We developed an analytical model of the Automated Call Distribution (ACD) system which allows us to obtain results for systems with possible retrials after a busy signal, with an impatient hang up, and with limitations on the number of access circuits and waiting time. A special iteration algorithm is applied to this model. As the comparison with simulation results shows, this approach allows us to obtain good results.

1. INTRODUCTION

One of the most important issues in provisioning Automated Call Distribution (ACD) systems is deciding the optimal configuration (the number of agents, access circuits, etc.) for the ACD system. In [FEIN 90] we developed a simulation model of ACD service and showed that correct sizing of the ACD system depends crucially on the choice of the performance metrics for estimating the efficiency of the ACD. We also found an optimal correspondence between the number of agents and the number of access circuits. In this work we develop an analytical model of the ACD system which allows us to obtain explicit results for systems that have both possible retrials after a busy signal, an impatient hang-up, and limitations on the number of access circuits and waiting time.

Let us give a brief description of the system. The number of access circuits to an agent pool in the ACD system is limited, and if the call finds all circuits occupied, it gets a busy signal, which terminates the call attempt. If there are some free circuits, the call picks one of them. If all the agents are busy, the call waits in line. The discipline "First Served" is applied. The waiting period in the queue can conclude in either of two ways. In the first case, the client gets service. In the second case, the waiting time may be too long, and the impatient client may hang up and abandon the call. The time of impatience is a random variable as is the waiting time for service. The minimum of these two variables is the true waiting time. A call can be terminated by a busy signal or by abandoning the queue. In both cases, a client can repeat his attempt with some probability and create a new call after some random time. These probabilities and times depend on whether the client got a busy signal or was impatient last time. In theory the behavior of each client can have an infinite number of trajectories in the system; these trajectories consist of a mixture of getting a busy signal and hanging up.

There are two retrial traffic loops. Each rejected call may repeat an attempt in a random time with probability \( p_1 \), and each impatient call may do the same with probability \( p_2 \). If we eliminate the case of impatience (abandoning a queue and calling back with some probability), we get a well-known system with repeated calls. To the best of our knowledge, there are no publications on the problem with two loops. There are a number of papers on the "repeated calls" problem. In the first of these papers, published in 1957 [COHE 57], J. Cohen found a mathematical solution for a model with an explicit treatment of repeated calls. He used a finite repetition space in which calls "stay" between rejection and the next attempt. The number of retries per caller is limited by a random variable called "staying time." The staying time, like the interarrival times of new calls, the repetition times, and the call duration, are all assumed to be exponentially distributed. An unsuccessful caller leaves the system when its staying time is over. Similar to Cohen, Brettschneider [BRET 70] also used a limited repetition space. In his model, rejected calls repeat with a repetition probability depending on the mood of the callers. A distinction is made between the repetition probability of new calls and repeated ones. In other work, LeGall [LEGA 70] validated certain parameter assumptions of repeated call models. Jonin and Sedol [JONI 70] developed a model similar to that of Brettschneider and gave a recursive algorithm for a small number of lines. Macfayden [MACF 79] observed repeated attempts in the arrival process by the statistical evaluation of measurement data. Stepanov [STEP 88] developed optimal numerical algorithms for estimation of stationary characteristics of Markov repeated call models. In [PATE 88] the influence of repeated calls on system performance characteristics was examined.

The major method in solving the problem in these papers is to consider a Markov chain with an additional space for rejected calls which retry. This approach leads to the problem of solving a system with a two-dimensional state space in which one dimension represents the number of
busy lines and the other one the number of calls in repetition. While the first dimension (the number of busy lines) is limited by the total number of access lines, the second dimension (the number of calls in repetition) is unlimited. If we apply this approach to our two loops model, we will have a three-dimensional state space in which two dimensions are unrestricted. Also, in order to apply a direct Markov chain approximation, we have to assume that both "sitting" times for retrials are exponential. However, our analyses [FEIN 90] have shown that these parameters are not exponential. To avoid these obstacles, we have developed an approximation model and an iteration algorithm for obtaining performance characteristics for systems of any size. If we have a more complicated problem with a larger number of loops (for example, if some unsatisfied clients call back after the service, thereby creating a third loop), we can apply our algorithm with only slight modifications; in the direct Markov chain approach, each new loop creates a new dimension in a state-transition diagram.

2. APPROXIMATION MODEL

Let us consider the system described in the introduction. A Poisson stream of new calls with parameter \( \lambda \) arrives at the system. There are \( n \) identical agents, each with average service time \( 1/\mu \). The waiting room size is \( m \), and calls arriving in the system with \( m+n \) calls are rejected. Regardless of the number of prior attempts, each rejected call retries in some random time with probability \( p_1 \). With probability \( 1-p_1 \), the rejected call abandons the system. Waiting time in the queue is limited by impatience time, a random variable with average \( 1/v \). If a call abandons the queue, it may call back in a while with probability \( p_2 \) or abandon the system with probability \( 1-p_2 \). It is assumed that the distributions of service time and impatience time are exponential.

Let us denote the intensity of the stream of rejected calls \( \lambda_{\text{rej}} \) and the intensity of the stream of calls which abandoned the queue \( \lambda_{\text{imp}} \). Then the intensity of the stream of new and repeated calls is \( \lambda_{\Sigma} = \lambda + p_1 \lambda_{\text{rej}} + p_2 \lambda_{\text{imp}} \), where \( \lambda_{\text{rej}} \) and \( \lambda_{\text{imp}} \) are unknown parameters.

To solve the problem, we create an approximation model: we break both rejected and impatient retry loops, and consider instead two additional sources of calls. The first source creates a Poisson stream of calls with parameter \( p_1 \lambda_{\text{rej}} \); the second source creates a Poisson stream with parameter \( p_2 \lambda_{\text{imp}} \). With this approach, we remove the correlation between the processes of rejection and impatient abandonment on the one hand, and the process of retries on the other. But we may suppose that this violation is not too serious for the following reason: the interarrival time for calls in a real ACD system is measured in seconds, and often in fractions of a second. On the other hand, the average retry delays are often measured in minutes, i.e., the scale of the delay times is many times larger than the scale for the interarrival times. This enables us to suppose that the correlation, mentioned above, is relatively small and we can neglect it. The major problem in this approach is to find approximations for \( \lambda_{\text{rej}} \) and \( \lambda_{\text{imp}} \).

To find these approximations, we use an iteration algorithm. We consider a new system on each step of the algorithm. At the first step, we assume that all rejected and impatient calls abandon the system, and the arrival process consists only of new calls. For this case, we consider a corresponding Markov process and obtain \((\lambda_{\text{rej}})_1, (\lambda_{\text{imp}})_1\). At step \( 2 \), we consider the system with the same process of new arrivals with parameter \( \lambda \), but in addition, source 1 creates a Poisson stream with parameter \( p_1 (\lambda_{\text{rej}})_1 \), and source 2 creates a Poisson stream with parameter \( p_2 (\lambda_{\text{imp}})_1 \). The sum of these three arrival processes is a Poisson stream with parameter \( \lambda + p_1 (\lambda_{\text{rej}})_1 + p_2 (\lambda_{\text{imp}})_1 \). We note that this system differs from the previous one only in the parameter of the arrival process, which means that we can obtain all the results for step \( 2 \) using the method developed for step 1. At step \( n (n>1) \), we use the same method for the input stream, with parameter

\[
(\lambda_{\Sigma})_n = \lambda + p_1 (\lambda_{\text{rej}})_{n-1} + p_2 (\lambda_{\text{imp}})_{n-1}. \tag{2.1}
\]

We will show below that this process converges for any system with a finite buffer. Thus, we reduce the problem with two loops of repeated calls to the problem without any loops considered in the next section.

3. SOLUTION FOR THE APPROXIMATION MODEL

As we described in Section 2, at each step of the iteration algorithm we consider a multichannel queuing system with Poisson input, finite buffer, and impatient calls. All rejected and impatient clients abandon the system.

3.1 Solution for the first step of the algorithm

We denote the parameter of the input process as \( \lambda \). The service time and the impatience time are exponential, with parameters \( \mu \) and \( v \) correspondingly. The maximum number of places in the queue is \( m \), and the number of agents is \( n \). If there exist \( k \) calls in the system \( (k \leq n) \), all of them are served (each call by one agent). If there are \( n+r \) calls in the system \( (r \leq m) \), \( n \) are served and \( r \) are waiting for service. If a call arriving finds \( n+m \) calls in the system, it is rejected.

Figure 1 shows a state transition diagram for this system. Let us explain all the states. 0 – there are no calls in the system (all the agents are free); \( k – k \) calls are in the system.
The average number of busy access circuits is

$$L_c = \sum_{k=0}^{n} kp_k + \sum_{r=1}^{m} rp_{n+r}. \quad (3.4)$$

For the next step of our iteration algorithm, we need the following parameters:

The probability that a call gets a busy signal is

$$P_{bus} = p_{n+m}. \quad (3.5)$$

The intensity of abandoning the queue:

$$\lambda_{imp} = vL. \quad (3.6)$$

The intensity of the rejected calls traffic:

$$\lambda_{rej} = \lambda P_{bus}. \quad (3.7)$$

### 3.2 Solution for the nth step of the algorithm

We use the same approximation model we used for the first step with one difference: the input stream consists of the sum of a stream of new calls and streams of rejected and impatient calls on the previous step. By analogy with (3.6) and (3.7) we have

$$\lambda_{imp}^n = vL_n. \quad (3.8)$$

and

$$\lambda_{rej}^n = (\lambda \Sigma)_{n-1} (P_{bus})_{n-1}. \quad (3.9)$$

According to (2.1), (3.8), and (3.9), the intensity of arrival process at step \( n \) is

$$(\lambda \Sigma)_n = \lambda + p_1 (\lambda \Sigma)_{n-1} (P_{bus})_{n-1} + p_2 vL_{n-1}. \quad (3.10)$$

Now we can formally describe the iteration algorithm as follows:

At step \( n (n \geq 1) \), we calculate \((\lambda \Sigma)_n \) (where \((\lambda \Sigma)_0 = L_0 = 0, \quad (\lambda \Sigma)_1 = \lambda \)). Then we examine the value \( \Delta_n = [(\lambda \Sigma)_{n-1} (\lambda \Sigma)_{n}] / (\lambda \Sigma)_n \). If \( \Delta_n \) is greater than some value \( c \) (for example, \( c = 0.01 \)), go to step \( n+1 \). If \( \Delta_n \leq c \), we stop the iteration process. The intensity of the sum of the three input processes in the approximation model is \((\lambda \Sigma)_n \). By substituting \((\lambda \Sigma)_n \) for \( \lambda \) in formulas in Section 3.1, we obtain probabilities of states, the average queue length, the average number of busy agents, the average number of busy circuits, etc. We can also obtain some important performance characteristics using the following formulas.
The intensity of dropped traffic (lost clients) is \( (\lambda_{\text{drop}})_n = (1-p_1)(\lambda_S)_n(P_{\text{bus}})_n + (1-p_2)vL_n \). The fraction of served clients (and the probability that a client is served) is \( (P_{\text{cl}})_n = 1 - (\lambda_{\text{drop}})_n/\lambda \). The probability that a call is served is equal to the relative throughput of the system \( (P_{\text{serv}})_n = \mu N_n/(\lambda S)_n \). The probability that a particular agent is busy is \( (P_{\text{agent busy}})_n = N_n/n \). The average waiting time (in queue) of all calls (including rejected and impatient ones) \( (W_q^*)_n = L_n/(\lambda S)_n(1-(P_{\text{bus}})_n) \). The average waiting time (in queue) of calls which got access circuit (including impatient ones) \( (W_q^{**})_n = L_n/(\lambda S)_n(1-(P_{\text{bus}})_n) \). The average waiting time (in queue) of calls which got service \( (W_q^{***})_n = L_n/(\lambda S)_n(P_{\text{serv}})_n \).

4. CONVERGENCE OF THE ALGORITHM

Let us prove that our iteration algorithm converges. At each step of the algorithm, we obtain a new value for the intensity of the input stream \( (\lambda S)_n \) for the system, which is in turn used at step \( n+1 \). To prove the convergence of the algorithm, we have to prove that the sequence \( \{(\lambda S)_n\} \) converges to some finite value as \( n \to \infty \). To do this, we will prove that the sequence \( \{(\lambda S)_n\} \) is a monotonic non-decreasing function of \( n \) and is bounded above.

**Lemma 4.1.** Sequence \( \{(\lambda S)_n\} \) is a monotonic non-decreasing function of \( n \).

**Proof** is by induction. For step 1 \( (\lambda S)_1 = \lambda \). For step 2 from (3.10):

\[
(\lambda S)_2 = \lambda + p_1(\lambda S)_1(P_{\text{bus}})_1 + p_2vL_1 \geq \lambda \text{ for } p_1 \geq 0, p_2 \geq 0.
\]

Suppose

\[
(\lambda S)_n \geq (\lambda S)_{n-1}.
\]

Let us prove that

\[
(\lambda S)_{n+1} \geq (\lambda S)_n.
\]

From (3.10) we obtain

\[
(\lambda S)_n = \lambda + p_1(\lambda S)_{n-1}(P_{\text{bus}})_{n-1} + p_2vL_{n-1}
\]

and

\[
(\lambda S)_{n+1} = \lambda + p_1(\lambda S)_n(P_{\text{bus}})_n + p_2vL_n.
\]

Let us compare (4.3) and (4.4). \( \lambda, p_1, p_2, \) and \( v \) are constants. \( (\lambda S)_n \geq (\lambda S)_{n-1} \) by assumption. From (3.5), (3.2), and (3.3), it follows that \( P_{\text{bus}} \) is greater for that system where the input intensity is greater. Thus, \( (P_{\text{bus}})_n \geq (P_{\text{bus}})_{n-1} \).

Similarly from (3.4), (3.1), (3.2), and (3.3), it follows that the average length of queue is greater for that system where the input intensity is greater. Thus,

\[
L_n \geq L_{n-1}.
\]

From (4.1), (4.5), and (4.6), we obtain (4.2).

**Lemma 4.2.** Sequence \( \{(\lambda S)_n\} \) is bounded above.

**Proof.** Let us consider (3.10). Because of \( (\lambda S)_n \geq (\lambda S)_{n-1}, (P_{\text{bus}})_{n-1} \leq 1, L_{n-1} \leq m \), we have

\[
(\lambda S)_n \leq \lambda + p_1(\lambda S)_n + p_2vL_n.
\]

or

\[
(\lambda S)_n \leq (\lambda + p_2vL)/ (1 - p_1).
\]

5. VALIDATION OF THE APPROXIMATION MODEL

We used a GPSS/H simulation model of ACD service [FEIN 90] for validation of our approximation model. We have tested many systems with different combinations of parameters. For almost all tested systems, the maximal error does not exceed 5%, and in more than 50% of the cases, it is less than 3% (in all our cases \( c = 0.01 \)).

One of the results of simulation is represented in Figure 2. The ACD system has 20 agents and 22 access lines. The average service time is 210 seconds. The average impatience time, probabilities of retry after a busy signal and after an impatient hang-up are equal to 152 seconds, 0.81 and 0.72 correspondingly. The average number of new calls per hour for the ACD system is 400. The average time intervals for retry after a busy signal and after an impatient hang-up, which are not modeled in the approximation model but exist in the simulation model, are equal to 346 seconds and 769 seconds correspondingly. We ran 8 independent simulation runs for each experiment (GPSS/H has 8 independent random number generators), where each run lasts for 250000 simulation time (694 hours of real time). We obtained sample means and 95% confidence intervals for each experiment. Since the confidence interval is very narrow for all cases (less than 0.5%), we present only the average value of the simulation results. Figure 2 shows the error for a fraction of served clients versus the offered load, which we define as \( \rho = \)
\( \lambda \mu \) (to change the offered load, we changed the number of new arrivals to the system per hour in the simulation model). When \( \rho \) is smaller than 0.5, the results of our approximation model almost coincide with the results of simulation. This is a reflection of the simple fact that in light traffic there are almost no rejections or impatient clients, and the ACD system works as a conventional queueing system without loops. When \( \rho \) exceeds 0.5, the numbers of rejected and impatient clients increase and the errors increase as well. However, the error begins to decline starting with some value of \( \rho \), because so many calls "sit" in both loops that input stream of repeated calls is very close to Poisson process.

![Graph showing fraction of served clients vs offered load.](image)

**Figure 2.** Fraction of served clients vs offered load.

### 6. CONCLUSION

We have developed an approximation model and an iteration algorithm for the Automated Call Distribution system. The fraction of served clients, the average waiting time, and many other performance characteristics have been obtained. The comparison of numerical results with results of simulation shows that the approximation method, described in this work, may be successfully applied for obtaining characteristics of ACD systems.

### ACKNOWLEDGMENT

The author thanks Peter O'Reilly for useful discussions and comments.

### REFERENCES


