Overload Control for the Wireless Intelligent Network

Zuojie Deng 1, 2* and Xiaolan Tan 2
1. School of Computer, Huazhong University of Science and Technology, Wuhan, Hubei, 430074, P. R. China
2. School of Computer and Communication, Hunan Institute of Engineering, Xiangtan, Hunan, 411104, P. R. China
*Corresponding Author, Email: zuojiedeng@gmail.com

Abstract—Wireless intelligent network (WIN) is a network in the mobile environment that supports the use of intelligent network capabilities which is mainly made up of a service control point (SCP) and some mobile switching centers (MSCs). When there are too more incoming services to SCP and MSCs in the WIN simultaneously than they can handle will cause the SCP or MSCs overload and degrades the performance of the WIN. To resolve this problem, we introduce a queue model, design an overload control model and put forward a two-level overload control algorithm for the WIN. The results of our analysis and experiments show that our overload control algorithm is effective.

Index Terms—Wireless Intelligent Network; Overload Control; Algorithm; Queue Theory

I. INTRODUCTION

Wireless Intelligent Network (WIN) is a communication network in the mobile environment that supports the use of intelligent network capabilities to provide some advanced services, such as personal mobility, seamless terminal, location based services, and so on [1-3]. Fig. 1 gives an example for the wireless intelligent network [1]. WIN is mainly made up of a service control point (SCP), some mobile switching centers (MSCs), some signal transfer points (STPs) and some intelligent peripherals (IPs). SCP is the centralized equipment that controls service processing and service delivery in the network. MSC provides switching function of the WIN. IP provides the functions that get information and translate it to data that can be used by the other elements in the network. STP is a packet switch that handles control signals between different equipments in the network, such as MSCs and HLRs or MSCs and SCPs.

In the WIN, SCP is the centralized node that controls service processing and service delivery. All the service logical programs and user data are stored in the SCP. When an intelligent network (IN) service come to the WIN, SCP handles the service by executing a service logical program corresponding to the service and exchanges some messages with MSCs in the WIN [3]. Note that in order to describe conveniently, we will use service and call interchangeable in the following section. If there are too many incoming services to the SCP simultaneously than it can handle, the SCP will become overload. When the SCP is overload, the IN service setup time will become long and the throughput of SCP will decrease much. When this status becomes serious, the SCP will suffer from shutdown, which will make users in WIN cannot use their IN services. Therefore, when SCP becomes overload, we must use some overload control methods to protect SCP and prevent its performance from decreasing significantly. To resolve the overload control problem of SCP in the traditional wired intelligent network, some researchers have designed some overload control algorithms or mechanisms [4-5, 7-14]. Pham. X.H. etc. found that appropriate congestion control in the special IN aspects is required, and performed a comparative study of the effectiveness of the call rate based control using the call gapping principle with the window method [7]. Smith. D.E. analyzed the performance of two different types of service control point (SCP) overload control algorithms, table-driven controls and an adaptive control, and found that the adaptive control is more robust to traffic patterns in SCP [10]. Kawahara. R. etc. proposes an overload control for intelligent networks based on an estimation of maximum number of calls in a node [14], but due to the randomness of the incoming calls, it is very difficult to estimate the maximum number of calls exactly. Kihl, M. etc. believed that all nodes in IN must have an overload control mechanism which protects the node by rejecting calls when the load on the node is too high, and found when the rejecting mechanisms are placed in the SSPs instead of the SCPs will achieve an efficient control[5]. They also design a ticket cache control mechanism which can control the incoming call traffic easily. But they only investigates overload control methods for the SCPS and not for the other nodes. Northcote. B. S. etc. analyzed implementations of automatic code gapping (ACG) in intelligent network SCPs and found that using the processor utilization measurements can yield reasonable...
call throughput [4]. Prouskas. K. etc. presented a multi-agent system, which makes use of a market-based approach to perform real-time control of IN traffic, but the agents in their multi-agent system are hard to be implemented [9]. Polykalas. S. E. etc. proposed an adaptive congestion control mechanism (ACCM) for the broadband service control point (BSCP) which provides a novel multimedia services in broadband networks [8], but its mechanism utilizes the feedback of the buffer capacity state in the destination will cause the destination node overload. Zhang. Q. Z. etc. proposed a self-adapted overload control algorithm that be implemented in Parlay AS [11], but his estimating the queue waiting time of a newly arriving message to detect the overload status cannot coincides with actual overload situation. Jabban. A. etc. brought forward an algorithm that give the optimal distribution of traffic arriving from the local exchanges on SSP nodes in order to have a minimum number of signaling relations [13], but his algorithm cannot eliminate the overload status in the SCP completely. Akyamac. A. A. etc. presented an overload control simulation tool which can be used to analyze the efficiency and operation of different overload control mechanisms in IN [12], but its simulation result cannot agree with real overload status exactly.

MSC provides switching function of the WIN, if a incoming call is an ordinary service, MSC will handle it locally without any interacting with the SCP. When there too much service call traffics to the MSC than it can handle, it will become overload as well. When the MSC become overload, the performance of the WIN will go down too. At the same time, the bandwidth of the WIN is expensive [16]. All traditional SCP overload control algorithm in the literature did not consider the overload in MSC, if there are too much interactions with MSCs when SCP become overload, it make the WIN congestion, and decline its performance. So all the traditional SCP overload control algorithms for IN cannot use for WIN directly. To resolve this problem, we must design a new overload control algorithm to prevent the SCP and MSCs from overload in the WIN at the same time.

In this paper, we study the problem of how to design an overload control algorithm to prevent the SCP and MSCs from overload at the same time. We summarize our contributions as follows: Firstly, we introduce a queue model for the wireless intelligent network. Secondly, we design overload control model for the WIN. Thirdly we put forward a two-level overload control algorithm for the WIN based on the overload control model. Fourthly, we analyze the validity of our overload control algorithm. Finally, we have evaluated the performance of our proposed overload control algorithm by some experiments.

The rest of the paper is organized as follows. We put forward a queue model for the wireless intelligent network in section 2. We design the overload control model in section 3. We design a two-level overload control algorithm in section 4. We analyze the validity of our overload control algorithm in section 5. Then, we evaluate the performance of our overload control algorithm in section 6. Finally, we give some conclusions and the future works.

II. A QUEUE MODEL FOR THE WIRELESS INTELLIGENT NETWORK

MSC is a key node in the WIN that provides switching function. Generally, there are three modules in MSC, that is to say authentication module, number analysis module and routing module [3]. When a service arrives at MSC, the authentication module will authenticates this service firstly. If it doesn’t pass this authentication process, it will be rejected; otherwise, it will be set to the number analysis module to analyze its service type. If the number analysis module finds that it is an ordinary service, it will send this service to the routing module in the MSC to route this service, and the service has been accomplished in the MSC; otherwise if it is an intelligent network service, MSCs will request the SCP to handle it. After receiving this request, the SCP will execute its corresponding service logic program to implement this service. After the service logic program have executed completely, the intelligent network service will be finished or be sent to the routing model in MSC to route this service.

From the above analysis, we know that the ordinary service is handled only in the MSC, but the IN service must be handled by the MSC and SCP together. According to Ref [16], we know that the speed for the service reaching MSC obey the poisson distribution, and we can regard all these modules in MSC and SCP as a stochastic service system respectively [6], and we can get a queue model for WIN which is shown in Fig. 2. The Q_i1 denotes the authentication module, Q_i2 denotes the number analysis module, Q_i3 denotes the routing module, and Q_{scp} denotes SCP. From Fig. 2, we know that the handling process of IN service in the WIN is Q_{i1} \rightarrow Q_{i2} \rightarrow Q_{scp} \rightarrow Q_{i3}, the handling process of the ordinary service is Q_{i1} \rightarrow Q_{i2} \rightarrow Q_{i3}. Note that in the following section, we assume that there are N MSCs in the WIN, M kinds of service. We use a number between 1 and L to denote the IN service, and use a number between (L + 1) and M to denote the ordinary service, where M > L, N > 1.

III. THE OVERLOAD CONTROL MODEL FOR THE WIRELESS INTELLIGENT NETWORK

Kihl. M. etc. [5] have designed a ticket cache control mechanism, which is made up of a limited cache. In each control interval K, every second is divided into N_t interval, and it generates a ticket to a limited ticket cache
in each end of the interval. Obviously, the number of the ticket is the maximum number to accept the services. When a service arrives, if the ticket cache is empty, it will be rejected, otherwise, it will be accepted, and the mechanism deceased the number of the ticket with one. If we control the generating speed of the ticket, we can control the speeds of the service to pass the ticket cache mechanism.

From the discussion in section 2, we know that a service call can be distinguished after it is sent to the number analysis module. So we can introduce some ticket cache control mechanisms for every kind service in the MSC, and put it after the number analysis module. In the following section we will call ticket cache control mechanisms as service ticket cache. When the wireless intelligent network becomes overload, we use this service ticket cache to control the service process speed in the MSC and SCP, and we get an overload control model for the wireless intelligent network. In our overload control model, we detect the overload status for the MSCs and SCP respectively by detecting their CPU occupancy ratio periodically. If CPU occupancy ratio in SCP is greater than its maximum value, the MSCs will limit intelligent network services; if the CPU occupancy ratio in MSCs is greater than its maximum value, the MSCs will limit all his arriving services.

IV. A TWO-LEVEL OVERLOAD CONTROL ALGORITHM

In section 3, we design an overload control model for the wireless intelligent network. In this section we design a two-level overload control algorithm according to the overload control model in section 3. It is made up of a SCP control algorithm and a MSC control algorithm, these algorithms send control information flow to accomplish the overload control in the wireless intelligent network jointly. The SCP algorithm determines the ticket number of each intelligent service for the service ticket cache control mechanism in the next control interval based on its current load. After receiving the control command from the SCP, the MSC algorithm determines the tickets of all the services according to its current load status. As different service will generate different profits for the service provider, the service provider wants to setup the ratio to be accomplished to maximize his profit when the wireless network becomes overload. He would like to make his wireless network to accomplish some high profitable services. In the two algorithms, the service provider can set up the service ratio parameters in the control interval.

A. The SCP Overload Control Algorithm

Step 1. If SCP detects that it is in overload, SCP sends notification command to MSCs that all the MSCs must make all the ticket of the IN services to be zero in the next T seconds. After the T seconds, let k = 1, and ask all the MSCs to send the number of IN service \( N_i(k - 1) \) to SCP, and calculates \( N_i(k - 1) \), where \( i \) \( (1 \leq i \leq M) \) denotes the identity of the service, \( j \) \( (1 \leq j \leq N) \) denotes the identity of the MSC.

Step 2. SCP calculates all the ticket number of service \( i \), \( N_i^*(k) \) \( (1 \leq i \leq M) \) in the next control interval \( k \) for all the IN services according to formula (1). In the formula (1), \( i \) \( (1 \leq j \leq M) \) denotes the IN service \( i \), \( j \) \( (1 \leq j \leq M) \) denotes the IN service \( j \), \( u_i \) \((1 \leq i \leq M)\) denotes the speed for SCP to accomplish the intelligent network service \( i \). \( p \) denotes the maximal CPU process capability. \( W_j(1 \leq i \leq M)\) denotes the ratio for intelligent network service \( i \), and \( W_i \) \((1 \leq j \leq M)\) denotes the ratio for intelligent network service \( j \).

\[
\begin{align*}
\sum_{i} N_i^*(k) = p^* \\
N_i^*(k) \leq \frac{W_i}{u_i} \quad (i \neq j) \\
N_i^*(k) \leq N_i(k - 1)
\end{align*}
\]

Step 3. SCP calculates \( N_{i,scp}^*(k) \) which is the ticket number for IN service \( i \) in MSC \( j \) in the next control interval \( k \) according to formula (2), and sends \( N_{i,scp}^*(k) \) to MSC \( j \) by an active service filtering information flow. In the formula (2), \( y_i \) denotes the ratio for the IN service \( i \).

\[
\begin{align*}
\sum_{i} N_{i,scp}^*(k) = N_i^*(k) \\
\frac{N_{i,scp}^*(k)}{\sum_{i} N_{i,scp}^*(k)} \leq y_i \\
N_{i,scp}^*(k) \leq N_i(k - 1)
\end{align*}
\]

Step 4. After \( T \) seconds, SCP determines whether finish this control. If the control is over, SCP informs all the MSCs to finish the overload controls for these IN services, and the algorithm end; otherwise, all the MSCs send \( N_{i,scp}^*(k) \) to SCP and SCP calculates \( N_i(k) \). Let \( k = k + 1 \), and jumps step 2.

B. The Overload Algorithm in MSC

Step 1. When MSC receives the overload control command from SCP, it makes the ticket number of all the IN service to zero. If the MSC detects that it is overload, it make all the ticket number to zero. After \( T \) seconds, let \( k = 1 \).

Step 2. The MSC calculates \( N_{i}^*(k) \) for the next interval according to formula (3), and make the ticket number of the service \( i \) in the MSC \( j \) equals to \( N_{i}^*(k) \). In the following \( T \) seconds, when a service call request arrives at the MSC, if its service ticket cache has a ticket, it will accept this service, otherwise, it reject it. In the formula (3), \( i \) denotes the service \( i \), \( u_i \) denotes the speed for MSC to accomplish the service \( i \). \( p^* \) denotes the maximal CPU process capability in MSC. \( W_i \) denotes the ratio for service \( i \), and \( W_i \) denotes the ratio for service \( i \).
Step 3. After T seconds, MSC decides whether finish this control. If it wants to finish this control, the algorithm ends, otherwise, it calculates the numbers of all incoming service $N_y(k)$ in MSC, let $k = k + 1$ and jumps step 2.

Note that the algorithm in SCP is executed first, and then execute the algorithm in MSCs, and the synchronous operation is executed by SCP.

V. THE PERFORMANCE ANALYSIS.

In this section, we use the queue theory to analyze our overload control algorithm. We first estimate the service arrival speed, throughout, and the average delay for each queue in our model, then we use these results to prove that our algorithm can control the overload in the SCP and SSP.

A. The Service Arrival Speed and Throughput for Each Queue

Support the arrival speed of service $k$ to the MSC obeys the poisson distribution with parameter $\lambda_k(t)$. According to the nature of poisson process and the service execution flow in our model, we can get the service arrival speeds for all queues in our mode, which are described in formula (4), (5) and (6) respectively. In the formulas (6), $p^r_k(t)$ denotes the possibility of the service $k$ to pass the ticket cache control mechanism in the MSCs in the control interval $t$.

$$\lambda_{01}(t) = \lambda_{01}(t) = \lambda(t) = \sum \lambda_k(t)$$

$$\lambda_{0p}(t) = \sum_{i=1}^N \sum_{k=1}^L p^r_k \lambda_k(t)$$

$$\lambda_{0i}(t) = \sum_{i=1}^N p^r_i \lambda_i(t)$$

As the service time in the $Q_{01}$, $Q_{02}$, and $Q_{03}$ obeys the exponential distribution with parameters $u^{-1}_{01}$, $u^{-1}_{02}$, and $u^{-1}_{03}$ respectively, we can regard $Q_{01}$, $Q_{02}$, and $Q_{03}$ as M/M/1 queue model respectively and we denote their throughout as $\rho_{01}(t)$, $\rho_{02}(t)$, and $\rho_{03}(t)$ respectively. According to the nature of the Poisson arrival process, we know that the service arrival process in the SCP also obeys the poison distribution. As different service is executed by different service logical programs and has different service time. Therefore, we can regard the queuing model of $Q_{SCP}$ as M/G/1, and denotes its average service time as $u^{-1}_{SCP}$ and its throughput as $\rho_{SCP}(t)$, and then we can get their queue throughputs as formula (7), (8), (9) and (10) respectively.

$$\rho_{0i}(t) = \frac{\lambda_{0i}(t)}{u_i}$$

$$\rho_{01}(t) = \frac{\lambda_{01}(t)}{u_1}$$

$$\rho_{02}(t) = \frac{\lambda_{02}(t)}{u_2}$$

$$\rho_{03}(t) = \frac{\lambda_{03}(t)}{u_3}$$

B. The Average Delay for Each Queue

From Fig. 2, we can see that the queues in our overload control model consist of a queue network, so we can use the decomposition method to approximate the queue average delay [15]. The decomposition method utilizes the idea of divide and conquer, the basic steps are as follows:

Step 1. Decompose the network into some appropriate subnets, some of which may be separate service stations.

Step 2. Solve each subnet or each service station independently.

Step 3. Gathered the solutions of each subnet or each service station to approximate the solution of the entire network.

Using the decomposition method, the average waiting time formulas of M/M/1 and the average waiting time formulas of M/G/1, we can decompose our overload control model, and gain the average delay of $Q_{01}$, $Q_{02}$, $Q_{03}$ and $Q_{SCP}$ respectively, which is described in formulas (11), (12), (13), (14) respectively.

$$E(D(t))_{0i} = \frac{\rho_{0i}(t)}{\lambda_{0i}(t)(1 - \rho_{0i}(t))}$$

$$E(D(t))_{01} = \frac{\rho_{01}(t)}{\lambda_{01}(t)(1 - \rho_{01}(t))}$$

$$E(D(t))_{02} = \frac{\rho_{02}(t)}{\lambda_{02}(t)(1 - \rho_{02}(t))}$$

$$E(D(t))_{03} = \frac{\rho_{03}(t)}{\lambda_{03}(t)(1 - \rho_{03}(t))}$$

$$E(D(t))_{SCP} = u_{SCP} + \frac{\lambda_{SCP}(t)\sigma_{SCP}^2}{2(1 - \rho_{SCP}(t))}$$

In the formulas (14), $\sigma_{SCP}^2$ is the service time distribution variance in SCP. The delay of service in the MSC equates $E(D(t))_{01} + E(D(t))_{02} + E(D(t))_{03}$. 
Theorem 1. The two-level overload control algorithm can control of the overload in the SCP and SSP.

Proof. From the above discuss, we know, as long as the overload control algorithm control the number of the ticket cache on the MSC, it can control the probability $p_c^k(t)$ which is service $k$ through the ticket cache $i$ and it can make both of the $E(D(t))_{scp}$ and $E(D(t))_{scp} + E(D(t))_{scp} + E(D(t))_{scp}$ less than 1 second [4] which is the permitted maximum delay time. From the description of our overload control algorithm, we know, when the wireless network becomes overload, the SCP and MCS will jointly control the probability of service through the ticket cache. In other words, the overload control algorithm can control the overload in MSC and SCP, and this completes the proof.

VI. EXPERIMENTAL RESULTS AND ANALYSIS

In section 4, we have designed a two-level overload control algorithm. In this section we will do some experiments to evaluate our algorithm. To best of our knowledge, there is no similar overload control algorithm for the WIN in the literature that can compare with our algorithm. In order to evaluate our algorithm fairly and validly, we must set up our experiment correctly and measure the right experiment performance data. According to those SCP overload control literatures that we reviewed in section 1, we know that we can measure the number of service accomplished, the CPU occupancy ratio and the average waiting time in our experiments.

Our experimental model is set up according to Fig. 2; it has one SCP and two MSCs. In our experiment, there are two kinds of intelligent service and one kind of ordinary service; we use 1, 2 and 3 to represent these services respectively. The speed of the service $i$ reaching MSG $j$ obey the poisson distribution with parameter $\lambda_i (1 \leq i \leq 3, 1 \leq j \leq 2)$, the parameters of the experiment is shown in Table 1, the handling speed of the intelligent network service 1 in the SCP is 200 Call /S, the handling speed of the intelligent network service 2 in SCP is 100Call /S; the speed for the MSG to handle the ordinary service or IN service is 100Call / S, the ticket timer of $T$ is 1S, when SCP and MSG is overload, they use its 90% processing capacity to process the incoming services. If the wireless intelligent network becomes overload, the ratio for the three services to be accomplished is $w_1: w_2: w_3 = 2:3:5$.

<table>
<thead>
<tr>
<th>Time</th>
<th>MSC1</th>
<th>MSC2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-200</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>200-400</td>
<td>20</td>
<td>30</td>
</tr>
<tr>
<td>400-600</td>
<td>30</td>
<td>40</td>
</tr>
</tbody>
</table>

Note that the unit for $\lambda_i (1 \leq i \leq 3, 1 \leq j \leq 2)$ is call/S, the unit for time is S.

We have done some experiments with our overload control algorithm for wireless intelligent network using the experimental parameters in Table 1. In our experiment, we measured the CPU occupancy of the SCP and MSC, throughput and the average waiting time for these services. The experiment results of the SCP is shown in Fig. 3, the experiment results of the MSC1 is shown in Fig. 4, and the experiment results of the MSC2 is shown in Fig. 5.

From the Fig. 3, we can see at 1-200 seconds and 400-600 seconds SCP is not overloaded, the algorithm does
average waiting time of the service in SCP is around 0.5 average waiting time. When the SCP is overload, the average waiting time in the MSC1 and MSC 2 is around 0.9.

From Figure 3a, we can see that the throughput of service 1 and service 2 is 36:60. From Figure 3b, we can see that the CPU occupancy rate of the service 1 fluctuates around 0.3, while the CPU occupancy of service 2 fluctuates around 0.6. The total CPU occupancy rate is about 0.9, which shows that our SCP overload control algorithm works efficiently.

From Figure 4, we can see that at 1-200 seconds the MSC1 is not overload and it does not limit the service, at 200-400, the MSC1 is overload, it limits all incoming service. At 400 -600 seconds MSC1 is overload due to the high arrival rate of service 1 while the arrival rate of service 2 and service 3 are relatively small, the MSC1 only limit service 1, its CPU occupancy is controlled around 0.9.

From Figure 5, we can see that the MSC2 is not overload at 1-200 seconds and 400-600 seconds, and MSC2 does not limit any service; MSC2 and SCP are overload at 200-400 seconds due to the high arrival rate of service 3, but arrival rates of service 1 and service 2 are relatively small, the MSC2 limit all service, but the limit amount of service 1 and service 2 is much less than the limit amount of service 3 and the CPU occupancy ratio of MSC2 was control around 0.9.

In the experiments, we also measure the service average waiting time. When the SCP is overload, the average waiting time of the service in SCP is around 0.5 second. When the MSC1 and MSC2 is overload, the average waiting time in the MSC1 and MSC2 is around 0.3 second which is less than the maximal allowable waiting time 1 second [4].

ACKNOWLEDGEMENTS

This work was supported in part by the Natural science fund project in Hunan province (No. 10JJ6099), the Science and technology plan project in Hunan province (No. 2010CK3029).

REFERENCES

Zuojie Deng was born in Leiyang city Hunan province, P. R. China on September 1972. He received the M.S. degree in Department of Computer Engineering from Hunan University, Changsha, P. R. China, in 2004. He is a Ph.D. candidate in the School of Computer Science and Technology at Huazhong University of Science and Technology, Wuhan, P. R. China. His research areas include network performance, network security, storage security, searchable encryption, and privacy enhanced technologies.

Xiaolan Tan was born in Hengshan city Hunan province, P. R. China on July, 6, 1973. She received the bachelor in computer science from Hunan normal University in 2004 and he is an instructor in school of computer and communication at Hunan Institute of Engineering, Xiangtan, P. R. China. Her research interests include network performance, network security and information privacy.