Combined Preemption and Queuing Policies for a Cellular Emergency Network

Jiazhen Zhou and Cory Beard
Department of Computer Science/Electrical Engineering
University of Missouri-Kansas City
Email: {jzmpc,beardc}@umkc.edu

Abstract—Preemption is a possible way to ensure emergency traffic's priority in a wireless network. But a pure preemption policy can be especially harsh for low priority calls. A way to improve this is by introducing a queuing mechanism, which means putting preempted calls into a queue so that they can resume when channels become available, as long as they do not abandon due to waiting too long in the queue. Also, to avoid calls being preempted multiple times, we bring out a scheme where any particular call can only be preempted once. The analytical results for performance analysis of these schemes are given, and based on them the performance of different schemes are evaluated and compared.

Keywords: Emergency traffic, preemption, queuing, multiple preemptions, handoff, aging process, scheduling

I. INTRODUCTION

Network congestion can happen due to a lot of reasons. In this paper, we mainly focus on disaster events as they cause congestion in wireless networks. After disaster events happen, tremendous stress is placed on networks due to the rise in traffic demand, including demand from public and emergency staff[1], [2], [3], and the network demand can be up to 10 times normal. Among the traffic demands, emergency traffic should be given special priority for saving life and property. As pointed out in [5], emergency traffic has requirements for (1) availability of channels, and (2) dependability to be able to successfully complete calls.

As studied in the admission control field, the priority can be implemented in different ways, and thus have different strengths. The strongest approach is preemption, which gives high priority calls immediate access, unless all channels are already taken by high priority ones. Also this makes admission of high priority calls virtually unaffected by an increase in low priority traffic demand. Early work for applying preemption scheme to emergency traffic is seen in [7].

A weaker approach is a queuing method, where emergency calls (and only emergency calls) are put into a waiting queue, and get access to channels once they are free. Work similar to this is seen in [9], [10], [11], where handoff traffic is treated as high priority.

The even weaker approach is to put both emergency calls and public calls into separate queues, and schedule the queues according to different scheduling schemes. An example for this is in [6] where both emergency traffic and public handoff traffic are put into queues, and scheduled according to a weighted earliest deadline scheme. Another example from real world application is the Public Use Reservation with Queuing All Calls policy (PURQ-AC) [4]. Here both emergency calls and public calls are queued, and upon a call’s departure, the queues are scheduled in a way similar to a round-robin algorithm. This ensures that when congestion happens, a recommended 1/4 schedule is used so that emergency users can only take about 25% of the channels while the other 75% of the channels are taken by public users. The purpose of this policy is to avoid starving public users, but it does not differentiate handoff calls from originating calls.

Our work in this paper is to make the preemption policy not so “preemptive”. The pure preemption approach is currently not allowed in today’s networks in the United States [4], but in this paper we propose ways to use preemption while not being as harsh for public users since their calls can resume after a short time. In other words, we study possible strategies that can help low priority calls to get a better chance to continue. The basic idea is combining a queuing method with a preemption method. Queues will be added for preempted calls and handoff calls, so preempted calls can be resumed when possible. To facilitate comparison with pure queuing and pure preemption policies, we also call the combined preemption and queuing policy as combined policy in the later sections. For our analysis we make following assumptions: (1) Same as mentioned in [6], the main three types of voice calls we are to deal with are emergency calls, public handoff calls and public originating calls, where the originating calls mean those staring inside the cell. These two types of public calls will also be mentioned as low priority calls in later sections. (2) Our study focuses on a single cell. (3) All call durations and inter-arrival times are independently, identically, and exponentially distributed. (4) There is no handoff for emergency calls; we assume most emergency users will be stationary within a disaster area. However, for assumption (4), the model given here can be easily extended to a more general situation.

A related work is in [12]; they studied two types of traffic: real-time (voice) and non-real-time (data). Each type consists of both originating and handoff traffic, and both handoff traffics have their own queues. They designed the network such that real-time handoff traffic can preempt resources from ongoing non-real-time traffic, and put the interrupted traffic into the non-real-time handoff traffic queue. However, the

This work is supported by the United States National Science Foundation under CAREER Award ANI-0133605.
behavior of expiration of calls in the queues was not studied, and no detailed analysis was done about multiple preemptions, possible ways to avoid them, or performance comparisons between different policies.

In our paper, all traffics we deal with are voice, and thus are sensitive to waiting long in the queue whether they are queued handoff calls or preempted calls. Based on this fact we make detailed analysis of the expiration of calls in a queue (users abandon because impatience or out of handoff area), the behavior of preemption, and its effect on system performance.

The main contributions of this paper include:

1) Introducing combined preemption and queuing methods for a network that supports emergency traffic, in which abandonment of calls waiting in queue is also considered. Performance metrics like blocking probability, preemption probability, and average numbers of preemptions per call are analyzed.

2) Both one queue and two queue cases are analyzed. With the one queue case, handoff calls and preempted calls share one queue and are scheduled by the FIFO rule when channels are available, which can help to make resources more efficiently used and save the cost of equipment and management. For the two queue case, we put preempted calls and handoff calls into different queues, scheduling of the 2 queues when channels are available can use priority queuing or weighted EDD scheduling as depicted in [6].

3) A priority modification policy is brought out to avoid multiple preemption. In essence, a low priority call becomes a high priority call once it returns from a queue, so it cannot be preempted again. Analytical results about performance metrics are given.

4) The behaviors of combined preemption and queuing schemes, the pure queuing scheme, and the pure preemption scheme are compared; thus deep insight into the benefits and shortcomings of each policy are provided. Operators can decide which scheme is closer to their requirements.

This paper shows that preemption is a good way to ensure emergency traffic’s priority in a wireless cellular network, but the pure preemption policy seems to be too strong and too harsh for public traffic. Combining queuing with preemption can increase the chance of public traffic to succeed and makes the system resources more wisely used. A policy that restricts preemptions to only happen once per call is even nicer to public traffic, but also affects emergency performance somewhat.

In Section II, different schemes and performance analysis is given, Section III evaluates the performance of each scheme, and Section IV compares these possible schemes, also with the pure queuing and pure preemption scheme. Section V concludes this paper.

II. COMBINED POLICIES AND PERFORMANCE ANALYSIS

A. Combined scheme with 1 queue

In Fig. 1 we illustrate the combined scheme with only 1 queue. Class 1 is emergency calls, Class 2 is handoff calls, and Class 3 is calls originating from within the cell. When an incoming call fails to find free channels, if it’s an emergency call, it will preempt resources from ongoing low priority calls randomly (either originating or handoff), and the preempted calls are put into a queue; if the incoming call is a handoff call, it will be put into the queue same as preempted calls; if it’s an originating call, it will be blocked immediately. When there are channels available later, one call from the queue will be served according to the FIFO policy. The queue is finite, so the blocking of handoff and preempted calls can also happen.

To facilitate analysis, we assume that the service times of preempted calls are renewed, and the expiration time is always the same after each resumption. In real life the user can become more and more impatient after multiple times of preemption. And the more impatient the user is, the larger the chance the user will drop the call. This will be studied in later work. Additionally, in this section where only one queue is used, we assume that the expiration time of preempted calls and handoff calls waiting in the queue are both exponentially distributed with the same average value.

The state diagram of this strategy is shown in Fig. 2. Each state is identified as (i,j,k), where i means the number of channels taken by emergency calls, j represents the number of channels taken by low priority ones (regardless of being originating or handoff calls before being admitted by this cell), and k is the number of calls in the queue, including both handoff and preempted calls. The arrival rate for emergency, handoff, and originating calls is \( \lambda_1, \lambda_2, \lambda_3 \), and the service rate for emergency and low priority calls are \( \mu_1, \mu_2 \) individually. The average expiration time for calls waiting in the queue is \( 1/\mu_{exp} \). The total number of channels is C and the queue length is L. Steady state results can be obtained by solving the balance equations directly.

The main performance metrics we consider include: admission probability and success probability (i.e., probability of finishing normally without expiring or dropping) for each class, preemption probability for a low priority call given that it is admitted, and the average number of times that a call is preempted. To provide the analysis for these performance metrics, we also find related metrics like the blocking probability of each class, the expiration probability of calls in the queue, and the probability mass function for the number of times a call can be preempted.

1) Blocking probability
When an emergency arrival finds no channels immediately available and no ones to preempt (all ongoing calls are emergency ones), it will be blocked directly. For a handoff call, it will be blocked only when the queue is full. And originating calls will be blocked whenever the channels are all occupied. From Fig. 2 we can see that:

\[ P_{B,1} = \sum_{k=0}^{L} P(C, 0, k) \]  
\[ P_{B,2} = \sum_{i=0}^{C} P(i, C - i, L) \]  
\[ P_{B,3} = \sum_{k=0}^{L} \sum_{i=0}^{C} P(i, C - i, k) \]  

(2) Expiration probability

Calls in the queue will expire if not resumed in time. Expiration probability is defined as the total expired calls divided by the total arrived calls to the queue; alternatively the total numbers in the numerator and denominator can be replaced by rates. The arrivals to the queue include preempted calls and handoff calls, among which the preempted calls are caused by the arrival of emergency calls when all channels are busy and some channels are available to be preempted (low priority ones), so the rate of preempted calls to the queue is \( \lambda_1(P_{B,3} - P_{B,1} - P_{PrmDrp}) \), where \( P_{B,3} \) means all channels are busy, \( P_{B,1} \) means no channels available to be preempted, \( P_{PrmDrp} \) is the probability that a call is preempted and dropped immediately because the queue is full, which is defined as: \( P_{PrmDrp} = \sum_{i=0}^{C-1} P(i, C - i, L) \), while the probability that a call is dropped given that it’s preempted is: \( P_{Drp}^{Prm} = P_{PrmDrp}/P_{Prm} \). The handoff calls will enter into the queue when all channels are busy and the queue is not full, so the rate of handoff calls into the queue is \( \lambda_2(P_{B,3} - P_{B,2}) \). So: \( Rate_{Arr2Q} = \lambda_1(P_{B,3} - P_{B,1} - P_{PrmDrp}) + \lambda_2(P_{B,3} - P_{B,2}) \). The expiration probability is thus calculated as following:

\[ P_{Exp} = \frac{\text{Expired calls/sec}}{\text{Arrival calls to queue/sec}} = \frac{\sum_{k=1}^{L} \sum_{i=0}^{C} P(i, C - i, k)k\mu_{exp}}{Rate_{Arr2Q}} \]

(4)

Because handoff calls and preempted calls in the queue have the same expiration rate, they also have the same expiration probability, which is denoted as \( P_{Exp} \).

(3) Preemption probability and average number of preemptions

The overall preemption probability for ongoing low priority calls is equal to the rate of calls preempted divided by the rate of low priority calls activated. The activated calls consist of two parts: the calls directly accepted, and those resumed from the queue. The latter part means those calls entered into the queue that didn’t expire. So, we can calculate the preemption probability as following:

\[ P_{Prm} = \frac{\text{Preempted calls/sec}}{\text{Activated calls/sec}} = \frac{\lambda_1(P_{B,3} - P_{B,1})}{(\lambda_2 + \lambda_3)(1 - P_{B,3}) + Rate_{Arr2Q}(1 - P_{Exp})} \]

(5)

The average number of preemptions given that a call is accepted can be computed in two ways. One is according to the definition:

\[ NumPrm = \frac{\text{Preempted calls/sec}}{\text{Accepted calls/sec}} = \frac{\lambda_1(P_{B,3} - P_{B,1})}{(\lambda_2 + \lambda_3)(1 - P_{B,3}) + \lambda_2(P_{B,3} - P_{B,2})(1 - P_{Exp})} \]

(6)

Here the accepted calls means calls accepted to be served; each accepted call will only be counted once even if it’s preempted and resumed multiple times.

Another approach is based on the preemption and expiration probability we already derived above. In Fig. 3 we show the probability flow of low priority calls. In the frame, F means failed, S means successful, while “@” symbols in the two left trees represent a jump to the ”@“ symbol in the right tree.
The right tree represents what might happen after a call is accepted. We notice that a call can be preempted multiple times, and the sub-flow is recursive here. From the figure we can get the probability of being preempted for 1 time, 2 times, ... n times. It’s a geometric process and the probability mass function is:

\[ Pr(\text{Preempted n times}) = P_{\text{Prm}}(1-A)A^{n-1} \]  

Here \( A = (1 - P_{\text{Drp}})(1 - P_{\text{Exp}})P_{\text{Prm}} \), and we know the expected value of this mass function is \( \frac{P_{\text{Prm}}}{1-A} \). So we obtain:

\[ \text{Num Preempted} = \frac{P_{\text{Prm}}}{1 - P_{\text{Prm}}(1-P_{\text{Exp}})P_{\text{Drp}}} \]  

This probability mass function could also be used to compute many other metrics as well.

Since the average number of preemptions we calculated above can be less than 1 or greater than 1, it is difficult to compare different scenarios or schemes. Here we use another concept–relative average number of preemptions, which is defined as the average number of preemptions given a call is preempted:

\[ \text{Relative Average Number of Preemptions} = \frac{\text{Average Number of Preemptions}}{\text{Call Preempted}} \]

\[ = \frac{1}{1 - P_{\text{Prm}}(1-P_{\text{Exp}})P_{\text{Drp}}} \]  

(4) Average waiting time

To compute the average waiting time in the queues, Little’s law will be applied. From Fig. 2 we know that the average queue lengths are computed as: \( \bar{L}_{\text{queue}} = \sum_{i=1}^{L} \sum_{k=0}^{C} P(k,C-k,i) \). The average arrival rate to the queue is \( \lambda_{\text{Arr2Q}} \). So:

\[ T_{\text{Wait}} = \frac{\bar{L}_{\text{queue}}}{\lambda_{\text{Arr2Q}}} = \frac{\sum_{i=1}^{L} \sum_{k=0}^{C} P(k,C-k,i)}{\lambda_{\text{Arr2Q}}} \]  

(5) Admission Probability

Now we show how to calculate the admission probability of each class. For emergency calls and originating calls, they will be admitted if not blocked. Handoff calls can be admitted in two ways: directly admitted or admitted after waiting some time in the queue. The directly admitted part is equal to the admission probability of originating calls, the admission after waiting part is \( (P_{B,3} - P_{B,2})(1 - P_{\text{Exp}}) \), which means those enter into the queue and not expire. So,

\[ P_{\text{Adm}}^{\text{Emr}} = 1 - P_{B,1} \]  

\[ P_{\text{Adm}}^{\text{Orig}} = 1 - P_{B,3} \]  

\[ P_{\text{Adm}}^{\text{Ho}} = (1 - P_{B,3}) + (P_{B,3} - P_{B,2})(1 - P_{\text{Exp}}) \]  

(6) Success probability

For emergency calls, all of the admitted calls will be successfully finished, thus satisfying the dependability requirement. But for low priority calls, this kind of dependability cannot be assured. To compute the successfully finished probability with possibility of preemption, from Fig. 3 we can see that firstly we need to get \( Pr(\text{A call succeeded}|\text{Accepted}) \). This can be computed as:

\[ Pr(\text{Succ}|\text{Accept}) = (1 - P_{\text{Prm}}) \sum_{i=0}^{\infty} (P_{\text{Prm}}(1 - P_{\text{Drp}})(1 - P_{\text{Exp}}))^i \]

\[ = \frac{(1 - P_{\text{Prm}})}{1 - P_{\text{Prm}}(1 - P_{\text{Drp}})(1 - P_{\text{Exp}})} \]  

Then obviously:

\[ P_{\text{Succ}}^{\text{Emr}} = P_{\text{Adm}}^{\text{Emr}} Pr(\text{Succ}|\text{Accept}) \]  

\[ P_{\text{Succ}}^{\text{Orig}} = P_{\text{Adm}}^{\text{Orig}} Pr(\text{Succ}|\text{Accept}) \]  

B. Combined scheme with 2 queues

In the above subsection we assumed that preempted calls and handoff calls have the same expiration time distribution and the same mean expiration time. Generally this is not true. Thus analysis with 1 queue will be very difficult, and makes us turn to using 2 queues, one for preempted calls and another for handoff calls. This is shown in Fig. 4.

With two queues there, we also can have different choices to schedule calls from the queue when channels are available. In this paper, we use the priority queue and assume that handoff calls have higher priority. In later work, we could use some strategy like a weighted earliest deadline scheduling scheme [6] to provide flexible ways suitable for operators.

The state diagram for the two queue case is a 3-dimensional Markov chain, which is much more complex than the one queue case. To make it clear, we just give the example with the number of channels \( C=2 \), the length of both queues is \( L=1 \). This is shown in Fig. 5. In this figure \( \mu_{\text{exp}1} \) is the expiration rate for calls in the handoff queue, and \( \mu_{\text{exp}2} \) is the expiration rate for calls in the preempted queue. Each state is identified as \((i,j,m,n)\), while \( i,j \) has the same meaning as the one queue case, \( m \) means the number of handoff calls in queue 1 (handoff queue), \( n \) means the number of preempted calls in queue 2 (preempted queue).

The performance analysis is similar to the one queue case, which is shown as follows:

(1) Blocking probability
From Fig. 5 we can see that:

\[
P_{B,1} = \sum_{m=0}^{L_1} \sum_{n=0}^{L_2} P(C, 0, m, n) \quad (17)
\]

\[
P_{B,2} = \sum_{i=0}^{C} \sum_{m=0}^{L_2} P(i, C - i, L1, n) \quad (18)
\]

\[
P_{B,3} = \sum_{i=0}^{C} \sum_{m=0}^{L_1} \sum_{n=0}^{L_2} P(i, C - i, m, n) \quad (19)
\]

(2) Expiration probability

Here we need to compute the expiration probability for handoff queue and preempted queue individually. For handoff queue:

\[
P_{Exp}^{Ho} = \frac{\text{Expired calls in queue 1/sec}}{\text{Arrival calls to queue 1/sec}} = \frac{\sum_{i=0}^{C} \sum_{m=1}^{L_1} \sum_{n=0}^{L_2} P(i, C - i, m, n) \mu_{exp1}}{\lambda (P_{B,3} - P_{B,2})} \quad (20)
\]

For preempted queue:

\[
P_{Exp}^{Prm} = \frac{\text{Expired calls in queue 2/sec}}{\text{Arrival calls to queue 2/sec}} = \frac{\sum_{i=0}^{C} \sum_{m=1}^{L_1} \sum_{n=1}^{L_2} P(i, C - i, m, n) \mu_{exp2}}{\lambda (P_{B,3} - P_{B,1} - P_{PrmDrp})} \quad (21)
\]

(3) Preemption probability and average preemption times

The computation is basically the same as the one queue case, the only difference is on the calculation of the traffic rate out of the queue: \(Rate_{out} = \lambda (P_{B,3} - P_{B,1} - P_{PrmDrp})(1 - P_{Exp}^{Prm}) + \lambda 2(P_{B,3} - P_{B,2})(1 - P_{Exp}^{Ho})\). Here \(P_{PrmDrp}\) is defined as:

\[
P_{PrmDrp} = \sum_{i=0}^{C-1} \sum_{m=0}^{L_1} P(i, C - i, m, L_2) \quad (22)
\]

(4) Average waiting time in the queues

The average queue lengths are computed as:

\[
\bar{q}_{Ho} = \sum_{i=0}^{C} \sum_{m=1}^{L_1} \sum_{n=0}^{L_2} P(i, C - i, m, n) m \quad (23)
\]

\[
\bar{q}_{Prm} = \sum_{i=0}^{C} \sum_{m=0}^{L_1} \sum_{n=0}^{L_2} P(i, C - i, m, n) n \quad (24)
\]

The average arrival rate to handoff and preempted queue is: \(\lambda 1(P_{B,3} - P_{B,1} - P_{PrmDrp})\) and \(2\lambda (P_{B,3} - P_{B,2})\) individually. So using Little’s law we can get the average waiting time in queues:

\[
T_{Ho} = \frac{\sum_{i=0}^{C} \sum_{m=1}^{L_1} \sum_{n=0}^{L_2} P(i, C - i, m, n) m}{\lambda (P_{B,3} - P_{B,1} - P_{PrmDrp})} \quad (25)
\]

\[
T_{Prm} = \frac{\sum_{i=0}^{C} \sum_{m=0}^{L_1} \sum_{n=0}^{L_2} P(i, C - i, m, n) n}{\lambda (P_{B,3} - P_{B,2})} \quad (26)
\]

(5) Admission and successfully finished probability

After comparing Fig. 6 with Fig. 3, we can see that the formula to compute the success probability with preemption possible are the same as 1 queue case. The difference is that generally, \(P_{Exp} \neq P_{Exp}^{Ho}\) and following equations are the only ones need to be replaced:

\[
P_{Adm}^{Ho} = 1 - P_{B,3} + (P_{B,3} - P_{B,2})(1 - P_{Exp}) \quad (27)
\]

\[
Pr(Succ|Accept) = (1 - P_{Prm}) \sum_{i=0}^{\infty} \left( P_{Prm}(1 - P_{Exp}^{Prm})(1 - P_{Exp}^{Prm}) \right)^i (1 - P_{Prm}) \quad (28)
\]

C. Single preemption scheme

For the combined policy given above, we can see that once a call is preempted, nothing stops it from possibly being preempted many times. This is annoying to users and may cause more calls to be abandoned than we have shown in the above analysis. To avoid it, we can increase the priority of calls after they are resumed from the preemption queue.
By introducing a priority change mechanism, we also hope to improve the low priority user’s chance of finishing the call while not harming the total system’s performance. Since this policy ensures at most one preemption per call, we call it the single preemption policy. Correspondingly the basic combined policy we have already introduced is called the multiple preemption scheme in later sections. The idea of this method is shown in Fig. 7. And the probability flow for the single preemption policy is shown in Fig. 8.

The state diagram for the single preemption policy is basically the same as the multiple preemption case; the only difference is that state transitions are different when there is a channel free and no calls are waiting in the handoff queue; the preempted calls will be resumed and become high priority calls which cannot be preempted again. The definition for state (i,j,m,n) is the same as the multiple preemption case.

Most computations here are similar to the multiple preemption case. Here we just show those that are different:

(1) Preemption probability

The only difference from multiple preemption for computing preemption probability lies in the average rate out of the queues. Because calls taken from the preempted queue become high priority ones and will never be preempted again, we have

\[
\text{Rate}_{\text{Out}_{Q}} = \lambda_2(P_{B,3} - P_{B,2})(1 - P_{\text{Prm}}^{\text{Exp}})
\]

In addition, since a call can be preempted one time at most, the expected number of preemptions is the same as the preemption probability, and the relative preemption times is always 1.

(2) Successfully finished probability

According to Fig. 8 we can get the success probability easily. For an accepted call, if it is finished directly or is preempted once and then resumes (becomes a high priority call, surely to be finished successfully), then it’s regarded as being a success. We can get

\[
\Pr(Succ|\text{Accpt}) = (1 - P_{\text{Prm}}) + P_{\text{Prm}}(1 - P_{\text{Prm}}^{\text{Exp}})(1 - P_{\text{Exp}}^{\text{Exp}})
\]

Then we can use equations (15) and (16) to get the success probability for handoff and originating calls.

III. PERFORMANCE EVALUATION OF EACH POLICY

A. Parameter specification

(1) The relation between the handoff calls and the originating calls

As mentioned in [10], based on the assumption of handoff in and handoff out in each cell are equal, the arrival rates of originating and handoff calls satisfy such condition:

\[
\lambda_h = \lambda_o \frac{P_h (1 - P_{\text{Org}}^{\text{Exp}})}{1 - P_h (1 - P_{\text{Org}}^{\text{Exp}})}
\]

Where \(P_h\) is the probability that a ongoing call will handoff, \(\lambda_h\) and \(\lambda_o\) is the rate for handoff call and originating call individually, corresponding to \(\lambda_2\) and \(\lambda_3\) in our above denotation. \(P_{\text{Org}}^{\text{Exp}}\) is the failure probability for originating call, and \(P_{\text{HO}}^{\text{Exp}}\) is the failure probability for handoff calls in a cell.

Given \(P_h\) is fixed, then we find that the ratio of originating calls to handoff calls are decided by the failure probability of handoff and originating calls. While from above subsections we can see that the failure probability is decided by the arrival rates. Obviously this is a recursive problem, and it’s hard to get a closed form solution for the scenario we study. In our following evaluations we just assume that \(\lambda_2 = 0.4\lambda_3\)

(2) Basic parameters

With the methods for computing performance metrics(performance metric calculation method) provided in the above sections, we can study how they affect the system performance by changing one parameter while keeping all other parameters fixed. The basic parameters we use are:

\[
\mu_1 = \mu_2 = 0.01(\text{average service time is 100 seconds}),
\]

the number of channels \(C = 10\), arrival rates: \(\lambda_1 = 0.05, \lambda_2 = 0.02, \lambda_3 = 0.05\). For the one queue case, expiration rate \(\mu_{\text{Exp}} = 0.02(50 \text{ seconds})\), and the queue length \(L = 10\); for the two queue case, expiration rates: \(\mu_{\text{Exp1}} = \mu_{\text{Exp2}} = 0.02(50 \text{ seconds})\), and queue lengths \(L_1 = L_2 = 5\).

B. Combined policy - 1 queue

In Figs. 9-14, we show the system performances’ change according to the change of arrival rates and expiration rates for calls waiting in the queue. The effect of the queue will be discussed in the later section where the pure preemption policy is compared with the combined policy. The detailed analysis is as follows:

(a) Figs. 9 and 10 show how the system performance changes corresponding to the change of the arrival rate of emergency calls. Here and also in the figures following, the “Average Preemption Times” means the relative average number of preemptions.
When $\lambda_1$ increases, the preemption probability and average preemption times both increase until the preemption probability is close to 1.0, which means almost every low priority call will be preempted once it is admitted or resumed. Correspondingly, the successfully finished probability of low priority calls becomes very low. This shows that the low priority calls’ chance of admission and success is seriously harmed when the traffic of emergency calls keeps increasing, which proves that high priority is really given to the emergency calls.

We also noticed that when $\lambda_3$ is large enough, the admission probability of handoff calls almost does not change despite the fact that success probabilities chance keep decreasing. This is because handoff calls are put into the queue and, once a channel becomes available, a handoff call will take the channel before a new coming emergency call, although probably it will be preempted by emergency calls later and make the success probability low.

(b) In Figs. 11 and 12 we show the effect of low priority traffic. We can see that when $\lambda_3$ and $\lambda_2$ increases, the admission of emergency calls is not affected at all. So the available channel resources for low priority calls stays the same despite their traffic volume (also arrival to the queue) going up, which is the reason that expiration probability keeps increasing.

Besides, when low priority traffic increases, more handoff calls will be waiting in the queue, so the chance for preempted calls to find the queue full is larger and larger, thus making more preempted calls dropped and making the average preemption times decrease after the system exceeds a certain degree of congestion.

We also find that the admission/success probability of both handoff and emergency calls decreases, but originating calls decrease much faster. This shows that with combined policy applied, the handoff calls have the advantage over the originating calls.

(c) Figs. 13 and 14 reflect the performance change according to the average expiration time. We can see that as the average expiration time becomes longer, the calls in the queue can endure longer waiting time, thus the expiration probability will become lower. Correspondingly the preemption probability and average preemption times become higher because more calls will be resumed from the queue.

A handoff call’s admission and success probability improves when the average expiration time becomes longer, while it becomes worse for the originating calls because the handoff calls have a better chance to acquire channels; the emergency calls are also not affected by the change of expiration time. The overall success probability improves when the expiration time becomes longer.

In conclusion, from the above analysis we can see that the
success probability of emergency calls is not affected by low priority traffic or the expiration time of calls in the queue. This shows that the emergency calls have absolute priority in this combined policy.

C. Combined policy - 2 queues

If we choose the same parameters as the one queue case, the analytical results about performance evaluation are mostly the same, so we will not repeat them here. The different phenomenon we can study here is the system performance when the expiration rates of two queues are different, which are shown in Fig. 15 through Fig. 20.

As shown in Figs. 15, 16 and 17, we keep the expiration time of the preempted queue (Queue 2) at 50 seconds, and watch how the handoff queue’s (Queue 1) expiration time affects the performance. While in Figs. 18, 19 and 20, we keep the expiration time of the handoff queue at 50 seconds, and then change the preempted queue’s expiration time.

From these results we can see that:

(a) When the average expiration time of the handoff queue is longer, the success probability for handoff calls also goes up, while the originating call’s chance to succeed becomes worse. Yet the total success probability increases slowly.

(b) Both the handoff and originating call’s waiting time in the queue will increase, which can be seen as the cost of improved success probability.

(c) When the average expiration time of the preempted queue goes up, success probability for the handoff call still increases while originating call’s decreases, but both slower than the case where the average expiration time of the handoff queue increases.

(d) When the expiration time of the preempted queue changes, average waiting time in the handoff queue almost does not change. This shows that the change in the preempted queue does not affect the admission of handoff calls from the handoff queue. It’s easy to understand here because the handoff queue has absolute priority over the preempted queue.

(e) After comparing Figs. 16 and 19 we can see that the change of expiration time of one queue generally does not change the expiration probability of another queue.

D. Single preemption scheme

Now we show that with single preemption policy applied, how those parameters’ change can affect the performance.

(a) As shown in Figs. 21 and 22, when low priority traffics go up, expiration probability of both handoff and preempted calls goes up, and preempted calls’ increases very fast, which shows they are mostly affected because they have lower priority and they need to wait much longer compared with
Fig. 17. 2 queues: Expiration time of handoff queue vs. Average waiting time

Fig. 18. 2 queues: Expiration time of preempted queue vs. Success Prob.

handoff calls. The preemption probability increases at first, then decreases slowly after some point. This is because more and more preempted calls are blocked after being preempted. The change of admission probability for emergency calls will be discussed in a later section.

(b) Figs. 23 and 24 show the performance change according to the average expiration time. We can see that as the average expiration time becomes longer, the calls in the queue can endure longer waiting time, thus the expiration probability will become smaller. But correspondingly the preemption probability will increase, which is because more preempted calls will be resumed with new higher priority and occupy more channels, making the number of channels that can be preempted become decrease.

The handoff call’s admission and success probability improves when the average expiration time becomes longer, while that for the originating calls becomes worse because the handoff calls have a better chance to acquire channels. Emergency calls also are not affected by the change of expiration time. But the total success probability still improves. In this aspect it is almost the same as the multiple preemption case.

When expiration rate is extremely high, then it becomes as if no queue exists.

IV. COMPARISON BETWEEN DIFFERENT POLICIES

A. System Utility Evaluation

To facilitate the comparison between different policies, we introduce two new concepts: admitted system utility and successful system utility, which are defined as

\[
SysUtilA = \sum_{i=1}^{3} \frac{\lambda_i P_{Adm,i}}{C\mu} \quad (30)
\]

\[
SysUtilS = \sum_{i=1}^{3} \frac{\lambda_i P_{Succ,i}}{C\mu} \quad (31)
\]

\(SysUtilA\) means the system utility with regard to admitted calls, and \(SysUtilS\) represents the system utility with regard to successfully finished calls. There are some calls admitted but terminated before normal ending because of failing to enter into the queue after preemption, or waiting too long in the queue so that they lose the patience and are abandoned (expired). Expired calls are not normally finished and will cause dissatisfaction, but still occupy some system resources. This can be viewed as kind of waste of resource.

It’s obvious that \(SysUtilA \geq SysUtilS\) and \(SysUtilS \leq 1\). The closer \(SysUtilS\) is to 1, or the closer is \(SysUtilA\) to \(SysUtilS\), the better is the chance for the channels to be taken by those calls which will be successfully finished after
being admitted, and we can say that resource is more wisely used.

B. Combined policy vs. Pure preemption policy

With a queuing policy added to the preemption method, the system needs to add memory and will have more to manage. But what’s the benefit? And will emergency call’s performance be affected? These questions will be answered in the following analysis.

In making comparisons, generally we make most parameters fixed and only one parameter will change. The basic parameters we use are the same as what we mentioned in Section III.

Figs. 25, 26 and 27 show how the performance metrics change when the queuing policy is applied ($L = 0$ vs. $L > 0$), we can see that:

(a) As shown in Fig. 25, when the queuing policy is applied, the low priority calls have higher success probability while the emergency calls stay the same, so the total success probability is improved. And it keeps increasing when the queue length become longer, yet the incremental improvement is lower and lower so that after some point an increase in the queue length does not have much effect. Note that a pure preemption policy corresponds to the queue length equal to zero.

(b) With or without a queue makes a big difference for handoff and originating calls. When the queuing policy is

---

**Fig. 21.** Success prob. vs. $\lambda_3$ for single preemption case

**Fig. 22.** Preemption vs. $\lambda_3$ for single preemption case

**Fig. 23.** Success prob. vs. $\mu_{exp}$ for single preemption case

**Fig. 24.** Preemption vs. $\mu_{exp}$ for single preemption case

**Fig. 25.** Comparison of queuing vs. no queuing for success prob.

**Fig. 26.** Comparison of queuing vs. no queuing for admission prob.
applied, handoff calls have a better chance to be admitted and succeed, and for originating calls the chance will be worse.

(c) Fig. 26 shows that the admission probability of low priority calls is lower when the queuing policy is applied, while for emergency calls it doesn’t change. So we can conclude that the total admission probability goes down when the queuing policy is applied and when queue length increases.

d) According to Fig. 27, preemption probability for low priority calls is also higher when the queuing policy is applied. This shows that with a queuing policy added, low priority calls might be preempted more often and need to wait some time in the queue, but also have a better chance to finish.

So, the introduction of queues makes fewer low priority calls to be admitted, but more to succeed. This shows that the system resources are more wisely used, and also proves that the combined policy works better than a preemption policy by itself.

C. Combined policy vs. Pure queuing policy

From the above subsection we can see that the combined policy is indeed an improvement upon a pure preemption policy. Now we will see how it is compared with a pure queuing policy. A pure queuing policy is what is widely used today for emergency calls.

In our work for pure queuing policy [6], we use two queues, one for emergency and one for handoff. This is shown in Fig. 28. The scheduling scheme we used is weighted earliest deadline scheduling, which can leave the choice to operators to decide either emergency or handoff calls have higher priority, and how much higher is the priority. This is accomplished by setting the weighting parameter.

In this experiment for the pure queuing case, we take the weighting parameter as 0, which means it’s a priority queue and emergency calls have the priority over handoff calls. In Fig. 29, 30 and 31 we show the performance change according to originating traffic for both policies (recall that handoff traffic is in fixed proportion to originating traffic). We can see that:

(a) From Fig. 29 we can see that, when low priority traffic (combination of handoff and originating) goes up, the success probability of the combined policy doesn’t change while that of the pure queuing policy keeps dropping.

(b) The total success probability for both schemes is almost the same despite how low priority traffic changes. The curves lie virtually on top of each other.
(c) Using the system utility concept we defined in subsection IV-A, through Fig. 30 we find that the successful system utility for both schemes also stays almost the same. So both schemes provide the same efficiency in using system resources. The admitted system utility for the preemption and queuing case is higher, and it can be higher than 1 when the system becomes very congested, which means that some calls are admitted but dropped before they succeed to finish (the system admits more than it can handle).

(d) As shown in Fig. 31, for a pure queuing policy, the average waiting time in the queue for emergency calls keeps steady when the low priority traffic changes. But for handoff calls the waiting time keeps increasing. For the combined policy, there is no queue for emergency calls, so they also do not need to wait before getting admitted.

So, we can conclude that compared with a pure queuing policy, the total system performance of the combined policy is almost the same. But it makes the emergency calls’ performance more guaranteed with no need to wait. As a tradeoff, both handoff and originating calls get degraded performance.

D. One queue vs. Two queues for combined policy

As pointed out in subsection III-C, one main benefit of applying 2 queues is that it can deal with wider conditions: the expiration time of handoff calls and preempted calls can be different, and we can have much more flexibility in choosing a scheduling policy. But splitting one queue into two queues also can have an effect on management and system performance. In this subsection, we will compare these two schemes when queue length changes. We assume the length of each queue in the two queue case is half of that in one queue case. So, the total queue length is the same, which means the same system resource used.

From Fig. 32, 33 and 34, we can see that:

(a) The success probability of the total system is higher when using only 1 queue, and is especially obvious when the total queue length is small. This shows that the splitting of the queue will degrade the system’s efficiency.

(b) If the queue is split into an individual handoff queue and a preempted queue, the handoff calls have a greater chance to be blocked directly, while the originating calls have less.

(c) Calls in the handoff queue will have shorter average waiting time than in the one queue case, while calls in the preempted queue have longer waiting time. This is because the handoff queue is given absolute priority. And as the queue length increases, this trend is more obvious.

E. Multiple preemption vs. Single preemption

Finally, this section illustrates the difference in the behavior between the multiple preemption and the single preemption policy. In Figs. 35 and 36, we show the success probability
when low priority traffic rate and queue length change. We can see that:

(1) As shown in Fig. 35, when the low priority traffic goes up, the admission probability of emergency calls is not affected in the multiple preemption case. But in the single preemption case, the emergency calls are affected. We notice that the admission of emergency calls will go down at first because some resources are taken by the low priority calls which can’t be preempted. But when the low priority traffic keeps going up, the emergency call’s admission probabilities go up instead. This is because after the system is congested, more and more preempted calls are blocked after being preempted or expire in the preempted queue, so the calls resumed from the preempted queue are less and less and the chance for emergency calls to be admitted is better and better.

(2) We can also see that the success probability of emergency calls in the single preemption case is always lower than the multiple preemption case. But the success probability of the total system is the same for both policies as low priority traffic changes.

(3) From Fig. 36 we can conclude that for the single preemption mechanism, emergency calls have a worse chance to succeed as the queue length increases while the low priority calls have a better and better chance.

So, we can see that with the single preemption policy applied, the low priority calls have a better chance to succeed compared with the multiple preemption case, while the emergency calls are not guaranteed as in the multiple preemption case. In other words, the single preemption policy is more friendly to low priority public calls, and thus is worthy of being considered to be applied in real life operation.

V. CONCLUSION

Preemption is a good way to ensure priority to emergency traffic in a wireless cellular network. Yet the pure preemption policy seems to be too strong and too unfair for public traffic, and thus is generally forbidden in real life networks. This paper shows that the introduction of a queuing policy can increase the chance of public traffic to succeed and make the system resources more wisely used. The performance of emergency traffic is not harmed at all for the basic combined policy, which has also been called the multiple preemption scheme in this paper.

For the combined policy, this paper also studies two possible schemes about the management of queues: using one shared queue or splitting the queue into two, one for the handoff calls and one for the preempted calls. With 2 queues it’s much easier to analyze the performance and we can have more flexible schemes, but with only one queue we can see that the system is even more efficient.

We also show that with the basic combined policy, there is the multiple preemption problem. To avoid this problem, we bring out a new policy which only allows the ongoing public call to be preempted at most once. Then we find that the emergency call’s admission will be affected, and as a tradeoff, public users have a better chance to finish the call. This provides another possible scheme to operators because public users can benefit more from it. So, combining preemption with queuing is a viable option to consider in today’s cellular systems, whether allowing multiple preemptions or single preemptions.

In future work, we will consider the case where the service time of resumed calls are not memoryless, and the fact that users will be more impatient after being preempted for several times. Also the retry of failed users will be included.

REFERENCES