Quaestiones Informaticae

Vol. 2 No. 3 September, 1983

Quaestiones Informaticae

An official publication of the Computer Society of South Africa and of the South African Institute of Computer Scientists

'n Amptelike tydskrif van die Rekenaarvereniging van Suid-Afrika en van die Suid-Afrikaanse Instituut van Rekenaarwetenskaplikes

Subscriptions

Annual subscriptions are as follows:

	SA	US	UK
Individuals	R6	$7	£3,0
Institutions	R12	$14	£6,0

Circulation manager
Mr E Anderssen
Department of Computer Science
Rand Afrikaanse Universiteit
P O Box 524
Johannesburg 2000
Tel.: (011) 726-5000

Quaestiones Informaticae is prepared for publication by Thomson Publications South Africa (Pty) Ltd for the Computer Society of South Africa and the South African Institute of Computer Scientists.

Modelling Blocking on Admission of Tasks to Computer Systems

S. Wulf
Comcon (Pty) Limited, Johannesburg

Abstract
The need for modelling techniques to allow blocking of tasks prior to admission to the computer system in queueing network models is described. A detailed algorithm which presents a technique to enforce blocking by priority of class for an arbitrary number of blocking constraints is presented. The resultant solution is exact and produces an analysis by class of admission queueing delays.

1. Introduction

Queueing network models [1] have been used extensively for modelling computer systems. Most of the modelling performed to date has been confined to construction of models to describe existing computer systems and extrapolate the growth of the component workloads so as to predict the effect of upgrading hardware components. In a few cases, attempts have been made to predict the effects of systems not yet implemented [2], [3] and [4]. The basic statistics required from queueing network models are response times per class together with the associated utilisations, waiting times and queue lengths at each component device of the system.

These statistics are consequent upon the accurate modelling of all system environmental components which contribute towards the overall response times. They include the data communications network, the user terminal characteristics and the behaviour of the computer system itself. In particular, the computer system component incorporates certain multiprogramming constraints by type of workload which affect overall response time. For example, a particular online system may only be capable of processing six tasks concurrently whereas the network volumes dictate the concurrent presence of ten tasks at the computer system. This means that a large proportion of the total number of tasks at the system must queue for admission into the system.

Unless the queueing network model used for analysing the system takes these constraints and resultant admission queueing into account, the resultant model output will not be realistic.

2. Approaches used to date

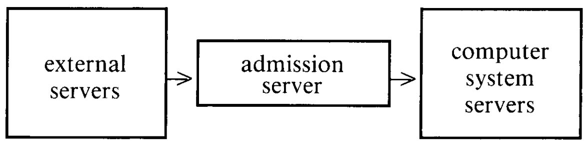
Decomposition [5] consists of analysing a submodel in isolation, (ie the computer system submodel component) and then replacing this subsystem in the original model by a single, composite load-dependent server which appears to the rest of the system to behave the same as the original subsystem. Such an approximation is accurate if the rate of interaction within the submodel is substantially higher than the rate of interactions between the submodel and the rest of the model. In the case under discussion, the submodel consists of computer system components with service times of the order of 10^{-2} seconds whereas the terminal network has service times of the order of 10 seconds. Consequently, decomposition has been widely used as a technique to model large terminal-based systems.

Although effective results can be derived using this approach, the multi-step nature of arriving at a model solution impedes the ease of formulation and solution of the model. Consequently, an alternative approach is needed to model blocking of tasks on admission from part of the queueing network (the external system) to the remaining part (the central computer system itself).

The approach to blocking presented here is based on a technique suggested in [6] and that presented in [4]. The latter reference contains a validation of the technique. We assume that the queueing network consists of a number of servers such that some are located external to the computer system and others are located within the computer system where the blocking constraints hold. We define an additional dummy server which serves as the system admission adjudicator. It initially has zero service time. The model is then solved. If the resultant solution is feasible in terms of the blocking constraints then the algorithm terminates. Otherwise, class-dependent service times are calculated so as to produce a feasible solution by delaying additional tasks at this admission adjudicator. The solution obtained at the end of this procedure is not only feasible — the extent of the delay prior to admission of tasks of each class can be measured.

3. Notation used in the Algorithm

We assume the the model topology is as follows:



where "external servers" refers to servers outside the computer system, "computer system servers" to those inside the computer system and "admission server" to the dummy server with initial class-dependent service times all set to zero.

Let K denote the number of classes and M be the number of queues (servers) in the network. We then define the notation used in the algorithm as follows:

N_k No. of class k jobs in the network for $k = 1, \ldots, K$
S_{mk} Mean service time of class k job at queue m.
v_{mk} Visit ratio of class k jobs at queue m.
Q_{mk} Mean no. of jobs of class k at queue m.
W_{mk} Mean waiting time of class k jobs at queue m.
T_{mk} Throughput of class k jobs at queue m.

We define further the following set of multiprogramming blocking constraints:

$$\sum_{r \epsilon C_c} \quad \sum_{\substack{m = \text{computer} \\ \text{system queues}}} Q_{mk} \leq P_c \qquad (1)$$

where P_c is a constant and C_c is a set of classes such that $k \epsilon C_c$ implies $k \epsilon \{1, \ldots, K\}$

Define the admission server to be device f so that external devices are such that $m = 1, \ldots, f-1$ and internal devices such that $m = f+1 \ldots, M$

Each blocking constraint (1) is defined for a given set of classes C_c. Each class C_c has a given priority G_k.

The following algorithm calculates values S_{fk} $(k = 1, \ldots, K)$ in such a way that if priority $G_r < G_s$ for $r,s \epsilon C_c$ then S_{fr} is increased before S_{fs}. (ie. lower priority classes are blocked on admission into the computer system before higher priority classes in order to satisfy given multiprogramming constraints).

4. The Blocking Algorithm

We assume the queue f is an infinite server device so that following [6]:

$$W_{fk} = S_{fk} \; v_{fk} \text{ for all } k = 1, \ldots, K \tag{2}$$

Initially $S_{fk} = 0$ for all $k = 1, \ldots, K$. Then by global application of Little's Theorem [7] :

$$T_{mk} = N_k \, v_{mk} / (\sum_{i \neq f} W_{ik} \, v_{ik} + S_{2k} \, v_{2k}) \tag{3}$$

By local application of Little's Theorem and (1) and (3) above:

$$\sum_{k \epsilon C_c} \sum_{m > f} Q_{mr} = \sum_{k \epsilon C_c} \sum_{m > f} (N_k v_{mk} W_{mk} / (\sum_{i \neq f} W_{ik} v_{ik} + S_{2k} v_{2k})) \tag{4}$$

The following algorithm is then applied to evaluate the S_{fk}:

1 $S_{fk} = 0$ for all $k = 1, \ldots, K$
2 Compute W_{mk} for all $m = 1, \ldots, M$ and $k = 1, \ldots K$
3 Sort constraints so that if C_{c_1} is a subset of C_{c_2} then constraint c_1 is processed before constraint c_2
4 For all constraints c

```
        For all priorities Gq of classes q∈C_c
        until H*_c ≤ P_c

          S¹ = 0
          H*¹_c = 0
          Do until H*_c ≤ P_c or H*_c = H*¹_c
H^c_r = Σ N_k v_mk W_mk /( Σ W_ik v_ik + S_2k v_2k)
                           m>f
                            i≠f
            for all k∈C_c
            H*_c = ΣH^c_k
                   k∈C_c
            If S¹ = 0
              S¹ = Σ H^c_k ( Σ W_mk W_mk v_mk)
                  r∈C_c  m>f
                  G_k = G_q
            else
              S¹ = S¹ (H*_c/P_c)
            Do for all k∈C_c such that G_k = G_q
              If S_2k < S¹
                S_2k = S¹
            Doend
          Doend
      Doend
  Doend
```

5. Conclusions

Execution of the above algorithm during solution of the model is iterative. Initially, no admission delays by class are assumed. If a feasible solution does not result then delay service times of an infinite server device are calculated so as to keep tasks outside the system. This continues until a feasible result is obtained. The result is not only intuitively understandable — a delay queue prior to admission but also exact in queueing network solution terms. The technique has been widely used [4] and can be extended to model blocking and adaptive routing in data communications networks.

References

[1] ACM Computing Surveys. 10, No. 3, 1978.

[2] SMITH C., and BROWNE J.C., Performance Specifications and Analysis of Software Designs, Conf. on Simulation, Measurement and Modelling of Computer Systems, 1979.

[3] BUZEN J.P., et al., Predicting Software Performance with Crystal, Third Annual Int. Conf. on Computer Capacity Management, 1981.

[4] WULF S., Performance Prediction During Database System Development, Ph.D. Dissertation, UNISA, 1982.

[5] COURTOIS P.J., Decomposability, Instabilities and Saturation in Multiprogramming Systems, Comm. ACM 18, No. 7, 1975.

[6] REISER M., and LAVENBERG S.S., Mean Value Analysis of Closed Multichain Queueing Networks, IBM Research Report, RC7023, 1978.

Notes for Contributors

The purpose of this Journal will be to publish original papers in any field of computing. Papers submitted may be research articles, review articles, exploratory articles of general interest to readers of the Journal. The preferred languages of the Journal will be the congress languages of IFIP although papers in other languages will not be precluded.

Manuscripts should be submitted in triplicate to: Prof. G. Wiechers at:

Department of Computer Science
University of South Africa
P.O. Box 392
Pretoria 0001
South Africa

Form of manuscript

Manuscripts should be in double-space typing on one side only of sheets of A4 size with wide margins. The original ribbon copy of the typed manuscript should be submitted. Authors should write concisely.

The first page should include the article title (which should be brief), the author's name, and the affiliation and address. Each paper must be accompanied by a summary of less than 200 words which will be printed immediately below the title at the beginning of the paper, together with an appropriate key word list and a list of relevant Computing Review categories.

Tables and figures

Illustrations and tables should not be included in the text, although the author should indicate the desired location of each in the printed text. Tables should be typed on separate sheets and should be numbered consecutively and titled.

Illustrations should also be supplied on separate sheets, and each should be clearly identified on the back in pencil with the Author's name and figure number. Original line drawings (not photoprints) should be submitted and should include all relevant details. Drawings, etc., should be submitted and should include all relevant details. Drawings, etc., should be about twice the final size required and lettering must be clear and "open" and sufficiently large to permit the necessary reduction of size in block-making.

Where photographs are submitted, glossy bromide prints are required. If words or numbers are to appear on a photograph, two prints should be sent, the lettering being clearly indicated on one print only. Computer programs or output should be given on clear original printouts and preferably not on lined paper so that they can be reproduced photographically.

Figure legends should be typed on a separate sheet and placed at the end of the manuscript.

Symbols

Mathematical and other symbols may be either handwritten or typewritten. Greek letters and unusual symbols should be identified in the margin. Distinction should be made between capital and lower case letters between the letter O and zero; between the letter l, the number one and prime; between K and kappa.

References

References should be listed at the end of the manuscript in alphabetical order of author's name, and cited in the text by number in square brackets. Journal references should be arranged thus:

1. ASHCROFT, E. and MANNA, Z. (1972). The Translation of 'GOTO' Programs to 'WHILE' Programs, in *Proceedings of IFIP Congress 71*, North-Holland, Amsterdam, 250-255.
2. BÖHM, C. and JACOPINI, G. (1966). Flow Diagrams, Turing Machines and Languages with only Two Formation Rules, *Comm. ACM*, **9**, 366-371.
3. GINSBURG, S. (1966). *Mathematical Theory of context-free Languages*, McGraw Hill, New York.

Proofs and reprints

Galley proofs will be sent to the author to ensure that the papers have been correctly set up in type and not for the addition of new material or amendment of texts. Excessive alterations may have to be disallowed or the cost charged against the author. Corrected galley proofs, together with the original typescript, must be returned to the editor within three days to minimize the risk of the author's contribution having to be held over to a later issue.

Fifty reprints of each article will be supplied free of charge. Additional copies may be purchased on a reprint order form which will accompany the proofs.

Only original papers will be accepted, and copyright in published papers will be vested in the publisher.

Letters

A section of "Letters to the Editor" (each limited to about 500 words) will provide a forum for discussion of recent problems.

Hierdie notas is ook in Afrikaans verkrygbaar.

Quaestiones Informaticae

Contents/Inhoud

*Presented at the second South African Computer Symposium held on 28th and 29th October, 1981.

MILITHO