2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

Perturbation Analysis for Optimal Update Intervals of Data Sets

Matthew C. Ruschmann and N. Eva Wu

Abstract— **Update intervals for data renewal in a 3-server distributed database are optimized to provide maximum system availability. A Semi-Markov model of the database unit is defined. Given probability distributions, analytical expressions are derived for the expected availability of the database. Under the case of unknown failure rates and renewal intervals, estimates for the gradients of the sample performance functions are derived using infinitesimal perturbation analysis. Optimal values of the update interval are found using a stochastic approximation algorithm.**

I. INTRODUCTION

THE 3-server database in Fig. 1 presents a distributed database designed to counteract the consequences of data failures such as data loss, corruption, and aging. In order to prevent failure and restore failed servers, this paper introduces a fixed-interval data renewal process that restores data integrity and removes server failures. Availability is defined as either the percentage of time all servers are available or the average number of available servers. Maximum availability of the database is obtained through optimization of the update interval between data renewals. In order to optimize this interval, gradients of the availability functions are derived with perturbations analysis. Using these gradients, a stochastic approximation algorithm optimizes the update interval. Comparisons are drawn to state-based overhaul and restoration techniques previously investigated by Wu, Metzler, and Linderman [1] and optimized by Ruschmann, Wu [2].

The introduction of a non-Poisson fixed update interval relaxes the reduced-state database model to a Semi-Markov process. This is contrary to the Markov chain models derived and analyzed in previous work [1], [2]. Sample performance functions representing availability of the system are defined. Although the lack of standard results for Semi-Markov processes limits typical approaches to simulations, analytical expressions for the expected sample performance functions are derived under the assumption of known exponentially distributed failure and data renewal rates.

In order to optimize availability under unknown distributions, perturbation analysis is applied to sample paths of the database model. More specifically, infinitesimal perturbation analysis (IPA) techniques derive estimates of the sample performance functions' gradients. Analysis of infinitesimally small perturbations in the fixed update interval provides unbiased estimators as originally introduced in Zazanis's doctoral thesis [3] and formally presented in [4], [5]. Cassandras provides a chapter on perturbation analysis in [6]. Unbiasedness of IPA estimators relies on several conditions and assumptions on the underlying Semi-Markov process. The conditions and assumptions are shown to be satisfied except the non-interruption condition requiring that all events executed are not cancelled prior to expiration of their event lives. Elimination of event cancellation in the model is conducted following the methods provided by Savage in [7].

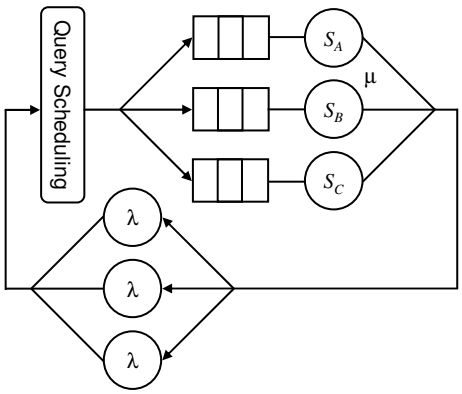

Fig. 1. A queuing network representation of a partitioned database unit with three servers

The IPA estimates of the availability gradients provide insight into the dynamics of the system and introduce the ability to optimize the system with respect to the update interval. Traditional gradient search methods, such as Newton's method, require exact knowledge of the functions gradient. Inaccurate gradient estimates require stochastic approximation algorithms such as the one presented by Robbins and Monro [8]. A multi-run optimization determines the optimal value before implementation of the database. Lastly, a single run optimization is also considered in which the parameter's optimal value is located while the simulated database runs.

The paper is organized as follows. Section II describes the reduced-state, Semi-Markov model of the database system, based on the presentation of [1]. This section also defines sample functions to measure availability of the system and their expected values. Section III derives infinitesimal perturbation analysis (IPA) estimators for the gradients of the sample performance functions. Section IV describes application of the gradient estimates to a stochastic approximation algorithm and discusses optimal results. Section V offers a comparison to previous work. The paper is concluded in section VI and future goals are noted.

This work was supported by AFOSR Grant FA9550-06-0456 and AFRL Contract FA8750-07-1-0172.

Matthew C. Ruschmann is a PhD candidate with the Department of Electrical and Computer Engineering, Binghamton University, Binghamton, NY 13902, matthew@ruschmann.net

N. Eva Wu is with the Department of Electrical and Computer Engineering, Binghamton University, Binghamton, NY 13902, evawu@binghamton.edu

II. REDUCED-STATE MODEL OF THE DISTRIBUTED DATABASE

Notation of the reduced-state database model without customer considerations is specified along with event-life distributions. A Semi-Markov process is defined with failure event cancellation. In order to satisfy the requirements of IPA, event cancellations are then eliminated within the Semi-Markov process. Sample performance functions are defined to measure performance of the database model.

A. Model Notation and Specification

The database system in Fig. 1 contains three servers in parallel to answer three classes A, B, C of queries for which relevant information can be found in the partitioned data sets A, B, C of the database, respectively. Each of the three servers, S_A, S_B, and S_C, contains data for serving the respective customer set. The queues preceding servers are named Q_A, Q_B, and Q_C, respectively. However for the reduced state model, database queries and queue sizes are ignored. Only failure events, the fixed data update interval, and data renewal procedures are considered.

The model is built in this study with the premise that event life distributions have been established for the process of data failure within a server ($\exp(\nu)$) and the process of system renewal ($\exp(\gamma)$) during which the active and inactive servers' data is renewed. All such processes are independent. The model also contains a fixed update interval, T, determining the interval between data renewal in the servers. Markov chains require the process to be independent of past state information and independent of the elapsed time in the current state [6]. A non-exponentially distributed fixed update interval introduces a dependency on the elapsed time in the current state. Therefore, the process is relaxed to a Semi-Markov process where only state independence holds.

B. Definition of a Semi-Markov Process

State space $\mathcal{X}$. A state name is coded into five states with a two-digit number indicative of the total number of unavailable servers and ongoing data renewal, Nr. The first and second digits represents the number of unavailable servers, $N \in \{0, 1, 2, 3\}$, and a binary indication of ongoing data renewal, r, respectively. Customers are not considered in the reduced-state model of the database system. In addition, symmetry of the model in Fig. 1 indicates state-based knowledge of the specific server with data failure is irrelevant. For example, state $x = 20$ indicates failure in two of the database servers with renewal inactive. $x = 31$ represents the only state with ongoing data renewal as all servers are assumed to be deactivated during this period. The entire state space is contained in $\mathcal{X} = \{00,10,20,30,31\}$ as seen in Fig. 2.

Initial state. It is assumed that the database system starts operation from state $x = 00$, i.e. the database begins operation with all data recently renewed. A regenerative process [9] results from the model in 1. The process enters state $x = 31$ upon expiration of the update interval. Proceeding completion of data renewal, the process returns to the initial state $x = 00$ resulting in regeneration of the process.

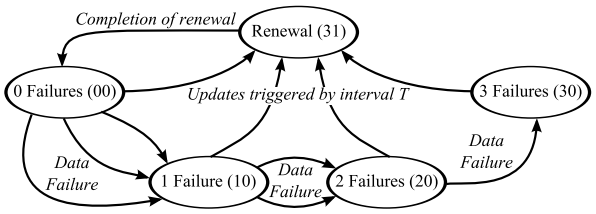

Fig. 2. Event transition diagram summarizing state propagation in a reduced state model of the distributed database system.

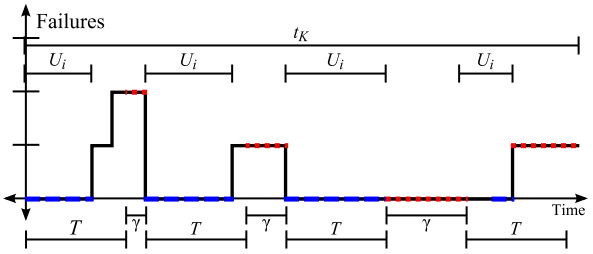

Fig. 3. A single sample path of the reduced-state model. Dotted red lines indicate periods in the renewal state, $x = 31$. Dashed blue lines represent periods that the database system is fully available.

Set of state transition events. Events that trigger the transitions and the corresponding transition distributions. Data failure occurs within a server at an exponential rate ν. This event triggers a state transition to $N = N + 1$. Renewal of data within the database is triggered by an update event after a fixed interval T. Expiration of the update event migrates the process to the renewal state, $x = 31$, and subsequently cancels all data failure events. Renewal time of data within the servers is determined by an exponential distribution of rate γ. Upon expiration, the process returns to the initial state, $x = 00$, and executing another update interval, T.

At this point a Semi-Markov model for the database system of Fig. 1 has been established. An event-transition diagram summarizes the model in Fig. 2. The aggregation of a single sample path is carried out in Fig. 3.

C. Elimination of Event Cancellations

It was previously noted that unbiased IPA estimators require a non-interruption condition to be satisfied [6]. After a failure event is executed, it must terminate by expiration of its event life. This is violated upon expiration of an update event and subsequent cancellation of the remaining failure events before their termination. However, computation theory dictates that event cancellation is never necessary [7]. Using methods of event cancellation elimination similar to those proposed in [7], a model without event cancellation is derived. Sets of three failure events are numbered and tracked until all failure events have expired. If the failure events expire after the corresponding update period, they are simply ignored.

The modified state space and transition events follows. The new state-space is now defined by an string of digits in the form $NrIf_1f_2...f_n$ where N and r were previously defined. I is the index of the currently active (not ignored)

set of failure events. Each f_i stores the remaining number of events in set i. Upon expiration of a failure event in set i, ν_i, f_i is decremented along with N if $i = I$ is the active set. Expiration of an update event now deactivates the current set of failure events by setting $I = 0$ eliminating event cancellation. When a renewal is completed, event γ, new failure events ν_i are activated where i is the smallest value such that $f_i = 0$.

D. Sample Performance Functions

In order to quantify the availability of the database, sample performance functions must be defined. As customers are not considered in the model, sample performance functions must rely solely on availability of the servers. Two such measures are expected uptime of the full system and the expected number of available servers. Simplicity of the model and event-life distributions allows simple analytical expressions to be derived for each measure.

1) Full System Uptime: Full system uptime is defined as the expected percentage of time all servers are available. These periods are indicated by dashed blue lines in Fig. 3 where the state $x = Nr = 0$. The system is considered unavailable as a result of data failure within a server or deactivation of the servers during the renewal process. Conveniently the system is a regenerative process, and there is only one period of uptime during each regeneration of the system. Therefore, the average uptime given a sample path, ω, with update interval T is defined as

$$L_{up}(T,\omega) = U_{ave} = \frac{\sum_{i=1}^{K} U_i}{t_K} = \frac{1}{K}\sum_{i=1}^{K}\frac{U_i}{T_i} \qquad (1)$$

where K is the number of regenerations, t_K is the total length in time of sample path ω, U_i is the length of uptime in each regeneration i as labeled in Fig. 3, and T_i is the length of time of each regeneration period i. The law of large numbers [6] dictates that this estimate of the full system uptime approaches the actual value as K approaches infinite.

An analytic expression results from the expected value of (1) as

$$E[U_{ave}] = \frac{1}{K}\sum_{i=1}^{K}\frac{E[U_i]}{E[T_i]}. \qquad (2)$$

The value $E[T_i]$ is formulated as the expected time of each regeneration period. Fig. 3 shows the expected time of each regeneration consists of the update interval T and the expect length of a renewal event. The renewal event-life distribution is defined as exponential with expected delay $\frac{1}{\gamma}$. $E[U_i]$ is the average amount of time to either a server failure or system update. The cumulative probability distribution of the uptime interval is represented by an exponential distribution with rate 3ν (the rate of failure in all servers superpositioned) truncated at T. It has a mean of $\frac{1}{3\nu}(1 - e^{-3\nu T})$. Hence,

$$E[U_{ave}] = \frac{\frac{1}{3\nu}(1 - e^{-3\nu T})}{T + \frac{1}{\gamma}}. \qquad (3)$$

For comparison, Fig. 4 plots the analytical value in (3) along with the value estimated by averaging the uptime

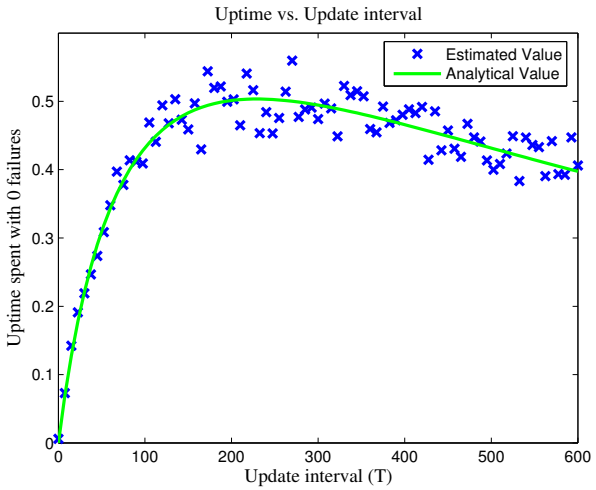

Fig. 4. The calculated analytic value of full system uptime and corresponding estimate versus update interval over $K = 100$ update periods.

intervals over $K = 100$ database update intervals. The estimates are consistent with the derived analytic value.

2) Average number of available servers: The average number of available servers is determined by sum of the fraction of time the system is in a state with a given number of available servers. The average number of available servers given sample path ω is then

$$L_{avail}(T,\omega) = S_{ave} = \frac{\sum_{i=1}^{M} A_i S_i}{t_K} = \frac{1}{M}\sum_{i=1}^{M}\frac{A_i S_i}{T_i} \qquad (4)$$

where M is represents the final, M^{th}, state of the system with each state lasting a duration of A_i with S_i servers available. T_i is again length of each individual period, and t_K is defined as before.

Analysis of the expected value results from considerations similar to the previous section. The expected availability of each individual server is given an availability distribution as a cumulative exponential distribution with rate ν truncated at update interval T. Multiplying the mean availability of an individual server by three servers results in

$$E[S_{ave}] = 3\frac{\frac{1}{\nu}(1 - e^{-\nu T})}{T + \frac{1}{\gamma}}. \qquad (5)$$

III. DERIVATION OF IPA ESTIMATORS

Infinitesimal perturbation analysis (IPA) derives estimators for the gradient of a sample function by considering infinitesimally small perturbation in the update interval, T, on a single sample path of the process, ω. The strict requirements on the target process for this derivation technique to be unbiased given in [6] are shown to be satisfied by the previously defined Semi-Markov process. Gradients of the sample performance functions are then taken and their estimates are derived by IPA techniques. Performance of the resulting estimators is investigated.

A. Conditions and Assumptions for Unbiased IPA

The unbiasedness and consistency of an IPA estimator is proven if the underlying process satisfies a set of basic conditions and assumptions [6]. These conditions is to guarantee smoothness of the sample functions and their derivatives over small perturbations in the update interval. Most simply, the non-interruption condition must be satisfied along with smoothness of the sample function and event-life distributions with respect to the parameter in question. In addition, a more cumbersome commuting condition [6] is placed on the underlying process.

1) Non-interruption condition: The non-interruption condition requires that events are not cancelled or interrupted after their execution [6]. The process originally defined required event cancellation, but this requirement was relaxed in the preceding section.

2) Smoothness of the sample functions: The sample performance functions are required to be smooth subject to the parameter of interest, $\theta = T$. Clearly, discontinuities in the performance function itself result in non-existent gradients and biased estimates. An example of a non-smooth sample function is the number of failed servers within a given period. Discontinuities exist because the function jumps between integer values. This condition is satisfied by the sample performance functions provided in section I. as they represent periods of time that change continuously with changes in θ.

3) Smoothness of event-life distributions: In order to prove the IPA estimates of the gradients are unbiased, two assumptions are made on the continuity of event-life distributions. All distributions must be not only continuous in the parameter of interest, $\theta = T$, but also almost surely continuously differentiable in θ [6]. Because the distribution of our update interval is discrete with $P(T) = 1$, these assumptions are not satisfied. However, a specific exception exists for the case of discrete distributions where only values, and not probabilities, vary with parameter θ [6].

4) The commuting condition: The commuting condition is often the most difficult to satisfy. In words, it requires that any sequence of events (α, β) bring state x to y must also bring state x to y when their order is reversed to (α, β) [6]. In this case, the commuting conditions holds as verified by manually observing the state-transitions switches in Fig. 2.

B. Derivatives of the sample performance functions

The IPA estimators are derived directly from the derivatives of the sample performance functions for full system uptime and the average number of available servers,

$$\frac{dL_{up}(T,\omega)}{dT} = \frac{t_K \sum_{i=1}^{K} \frac{dU_i}{dT} - \frac{dt_K}{dT} \sum_{i=1}^{K} U_i}{t_K^2}, \quad (6)$$

$$\frac{dL_{avail}(T,\omega)}{dT} = \frac{t_K \sum_{i=1}^{M} \frac{dY_i}{dT} A_i - \frac{dt_K}{dT} \sum_{i=1}^{M} S_i A_i}{t_K^2}, \quad (7)$$

respectively. t_K, U_i, A_i, and S_i are directly calculated from sample path ω as previously defined. $\frac{dt_K}{d\theta}$, $\frac{dU_i}{d\theta}$, and $\frac{dY_i}{d\theta}$ are calculated with IPA estimations.

C. Derivation of IPA estimators

To simplify (6), the values of $\frac{dt_K}{dT}$ and $\frac{dU_i}{dT}$ are formulated using IPA. The fixed update interval, T, can be described as a location parameter, $\frac{dT}{dT} = 1$. A change in the update interval results in a corresponding change in the discrete update interval distribution. As the set of K update intervals over sample path ω is perturbed, the termination time of the sample path is perturbed by $\frac{dt_K}{dT} = K$. By substituting these values into (6), the IPA estimator is found to be

$$\left[\frac{dL_{up}(T,\omega)}{dT}\right]_{\text{IPA}} = \frac{t_K \sum_{i=1}^{K} \frac{dU_i}{dT} - \sum_{i=1}^{K} U_i}{t_K^2}. \quad (8)$$

Each interval of uptime, U_i, is defined by $U_i = t_{i,f} - ti, 0$ where $t_{i,f}$ and $t_{i,0}$ are the final time of the period and initial time of the period, respectively. The perturbation in the initial time, $\frac{dti,0}{dT}$, is simply the number of preceding update intervals, $\frac{dti,0}{dT} = i$. If $t_{i,f}$ is determined by server failure, then $\frac{dti,f}{dT} = 0$. Otherwise $t_{i,f}$ is determined by the termination of an extra update interval, and $\frac{dti,f}{dT} = i$. Therefore,

$$\frac{dU_i}{dT} = \begin{cases} 0 \text{ if terminated by a failure event,} \\ 1 \text{ if terminated by the update event.} \end{cases} \quad (9)$$

By substituting (9) into (8) and defining I as the number of uptime intervals terminated by an update event, the IPA estimator becomes

$$\left[\frac{dL_{up}(T,\omega)}{dT}\right]_{\text{IPA}} = \frac{I - KL_{up}(T,\omega)}{t_K}. \quad (10)$$

The IPA estimate for the gradient of $L_{avail}(T,\omega)$ is similarly derived. The period Y_i is determined to remain constant under perturbations in T unless terminated by an update event. Therefore,

$$\frac{dY_i}{d\theta} = \begin{cases} 1 \text{ when terminated by an update event,} \\ 0 \text{ otherwise.} \end{cases} \quad (11)$$

After substituting $\frac{dt_K}{dT} = K$ and $\frac{dY_i}{d\theta}$,

$$\left[\frac{dL_{avail}(T,\omega)}{dT}\right]_{\text{IPA}} = \frac{\sum_{i=1}^{K} A_i - KL_{avail}(T,\omega)}{t_K} \quad (12)$$

where the sum of A_i only includes intervals terminated by the update interval.

D. Performance of IPA estimators

In order to verify the performance of the estimates (10) and (12), simulations of the database are conducted. As a stopping rule, the simulations are carried out over $K = 100$ update intervals. These results are plotted against the analytical gradients in Fig. 5 and Fig. 6 for the full system uptime and average number of available servers, respectively.

The estimates appear to be unbiased as previously determined by satisfying the assumptions and conditions of IPA. In fact, the standard deviation of the IPA estimate for full system uptime appears smaller than the deviation of the sample performance function in Fig. 4.

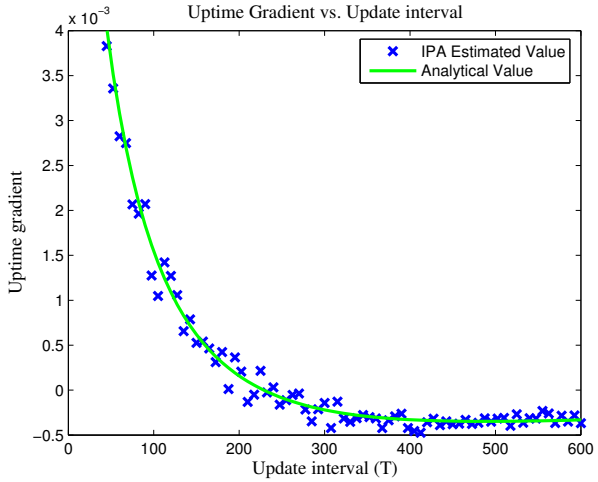

Fig. 5. Comparison of the IPA estimate and analytical value for the gradient of the full system uptime.

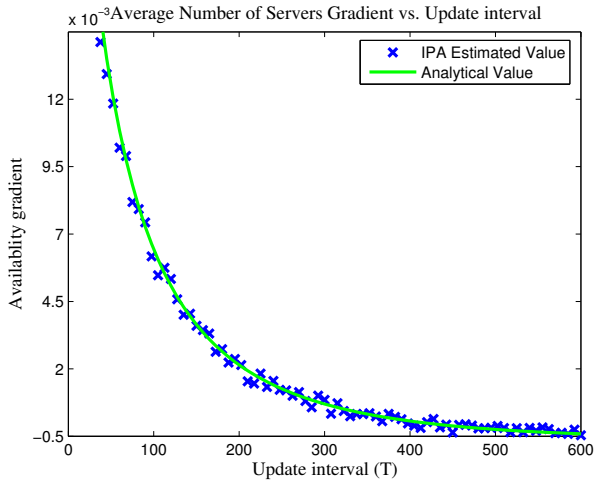

Fig. 6. Comparison of the IPA estimate and analytical value for the gradient of the average number of available servers.

IV. STOCHASTIC APPROXIMATION OF THE OPTIMAL UPDATE INTERVAL

Gradient estimates are derived to analyze and optimize the performance of the database under a data-update policy. The uncertain gradient estimations require a stochastic approximation method as opposed to a standard optimization algorithm. The optimal update interval is found over a series of different failure rates. Additionally, a single-run optimization scheme is formulated to optimize the database live.

Optimization requires a stochastic approximation algorithm with consideration of inaccurate gradient estimates. The first and simplest of such iterative algorithms was introduced by Robbins and Monroe [8]. The iterative Robbins-Monroe algorithm takes the following form:

$$\theta_{n+1} = \theta_n - a_n \tilde{\nabla} L(\theta_n, \omega) \tag{13}$$

where a_n, $n = 1, 2, ...,$ is an appropriately chosen set of gains.

Robbins and Monroe originally proved the algorithm to converge using an unbiased estimator with independent noise between samples [8]. Convergence is not proven for this particular process and sample performance functions because noise is not independent. However, convergence has been proven for the $GI/G/1$ queue and other IPA estimators [10], [11]. Convergence under independent samples requires [8], [12]

$$\lim_{n\to\infty} (a_n) = 0, \tag{14}$$

$$\sum_{n=1}^{\infty} |a_n| = \infty, \text{ and} \tag{15}$$

$$\sum_{n=1}^{\infty} a_n^2 < \infty. \tag{16}$$

In order adherence to these convergence requirements,

$$a_n = \frac{C}{n} \tag{17}$$

where the rate of convergence is dependent on selection of C. $C = 10000$ is chosen by educated guesses and checks of the convergence rate.

A. Multi-run Optimization

A multi-run optimization locates the optimal value of θ using multiple simulations prior to implementing the database. The optimal update interval in Fig. 7 compares the optimal solutions found with (13) to the known optimum calculated analytically. In this case, $K = 100$ for each individual gradient estimation.

B. Single-run Optimization

A single-run optimization algorithm is also desired to update the database as it runs live. Instead of simulating the database before it is implemented, optimization is done as the database runs live. The stochastic approximation is carried out as before, but the update interval is changed to θ_{n+1} after ever $K = 100$ update intervals. The results over 88 update intervals are shown in Fig. 8. The live value of update interval, T, converges to the known analytic optimum.

V. COMPARISON TO PREVIOUS WORK

Previously Wu, Metzler, and Linderman investigated the effectiveness of state-based overhaul and restoration techniques in a similar database unit [1] through simulation and Markov chain analysis. In their investigation, a state-based overhaul policy renewed all servers whenever a single failure occurred. A second restoration policy was also defined where secondary data stores in adjacent servers reduce unavailability by restoring failed servers. Later, Ruschmann and Wu formulated a Markov Decision Process to solve optimal state based restoration policies based on the service time of this model [2].

Table I summarizes the performance of the overhaul, restoration, and optimal service time policies implemented in [13], [2] through state-based control policy. In this case, failure rate is $\nu = 0.001$, overhaul rate is 0.01, and restoration

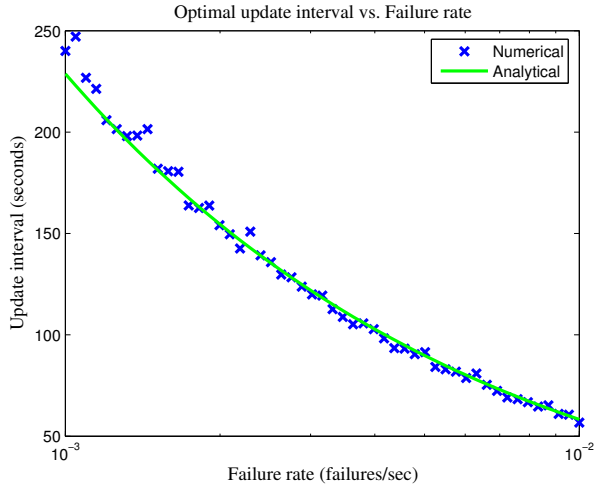

Fig. 7. The optimal update interval versus increasing failure rate. The stochastic approximation algorithm determines the numerical result

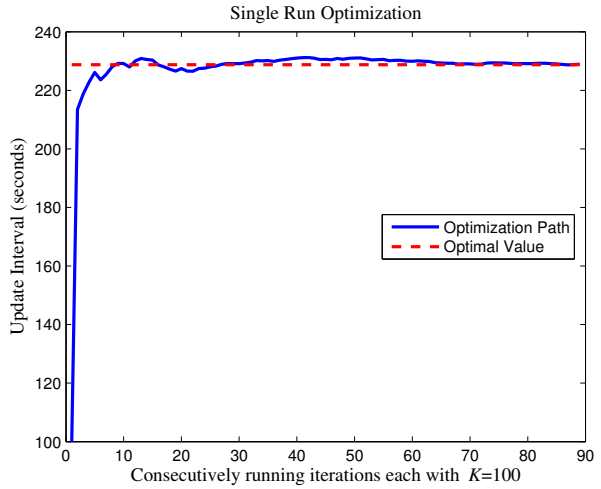

Fig. 8. Convergence of a single run optimization to the analytic optimal.

rate is 0.05. Table II summarizes optimal performance of the renewal policy implemented in this database system under similar system parameters. The loss of state information clearly reduces the systems ability to perform.

VI. CONCLUSIONS AND FURTHER WORK

A Semi-Markov model of the database under a fixed update interval renewal policy is formulated. Sample performance functions are defined and solved analytically given known exponential failure and renewal rates. Although analytical solutions of the sample performance functions exist, gradient estimates are derived using IPA techniques. These estimates perform regardless of the event-life distributions. The conditions and assumptions for application of IPA are validated. Failure event cancellation is eliminated by introducing a larger process and state-space. The gradients are applied to a stochastic approximation algorithm in order to locate the optimal fixed update interval of the process.

As further work, a proof of convergence regarding stochastic approximation presents itself. Furthermore, reintroducing customers to the model is of significant interest to the authors. Maximizing response times for customer process can be the most significant performance consideration of a database system. Unfortunately, the resulting system violates the commuting condition. Therefore, smoothed perturbation analysis (SPA) will be required to derive gradient estimator.

TABLE I

AVAILABILITY UNDER THE PREVIOUSLY INVESTIGATED POLICIES.

Control Policy	Uptime	Average Available Servers
Restoration	88%	2.74
Overhaul	77%	2.31
Optimal	87%	2.74

VII. ACKNOWLEDGEMENTS

The authors would like to thank Christos Cassandras for the inspiration of perturbation analysis in his book [6] and encouragement to attempt the application of infinitesimal perturbation analysis.

REFERENCES

[1] N. Wu, J. Metzler, and M. Linderman, "Supervisory Control of a Database Unit," *Proc. IEEE Conference on Decision and Control*, pp. 7615–7620, 2005.
[2] M. C. Ruschmann and N. E. Wu, "Optimal Control of a Database with Reduced & Full State Models," *Proc. IEEE Conference on Decision and Control*, 2007.
[3] M. A. Zazanis, "Statistical Properties of Perturbation Analysis Esrimates for Discrete Event Systems," Ph.D. dissertation, Harvard University, 1986.
[4] R. Suri, "Infinitesimal perturbation analysis for general discrete event systems," *Journal of the ACM (JACM)*, vol. 34, no. 3, pp. 686–717, 1987.
[5] R. Suri and M. Zazanis, "Perturbation Analysis Gives Strongly Consistent Sensitivity Estimates for the M/G/1 Queue," *Management Science*, vol. 34, no. 1, pp. 39–64, 1988.
[6] C. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Kluwer Academic, 1999.
[7] E. Savage and L. Schruben, "Eliminating event cancellation in discrete event simulation," *Proceedings of the 27th conference on Winter simulation*, pp. 744–750, 1995.
[8] H. Robbins and S. Monro, "A Stochastic Approximation Method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.
[9] E. Kao, *An Introduction to Stochastic Processes*. Duxbury Press, 1997.
[10] M. Fu, "Convergence of a stochastic approximation algorithm for the GI/G/1 queue using infinitesimal perturbation analysis," *Journal of Optimization Theory and Applications*, vol. 65, no. 1, pp. 149–160, 1990.
[11] E. Chong and P. Ramadge, "Convergence of recursive optimization algorithms using infinitesimal perturbation analysis estimates," *Discrete Event Dynamic Systems*, vol. 1, no. 4, pp. 339–372, 1992.
[12] A. Gelb, *Applied Optimal Estimation*. Mit Press, 1974.
[13] J. Metzler and N. Wu, "A Simulation Study of the Effect of Supervisory Control on a Redundant Database Unit," *Technical Report to AFRL RRS*, 2005.

TABLE II

AVAILABILITY OF THE RENEWAL POLICIES WITH UPDATE INTERVAL T.

Refresh Rate	Maximization Target	T	Uptime	S_{ave}
$\gamma = 0.05$	Uptime	115	72%	2.41
$\gamma = 0.05$	Average Servers	195	69%	2.47
$\gamma = 0.01$	Uptime	230	50%	1.87
$\gamma = 0.01$	Average Servers	415	46%	1.98