A SIMPLE MODEL FOR
REPEATED CALLS DUE TO TIME-OUTS

Pierre BOYER, Alain DUPUIS, Abdellatif KHELLADI

Centre National d'Etudes des Télécommunications
LAA/SLC/EVP, Route de Trégastel
22301 Lannion, FRANCE

Abstract In a distributed exchange, message-based cooperation between processors is
associated to a time-out mechanism for security reasons. After emission of a
question, a requesting processor waits for a reply a given amount of time which we call
its patience; if the reply is overlaid, the processor may re-emit another question -
thus performing a repetition - or abandon the desired session.

This paper addresses the topic of time-out dimensioning; this last appears to be
a trade-off between the two following behaviours:

- in case of message loss or defective required processor, patience is useless, calling
  for short values
- in case of system congestion, rapid repetitions cause extra-load, calling for large
  values

Here appears a queuing system with repeated attempts due to the impatience
of customers during their service. Little attention have been paid to such models in
previous teletraffic studies since repetitions are considered in the general framework
of blocking in a loss system.

This model has been solved numerically in the markovian case; its main queueing
characteristics have been computed for a wide range of parameters values and gath­
ered into traffic tables for design purposes.

1 - INTRODUCTION

In an exchange distributed operating system, processors cooperate to the set-up of the calls
using messages.

There always exists a kind of messages - say, question - by which a processor begins a session with
another; the translation phase provides an obvious example of such a session but questions also
appear in teleprocessing, adaptative routing, recovery .... [PeA84]

After having emitted a question, any processor waits for the answer - say, a reply - which may
be delayed for two reasons:

- the question or its reply is queued in the overloaded or defective required processor
- the question or its reply has been lost because of error transmission or saturation of a re-em­
mitting queue.

The time the processor may wait between the emission of a question and reception of its
reply must be obviously bounded.

System designer generally use a time-out mechanism: after a fixed amount of waiting time, the
processor is re-activated to re-emit another question or abandon the desired session; if the old
question has not been lost, it will be destroyed by layers protocols - with respect to time-stamps,
for example.

This mechanism avoids bursts of question: in the worst case, questions are periodically emitted;
this looks like the "token local fabric" of DOSHI and HEFFES [DoH83] for flow control; these
authors emphasize the fact that the reemission period value must be chosen with great care to ensure good performances.

Indeed, the time-out dimensioning is a key-element of system design. On the one hand, a too large value will lead to bad performances as losses or defective required processors will be detected too late; on the other hand, a too short value will cause an extra-load to the probably congestionned system.

Hence, dimensioning the time-out gives birth to an interesting problem of a queueing system with repeated questions.

2 - THE QUEUEING MODEL

In the distributed exchange, let us consider a node dedicated to a special processing - say TN, for translation node; this node concentrates a traffic of questions originated by all the other nodes and competing for their respective processing.

At the receiving side of TN, any question is transferred by the different layer protocols, then processed by the required processor which finally sends a reply back through the same layer protocols.

The question, and its reply will be modelled by the same unique customer. The service in TN is performed by a single server ruled according to the processor-sharing discipline to capture the essential phenomena of competing questions.

In order to begin a new session, a requiring processor emits a first question which can be considered as a primary attempt; successive other questions caused by the processor impatience will be considered re-attempts.

As an attribute, both primary attempts and re-attempts own a clock initialized with the time-out value when they depart from the originating node; this clock is given to the reply so that we may consider that each customer owns a time-out attribute.

A timed-out customer is assumed to give up at once its service; in fact, it would be actually destroyed by the layer protocols after some delay assumed to be small - see "Typical values" page 4.

At the same time, the requiring processor becomes impatient; it may decide to re-attempt after some overhead - or thinking - time, the customer spends in an infinite server queue; it may also abandon the desired session and the timed-out customer is destroyed. This perseverance phenomena is modelled by a Bernoulli trial.

This is a queueing system with repeated attempts due to the impatience of customers during their service. Little attention have been paid to such models in previous teletraffic studies since repetitions are considered in the general framework of blocking in a loss system. It has been fortunately possible to parallel the analysis of JONIN and SEDOL [JoS70] and develop an efficient algorithmic solution. Indeed, it must be noted that these queueing systems with losses and repeated attempts have not yet received a closed-form solution except in the case of one [LeG77], [LuR84] or two servers [Han87].

2.1 A two-station open markovian network

Station S2 represents the required node TN; incoming customers are processed by a single sever according the Processor-Sharing discipline. Each customer owns a limited patience defined as an upper bound on its service duration; when its patience is exhausted, the timed-out customer gives up its processing.

The impatient customer may quit the network with a fixed probability (1-h); otherwise, it joins the infinite-server station S1 where it stays for a thinking time; when this time is elapsed, the customer joins station S2 back for a re-attempt.

We assume all random durations to be exponentially distributed - for sake of simplicity, but the analysis easily extends to the case of one coxian law.

The state of this network can be described by a couple \((i,j)\) where \(i\) (resp. \(j\)) is the number of customers in station S1 (resp. S2).

When \(j \geq 1\), the probability of
• a service completion in S2 between t and t + dt is $\mu dt$
• an impatience occurring between t and t + dt is $j' dt$.

When $i \geq 1$, the probability of a departure from S1 between t and t + dt is $i\lambda dt$.

Primary attempt arrivals in S2 are assumed to be Poisson with rate $A$.

Owing to the customer impatience in station S2, BCMP theorem is not relevant and the steady-state probabilities of this small network have not yet been given a closed-form expression. In this paper, we use an efficient numerical method, already known as the global balance method [PuP79], [BDK87]; this method applies to finite capacity queues so that we have to modify slightly the above network.

2.2 Truncating the balance equations for numerical resolution

We assume that station S1 (resp. S2) may contain no more than $K1$ (resp. $K2$) customers; when faced to a saturated station, an arriving customer will be destroyed.

In the following, we shall choose the values of $K1$ and $K2$ large enough so that the probability of saturation of any queue will remain negligible - say, less than $10^{-10}$. Hence, truncating will have little effect and the resulting state-probabilities will be considered as an accurate approximation of those of the infinite-capacity network.

Let $P_{i,j}$ be the steady-state probability that there are $i$ customers in station S1 and $j$ customers in station S2; As a convention, we state that $P_{i,j} = 0$ if $i$ (resp. $j$) is negative or larger than $K1$ (resp. $K2$).

There follows the balance equations of the finite-capacity network (see also figure 1 above):

- 1st equation $0 \leq i < K1$, $0 < j < K2$
  \[(A + \mu + j.\tau + i.\lambda).P_{i,j} = A.P_{i,j-1} + [\mu + (1 - h).(j + 1).\tau].P_{i,j+1} + h.(j + 1).\tau.P_{i-1,j+1} + (i + 1).\lambda.P_{i+1,j-1}\]

- 2nd equation $0 \leq i \leq K1$, $j = K2$
  \[(\mu + K2.\tau + i.\lambda).P_{i,K2} = A.P_{i,K2-1} + (i + 1).\lambda.[P_{i+1,K2-1} + P_{i+1,K2}].1(i<K1)\]
2.3 Conservation laws

We note $P(i,j) = \sum_{n=0}^{K_2} P(i,n)$ and $P(i,j) = \sum_{n=0}^{K_2} P(i,n)$ the marginal probabilities with respect to $i$ and $j$ and $EN_1$ (resp. $EN_2$) the mean number of customers in station $S_1$ (resp. $S_2$)

$$EN_1 = \sum_{i=1}^{K_1} i.P(i,.), \quad EN_2 = \sum_{j=1}^{K_2} j.P(.,j)$$

The main network queueing characteristics may be obtained from its steady-state probabilities using conservation laws as Little's formula; but they refer to any customer - served or finally abandoning. As for loss systems, available statistics refering only to served attempts represents an open teletraffic problem at now.

Here follows the name and value of the customer flows rates inside the network

- $EPA = A.(1 - P(.,K_2))$: Rate of S2-Entering Primary Attempts flow (thanks to PASTA property [Wol82])
- $RPA = A.P(.,K_2)$: Rate of S2-Rejected Primary Attempt flow (finite-capacity model imbedded distortion)
- $SA = \mu.(1 - P(.,K_2))$: Rate of Served Attempts flow
- $IA = \sum_{j=1}^{K_2} j.\tau.P(.,j)$: Rate of Impatient Attempts flow
- $IA = \tau.EN_2$: Rate of Served Attempts flow
- $AA = (1 - h).IA$: Rate of Abandoning Attempts flow
- $RIA = \sum_{i=1}^{K_1} i.\tau.h.P(K_1,.)$: Rate of S1-Rejected Impatient Attempts flow (finite-capacity model imbedded distortion)
- $RRA = \sum_{i=1}^{K_1} i.\lambda.P(.,K_2)$: Rate of S2-Rejected Reattempts flow (finite-capacity model imbedded distortion)
- $ERA = \lambda.EN_1 - RRA$: Rate of S2-Entering ReAttempts flow

and the main network queueing characteristics:

- $P_{abn} = \frac{AA}{A}$: Abandonment probability
- $\beta = 1 + \frac{ERA}{EPA}$: Attempt repetition coefficient
- $ES = \frac{EN_2}{EPA + ERA}$: Mean service time in S2
- $EW = \frac{\beta - 1}{\lambda} + \beta ES$: Mean sojourn time in the system, between first arrival and final service or abandonment.
\[ \rho = \beta \frac{EPA}{\mu} \] : S2 single-server load or "system" load.

### 2.4 Limiting cases

For some limiting cases, an analytical solution is available and the reader is referred to [BDK88]; the most useful are the following:

- In case of light traffic \((\Lambda \approx 0)\), a tagged customer rambles in an empty network and its successive service times are independent. A closed-form solution for \(P_{abn}, \beta, ES, EW\) follows:

  \[
  P_{abn} = \frac{\tau}{\mu + \tau(1-h)}.
  \]

  \[
  \beta = \frac{\mu + \tau}{\mu + \tau(1-h)}.
  \]

  \[
  ES = \frac{1}{\mu + \tau}.
  \]

  \[
  EW = \frac{1}{\lambda} \frac{\lambda + \tau h}{\mu + \tau(1-h)}.
  \]

- Assuming an infinite patience duration, a requiring processor will never repeat its question; the required node behaves as a M/M/1 queue with Processor-Sharing service discipline. It follows that \(P_{abn} = 0, \beta = 1, ES = EW = 1/(\mu - A)\)

- Assuming a null patience duration, a requiring processor repeats continuously its questions until it gives up the desired session while the required node remains empty; it follows that \(P_{abn} = 1, ES = 0, P = 1, EW = \frac{h}{1 - h}\).

### 3 - THE EFFECT OF REPEATED ATTEMPTS

The processor impatience - think of a too short time-out - causes useless then repeated work of the system and results in a server extra-load; Moreover, although abandonment is possible, the customer mean sojourn time \(EW\) may be larger than experimented without repetition!

#### 3.1 Typical values

A question service duration in S2 involves propagation, layer protocols processing and reply assembling; the main delay is due to layer protocols: 5 ms per layer on a 68020 \(\mu\)P; assuming that question and reply are carried respectively through 3 layers, a typical service time is \(1/\mu = 35\) ms. When a requiring processor becomes impatient, it may decide to re-attempt after some overhead time - due to polling and swapping - a typical value of which is 1 to 5 ms - ie. \(7\mu \leq \lambda \leq 35\mu\).

The number of repetitions a processor may perform is generally deterministic and greater than \(R = 4\); assuming a Bernoulli trial - for markovian modelling - the perseverance probability \(h = R/(1 + R)\) is larger than 0.80.

We have computed the main network queueing characteristics when

- The primary attempt arrival rate \(A\) is ranging from 0.05\(\mu\) to 1.0\(\mu\) by step 0.05\(\mu\).
- The processor mean patience duration \(1/\tau\) is ranging from 20/\(\mu\) to 5/\(\mu\) by step 2/\(\mu\)

For design purposes, we have gathered these results into traffic tables available in [BDK88].

#### 3.2 On the system load

Figure 2 shows \(100(\beta - 1)\) vs. the mean processor patience duration \(1/\tau\) for different primary arrival rate values. Recall that \(\beta\) is the question repetition coefficient - ie the average number of attempts per primary attempt; then \(100(\beta - 1)\) is the percentage of extra-load due to question repetitions.
As a quite foreseeable behaviour, this percentage increases rapidly when the requiring processor patience duration decreases - situation becoming worse with higher primary attempt arrival rate $A$.

![Figure 2. System extra-load percentage vs. processor mean patience duration: $100(\beta - 1)$ for $A = 0.3, 0.6, 0.9$ with $\mu = 1.0$, $\lambda = 20.0$ and $h = 0.85$](image)

### 3.3 On the sojourn time

Figure 3 shows a customer network mean sojourn time - between the arrival of a primary attempt and its service completion or abandonment - vs. the mean processor patience duration $1/\tau$ for different repetition mean interarrival times $1/\lambda$.

Limits of $EW$ for the null and infinite patience duration cases have a closed-form expression - see “Limiting cases” page 4 - so that they may be assigned a given value; figure 3 shows three sojourn times taking the same value when the processor patience is infinite.

The sojourn time study highlights the nasty effect of repetitions on the system traffic behaviour: the more you repeat, the more you may wait!.. depending on the processor thinking time $1/\lambda$, the sojourn time $EW$ may be larger than experimented without repetition, although abandonment is possible.

This is the case for $\lambda = 1.0$; $EW$ is increasing while the processor patience $1/\tau$ decreases, although the abandonment probability increases: in this case, it is not worth repeating.

For $\lambda = 2.0$ the sojourn time has the same value ($EW = 2.0$) for both null or infinite patience; it is decreasing then increasing with the patience duration. For $\lambda = 8.0$, the sojourn time is steadily increasing with the processor patience, what should be reasonable as abandonments occur less frequently.

The reader is referred to [BDK88] for a complete discussion.

### 3.4 On the abandonment probability

Figure 4 shows the abandonment probability $P_{\text{abn}}$ vs. the mean processor patience duration $1/\tau$ for different primary attempt arrival rates $A$. As for the attempt repetition coefficient $\beta$, it is a run-of-the-mill behaviour: the abandonment probability increases rapidly when the processor mean patience duration decreases - situation becoming worse with higher primary arrival rate. As a
Figure 3. Mean sojourn time in the system vs. processor mean patience duration: between the primary attempt arrival and its service completion or abandonment epochs with $\mu = 1.0, A = 0.5, h = 0.8$

Figure 4. Abandonment probability vs. processor mean patience duration: $P_{ab}$ for $A = 0.3, 0.6, 0.9$ with $\mu = 1.0, \lambda = 20.0$ and $h = 0.85$
matter of fact, customers more and more often experience the abandonment Bernoulli trial when the processor patience decreases.

4 - CONCLUSION

The customer impatience during its service causes useless then repeated work of the server and results in an extra-load. This model applies to the time-out mechanism dimensioning used in distributed exchange operating systems. In the markovian case, its main queueing characteristics have been computed and gathered into traffic tables for design purposes. It must be noted for further studies, that these queueing systems with repeated attempts have not yet received a closed-form solution in the multi-server case; moreover, statistics referring only to finally-served customers are not yet available from algorithmic methods.

APPENDIX - REFERENCES


[DoH83] B.T. DOSHI, H. HEFFES: Analysis of overload control schemes for a class of distributed switching machines. ITC'10 June 1983 Montreal (Canada)


[JoS70] G.L. JONIN, J.J. SEDOL: Telephone systems with repeated calls ITC'6 1970 Munich (Germany)

[LeG77] P. LE GALL: La notion de qualité de service en téléphonie et les répétitions d'appels Commutation et Electronique n°59 1977


[PeA84] J-C. PETIT, J-P. ANDRE: GALAXIE: Towards adaptative distributed control systems ISS'84 May 1984 Firenze (Italy)
