Multilevel Splitting for Reachability Analysis of Stochastic Hybrid Systems

Derek Riley
Vanderbilt University
Institute for Software and Integrated Systems
Nashville, TN
derek.riley@vanderbilt.edu

Xenofon Koutsoukos
Vanderbilt University
Institute for Software and Integrated Systems
Nashville, TN
xenofon.koutsoukos@vanderbilt.edu

Kasandra Riley
Yale University
New Haven, CT
kasandra.riley@yale.edu

Keywords
Monte Carlo, Stochastic Hybrid Systems, Variance Reduction

ABSTRACT
Biochemical research is increasingly using formal modeling, simulation, and analysis methods to improve the understanding of complex systems. Probabilistic analysis techniques such as Monte Carlo methods can be used to determine reachability or safety probabilities for large Stochastic Hybrid System (SHS) models, but systems containing influential rare events may require prohibitively large numbers of realizations to generate accurate estimates. In this work we present a multilevel splitting variance reduction method for SHS that improves the accuracy and efficiency of Monte Carlo methods for rare events. We apply the approach for reachability analysis of a SHS model of glycolysis, which is a biochemical energy conversion process found in virtually every living cell. We also present a method for selecting the variance reduction parameters as well as accuracy and efficiency analysis of our techniques.

1. INTRODUCTION
Modeling, simulation, and analysis tools are important for improving the understanding of complex biochemical systems as the cost and difficulty of testing physical systems increases. Probabilistic analysis techniques such as Monte Carlo methods can be used to determine reachability or safety probabilities for systems with inherent uncertainty such as Stochastic Hybrid System (SHS) models. Variance reduction techniques can be used to improve the accuracy of Monte Carlo methods for systems containing influential rare events. In this work we present a variance reduction technique for Monte Carlo methods called MultiLevel Splitting (MLS) for SHSs, and we present experimental results for our MLS algorithm for a SHS model of glycolysis.

Glycolysis is an energy conversion process found in virutally all living cells. It is a fundamental process in the life of a cell, and it has been studied extensively in many organisms [8]. The emergent behaviors of glycolysis are well understood, but the minute complexities of the chemical reactions involved are not. Therefore, modeling and analyzing glycolysis is important because it can provide insights into the minute complexities of the system, which may further the understanding of the biochemical process.

Accurate and efficient analysis of large biochemical systems is inherently difficult due to their complex, interconnected nature and the likelihood of the systems to have influential rare events. Monte Carlo methods are useful for determining probabilities of event occurrences. They can be used to determine reachability probabilities [12]; however, if events critical to the reachability conditions happen rarely, the accuracy and efficiency are diminished. Biochemical systems may have rare events that are difficult to accurately detect with Monte Carlo methods, so variance reduction techniques are often necessary to produce accurate, efficient results.

MLS and importance sampling are two common Monte Carlo variance reduction methods. Previous MLS algorithms have been applied to stochastic differential equations and have been shown to be effective for reducing the variance in the presence of rare events [11]. Importance sampling has been used for switching diffusions [10] and a SHS model in [1]. Importance sampling requires the adjustment of the probability laws that drive the system to increase the event probability, and the main challenge is determining appropriate ways to adjust the probability laws [7, 2]. Much research work has focused on improving importance sampling; however, each system must be analyzed individually to determine the appropriate change of measure for the dynamics to reduce the variance. If the incorrect change of measure is utilized, worse results than straightforward simulation may be achieved [6]. MLS methods are generally easier to tune than importance sampling methods to achieve greater efficiency and accuracy.

Several contributions to modeling methods and analysis tools for SHS are made in this work. We present a formulation of the reachability analysis problem for SHS using Monte Carlo methods. Further, the Monte Carlo methods are extended using MLS methods for variance reduction, and parameter selection methods are presented for the MLS technique. We also develop and present a parallel version of the analysis algorithms that further improves performance. A SHS model

of glycolysis is developed to demonstrate the approach, and experimental results for the MLS technique are presented and compared with the results of the Monte Carlo approach.

The rest of this paper is organized as follows: Section 2 presents the Stochastic Hybrid System model and simulation methods, Section 3 presents the reachability problem formulation, Section 4 presents the MLS technique for SHS, Section 5 describes the SHS glycolysis model, Section 6 presents the experimental results, and Section 7 concludes the work.

2. SHS MODELING AND SIMULATION

Modeling and simulating SHS are important because they can be used in conjunction with Monte Carlo methods to reveal the intricacies of large, complicated models. It is necessary to use formal modeling methods to be able to formally analyze the systems. Further, it is important to use the most accurate and efficient simulation methods available because they directly contribute to the overall accuracy and efficiency of Monte Carlo analysis methods.

SHS contain a set of discrete states $q \in Q$, invariants associated with the discrete states $X_q \subseteq \mathbb{R}^n$, and continuous dynamics associated with the discrete states $x(t) \in \mathbb{R}^n$. Discrete transitions between the states occur either because the continuous state x satisfies the transition guard $x(t) \in \partial X^q$ (guarded transition) or based on an exponential firing rate λ (probabilistic transition). A reset measure R is associated with any transition. The hybrid state at time t is $s(t) = (q(t), x(t))$.

To define the execution of the system, we denote $(\Omega, \mathcal{F}, P)$ the underlying probability space, and consider an $\mathbb{R}^p$-valued Wiener process $w(t)$ and a sequence of *stopping times* $\{t_0 = 0, t_1, t_2, \ldots\}$. Let the state at time t_i be $s(t_i) = (q(t_i), x(t_i))$ with $x(t_i) \in X^{q(t_i)}$. While the continuous state stays in $X^{q(t_i)}$, $x(t)$ evolves according to the stochastic differential equation (SDE)

$$dx = b(q, x)dt + \sigma(q, x)dw \tag{1}$$

where the discrete state $q(t) = q(t_i)$ remains constant and the solution of (1) is understood using the Itô stochastic integral [9]. A sample path of the stochastic process is denoted by $x_t(\omega), t > t_i, \omega \in \Omega$.

The next stopping time t_{i+1} represents the time when the system transitions to a new discrete state. The discrete transition occurs either because the continuous state x exits the invariant $X_{q(t_i)}$ of the discrete state $q(t_i)$ (guarded transition) or based on an exponential distribution with transition rate function λ (probabilistic transition). Therefore, t_{i+1} can be defined as the minimum between two other stopping times: (i) The first hitting time of the boundary $\partial X^{q(t_i)}$ defined as $t^*_{i+1} = \inf\{t \geq t_i, x(t) \in \partial X^{q(t_i)}\}$ and (ii) a stopping time τ_{i+1} described by an exponential distribution with a firing rate λ. At time t_{i+1} the system will transition to a new discrete state and the continuous state may jump according to the reset measure R. The evolution of the system is then governed by the SDE (1) with $q(t) = q(t_{i+1})$ until the next stopping time.

Figure 1 shows a generic SHS model with two states and two transitions (one probabilistic and one guarded). The

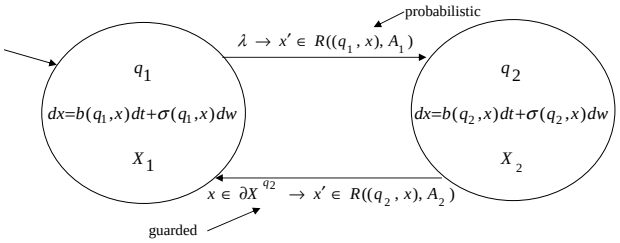

Figure 1: Stochastic hybrid system

continuous dynamics of each state are defined by the associated stochastic differential equations. The probabilistic transition fires at the firing rate λ, and the guarded transition fires when x hits the boundary $x \in \partial X^{q_2}$. The logical condition $x \in \partial X^{q_2}$ is often referred to as the guard of the transition. Upon firing of a transition, the state resets according to the map $R((q, x), A)$.

The following assumptions are imposed on the model. The functions $b(q, x)$ and $\sigma(q, x)$ are bounded and Lipschitz continuous in x for every q, and thus the SDE (1) has a unique solution for every q. The transition rate function λ is a bounded and measurable function which is assumed to be integrable for every $x_t(\omega)$. A full description of the formal model for SHS can be found in [14]. It has been shown that $s(t)$ is a strong Markov process [3].

To perform simulation of SHS, simulation of SDEs must be combined with simulation of the guarded and probabilistic discrete transitions in a way that captures the formal execution semantics as accurately as possible. We have developed a simulation method for SHS that performs numerical integration of SDEs with the Milstein Method (MM) and uses probabilistic discrete transition detection to improve accuracy. Our improved simulation methods have been shown to improve the accuracy of SHS simulation compared to traditional simulation methods [13].

Existing simulation techniques for SHS utilize fixed step methods, which can be inefficient, so we have also developed an adaptive timestepping version of our improved simulation methods, which can be used to improve the efficiency of the estimator in conjunction with the improved accuracy methods. Our improved algorithm is called $ATHMM$. Simulation trajectories of SHS can be combined and analyzed to determine reachability or safety probabilities for SHS using Monte Carlo methods.

3. PROBLEM FORMULATION

In this section we formulate the reachability and safety problems for SHS and we describe how Monte Carlo methods can be used to estimate the probabilities.

Consider a strong Markov process $s(t)$ representing the execution of the SHS. We define two disjoint subsets U and T for the unsafe and target sets respectively. The stopping times $\tau_U = inf\{t > 0 : s(t) \in U\}$ and $\tau_T = inf\{t > 0 : s(t) \in T\}$ occur when the trajectory hits either the unsafe or target set. For the reachability problem we want to determine the probability $P_R = \mathbb{P}[\tau_T < \tau_U]$, or that $s(t)$ will hit the target set T without first hitting the unsafe set U.

Monte Carlo methods can be used to estimate P_R by executing n independent simulations of the process $s(t)$ and comparing the outcomes. The number of trajectories that reach the set T before reaching the set U are divided by the total number of trajectories n to determine the reachability probability. This is given by $\widehat{P_R} = \frac{1}{n}\sum_{i=1}^{n} H_{R,i}$ where

$$H_R = \begin{cases} 1 & \text{if } \tau_T < \tau_U \\ 0 & \text{otherwise} \end{cases}$$

The formulation of the reachability problem can be modified to describe safety. For a safety problem, we are given a set of unsafe states and we want to compute the probability that the system execution from an arbitrary (safe) initial state will avoid the unsafe set. It is given by $P_S = \mathbb{P}[\tau_U < \tau_{max}]$.

Monte Carlo methods can also be used to estimate P_S. The number of runs that reach the set U before time τ_{max} are divided by the total number of runs n to determine the safety probability given by $\widehat{P_S} = \frac{1}{n}\sum_{i=1}^{n} H_{S,i}$ where

$$H_S = \begin{cases} 0 & \text{if } \tau_U < \tau_{max} \\ 1 & \text{otherwise} \end{cases}$$

The variance of the hitting probability for Monte Carlo methods for both reachability or safety is given by

$$Var[\widehat{P}] = \frac{\sum_{i=1}^{n}\left(H_i - \widehat{P}\right)^2}{n}$$

If n is very small, then the estimate $\widehat{P}$ will have a large variance and may not be reliable. The only way to reduce the variance of the estimator using traditional Monte Carlo methods is to increase n. Rare events in a system can strongly increase the variance of the reachability or safety results of Monte Carlo methods reducing the accuracy of the estimator. As an influential event becomes more rare, the error it can create increases dramatically, so variance reduction methods for rare events are necessary.

Multilevel Splitting MLS is a variance reduction method for rare events that extends Monte Carlo methods by splitting individual trajectories of the Monte Carlo estimator in the region of the a rare event. This regional splitting reduces the variance of the estimator by increasing the density of the trajectories in the region near the rare event, but care must be taken to choose when and how the trajectories are split to guarantee reasonable efficiency improvements. We define the region of the state space where the rare event exists A as a subset of the state space.

Regions of the state space may include events that occur rarely but have a large influence on the system. We define a region of the state space where an influential event is reached with a probability of less than two percent as a rare event region A. We define MLS splitting levels which create proper supersets of the set A: $A \subset A_1 \subset A_2 \subset \ldots A_g$. When a simulated trajectory crosses from a larger set A_k into a smaller set A_{k-1}, the trajectory is split into j new trajectories which evolve using unique Wiener processes. An example MLS scenario is shown in Figure 2.

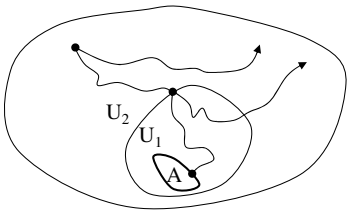

Figure 2: An example MLS scenario

Trajectories are assigned importance values v_i to represent the amount of influence the trajectory has on the approximation. Initially $v_i = 1/n$ where n is the original number of trajectories. When a trajectory is split, the importance value is divided evenly between the split forks of the trajectory, and the total number of trajectories n_m is incremented $n_m += j - 1$. Multiple splitting policies can be used to split trajectories differently at different levels according to the variance reduction desired.

The variance of the Monte Carlo estimator is reduced by increasing the number of samples n to n_m for a region of the state space around A. An artificial drift is created toward the region A by the reinforcement of trajectories through splitting. The variance reduction is unbiased despite the fact that the trajectories are not completely independent [4]. Further, the variance reduction is accomplished with a significantly improved efficiency compared to traditional Monte Carlo methods [11].

Rare events may occur in any region of the state space, but they are most often found as part of the unsafe set U, so we assume that the rare set is given by $A \subseteq U$. The safety probability for MLS is determined by $\widehat{P_S} = \sum_{i=1}^{n_m} H_{S,i} v_i$. Splitting in the region near A will increase the total number of trajectories n_m and change the influence of trajectories that are split v_i.

4. MLS FOR SHS

Monte Carlo methods using MLS can be useful for reducing the variance of estimators when in the presence of rare events, but care must be taken when applying the MLS method to systems with both continuous and discrete dynamics. Therefore, we extend the notion of MLS for SHS and propose solutions for the challenges of handling the continuous and discrete dynamics.

To extend MLS for SHS the standard notion of a Markov process is extended to include discrete modes $q(t)$: $s(t) = (x(t), q(t))$ where s(t) is the Markov process defined by the SHS model. We can redefine the stopping times $\tau_U = inf\{t > 0 : s(t) \in$ and $\tau_T = inf\{t > 0 : s(t) \in T\}$ to extend the notions of reachability and safety to the hybrid case.

The discrete transitions found in SHS can cause discontinuities that require special care in the presence of splitting

boundary crossings. Figure 3 shows an example SHS where the trajectory crosses a splitting and discrete boundary simultaneously. The trajectory starts at state $s_0 = (q_1, x_0)$, and evolves until it reaches the boundary for A_2 or the guards for a discrete transition are satisfied. In this example both the discrete transition is fired and the splitting level is crossed simultaneously, and the reset of the discrete transition updates the state of the trajectory to $s = (q_2, x_t)$.

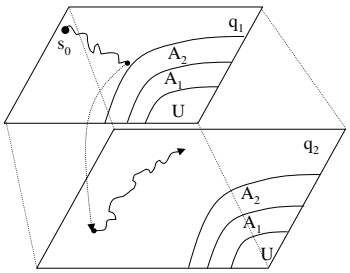

Figure 3: Example MLS problem in a hybrid state space

Because the new state is not in the splitting region A_2, splitting the trajectory before applying the reset may not necessarily reduce the variance, and could decrease the efficiency, so it should be avoided. This problem is further exacerbated if the splitting coefficient j is large. Therefore, care must be taken to ensure that discrete transitions are fired before testing splitting boundaries.

Another situation that must be handled is where the trajectory begins outside a splitting region, and the reset causes the trajectory to jump into a splitting region A_i. In this case, it is important to split the trajectory if it has not been previously split to maintain the greatest variance reduction. It is possible that the trajectory will jump into a region such as A_1 before it has entered the superset A_2. In this case, the splitting coefficient j must be chosen to ensure the variance is effectively reduced while the efficiency is not unnecessarily decreased. Our algorithm tests for these cases to ensure that they are handled appropriately.

Further, the use of accurate simulation methods including detection and handling of boundaries is important because MLS and Monte Carlo methods require highly accurate trajectories to ensure appropriate estimates. If low order methods or large step sizes are used, the Monte Carlo methods may not provide a reliable result.

We use an improved SHS simulation algorithm ($ATHMM$) presented in [13] implement the MLS technique for SHS. Boundary crossing conditions are tested after discrete transitions are fired to avoid the potentially inefficient situation where the discrete transition and splitting level conditions are both satisfied, but the reset moves the state to a region away from the rare event. $StartNewTraj$ keeps a list of the split trajectories and conditions when the trajectories were split and starts the most recently split trajectory in the **while** loop. If no split trajectories exist, the function exits the **while** loop and starts the next new trajectory in the **for** loop. The pseudocode for the algorithm is given below.

Algorithm 4.1: MLSforSHS(τ_{max})

for $j = 1; j < n; j++$
- $t = 0$
- $ResetInitialConditions()$
- $influence_j = \frac{1}{n}$
- **while** $X_t \notin U \wedge X_t \notin T \wedge t < \tau_{max}$
  - **do** $ATHMM(X_t)$
  - **if** $X_t \in A_k$
    - **then** $ForkTrajectory(j), split(influence_j)$
- **if** $X_t \in U$
  - **then** $unsafecount+ = influence_j, StartNewTraj$
- **if** $X_t \in T$
  - **then** $targetcount+ = influence_j, StartNewTraj$
- **if** $t > \tau_{max}$
  - **then** $StartNextSplitTrajectory$

return $(\frac{targetcount}{n}, \frac{unsafecount}{n})$

4.1 Choosing MLS Parameters

MLS has the potential to significantly reduce the variance and improve the efficiency of the estimator; however, set placement and splitting policies must be appropriately chosen to ensure the method performs well.

Choosing set placement If the A_k sets are chosen to be too close to A, not enough splitting will occur, and the variance reduction will be small (although the efficiency will be high). If the sets are too far away, too much splitting may occur, and the efficiency will be adversely affected without significant variance reduction. Splitting more trajectories at each set boundary has the potential to further reduce variance, but if the boundaries are placed improperly, the efficiency can be significantly decreased without a significant decrease in variance.

There is no universally optimal method for choosing the placement of the sets A_k for a multidimensional system; however, it has been determined that they should be chosen to cause splitting of the trajectory in regions that are most likely to lead to the rare set A for optimal efficiency [11]. Because this choice is crucial to the efficiency of MLS, we introduce a method for determining the best locations to place the boundaries.

First, it must be determined which states are most likely to transition to the set A. To do this we use Monte Carlo methods with initial conditions in the region near A to determine the safety probability of each location reaching the rare set A. Using more starting locations provides more accurate information but takes more computational time, so a trade off must be found.

We can use the safety probabilities in the region near A to determine where the sets are best placed by including the regions that are most likely to lead to the set A. The region of highest probability (typically $> .9$) of leading to A is chosen to be A_1. Monte Carlo simulations can then be used to determine the regional probabilities of transitioning to the new region A_1, and A_2 can be defined by the states

that have highest probability of leading to A_1. This method can be used to recursively define all sets A_k.

Choosing the number of levels The number of levels to use k must be determined to create an efficient implementation of the MLS algorithm. Using more levels has the potential to reduce the variance further, but also decreases the efficiency (sometimes significantly). Therefore, a trade off must be found to choose the number of levels to use. Typically a small number of levels (< 5) reduces the variance while maintaining sufficient efficiency.

Choosing the number of trajectories to split The splitting coefficient j is another important parameter that has implications on the performance and accuracy of MLS. If more trajectories are forked, the overall efficiency is decreased, but the variance may also be significantly decreased.

If the A_k sets and splitting policy are chosen well, the efficiency gains over traditional Monte Carlo simulation can be significant. Efficiency can also be further improved by terminating trajectories if they stray far from the rare event region A or unsafe set U. This ensures that trajectories that do not influence the outcome of the system do not reduce the efficiency of the approximation; however, care must be taken to ensure trajectories are not prematurely terminated if they may eventually affect the system outcome.

Quality of variance reduction MLS methods can reduce the variance of Monte Carlo methods, and thus increase the accuracy of the approximation by increasing the number of simulations in a certain region. The specific amount of variance reduction is dependent upon the system dynamics, the placement of the sets A_i, and the splitting policy, but MLS has the potential to reduce the variance by at least an order of magnitude [11]. The hybrid state space increases the difficulty of determining the optimal boundary placement and splitting coefficient j, but most non-optimal solutions still provide good variance reduction.

The efficiency of the estimator $\widehat{P}$ is dictated by the set placement, splitting policy, and dynamics of the model. We define the efficiency as $Eff\left[\widehat{P}\right] = \frac{1}{Var[\widehat{P}]C(\widehat{P})}$ where $C\left(\widehat{P}\right)$ is the expected execution time to compute the estimator [11]. The efficiency can be increased by decreasing the variance and/or decreasing the computation time.

Simulating more trajectories decreases the efficiency of the estimator by increasing the execution time C, so it is important to ensure that the trajectories are split in regions of interest, so the variance is ultimately reduced to improve efficiency. MLS decreases C quickly compared to traditional Monte Carlo methods by partially reusing previously computed paths and therefore reducing the cost C to achieve the same variance reduction Var for a limited region of the state space. Knowledge of the dynamics in the regions of interest is important to determine the most effective placement of the boundaries to ensure efficient and accurate results. For SHS, it is important that the most up-to-date state information is used to determine if switching boundaries are crossed to ensure that wasteful splitting does not increase C.

4.2 Parallelization

Monte Carlo methods benefit from large numbers of simulations by increasing in accuracy as the number of sample trajectories increases, but generating many trajectories can be comptuationally expensive. Parallelization can be used to generate significnatly more sample trajectories of a system using multiple cores or processors while not significantly increasing the overall execution time. Further, the number of required simulations may still be quite large even when using variance reduction methods, so parallel Monte Carlo methods using MLS methods are important to consider.

During simulation of the trajectories using our algorithm there are no dependencies between the individual Monte Carlo trajectories, so the algorithm can be parallelized simply by running multiple trajectories concurrently on multiple processors. After all trajectories are complete on all processors, the results can be compiled and reported. Because the collection of the results is the only communication overhead necessary, the speedup is nearly linear, and parallelization is quite effective and advantageous. This type of parallelization has been used previously with Monte Carlo methods [15], and care must be taken to ensure that the random number generators used to generate the Wiener processes do not introduce bias across the multiple processors or cores.

5. MODELING GLYCOLYSIS

We present the glycolysis model to demonstrate our analysis method for a large, realistic system. Glycolysis is a series of biochemical reactions that converts carbohydrates into various waste products and energy in a currency useful to cells. As it is a fundamental process to all living cells, it has been studied and modeled extensively in many organisms [8]. Although the individual steps of glycolysis have been thoroughly examined, the interaction of glycolytic enzymes, substrates, and products with the intracellular environment is not fully understood. Modeling and analyzing glycolysis can further our understanding of contextual cellular respiration.

Twenty-two chemical species have been identified that play an important role in glycolysis. The chemical species interact in a series of 37 interconnected chemical reactions as seen in Figure 4. Glucose (Glc) is added to the system, and glycogen ($Glyc$), ethanol ($EtOH$), ATP, and other minor chemicals are produced. The reaction rates for the system are developed in previous work [8].

We model the glycolysis system using the SHS modeling paradigm because it can capture the stochastic, discrete, and continuous dynamics found in the system. The model presented in [8] is a deterministic model, but the chemical reactions in the real system actually behave in a stochastic manner due to the uncertainty of molecular motion. The stochastic master equation can be used to express stochastic chemical species concentration fluctuations using stochastic differential equations [5]. We model individual chemical species concentration fluctuations using the stochastic master equation and discrete dynamics using a method developed in [14].

Glucose must be added to the system to continue production of the energy molecules, and when the concentration

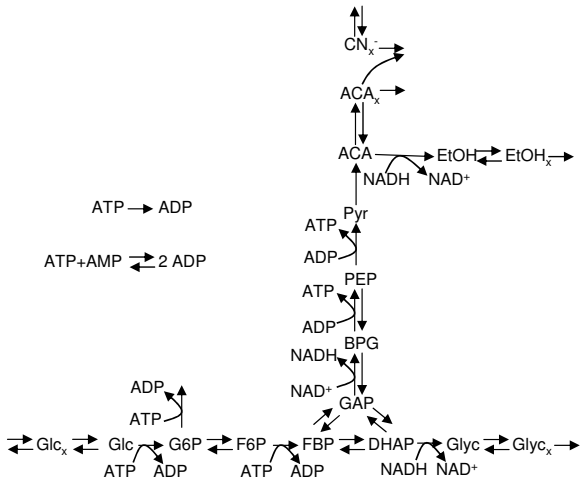

Figure 4: Network of glycolysis reactions

of glucose diminishes, the amount of energy molecules that the system can produce decreases. In many organisms, this reduction in energy output triggers mechanisms that encourage the introduction of more glucose (i.e, feeding). The feeding dynamics of the glycolysis model are incorporated using two discrete states: saturated and deficient shown in Figure 5. In the saturated state the glucose intake rate is less than in the deficient state. Switching between the states is regulated by the concentration of ATP (x_3). When the amount of ATP drops below a given level, the transition guard is satisfied, and the discrete state switches from the saturated to the deficient state. If the level of ATP climbs back above the given amount, then the guard is satisfied, and the state switches back to the saturated mode from the deficient mode.

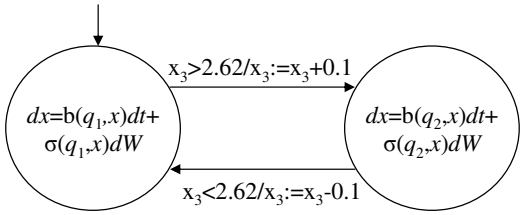

Figure 5: SHS model of glycolysis

6. EXPERIMENTAL RESULTS

Glycolysis is a process that turns glucose into energy in a cell, so the supply of glucose is crucial to the function of the system. If the amount of glucose drops below a certain level, a cascade of problems is set off ultimately leading to insufficient energy production and potentially cell death. We define an unsafe condition for the system when the glucose drops below a certain level, and we examine the safety probability of the system. The probability of the system reaching the unsafe state is small, so the rare event region $A \subseteq U$. Our experiments determined that the safety probability is approximately 2.2%.

Single simulation trajectories of a model can be useful to collect specific information about the system. It is very important to understand the general characteristics of a simulation to choose appropriate parameters for MLS or other rare event enhancement methods. In Figure 6, we show a single trajectory of the glycolysis model using MLS with $g = 2$ and $j = 2$. The unsafe region is marked at $Glc = 2.5$, and the two other levels are located at $Glc = 4$ and $Glc = 3$.

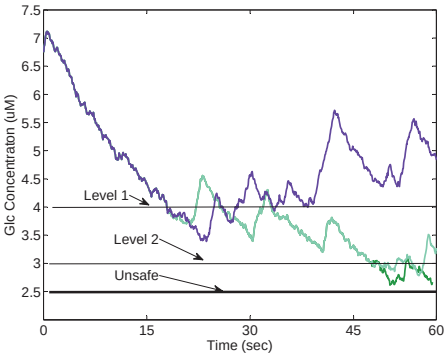

Figure 6: Trajectory of the glycolysis model with MLS

Monte Carlo efficiency and variance To evaluate the variance and efficiency of Monte Carlo methods, we tested the outcomes of the safety probability for the glycolysis system using various numbers of iterations n. The results of this analysis can be seen in Figure 7. It is shown in the figure that increasing the number of iterations n decreases the efficiency and the variance, but the efficiency decreases much faster than the variance. The 95% confidence intervals for the four trials are shown in Figure 8.

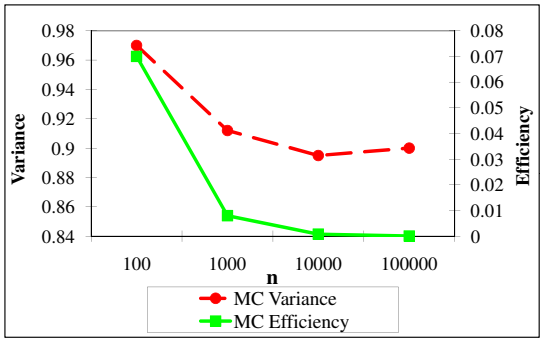

Figure 7: Monte Carlo results for the glycolysis model

Splitting boundary placement We first demonstrate the importance of choosing appropriate splitting levels by examining MLS methods with various placements of the splitting boundaries. The starting value for $Glc_x = 4.25$ usually causes the system to avoid the unsafe region, but it is reached rarely, so variance reduction methods can be used to improve the accuracy of the safety probability.

Splitting policy We use three splitting levels ($L1$, $L2$, and $L3$) as shown in Figure 9. We used three different placements of the splitting boundaries where placement A uses

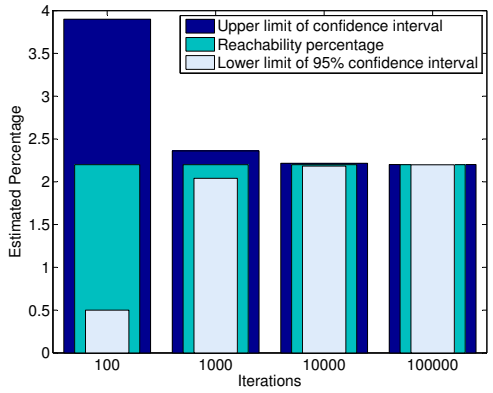

Figure 8: 95% Confidence Intervals

a wide spacing near the unsafe region, placement B uses a medium spacing, and placement C uses a tight spacing.

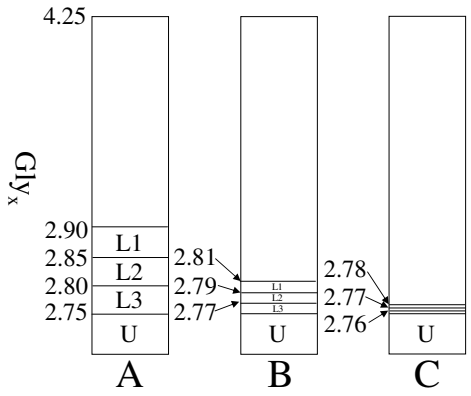

Figure 9: Boundary placement scenarios for the glycolysis model

We demonstrate the importance of choosing appropriate MLS splitting policies by examining three different example splitting policies. We consider three splitting levels for the glycolysis model ($L1$, $L2$, and $L3$) and we use three different splitting policies where policy I splits all trajectories two times at each level, policy II splits all trajectories four times at each level, and policy III splits the trajectory in two at $L1$, in four at $L2$, and six at $L3$. Examples of these splitting methods are shown in Figure 10.

Variance We compared the three boundary placement schemes and three splitting policies using 1000 initial MLS trajectories. In Figure 11 we show the variance results for all nine possible combinations of methods. The variance is significantly reduced when using MLS in comparison to traditional Monte Carlo methods; however, the choice of splitting policy and boundary placement is important. It can be seen that the variance is reduced the most by using the boundary placement scheme B with splitting policy II.

Efficiency In Figure 12 we show the efficiency results for all nine possible combinations of methods. It can be seen that the efficiency is largest when using the boundary placement

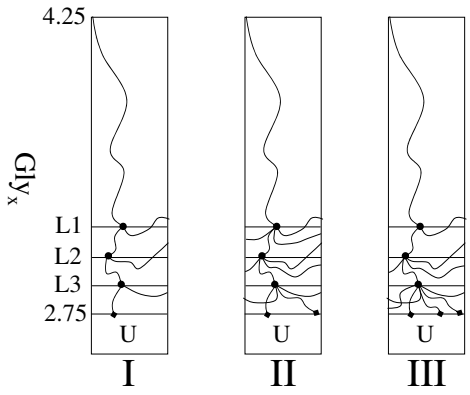

Figure 10: Splitting policies for the glycolysis model

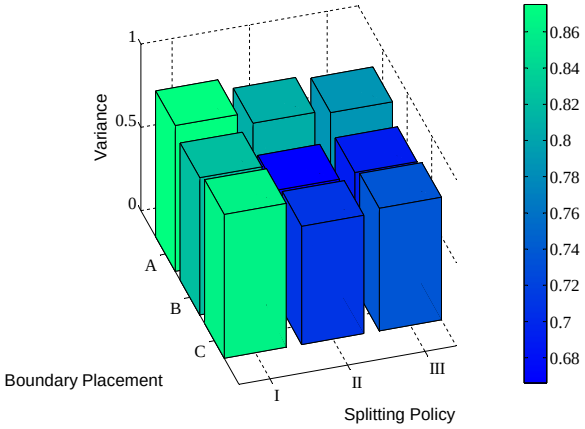

Figure 11: Comparison of variance for the MLS methods for the glycolysis model

scheme C with splitting policy I; however, this combination has a large variance. Compromises between efficiency and variance can be found depending on whether variance reduction or efficiency improvement are higher priority. Any of these methods work adequately to reduce the variance and improve the efficiency when compared to traditional Monte Carlo methods. Tuning to the methods holds potential for further variance reduction and efficiency gains.

Parallelization We performed experiments to test the parallel scalability of the MLS Monte Carlo reachability analysis algorithm. We executed identical versions of the algorithm on multiple processors using the same number of trials per processor regardless of the number of processors used. We used different pseudo-random number generator seeds to ensure the trials were as unbiased as possible. The results from the individual processors were collected after all processors completed to generate the final probability estimation. The results generated using more processors were more accurate with tighter confidence intervals, but since we are focused on the efficiency gains for parallel methods, we only present the efficiency results.

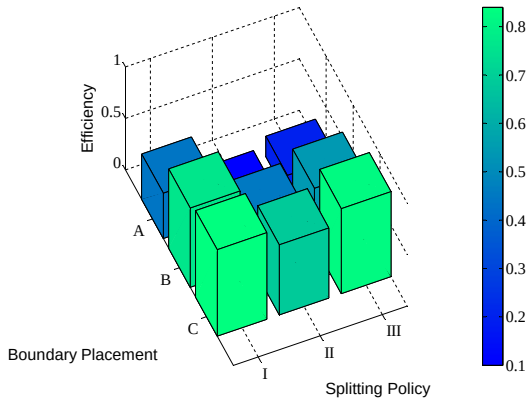

Figure 12: Comparison of efficiency for the MLS methods for the glycolysis model

As expected, the parallel speedup is nearly linear, as seen in Table 1. The slight super linear speedup can be explained by the variability of the noise of the system and the varying MLS trajectories taken. Noise causes the splitting policies to use different amounts of time because of the varying times the splitting occurs. These performance results indicate that our algorithm can be distributed between multiple processors to achieve a nearly linear speedup with little restriction on the total number of processors required.

Table 1: Parallel performance results for the glycolysis model

Processors	Time to Execute
16	8.6
8	8.5
4	8.2
2	8.3
1	8.5

The Advanced Computing Center for Research and Education at Vanderbilt University provides the parallel computing resources for our experiments[1]. The computers form a cluster of 348 JS20 IBM PowerPC nodes running at 2.2 GHz with 1.4 Gigabytes of RAM per machine. We use C++ as the implementation language, and we use the MPI standard for communication between processors.

7. CONCLUSIONS

Analysis of SHS is an important and challenging task that has the potential to expose insights into complex models that can be expensive or impossible to test physically. Rare events can be especially difficult to accurately estimate using traditional Monte Carlo methods, but the MLS technique we developed for SHS can be used to reduce the variance of the estimator. While choosing parameters for MLS is difficult, our methodology for choosing MLS parameters provides results that significantly outperform Monte Carlo methods.

[1]http://www.accre.vanderbilt.edu

Acknowledgements This research is partially supported by the National Science Foundation grants CNS-0347440 and CCF-0820088

.

References

[1] H Blom, G Bakker, and J Krystul. Probabilistic reachability analysis for large scale stochastic hybrid systems. *IEEE Conference on Decision and Control*, pages 3182–3189, 2007.

[2] J Bucklew. *Introduction to Rare Event Simulation*. New York: Springer-Verlag, 2004.

[3] M Bujorianu and J Lygeros. Theoretical foundations of general stochastic hybrid systems: Modeling and optimal control. In *IEEE Conf. on Dec. and Cont.*, 2004.

[4] M. Garvels. *The splitting method in rare event simulation*. PhD thesis, University of Twente, the Netherlands, 1997.

[5] D Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comp. Phys.*, 22:403–434, 1976.

[6] P Glasserman, P Heidelberger, P Shahabuddin, and T Zajic. Multilevel splitting for estimating rare event probabilities. *Operations Research*, 47(4):585–600, 1999.

[7] P Glynn and D Iglehart. Importance sampling for stochastic simulations. *Management Science*, 35:1367–1392, 1989.

[8] F Hynne, S Dano, and P Sorensen. Full-scale model of glycolysis in *Saccharomyces cerevisiae*. *Biophysical Chemistry*, 94:121–163, 2001.

[9] A Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

[10] J Krystul and H Blom. Sequential Monte Carlo simulation of rare event probability in stochastic hybrid systems. *16th IFAC World congress*, 2005.

[11] P. L'Ecuyer and B. Tuffin. Splitting for rare-event simulation. *Winter Simulation Conference*, pages 137–148, 2006.

[12] M Prandini and O Watkins. Probabilistic aircraft conflict detection. In *HYBRIDGE, IST-2001-32460*, 2005.

[13] D Riley, X Koutsoukos, and K Riley. Simulation of stochastic hybrid systems with switching and reflective boundaries. In *Winter Simulation Conference*, pages 804–812, 2008.

[14] D. Riley, X. Koutsoukos, and K. Riley. Modeling and analysis of the sugar cataract development process using stochastic hybrid systems. *IET: Systems Biology*, 3(3):137–154, 2009.

[15] M Troyer, B Ammon, and E Heeb. Parallel object oriented Monte Carlo simulations. *ISCOPE*, LNCS 1505:191–198, 1998.