. The use of intelligent agent in Cheikhrouhou [4] brings network management awareness by splitting objectives to different layers to handle sub goals. In papers [2, 5] greater autonomy is given to the base station resulted in a distributed resource allocation scheme for first generation mobile networks using intelligent agents that offered an efficient solution for resource allocation under moderate and heavy loads. It emphasized on the choice of planning layer.

The most direct previous work is that of IST Project SHUFFLE (IST-1999-11014) [5]. In that project, the work of Bodanese was extended to 3G networks. However, the SHUFFLE project only offered the hypothesis that the agents could control SLAs, giving no details of implementation, or any results on SLA management. Gibney [8] proposed market based call routing based on self interested networks where as work done by Chantarash [3] is on Case Based Reasoning for congestion control.

Our work is extension of the SHUFFLE model, and intends in refining the model/architecture by introducing Network Provider Cell Agent (NPCA). It gives details of multi agent system for call admission control, complete with analysis and simulation for guaranteeing the required SLA.

3. EXTENDING SHUFFLE MODEL
The SHUFFLE Model divides the agents according to their responsibility assigned in Negotiation Plane and Resource Plane as shown in figure 1.

3.1 User Agent (UA)
UA resides within or near the User Terminal (UT) and acts on behalf of the user QoS. It maintains SLAs with all service providers to which the user subscribes by interacting with the service provider negotiation agent.

3.2 Service Provider Agent and Network Provider Agent (SPA & NPA)
They act on behalf of the Service Provider or Network Provider to control the overall policies. They coordinate the activities between different provider agents.

3.3 Service Provider Negotiation Agent & Network Provider and Negotiation Agent (SPNA & NPNA)
SPNA acts on behalf of a SP, being responsible for negotiating SLAs with the UAs of subscribing users and the network provider negotiation agents of the NPs that could be carrying the traffic from that SP. The NPNA acts on behalf of each NP to negotiate SLAs with SPs in order to manage the contract as actual traffic conditions vary.

3.4 The Network Provider Resource Agent (NPRA)
NPRA manages the resources. It acts on behalf of the NP to implement the policies of the NPA and to manage the radio resource of the NP.
4. MAS - CALL ADMISSION CONTROL (MA-CAC)

The MAS systems use either of the following schemes to do call admission.

The Static scheme requires NPRA-NPCA interaction for each incoming call where as Dynamic scheme calls for cluster of NPCA-NPCA interaction catering to real time traffic.

4.1 Static Cutoff priority based (S-CAC)

The Static Cutoff priority scheme [6] admits the calls in the cell if number of on-going calls in a cell is less than the threshold when new call arrives else the new call will be blocked. The handoff call is rejected only when all channels are used up.

If $\lambda$ and $\lambda_h$ is the arrival rate of new calls and handoff call respectively and $1/\mu$ and $1/\mu_h$ are average channel holding time for new calls and handoff calls, also if $m$ is the chosen cutoff threshold for j busy channels then the following stationary distribution for the approximate model is obtained with new call blocking probability and handoff call blocking probability as:

$$P_{nb} = \frac{\sum_{i=0}^{C} (p+\rho)^i \rho_j^{j-m}}{\sum_{i=0}^{C} (p+\rho)^i \rho_j^{j-m}}$$  \hspace{1cm} (1)

$$P_{hb} = \frac{C!}{\sum_{i=0}^{C} (p+\rho)^i \rho_j^{j-m}}$$  \hspace{1cm} (2)

4.2 Dynamic Cutoff priority based (D-CAC)

The Dynamic scheme requires the threshold to be calculated based on the ongoing calls in the neighboring cells. The analytical model for this Dynamic CutOff Threshold is described in paper [9, 11]. This threshold affects the connection level parameters, the new call blocking probability and handoff call blocking probability in the current cell.

The D-CAC is modeled to give higher priority to handoff calls as compared to a new call generated in the current cell. The simulation results of these parameters are presented in section 6.

5. MULTI AGENT SYSTEM BASED SERVICE ARCHITECTURES

The two architectures proposed are fundamentally different in the call admission decision-making aspect and thus require different interaction model for agent communication.

5.1 Centralized (NPRA- Based) Architecture

Figure 2 shows a heavy Network Resource Provider Agent (NPRA) catering to a set of Network Provider Cell agents (NPCA) in Centralized Architecture. The complexity of NPRA lies in the three sub layers of this agent namely Cooperative Planning layer, which interacts with peer NRPA, Local Planning layer which decides on the policies to be chosen for all the cells along with congestion control mechanism required and Reactive Layer which handles the call admission request initiated by the NPCAs. This layer is designed to be fast, performing the same function that would be in a conventional RNC (Radio Network Controller), assigning the connection to a NPCA, and performing CAC according to policies assigned by the planning layer. The Multi agent interactions along with the performatives used are presented in figure 3.

```
Figure 2. Centralized Multi Agent System: NPRA
```

```
Figure 3. Centralized MAS: NPRA- NPCA Dynamic Interaction
```
The connection ‘Request’ from cell agent ‘N’ along with the current number of ongoing calls is send to NPRA. In case of S-CAC the NPRA can take this decision depending on the preset limit of threshold and sends either ‘Accept/Reject’ to NPCA.

For D-CAC interaction, the NPRA ‘Proposes’ to do call admission for cell ‘N’ and accepts an ‘Inform’ from each neighboring NPCA as a response of this ‘Propose’. These response messages contain the information about ongoing calls in each of the NPCAs. The NPRA now calculates the threshold based on these response messages and conveys its decision of ‘Accept/Reject’ of incoming call to NPCAs of cell ‘N’.

As shown in figure 4 the NPCA agent comprises of Local Planning Layer and Call Establishment Layer. Depending on the type of architecture implemented the layers interact differently with each other. The NPCA in Centralized Architecture is a thin agent requiring to just respond to NPRA’s requests using Call Set Up Module in Connection Establishment Layer, through Centralized Module of Local Planning layer.

Hence a suitable policy can be chosen to match the current QoS, reporting to the Planning Layer from the Reactive Layer in order to maintain the system performance. Also for congestion control, resource reallocation might be required to be done for all NPCAs in centralized fashion.

5.2 Distributed (NPCA- Based) Architecture

The Distributed architecture uses NPCA along with D-CAC strategy to do call admission according to ongoing calls in neighboring cells and thus considering the neighboring cell’s load. This NPCA-NPCA cooperation (neighboring cell) can be termed as true Multi Agent System for SLA. It requires NPCA to be proactive as well be reactive and also thus actively participate in CAC and handle exceptions using Call Establishment Layer. The decision making of call admission is more granular and it lies with the cell itself (NPCA) rather than NPRA. The role of NPRA is reduced to overall management and changes in basic policies for congestion control, etc. whereas actual call admission and flow control needs to be now handled by NPCA.

This agent interaction for distributed architecture is shown in figure 5. The advantage of Dynamic Architecture is reduction in bottleneck at NPRA, which results in increasing scalability and faster decision making and less message overhead.

6. SIMULATIONS AND RESULTS

The two agent based systems are implemented using JADE 3.1. Traffic model has single traffic class and all mobile stations have arbitrary trajectories. The velocity remains constant during simulation. The calls are admits in NPRA of 25 NPCAs. Manhattan model of 9 NPCA cluster having cell area of 4 square kilometer is considered. Each cell is given 30 channels.

The new calls arrival rate, which forms a Poissons process, is \( \lambda \). The cutoff threshold chosen for simulations are 21, 25 and 27 for S-CAC and is dynamically calculated as explained in section 4 for D-CAC. The traffic load is varied from 10 to 100 Erlang.
Handoffs are performed automatically based on RSS value or parameterized handoff threshold. Emulation of forced call termination on insufficient signal strength is implemented. The system developed in the simulation model is a combination between discrete and continuous systems as some state variables change continuously with respect to time and some change at discrete points in time.

Figures 6 and 7 present the simulated results of new call blocking and handoff call blocking probabilities respectively for MA-CAC schemes. We see that in comparison with S-CAC, the handoff call blocking probability for D-CAC reduces significantly whereas the new call blocking probability dynamically varies and remains in the range defined by S-CAC $m=21$ and $m=27$ for higher load but is low for less load. The two Multi agent based service architectures are compared with Non agent based architectures in terms of elapsed time and ACL message overhead for different number of calls. This gives us a fair estimate of their efficiency over Non agent based service architectures.

### 7. ANALYSIS AND CONCLUSION

The time required for simulation is tabulated in tables II and III which show that it is much higher in Non Agent based architecture for D-CAC scheme as compared to that required for MAS based architectures. Also MAS based Distributed architecture using D-CAC has reduced handoff call blocking probability.

Although Message overhead in table I is higher in Distributed Architecture as compared to Centralized Architecture but again it is less by about 20% in D-CAC scheme as compared to S-CAC scheme for Distributed Architecture.

### Table I: Message Overhead

<table>
<thead>
<tr>
<th>Number of Calls</th>
<th>Centralized S-CAC</th>
<th>Multi Agent Based S-CAC</th>
<th>Distributed D-CAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>25000</td>
<td>503493</td>
<td>1509792</td>
<td>1258740</td>
</tr>
<tr>
<td>50000</td>
<td>1018441</td>
<td>3015494</td>
<td>2519904</td>
</tr>
<tr>
<td>75000</td>
<td>1525696</td>
<td>4528653</td>
<td>3778638</td>
</tr>
<tr>
<td>100000</td>
<td>2039653</td>
<td>6038237</td>
<td>5038136</td>
</tr>
<tr>
<td>1250000</td>
<td>2545996</td>
<td>7548497</td>
<td>6293734</td>
</tr>
</tbody>
</table>
Table II: Simulation Time in seconds: For S-CAC and D-CAC

<table>
<thead>
<tr>
<th>Number of Calls</th>
<th>Non agent Based S-CAC</th>
<th>D-CAC</th>
<th>Multi Agent Based NPRA S-CAC</th>
<th>NPRA D-CAC</th>
<th>NPCA D-CAC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25000</td>
<td>194.34</td>
<td>1694.58</td>
<td>628.10</td>
<td>1301.40</td>
<td>983.57</td>
</tr>
<tr>
<td>50000</td>
<td>371.28</td>
<td>3372.50</td>
<td>1338.14</td>
<td>2584.57</td>
<td>1928.92</td>
</tr>
<tr>
<td>75000</td>
<td>728.82</td>
<td>5120.25</td>
<td>1769.93</td>
<td>3977.56</td>
<td>3075.95</td>
</tr>
<tr>
<td>100000</td>
<td>869.20</td>
<td>6815.72</td>
<td>2506.98</td>
<td>5302.04</td>
<td>3992.45</td>
</tr>
<tr>
<td>125000</td>
<td>1224.5</td>
<td>8740.68</td>
<td>3277.75</td>
<td>6682.01</td>
<td>4986.95</td>
</tr>
</tbody>
</table>

Table III: Simulation Time in seconds: Non agent and Multi Agent based D-CAC

<table>
<thead>
<tr>
<th>Number of Calls</th>
<th>Non agent Based D-CAC</th>
<th>Multi Agent Based Centralized D-CAC</th>
<th>Appr. % Reduction</th>
<th>Distributed D-CAC</th>
<th>Appr. % Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25000</td>
<td>1694.58</td>
<td>1301.40</td>
<td>24</td>
<td>983.57</td>
<td>42</td>
</tr>
<tr>
<td>50000</td>
<td>3372.50</td>
<td>2584.57</td>
<td>25</td>
<td>1928.92</td>
<td>43</td>
</tr>
<tr>
<td>75000</td>
<td>5120.25</td>
<td>3977.56</td>
<td>23</td>
<td>3075.95</td>
<td>40</td>
</tr>
<tr>
<td>100000</td>
<td>6815.72</td>
<td>5302.04</td>
<td>25</td>
<td>3992.45</td>
<td>42</td>
</tr>
<tr>
<td>125000</td>
<td>8740.68</td>
<td>6682.01</td>
<td>25</td>
<td>4986.95</td>
<td>43</td>
</tr>
</tbody>
</table>

8. REFERENCES
Multi Agent based Call Admission Control for managing Multi Class Traffic in Cellular Networks

Nupur Giri¹, Keyur Dhami¹, Sumitabh Swarnakar¹, Srikant Bodhe²

¹Computer Engineering Department, Vivekanand Education Society’s Institute of Technology, Chembur, Mumbai, India
nupur.giri@gmail.com, keyur_dhami@yahoo.co.in, sumitabh_007@yahoo.co.in

²Electronics and Telecommunication Department, Bharti Vidyapeeth College of Engineering, Pune, India
s_k_bodhe@yahoo.co.in

Abstract. In wireless cellular networks, collaboration, coordination and negotiation amongst cells help in performing channel allocation and call admission in fair, efficient and distributed manner. Thus help in increasing utility function of the system as a whole and providing Quality of Service guarantee as specified in Service Level Agreement. Multi Agent System (MAS) extends intelligent agent paradigm to improve upon the conceptual model of designing problems through agent interaction. Our paper models (priority/non-priority) call admission control for multi class traffic in cellular networks as multi agent resource provisioning problem, where agents interact with each other to find the optimal call admission thresholds. The MAS is modeled using open source Java Agent DEvelopment framework (JADE). The paper presents the design of Network Provider Cell Agents (NPCA), their interaction models and performatives. The performance evaluation of connection level parameters for multi agent Static and Dynamic call admission control strategies is also presented.

Keywords: multi class traffic, multi agent, call admission, JADE

1 Introduction

Quality of Service (QoS) Engineering comprises of traffic characterization, admission control, network resource allocation, server/link scheduling, traffic policing/shaping, QoS mapping between networks and different layers, Service Level Agreements (SLA), policy management, radio channel selection, bandwidth allocation, RF resource setup and teardown strategies. This complex domain can exploit the power of agent paradigm to enhance performance along the dimensions of computational efficiency, reliability, extensibility, robustness, maintainability, responsiveness, flexibility and reuse [7].

Multi Agent System (MAS) [9], [10] can be defined as a loosely coupled network of problem solver agents that interact to solve problems that are beyond the individual capabilities or knowledge of each. These problem solving agents are autonomous and
can be heterogeneous in nature. MAS typically find their use in solving problems that are too large for a centralized agent to solve because of resource limitations or the sheer risk of having one centralized system that could be a performance bottleneck or could fail at critical times.

With this paper we attempt to model distributed/decentralized call admission as MAS, using dynamic strategies for multi class traffic (priority/non priority based) to give the above mentioned advantages.

The cellular system possess following properties:
1. The individual cell has incomplete information or capabilities to perform call admission on its own, and have limited viewpoint of the cellular network.
2. There is no global system control.
3. Computations are asynchronous.

MAS is the most suitable model for multi tier based SLA modeling because as in MAS, our cells too require coordination to admit the calls according to ongoing calls in the neighboring cells.

In section 2 we present the review of agent based call admission. Section 3 presents the system model and traffic model for call admission and formulates the problem in hand. Section 4 models multi class call admission as multi agent system and also explains the interaction model for the same. Section 5 explains the JADE based implementation of MAS. Section 6 presents the validated results. Section 7 and section 8 present the analysis and conclusion respectively.

2 Review

Although various call admission control (CAC)[6], [7] schemes have been researched thoroughly, here we review only agent based work in the domain of network management and resource allocation. In papers [2], [3] greater autonomy is given to the base station resulted in a distributed resource allocation scheme for first generation mobile networks using intelligent agents. It offered an efficient solution for resource allocation under moderate and heavy loads. The use of intelligent agent in Cheikhrouhou [4] brings network management awareness by splitting objectives to different layers to handle sub goals. The most direct previous work is that of IST Project SHUFFLE [5]. In this work of Bodanese he only offered the hypothesis that the agents could control SLAs, giving no details of implementation, or any results on SLA management. Work done by Chantaraskul [3] is on case based reasoning for congestion control.

Our work is influenced by the SHUFFLE project, and extends the model to define Network Provider Cell Agent (NPCA) and Network Resource Provider Agent (NPRA) for call admission control. We also present the interaction model of these agents and their implementation in JADE. The performance of this MAS is evaluated by simulating single class traffic using different CAC schemes (static, dynamic and mobility based thresholds) in Giri [8]. In this paper we present performance analysis of the same for multi-class traffic for static/dynamic threshold based CAC, as well as priority based non priority based CAC schemes.
3 Problem Formalization

Fig. 1 presents the multi class traffic model of a cell having C channels (TDMA/FDMA/CDMA). We define two classes of multi media traffic typically class 1 for voice and class 2 for data. The new calls of each traffic class are assigned channels till an ongoing call threshold is reached (static/dynamic: depending on the Multi Agent-CAC (MA-CAC) scheme chosen) or the call is dropped. This threshold could be either static or could be found out dynamically by exchanging information with the neighboring cells.

In non priority based call admission voice and data hand off calls are given equal preference and thus no queue is assigned. Where as in priority based call admission voice handoff call is given priority over data handoff calls, thus voice handoff calls are not blocked but buffered/queued if channels are not available.

3.1 Multi Class Traffic Model

Utility functions describe satisfaction level with the perceived quality of service. The higher the utility, the more satisfied the customer. The utility of the system depends on the increased performance of the system in terms of its reactiveness. It requires balancing the utility of each cell, so that the entire system is in steady state and not congested, thus guaranteeing the optimally balanced utility for the entire system. The optimal utility of the system needs to be calculated by coordinating with individual cell.

We define utility function $U_c$ for each cell.
We assume that the utility function of a single cell $U_s$ is differentiable and monotonically decreasing concave function of the QoS parameter $P_{hd}$ where $P_{hd}$ is handoff dropping probability of a cell.

Therefore $U_s = F(P_{hd})$ where function $F(P_{hd})$ has following properties.

\[ F(P_{hd}) \geq 0; \quad F'(P_{hd}) < 0; \quad F''(P_{hd}) < 0 \]  

(1)

$U_s$ is maximum at $P_{hd} = 0$, which means that if the handoff dropping probability is zero for a cell, then user using the cell has the highest level of satisfaction.

$U_s(P_{hd} = 0) = (U_s)_{max}$ Let there exist a $(P_{hd})_{max}$ such that

\[ U_s(P_{hd} = \text{max}) = 0 \]  

(2)

There exists a $0 \leq (P_{hd})^{old} \leq (P_{hd})_{\text{max}}$ that can be tolerated so that the quality of service and utility $(U_s)^{old}$ is considered acceptable. Based on the above definitions and assumptions we need to find $(P_{hd})^{old}$.

We already know that as number of calls in a particular cell increases, the $P_{hd}$ will also increase.

Thus, now we need to find $n_c^{old}$, which is the optimal number of ongoing calls in a cell, say $S$, which maintains the handoff dropping probability within the tolerable limit.

We further assume that number of ongoing calls in the neighborhood (defined by top, right, left and bottom neighboring cells) of cell $S$ affects the maximum number of ongoing calls in cell $S$, as few of those calls could be handed off to the cell with some probability. Also, few ongoing calls from cell $S$ might be handed off to neighboring cells.

Thus, the problem now is to find dependency of $n_c^{old}$ to the neighboring cell such that $P_{hd}$ remains acceptable = $(P_{hd})^{old}$.

The dynamic new call threshold calculation [12] process requires the threshold of call admission of a new call to be calculated based on the “ongoing calls”.

Let a new call admission request be made in addition to the state of its adjacent cells, $T$ units of time after the call admission. Let $C$ denote the number of calls which a cell can support. If $n$ denotes ongoing calls in the current cell $S$, then $t$, $r$, $b$ and $l$ are the number of ongoing calls in neighboring cells (top, right, bottom, left) respectively.

To calculate this threshold following conditions should be checked:

1) whether by admitting a new call at $t_0+T$, the current handoff probability of cell $S_n$, due to incoming handoff calls from $S_n$, $S_t$, $S_l$, and $S_r$ and also outgoing handoffs from $S_n$ is less than the threshold of call admission.

2) and whether the handoff probability of cell $S_t$ $(S_t/S_l/S_r)$ affected by handoffs from cell $S_n$ (top), $S_t$ (right), $S_l$ (bottom) and $S_r$ (left) is less than threshold of call admission.

Let $E(n)$ be the average calls per cell in the neighboring cells of $S_t$ $(S_t/S_l/S_r)$. Let’s assume $P_s$ is the probability that the call remains in the same cell. Let $P_{hd}/4$ be the probability that call will be handed off to neighboring cell during time $T$ and ‘a’ be tunable.
\[ n_c = \frac{a^2(1-p_s) + 2C - SUMp_m}{2-a^2(1-p_s)^2 + 4C(1-p_s) + SUMp_m p_s - SUMp_m^2 / 4} \]  
\[ n_s = \frac{a^2(4-p_s) + 8(C - \lambda T - rp_s) - 2E(n)p_m - a^2(4-p_s)^2 + 64C + 16rp_s(\lambda T + rp_s - C) - 64rp_s^2}{2p_m} \]  
\[ M = n_c \text{ tol} = \min(n_c, n_t, n_r, n_b, n_t) \]

Thus \( M \) is the dynamic threshold which is a function of calls in the neighboring cells.

### 3.2 Modeling Multi class Call Admission Control (M-CAC)

To further increase the utility, instead of blocking the handoff calls, the voice handoff calls are queued. The queued calls are allotted the channel as the channels become available (taking care that the call does not terminate in the queue). By varying the size of the queue (\( K=0 \) for non priority, \( K>0 \) for priority) one can ensure the required SLA for a particular class of traffic.

The Fig. 2 represents the state transition diagram for traffic model presented in the section above [10]. The state of the is defined by state \( i = 0, 1, \ldots, M, M+1, \ldots, C, C+1, \ldots, C+K \) where \( i \) is the number of connected calls in the cell plus the number of voice handoff requests in the queue. The call holding time, resident time of the call and time spent in the cell residency time are all assumed to be exponentially distributed with mean as \( \mu^{-1} \) and \( \eta^{-1} \) and \( \alpha^{-1} \) respectively. The average channel holding time \( \tau^{-1}=(\eta+\mu)^{-1} \) for both classes of traffic is assumed to be the same. Let \( \lambda_T \) be the call arrival rate, \( \lambda_{nv}, \lambda_{nd} \) be call arrival rate for new voice and data calls and \( \lambda_{hv}, \lambda_{hd} \) be call arrival rate for handoff voice and data calls such that:

\[ \lambda_T = \lambda_{nv} + \lambda_{nd} + \lambda_{hv} + \lambda_{hd} \]
To find the steady state probability $P_i$, the balanced equations are:

\[ i \sigma P_i = \lambda_i P_{i-1} \quad 0 \leq i \leq M \]  

\[ i \sigma P_i = (\lambda_{hv} + \lambda_{hd}) P_{i-1} \quad M < i \leq C \]  

\[ C \sigma + (i-C) \alpha P_i = \lambda_{hv} P \quad C < i \leq C + K \]  

\[ P_i = \begin{cases} 
  \frac{\lambda_i^M}{\prod_{i} i \sigma_i} & 0 \leq i \leq M \\
  \frac{\lambda_{hv}^M (\lambda_{hv} + \lambda_{hd})^{y-M}}{C \sigma C} & M < i \leq C \\
  \frac{\lambda_{hv}^M (\lambda_{hv} + \lambda_{hd})^{y-M}}{C \sigma C} \prod_{j=1}^{C} \frac{\lambda_{hv}^j}{C \sigma + j \alpha} & C < i \leq C + K 
\end{cases} \]  

The blocking probabilities $B_{hv}$, $B_{hv}$ (voice) and $B_{hd}$, $B_{hd}$ (data) are:

\[ B_{hv} = P_{C+K} = \left[ \frac{\lambda_{hv}^M (\lambda_{hv} + \lambda_{hd})^{y-M}}{C \sigma C} \prod_{j=1}^{C} \frac{\lambda_{hv}^j}{C \sigma + j \alpha} \right]^{-1} \]  

\[ B_{hd} = \sum_{i=1}^{C+K} P_i \]  

\[ B_{nv} = B_{nd} = \sum_{i=M}^{C+K} P_i \]  

4 Modeling of MA-CAC as Multi Agent System

To model call admission in cellular network as a multi agent system, we consider that each cell has an agent called Network Provider Cell Agent (NPCA) as shown in Fig. 3. The channel resources to this NPCA are assigned by Mobile Switching Center.
(MSC) which is also modeled as an agent called Network Provider Resource Agent (NPRA). A multi agent based 3X3 Cluster of 9 NPCAs represents the micro cell, Manhattan model. We assume that the mobile terminal can move in only 4 directions (top, right, bottom, left). The NPCA cluster is presented in Fig. 4.

Fig. 3: Network Provider Cell Agent (NPCA) Architecture

Fig. 4: Network Provider Cell Agents (NPCA) Cluster
4.1 Agent Architecture

As shown in Fig. 3 the Network Provider Cell Agents (NPCA) comprises of Local planning layer and Call establishment layer. This layer participates in cooperative interaction with other neighboring NPCAs. They form a multi agent cluster to find the optimal threshold for the new calls, which will ensure that the network does not lead to congestion. This is peer to peer or horizontal multi agent interaction as shown in Fig. 4. The Call establishment layer actually performs the admission/rejection according to the threshold found by Distributed module of the Local planning layer. If the call is accepted, Call set up module registers the call.

5 Modeling Multi Agent System in JADE

The implementation of architecture for all three MA-CACs is using open source JADE 3.1.

The reason for selection of JADE as compared to other [13] is that it is Foundation for Intelligent Physical Agent (FIPA) complaint and runs on variety of operating systems including Windows and Linux. JADE offers support for multi agent interaction, coordination and competition through Agent Communication Language (ACL) messages using FIPA Performatives.

This matches the requirements of high number of agents and message exchange. A dynamic scheme calls for cluster of NPCA-NPCA interaction. The process of starting NPCA agent consists of “registration” with corresponding NPRA.

One NPRA agent can at most cater to 25 NPCAs. The agents implement the “cyclic behavior” of JADE to simulate the discrete event generation of call admission. The Controller agent is used to maintain synchronization amongst NPRA and NPCAs. We define one NPRA agent in the main container. NPCA agents can be invoked in different containers.

Table 1: JADE performatives modeled for MA-CAC

<table>
<thead>
<tr>
<th>Performatives</th>
<th>Meanings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Propose</td>
<td>Issued by the NPCA to propose performing call admission control to neighboring cells</td>
</tr>
<tr>
<td>Inform</td>
<td>Issued as response to ‘Propose’. The message carries ‘Ongoing Calls’, ‘percentile of high/low velocity traffic’ and also the ‘class of traffic’ information</td>
</tr>
<tr>
<td>Request</td>
<td>Requests NPRA to perform congestion control.</td>
</tr>
<tr>
<td>Accept</td>
<td>Confirms acceptance of call</td>
</tr>
<tr>
<td>Reject</td>
<td>Rejects the call</td>
</tr>
</tbody>
</table>

JADE offers 12 message performatives out of which five have been implemented as shown in Table 1. The interaction amongst these agents could be of the form of: cooperation (agents share a common goal); coordination (scheduling and
synchronization) for mutual benefit and negotiation (coming to an agreement for mutual beneficial solution). Here, we use coordination as the interaction model (Fig.5) and ensure that the agents synchronize with each other and communicate the number of ongoing calls and mobility information. The NPCA of cell ‘S’ (on which the call arrives) ‘Proposes’ to perform call admission for cell ‘S’ by interacting with all its neighbors, thus performing MA-CAC interactions. It accepts an ‘Inform’ from each neighboring NPCAs as a response of this ‘Propose’. These responses contain the information about ongoing calls/mobility pattern in each of these NPCAs. The NPCA now calculates the threshold based on these responses and ‘Accept/Reject’ an incoming new call. In case of handoff calls the call is admitted if the channel is available. JADE supports the Sniffer Agent to sniff the ACL messages exchanged between agents.

Congestion control could be achieved either by reallocation of channels to cells (NPCAs), involving NPRA-NPRA negotiations or could be by local flow control (adjusting the new call blocking thresholds), which could be handled by NPCAs. As handoff calls are given priority over new calls, handoff calls could be queued.

Fig. 5: Agent interaction for Distributed Architecture
6 Simulation and Results

We have assumed Manhattan model for simulation. The traffic is urban traffic, thus the maximum speed of a mobile user can be 40 km/hr. This velocity is assumed to remain constant. The traffic distribution is as follows:

Table 2: Traffic Classes. (Voice, data e-mail-Fax & video conferencing)

<table>
<thead>
<tr>
<th>Traffic Class</th>
<th>Traffic Class</th>
<th>Traffic Distribution</th>
<th>Call Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voice Service</td>
<td>Class 1</td>
<td>0.6</td>
<td>60 seconds</td>
</tr>
<tr>
<td>E-mail, Fax</td>
<td>Class 2</td>
<td>0.30</td>
<td>180 seconds</td>
</tr>
<tr>
<td>Video conferencing</td>
<td>Class 3</td>
<td>0.10</td>
<td>600 seconds</td>
</tr>
</tbody>
</table>

The Figures 6 and 7 illustrate the new call blocking and handoff call blocking probabilities for non-priority based static threshold CAC strategy. The graph compares new call blocking probabilities for data and voice calls for various thresholds in Non Agent (NA) as well as Multi Agent (MA) environment.

We see that as the threshold increases (reserved channels for handoff call decreases) new call blocking probability decreases. This can be analyzed as more number of channels are available for new calls, the new call blocking probability definitely will decrease where as handoff call blocking for voice as well as data call will increase because the number of reserved channel for handoff calls is decreasing, i.e. the new call blocking for 21 static threshold is higher than that for 27, where handoff blocking more.

Table 3: Simulation Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traffic</td>
<td>10 - 100</td>
<td>Traffic Erlang</td>
</tr>
<tr>
<td>$\lambda_T$</td>
<td>Traffic / $\mu$</td>
<td>Total rate of arrival \ $\lambda_n + \lambda_d + \lambda_{hv} + \lambda_{hd}$</td>
</tr>
<tr>
<td>$\lambda_{nv}$</td>
<td>0.75 * $\lambda_T$</td>
<td>Rate of new voice call arrival</td>
</tr>
<tr>
<td>$\lambda_{nd}$</td>
<td>0.35 * $\lambda_T$</td>
<td>Rate of new data call arrival</td>
</tr>
<tr>
<td>$\lambda_{hv}$</td>
<td>0.25 * $\lambda_{nv}$</td>
<td>Hand off voice call arrival rate</td>
</tr>
<tr>
<td>$\lambda_{hd}$</td>
<td>0.10 * $\lambda_{nd}$</td>
<td>Hand off data call arrival rate</td>
</tr>
<tr>
<td>C</td>
<td>30</td>
<td>Total number of channels</td>
</tr>
<tr>
<td>$M = r_c^{no}$</td>
<td>21/25/27/dynamic</td>
<td>New call blocking Threshold</td>
</tr>
<tr>
<td>K</td>
<td>0,1-5</td>
<td>Queue length of handoff call</td>
</tr>
<tr>
<td>Cell size</td>
<td>4 sq km</td>
<td>Square shaped cells</td>
</tr>
<tr>
<td>MAS Cluster</td>
<td>9</td>
<td>Network Provider Cell Agent (NPCA)</td>
</tr>
</tbody>
</table>
Fig. 6: Comparing Multi Agent (MA) and Non Agent (NA), Non-Priority Based (K=0), Static Threshold CAC schemes: New Call Blocking Probability.

Fig. 7: Comparing Multi Agent (MA) and Non Agent (NA), Non-Priority Based (K=0), Static Threshold CAC schemes: Handoff Blocking Probability.

Though the nature of the graph remains same in Fig. 8 and Fig. 9 (priority/queue based CAC) and the new call blocking for voice call has marginally increased the handoff call blocking probability of voice reduces substantially for all thresholds for K>0 as compared to that of K=0 as the less number of voice handoff calls are now blocked. But the blocking probability of data handoff calls increases.
Fig. 8: Simulation of Multi Agent (MA) and Non Agent (NA), Priority Based (K=2), Static Threshold CAC schemes: New Call Blocking Probability.

Fig. 9: Simulation of Multi Agent (MA) and Non Agent (NA), Priority Based (K=2), Static Threshold CAC schemes: Handoff Blocking Probability.
Fig. 10 and 11 illustrate the natures of dynamic-distributed threshold based CAC scheme the new call blocking probability varies and remains in the range as defined by static thresholds (21 and 27), but for lower traffic values it is much lesser.

**Fig. 10:** Comparing Multi Agent (MA) and Non Agent (NA), Priority and Non Priority based Distributed Threshold CAC schemes: New Call Blocking Probability

**Fig. 11:** Comparing Multi Agent (MA) and Non Agent (NA), Priority and Non Priority based Distributed Threshold CAC schemes: Handoff Blocking Probability
The handoff call blocking is comparatively very less as the dynamic threshold is
calculated keeping the ongoing calls in neighboring cell in mind. Thus using this
scheme further increases the utility function of the voice handoff calls.

Tables 4 give us an estimate of the reactivity of multi agent based CAC over non
agent based CACs, simulated on the same machine configuration. We see that by using
agent based schemes there is a significant reduction in simulation time, approximately
of 11-12%.

Table 4: Simulation time for Non Agent Vs Multi Agent System

<table>
<thead>
<tr>
<th>No. of Calls</th>
<th>Distributed - Dynamic Threshold</th>
<th>Appr. % reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Non Agent based</td>
<td>Agent based</td>
</tr>
<tr>
<td>25000</td>
<td>873.422</td>
<td>778.33</td>
</tr>
<tr>
<td>50000</td>
<td>1625.47</td>
<td>1455.38</td>
</tr>
<tr>
<td>75000</td>
<td>2947.5</td>
<td>2625.90</td>
</tr>
<tr>
<td>1,00,000</td>
<td>3736.21</td>
<td>3344</td>
</tr>
<tr>
<td>1,25,000</td>
<td>4614.02</td>
<td>3992.64</td>
</tr>
</tbody>
</table>

7 Conclusion and Future Scope

The results presented in this paper are for multi class traffic (non priority /priority)
CAC schemes having uniform user mobility pattern. These results very clearly point
out the benefits of multi agent based over non agent based Systems in terms of
computational efficiency and responsiveness. The multi agent system designed here
can be easily integrated with SHUFFLE model, thus making the system extendable
and extensible by easily changing/varying the capabilities of the agents according to
the type of traffic and required flow control to meet the SLA guarantee. Besides these
explicit benefits the implicit benefits of using multi agent system are maintainability
due to the modular and layered NPCA agent design and low coupling of NPCA-
NPCA only through ACL message passing performatives.

Integrating congestion control schemes will further explore the full benefit of multi
agent system for wireless cellular networks. Various interactions models,
coordination, co-operation and negotiation can be used to understand the advantage of
one over other for the domain.

References

2. Bodanese, E., Cuthbert L.G.: A Multi-Agent Channel Allocation Scheme for Cellular Mobile
   Networks. In: 4th International Conference on Multi-Agent Systems, pp. 63--70, Boston,
   MA, USA (2000)
   workshop of HP OpenView University Association, Bologna, Italy (1999)


Evaluating Multi Agent based Service Architectures for Call Admission Control

Nupur Giri  
Assistant Professor, Computer Engineering Dept., VESIT, Mumbai, India.

Charu Jain  
Computer Engineering Dept., VESIT, Mumbai, India

Anjana Rao  
Computer Engineering Dept., VESIT, Mumbai, India

Ishita Chakraborty  
Computer Engineering Dept., VESIT, Mumbai, India

Shrikant Bodhe  
Principal, SVERI’s College of Engineering, Pandharpur, Maharashtra, India.

Abstract—Multi Agent Systems find their use in various diverse domains where interaction, collaboration, cooperation, or negotiation with other agents and with the environment is required to solve a particular problem. In this paper we model call admission control as a multi agent system and attempt to evaluate the impact that the degree of distribution of agents and interaction models of multi agent architectures have on the call admission control performance attributes such as reactivity, responsiveness, message overhead, utility and resource utilization, as well as general quality attributes such as robustness, modifiability, and scalability. This knowledge becomes essential while choosing the correct MAS architecture to meet the specified service level agreement.

Keywords—Multi Agent Architecture, Call Admission Control, Cellular Networks

I. EVALUATING MULTI AGENT ARCHITECTURES

The degree of distribution of agents in a multi agent based architectures affects performance characteristics according to the domain of implementation [1]. The work presented in this paper demonstrates this effect for the domain of MAS based Call Admission Control (CAC) for Service Level Agreement (SLA) guarantee [2,3,4]. The degree of distribution categorizes the architectures into Centralized Service Architecture (CSA) and Distributed Service Architecture (DSA) as presented in [4]. Network Provider Resource Agent (NPRA) acts as the controller agent for CSA that controls Network Provider Cell Agents (NPCA), whereas in DSA peer-to-peer communication takes place between NPCA-NPCA. [5]

The following sections presents the theoretical, or qualitative analysis of the influence, the degree of distribution of agents and the interaction models of these two multi agent based service architectures have on the performance related attributes for Dynamic-Call Admission Control [6]. The later section presents the experimental results of these qualitative analyses. The attributes can be defined as:

1) Reactivity (T_{RVTY}) : It is a measure of how promptly the Multi agent architecture reacts to the event of call arrival at a particular NPCA cell and assigns the call. It can be defined as the sum of the times taken by the agents to interact (T_{RES}) (which is also called responsiveness), make a decision about call admission/rejection by calculating dynamic threshold (T_{THRESH}), and finally allocate a channel to the call (T_{ALLOC}), from the time of call arrival (T_{ARRV}).

\[ T_{RVTY} = T_{RES} + T_{THRESH} + T_{ALLOC} - T_{ARRV} \]  \hspace{1cm} (1)

T_{THRESH} is the CPU computation time for threshold calculation. Also T_{ALLOC} depends on the resources available at that time and not on the service architecture.

2) Responsiveness (T_{RES}): This is defined as the time for which the agents are involved in message passing. These interactions, which are in the form of request and reply, enable the requester agent to calculate the dynamic threshold according to D-CAC. Thus the responsiveness of this multi agent interaction is defined as the time elapsed from the sending of the first request message (performative) by an agent till the arrival of a response message from all its neighboring agents.

\[ T_{RES} = \sum_{i=1}^{C} T_{INFORM} - T_{PROPOSE} \]  \hspace{1cm} (2)

3) Communication Overhead: Communication overhead can be measured either by the number of messages exchanged, during agent interaction or by the bandwidth required for allocation. Communication tends to be concentrated in smaller areas (localised) in DSA as compared to CSA.

4) Sustainability under high traffic conditions: This parameter is a measure of the ability of the architecture to sustain a higher traffic intensity while still maintaining the utility of the system. The user utility function is defined as the measure of the satisfaction level of the user with respect to a
perceived quality of service. The higher the utility, higher the user satisfaction.

5) Utilization of resources: It is a measure of the efficiency of cell agents with respect to available resource utilization. It can be measured by the ratio of carried load to offered load at any point of time.

\[
U_T = \frac{L_{\text{Carried}}}{L_{\text{Offered}}}
\]

\[L_{\text{Offered}} = (\text{New call arrival rate / cell}) \times \text{mean call hold time.}
\]
\[L_{\text{Carried}} = \text{Average no. of ongoing calls in each cell at any time}\]

6) Modifiability: It is the ease with which a system can be changed after it has been implemented or deployed. This is measured in terms of two parameters: first is the ease with which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling which agents can be added/removed from the service architecture which depends on the modularity i.e. coupling.

When new agents are added to a CSA system, the change is reflected only at the NPRA, whereas in DSA, addition of an agent changes the cluster dimensions, i.e. the number of clusters to handle or the size of the cluster for D-CAC threshold calculations. These changes need to be reflected at the every NPCA.

Moreover, since the call admission decision-making is entirely with the NPRA in case of centralized architecture, NPRA can pass CAC schemes down to the NPCAs as policies. Since these policies change dynamically at the NPRA, each NPCA needs to change its local CAC strategies according to the NPRA’s suggestions. The global view of the network, at the NPRA, makes manageability easier.

3) Scalability: Scalability can be measured by the change in performance parameters with increase in the number of NPCA agents. Scalability seems to be better supported by distributed architecture than centralized architecture. This is because the computational load of call assignment is divided between a number of NPCAs in a distributed architecture. Also, there remains no possibility of a communication bottleneck at the NPRA, thus responsiveness of the system is maintained.

7) Robustness: Robustness is defined by the vulnerability of the system to agent failures. Regarding robustness we conclude that more centralized the control is, the more vulnerable the system gets. This is because the system cannot perform call assignment, if the NPRA agent, responsible for centralized control fails. In a more distributed architecture, the assignment may function partially even though some NPCA agents have failed.

II. SIMULATION MODEL

A. Radio Network Model:

Manhattan model of a city consisting of square blocks with streets in between is used with 25 base stations modeled as NPCA agents. Radio Network Controller (RNC) is modeled as NPRA. Each NPRA caters to 25 NPCA.

B. Traffic and mobility model:

Single class (voice) traffic is considered. Average new call arrival rate follows Poisson distribution. The call holding time is assumed to have exponential distribution with mean as 120s. To eliminate the edge effect the system is wrapped around at the edges.

C. Mobility model:

Normal Distribution speed (v ∈ [30km/h, 90km/h]). The motion of the mobile is restricted along the streets, and can move to left, right as well as top, bottom streets. The mobile user moves with uniform speed throughout the call direction and does not change the direction during that ongoing call.

D. Channel allocation and CAC scheme:

Fixed channel allocation is assumed for evaluation of multi agent architecture but 2-D Dynamic Call Admission control scheme (D-CAC) is chosen for call admission, i.e. a new call is admitted into the cell depending on dynamic cutoff threshold determined by the ongoing calls in the neighboring cells (D-CAC). This neighborhood cluster is made of right, left, top and bottom NPCA cells of Manhattan model.

E. Agent Model:

These NPCA as well NPRA agents are simulated in JADE 3.1 (Java Agent Development Environment) [10]. It is an open source framework from TILab (Telecom Italia labs) and is Foundation for Intelligent Physical Agent (FIPA)[11] compliant. The 25 NPCAs are modeled in a single JADE container whereas NPRA is implemented as a separate agent belonging to a different container. The interaction model of the two architectures implements five of the JADE performatives.

III. SIMULATION RESULTS AND ANALYSIS

For testing the reactivity of both MAS architectures, the traffic was increased from 10 to 100 Erlang in steps of 10 and the corresponding \(T_{\text{RTY}}\), i.e. the time taken for architectures to react, was noted. Figure 1 shows that DSA was more reactive, as time \(T_{\text{RTY}}\) was less as compared to that of centralized architecture, with the load increase. The percentage difference of the reactivity between the two architectures varied from 13% to 15%.

![Figure 1. Reactiveness of Multi Agent Architectures](image-url)
against the carried load. Figure 2 shows how the utilization of resources declined with the increase in offered traffic. The results were same as the call acceptance probability for each cell. The utilization of resources in DSA was better for higher loads but remained same as that of centralized architecture for low loads.

![Utilization of Resources](image)

**Figure 2. Carried Load v/s Offered load**

Utilization of Resources

<table>
<thead>
<tr>
<th>Traffic (Erlang)</th>
<th>Centralized SA</th>
<th>Distributed SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>0.9</td>
<td>0.8</td>
</tr>
<tr>
<td>100</td>
<td>0.8</td>
<td>0.7</td>
</tr>
</tbody>
</table>

Scalability

<table>
<thead>
<tr>
<th>No. of cells (NPCA)</th>
<th>Centralized SA</th>
<th>Distributed SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>0.9</td>
<td>0.8</td>
</tr>
<tr>
<td>135</td>
<td>0.8</td>
<td>0.7</td>
</tr>
</tbody>
</table>

**Figure 3. Reactivity v/s No of cells**

To test the scalability of the architectures, the number of cells was increased from 25 to 135 in clusters of 5 and the reactivity of both the architectures was measured. As seen in Figure 3 the time $T_{RV}$ by both the architectures increased almost linearly with increase in the number of cells. The CSA could not scale in terms of reactivity as time taken by it to react with increase in number of cell agents beyond 115 was very high. This was due to the bottleneck at NPRA whereas DSA performed relatively well.

Communication overhead in terms of number of messages per call remained constant at 9 messages for DSA as compared to only 12 messages for centralized architecture. It depended on various factors such as periodicity, round trip time of the communication that depended on the distance between the agents. As the NPCA cluster was a cluster of peer agents, they were present in the same container and thus, this localized nature of agents in DSA resulted in better reactivity.

The sustainability of the two architectures under high traffic conditions was tested by increasing traffic. The handoff call blocking probability was chosen as 0.055 towards QoS. It was observed that the DSA could sustain more traffic load as compared to CSA for the same handoff call blocking probability.

**IV. CONCLUSION AND FUTURE WORK**

MAS based NPCA cells are self-contained. The autonomous feature of these cells increases scalability and also robustness of the system.

The results presented here point out the impact of each of the Multi Agent architectures on different types of QoS parameters. DSA reduces the response time, which in turn increases service availability, guarantee timely QoS and increase utility, whereas simplicity, modularity, and modifiability are achieved by using centralized service architecture. Thus the work highlights the effect of degree of distribution of agent in MAS service architecture on the system performance parameters for call admission control for cellular system.

Integrating congestion control schemes for cellular networks will further explore the full benefit of Multi Agent Systems. The multi agent interaction is of coordination type, since all the cell agents agree on distributed cooperation for D-CAC. One could explore QoS based negotiations amongst the agents.

**REFERENCES**


Resource Management for Cellular Network using Socially Intelligent Multi Agent System

Mamal Karani  
Computer Engineering Department,  
VESIT,  
Mumbai, India.

Nupur Giri  
Computer Engineering Department,  
VESIT,  
Mumbai, India.

Shrikant Bodhe  
Principal, SVERI’s College of Engineering,  
Pandharpur,  
Maharashtra, India.

Abstract - Channel borrowing schemes are used to increase the utility of the system in terms of cell-level quality of service parameters and reduce congestion in cellular system. Multi Agent based interaction helps in doing so in coordinated ways. This paper investigates the effectiveness of Multi Agent based social reasoning strategies, which not only maximize the utility but also balance utility of individual cells and thus work towards enhancing the social welfare of the network.

Keywords—Channel Borrowing, Multi Agent System, Social welfare

I. INTRODUCTION

Utility functions describe satisfaction level of the user with the perceived Quality of Service (QoS); the higher the utility, the more satisfied is the customer [1].

The goal of every agent in MAS is to maximize its own utility. However, this decision of the agent has social implications too. Many a times, overall utility of the system increases but fails to maximize the utilization of resources at every node and thus is not fair to the individual node. Socially intelligent agents (SIA), belonging to MAS aim at maximizing the utility of the system by balancing the utility of individual cell with effective utilization of resources [2] of every agent. Thus agents make decision based on how their increase in individual utility increases the overall utility of the society.

For a cellular system, fixed channel assignment schemes provide lesser utility due to congestion caused by unavailable channels for handoff calls, whereas dynamic channel borrowing schemes have been designed [3,4] to increase the utility. We attempt to apply social intelligence for the domain of dynamic channel borrowing as a congestion control strategy to measure the effect of different attitudes of agents on fairness of resource distribution.

Section 2 reviews the previous work, section 3 presents the problem in hand. Section 4 proposes the solution to the problem by suggesting social intelligent interaction. Section 5 presents the simulation model. Section 6 presents the simulated results with its analysis followed by conclusion in next section.

II. SOCIAL WELFARE: THE GENERIC STRATEGY

In a social environment, the agent can have different weightage for its own utility and society’s utility which characterizes its attitude [5]. The effect of attitudes on exchange of resources is detailed as follows:

Let \( w_i \), \( w_{soc} \) be the weights given by the agents to its own utility and system utility respectively. Let \( U_i \) be defined as the utility function of cell \( i \). Let \( U_k^{'} \) define the utility function of cell \( k \) as expected by cell \( i \). Then,\

\[
W(ask_n) = w_i \times U_i(ask_n) + w_{soc} \times \sum_{k=1}^{4} U_k^{'}(ask_n)
\]

(1)

Where \( W(ask_n) \) represents the social welfare function of the cell \( i \) asking cell \( n \) for its resources.

III. SOCIAL WELFARE BASED RESOURCE MANAGEMENT

Fixed channel allocation strategy bounds a cell to fixed number of channels. With a sudden increase in handoff call traffic, the cell has to block calls due to unavailability of free channels whereas its neighboring cells experiencing low traffic will have some channels being unutilized. Thus overall system’s utilization of resources decreases.

A solution to this problem is to use channel borrowing schemes. In a simple borrowing scheme, an agent which has maximum number of free channels donates a channel to the requestor cell. This is done without the knowledge of the effect of donation of a channel on the donor cell, i.e. how the donation of the channel affects the utility of the cell itself. By adding social intelligence to the agents we can get maximum utilization of the resources with balance in resource distribution.

We model 5-cell cluster as MAS where agents interact for channel borrowing. All cell agents are motivated towards the goal of reducing \( p_{ab} \). We have defined five MAS
performatives for channel borrowing scheme as presented in table 1. The finite state diagram shown in figure 1 and 2 illustrate the state transition of borrower and donor cell agents.

<table>
<thead>
<tr>
<th>TABLE 1: PERFORMATIVES FOR CHANNEL BORROWING</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Performatives</strong></td>
</tr>
<tr>
<td>QUERY REF</td>
</tr>
<tr>
<td>INFORM</td>
</tr>
<tr>
<td>PROPOSE</td>
</tr>
<tr>
<td>ACCEPT PROPOSAL</td>
</tr>
<tr>
<td>REJECT PROPOSAL</td>
</tr>
</tbody>
</table>

Figure 1. Finite State diagram of Borrower Agent

Cell agent undergoing congestion enters into ‘Congestion’ state. QUERY REF is sent to the neighboring cells asking for the number of channels they possess. Reply is received in the form of INFORM. The borrower cell agent calculates the social welfare function for borrowing a channel from each of its neighbors. This is done by estimating the utility of all its neighbors if a channel is borrowed from a particular neighbor cell agent. If an action results in a higher utility, then that action is selected and a PROPOSE is sent to that neighbor. It is received, channel is borrowed, congestion is resolved and an ACK is sent back, else the call is blocked.

Figure 2. Finite State diagram of Donor Agent

The donor cell dwells in the ‘CAC’ state. Reply to QUERY REF is sent with an INFORM with the required information. If it receives a PROPOSE, it either sends ACCEPT PROPOSAL and donates the channel, or it sends REJECT PROPOSAL. If the channel is donated, the donor cell waits for an ACK and then resumes back to its work.

These agents are implemented and simulated in JADE 3.1 (Java Agent Development Environment)[6], an open source framework and is Foundation for Intelligent Physical Agent (FIPA) compliant.

IV. SIMULATION MODEL

Manhattan model of a city consisting of square blocks with streets in between is used with 25 base stations modeled as cell agents. Single class (voice) traffic is considered. Average new call arrival rate follows Poisson distribution. The call holding time is assumed to have exponential distribution with mean as 120s. To eliminate the edge effect, the system is wrapped around at the edges. Normal Distribution speed (v ∈ [30km/h, 90km/h]) is assumed. The motion of the mobile is restricted along the streets, and can move to left, right as well as top, bottom streets. The mobile user moves with uniform speed throughout the call direction and does not change the direction during that ongoing call.

Fixed channel allocation is assumed for evaluation of multi agent architecture but 2-D Dynamic Call Admission control scheme (D-CAC) [7] is chosen for call admission, i.e. a new call in the cell is admitted in the cell depending on dynamic cutoff threshold determined by the ongoing calls in the neighboring cells. This neighborhood cluster is made of right, left, top and bottom agent cells of Manhattan model. The system consists of 25 cells and initially each cell is allocated 30 channels. In time of need, cells can negotiate with each other and borrow channels. We have restricted the number of channels that can be borrowed or loaned to 3, i.e. the channels in a cell at any instant will be between 27 and 33. Decision of handoff call blocked or a channel borrowing is taken by the agent. Simulations were carried out for traffic from 10 to 100 Erlang increased in steps of 10.

V. SIMULATIONS AND RESULTS

Figure 3 shows the handoff call blocking probability of varied attitudes of SIA. The self biased attitude of the SIA makes the agent unable to borrow channel as there is no agent to accept the request because of their selfish nature. This leads to SIA based non-Channel Borrowing (CB) scheme. It can be seen that $P_{bh}$ is reduced greatly for SIA-CB as compared to SIA-non CB.

Figure 3. Utility ($P_{bh}$) of DCA vs FCA (using Social Welfare)
This is because SIA based CB exchanges channels amongst cells such that the $P_{ah}$ is kept as minimum as possible. Cells are intelligent to take a decision to ask that neighboring cell which has the highest probability of donating the channel when congestion occurs.

Figure 4. Utility($P_{ah}$) of System with SIA-DCA

Figure 4 and 5 depicts the effect of varied attitudes of SIA based CB on $P_{ah}$ and $P_{nb}$ of the system. The balanced attitude gives lower $P_{ah}$ compared to partially society or partially self biased agents whereas $P_{nb}$ increases with reduction in $P_{ah}$.

This is because partially self biased agents try to gain extra channels whenever there is slight increase in its utility. But, since other agents are also partially self biased in the environment, they are reluctant to donate their extra channels and reserve them for future unexpected crisis. Partially social biased agents are ready to donate their channel at all times. Every agent waits for a request from other agents for their channel. When congestion takes place, social welfare functions are calculated and a channel is asked to its neighbor only if there is a significant increase in its own utility.

Figure 5. Utility ($P_{ah}$) of System with SIA-DCA

Fairness can be demonstrated by measuring the effect of attitudes on the utility of the system verses utility of agent. As the mean of utility in each case is different, the difference from the mean is considered. Thus, standard deviation of this difference is shown in figure 6.

We observe that the fairness towards the individual cell agents is increased for balanced attitude as standard deviation is reduced by nearly 90% as compared to self biased attitude. Whereas 61% reduction for partially society biased and 56% reduction for partially self biased attitudes is observed. This shows that SIA balances the utility of each cell and thereby increases the overall utility of the system.

VI. CONCLUSION AND FUTURE WORK

The work establishes the relationship between different social attitudes of the agents towards the fairness of resource distribution for the domain of cellular networks. The work has been modeled towards dynamic channel borrowing but the concept can be extended towards any domain requiring distributed negotiations.

The applicability of the concept for social welfare with varied degree of distribution of attitudes, amongst the agents in MAS is yet to be studied.

REFERENCES


Appendix C

Towards Incorporating the Comments

C. 1 Comments/Queries by Dr. Abhay Karandikar:

The author may consider the following comments/questions at the time of viva-voce or incorporate them in the final archival copy of the thesis.

A) The does not motivate the reader for implementing agent based system in cellular network and how this framework will be effective against what is already implemented.

B) The author has not made it clear whether a new layer will be required to be introduced for various protocol function in a typical base station protocol stack or whether it is a fundamental extension of “Shuffle” which may be addressed other functionality.

C) It is not clear what is authors contribution in chapter 4.1.

D) It is not clear how does this framework help the design of a practical cellular system.

E) Literature survey could be shortened and applications of agent based cellular system in cellular system may be included.

C. 2: Answers/Justifications

Answer D: Rationale behind Shuffle Model and integration with practical

The Shuffle project aims to create a novel architecture for efficient, scaleable and robust real time control of third generation mobile systems in the context of realistic business models of network providers, service providers and customers and the relationships between these actors.

This requires that scalability issues are addressed directly and. Successful solution will bring financial benefits to mobile users through improved efficiency, but extensions to the same approach would allow management of a variety of contingencies that could have social benefits.

1
The Shuffle Project basically models RNC agent as NRPA agent and its functionality are integrated with the practical 3G cellular System. To handle congestion NRPA agent can take following cell level actions:

**Cell-scale operations**

- At call select, choose a different type of cell, probably providing a lower QoS to the user. (For instance, choosing a macro-cell rather than a pico-cell, which would provide lower bitrate to the application).
- Forcibly handover a connection to a different type of cell (using a different frequency band), again with QoS implications.
- **Offload the call to an overlapping GSM/GPRS network owned by the same operator.**

*(Here is where the research gets integrated with Shuffle)*
- Degrade the QoS of a connection without use of handover.

Answer A: Motivation for the research Work

However Shuffle Model has following limitations:

1. It only offered the hypothesis that the agents could control SLAs at cell level. Performance evaluation of the MAS for call admission control was absent.
2. The QoS guarantee was limited to only Resource Plane, not making it granular at cell level.
3. The Planning Layer of the hybrid agent did not accommodate new traffic patterns or real time traffic.
4. SP-NP negotiation for NP selection though explained in detail, NP-Cell interaction and Cell-Cell interaction were missing.

The work proposed in this thesis extends 'Shuffle' project, and aims at addressing the issues mentioned above. The sub goals of the research are as follows:-

1. To implement cell level SLA-QoS provisioning, a new cell/connection level layer needs to be introduced which should handle call admission control, complete with channel borrowing. This layer should be fully compatible with the 'Shuffle' model and agents modeled in this layer should interact with existing agents through vertical interaction.
2. The Multi Agent based Call Admission Control (MA-CAC) policies in Local Planning Layer, should consider different classes (voice/data) of traffic and mobility patterns (high/low) of mobile user along with priority and non-priority handoff traffic.
3. The research should present performance evaluation of MA-CAC strategies, in terms of connection level parameters such as new call blocking probability, handoff call blocking probability, effect of queue on blocking probability.
4. Congestion control through channel borrowing strategies should demonstrate the effectiveness of using socially intelligent agent based MAS.
5. The work should establish the effect of degree of distribution of agents in MAS based interaction in terms of performance parameters.

Answer B: The connection layer proposed is not a fundamental extension of shuffle but has been proposed to ensure faster reactivity (refer to answer C) at base station level, which was not part of the work carried out in Shuffle. (also refer to the answer A: Motivation for the research work)

Answer C: Contributions of section 4.1

The analytical models presented in section are used for validation, and verification. This section presents the simulation results of Multi Agent Call Admission Control (MA-CAC) strategies for single and multi class (non-priority and priority) traffic models. The four MA-CAC strategies namely: Static cutoff priority (S-CAC), Dynamic cutoff priority (D-CAC), Integral Mobility based channel reservation (IMBCR-CAC) and Fractional Mobility based channel reservation (FMBCR-CAC) schemes are simulated for single class traffic. And Static cutoff priority (S-CAC), Dynamic cutoff priority (D-CAC) are simulated for multi class traffic in Priority (P) as well as in Non-Priority (NP) mode of traffic model using multi agent environment i.e. using JADE. These MA-CAC strategies are presented for Centralised (C) as well as Distributed (D) Service Architecture.

The results measure the reactive estimates of Non agent Based and Multi Agent Based systems as shown in table 1.

<table>
<thead>
<tr>
<th>Number of Calls</th>
<th>Non agent Based</th>
<th>Multi Agent Based</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C_NA_ S-CAC</td>
<td>C_NA_ D-CAC</td>
</tr>
<tr>
<td>25000</td>
<td>194.34</td>
<td>1694.58</td>
</tr>
<tr>
<td>50000</td>
<td>371.28</td>
<td>3372.50</td>
</tr>
<tr>
<td>75000</td>
<td>728.82</td>
<td>5120.25</td>
</tr>
<tr>
<td>100000</td>
<td>869.20</td>
<td>6815.7</td>
</tr>
<tr>
<td>125000</td>
<td>1224.5</td>
<td>8740.68</td>
</tr>
</tbody>
</table>
### Table 2 Simulation Time (seconds): Analysis
Comparison of MA based D-CAC for different service architectures

<table>
<thead>
<tr>
<th>No. of Calls</th>
<th>Dynamic schemes (D-CAC)</th>
<th>D MA (D-CAC)</th>
<th>Appr. % reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C MA (D-CAC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25000</td>
<td>1301.40</td>
<td>983.57</td>
<td>24</td>
</tr>
<tr>
<td>50000</td>
<td>2584.57</td>
<td>1928.92</td>
<td>25</td>
</tr>
<tr>
<td>75000</td>
<td>3977.56</td>
<td>3075.95</td>
<td>23</td>
</tr>
<tr>
<td>1,00,000</td>
<td>5302.04</td>
<td>3992.45</td>
<td>25</td>
</tr>
<tr>
<td>1,25,000</td>
<td>6682.01</td>
<td>4986.95</td>
<td>25</td>
</tr>
</tbody>
</table>

### Table 3 Simulation Time (seconds): Analysis
Comparison of centralised architecture D-CAC for agent based and non agent based

<table>
<thead>
<tr>
<th>No. of Calls</th>
<th>Centralised Dynamic scheme (D-CAC)</th>
<th>D MA (D-CAC)</th>
<th>Appr. % reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C NA (D-CAC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25000</td>
<td>1694.58</td>
<td>1301.40</td>
<td>23</td>
</tr>
<tr>
<td>50000</td>
<td>3372.50</td>
<td>2584.57</td>
<td>23</td>
</tr>
<tr>
<td>75000</td>
<td>5120.25</td>
<td>3977.56</td>
<td>22</td>
</tr>
<tr>
<td>1,00,000</td>
<td>6815.72</td>
<td>5302.04</td>
<td>22</td>
</tr>
<tr>
<td>1,25,000</td>
<td>8740.68</td>
<td>6682.01</td>
<td>23</td>
</tr>
</tbody>
</table>

### Table 4 Simulation Time (seconds): Analysis
Comparison of MA and NA based D-CAC for different service architectures

<table>
<thead>
<tr>
<th>No. of Calls</th>
<th>Dynamic schemes (D-CAC)</th>
<th>D MA (D-CAC)</th>
<th>Appr. % reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C NA (D-CAC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25000</td>
<td>1694.58</td>
<td>983.57</td>
<td>42</td>
</tr>
<tr>
<td>50000</td>
<td>3372.50</td>
<td>1928.92</td>
<td>43</td>
</tr>
<tr>
<td>75000</td>
<td>5120.25</td>
<td>3075.95</td>
<td>40</td>
</tr>
<tr>
<td>1,00,000</td>
<td>6815.72</td>
<td>3992.45</td>
<td>42</td>
</tr>
<tr>
<td>1,25,000</td>
<td>8740.68</td>
<td>4986.95</td>
<td>43</td>
</tr>
</tbody>
</table>

- Tables 1 and 2 give us an estimate of the reactiveness of MA-Based CAC over non agent based CACs for Centralized as well as Distributed SLA architectures. We see that by using Multi agent based schemes there is a significant reduction in simulation time, approximately of 22-23% (table 3 if used centralized architecture and and
around 42% (table 4) if we used Multi agent based distributed D-CAC. The Distributed D-CAC performs better (22%) as compared to centralized D-CAC (table 2) if compared even in multi agent environment alone. The Distributed architecture scheme (D-MA-D-CAC) using Multi Agent schemes is faster as compared to the Centralized Non Agent based (C_NA-D-CAC) by 42% as well as Centralized Multi Agent based schemes (C_MA-D-CAC).

- Although Message overhead in table 5 is higher in Distributed Agent based as compared to Centralized Agent based architecture but again it is less by 16-17% in D-CAC as compared to S-CAC.

Contributions:

The above results contribute towards proving that agent based systems are much more reactive as compared to no agent based system. This in effect contributes towards NP selection and could be considered as a parameter for SLA negotiation. The centralized verses distributed service architecture response time evaluation (section ) also contribute that Agent based Base station implementation is much more reactive and thus efficiency of the system in greatly increased by introducing agent based connection level layer. This parameter was not considered in Shuffle Project. This is purely our contribution.

Answer E: Applications of Multi Agent System for cellular Network

1. Automation of the task using goal oriented (BDI) modeling
2. For self organizing cellular system
3. For self Recovery cellular network (Provide Robustness)
4. For Market based prizing model for increasing competitiveness
5. Swarm intelligence based MAS could be used better routing characteristics
6. MAS of Mobile Agents could be used to ensure less bandwidth usage

Etc.