Disseminating Dynamic Data with QoS Guarantee in a Wide Area Network: A Practical Control Theoretic Approach

Ratul kr. Majumdar, *Krithi Ramamritham*, *Ravi N. Banavar*, and *Kannan Moudgalya*

Indian Institute of Technology Bombay
{ratul, krithi}@cse.iitb.ac.in, banavar@ee.iitb.ac.in, kannan@che.iitb.ac.in.

Abstract

Often, data used in on-line decision making (for example, in determining how to react to changes in process behavior, traffic flow control, etc.) is dynamic in nature and hence the timeliness of the data delivered to the decision making process becomes very important. The delivered data must conform to certain time or value based application specific consistency requirements. The design of mechanisms for such data delivery is challenging given that dynamic data changes rapidly *and* unpredictably*, the latter making it very hard to use simple prediction techniques. To address these challenges we develop mechanisms to obtain timely and consistency-preserving updates for dynamic data by* pull*ing data from the source at strategically chosen points in time, providing Quality of Service (QoS) Guarantees. Motivated by the need for practical system design, but using formal analytical techniques, we offer a systematic approach based on control-theoretic principles. We present a stochastic controller based on the Linear Quadratic Gaussian (LQG) technique as a means for deciding when to next refresh data from a source. A simple enhancement of the LQG algorithm allows us to provide QoS guarantees. Using real-world traces of real-time data we show the superior performance of our feedback-driven control-theoretic approach by comparing with a previously proposed adaptive refresh technique, a pattern matching technique, and a proportional controller with dynamically changing tuning criteria.*

1 Introduction

Data in many Web sources is time-varying (i.e., changes frequently): Examples include sports information, news, financial information such as stock prices, and traffic data. An important issue in accessing such time varying data is the maintenance of temporal coherency. The coherency requirements on a data item depends on the nature of the item and user tolerances. To illustrate, a user may be willing to receive sports and news information that may be out-of-sync by a few minutes with respect to the source, but may desire stronger coherency requirements on the status of patients' critical parameters.

In a typical data dissemination environment, maintaining consistency of the source and the user can be achieved in one of two ways: 1) In the `Source push` model, the source of the data *push*es data to the user (whenever data changes at the source). In this model, the source must maintain state information, in particular, it must keep track of user requirements and push the data to a user at the appropriate time. 2) In the `User Pull` model, the user *pulls* data from the source (whenever it suspects that data might have changed at the source). So, consistency of the data at the user depends on how often the user polls, i.e., pulls from the source. Even if a source has push capability, pushing only when required leads to computational overheads and state-space overheads at the sources[1]. So, we look at the possible ways in which judicious pulling can be accomplished. We define *Time to Refresh (TTR)* as the time gap after which a user refreshes the data item from the source. For minimizing the incurred communication overheads, the value of TTR must be high. But a high TTR value may compromise temporal consistency. We must also dynamically update this TTR value using an algorithm that decides the value depending on the present and past rates of source changes, with the goal of keeping remote requests to a minimum while maintaining the needed temporal accuracy of the data.

1.1 Maintaining Temporal Coherency

We assume that a user specifies a temporal coherency requirement, c, for each cached item of interest. The value of c denotes the maximum permissible deviation of the value known to the user from the value at the source and thus constitutes the user-specified tolerance. Observe that c can be specified in units of *time* (e.g., the item should never be out-of-sync by more than 5 minutes) or *value* (e.g., the stock price should never be out-of-sync by more than a dollar).

In this paper, we only consider temporal coherency requirements specified in terms of the value of the data as maintaining temporal coherency specified in units of time is a simpler problem. Thus, we assume that a user specifies a temporal coherency requirement c, for each item of interest. The value of c specifies the maximum permissible deviation of the date item value seen by the user from the actual value at the source:

$$|data\ item\ value_{source} - data\ item\ value_{user}| \leq c \quad (1)$$

The degree to which users' coherency needs are met is measured in terms of the *fidelity* of the data seen by users. We define the fidelity F observed by a user to be the total length of time that the above inequality holds (normalized by the total length of the observations). In addition to specifying the coherency requirement, users can also specify their fidelity requirement F for each data item so that an algorithm that is capable of handling users' fidelity and temporal coherency requirements (c's) can adapt to users' needs. The problem is hence to devise an algorithm to generate the sequence of TTRs for which the temporal coherency requirement is satisfied, delivering the desired fidelity, at the lowest cost. Note that choosing large TTR values translates to low cost.

1.2 Overview of Prior Approaches

A simple approach to meet the user specified coherency requirements is to use a constant refresh rate, i.e., periodic polls. However, this technique requires an *a priori* estimation of the data dynamics.

To achieve coherency requirement of a dynamic data item a heuristic approach, *Adaptive TTR* is proposed in [5]. A heuristic approach, Adaptive pattern matching TTR (*APMT* algorithm) as well as a control theoretic approach are proposed in [3]. The *Adaptive TTR* algorithm varies the TTR value based on the rate of change of the data item whereas *APMT* algorithm predicts TTR based on past known values of TTR and direction of change of data. A control theoretic proportional controller approach, tuned with Ziegler-Nichol's method, is shown to work well for low and medium speed traces.

1.3 Contribution of this paper: A New LQG based Approach

The adaptive proportional control theoretic approach [3] is not suitable for providing QoS guarantees as it controls only one parameter, the output error, and uses this to determine the control effort. This motivates us to use stochastic control theory based LQG approach to solve the problem of providing QoS guarantees in disseminating dynamic data. We use the LQG approach in an adaptive manner to provide a performance trade-off between the control effort (TTR) and output error to provide QoS guarantees. Performance studies are done with real world dynamic data traces to show that the algorithm indeed provides the required QoS guarantees. We describe general control theory approaches to address the problem in Section 2. Stochastic control theory based LQG algorithm is proposed in Section 3 and QoS Guarantee providing algorithm (QSG) is given in Section 4. We describe details of performance evaluation, and discuss the results in Section 5. Section 6 provides related work. Finally conclusion and future work is presented in Section 7.

2 Control Theoretic Approach

2.1 Challenges in Control Theoretic Formulation

The control objective is to decide the time of pull so that a change of c is not missed (cf. Equation (1)). This immediately suggests that we use the time of pulling as the input variable and the value of the data item at that time as the output variable. Note that this input is quite different from what we see in the usual control problems where *time* is an independent variable.

The process that we deal with is also unusual in that we are required to find the absolute difference between the data item values at two successive samples, say q_i and q_{i-1}. Let $\Delta q_i = q_i - q_{i-1}$ be the difference between the data item values at the two successive samples at t_i and t_{i-1} respectively. In order to decide a strategy for providing a high fidelity together with a high average TTR, we are interested in knowing when the absolute value of the difference Δq_i will cross c. That is, we require that at all times,

$$|\Delta q_i| - c \leq 0 \quad (2)$$

If we take $(|\Delta q_i| - c)/c$ as the output y_i of the system, then our fidelity requirement can be stated as demanding y_i to be equal to 0. This hence takes the form of a regulation problem. Note that the absolute function is a process model and is also unusual.

2.2 Problem Formulation

As discussed above, we define the input u_i to be,

$$u_i = t_i - t_{i-1} \quad (3)$$

where t_i and t_{i-1} are respectively the times at which the i^{th} and $(i-1)^{th}$ pulls are effected. We define the corresponding output, y_i to be,

$$y_i = \frac{|q_i - q_{i-1}| - c}{c} \quad (4)$$

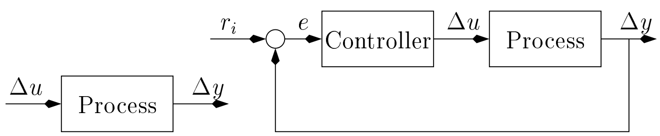
We want to predict u_i such that we get y_i close to 0. The independent variable i is the sample number. As is done in control problems, we define the input and output variables to be deviation variables from some reference values.

$$\Delta u_i = u_i - U_r \quad (5)$$

and

$$\Delta y_i = y_i - Y_{ss} \quad (6)$$

where U_r is some reference value of u (in traditional control applications, this is taken as the steady state value). U_r



a) Open Loop System b) Closed Loop System

Figure 1. Open loop and Closed Loop Systems

has been arbitrarily chosen as the mean of TTR_{min} and TTR_{max}. It has been found that U_r is not a sensitive parameter. Y_{ss} is the steady state value of the output. As the desired value of Y_{ss} is 0, we have,

$$\Delta y_i = y_i \tag{7}$$

The open loop system depicting our process can be diagrammatically represented as in Figure 1(a).

2.3 Logic of Closed Loop Control

We next explore whether putting this process in a feedback loop with a controller will help achieve the required regulation: maintaining y_i at 0 for all i. Consider the behavior of the closed loop system in Figure 1(b). Here r_i denotes the setpoint for y_i. In our case, r_i is 0. In this figure, the output $\Delta y_i = y_i$ is compared with 0 and the difference, $e_i = -y_i$ is sent as input to the controller. One can easily observe that a negative error, $e_i < 0$, corresponds to $y_i > 0$ or equivalently to $|q_i - q_{i-1}| > c$. It follows that the user should have polled earlier. Hence, when $e_i < 0$, we want Δu_i to be negative so that u_i is decreased. Similarly, $e_i > 0$ implies that Δu_i should be increased. This indicates that a positive control action is required. Here we shall assume that the dynamics relating Δu and Δy can be modelled in a linear shift-invariant framework. With this assumption, we build an ARX model of these dynamics by collecting input/output data over a sufficiently large period of time. We now describe this procedure in brief.

2.4 The Process Model

Let us suppose that the output Δy_i and input Δu_i of the process in Figure 1(a) be represented by an Auto-Regressive eXogenous (ARX) model [6] of the form,

$$\begin{aligned}&\Delta y_k + a_1 \Delta y_{k-1} + a_2 \Delta y_{k-2} + \cdots + a_n \Delta y_{k-n} \\ &= b_0 \Delta u_k + b_1 \Delta u_{k-1} + b_2 \Delta u_{k-2} + \cdots + b_n \Delta u_{k-n} + e_k\end{aligned} \tag{8}$$

where e_k is an error term. The parameters n, a_1, a_2, $\ldots$, a_n and b_0, b_1, b_2, $\ldots$, $b_n \in \Re$, are determined using the observed input-output data as follows [7]:

We write equation (8) as,

$$e_k = \Delta y_k - \phi_k^T \theta \tag{9}$$

where

$$\begin{aligned}\phi_k^T &= \begin{bmatrix}-\Delta y_{k-1} & \cdots & -\Delta y_{k-n} & \Delta u_k & \cdots & \Delta u_{k-n}\end{bmatrix} \\ \theta &= \begin{bmatrix}a_1 & \cdots & a_n & b_0 & \cdots & b_n\end{bmatrix}; \theta \in \Re^{2n}\end{aligned} \tag{10}$$

We stack the values of the errors for the observed input and output sequences $\{\Delta u_1, \Delta u_2, ..., \Delta u_N\}$ and $\{\Delta y_1, \Delta y_2, ..., \Delta y_N\}$ respectively, thereby obtaining a tall system of the form,

$$\begin{bmatrix} e_{n+1} \\ e_{n+2} \\ \vdots \\ e_N \end{bmatrix} = \begin{bmatrix} \Delta y_{n+1} \\ \Delta y_{n+2} \\ \vdots \\ \Delta y_N \end{bmatrix} - \begin{bmatrix} \phi_{n+1}^T \\ \phi_{n+2}^T \\ \vdots \\ \phi_N^T \end{bmatrix} \theta, \tag{11}$$

$$\Phi_N = \begin{bmatrix}\phi_{n+1} & \phi_{n+2} & \cdots & \phi_N\end{bmatrix}^T, \tag{12}$$

$$Y_N = \begin{bmatrix}\Delta y_{n+1} & \Delta y_{n+2} & \cdots \Delta y_N\end{bmatrix}^T \tag{13}$$

where N is a large number such that the matrix inverse exists in equation (14).

$$\hat{\theta}_N = (\Phi_N^T \times \Phi_N)^{-1} \times \Phi_N^T \times Y_N \tag{14}$$

Although we would like the left hand side in equation (11) to be zero, we can only hope for a least squares solution as it is a system with more number of equations than unknowns. The least squares estimate using the non-recursive technique is then given by [7], where,

2.5 The Adaptive Proportional Controller Approach (APC approach)

Since proportional controllers are commonly used in systems where it is desirable to speed up the response, we would like to determine if they are sufficient to achieve our objective of making $y_i \rightarrow 0$. A proportional control is of the following form ,

$$\Delta u_{n+1} = \Delta u_n + K e_n \tag{15}$$

where K is the proportional gain. A popular way to select the parameter K is the Ziegler-Nichol's method [7]. In this technique, the controller in Figure 1(b) is chosen as a proportional controller with a gain K. The value of K is gradually increased until the output, y_i starts oscillating. The value of K when this happens is taken as the critical gain, K_u. In order to determine the critical gain K_u, we take $\Delta \widehat{y_i}$ to be the output of the prediction model of Equation (8). If $\widehat{Y}(z)$ and $U(z)$ denote the z-transforms of $\Delta \widehat{y_i}$ and Δu_i respectively, then,

$$\begin{aligned}\left(1 + a_1 z^{-1} + ... + a_n z^{-n}\right) \widehat{Y}(z) = \\ \left(b_1 z^{-1} + ... + b_n z^{-n}\right) U(z)\end{aligned} \tag{16}$$

We hence obtain the transfer function of the process as,

$$H = \frac{\widehat{Y}(z)}{U(z)} = \frac{b_1 z^{-1} + b_2 z^{-2} + ... + a_n z^{-n}}{1 + a_1 z^{-1} + a_2 z^{-2} + ... + b_n z^{-n}} \tag{17}$$

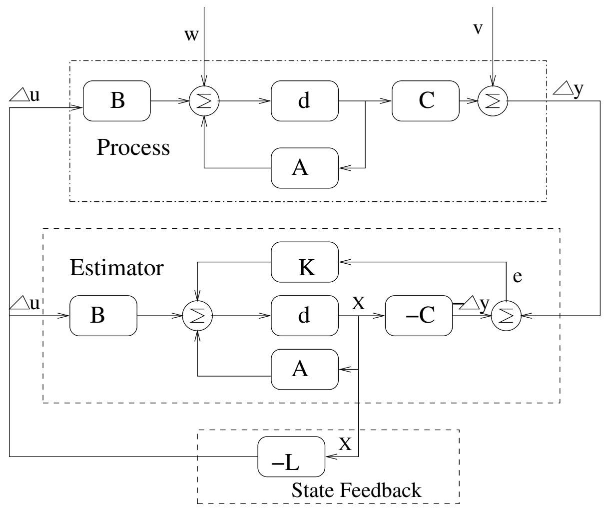


Figure 2. LQG regulator

Let the numerator and denominator polynomials of H be N and D respectively. The closed loop gain of the system using proportional controller with a gain K is then given as,

$$G_{proportional} = \frac{KH}{1+H} = \frac{KN}{D+KN} \tag{18}$$

Simple algebraic manipulation shows that

$$D + KN = z^n + \Sigma_{i=1}^{n}(a_i + Kb_i)z^{n-i} \tag{19}$$

Since the stability of a system requires that all the poles of the z-transform of the transfer function lie within the unit circle, the problem of finding the critical gain reduces to that of finding the value of K for which there exists a root of the polynomial in Equation (19) whose absolute value is equal to one. Call the corresponding gain as K_u. The Ziegler-Nichol's setting for proportional controller is $K = 0.5K_u$. A novel detuning strategy is given in [3].

Adaptive proportional control theoretic approach is not suitable for providing QoS guarantees or providing performance trade-off between control effort (network overheads) and fidelity (QoS) as APC approach is a simple approach which amplifies control error to produce control effort. In the next section we describe a stochastic control approach to provide QoS guarantee.

3 Stochastic Control Theoretic Approach

The Stochastic control theory based LQG algorithm is a state-space algorithm that yields a controller for a linear shift-invariant system corrupted with additive gaussian white noise for a quadratic performance measure. The measurements are assumed to be corrupted with additive guassian white noise as well. The process is described by a state space model as follows.

$$X(k) = \begin{bmatrix} \Delta y_{k-1} & \cdots & \Delta y_{k-n} & \Delta u_{k-1} & \cdots & \Delta u_{k-n} \end{bmatrix}^T \tag{20}$$

$$X(k+1) = AX(k) + Bu(k) + w(k) \tag{21}$$

$$A = \begin{bmatrix} -a_1 & -a_2 & \ldots - a_n & b_1 \ldots & b_{n-1} & b_n \\ 1 & 0 & \ldots & 0 \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 \ldots & 0 & 0 \\ \vdots & & & & & \\ 0 & 0 & \ldots & 0 \ldots & 1 & 0 \end{bmatrix} \tag{22}$$

$$B = \begin{bmatrix} b_0 \\ 0 \\ \ldots \\ 0 \end{bmatrix}, u(k) = \Delta u_k \tag{23}$$

$$\Delta y_k = \begin{bmatrix} 1 & 0 & \ldots & 0 \end{bmatrix} X(k) \tag{24}$$

where matrix $A^{n \times n}$ relates state at time *k+1* to the state at time *k*, in the absence of either a driving function or process noise. The matrix $B^{n \times m}$ relates the control input, *u*, to the states *X*. The matrix $C^{m \times n}$ in the measurement equation relates the states *X* to the output Δy.

A block diagram of a single-input-single-output system for a discrete plant with noise in the process and the measurement and with optimal control law is shown in Figure 2, where d is the delay operator. The state space model of this system is given in Equations 20, 21, 22 and 23. Equations 20 and 24 denote state equation and measurement equation respectively.

The random variables *w(k)* and *v(k)* represent the process and measurement sequences noise respectively. They are assumed to be independent of each other, white noise and with normal probability distributions such that,

$$E[w(k).w^T(j)] = W.\delta_{kj} \tag{25}$$

$$E[v(k).v^T(j)] = V.\delta_{kj} \tag{26}$$

where δ_{kj} is known as Kronecker delta function with property,

$$\delta_{kj} \begin{array}{l} = 1, k = j \\ = 0, k \neq j. \end{array} \tag{27}$$

and *E(.)* denotes the expectation operator. The matrix *W* is assumed symmetric and positive semi-definite and *V* is assumed symmetric and positive definite.

where *X(k)* is called state vector, *u(k)* the control effort, *A* the state matrix, *B* the input matrix, *C* the output matrix and *w* and *v* are noise in process and measurement. For optimal control we want to pick *u(k)* so that a quadratic cost function to be minimized of the form,

$$J = E[\textstyle\sum_{k=0}^{N}[\Delta y^T(k)Q\Delta y(k) + u^T Ru(k)]] \tag{28}$$

where *Q* and *R* are symmetric weighting matrices, and further $Q \geq 0$ and $R > 0$. Equation 28 provides a performance trade-off between $\Delta y(k)$ and $u(k)$ by adjusting the matrices Q and R.

Since all control efforts have a cost associated with them.

$$\begin{array}{l} u^T R u > 0, \forall u \neq 0 \\ \Delta y^T Q \Delta y \geq 0, \forall \Delta y \end{array} \tag{29}$$

We wish to minimize Equation (28) subject to

$$\begin{array}{l} -X(k+1) + AX(k) + Bu(k) = 0 \\ k = 0, 1, 2, ...N \end{array} \tag{30}$$

to get optimal control law. The control law is of the form $u = -L(k)\hat{X}(k/k)$, where *L(k)* is a constant gain matrix and $\hat{X}(k/k)$ is an estimate of the states at stage *k* based on information available up-to *k*. *L(k)* is obtained as the unique, positive definite solution of a discrete time Riccati equation [7] as follows.

$$\begin{array}{l} S(k) = A^T S(k+1) A - (A^T S(k+1) B) \\ .(B^T S(k+1) + R)^{-1} (B^T S(k+1) A) \end{array} \tag{31}$$

The equation of *L(k)* is as follows.

$$L(k) = (Q + B^T S(k+1) B)^{-1} (B^T s(k+1) B) \tag{32}$$

As a part of the solution, the LQG algorithm yields a linear estimator of states called the kalman filter. Kalman filter recursively estimates states in two stages, first is prediction prior to actual measurements and second is correction based on current measurements [7].

Innovation (*I(k)* to the process is defined as follows.

$$I(k) = \Delta y(k) - C\hat{X}(k/k-1) \tag{33}$$

Residual (*R(k)* to the process is defined as follows.

$$R(k) = \Delta y(k) - C\hat{X}(k/k) \tag{34}$$

where $\hat{X}(k/k-1)$ is the prediction of states at stage *k*, based upon information available up-to *(k-1)* and $\hat{X}(k/k)$ is the correction of states at stage *(k)*.

4 QoS Guarantee providing Algorithm (QSG Algorithm)

We describe below algorithm to provide QoS guarantee in disseminating dynamic data in wide area network. We define fidelity, F, in Section 1.1. Let us assume the fidelity requirement for a user be F_{req}. Let us consider the fidelity achieved so far computed by the client be F_{comp}.

Client polls after every *t* (say, 10) refreshes, the number of miss refreshes from the server and computes the fidelity achieved so far. If F_{comp} is less than F_{req}, *R* is decreased by a factor λ_1. If F_{comp} is greater than F_{req}, *R* is increased by a factor λ_2.

Lower and upper bounds of *R* are taken heuristically to be 1 and 1000 respectively.

In the next section we describe the results of experimental evaluation of control-theoretic TTR adjustment and compare the overall performance with the Adaptive TTR and the APMT approaches.

5 Performance Evaluation

In this section we will discuss the simulation environment, metrics used for the experiment, and present the results.

5.1 Simulation environment

The algorithm was evaluated using a prototype source that replayed traces of dynamic data. The experiments assume that the network latency in polling and fetching dynamic data items from the source is fixed and is negligible.

The performance of the algorithm was evaluated using real-world traces. The presented results are based on stock price traces (i.e., history of stock prices) of a few companies obtained from *http://finance.yahoo.com*. The traces were collected at a rate of 2-3 stock traces per second. Since the rate of change of any stock quote is much greater than even one change per second, the traces can be considered to be real time traces. We have categorized the traces as fast, medium and slow based on a statistical measure *standard deviation* of the data item. We have taken 1000 traces, each of length 10,000, for categorization. The top one-third of the traces, which show rapid changes (i.e., *large standard deviation*) have been considered as fast changing trace, the middle one-third and the rest have been considered as medium and slow changing traces respectively. Table 1 shows examples of different types of traces and their statistical measure standard deviation. Table 2 shows the values of initial settings of the experiment.

5.2 Metrics

The algorithm was evaluated using the following metrics:

1. Network Overhead in (%), which is the number of polls normalized by the length of the trace and multiplied by 100. So, 5% Network Overhead means five polls over a trace of length 100.

2. Loss of Fidelity, F_l, which can be measured based on the total time duration for which the client was oblivious to changes exceeding user specified *c* value.

$$F_l = 1 - F$$

5.3 Comparison of Adaptive TTR, APMT and Control Theoretic Approaches

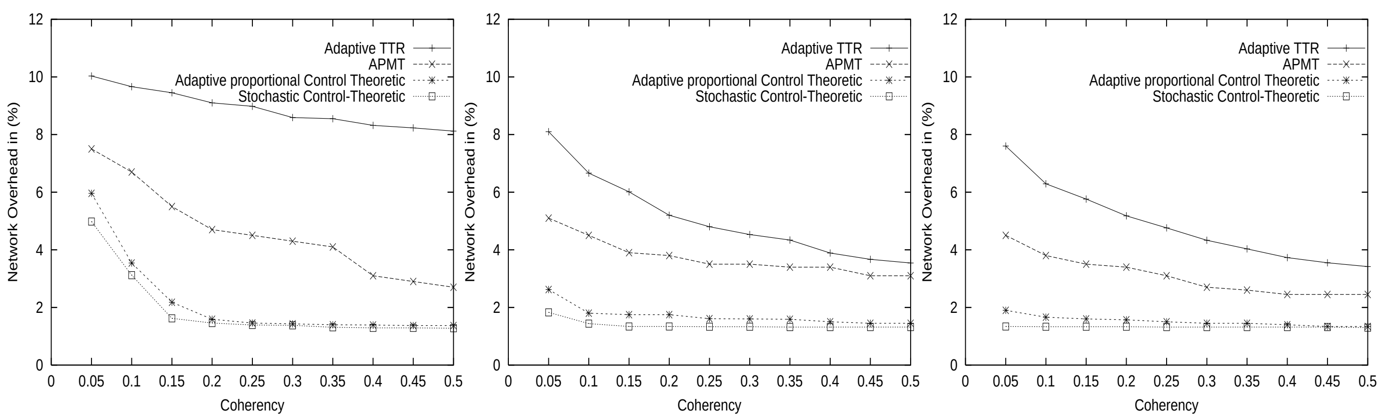
We take 25 traces of each type for conducting experiments and the plots are based on average of result obtained from these traces. Figure 3 shows Network overheads of Adaptive TTR (Heuristic), APMT and Control-theoretic approaches. From Figure 3(a) we see that for fast changing traces we can reduce considerable network overheads, as much as 75% using LQG approach as compared to Adaptive TTR algorithm. Similarly from Figure 3(b) and Figure 3(c) we see that even for medium and slow changing traces more than 67% and 56% improvement can be obtained.

Type of trace	Company	Max value	Min Value	Standard Deviation
Fast Changing	Veritas	135.75	131.50	0.0100121
Medium Changing	INTC	134.5	132.5	.00587
Slow Changing	IBM	86.48	86.16	.000335

Table 1. Type of Traces

parameter	Definition	Value
Q	weight associated with control error	1
R	weight associated with control effort	0.05
$\hat{X}(0/0)$	Initial State matrix	$\begin{bmatrix} 20 \\ 20 \\ 20 \end{bmatrix}$
W	Initial Process Covariance Matrix	$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{bmatrix}$
$p(0/0)$	Initial Error Covariance Matrix	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Table 2. Initial Settings used in the Experiment

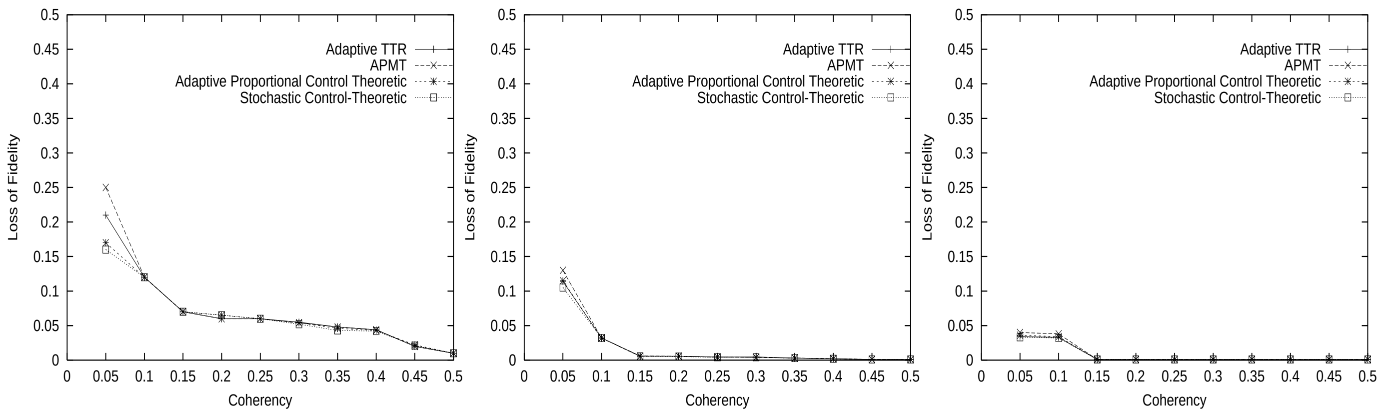


(a) Fast Changing Trace (b) Medium Changing Trace (c) Slow Changing Trace

Figure 3. Network Overheads (%) vs. Coherency

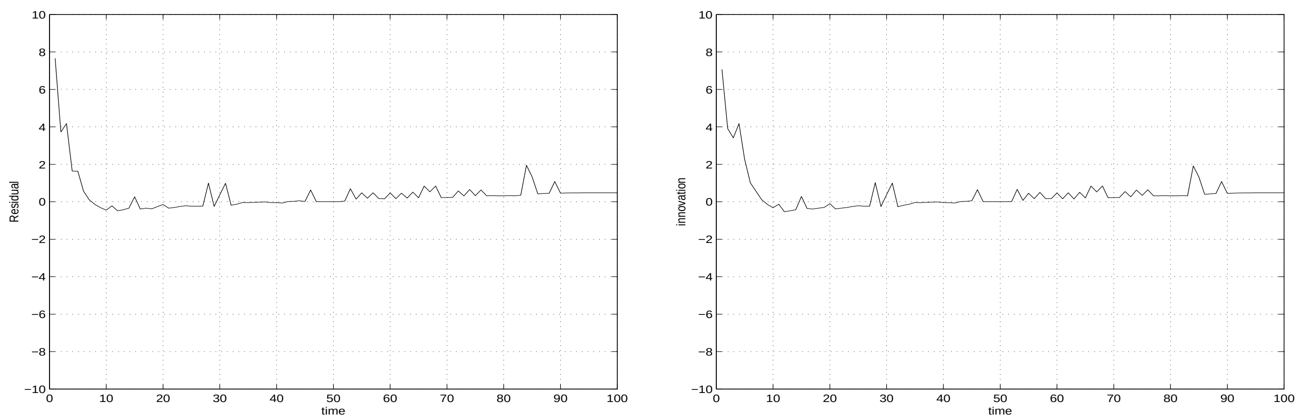
Figures 4(a), 4(b) and 4(c) show comparison of loss of fidelity for fast, medium and slow changing traces respectively. We see from figure 4(a) that LQG approach achieves less loss of fidelity as compared to other approaches for lower coherency requirements. We also see from Figures 4(b) and 4(c) that loss of fidelity is almost the same for all four approaches for medium and slow changing traces. In summary, from the experimental results we see that stochastic control theory based LQG approach performs better than other approaches in terms of achieving higher fidelity and lowering network traffic overheads. Figures 5(a) and 5(b) show performance of Kalman filter for coherency requirement=0.2 for fast changing traces. We see from Figures 5(a) and 5(b) that both innovations and residuals approache to zero over time, which ensures that Kalman filter constructs the states of the uncertain dynamical system.

Figure 6 shows performance of QSG algorithm in terms of fidelity achieved vs. fidelity required by a user after every twenty refreshes by the user in fast changing trace. Here fidelity required is taken to be constant over entire period. Figure 6(a) shows performance of QSG algorithm for fidelity requirement of .9, whereas 6(b) shows performance of QSG algorithm for fidelity requirement of .8. Figure 6(a) shows there is a 1-2 % loss of fidelity for higher fidelity requirement of .9. Figure 6(b) shows we can achieve 100 % fidelity requirement for fidelity requirement of .8. Figure 7(a) shows performance of QSG algorithm for network delays less than 1000 msec in achieving QoS guarantee for co-



(a) Fast Changing Trace (b) Medium Changing Trace (c) Slow Changing Trace

Figure 4. Loss of Fidelity vs. Coherency



(a) Innovations vs. Time (b) Residual vs. Time

Figure 5. Performance of Kalman Filter for coherency = 0.2 for fast changing Trace

herency requirement =.2 for fast changing traces. Here we take fidelity requirement varying from .65 to .9 after every twenty refreshes by the user. From Figure 7(a) we see that for fidelity .65 to .85 QSG algorithm achieves the required fidelity. Figure 7(b) shows performance of QSG algorithm for network delays in the range of 300 msec to 3000 msec (uniformly distributed) in achieving QoS guarantee for coherency requirement =.2 for fast changing traces. Here we take fidelity requirement changed from .65 to .9 after every twenty refreshes by the user. From Figure 7(b) we see that for fidelity .65 to .82 QSG algorithm achieves the desired fidelity.

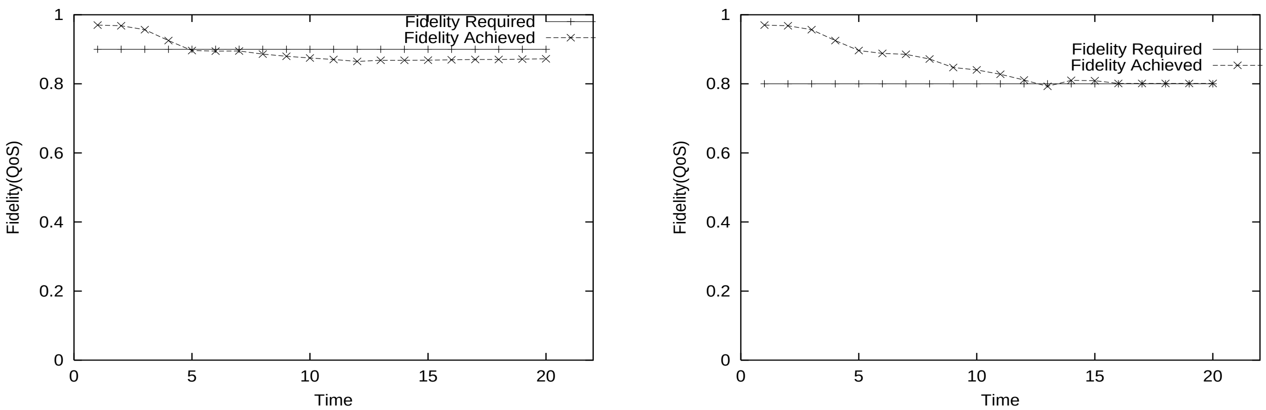
6 Related Work

The design of coherency mechanisms for dynamic data has received significant attention recently. Proposed techniques include strong and weak consistency and the leases approach [2]. Our contributions in this area lie in the definition of temporal coherency in combination with the fidelity requirements of users.

Several research groups have designed adaptive techniques for web workloads [4]. A new approach, More-less principle, is proposed in [8] to derive deadlines and periods in update transactions in real-time databases. This approach does not deal with unpredictable dynamics. We focus on adapting time-to-refresh for time-varing data, reducing network overheads.

7 Conclusions and Future Work

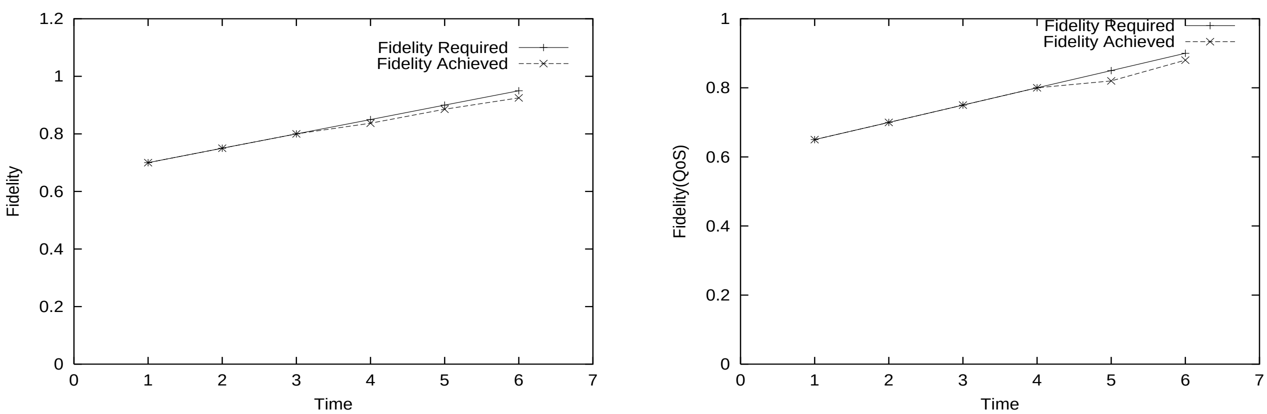
One of the attractive features of the novel approaches discussed in this paper is that these do not require sources to push changes, thus avoiding the computational overheads and state space overheads at the source. In this paper, we explored the possibility of using stochastic control theory based LQG algorithm to address the problem of intelligent polling. We show superior performance of LQG approach over other approaches in terms of communication overheads and fidelity given to the users. Our experimental results show that we can save substantial network overheads, as much as 75% in fast chaging traces, 67% in medium changing traces and 56% in slow chaging traces as compared to



(a) Fidelity Requirement = .9

(b) Fidelity Requirement =.8

Figure 6. Performance of QSG Algorithm



(a) Network Latency less than 1000 msec

(b) Network Latency: 300 msec to 3000 msec

Figure 7. Performance of QSG Algorithm for Coherency = 0.2

adaptive TTR approach. We showed that we can obtain better fidelity using LQG approach compared to other approaches. We also showed that we can provide QoS gurantee in intelligent polling using stochastic control theory based LQG approach. Finally, the control theory based solutions have the potential to scale to multivariable systems, as these are backed by formal methods.

References

[1] M.Bhide, P. Deolasee, A. Katkar, A. Panchbudhe, K. Ramamritham, and P. Shenoy, Adaptive Push-Pull: Disseminating Dynamic Web Data *IEEE Transactions on Computers special issue on Quality of Service*, June 2002, Vol 51, pp 652-668.

[2] V. Duvvuri, P. Shenoy and R. Tewari, *Adaptive Leases: A Strong Consistency Mechanism for the World Wide Web. IEEE InfoCom March 2000, Tel-Aviv, Israel.*

[3] R. Majumdar, K. Moudgalya, K. Ramamritham, Adaptive Coherency Maintenance Techniques for Time-Varying Data. In*Proceedings of the 24th IEEE Real-Time Systems Symposium*, December 2003.

[4] C. Olston, B. T. Loo, and J. Widom. Adaptive precision setting for Cached Approximate Values. In *Proceedings of the ACM SIGMOD Conference*, May 2001, Santa Barbara, USA.

[5] R. Srinivasan, C. Liang, and K. Ramamritham. Maintaining temporal coherency of virtual warehouses. In *Proceedings of the 19th IEEE Real-Time Systems Symposium*, December 1998, Madrid, Spain.

[6] P. P. J. Van-den Bosch and A. C. Van-der Klauw. *Modeling, identification and simulation of dynamical systems*. CRC Press, Boca Raton, 1994.

[7] K. J. Astrom and B. Wittenmark. *Computer-Controlled Systems - Theory and Design*. chapter 13, pages 416–437.

[8] M. Xiong and K. Ramamritham, *Deriving Deadlines and Periods for Update Transactions in Real-Time Databases*, 20th IEEE Real-Time Systems Symphosium, 1999, Phoenix, USA