Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Supervisory Control of a Database Unit

N. Eva Wu, *Senior Member, IEEE*, James M. Metzler, and Mark H. Linderman, *Member*, *IEEE*

***Abstract* — To effectively enhance service availability, this paper proposes a redundancy configuration for a database unit residing in a command and control (C2) system that supports air operations. The results of modeling, supervisory control, and performance analysis of the database unit are presented. The unit is modeled as a closed Markovian queuing network. State variable feedback is used to implement the functions of restoration and routing upon the identification of the failure of one of the database servers in the unit. Several control policies are evaluated in terms of the resulting mean time to unit failure, the steady state availability, the expected response time, and the service overhead of the database unit.**

I. Introduction

THE recent effort to install and test monitoring tools and to increase the level of redundancy in critical subsystems in air operation centers [1] has provided opportunities for vast performance improvement in its command and control (C2 hereafter) supporting systems. Our previous work on a controlled C2 processing unit [2] has demonstrated that reduced response time to service requests and shortened periods of system unavailability, as a result of automated monitoring and control, can raise significantly the probability to attain the desired outcome in an air operation. This paper shifts focus to one other critical C2 subsystem, a database unit. A simulation study [3] has been performed recently using Arena [4], [5] on a controlled database unit. The results indicate, however, that the architecture shown in Fig.1 is extremely inefficient, where the service burden rests almost entirely on the primary server, while the secondary server, though indispensable for the required system availability, is rarely utilized.

Fig.2 shows an alternative architecture for which the potential improvements in response time and in service availability are to be examined. The partition of the database into multiple sets of data (to be called data classes hereafter), and the simultaneous access to multiple servers allow the reduction of the response time to queries, whereas the presence of a secondary data class in every server leads to fault-tolerance and therefore higher service availability. The performance improvement, however, cannot be achieved in a cost-effective manner without a reconfiguration scheme called a supervisory control that acts on the state information of the database system. This effort investigates several such schemes that differ by their control authorities. To assess the effectiveness of these schemes in a quantified manner, the model in Fig.2 (and that in Fig.1) is given the interpretation of a queuing network [6] with specific sets of operating policies and structural parameters. The control authorities considered include the ability to restore the lost data and/or the ability to route queries. In order to obtain an analytic model of manageable size for scrutinizing the effects of supervisory control, the archiving process is ignored, and the queuing network is of the closed type [7]. A simulation study is being conducted currently without these simplifications.

The paper is organized as follows. Section II of the paper models the database system in Fig.2 as a Markov chain [8] with supervisory control. Section III evaluates a set of performance measures under several supervisory control policies. Section IV concludes the paper. Section V acknowledges the contributions from our colleagues. Details of the database model are given in Appendix.

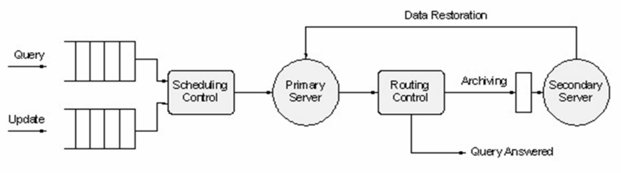

Fig.1 Redundant database unit

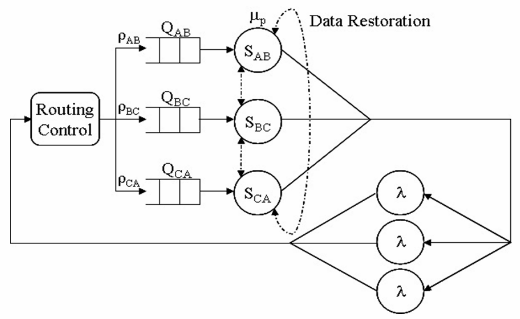

Fig.2 Partitioned database unit

II. Modeling and Control

A. Modeling

The database unit in Fig.2 contains three servers in parallel to answer three classes (A, B, C) of queries for which relevant information can be found in the partitioned

Manuscript received March 1, 2005. This work was supported in part by the U.S. Air Force Research Laboratory Contract F30602-020-C-0225.

N. Eva Wu is with the Department of Electrical and Computer Engineering, Binghamton University, Binghamton, New York, 13902-6000, USA. (telephone: 607-777-4375; fax: 607-777-4464; e-mail: evawu@binghamton.edu)
James M. Metzler is with the Department of Electrical and Computer Engineering, Binghamton University, Binghamton, NY 13902-6000, USA. (e-mail: james.metzler@binghamton.edu
Mark H. Linderman is with the Information Directorate of the Air Force Research Laboratories, Rome Research Site, Rome, NY, 13441-4505, USA. (e-mail: mark.linderman@rl.af.mil)

sets A, B, C of the database, respectively. Server S_{AB} contains database class A as the primary class and database class B as the secondary class. Server S_{BC} contains database class B as the primary class and database class C as the secondary class. Server S_{CA} contains database class C as the primary class and database class A as the secondary class. The failure of a server implies the loss of two classes of data within the server. A system level failure is declared when two servers fail, in which case one class of data is said to be lost. The queues preceding servers S_{AB}, S_{BC}, and S_{CA} are named Q_{AC}, Q_{BC}, and Q_{CA}, respectively. All queues are of sufficient capacity. Service is provided on a FCFS basis at each server.

The three delay elements imply that there are always three customers present in the unit at any given time. A new query is generated at a delay element upon the completion of the service to a query at one of the servers. The delay elements are intended to be also reflective of the response time to the querying customers by other service nodes in the C2 supporting system, which are not explicitly modeled. Any new query is assumed to be equally likely to seek database class A or B or C. Therefore routing probabilities ρ_{AB}, ρ_{BC}, and ρ_{CA} are assigned the same values under the normal operation condition.

The use of a queuing network model for the database is based on its suitability to involve control actions and our intention to capture their effects on the system performance. The model is built in this study with the premise that event life distributions have been established for the process of query generation $(\exp(\lambda) \equiv 1-e^{-\lambda t})$, the process of service completion $(\exp(\mu))$, the process of server failure $(\exp(\nu))$, the process of data restoration $(\exp(\gamma))$, and the process of unit overhaul $(\exp(a))$ when the failed database unit is repaired. All such processes are independent. Standard statistical methods that involve data collection, parameter estimation, and goodness of fit tests [9] exist for identifying event life distributions. Since all event lives are assumed to be exponentially distributed, the database unit can be conveniently modeled as a Markov chain specified by a state space $\mathcal{X}$, an initial state probability mass function (pmf) $\pi_x(0)$, and a set of state transition rates Λ [7], [8]. The reader uninterested in the details of model building can advance to the paragraph right above Equation (1).

1) *State space* $\mathcal{X}$

A state name is coded with a 6-digit number indicative of all queue lengths and server states in the unit. With some abuse of notations, a valid state representation is given by $x=Q_{AB}Q_{BC}Q_{CA}S_{AB}S_{BC}S_{CA}$, where queue length Q_{AB}, Q_{BC}, $Q_{CA} \in \{0, 1, 2, 3\}$ with total length $L \equiv Q_{AB}+Q_{BC}+Q_{CA} \leq 3$, and server state $S_{AB}, S_{BC}, S_{CA} \in \{0, 1, 2\}$. Server state "$2$" ≡ data are lost in both the primary and the secondary classes in a server, "1" ≡ the data in the primary class have been restored and data in the secondary class have not been restored, and "0" ≡ data in both primary class and secondary class in a server are intact. A server is said to be in the down state if it is either at state "1" or at state "2". For example, state *110020* indicates that server S_{AB} is up with one customer in its queue, server S_{BC} is down with both classes of data gone and one customer in its queue, and server S_{CA} is up and idle. Note that the queue length includes the customer being served. There are *540* valid states in the system. The total number of states is reduced to *147* when the states of system level failures are aggregated. The symmetry of the system permits the arrangement of customers in the queues at the time of system level failure to be captured in one of seven states, allowing the system to return to an equivalent state upon completion of the system overhaul. A set of alternative state names are assigned from $\mathcal{X} = \{1, 2, \ldots, 147\}$ with *000000* mapped to $x = 1$ and the aggregated system failure states mapped to $x \in \{141, 142, 143, 144, 145, 146, 147\}$.

2) *Initial state pmf* $\{\pi_x(0), x = 1,2,\ldots,147\}$

It is assumed that the database unit starts operation from state $x = 1$, i.e., the initial state probability is given by vector $\pi(0) = [1\ 0\ \ldots\ 0]$. When overhaul is considered at the occurrence of a system level failure, the system returns to a state with an equivalent arrangement of customers in the queues once the database unit is renewed [8] and ready for operation again.

3) *Set of state transition rates* Λ

A transition rate table containing all transition rates is created following a similar procedure as that described in [10], however with a more compact representation. The state transition table is given in Appendix. The list of current states occupies the first column of the table. In the row corresponding to each state, the set of all feasible next states are listed with each next state followed by the rate at which the next state is reached. Events that trigger the transitions and the corresponding transition rates are given as follows. A newly generated query enters one of the servers with rate $\rho^{u_2}(3-L)\times\lambda$ where ρ^{u_2} is a controlled routing probability by control variable u_2. A query is answered at a server with rate μ. A complete data loss occurs at a server with rate ν. Data in the primary data class of a server are restored with rate $\gamma_p\, u_1$ where u_1 authorizes whether to restore the lost data. Data in the secondary data class of a server are restored with rate $\gamma_s\, u_1$. Finally, the failed database unit is renewed with rate $\omega\, u_3$, where u_3 decides whether to repair the failed system. All rates are relative, for their net effects depend on the time unit specified.

Let $X \in \mathcal{X}$ denote the random state variable at time t. The set of state transition functions

$$p_{i,j}(t) \equiv P[X(t) = j \mid X(0) = i], i, j = 1,2,\cdots,147 \tag{1}$$

for the continuous-time Markov chain can be solved from the forward Chapman-Kolmogorov equation [7]

$$\dot{P}(t) = P(t)Q,\ P(0) = I,\ P(t) = [p_{i,j}(t)]\ , \tag{2}$$

where Q is called an infinitesimal generator or a rate transition matrix whose (i,j)th entry is given by the rate associated with the transition from current state i to next state j in the rate transition table. State probability mass function at time t

$$\pi(t)=[\pi_1(t) \quad \pi_2(t) \quad \cdots \quad \pi_{147}(t)],\, t\geq 0 \tag{3}$$

is computed by

$$\pi(t)=\pi(0)P(t)\,. \tag{4}$$

At this point a Markov model for the database unit of Fig.2 has been established. The state probabilities are the basis for evaluating the performance of the database unit, which is conducted in Section III.

B. *Control policies*

Our ultimate goal is to eliminate all single point failures, and to mitigate the effects of a single server failure on the performance of the database unit. Our approach is to base the supervisory control actions on the state information, which effectively alter the transition rates when loss of data occurs in a single server.

Taking into consideration the symmetry of the model, the control policy is described only for the case of a failed server S_{AB}. When routing control is effective, the routing probabilities are determined by the state of S_{AB} and by whether the lost data can be restored. Thus, $\rho^{u_2}=\rho_{AB}(S_{AB},u_1),\;\; \rho_{BC}(S_{AB},u_1),\;\; \rho_{CA}(S_{AB},u_1)$ with $\rho_{AB}+\rho_{BC}+\rho_{CA}=1$. The control policies considered for this study are summarized as follows.

$$u_1=\begin{cases}0, & S_{AB}=2, S_{BC}\text{ serves}, S_{CA}\text{ serves (no restoration)}\\ 1, & \begin{cases}S_{AB}=2, S_{BC}\text{ serves}, S_{CA}\text{ restores class A data}\\ S_{AB}=1, S_{CA}\text{ serves}, S_{BC}\text{ restores class B data}\end{cases}\end{cases}, \tag{5}$$

$$u_2=\begin{cases}0, & S_{AB}=2,\, \rho_{AB}=\rho_{BC}=\rho_{CA}=\dfrac{1}{3}\\ 1, & \begin{cases}S_{AB}=2,\, \rho_{AB}(2,u_1), \rho_{BC}(2,u_1), \rho_{CA}(2,u_1)\\ S_{AB}=1,\, \rho_{AB}(1,u_1), \rho_{BC}(1,u_1), \rho_{CA}(1,u_1)\end{cases}\end{cases}. \tag{6}$$

Four sets of routing probabilities are shown in the following table as examples, where $S_{BC}=0$ and $S_{CA}=0$ are assumed.

Table 1 Examples of routing probabilities

u_1	u_2	S_{AB}	ρ_{AB}	ρ_{BC}	ρ_{CA}
0	*1*	*2*	*0*	*1/2*	*1/2*
1	*0*	*2 (1)*	*1/3(1/3)*	*1/3(1/3)*	*1/3(1/3)*
1	*1*	*2 (1)*	*0 (1/6)*	*2/3(1/6)*	*1/3(2/3)*
1	*1*	*2 (1)*	*0 (0)*	*1 (0)*	*0 (1)*

The composition of u_1 and u_2 gives rise to four different control policies. The case of $(u_1, u_2) = (0, 0)$ corresponds to the case of a single point failure, and is therefore not considered in the performance analysis. The control policies in the other three cases are named

$$\begin{aligned}&\text{Policy 1: } (u_1, u_2) = (0, 1) \text{ when a server is down,}\\ &\text{Policy 2: } (u_1, u_2) = (1, 0) \text{ when a server is down,}\\ &\text{Policy 3: } (u_1, u_2) = (1, 1) \text{ when a server is down.}\end{aligned} \tag{7}$$

Note that policy 2 does not permit routing, whereas policy 1 does not permit restoring. As can be seen, policy 3 allows variations in the routing probabilities to the intact servers. A special consideration with the case $u_1=0$ is the rerouting of the customers who have arrived at a server before the server fails to the delay elements.

The presence of supervisory control in the transition rate table is seen via u_1, u_2, u_3, $n_1 = 1\text{-}u_1$, $n_2 = 1\text{-}u_2$, and $n_3 = 1\text{-}u_3$. The values of u_1, u_2, u_3 represent specific control actions associated with data restoration, query routing, and unit overhaul, respectively.

III. Performance Analysis

A. *Time to system failure*

When $u_3 = 0$, the Markov chain model for the database unit contains seven absorbing states $x\in\{141, 142, 143, 144, 145, 146, 147\}$ at which the chain remains forever once it is entered. These are the states of system level failure. The rest of the 140 states are transient states. Decompose the state probability vector

$$\pi(t)\equiv[\underbrace{\pi_\tau(t)}_{1\times 140} \quad \underbrace{\pi_\alpha(t)}_{1\times 7}], \tag{8}$$

where vector $\pi_\tau(t)$ contains the transient state probabilities, and $\pi_\alpha(t)$ are the absorbing state probabilities. Decomposing the rate transition matrix Q and the state transition function matrix $P(t)$ solved from (2) accordingly yields

$$Q=\begin{bmatrix}Q_{11} & Q_{12}\\ 0 & 0\end{bmatrix},\, P(t)=\begin{bmatrix}P_{11}(t) & P_{12}(t)\\ 0 & I\end{bmatrix}. \tag{9}$$

From (2), (4), and (9), it can be determined that the probability density function of time to system failure, or time to absorption, is given by

$$\dot{\pi}_\alpha(t)=\pi_\tau(0)P_{11}(t)Q_{12},\, \pi_\alpha(0)=0, \tag{10}$$

where

$$\pi_\tau(0)=[1 \quad 0 \quad \cdots],\; P_{11}(t)=e^{Q_{11}t}. \tag{11}$$

In addition, the mean time to failure of the database unit can be shown to be [8].

$$MTTF=-\pi_\tau(0)Q_{11}^{-1}1_\tau,\, 1_\tau=[1 \quad \cdots \quad 1]^T \tag{12}$$

Fig.3 below shows the dependence of mean time to failure of the database unit on the restoration rate.

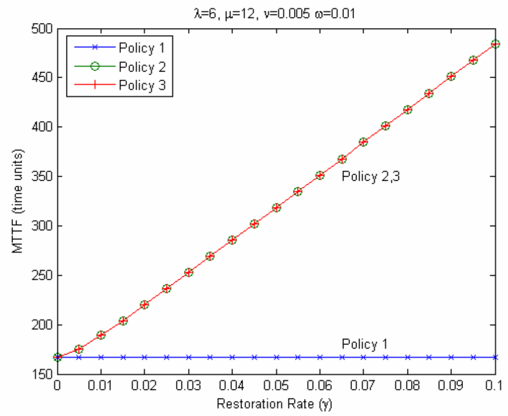

Fig.3 Database unit mean time to failure versus restoration rate

B. *Steady-state availability*

Suppose as soon as the database unit reaches a system level failure, an overhaul process starts. Suppose with a rate

ω the unit is repaired, and at the completion of the repair, the unit immediately starts to operate again. In this case u_3 is set to 1 in the model, whereas it is set to 0 in the case of an absorbing chain. The existence of a unique steady-state distribution of the Markov chain when $u_3=1$ is guaranteed if the chain is irreducible (or ergodic) [7]. Ergodicity is satisfied under policy 2 and policy 3. Although ergodicity is not met under policy 1 without eliminating the few unreachable states in this case, a unique steady state distribution is obtained nevertheless in our computation. The steady state availability, which can be roughly thought of as the fraction of time the database unit is up, is given by

$$A_{sys} = 1 - \pi_F(\propto), \qquad (13)$$

where $\pi_F(\propto)$ is the sum of the system level failure state probabilities, determined by solving

$$\pi(\infty)Q = 0, \text{ and } \sum_{x=1}^{147} \pi_x(\infty) = 1. \qquad (14)$$

Fig.4 shows the steady-state availability as a function of restoration rate at a fixed overhaul rate. Fig.6 demonstrates the benefit of the success in supervisory control to steady-state availability.

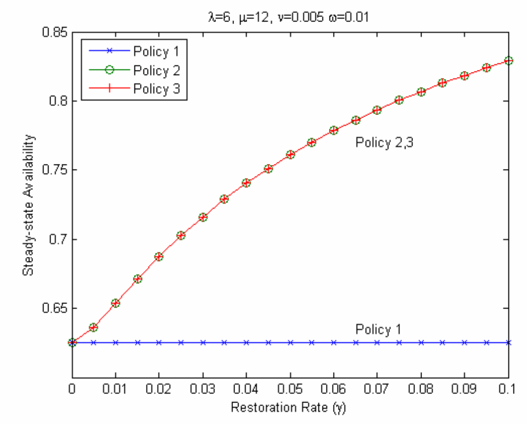

Fig.4 Steady-state availability of the database unit versus restoration rate

C. Response time

The average response time $E[R]$ is the expectation of the ratio of total amount of time that all customers spend in the upper portion of the system to the number of customers that are serviced. A loose argument is given below to justify the way $E[R]$ is computed in this paper. Define the vector C where $c(i)$ is the number of customers in the system at state i. The numerator of $E[R]$ is then $\pi(\infty)Ct$. Computing the number of customers that are serviced requires counting the number of transitions from one state to another that have occurred that have introduced a new customer to the system. Define a matrix N such that $n(i,j)$ is equal to the number of customers introduced into the system when the system transitions from state i to state j. The total number of transitions for a given i and j is then

$$T(i,j) = tN(i,j)\pi_i(\propto)Q(i,j). \qquad (15)$$

Therefore, the average response time $E[R]$ of the system is taken as

$$\frac{\pi(\propto)Ct}{\sum_{i=1}^{147}\sum_{j=1}^{147} T(i,j)} = \frac{\pi(\propto)C}{\sum_{i=1}^{147}\sum_{j=1}^{147} N(i,j)\pi_i(\infty)Q(i,j)}. \qquad (16)$$

Fig.5a and Fig.6 show the average response time as a function of restoration rate with the overhaul rate fixed, and a function of overhaul rate with the restoration rate fixed, respectively, for all three policies. The routing probabilities in rows 1 through 3 in Table 1 are in fact used for calculating all performance measures resulting from Policies 1 through 3, respectively. Policy 1 enjoys a lower response time because the intact servers need not deny customers in order to restore the failed server. Also, customers present at the time of server failure in policy 1 are emptied into the delay elements and incur no response time gains.

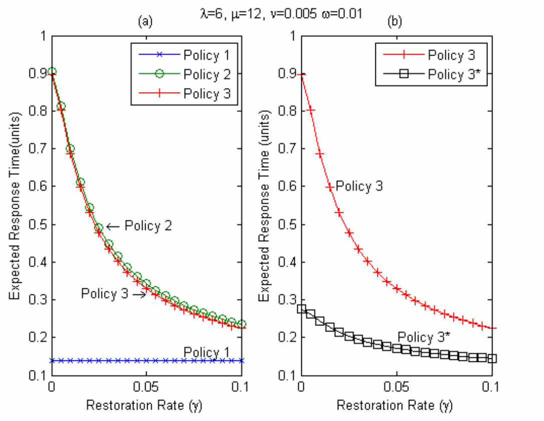

Fig.5 Average query response time versus restoration rate

Fig.5b shows the effect of applying Policy 3^*: routing all customers to the intact server that is not restoring the failed server, an alternative to Policy 3. The reduced response time in policy 3^* results from customers not waiting at a failed server. This policy may not be as advantageous in a system of higher traffic intensity.

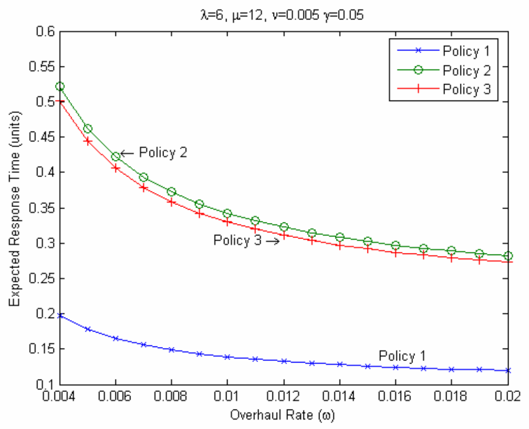

Fig.6 Average query response time versus overhaul rate

D. Overhead

Overhead is a quantity introduced to reflect the ratio of the time invested on helping the database unit to survive longer to its overall busy time. It is a measure of the cost of supervisory control. More specifically,

$$\theta \equiv \frac{\Pr[S_{AB} \text{ restores or fails} \mid \text{unit is not failed}]}{\Pr[S_{AB} \text{ restores or fails or serves} \mid \text{unit is not failed}]} \qquad (17)$$

Overhead θ is calculated for both the absorbing chain ($u_3 = 0$) as a function of time, and the irreducible chain ($u_3 = 1$) as a function of server failure rate. These are shown in Fig.7 and Fig.8. In Fig.7, it is seen that restoration incurs a higher overhead in the early life of the unit. As the database unit ages, its server becomes more likely to fail. A control policy that permits restoration becomes advantageous. There is a reduction in overhead across all polices with an increase in the arrival rate because of the resulting increased utilization. In Fig.8, for sufficiently low server failure rate, overhead is always lower with restoration. When server failure rate passes some threshold, however, restoration becomes expensive. Overhead is expected to gain more significance as a function of time and a function of server failure rate when the server life distributions have an increasing failure rate, such as in the case of Weibull distribution.

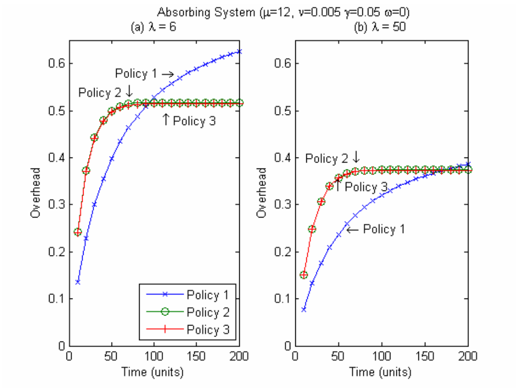

Fig.7 Overhead versus time in the absorbing chain

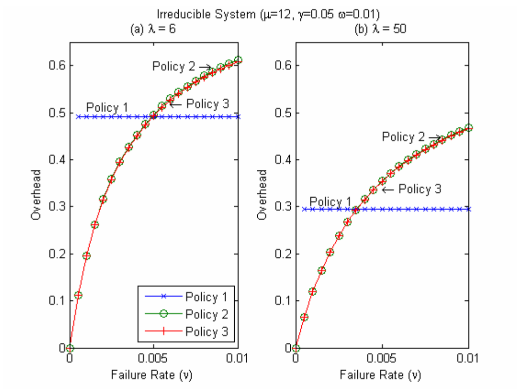

Fig.8 Overhead versus failure rate in the irreducible chain

IV. Conclusions

This paper modeled a redundant database unit in C2 for investigation of fault-tolerance and responsiveness afforded by a set of supervisory control policies. In all the performance measures examined, restoration (u_1) is more effective than routing (u_2). It is expected that when the number of queries increase, or the traffic becomes more intensive, the effectiveness of routing will be more apparent.

The study presented in this paper is limited by our ability to deal with complex problems analytically. Most restrictive is the size of the state space. The closed-queuing network model shown in Fig.2 presents perhaps the smallest possible state space for which the investigation on control policies is nontrivial. Besides answering queries, the database unit also must be updated from time to time. In that case, two types of service requests exist and the state space must be expanded. Almost equally restrictive is the assumption that times to event occurrence are exponentially distributed. Since there is only one parameter in an exponential distribution, it is likely to be unsuitable to truthfully describe some of the processes. Discrete event simulations are being carried out where the simplifying assumptions are removed to substantiate our claims on the benefit of supervisory control under more general settings in terms of the types of services, the number of customers, and the types of distributions of event lives.

Also ongoing is the extension of this study to incorporate the effect of decision and control under uncertainty and time delay due to, for example, incomplete state information and the time required for state estimation, respectively. The results will be reported in a future paper.

Acknowledgment

Both authors thank Ms. Sudha Thavamani, a student at Binghamton University pursuing her Ph.D. degree under N. Eva Wu, for her assistance in the simulations with Arena [3] that supported the preliminary study [2] of the database unit shown in Fig.1. N. Eva Wu also thanks Dr. Timothy Busch at the AFRL Rome Research Site for his insights in many sessions of discussion on the issue of architecture of the command and control supporting systems.

References

[1] N. E. Wu, and T. Busch, "Operational reconfigurability in command and control," *Proc. American Control Conference*, 2004.
[2] N. E. Wu, and T. Busch, "An example of supervisory control in C2," *Proc. IEEE Conference on Decision and Control*, 2004.
[3] N.E. Wu, and J. M. Metzler, "Reconfigurability in command and control systems: data loss prevention in a redundant database unit," *Report* to *AFRL RRS*, 2005.
[4] Rockwell Software, Inc. *Arena,* Academic Version 7.01.00, 2004.
[5] W. D. Kelton, R. P. Sadowski and D. T. Sturrock, *Simulation with Arena*, 3rd Ed., McGraw-Hill, 2004.
[6] K.S. Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications*, Prentice-Hall, 1982.
[7] C.G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, Kluwer, 1999.
[8] E.P.C., Kao, *An Introduction to Stochastic Processes*, Duxbury Press, 1997.
[9] S. Zacks, *Introduction to Reliability Analysis: Probability Models and Statistics Methods*, Springer-Verlag, 1992.
[10] N. E. Wu, "Coverage in fault tolerant control," *Automatica*, vol.40, 2004, pp.537-548.

APPENDIX

Table 2 Transitions and transition rates of the database unit model with all rates valid for all policies, based on which matrix Q is formed

		States						Arrivals						Completions						Failures											Restorations			
	State #	Q_{AB}	Q_{BC}	Q_{CA}	S_{AB}	S_{BC}	S_{CA}	λ_1	R	λ_2	R	λ_3	R	μ_p			R	μ_p	R	v			R	ReRoute		R	System Failure			R	γ_p	R	γ_s	R
	1	0	0	0	0	0	0	2	h	3	h	4	h	x	x	x	x	x	x	14	15	16	v	x	x	x	x	x	x	x	x	x	x	x
Ones	2	1	0	0	0	0	0	11	2/3*h	8	2/3*h	9	2/3*h	1	x	x	mp	x	x	36	37	38	(u1*n2+u1*u2)*v	37	38	n1*u2*v	142	37	38	n1*n2*v	x	x	x	x
	3	0	1	0	0	0	0	8	2/3*h	12	2/3*h	10	2/3*h	1	x	x	mp	x	x	39	40	41	(u1*n2+u1*u2)*v	15	41	n1*u2*v	39	142	41	n1*n2*v	x	x	x	x
	4	0	0	1	0	0	0	9	2/3*h	10	2/3*h	13	2/3*h	1	x	x	mp	x	x	42	43	44	(u1*n2+u1*u2)*v	43	16	n1*u2*v	42	43	142	n1*n2*v	x	x	x	x
	5	0	0	0	1	0	0	17	(u1*n2*1/3 + u1*u2*1/6)*h	20	(u1*n2*1/3 + u1*u2*1/6)*h	23	(u1*n2*1/3 + u1*u2*2/3)*h	x	x	x	x	x	x	141	x	x	2*v	x	x	x	x	x	x	x	x	x	1	u1*gs
	6	0	0	0	0	1	0	18	(u1*n2*1/3 + u1*u2*2/3)*h	21	(u1*n2*1/3 + u1*u2*1/6)*h	24	(u1*n2*1/3 + u1*u2*1/6)*h	x	x	x	x	x	x	141	x	x	2*v	x	x	x	x	x	x	x	x	x	1	u1*gs
	7	0	0	0	0	0	1	19	(u1*n2*1/3 + u1*u2*1/6)*h	22	(u1*n2*1/3 + u1*u2*2/3)*h	25	(u1*n2*1/3 + u1*u2*1/6)*h	x	x	x	x	x	x	141	x	x	2*v	x	x	x	x	x	x	x	x	x	1	u1*gs
Twos	8	1	1	0	0	0	0	50	1/3*h	48	1/3*h	26	1/3*h	2	3	x	mp	x	x	66	67	68	(u1*n2+u1*u2)*v	37	68	n1*u2*v	143	143	68	n1*n2*v	x	x	x	x
	9	1	0	1	0	0	0	49	1/3*h	26	1/3*h	47	1/3*h	2	4	x	mp	x	x	72	73	74	(u1*n2+u1*u2)*v	73	38	n1*u2*v	143	73	143	n1*n2*v	x	x	x	x
	10	0	1	1	0	0	0	26	1/3*h	46	1/3*h	45	1/3*h	3	4	x	mp	x	x	69	70	71	(u1*n2+u1*u2)*v	43	41	n1*u2*v	69	143	143	n1*n2*v	x	x	x	x
	11	2	0	0	0	0	0	60	1/3*h	50	1/3*h	49	1/3*h	2	x	x	mp	x	x	93	94	95	(u1*n2+u1*u2)*v	94	95	n1*u2*v	145	94	95	n1*n2*v	x	x	x	x
	12	0	2	0	0	0	0	48	1/3*h	61	1/3*h	46	1/3*h	3	x	x	mp	x	x	96	97	98	(u1*n2+u1*u2)*v	15	98	n1*u2*v	96	145	98	n1*n2*v	x	x	x	x
	13	0	0	2	0	0	0	47	1/3*h	45	1/3*h	62	1/3*h	4	x	x	mp	x	x	99	100	101	(u1*n2+u1*u2)*v	100	16	n1*u2*v	99	100	145	n1*n2*v	x	x	x	x
	14	0	0	0	2	0	0	36	(u1*n2*1/3)*h	39	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*h	42	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*h	x	x	x	x	x	x	141	x	x	2*v	x	x	x	x	x	x	x	5	u1*gp	x	x
	15	0	0	0	0	2	0	37	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*h	40	(u1*n2*1/3)*h	43	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*h	x	x	x	x	x	x	141	x	x	2*v	x	x	x	x	x	x	x	6	u1*gp	x	x
	16	0	0	0	0	0	2	38	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*h	41	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*h	44	(u1*n2*1/3)*h	x	x	x	x	x	x	141	x	x	2*v	x	x	x	x	x	x	x	7	u1*gp	x	x
	17	1	0	0	1	0	0	51	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	27	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	30	(u1*n2*1/3 + u1*u2*2/3)*2/3*h	x	x	x	x	x	x	142	x	x	2*v	x	x	x	x	x	x	x	x	x	2	u1*gs
	18	1	0	0	0	1	0	52	(u1*n2*1/3 + u1*u2*2/3)*2/3*h	28	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	31	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	6	x	x	mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	x	x	2	u1*gs
	19	1	0	0	0	0	1	53	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	29	(u1*n2*1/3 + u1*u2*2/3)*2/3*h	32	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	7	x	x	n1*mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	x	x	2	u1*gs
	20	0	1	0	1	0	0	27	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	54	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	33	(u1*n2*1/3 + u1*u2*2/3)*2/3*h	5	x	x	n1*mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	x	x	3	u1*gs
	21	0	1	0	0	1	0	28	(u1*n2*1/3 + u1*u2*2/3)*2/3*h	55	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	34	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	x	x	x	x	x	x	142	x	x	2*v	x	x	x	x	x	x	x	x	x	3	u1*gs
	22	0	1	0	0	0	1	29	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	56	(u1*n2*1/3 + u1*u2*2/3)*2/3*h	35	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	7	x	x	mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	x	x	3	u1*gs
	23	0	0	1	1	0	0	30	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	33	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	57	(u1*n2*1/3 + u1*u2*2/3)*2/3*h	5	x	x	mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	x	x	4	u1*gs
	24	0	0	1	0	1	0	31	(u1*n2*1/3 + u1*u2*2/3)*2/3*h	34	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	58	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	6	x	x	n1*mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	x	x	4	u1*gs
	25	0	0	1	0	0	1	32	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	35	(u1*n2*1/3 + u1*u2*2/3)*2/3*h	59	(u1*n2*1/3 + u1*u2*1/6)*2/3*h	x	x	x	x	x	x	142	x	x	2*v	x	x	x	x	x	x	x	x	x	4	u1*gs
Threes	26	1	1	1	0	0	0	x	0	x	0	x	0	8	9	10	mp	x	x	111	112	113	(u1*n2+u1*u2)*v	73	68	n1*u2*v	144	144	144	n1*n2*v	x	x	x	x
	27	1	1	0	1	0	0	75	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	87	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	63	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	17	x	x	n1*mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	x	x	8	u1*gs
	28	1	1	0	0	1	0	76	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	88	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	64	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	21	x	x	mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	x	x	8	u1*gs
	29	1	1	0	0	0	1	77	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	89	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	65	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	22	19	x	n1*mp	19	u1*mp	143	x	x	2*v	x	x	x	x	x	x	x	x	x	8	u1*gs
	30	1	0	1	1	0	0	78	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	63	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	90	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	17	x	x	mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	x	x	9	u1*gs
	31	1	0	1	0	1	0	79	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	64	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	91	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	24	18	x	n1*mp	24	u1*mp	143	x	x	2*v	x	x	x	x	x	x	x	x	x	9	u1*gs
	32	1	0	1	0	0	1	80	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	65	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	92	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	25	x	x	n1*mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	x	x	9	u1*gs
	33	0	1	1	1	0	0	63	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	81	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	84	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	23	20	x	n1*mp	20	u1*mp	143	x	x	2*v	x	x	x	x	x	x	x	x	x	10	u1*gs
	34	0	1	1	0	1	0	64	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	82	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	85	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	21	x	x	n1*mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	x	x	10	u1*gs
	35	0	1	1	0	0	1	65	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	83	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	86	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	25	x	x	mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	x	x	10	u1*gs
	36	1	0	0	2	0	0	93	(u1*n2*1/3)*2/3*h	66	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*2/3*h	72	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*2/3*h	x	x	x	x	x	x	142	x	x	2*v	x	x	x	x	x	x	x	17	u1*gp	x	x
	37	1	0	0	0	2	0	94	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*2/3*h	67	(u1*n2*1/3)*2/3*h	73	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*2/3*h	15	x	x	n1*mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	18	u1*gp	x	x
	38	1	0	0	0	0	2	95	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*2/3*h	68	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*2/3*h	74	(u1*n2*1/3)*2/3*h	16	x	x	mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	19	u1*gp	x	x
	39	0	1	0	2	0	0	66	(u1*n2*1/3)*2/3*h	96	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*2/3*h	69	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*2/3*h	14	x	x	mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	20	u1*gp	x	x
	40	0	1	0	0	2	0	67	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*2/3*h	97	(u1*n2*1/3)*2/3*h	70	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*2/3*h	x	x	x	x	x	x	142	x	x	2*v	x	x	x	x	x	x	x	21	u1*gp	x	x
	41	0	1	0	0	0	2	68	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*2/3*h	98	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*2/3*h	71	(u1*n2*1/3)*2/3*h	16	x	x	n1*mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	22	u1*gp	x	x
	42	0	0	1	2	0	0	72	(u1*n2*1/3)*2/3*h	69	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*2/3*h	99	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*2/3*h	14	x	x	n1*mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	23	u1*gp	x	x
	43	0	0	1	0	2	0	73	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*2/3*h	70	(u1*n2*1/3)*2/3*h	100	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*2/3*h	15	x	x	mp	x	x	142	x	x	2*v	x	x	x	x	x	x	x	24	u1*gp	x	x
	44	0	0	1	0	0	2	74	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*2/3*h	71	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*2/3*h	101	(u1*n2*1/3)*2/3*h	x	x	x	x	x	x	142	x	x	2*v	x	x	x	x	x	x	x	25	u1*gp	x	x
	45	0	1	2	0	0	0	x	0	x	0	x	0	13	10	x	mp	x	x	114	120	126	(u1*n2+u1*u2)*v	100	41	n1*u2*v	114	146	146	n1*n2*v	x	x	x	x
	46	0	2	1	0	0	0	x	0	x	0	x	0	10	12	x	mp	x	x	115	121	127	(u1*n2+u1*u2)*v	43	98	n1*u2*v	115	146	146	n1*n2*v	x	x	x	x
	47	1	0	2	0	0	0	x	0	x	0	x	0	13	9	x	mp	x	x	116	122	128	(u1*n2+u1*u2)*v	122	38	n1*u2*v	146	122	146	n1*n2*v	x	x	x	x
	48	1	2	0	0	0	0	x	0	x	0	x	0	12	8	x	mp	x	x	117	123	129	(u1*n2+u1*u2)*v	37	129	n1*u2*v	146	146	129	n1*n2*v	x	x	x	x
	49	2	0	1	0	0	0	x	0	x	0	x	0	9	11	x	mp	x	x	118	124	130	(u1*n2+u1*u2)*v	124	95	n1*u2*v	146	124	146	n1*n2*v	x	x	x	x
	50	2	1	0	0	0	0	x	0	x	0	x	0	8	11	x	mp	x	x	119	125	131	(u1*n2+u1*u2)*v	94	131	n1*u2*v	146	146	131	n1*n2*v	x	x	x	x
	51	2	0	0	1	0	0	102	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	75	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	78	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	x	x	x	x	x	x	145	x	x	2*v	x	x	x	x	x	x	x	x	x	11	u1*gs
	52	2	0	0	0	1	0	103	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	76	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	79	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	18	x	x	mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	x	x	11	u1*gs
	53	2	0	0	0	0	1	104	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	77	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	80	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	19	x	x	n1*mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	x	x	11	u1*gs
	54	0	2	0	1	0	0	87	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	105	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	81	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	20	x	x	n1*mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	x	x	12	u1*gs
	55	0	2	0	0	1	0	88	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	106	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	82	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	x	x	x	x	x	x	145	x	x	2*v	x	x	x	x	x	x	x	x	x	12	u1*gs
	56	0	2	0	0	0	1	89	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	107	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	83	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	22	x	x	mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	x	x	12	u1*gs
	57	0	0	2	1	0	0	90	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	84	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	108	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	23	x	x	mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	x	x	13	u1*gs
	58	0	0	2	0	1	0	91	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	85	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	109	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	24	x	x	n1*mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	x	x	13	u1*gs
	59	0	0	2	0	0	1	92	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	86	(u1*n2*1/3 + u1*u2*2/3)*1/3*h	110	(u1*n2*1/3 + u1*u2*1/6)*1/3*h	x	x	x	x	x	x	145	x	x	2*v	x	x	x	x	x	x	x	x	x	13	u1*gs
	60	3	0	0	0	0	0	x	0	x	0	x	0	11	x	x	mp	x	x	132	135	138	(u1*n2+u1*u2)*v	135	138	n1*u2*v	147	135	138	n1*n2*v	x	x	x	x
	61	0	3	0	0	0	0	x	0	x	0	x	0	12	x	x	mp	x	x	133	136	139	(u1*n2+u1*u2)*v	15	139	n1*u2*v	133	147	139	n1*n2*v	x	x	x	x
	62	0	0	3	0	0	0	x	0	x	0	x	0	13	x	x	mp	x	x	134	137	140	(u1*n2+u1*u2)*v	137	16	n1*u2*v	134	137	147	n1*n2*v	x	x	x	x
Fours	63	1	1	1	1	0	0	x	0	x	0	x	0	30	27	x	n1*mp	27	u1*mp	144	x	x	2*v	x	x	x	x	x	x	x	x	x	26	u1*gs
	64	1	1	1	0	1	0	x	0	x	0	x	0	34	28	x	n1*mp	34	u1*mp	144	x	x	2*v	x	x	x	x	x	x	x	x	x	26	u1*gs
	65	1	1	1	0	0	1	x	0	x	0	x	0	35	32	x	n1*mp	32	u1*mp	144	x	x	2*v	x	x	x	x	x	x	x	x	x	26	u1*gs
	66	1	1	0	2	0	0	119	(u1*n2*1/3)*1/3*h	117	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	111	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	36	x	x	mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	27	u1*gp	x	x
	67	1	1	0	0	2	0	125	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	123	(u1*n2*1/3)*1/3*h	112	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	40	x	x	n1*mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	28	u1*gp	x	x
	68	1	1	0	0	0	2	131	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	129	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	113	(u1*n2*1/3)*1/3*h	41	38	x	n1*mp	41	u1*mp	143	x	x	2*v	x	x	x	x	x	x	x	29	u1*gp	x	x
	69	0	1	1	2	0	0	111	(u1*n2*1/3)*1/3*h	115	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	114	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	42	39	x	n1*mp	42	u1*mp	143	x	x	2*v	x	x	x	x	x	x	x	33	u1*gp	x	x
	70	0	1	1	0	2	0	112	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	121	(u1*n2*1/3)*1/3*h	120	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	40	x	x	mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	34	u1*gp	x	x
	71	0	1	1	0	0	2	113	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	127	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	126	(u1*n2*1/3)*1/3*h	44	x	x	n1*mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	35	u1*gp	x	x
	72	1	0	1	2	0	0	118	(u1*n2*1/3)*1/3*h	111	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	116	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	36	x	x	n1*mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	30	u1*gp	x	x
	73	1	0	1	0	2	0	124	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	112	(u1*n2*1/3)*1/3*h	122	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	43	37	x	n1*mp	37	u1*mp	143	x	x	2*v	x	x	x	x	x	x	x	31	u1*gp	x	x
	74	1	0	1	0	0	2	130	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	113	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	128	(u1*n2*1/3)*1/3*h	44	x	x	mp	x	x	143	x	x	2*v	x	x	x	x	x	x	x	32	u1*gp	x	x
	75	2	1	0	1	0	0	x	0	x	0	x	0	51	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	50	u1*gs
	76	2	1	0	0	1	0	x	0	x	0	x	0	28	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	50	u1*gs
	77	2	1	0	0	0	1	x	0	x	0	x	0	29	53	x	n1*mp	53	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	x	x	50	u1*gs
	78	2	0	1	1	0	0	x	0	x	0	x	0	51	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	49	u1*gs
	79	2	0	1	0	1	0	x	0	x	0	x	0	31	52	x	n1*mp	31	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	x	x	49	u1*gs
	80	2	0	1	0	0	1	x	0	x	0	x	0	32	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	49	u1*gs
	81	0	2	1	1	0	0	x	0	x	0	x	0	33	54	x	n1*mp	54	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	x	x	46	u1*gs
	82	0	2	1	0	1	0	x	0	x	0	x	0	55	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	46	u1*gs
	83	0	2	1	0	0	1	x	0	x	0	x	0	35	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	46	u1*gs
	84	0	1	2	1	0	0	x	0	x	0	x	0	57	33	x	n1*mp	33	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	x	x	45	u1*gs
	85	0	1	2	0	1	0	x	0	x	0	x	0	34	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	45	u1*gs
	86	0	1	2	0	0	1	x	0	x	0	x	0	59	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	45	u1*gs
	87	1	2	0	1	0	0	x	0	x	0	x	0	27	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	48	u1*gs
	88	1	2	0	0	1	0	x	0	x	0	x	0	55	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	48	u1*gs
	89	1	2	0	0	0	1	x	0	x	0	x	0	56	29	x	n1*mp	29	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	x	x	48	u1*gs
	90	1	0	2	1	0	0	x	0	x	0	x	0	30	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	47	u1*gs
	91	1	0	2	0	1	0	x	0	x	0	x	0	58	31	x	n1*mp	58	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	x	x	47	u1*gs
	92	1	0	2	0	0	1	x	0	x	0	x	0	59	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	x	x	47	u1*gs
	93	2	0	0	2	0	0	132	(u1*n2*1/3)*1/3*h	119	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	118	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	x	x	x	x	x	x	145	x	x	2*v	x	x	x	x	x	x	x	51	u1*gp	x	x
	94	2	0	0	0	2	0	135	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	125	(u1*n2*1/3)*1/3*h	124	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	37	x	x	n1*mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	52	u1*gp	x	x
	95	2	0	0	0	0	2	138	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	131	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	130	(u1*n2*1/3)*1/3*h	38	x	x	mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	53	u1*gp	x	x
	96	0	2	0	2	0	0	117	(u1*n2*1/3)*1/3*h	133	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	115	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	39	x	x	mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	54	u1*gp	x	x
	97	0	2	0	0	2	0	123	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	136	(u1*n2*1/3)*1/3*h	121	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	x	x	x	x	x	x	145	x	x	2*v	x	x	x	x	x	x	x	55	u1*gp	x	x
	98	0	2	0	0	0	2	129	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	139	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	127	(u1*n2*1/3)*1/3*h	41	x	x	n1*mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	56	u1*gp	x	x
	99	0	0	2	2	0	0	116	(u1*n2*1/3)*1/3*h	114	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	134	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	42	x	x	n1*mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	57	u1*gp	x	x
	100	0	0	2	0	2	0	122	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	120	(u1*n2*1/3)*1/3*h	137	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	43	x	x	mp	x	x	145	x	x	2*v	x	x	x	x	x	x	x	58	u1*gp	x	x
	101	0	0	2	0	0	2	128	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*2/3)*1/3*h	126	(u1*n2*1/3 + n1*u2*1/2 + u1*u2*1/3)*1/3*h	140	(u1*n2*1/3)*1/3*h	x	x	x	x	x	x	145	x	x	2*v	x	x	x	x	x	x	x	59	u1*gp	x	x
	102	3	0	0	1	0	0	x	0	x	0	x	0	x	x	x	x	x	x	147	x	x	2*v	x	x	x	x	x	x	x	x	x	60	u1*gs
	103	3	0	0	0	1	0	x	0	x	0	x	0	52	x	x	mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	x	x	60	u1*gs
	104	3	0	0	0	0	1	x	0	x	0	x	0	53	x	x	n1*mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	x	x	60	u1*gs
	105	0	3	0	1	0	0	x	0	x	0	x	0	54	x	x	n1*mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	x	x	61	u1*gs
	106	0	3	0	0	1	0	x	0	x	0	x	0	x	x	x	x	x	x	147	x	x	2*v	x	x	x	x	x	x	x	x	x	61	u1*gs
	107	0	3	0	0	0	1	x	0	x	0	x	0	56	x	x	mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	x	x	61	u1*gs
	108	0	0	3	1	0	0	x	0	x	0	x	0	57	x	x	mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	x	x	62	u1*gs
	109	0	0	3	0	1	0	x	0	x	0	x	0	58	x	x	n1*mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	x	x	62	u1*gs
	110	0	0	3	0	0	1	x	0	x	0	x	0	x	x	x	x	x	x	147	x	x	2*v	x	x	x	x	x	x	x	x	x	62	u1*gs
Fives	111	1	1	1	2	0	0	x	0	x	0	x	0	72	66	x	n1*mp	72	u1*mp	144	x	x	2*v	x	x	x	x	x	x	x	63	u1*gp	x	x
	112	1	1	1	0	2	0	x	0	x	0	x	0	70	67	x	n1*mp	67	u1*mp	144	x	x	2*v	x	x	x	x	x	x	x	64	u1*gp	x	x
	113	1	1	1	0	0	2	x	0	x	0	x	0	71	74	x	n1*mp	71	u1*mp	144	x	x	2*v	x	x	x	x	x	x	x	65	u1*gp	x	x
	114	0	1	2	2	0	0	x	0	x	0	x	0	99	69	x	n1*mp	99	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	84	u1*gp	x	x
	115	0	2	1	2	0	0	x	0	x	0	x	0	69	96	x	n1*mp	69	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	81	u1*gp	x	x
	116	1	0	2	2	0	0	x	0	x	0	x	0	72	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	90	u1*gp	x	x
	117	1	2	0	2	0	0	x	0	x	0	x	0	66	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	87	u1*gp	x	x
	118	2	0	1	2	0	0	x	0	x	0	x	0	93	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	78	u1*gp	x	x
	119	2	1	0	2	0	0	x	0	x	0	x	0	93	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	75	u1*gp	x	x
	120	0	1	2	0	2	0	x	0	x	0	x	0	70	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	85	u1*gp	x	x
	121	0	2	1	0	2	0	x	0	x	0	x	0	97	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	82	u1*gp	x	x
	122	1	0	2	0	2	0	x	0	x	0	x	0	100	73	x	n1*mp	73	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	91	u1*gp	x	x
	123	1	2	0	0	2	0	x	0	x	0	x	0	97	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	88	u1*gp	x	x
	124	2	0	1	0	2	0	x	0	x	0	x	0	73	94	x	n1*mp	94	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	79	u1*gp	x	x
	125	2	1	0	0	2	0	x	0	x	0	x	0	67	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	76	u1*gp	x	x
	126	0	1	2	0	0	2	x	0	x	0	x	0	101	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	86	u1*gp	x	x
	127	0	2	1	0	0	2	x	0	x	0	x	0	71	x	x	n1*mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	83	u1*gp	x	x
	128	1	0	2	0	0	2	x	0	x	0	x	0	101	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	92	u1*gp	x	x
	129	1	2	0	0	0	2	x	0	x	0	x	0	98	68	x	n1*mp	98	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	89	u1*gp	x	x
	130	2	0	1	0	0	2	x	0	x	0	x	0	74	x	x	mp	x	x	146	x	x	2*v	x	x	x	x	x	x	x	80	u1*gp	x	x
	131	2	1	0	0	0	2	x	0	x	0	x	0	68	95	x	n1*mp	68	u1*mp	146	x	x	2*v	x	x	x	x	x	x	x	77	u1*gp	x	x
	132	3	0	0	2	0	0	x	0	x	0	x	0	x	x	x	x	x	x	147	x	x	2*v	x	x	x	x	x	x	x	102	u1*gp	x	x
	133	0	3	0	2	0	0	x	0	x	0	x	0	96	x	x	mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	105	u1*gp	x	x
	134	0	0	3	2	0	0	x	0	x	0	x	0	99	x	x	n1*mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	108	u1*gp	x	x
	135	3	0	0	0	2	0	x	0	x	0	x	0	94	x	x	n1*mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	103	u1*gp	x	x
	136	0	3	0	0	2	0	x	0	x	0	x	0	x	x	x	x	x	x	147	x	x	2*v	x	x	x	x	x	x	x	106	u1*gp	x	x
	137	0	0	3	0	2	0	x	0	x	0	x	0	100	x	x	mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	109	u1*gp	x	x
	138	3	0	0	0	0	2	x	0	x	0	x	0	95	x	x	mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	104	u1*gp	x	x
	139	0	3	0	0	0	2	x	0	x	0	x	0	98	x	x	n1*mp	x	x	147	x	x	2*v	x	x	x	x	x	x	x	107	u1*gp	x	x
	140	0	0	3	0	0	2	x	0	x	0	x	0	x	x	x	x	x	x	147	x	x	2*v	x	x	x	x	x	x	x	110	u1*gp	x	x
System Failures	141	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	x	x	u3*w	x	x	x	x	x	x	x	x	x	x	x
	142	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	2	x	x	u3*w	x	x	x	x	x	x	x	x	x	x	x
	143	1	1	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	8	x	x	u3*w	x	x	x	x	x	x	x	x	x	x	x
	144	1	1	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	26	x	x	u3*w	x	x	x	x	x	x	x	x	x	x	x
	145	2	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	11	x	x	u3*w	x	x	x	x	x	x	x	x	x	x	x
	146	2	1	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	50	x	x	u3*w	x	x	x	x	x	x	x	x	x	x	x
	147	3	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	60	x	x	u3*w	x	x	x	x	x	x	x	x	x	x	x