CHARACTERIZING ROLLBACKS IN PARALLEL OPTIMISTIC SIMULATIONS

Dhananjai M. Rao

Department of Computer Science and Software Engineering
Miami University
510 E. High Street, Oxford, OH, USA
raodm@miamiOH.edu

ABSTRACT

Rollbacks are widely used to maintain causality parallel optimistic simulations, specifically in Time Warp synchronized simulations. Despite their significance, literature is scant on fundamental characterization of the two key metrics of rollbacks, namely – ① *inter-rollback cycles*, *i.e.,* how many event processing cycles elapse before a rollback occurs, and ② *rollback lengths*, *i.e.,* how many cycles does a rollback cancel. This study proposes an experimental method to characterize rollbacks via statistical analysis. We have conducted experimental analyses using a widely used synthetic benchmark called PHOLD. We have conducted 1000s of simulations with different combinations of PHOLD settings on two different computational-clusters to analyze rollback-profiles of a broad spectrum of parallel simulation configurations. Our analysis shows that both rollback metrics are geometrically distributed with their aggregate characteristics following a normal distribution. Interestingly, the overarching metrics from 500 different simulation configurations are also normally distributed.

Keywords: Parallel Simulation, Rollback, Time Warp, Statistical Fitting

1 INTRODUCTION, BACKGROUND, AND MOTIVATION

Parallel Discrete Event Simulation (PDES) is a widely used methodology for accelerating simulations of large, complex models. For PDES, a model is partitioned into independent parallel processes running on different compute nodes. However, to maintain causality, *i.e.,* the correct order of event processing, the processes have to be synchronized with each other. There are two major categories of synchronization approaches, namely: conservative and optimistic methods (Jafer, Liu, and Wainer 2013). Recent growth number of cores, CPU speeds, and RAM capacity has catalyzed increased attention on optimistic synchronization approaches because they have shown to deliver improved scalability and performance (Rao 2016). The most widely used algorithm for optimistic synchronization is the Time Warp algorithm.

1.1 Background and Terminology on Time Warp

In Time Warp a PDES is organized as a set of independent Logical Processes (LPs) that interact with each other by exchanging *virtual timestamped* events. Each LP "optimistically" (*i.e.,* as quickly as it can) processes events scheduled for it, updates its state, generates new events, and advances its Local Virtual Time (LVT). Due to optimistic processing, an LP may receive events in the past (*i.e.,* at a lower LVT), typically due to network latency. Such an event is called a *straggler* and signifies a causal violation. To recover from

a causal violation, the LP triggers a rollback mechanism which involves three key steps. First, the LP uses the saved states to revert back to a LVT prior to the causal violation. Second, the LP cancel events sent earlier using anti-messages, which in turn can cause further rollbacks. Finally, the LP reprocess messages in their correct time stamp order. Rollbacks are manifestations of the synchronization overheads in a Time Warp PDES. There are two key metrics associated with rollbacks, namely:

1. **Inter-rollback cycles**: This is the number of event processing cycles (a cycle involves processing 1 or more concurrent events) that are completed between rollbacks. This metric essentially describes the frequency of rollbacks as experienced by an LP. This metric is independent of LVT as LVT advancement is arbitrary and depends on the model.
2. **Rollback length**: The number of event processing cycles reverted by a rollback is called the *rollback length*. This essentially determines the wasted work that has to be canceled due to a rollback.

1.2 Motivation for the proposed work

Rollbacks are overheads of optimistic synchronization and need to be controlled to enable efficient, high-performance PDES. Consequently, rollback management has been an active area of research for over two decades now. Many investigations have proposed different static and adaptive methods for managing rollbacks and mitigating their negative impacts. However, literature characterizing the two fundamental metrics of rollbacks is scarce, if not absent. Having a sound, clear understanding of rollback characteristics is important for the following reasons: ❶ Having a model of rollback characteristics enables researchers and practitioners of PDES to utilize the knowledge to assess and optimize their PDES platforms. ❷ Having a generic reference model for comparison enables systematic and scientific analysis of rollbacks and associated issues. ❸ It provides explanatory power to reason about observed patterns – a weak analogy would be to compare it with the big-O notation in computing, which facilitates reasoning about the performance of algorithms. ❹ The characteristics can be used to develop straightforward approaches to mitigate rollbacks, possibly even in hardware, and inspire and inform novel approaches. ❺ It can be used to generate data for training and calibrating adaptive algorithms for controlling optimism. ❻ By-products of analysis can also be used to assess the effectiveness of computational platforms for PDES.

1.3 Contributions of this paper

Motivated by the aforementioned reasons this paper proposes an experimental and statistical method in Section 5 for characterizing the two key metrics of rollbacks in optimistic simulations. The simulator and synthetic benchmark used in this study are discussed in Section 3 and Section 4 respectively. The proposed method is validated using results from experiments conducted on two different compute clusters with 500 different parallel simulation configurations. Section 6 presents the experiments and results obtained by applying our method. Our experiments show that the proposed methodology elicits an effective summary of rollback characteristics. Key assumptions underlying our results are discussed in Section 7. Section 8 concludes the paper with a discussion on the key findings along with pointers to future work.

2 CLOSELY RELATED PRIOR RESEARCH

The topic of monitoring and mitigating the impact of rollbacks in optimistic PDES is an active area of research and has been ongoing for over two decades. Hence, there is a rich history and depth of literature on this topic (Jafer, Liu, and Wainer 2013). Consequently, detailed coverage is beyond the scope of this paper and we only focus on closely related research. Readers are referred to citations in our references for a literature trail on this topic. Recently, Wilsey (2016) proposed a DESMetric project to enable quantitative

modeling of simulations by generating a profile of their execution. This study has a similar theme except we focus on rollbacks and developing a simplifying statistical model for it. Prior investigations by Quaglia (2015), Carothers and Perumalla (2010), Ferscha and Luthi (1995), Chetlur and Wilsey (2006), and Liu and Wainer (2009) all involve analysis and management of rollback characteristics with the objective to improve memory and/or run time overheads of a Time Warp simulation. In contrast to these investigations, the focus of this paper is to fundamentally characterize rollback behaviors on contemporary hardware platforms.

3 MUSE: A PARALLEL SIMULATOR

Our experimental methodology for characterizing rollbacks has been assessed using an optimistic parallel simulator called MUSE (Rao 2018). MUSE has been developed in C++ and has been parallelized using the Message Passing Interface (MPI) library. MUSE uses the Time Warp algorithm with state saving approach to accomplish optimistic synchronization. A MUSE simulation is organized as a set of Logical Processes (LPs) that interact with each other by exchanging virtual timestamped events. An LP is implemented as a C++ class by overriding necessary methods in an `Agent` base class. The input, output, and state queues used for rollback operations in Time Warp are managed by the `Agent` base class in coordination with the simulation kernel. The simulation kernel implements core functionality associated with LP registration, event processing, state saving, synchronization, and Global Virtual Time (GVT) based garbage collection. The kernel also handles the task of interacting with MPI for exchanging events between LPs on different MPI-processes. The kernel uses a centralized Least Timestamp First (LTSF) scheduler queue for managing pending events and scheduling event processing for local LPs. In the context of this study, the events exchanged by LPs are broadly classified into the following two categories:

1. **Local events:** Events exchanged between LPs on the same MPI-process are called *local* events. In MUSE, all local events are directly managed by the centralized LTSF scheduler. Furthermore, LPs are permitted to generate events only into the future – *i.e.,* the timestamp on events must be greater than their Local Virtual Time (LVT). Consequently, local events are always processed in correct causal order and cannot trigger (or initiate) a rollback. In other words, if all events in a simulation are local events, then the simulation would essentially simplify to just a set of sequential simulation. However, if a rollback is triggered due to a "remote" event, then local events can cause cascading rollbacks. Consequently, local events are included in our rollback characterization method.
2. **Remote events:** Events exchanged between LPs on different MPI-processes are called *remote* events. These events are exchanged via MPI calls and can be "stragglers" – *i.e.,* have a timestamp below the LVT. Consequently, in MUSE only remote events are the primary triggers of rollbacks. However, note that when an LP rolls back, it can induce cascading rollbacks in other LPs to which it has scheduled *local* events in the future.

Another key aspect of MUSE that influences rollback behavior is its event delivery policy. In MUSE, concurrent events (*i.e.,* events with the same receive timestamp) are delivered to be processed as a single batch. Such a batch wise processing of concurrent events is important and streamlines many modeling scenarios, including digitial logic simulations, epidemic simulations (Rao 2016), etc. Since concurrent events are delivered in a single batch, *zero-length* rollbacks occur in MUSE. A zero-length rollback occurs when a straggler event with timestamp t_x arrives to an LP with LVT of t_x (*i.e.,* the LP has just finished processing events with timestamp t_x).

4 PHOLD BENCHMARK

The experimental analyses in this paper have been conducted using a synthetic benchmark called PHOLD. Proposed by Fujimoto (1990), PHOLD is widely used in PDES investigations because it has shown to effectively emulate the steady-state phase of a broad range of model (Rao and Higiro 2019). A key advantage of

PHOLD is that it includes a variety of settings that can be used to configure the benchmark to mimic characteristics of different real world models. Our PHOLD benchmark provides several parameters (specified as command-line arguments) summarized in Table 1. The benchmark consists of a 2-dimensional toroidal grid of interacting Logical Processes (LPs). The dimensions of the torus is specified via the `rows` and `cols` parameters. The total number of LPs in the simulation is `rows` × `cols`. By default, LPs are evenly partitioned across the MPI-processes used for simulation. However, the `imbalance` parameter is used to influence the partition, with larger values skewing the partition such that more LPs are assigned to some partitions. In this study we vary `imbalance` by ±25% to reflect practical parallel simulation settings where the load is reasonably balanced.

Table 1: Parameters in PHOLD benchmark

Parameter	Description	Parameter	Description
`rows`	Number of rows in model.	`cols`	Number of columns in model.
`eventsPerLP`	Initial number of events per LP.	`simEndTime`	Simulation end time.
`delay-distrib`	Event timestamp distribution*	`recvr-distrib`	Receiver ID distribution – one of: "local-remote", "uniform", "poisson", or "exponential"
`%self-Events`	Fraction of events LPs send to themselves.		
`delay` or λ	Parameter for distribution specified by `delay-distrib`.	`recvr-range`	Parameter for distribution specified by `recvr-distrib`.
`granularity`	Additional compute load per event.	`imbalance`	Imbalance in partition, *i.e.,* more LPs on some MPI-processes.
`remote-events`	Fraction of remote events when `recvr-distrib` is `local_-remote`		

The PHOLD simulation commences with a fixed number of events for each LP, specified by the `eventsPerLP` parameter. This parameter influences the number of concurrent events received by an LP, which in turn impacts rollback characteristics. For each event received by an LP a fixed number of trigonometric operations determined by value of `granularity` parameter are performed to place CPU load. In this study, the `granularity` has been varied from 0 to 50 to reflect both fine-grained and coarse-grained simulations.

For each event, an LP schedules another event to a randomly chosen adjacent LP determined by `recvr-distrib` and `recvr-range` parameters. The `selfEvents` parameter controls the fraction of events that an LP schedules to itself. The event timestamps are determined by a given `delay-distrib` and `delay` parameter. Our PHOLD benchmark supports three prevalent types of distributions observed in different types of simulations, namely: `uniform`, `exponential`, and `poisson`. The combination of parameters can be used to model different interaction patterns and simulation-time behaviors of various models. Moreover, the parameter settings can also be used to reflect both "strong scaling" and "weak scaling" scenarios as discussed in Rao (2018).

5 METHODS

Rollback characteristics are primary described using two key attributes, namely rollback frequency and rollback length. In our method, rollback frequency is determined by the number of simulation-cycles that elapse before a logical process (LP) experiences a rollback. Note that each schedule involves an LP processing 1 or

more concurrent events (*i.e.,* events with the same timestamp). The rollback length indicates how far back in virtual time a rollback resets the LVT. For example, assume LP_x optimistically completes event processing for four cycles at LVTs {5, 7, 10, 12}. Note that each cycle involves an LP processing 1 or more concurrent events (*i.e.,* events with the same timestamp). Now, assume a straggler event is then received at LVT 11. In this scenario, the *inter-rollback cycles* (first metric) is 4 cycles and the *rollback length* (second metric) is 1, because only one cycle is rolled-back. An overview of our proposed experimental method to characterize rollbacks is summarized in Figure 1. Our method involves 6 steps as discussed below:

Step 1: Instrumentation – The first step is to instrument the parallel simulator to record rollback occurrences. A key aspect of this instrumentation is it should be minimally intrusive to ensure that the instrumentation does not skew the behavior of the simulation. In our simulator, we have used two strategies to minimize intrusion. First, we record statistics in an internal memory buffer. We periodically write the buffer to local temporary storage on each compute node.

Step 2: Run simulation and gather profile – In this step a specific configuration of the simulation is executed k times and rollback profiles are recorded for further analysis.

Step 3: Auto-Correlation Function (ACF) analysis – Prior to fitting statistical distributions, it is important to establish that the samples are independent – *i.e.,* for a given agent, the rollbacks are mutually independent. In this study we have used the standard Auto-correlation Function (ACF) analysis approach to determine if inherent correlations exists between rollbacks. Auto-correlation is computed by sliding the sequence of inter-rollback cycles, 1 at a time (*i.e.,* lag of 1), and comparing them against the original sequence to compute the Pearson correlation coefficient.

Independence of the rollback occurrences is established by comparing the auto-correlation coefficients against 95% significance ($\alpha = 0.05$) ACF threshold computed as $\pm 2/\sqrt{N}$, where N is the number of observations. The method is based on the statistical inference that – if a time series is completely random, and the sample size is large, the lagged-correlation coefficient is approximately normally distributed with mean 0 and variance $\frac{1}{N}$. Accordingly, if a large number of auto-correlation values are below the ACF threshold then the observations are deemed purely random or independent. The ACF analysis needs to be performed for each agent that experiences rollback and collectively analyzed to ensure the data is amenable to further statistical analysis.

Step 4: Per-LP statistical distribution fitting – Having established overall independence of samples, we propose to use statistical distribution to summarize the observed inter-rollback cycles and rollback lengths. Fitting statistical distributions to the observed data has been conducted using R version 3.2.1 and the `fitdistrplus` package. We analyzed several different statistical distributions, including: Exponential, Poisson, Negative binomial, Geometric, Normal, Uniform, and Weibull using Maximum Likelihood Estimation (MLE) method for fitting. Among the standard distributions, the Geometric, Exponential, and Poisson distributions were the only ones that provided sufficiently low Standard Error (SE) among the various fits. The distribution with the lowest SE is chosen for each LP that experiences rollbacks. In our experiments the Geometric distribution yielded the best fit for per-LP inter-rollback cycles in almost all of the configurations explored in this study.

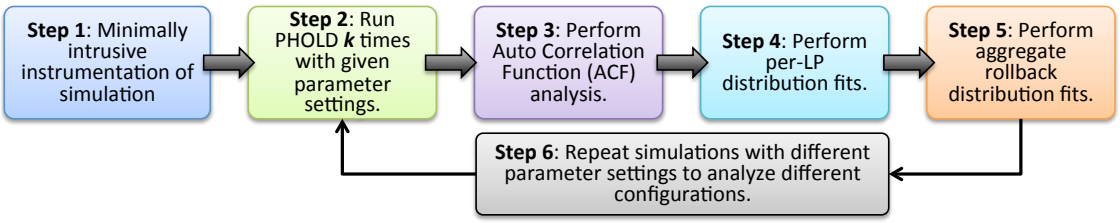


Figure 1: Overview of proposed methodology for characterizing rollbacks in parallel simulations

Step 5: Aggregate distribution fitting – The previous step yields statistical approximations of rollback characteristics on a per-LP basis. Consequently, there could be 1000s of such fits. In this step, we perform a second level of statistical fitting to elicit aggregate characteristics. First, the per-LP fits from the previous step are separated into distinct sets based on the best fitting distribution types. Next, statistical fitting is performed for each set to identify aggregate characteristics. In our experiments, the Geometric distribution was the only set of per-LP fits that needed to be analyzed. We found that the probability of Geometric distributions (g_p) of inter-rollback cycles and rollback lengths followed a Normal distribution, *i.e.,* $g_p \sim N_{g_p}$, where $N_{g_p} = N(\mu_{g_p}, \sigma^2_{g_p})$.

Step 6: Simulation configuration exploration – The previous steps are performed for a broad range of parameter settings to explore different simulation configurations. For each configuration, the previous three steps are repeated to obtain aggregate fits. The aggregate fits from many configuration are collectively analyzed to draw inferences and characterize rollbacks in parallel optimistic simulations. In our study, we performed a third-level of fitting and found that μ_{g_p} also follow a Normal distribution as shown in Section 6.

6 EXPERIMENTS AND RESULTS

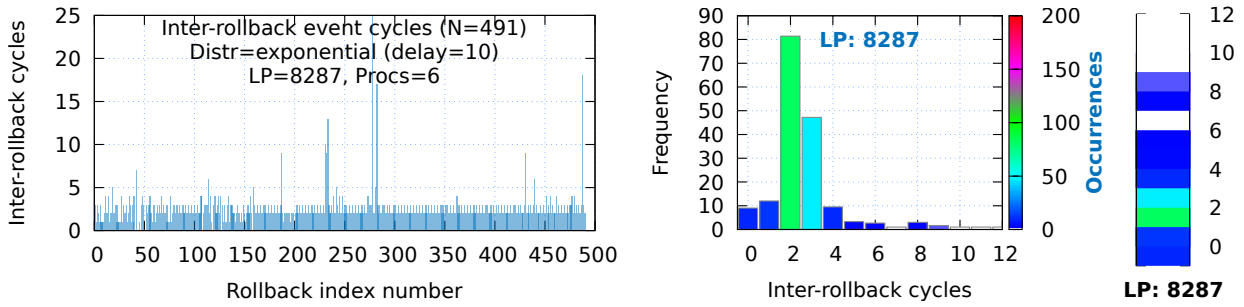
The methodology discussed in Section 5 has been used to characterize rollbacks. Specifically, profiles of rollbacks has been collected through minimally intrusive instrumentation of MUSE kernel. A broad range of PHOLD parameter settings (see Section 4) have been used to mimic a diverse set of parallel simulation configurations. The experiments have been conducted on the following two different computational clusters for cross-validation and robustness –

1. **RedHawk cluster:** Each compute node has two 12-core Intel Xeon Gold 6126 (Skylake) CPUs @ 2.6 GHz (no hyperthreading) for a total of 24-cores per compute node. Each compute node has 96 GB of DDR4 memory in Non-Uniform Memory Access (NUMA) configuration. The compute nodes are interconnected by 10 Gigabit Ethernet and run CentOS version 7.6. Our simulation software was compiled using GCC version 6.3 (at `-O3` optimization level) with OpenMPI version 4.0.
2. **Owens cluster:** Each compute node has two 14-core Intel Xeon E5-2680 v4 (Broadwell) CPUs @ 2.4 GHz (no hyperthreading) and 128 GB of RAM in NUMA configuration. The interconnect on this cluster is 100 GBPS Infiniband network and runs RedHat Enterprise Server version 7.5. On this cluster, our simulation software was compiled using Intel Compiler Collection (ICC) version 16 (at `-O3` optimization level) with `mvapich2` MPI-library version 2.2.

6.1 Overview of per-LP rollback characteristics

The profile of rollbacks for an LP (id=8287) in a parallel simulation of 10,000 LPs using 6 MPI-processes is shown in Figure 2. The chart shows average rollback profiles from 5 independent runs without outlier data for readability. The chart in Figure 2(a) shows each rollback with y-axis showing the number of event processing cycles between each rollback. Recollect that each cycle involves an LP processing 1 or more concurrent events (*i.e.,* events with the same timestamp). Figure 2(a) shows the number of schedules between each consecutive rollback for an agent (agent id = 8287) in the simulation. However, what is not apparent in Figure 2 are zero-length rollbacks. That is, several rollbacks occur consecutively before the next cycle of forward event processing.

Figure 2(b) shows a histogram of the inter-rollback event schedules averaged from 5 independent replications of the simulation for the agent. The histogram is color-coded based on the frequency of occurrences, with rollback length of 2 being the cycles the most frequent for this specific agent. Figure 2(c) shows a more compact representation of the histogram from Figure 2(b) color-coded to show higher and lower occurrence frequencies for different inter-rollback cycles.



(a) Inter-rollback schedules (b) Viewed as histogram (c) Coded

Figure 2: An example of rollbacks observed for an LP during a parallel simulation. (b) shows a histogram of the inter-rollback cycles color coded based on the frequency of occurrence. (c) shows the histogram a color-coded bar with colors representing frequency.

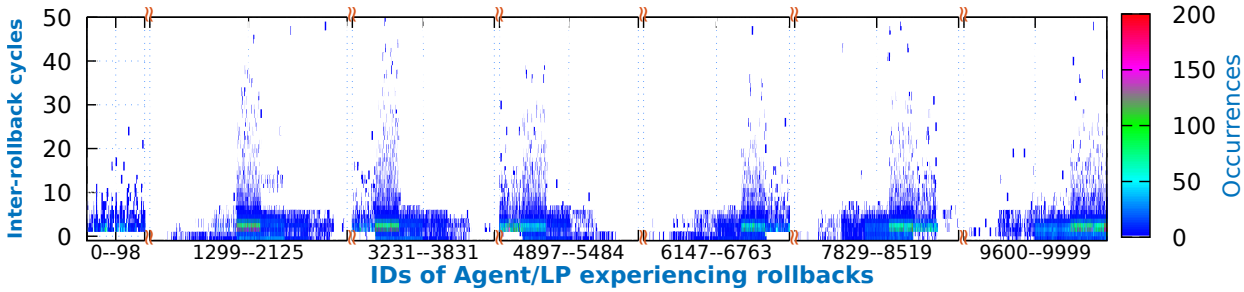


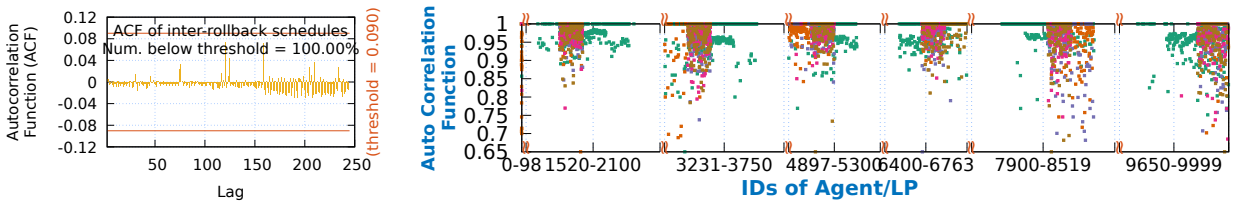
Figure 3: Histogram of inter-rollback cycles. Agents that did not experience rollbacks indicated by discontinuities in the x-axis

The compact representation introduced in Figure 2(c) has been used to enable visualization of data for all LPs that experience rollbacks, in Figure 3. The LPs that do not experience rollbacks are shown as discontinuities on the x-axis. The chart in Figure 3 shows the 7 distinct regions from the 6 process parallel simulation. The 7th region, involving agents 9600–9999, arises due to wrap around in the torus structure of the PHOLD model. The agents that are at the inter-process (accomplished via MPI) communication boundaries of the model experience higher rollback frequencies. Conversely agents further away from the inter-process communication boundaries experience fewer rollbacks.

6.2 Example results from Auto-correlation Function (ACF) analysis

Prior to fitting statistical distributions, ACF analysis has been used to ensure that the data samples are independent – *i.e.,* for a given agent, the inter-rollback cycles are mutually independent. The chart in Figure 4(a) shows the ACF corresponding to the rollback profile in Figure 2. The chart in Figure 4(a) shows the resulting auto-correlation values with lag extended to 50% of the number of observations. In this example, the resulting ACF value is 100% – *i.e.,* all of the auto-correlation coefficients are below the ACF threshold of ± 0.097, $N = 425$), leading us to infer that the inter-rollback cycles are independent. Independent observations do not have correlations between samples and are amenable to be fitted to classic statistical distributions.

The chart in Figure 4(b) shows the ACF values for all the agents that experience rollbacks in the simulation. ACF was not computed for agents with fewer than 20 rollbacks to avoid spurious outliers in statistical analyses. As illustrated by the chart, for most of the agents, their rollback characteristics do not have much



(a) Auto-correlation for LP 8287 (b) ACF for all LPs

Figure 4: Detailed and aggregate statistics from Auto Correlation Function (ACF) analysis to establish independence of samples

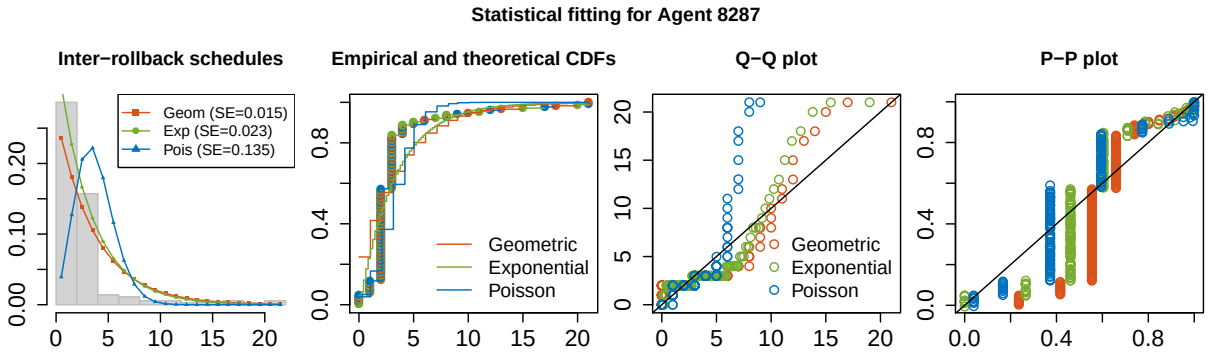


Figure 5: Result of statistical fits for inter-rollback cycles

auto-correlation with ACF values above 80%. However, some of the rollbacks do show stronger correlations due to two reasons – ❶ there were very few rollbacks for some of the agents causing them to be outliers in the data and ❷ some agents send events to themselves, and hence event exchange patterns of agents have some correlation. Consequently, rollbacks used for event cancellation also reflect this characteristic. Overall the chart in Figure 4 shows that the rollback characteristics show independence (*i.e.,* they are randomly distributed) and are amenable to further statistical analyses.

6.3 Results from per-LP statistical distribution fitting

Having established overall independence of the inter-rollback schedules, the next step (*i.e., Step 4* from Section 5) is to identify a suitable statistical distribution to characterize the data. The chart in Figure 5 shows example results from statistical fitting for the same LP (id=8287) from earlier subsections. As illustrated by the chart, the geometric distribution is the best fit, with only a small Standard Error (SE) of 0.015. The Q-Q plots show good agreement with higher frequency regions (0 to 10) but deviate at longer inter-rollback lengths due to fewer samples in those regions as this agent does not experience long inter-rollback cycles. However, the empirical vs. theoretical quantiles and the P-P plot show good agreement with the statistical fit to a Geometric distribution.

The results from fitting statistical distributions to all of the agents experiencing rollbacks is a given simulation is shown in Figure 6. As illustrated by the Standard Error (SE) curves (plotted against Y1-axis), overall the Geometric distribution had the lowest errors for all of the 3800 agents that experience rollbacks in this specific run. The exponential distribution was a close second. The Poisson distribution did not fit the data at all. The chart in Figure 6 also shows the probabilities for the Geometric distributions for each of the agents (in light orange impulses, *i.e.,* ▮) plotted against the Y2-axis.

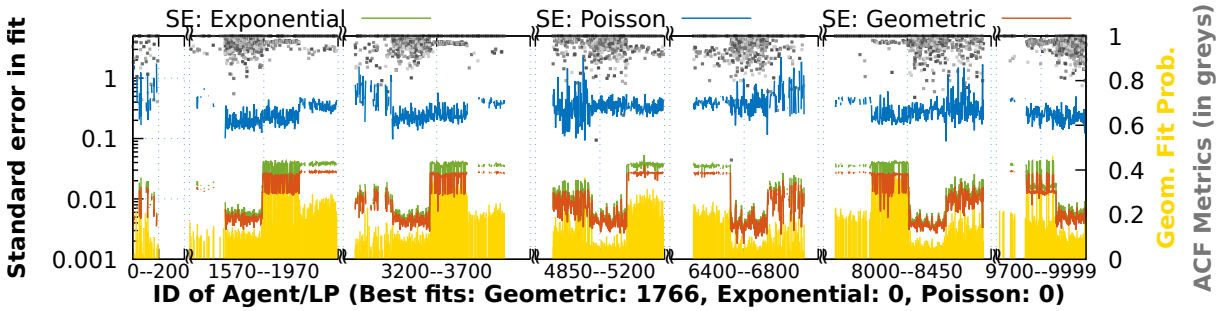
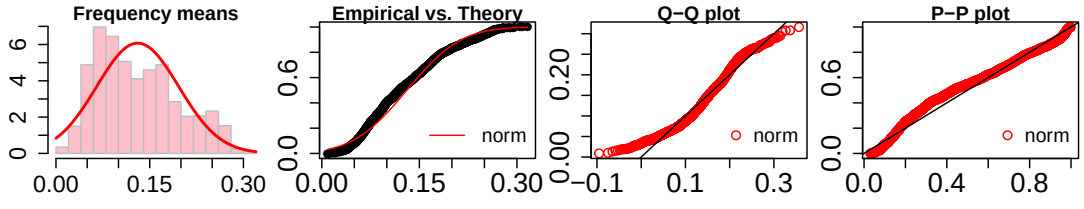


Figure 6: The overall Standard Error (SE) observed from statistical fittings for the agents experiencing rollbacks in the simulation. LPs that do not experience rollbacks are not shown.

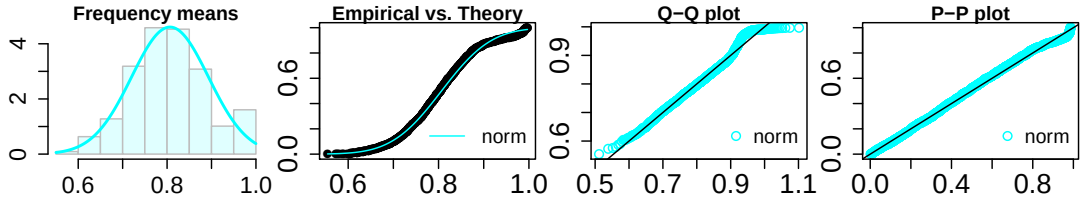
Interestingly, the rollback lengths also had a similar characteristic with the geometric distribution best fitting the observations. Moreover, the data from both `Owens` and `RedHawk` clusters showed similar characteristics across the broad range of configurations explored on both clusters.

6.4 Per-simulation aggregate statistical distribution fitting results

The next step in our method, namely *Step 5*, involves fitting a distribution to aggregate rollback characteristics. In this step, we explored different distributions and identified that the Normal distribution best characterizes both inter-rollback cycles and rollback lengths as illustrated by the charts in Figure 7.



(a) Inter-rollback cycles



(b) Rollback length

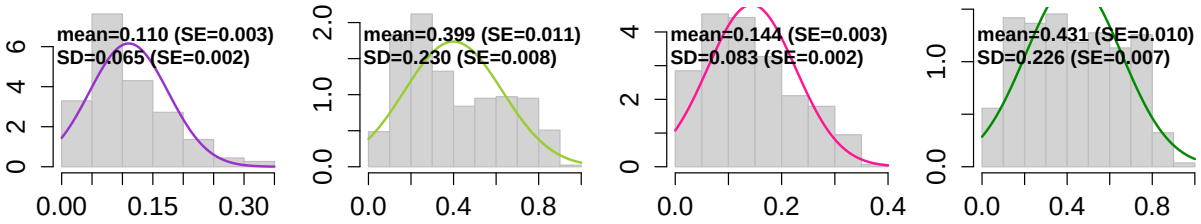
Figure 7: Aggregate fit of rollback characteristics for all LPs in one simulation configuration.

6.5 Multi-configuration statistical analysis

The final step of our methodology involves analyzing the aggregate fits from multiple simulation configurations. We have explored a broad range of simulation configurations on both clusters, including ❶ varying number of compute nodes (1–4), ❷ varying processes per-node (2–8), ❸ varying number of LPs (between 2,500–250,000), ❹ varying simulation end times (300–1000), ❺ varying optimism time window (10–1000), ❻ the scheduler queue (`3tHeap` or `2tLadderQ` (Rao and Higiro 2019)), and ❼ various PHOLD parameters in Table 1 introduced in Section 4. Overall we have explored over 500 different combinations on both

clusters, with 5 simulations per combination resulting in a total of 10,000 independent replications. The combinations have been generated using Sobol random numbers to enable systematic exploration of the multidimensional parameter space.

For each parameter-combination, the per-LP geometric fits for inter-rollback cycles and rollback lengths were computed. Next, the aggregate per-simulation Normal distribution fit was computed as shown in Figure 7. The distribution fits from each simulation configuration were collectively analyzed to elicit overarching characteristics of rollbacks from all 500 different configurations, as summarized in Figure 8. As illustrated by the charts in the figure, overall rollback characteristics on `Owens` show a normal distribution but with slight skews. The overall fits for `RedHawk` are similar, except for the rollback lengths which are sharply skewed left due to a much longer tail.



(a) Inter-rollback cycles (`Owens`) (b) Rollback length (`Owens`) (c) Inter-rollback cycles (`RedHawk`) (d) Rollback length (`RedHawk`)

Figure 8: Aggregate fits for mean (μ) of Normal fits from 500 configurations on 2 different clusters

On both clusters, the inter-rollback frequencies are slightly skewed to the left indicating that most rollbacks occur often. This inference is also supported by the rollback histogram of all LPs shown in Figure 3. The frequent rollbacks occur at the MPI-process boundaries because in MUSE only remote events can trigger rollbacks. On the other hand, LPs away from the process boundaries experience rollbacks less frequently. The inter-rollback cycles mean on `Owens` and `RedHawk` are comparable. Similarly, the rollback length characteristics (Figure 8(b) vs. Figure 8(d)) on these two clusters are comparable. However, the overall rollback length on `RedHawk` is spread over a much wider range (0.0–08) versus that on `Owens` (0.0–0.4). The longer rollbacks suggest that straggler events are arriving much later on `RedHawk` when compared to `Owens`. This observation is very symptomatic of the slower interconnect (10 GBPS) on `RedHawk` than the one on `Owens` (100 GPBS). Nevertheless, the overall patterns elicited by our approach on the two clusters are consistent. Importantly, the results bolster the applicability of our proposed method for characterizing rollbacks in optimistic parallel discrete event simulations.

7 ASSUMPTIONS AND LIMITATIONS

The approach for characterizing rollbacks is general purpose and can be applied to any optimistic parallel simulator and its models. Nevertheless, in this study, we applied our approach and conducted experiments using MUSE and PHOLD. Consequently, our experiments implicitly involve assumptions and circumscribe the inferences that can be drawn from the results, as follows – ❶ This study assumes a standalone parallel simulation that does not involve any external interactions, such as human-in-the-loop simulations. ❷ A key aspect of the parallel simulator used in this study is that only *remote* events (*i.e.,* events received from a different MPI-process) can trigger rollbacks. This study assumes that there are no other external sources (such as human-in-the-loop simulations) that can trigger rollbacks. ❸ In our study we do include zero-length rollbacks as MUSE processes concurrent events as a single batch. However, if a parallel simulator does not involve zero-length rollbacks, then their rollback profiles could be different. ❹ The rollback characterization methodology is broadly applicable. However, the experiments reported in this paper are based on a synthetic benchmark called PHOLD. Although, PHOLD is widely used as a representative benchmark in the

parallel simulation community for different analyses, it is still not an actual model. It is possible that an actual model could experience different rollback characteristics. ❺ In continuation with the previous item, our experiments used parallel simulations in their nominal operating conditions – *i.e.,* the load is reasonably balanced (we have explored some skew), the rollbacks are reasonably bounded (we have explored different bounds), computational resources such as CPU, RAM, disk, network are in nominal operating conditions etc. These settings reflect typical operating conditions in which parallel simulations would be used. If the operating conditions skew significantly then the rollback characteristics could be different. ❻ It is also important to add that our experiments were conducted on contemporary computational clusters. Consequently, significant changes to the hardware platform would also influence the observed rollback characteristics. ❼ Our experiments have explored p processes ($1 < p \leq 8$) running on n compute-nodes ($1 \leq n \leq 4$) for a maximum of 32 parallel processes. Scaling-out to much larger configurations could impact the rollback characteristics.

8 DISCUSSIONS AND CONCLUSIONS

The overall efficiency and performance of a Parallel Discrete Event Simulation (PDES) is strongly influenced by synchronization overheads. In the case of optimistic PDES, the synchronization overheads manifest themselves in the form of rollbacks. Consequently, having a strong understanding of rollback behaviors is critical to enable efficient and performant optimistic PDES.

Accordingly, this paper proposed an experimental method for characterizing rollbacks. Our method has six key steps in which begins with instrumentation and collecting data on two key metrics, namely: inter-rollback cycles and rollback lengths. In the next step, statistical fitting is used to summarize the raw data at LP-level. We found that geometric distribution best fit both metrics. In the next step, we summarize the rollback characteristics for all LPs to obtain a simulation-level summary. At a simulation level, we found a normal distribution best summarized the means of the geometric distributions of the two metrics. In order to further analyze rollback patterns, we collectively analyzed 500 different simulation configurations on two different computational clusters. Our analysis showed that, immaterial of the hardware and simulation configuration, the overall rollback characteristics do follow statistical patterns.

Establishing sound, clear rollback characteristics enables investigators and practitioners of PDES to utilize this knowledge and our method to assess and optimize their PDES platforms. Comparing their rollbacks to nominal behaviors can facilitate identification of performance and efficiency bottlenecks. The data can also be used to develop, train, and calibrate adaptive algorithms for effectively managing optimism. The proposed methodology can also be used for selecting of suitable hardware configurations for a given application. For example, the experiments also highlight the advantage of investing in a higher-speed interconnect. The `owens` cluster with its 100 GPBS interconnect enables reducing both the inter-rollback cycle frequencies and the rollback lengths. This translates to faster, more efficient optimistic parallel simulation performance, as evidenced by the overall faster runtime of 37.13 seconds on `owens` (Broadwell running at 2.4 GHz) versus an overall 80.54 seconds on `RedHawk` (a newer Skylake, running at 2.6 GHz).

8.1 Future work

We are currently pursuing Generalized Sensitivity Analysis (GSA) to identify both platform and model parameters that have a strong influence on the rollback characteristics. It would also be beneficial to analyze the correlation between inter-rollback cycles and rollback lengths to determine if a relationship exists. Intuitively, LPs that frequently rollback, *i.e.,* have a smaller inter-rollback cycle, would have shorter rollback lengths. However, this intuition needs to be validated. We also plan to utilize our findings to implement

an adaptive optimism management algorithm to reduce rollbacks and thereby improve the overall efficiency and performance of parallel discrete event simulations.

REFERENCES

Carothers, C. D., and K. S. Perumalla. 2010. "On Deciding Between Conservative and Optimistic Approaches on Massively Parallel Platforms". In *Proceedings of the Winter Simulation Conference*, WSC '10, pp. 678–687. Baltimore, Maryland, IEEE.

Chetlur, M., and P. A. Wilsey. 2006, Dec. "Causality Information and Fossil Collection in Time Warp Simulations". In *Proceedings of the 2006 Winter Simulation Conference*, pp. 987–994.

Ferscha, A., and J. Luthi. 1995, April. "Estimating rollback overhead for optimism control in Time Warp". In *Proceedings of Simulation Symposium*, pp. 2–12.

Fujimoto, R. M. 1990. "Performance of Time Warp under Synthetic Workloads". In *Proceedings of SCS Multiconference on Distributed Simulation*, Number 1, pp. 23–28. SCS.

Jafer, S., Q. Liu, and G. Wainer. 2013. "Synchronization methods in parallel and distributed discrete-event simulation". *Simulation Modelling Practice and Theory* vol. 30, pp. 54–73.

Liu, Q., and G. Wainer. 2009, June. "A Performance Evaluation of the Lightweight Time Warp Protocol in Optimistic Parallel Simulation of DEVS-Based Environmental Models". In *2009 ACM/IEEE/SCS 23rd Workshop on Principles of Advanced and Distributed Simulation*, pp. 27–34.

Quaglia, F. 2015. "A low-overhead constant-time Lowest-Timestamp-First CPU scheduler for high-performance optimistic simulation platforms". *Simulation Modelling Practice and Theory* vol. 53, pp. 103–122.

Rao, D. M. 2016. "Efficient parallel simulation of spatially-explicit agent-based epidemiological models". *Journal of Parallel and Distributed Computing* vol. 93-94, pp. 102–119.

Rao, D. M. 2018. "Performance Comparison of Cross Memory Attach Capable MPI vs. Multithreaded Optimistic Parallel Simulations". In *Proceedings of the 2018 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM-PADS '18, pp. 37–48. New York, NY, USA, ACM.

Rao, D. M., and J. D. Higiro. 2019, March. "Managing Pending Events in Sequential and Parallel Simulations Using Three-tier Heap and Two-tier Ladder Queue". *ACM Trans. Model. Comput. Simul.* vol. 29 (2), pp. 9:1–9:28.

Wilsey, P. A. 2016. "Some Properties of Events Executed in Discrete-Event Simulation Models". In *Proceedings of the 2016 Annual ACM Conference on SIGSIM Principles of Advanced Discrete Simulation*, pp. 165–176. New York, NY, USA, ACM.

AUTHOR BIOGRAPHIES

DHANANJAI M. RAO is an Associate Professor in the Computer Science and Software Engineering (CSE) department at Miami University, Oxford, Ohio, USA. His ongoing research is in parallel and distributed computing with application to parallel simulation, epidemiology, bioinformatics, and phylodynamics. He research interests include modeling, object-oriented design, and web-based systems. He also has over 7 years of industry experience in related areas. His research products are available online at https://pc2lab.cec.miamiOH.edu/ and he can be reached at raodm@miamiOH.edu.