Internal Server State Estimation Using Event-based Particle Filtering

Ruuskanen, Johan; Cervin, Anton

Published in:
Proceedings of the 4th International Conference on Event-Based Control, Communication, and Signal Processing

2018

Document Version:
Publisher's PDF, also known as Version of record

Internal Server State Estimation Using Event-Based Particle Filtering

Johan Ruuskanen and Anton Cervin
Department of Automatic Control
Lund University, Sweden
Email: {johan.ruuskanen, anton.cervin}@control.lth.se

***Abstract*—Closed-loop control of cloud resources requires there to be measurements readily available from the process in order to use the feedback mechanism to form a control law. If utilizing state-feedback control, sought states might be unfeasible or impossible to measure in real applications; instead they must be estimated. However, running the estimators in real time for all measurements will require a lot of computational overhead. Further, if the observer and process are disjoint, sending all measurements will put extra strain on the network.**

In this work-in-progress paper, we propose an event-based particle filter approach to capture the internal dynamics of a server with CPU-intensive workload whilst minimizing the required computation or inter-system network strain. Preliminary results show some promise as it outperforms estimators derived from analytic expression for stationary systems in service rate estimation over number of samples used for a simulation experiment. Further we show that for the same simulation, an event-based sampling strategy outperforms periodic sampling.

I. Introduction

Data centers for hosting software services consume high amounts of energy, and by utilizing dynamic resource management previous work has shown that large savings in the running cost can be achieved [1]. Using a control-theoretical approach to determine the actual management of the server resources has further shown great promise [2]. However, designing suitable control laws depends on information about server states (or parameters) that are often considered to be known [3]. This might not always be the case, as some states can depend on model choices that have no direct correspondence in reality. To utilize these states for feedback, they need to be estimated.

Practical limitations on the system itself might further make it unfeasible to estimate at each measurement event, as it will result in a great deal of overhead. Also if we consider the observer as not being a part of the server itself, sending all measurements over the network will add strain between observer and server. This overhead can be reduced by subsampling the available events either periodically or by an event-based strategy. Previous work in event-based sampling has shown good results in reducing the number of samples required to capture system dynamics [4].

To limit the scope of our research we have focused on estimating the server states of CPU-intensive software applications, such as web servers, whose performance can be accurately captured using a single server queue model [5]. As a proof of concept, we chose the M/M/1 model, which assumes Poisson distributed arrival and service rates with average rates of λ and μ respectively [6].

From the perspective of an outside observer, an intuitive way of measuring a server is by considering the *response time* of packets, i.e., the time it takes from arrival to departure [2]. We have thus considered the response times as our measurements and the service rate μ_k and queue length q_k as our unknown internal states, where k is the instance of a measurement.

Further, we have assumed that the arrival rate λ is known. This assumption corresponds a real situation with a single load balancer [7]: The balancer sets the arrival rate of the receiving servers, but the queue lengths and service rates of the different servers are still unknown and potentially of interest in forming the control law. An alternative scenario where μ is known but λ is unknown is studied in the companion paper [8].

Standard queuing models are however both non-linear and non-Gaussian, which complicates the use of standard estimators such as the Kalman Filter. Previous work in [9] has focused on the event-based version of the Extended Kalman Filter in estimating server states.

In this work we instead focus on the sequential Monte Carlo methods known as Particle Filters, as they are not limited by assumptions of linearity or Gaussian noise [10]. We introduce an event-based scheme for the bootstrap particle filter for server state estimation [11]. The idea of event-based particle filtering has not yet received much attention in the research community but includes the recent work in [12] among others.

II. State-Space Representation

For the assumed M/M/1 model, the response times are distributed according to an Erlang distribution, $T_k \sim \mathcal{E}(q_k, \mu_k)$. Further, the probability that the queue is q long at time t given an initial length of i is given by the following analytic expression:

$$P_q(t) = e^{-(\lambda+\mu)t}\Big(\rho^{(q-i)/2} I_{q-i}(at) + \rho^{(q-i-1)/2} I_{q+i+1}(at) + (1-\rho)\rho^q \sum_{j=q+i+2}^{\infty} \rho^{-j/2} I_j(at)\Big). \quad (1)$$

Here, $\rho = \lambda/\mu$, $a = 2\mu\sqrt{\rho}$, and $I_i(x)$ is the modified Bessel function of the first kind of order i. At first glance, this expression looks horrendous, but luckily there exists good theory for speeding up its evaluation [13]. A discrete distribution of the queue length q at instance k can be

formed as $q_k \sim \mathcal{P}(q_{k-1}, \mu_{k-1}, \lambda_{k-1}, t_k)$, where each entry has the probability shown by (1). Using this knowledge we can construct a discrete-time stochastic state-space model of the server as

$$\begin{aligned} \mu_k &\sim \mathcal{N}(\mu_{k-1}, \sigma_\mu t_k), \\ q_k &\sim \mathcal{P}(q_{k-1}, \mu_{k-1}, \lambda_{k-1}, t_k), \\ T_k &\sim \mathcal{E}(q_k, \mu_k). \end{aligned} \tag{2}$$

Here we have assumed that no prior knowledge of the service rate is known, and therefore it is given the dynamics of a random walk. Since our aim is to implement an event-based version the particle filter, our state-space model must be adapted to compensate for the varying intersample time. For the distribution of the queue length this is already included, but not for the service rate. Instead we extend the random walk with a time dependent variance and assume that it grows linearly with time.

III. Estimation

A. Analytic Expression for Stationary System

Good analytic results exist to calculate the expected queue length using Little's Law. Remarkably, Little's Law does not make any assumptions on service and arrival rate distributions, the amount of servers, or even if every server has its own queue or not [14]. The expected response time can be calculated using results explained in [6]:

$$\begin{aligned} q &= \lambda T, \\ T &= \frac{1}{\mu - \lambda}. \end{aligned} \tag{3}$$

From these expressions we can form a pair of naive estimators for the service rate and the queue length as

$$\begin{aligned} \hat{q}_k &= \lambda_k T_k, \\ \hat{\mu}_k &= \lambda_k + \frac{1}{T_k}. \end{aligned} \tag{4}$$

Here however we run into a problem with the estimator for the service rate, as the Erlang distribution is a generalization of the Gamma distribution. The smaller the queue length, the higher the probability becomes of measuring a response time close to 0, and at $q = 1$ the distribution reduces to an Exponential distribution whose PDF actually grows towards infinity as $T \to 0$. Since the naive service rate estimator depends on dividing by our response time measurement, this will lead to problems with very large and queue length dependent variance. To combat this we introduced a lowpass filter for the response times before utilizing the service rate estimator. The response times for the queue length was kept unfiltered.

B. Particle Filter

A problem with the estimators derived from the analytic expressions is that they assume the underlying stochastic process to be stationary, which disregards any transient behaviour of the queue in cases where μ or λ are time dependent. Tracking can potentially be improved if this is taken into consideration. Further, at least for the service rate estimator the underlying analytic expression is only generalized to the M/M/1 case and similar analytic expressions for other queue models might be intractable.

To account for the non-stationarity we have considered a particle filter built upon our stochastic state-space model (2). Particle filters are a group of estimators that uses a Monte Carlo-based inference approach to approximate the Bayesian filter equations

$$\begin{aligned} p(\boldsymbol{x}_k|\boldsymbol{Y}_k) &= \frac{p(\boldsymbol{y}_k|\boldsymbol{x}_k)p(\boldsymbol{x}_k|\boldsymbol{Y}_{k-1})}{p(\boldsymbol{y}_k|\boldsymbol{Y}_{k-1})}, \\ p(\boldsymbol{x}_k|\boldsymbol{Y}_{k-1}) &= \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})p(\boldsymbol{x}_{k-1}|\boldsymbol{Y}_{k-1})d\boldsymbol{x}_k. \end{aligned} \tag{5}$$

Suppose that we have a set of N particles $\boldsymbol{X}_{k-1}$ with the corresponding weights $\boldsymbol{W}_{k-1}$, a time series of measurements $\boldsymbol{Y}_{1:M}$, and a proposal distribution $q(x_k|x_{k-1}, y_k)$ that we can draw samples from. Further, assume that the particles and their weights form an empirical approximation to the posterior distribution at step $k-1$,

$$p(\boldsymbol{x}_{k-1}|\boldsymbol{Y}_{k-1}) \approx \sum_i W_{k-1}^i \delta(\boldsymbol{x}_{k-1} - X_{k-1}^i), \tag{6}$$

We can then form an empirical approximation of the posterior distribution at the next step k by first drawing a set of new particles using the proposal distribution

$$X_k^i \sim q(x_k|X_{k-1}^i, y_k), \tag{7}$$

and then updating the corresponding weights as

$$\begin{aligned} W_k^i &= \frac{p(y_k|X_k^i)\omega_k^i}{\sum_j p(y_k|X_k^j)\omega_k^j}, \\ \omega_k^i &= \frac{p(X_k^i|\boldsymbol{Y}_{k-1})}{q(x_k|x_{k-1}, y_k)} W_{k-1}^i. \end{aligned} \tag{8}$$

The initial set of particles can be drawn from some initial distribution $X_0^i = p(x_0)$. Since we have not assumed any connection between the initial distribution and any measurement it is standard to give the initial weights the uniform value of $W_0^i = 1/N$. From the set of estimated posterior distributions, the first and second moments can be calculated as

$$\begin{aligned} \hat{\boldsymbol{x}}_{k|k} &= E_{p(\boldsymbol{x}_k|\boldsymbol{Y}_k)}(\boldsymbol{x}_k) \approx \sum_i W_k^i X_k^i, \\ \hat{P}_{k|k} &\approx \sum_i W_k^i (X_k^i - \hat{\boldsymbol{x}}_{k|k})(X_k^i - \hat{\boldsymbol{x}}_{k|k})^T. \end{aligned} \tag{9}$$

Only following this scheme does however often lead to a condition called weight degeneracy, where the particles start to deviate more and more from the true underlying path, and the variance of the weights continue to grow for each new step. Most particles will thus receive almost no weight, which will give worse and worse performance as time grows. By introducing a resampling step, where the particles are resampled based on their weight as $P(X_k^{a_k^i} = X_k^i) = W_k^i$ after the weight update step, the degeneracy can be mitigated [10].

Finding a good proposal distribution $q(x_k|x_{k-1}, y_k)$ can be tricky. Instead we can disregard the current measurement

value y_k and choose the proposal distribution as simply the state propagation distribution of our state-space model. The corresponding weight update then reduces to a normalized version of the measurement distribution of the model,

$$\begin{cases} X_k^i & \sim p(\boldsymbol{x}_k | X_{k-1}^{a_{k-1}^i}), \\ W_k^i & = \frac{p(\boldsymbol{y_k}|X_k^i)}{\sum_j p(\boldsymbol{y_k}|X_k^j)}. \end{cases} \tag{10}$$

This algorithm is referred to as the bootstrap particle filter [11], and it has the advantages of being simple to implement as long as the state-space model exists. Using our model (2), the proposal distribution and weighting function of the bootstrap particle filter become

$$\begin{aligned} p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) &= \begin{cases} \mathcal{N}(\mu_{k-1}, \sigma_\mu t_k), \\ \mathcal{P}(q_{k-1}, \mu_{k-1}, \lambda_{k-1}, t_k), \end{cases} \\ p(\boldsymbol{y}_k|\boldsymbol{x}_k) &= \mathcal{E}(T_k|\hat{q}_k,\ \hat{\mu}_k). \end{aligned} \tag{11}$$

IV. Event-Based sampling

For event-based sampling, we consider the well-known strategy known as Send-On-Delta (SOD) [15]. The SOD method generates a new event, referred to as a point-value measurement z_k, if the actual measurement y_k differs from the old event z_{k-1} more than a set limit Δ:

$$z_k = \begin{cases} y_k & \text{if } |y_k - z_{k-1}| \geq \Delta, \\ z_{k-1} & \text{if } |y_k - z_{k-1}| < \Delta. \end{cases} \tag{12}$$

If the difference is below the limit the old value is simply kept; this is referred to as a set-value measurement.

Since the goal of our event-based strategy is to save overhead, no set-valued measurements will be used to update the estimations. Only when a new measurement is triggered will the filters compute a new update. For the stationary estimator, this will not affect the queue length estimation as it is only dependent on the current measurement. The service rate estimator might however receive an improvement from the set-values as it uses a lowpass filtered version of the response time. Improvements might also be gained in the particle filter, giving it time to adapt beyond possible transients. The question of how much the set values could improve the estimations is out of scope of this work-in-progress paper.

By only updating the estimate at the point-value measurements, extending our two filters to account for the SOD is a straightforward endeavour. We can simply update the estimates at the point values, taking into account the varying intersample time t_k.

V. Simulation & Results

To evaluate the performance, an environment was implemented in Julia to accurately simulate a M/M/1 queue realization. In this environment a simulation experiment was performed in order to test how accurately the dynamics could be captured using the two estimators with different subsampling techniques.

The simulation included a sequence over 150 seconds, where $\lambda = 4$ and $\mu = 10$. In the interval $t = [50, 100]$ the service rate was dropped to $\mu = 5$ to simulate a sudden disruption. For the particle filter we used 1000 particles and $\sigma_\mu = 2$. The lowpass filter for the stationary estimator was implemented as an exponential smoothing filter with $\alpha = 0.2$.

The subsampling itself was performed on the measurements generated from a simulation. To subsample periodically, the time from the last sample was tracked. Then the next measurement that occurred with a time difference greater than a set sampling time would be chosen as the new sample. This approach will however have some jitter in the time between samples, but it was deemed acceptable. Changing the sampling time would then yield different levels of subsampling. The event-based subsampling was instead performed by keeping track of the value of the previous sample, and a new sample was only generated when the measurement difference was larger than a set Δ. By varying Δ, different levels of event-based subsampling could be achieved.

To capture the statistics of the estimators, a Monte Carlo estimation for the sequence with 1000 repeated runs was performed. In each run, the realization of the M/M/1 queue dynamics was also resimulated to guarantee that the statistics are true to the underlying dynamics and not one particular realization. The resulting mean squared error of the service rate and the queue length for the different estimators and sampling strategies are shown in Figure 1 and 2. The lines show estimated means and the opaque areas show the interval of a standard deviation from the estimate.

It should be stressed that this is only one simulation experiment, which is definitely not enough ground for judgment. More experiments will be performed in future work.

VI. Conclusion & Future Work

The results show that the particle filter yields better estimation of the service rate than the stationary estimator in terms of the mean and standard deviation in this simulation. The SOD strategy gives further improvements. For the queue length the two estimators performed equally well for both sampling strategies.

An interesting result is that the estimates sometimes seem to improve even though we do not use all available samples. A plausible explanation could be that both our state-space model and the analytic formulas assume nothing about the dependence of response time measurements. Consecutive packets waiting to be served in the queue share the same realizations of the departure times of packets further ahead in the queue, which introduces dependence in the measurements. Further investigation is warranted.

The choice of the M/M/1 model is mainly done for the purpose of demonstrating the feasibility of using particle filters to capture unknown states in the server. In order for these methods to be usable in any real applications, extensions must be made to more realistic server models.

So far we have simply ignored the set-valued measurement on the premise that we want to save as much computations as possible. Here the question naturally arises of how much better the estimation would become if all these were used

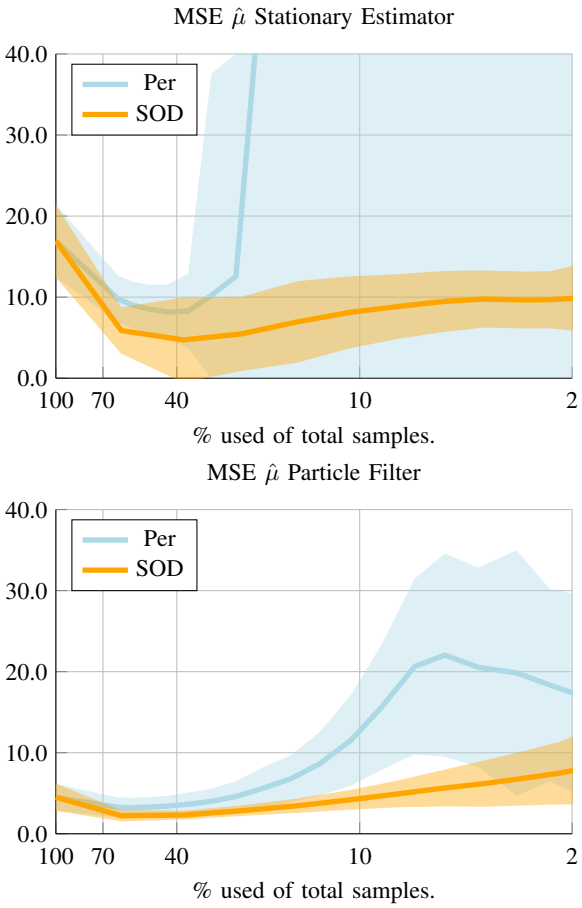

Figure 1: Service rate error over the amount of samples used for the periodic (Per) and SOD subsampling schemes.

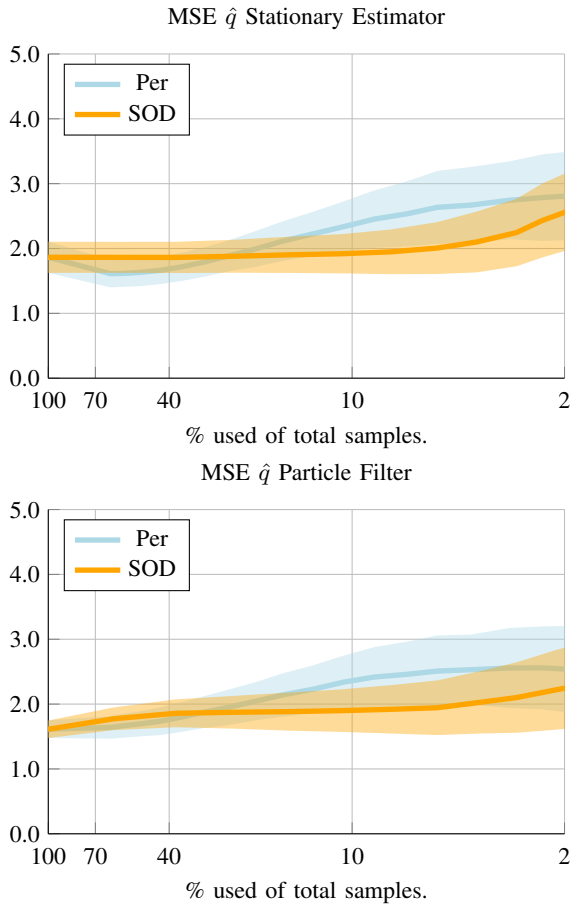

Figure 2: Queue length error over the amount of samples used for the periodic (Per) and SOD subsampling schemes.

as well. Or even, could there be improvements if one would only use some set-values to minimize the effects of possible transients? If this were the case then there could be a trade off in the amount of set-values included and the acceptable gain in computational overhead.

Acknowledgments

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The authors are members of the LCCC Linnaeus Center and the ELLIIT Excellence Center at Lund University.

References

[1] R. Bianchini and R. Rajamony, "Power and energy management for server systems," *IEEE Computer*, pp. 68–76, 2004.

[2] T. Patikirikorala, A. Colman, J. Han, and L. Wang, "A systematic survey on the design of self-adaptive software systems using control engineering approaches," in *Proc. 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, 2012.

[3] M. Kitaev and V. Rykov, *Controlled Queueing Systems*. CRC Press, 1995.

[4] Q. Liu, Z. Wang, X. He, and D. Zhou, "A survey of event-based strategies on control and estimation," *System Science & Control Engineering*, vol. 2, pp. 90–97, 2014.

[5] J. Cao, M. Andersson, C. Nyberg, and M. Kihl, "Web server performance modeling using an M/G/1/K*PS queue," in *10th International Conference on Telecommunication*, Papeete, Tahiti, 2003.

[6] L. Kleinrock, *Queueing Systems Volume 1: Theory*. Wiley-Interscience, 1975.

[7] J. Dürango, M. Dellkrantz, M. Maggio, C. Klein, A. Papadopoulos, F. Hernández-Rodriguez, E. Elmroth, and K.-E. Årzén, "Control-theoretical load-balancing for cloud applications with brownout," in *53rd IEEE Conference of Decision and Control*, Los Angeles, CA, USA, 2014.

[8] A. Cervin, "Event-based control and estimation of server systems," in *Proc. 4th International Conference on Event-Based Control, Communication and Signal Processing*, Perpignan, France, 2018.

[9] M. Dellkrantz, M. Kihl, A. Robertsson, and K. Åström, "Event-based response time estimation," in *7th International Workshop on Feedback Computing*, San Jose, CA, USA, 2012.

[10] A. Doucet and A. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," in *The Oxford Handbook of Nonlinear Filtering*, D. Crisan and B. Rozovsky, Eds. Oxford University Press, 2011.

[11] N. Gordon, D. Salmond, and A. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEEE Proceedings F (Radar and Signal Processing)*, vol. 140, 1993.

[12] S. Davar and A. Mohammadi, "Event-based particle filtering with point and set-valued measurements," in *25th European Signal Processing Conference*, Kos, Greece, 2017, pp. 211–215.

[13] J. Abate and W. Whitt, "Calculating time-dependent performance measures for the M/M/1 queue," *IEEE Transactions of Communcation*, vol. 37, pp. 1102–1104, 1989.

[14] J. Little and S. Graves, "Little's law," in *Building Intuition: Insights From Basic Operations Management Models and Principles*. Springer Science + Business Media, LLC, 2008, ch. 5, pp. 81–100.

[15] M. Miskowicz, "Send-on-delta concept: An event-based data reporting strategy," *Sensors*, vol. 6, pp. 49–63, 2006.