Proceedings of the 2014 Winter Simulation Conference
A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, eds.

IMPROVING THE EFFICIENCY OF STOCHASTIC COMPOSITE SIMULATION MODELS VIA RESULT CACHING

Peter J. Haas

IBM Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099, USA

ABSTRACT

Stochastic composite simulation models, such as those created via the IBM Splash prototype platform, can be used to estimate performance measures for complex stochastic systems of systems. When, as in Splash, a composite model is made up of loosely coupled component models, we propose a method for improving the efficiency of composite-model simulations. To run *n* Monte Carlo replications of the composite model, we execute certain component models fewer than *n* times, caching and re-using results as needed. The number of component-model replications is chosen to maximize an asymptotic efficiency measure that balances computation costs and estimator precision. We initiate the study of result-caching schemes by giving an exact theoretical analysis for the most basic two-model scenario, as well as outlining some approaches for obtaining the parameter values needed for result caching.

1 INTRODUCTION

In this paper, we consider efficiency-improvement techniques for composite simulations, motivated by the IBM Splash prototype. Splash (Haas et al. 2012, Tan et al. 2012) is a platform for combining existing heterogeneous models and datasets to create composite simulation models of complex "systems of systems", thereby facilitating cross-disciplinary collaborative modeling and simulation as well as re-use of existing models. In Splash, the component models may reside on different platforms and are loosely coupled via data exchange; that is, models communicate by reading and writing data sets. When model and data contributors initially register their models and datasets with Splash, they provide metadata that describe key characteristics such as (i) the syntax and semantics of model inputs, model outputs, and datasets, (ii) model execution information, (iii) the provenance of models and datasets, and (iv) information useful for experimental design, such as recommended ranges and default values for model parameters. This metadata is expressed using the "Splash Actor Description Language" (SADL); see Figure 1 for examples of a SADL file for a model (the FR model discussed below) and for an input parameter file for the model. The SADL file for the output dataset from the FR model is similar to the file for the input, except that the output file contains time-series data, so there is an additional tag showing that the time series consists of values observed in continuous time, at a frequency of one per year:

```
<time type="continuous" observations="regular" field="yr" unit="year" value="1">
```

(In future work, many of the free-text descriptions in a SADL file will be replaced by a formal ontology.) Using the SADL metadata, Splash provides a drag-and-drop graphical user interface (GUI) for creating a composite model. Moreover, Splash automatically detects any data mismatches between models and, in the case of mismatches, pops up GUIs for the design of transformations from the outputs of one or more upstream "source" component models to a form suitable for input to a downstream "target" component model. In general, the SADL descriptions of models and datasets allow Splash to detect time-scale mismatches as well

```
<actor name="Financial Rate Model"
 actor_type="model"
 model_type="Simulation"
 simulation_type="Continuous-stochastic"
 owner="IBM"
 version="1.0"
 reference="See comments in Source Code">
<description>
 Simple lognormal random walk model of
 interest and inflation rates
 usage: FRmod -p parFile -seed x
</description>
<execution>
 <command>EXEC_DIR/FRmod</command>
</execution>
<arguments>
 <input name="Financial_Params"
  filepath="FR_params.txt"
  sadl="METADATA_DIR/FR_params.sadl"
 />
...
```

(a) SADL for the model

```
<actor name="Financial_Params"
 actor_type="data
 data_type="parameters"
 owner="IBM">
<description>
Input file for FRmod.
</description>
<data>
 <schema type="xsd" uri="FR_params.xsd"/>
 <format type="csv" delimiter=","/>
</data>
<!-- list of parameters -->
<attributes>
 <attribute name="id"
  description="initial discount rate"
  measurement_type="numerical"
  experiment_default_values=".01 .05 .09"
  experiment_factor="true"
  datatype="double"
 />
...
```

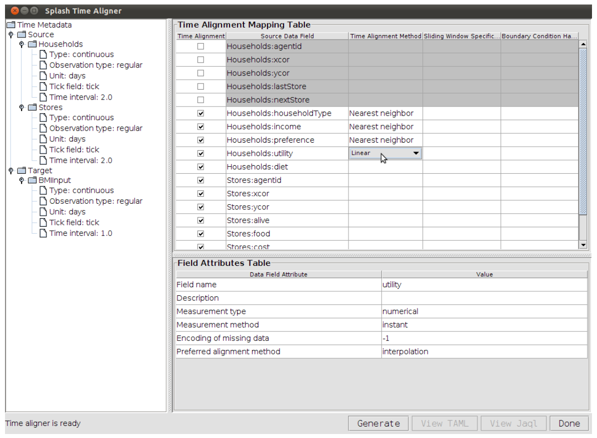
(b) SADL for the input parameter file

Figure 1: Snippets of SADL code for the FR model.

as "structural" mismatches, i.e., differences between the schemas of data output by upstream models and the input schema of a downstream model; see Figure 2. Graphical specifications of data transformations are automatically compiled into runtime code. Many simulation models—such as climate models and massive-scale agent-based models—consume or produce huge amounts of data, so Splash supports execution of data transformations in a MapReduce parallel processing environment (Dean and Ghemawat 2004). Splash also supports experimental design and simulation-based optimization for composite models; Figure 1(b) shows how a model parameter can be designated as a potential experimental factor having specified default levels. Thus Splash combines techniques from both simulation and data integration. Other analytical modules, such as data visualizers or R statistical routines, can easily be inserted into the dataflow. The loose-coupling approach and extensive use of metadata encourage comprehensive documentation and curation of models, help prevent users from combining conceptually incompatible models, and help avoid the need for massive re-coding or enforcement of a common platform, API, or communication protocol—an impossible task when dealing with experts in vastly different domains.

As an example, Figure 3 shows a screenshot of a simple composite model comprising two component models. The PHI (Predictive Health Institute) model is derived from an agent-based simulation model developed by Park et al. (2012) to explore the economic performance of a wellness program for managing heart disease and diabetes under alternative payment models: capitated (i.e., fixed) payments to the healthcare provider, outcome-based payments, or a combination of the two. The PHI model takes as input a time series of the annual healthcare inflation rate, general economic inflation rate, and discount rate (cost of capital). This time series is provided by the displayed financial-rate (FR) model. In the figure, the component labeled "SplashDataTransformer" transforms the quarterly time series output by the FR model to the yearly time series expected by the PHI model; the need for this transformation is automatically detected by Splash, based on SADL metadata. Splash uses the Kepler scientific workflow engine (Altintas et al. 2004) to provide the design environment and orchestrate the execution of the composite model.

Once a composite model has been created, it is repeatedly executed in order to explore system behavior. Such exploration often takes the form of estimating the expected values of various performance measures of interest, such as cost, profit, reliability, and so on, under one or more scenarios. Thus the basic task for a given scenario is to estimate $\theta = E[Y]$, where Y is a random variable that represents a noisy observation of a performance measure of interest. To estimate θ, the most direct approach is to run n multiple simulation replications in order to generate n independent and identically distributed (i.i.d.) observations $Y_0, Y_1, \ldots, Y_{n-1}$.



Splash Time Aligner

```
Time Metadata
  Source
    Households
      Type: continuous
      Observation type: regular
      Unit: days
      Tick field: tick
      Time interval: 2.0
    Stores
      Type: continuous
      Observation type: regular
      Unit: days
      Tick field: tick
      Time interval: 2.0
  Target
    BMIInput
      Type: continuous
      Observation type: regular
      Unit: days
      Tick field: tick
      Time interval: 1.0
```

Time Alignment Mapping Table

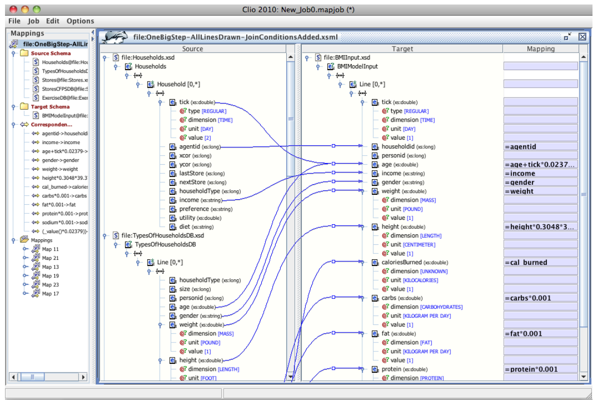
Time Alignment	Source Data Field	Time Alignment Method	Sliding Window Specific...	Boundary Condition Ha...
☐	Households:agentid			
☐	Households:xcor			
☐	Households:ycor			
☐	Households:lastStore			
☐	Households:nextStore			
✓	Households:householdType	Nearest neighbor		
✓	Households:income	Nearest neighbor		
✓	Households:preference	Nearest neighbor		
✓	Households:utility	Linear		
✓	Households:diet			
✓	Stores:agentid			
✓	Stores:xcor			
✓	Stores:ycor			
✓	Stores:alive			
✓	Stores:food			
✓	Stores:cost			

Field Attributes Table

Data Field Attribute	Value
Field name	utility
Description	
Measurement type	numerical
Measurement method	instant
Encoding of missing data	-1
Preferred alignment method	interpolation

Time aligner is ready

(a) GUI for time alignment



Clio 2010: New_Job0.mapjob (*)

```
Mappings
file:OneBigStep-AllLin
  Source Schema
    Households@file:Hou
    TypesOfHouseholdsD
    Stores@file:Stores.xs
    StoresCFPSDB@file:S
    ExerciseDB@file:Exer
  Target Schema
    BMIModelInput@file:
  Corresponden...
    agentid->householdi
    income->income
    age+tick*0.02379->
    gender->gender
    weight->weight
    height*0.3048*39.3
    cal_burned->calories
    carbs*0.001->carbs
    fat*0.001->fat
    protein*0.001->prot
    sodium*0.001->sodi
    (_value()*0.02379))-
  Mappings
    Map 11
    Map 21
    Map 13
    Map 19
    Map 23
    Map 17
```

file:OneBigStep-AllLinesDrawn-JoinConditionsAdded.xsml

Source

```
file:Households.xsd
  Households
    {...}
      Household [0,*]
        {...}
          tick (xs:double)
            type [REGULAR]
            dimension [TIME]
            unit [DAY]
            value [2]
          agentid (xs:long)
          xcor (xs:long)
          ycor (xs:long)
          lastStore (xs:long)
          nextStore (xs:long)
          householdType (xs:long)
          income (xs:string)
          preference (xs:string)
          utility (xs:double)
          diet (xs:string)
file:TypesOfHouseholdsDB.xsd
  TypesOfHouseholdsDB
    {...}
      Line [0,*]
        {...}
          householdType (xs:long)
          size (xs:long)
          personid (xs:long)
          age (xs:double)
          gender (xs:string)
          weight (xs:double)
            dimension [MASS]
            unit [POUND]
            value [1]
          height (xs:double)
            dimension [LENGTH]
            unit [FOOT]
```

Target	Mapping
file:BMIInput.xsd	
BMIModelInput	
{...}	
Line [0,*]	
{...}	
tick (xs:double)	
type [REGULAR]	
dimension [TIME]	
unit [DAY]	
value [1]	
householdid (xs:long)	=agentid
personid (xs:long)	
age (xs:double)	=age+tick*0.0237...
income (xs:string)	=income
gender (xs:string)	=gender
weight (xs:double)	=weight
dimension [MASS]	
unit [POUND]	
value [1]	
height (xs:double)	=height*0.3048*3...
dimension [LENGTH]	
unit [CENTIMETER]	
value [1]	
caloriesBurned (xs:double)	=cal_burned
dimension [UNKNOWN]	
unit [KILOCALORIES]	
value [1]	
carbs (xs:double)	=carbs*0.001
dimension [CARBOHYDRATES]	
unit [KILOGRAM PER DAY]	
value [1]	
fat (xs:double)	=fat*0.001
dimension [FAT]	
unit [KILOGRAM PER DAY]	
value [1]	
protein (xs:double)	=protein*0.001
dimension [PROTEIN]	

(b) GUI for structural transformations

Figure 2: Splash GUIs for designing data transformations.

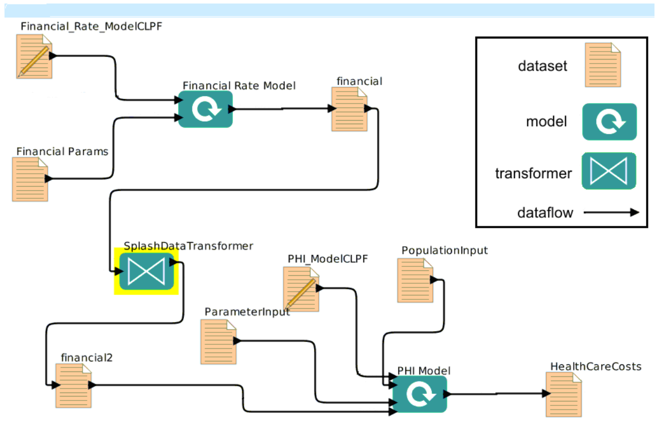


Figure 3: Design environment showing a simple composite healthcare model.

These observations are then used to compute a point estimate θ_n of θ, as well as confidence intervals and other statistics of interest, according to standard statistical methodology; the value of n is chosen based on precision requirements and constraints on simulation costs.

Often, the component models are large and complex, hence slow to run, so that composing these models compounds the problem of long execution times. Moreover, the inter-model data transformations can be very time consuming when the data is massive, as mentioned earlier. Thus techniques for increasing simulation efficiency are critically important for composite modeling. Many classical efficiency-improvement techniques can potentially be applied, such as common random numbers (CRN), importance sampling, control variates, and so on. For example, the current Splash implementation has facilities for coordinating pseudorandom number seeds between models that could be adapted to support CRN. This technique may not be applicable, however, if one or more of the component models does not allow a user to specify seed values, e.g., via a command-line argument; see Section III.C in Haas et al. (2012).

In this paper we propose an efficiency-improvement technique called *result caching (RC)* that is specific to composite modeling and can complement more traditional techniques such as CRN. The main idea can be summarized as follows. A naive approach to generating n simulation replications simply executes the entire composite model n times, which requires executing each component model n times. In contrast, the RC method re-uses the outputs of component models over multiple simulation replications. When applied judiciously, this technique inflates the variance of the estimator θ_n but reduces the execution costs so that there is an overall increase in simulation efficiency. We follow the classical approach of Hammersley and Handscomb (1964) and quantify efficiency as an inverse product of execution cost and variance; specifically, we apply results in Glynn and Whitt (1992) to rigorously define and justify an asymptotic efficiency metric appropriate for our setting. The following simple example motivates our approach.

Example 1 Consider a composite model M comprising two component models M_1 and M_2 in series. An execution of M proceeds by first executing M_1, which produces a random output Y_1 that is written to disk. Then M_2 is executed, taking Y_1 as input (after appropriate transformation) and generating a final output Y_2, where Y_2 is distributed according to a conditional cumulative distribution function $F_2(\cdot|Y_1)$. For example, M_1 might be a demand model that generates a sequence Y_1 of customer arrival times. The data in Y_1 might then fed into a queueing model M_2, which in turn produces an output Y_2, which might correspond to the average waiting time of the first 100 customers. To generate i.i.d. samples $Y_{2;1}, Y_{2;2}, \ldots, Y_{2;n}$ from the distribution of Y_2, one could execute the composite model n times, thereby executing M_1 and M_2 a total of n times each, for a total of $2n$ model executions. Suppose, however, that M_1 is in fact deterministic, so that the same output Y_1 is produced every time the model is executed. Rather than repeatedly executing M_1, it is clearly more efficient to execute M_1 only once and then use the resulting output Y_1 during each of the n replications as an input to M_2. If the cost of executing M_1 is large relative to the cost of executing M_2, then the cost savings can be significant.

In what follows, we initiate the study of RC schemes by giving an exact theoretical analysis for the most basic two-model scenario, which generalizes the above example in that both models are stochastic. Our goal is to illustrate the potential benefits of the method, while laying the groundwork for extension of the technique to more complex composite models. We assume throughout that, as in the current Splash prototype, there is no overlap in the execution of component models, so model M_1 completes before model M_2 starts.

2 RESULT CACHING FOR TWO MODELS IN SERIES

Consider two simulation models M_1 and M_2 in series, as in Example 1, but now assume that both models are stochastic. Using and expanding upon the notation of the example, denote by $\{Y_{i;n}\}_{n\geq 0}$ and $\{\tau_{i;n}\}_{n\geq 0}$ the random outputs and computation costs for M_i $(i = 1,2)$ over successive simulation replications. For each n, the quantity $\tau_{i;n}$ includes the cost of initializing the simulation of M_i for replication n, reading the inputs, if any, from disk, executing the simulation for one replication, transforming the outputs as needed, and writing the results to disk. We assume that all costs are a.s. finite.

Suppose that the goal of the simulation is to estimate $\theta = E[Y_{2;0}]$, the expected value of the output from M_2. (We assume throughout that θ is real-valued.) Consider a strategy under which, for n simulation replications of M_2, only $m_n = \lceil \alpha n \rceil$ replications of M_1 are executed, where $\alpha \in (0,1]$ is called the *replication fraction*. (Throughout, $\lfloor x \rfloor$ and $\lceil x \rceil$ denote the largest integer less than or equal to x and the smallest integer greater than or equal to x.) We write the (possibly transformed) output of M_1 to disk after each of the m_n simulation replications and then repeatedly cycle through these outputs in a fixed order to obtain inputs to M_2. Thus each M_1 output $Y_{1;l}$ is used in approximately n/m_n executions of M_2. Note that the deterministic cycling scheme produces a stratified sample of the outputs of M_1 and helps minimize estimator variance. (In Splash, this strategy can easily be implemented by modifying the Kepler "wrapper" code that invokes M_1.) Finally, the estimate of θ is computed as $\theta_n = (1/n)\sum_{l=0}^{n-1} Y_{2;l}$; note that the averaged variables are identically distributed but *not* independent, because $Y_{2;l}$ and $Y_{2;l'}$ are based on a common M_1 input whenever $l = l' \mod m_n$.

To formulate this setup precisely, we start with an i.i.d. sequence $\{Z_j\}_{j\geq 0}$ where $Z_j = (X_{1;j}, T_{1;j}, X_{2;j,0}, T_{2;j,0}, X_{2;j,1}, T_{2;j,1}, \ldots)$ for $j \geq 0$. Each j value corresponds to an execution of M_1, which creates an output $X_{1;j}$ at a cost of $T_{1;j}$, where this cost may depend on $X_{1;j}$. For example, an output consisting of a sequence of simulated customer delays may take longer to produce when there are many delays than

Table 1: Example of notation for basic random variables ($n = 9$, $\alpha = 0.5$, $m_n = 5$).

Rep.	0	1	2	3	4	5	6	7	8
	$Y_{1;0}$	$Y_{1;1}$	$Y_{1;2}$	$Y_{1;3}$	$Y_{1;4}$				
	$X_{1;0}$	$X_{1;1}$	$X_{1;2}$	$X_{1;3}$	$X_{1;4}$				
	$Y_{2;0}$	$Y_{2;1}$	$Y_{2;2}$	$Y_{2;3}$	$Y_{2;4}$	$Y_{2;5}$	$Y_{2;6}$	$Y_{2;7}$	$Y_{2;8}$
	$X_{2;0,0}$	$X_{2;1,0}$	$X_{2;2,0}$	$X_{2;3,0}$	$X_{2;4,0}$	$X_{2;0,1}$	$X_{2;1,1}$	$X_{2;2,1}$	$X_{2;3,1}$
	$\tau_{1;0}$	$\tau_{1;1}$	$\tau_{1;2}$	$\tau_{1;3}$	$\tau_{1;4}$				
	$T_{1;0}$	$T_{1;1}$	$T_{1;2}$	$T_{1;3}$	$T_{1;4}$				
	$\tau_{2;0}$	$\tau_{2;1}$	$\tau_{2;2}$	$\tau_{2;3}$	$\tau_{2;4}$	$\tau_{2;5}$	$\tau_{2;6}$	$\tau_{2;7}$	$\tau_{2;8}$
	$T_{2;0,0}$	$T_{2;1,0}$	$T_{2;2,0}$	$T_{2;3,0}$	$T_{2;4,0}$	$T_{2;0,1}$	$T_{2;1,1}$	$T_{2;2,1}$	$T_{2;3,1}$

when there are relatively few delays. The output $X_{1;j}$ is fed into M_2 multiple times, to create outputs $\{X_{2;j,i}\}_{i\geq 0}$ at respective costs $\{T_{2;j,i}\}_{i\geq 0}$, where these costs may depend both on the input from M_1 and the output from M_2. The quantities $X_{1;j}$ and $T_{1;j}$ have distribution functions $F_1^{(X)}(x)$ and $F_1^{(T)}(t \mid X_{1;j})$. Conditioned on $X_{1;j}$, the pairs $\{(X_{2;j,i}, T_{2;j,i})\}_{i\geq 0}$ form an i.i.d. sequence, with $X_{2;j,i}$, and $T_{2;j,i}$ having distribution functions $F_2^{(X)}(x \mid X_{1;j})$ and $F_2^{(T)}(t \mid X_{1;j}, X_{2;j,i})$. Finally, the time-ordered Y and τ random variables are formally defined by setting, for each $n \geq 0$, $(Y_{1;l}, \tau_{1;l}) = (X_{1;l}, T_{1;l})$ for $l = 0, 1, \ldots, m_n - 1$ and $(Y_{2;l}, \tau_{2;l}) = (X_{2;l \bmod m_n, \lfloor l/m_n \rfloor}, T_{2;l \bmod m_n, \lfloor l/m_n \rfloor})$ for $l = 0, 1, \ldots, n-1$.

See Table 1 for an illustration of these formulas—the table gives the X and T values corresponding to the Y and τ values for the given scenario. For example, $Y_{2;0}$ and $Y_{2;5}$ are statistically dependent because they share the same input $X_{1;0}$.

3 OPTIMIZING STATISTICAL EFFICIENCY

Given the foregoing setup, we choose α to maximize the asymptotic efficiency of the estimation procedure, as defined below. The asymptotic efficiency metric allows principled trade-offs between the precision of an estimate and the computational cost of producing the estimate. It was originally proposed informally by Hammersley and Handscomb (1964) and is rigorously treated in subsequent work (Fox and Glynn 1990, Glynn and Whitt 1992). The idea is to consider the (normalized) variance of the estimator for θ under a given computing budget. Although this variance is hard to compute exactly, it converges to a limit as the computing budget increases. The inverse of this limiting variance is our efficiency measure, which can then be maximized with respect to α. Thus an optimal choice of α approximately maximizes the statistical precision of our estimator of θ under a given budget or, conversely, approximately minimizes the cost of estimating θ to a specified precision.

Specifically, denote by C_n the cost of generating n outputs from M_2 under the foregoing strategy. That is,

$$C_n = \sum_{l=0}^{m_n-1} \tau_{1;l} + \sum_{l=0}^{n-1} \tau_{2;l} = \sum_{j=0}^{m_n-1} \left[T_{1;j} + \sum_{i=0}^{r_{n,j}-1} T_{2;j,i} \right].$$

Here $r_{n,j}$ is the number of times that the output of the jth execution of M_1 is input into M_2. Formally,

$$r_{n,j} = \lfloor n/m_n \rfloor + I_{n,j} \tag{1}$$

where

$$I_{n,j} = \begin{cases} 1 & \text{if } m_n \lfloor n/m_n \rfloor + j < n; \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

(We suppress the dependence on α in the above notation.) Under a budget c, the number of M_2 outputs that can be generated is

$$N(c) = \sup\{n \geq 0 : C_n \leq c\}, \tag{3}$$

resulting in the estimate $U(c) = \theta_{N(c)}$.

Theorem 1 below gives a strong law of large numbers (SLLN) and central limit theorem (CLT) for $U(c)$ as the budget c becomes large, thereby showing that $U(c)$ is strongly consistent for θ and yielding an asymptotic normal approximation to the distribution of $U(c)$ in the large-budget regime; the proof is in Appendix A. Denote by $c_1 = E[T_{1;0}]$ and $c_2 = E[T_{2;0,0}]$ the expected computation costs for a single run of M_1 and M_2. Also denote by $V_1 = \mathrm{Var}[X_{2;0,0}]$ the variance of the output from M_2 based on a single replication and by $V_2 = \mathrm{Cov}[X_{2;0,0}, X_{2;0,1}]$ the covariance of two outputs from M_2 when they share a common input from M_1. Finally, set $r_\alpha = \lfloor 1/\alpha \rfloor$ and

$$g(\alpha) = (\alpha c_1 + c_2)h(\alpha) \tag{4}$$

for $\alpha \in (0,1]$, where

$$h(\alpha) = V_1 + [2r_\alpha - \alpha r_\alpha(r_\alpha + 1)]V_2. \tag{5}$$

Denote by $\Rightarrow$ convergence in distribution and by $N(0,1)$ a standard (mean 0, variance 1) normal random variable.

Theorem 1 Suppose that $E[T_{1;0} + T_{2;0,0} + X_{2;0,0}^2] < \infty$ and $h(\alpha) > 0$. Then

$$U(c) \to \theta \tag{6}$$

almost surely and

$$c^{1/2}[U(c) - \theta] \Rightarrow \sqrt{g(\alpha)}N(0,1) \tag{7}$$

as $c \to \infty$.

Thus, for large values of c, the estimator $U(c)$ is approximately normally distributed with mean θ and variance $g(\alpha)/c$. We define our efficiency metric as $1/g(\alpha)$. As per the Hammersley-Handscomb framework, the first factor of g in (4) quantifies the amortized (expected) cost per execution of the composite model M, which is increasing in α, and the second factor quantifies the variance of an output from M, which is decreasing in α (see below). We assume that $V_2 \geq 0$, which is usually true in practice.

The optimal value of $\alpha \in (0,1]$ can therefore be found by maximizing the efficiency metric or, equivalently, by minimizing $g(\alpha)$ and hence minimizing the asymptotic variance. To gain insight into the nature of the optimal solution, approximate r_α by $1/\alpha$ and write $g(\alpha) \approx \tilde{g}(\alpha) \overset{\text{def}}{=} (\alpha c_1 + c_2)\big(V_1 + (\alpha^{-1} - 1)V_2\big)$. It is then easy to verify that $\tilde{g}(\alpha)$ is maximized by setting $\alpha = \alpha^*$, where

$$\alpha^* = \left(\frac{c_2/c_1}{(V_1/V_2) - 1}\right)^{1/2}.$$

(Truncate α^* at $1/n$ or 1 as needed to ensure a feasible solution. Note that $V_1/V_2 \geq 1$ by the Cauchy-Schwarz inequality.) If c_1 is large relative to c_2, so that M_1 is relatively expensive to execute, then α^* is small, and it is optimal to execute M_1 a relatively small number of times. If M_2 is relatively insensitive to the input from M_1, so that most of the variability arises from randomness within M_2, then $V_2 \ll V_1$, and again we simulate M_1 only a small number of times. In the extreme case where $V_2 = 0$, we simulate M_1 only once, thereby recapturing the result of Example 1. If, on the other hand, M_2 is a deterministic function of its inputs and M_1 is stochastic, then $V_1 = V_2$ and it is optimal to run n replications of M_1. In this case, M_2 is merely a transformer of the output of M_1. Finally, denote by ρ the ratio of the asymptotic efficiencies of the RC and naive methods, so that

$$\rho = \frac{1/g(\alpha^*)}{1/g(1)} \approx \frac{\tilde{g}(1)}{\tilde{g}(\alpha^*)} = \frac{(\gamma_1 + 1)\gamma_2}{(\gamma_2 - 1)\big(1 + \sqrt{\gamma_1/(\gamma_2 - 1)}\big)^2},$$

where $\gamma_1 = c_1/c_2$ and $\gamma_2 = V_1/V_2$. Observe that, by choosing sufficiently large values of γ_1 and γ_2, we can make ρ arbitrarily large. Thus extremely large efficiency improvements are possible in principle.

4 POINT AND INTERVAL ESTIMATION

Assume that a value of α has been selected and we are given either a budget c under which to compute a $100(1-\delta)\%$ confidence interval or a precision target, e.g., to estimate θ to within $\pm 100\varepsilon\%$ with probability $100(1-\delta)\%$ for some $\varepsilon \in (0,1)$. We cannot use Theorem 1 directly to obtain point and interval estimates. The problem is that the RC method needs to know the total number n of replications in advance in order to determine m_n but, in the fixed-budget scenario, $n = N(c)$ is unknown a priori; a similar problem occurs in the fixed-precision scenario. One solution is to employ an adaptive variant of RC that uses cached output $\lfloor \alpha l \rfloor$ from M_1 for the lth simulation replication of the composite model. This variant does not stratify the sample outputs from M_1 very well, however.

Instead, we consider an approach that first computes an estimate n of the number of required composite model replications, and then applies RC as described previously. In more detail, Remark 1 in Appendix A shows that $\theta_n \xrightarrow{\text{a.s.}} \theta$, so that, for any fixed value of n as above, θ_n is a strongly consistent point estimator for θ. Also observe that $E[Y_{2;i}] = \theta$ for each i, so that θ_n is unbiased. Next, set

$$h_n(\alpha) = \frac{1}{n} \sum_{j=0}^{m_n-1} \left(\sum_{i=0}^{r_{n,j}-1} (X_{2;j,i} - \theta_n) \right)^2 \tag{8}$$

Remark 3 in Appendix A shows that

$$n^{1/2}[\theta_n - \theta] \Rightarrow \sqrt{h(\alpha)} N(0,1) \tag{9}$$

as $n \to \infty$, and Remark 2 shows that $h_n(\alpha) \xrightarrow{\text{a.s.}} h(\alpha)$, so that a standard argument leads to an asymptotic $100(1-\delta)\%$ confidence interval for θ having endpoints $\theta_n \pm z_\delta \big(h_n(\alpha)/n\big)^{1/2}$, where z_δ is the $\big((1+\delta)/2\big)$-quantile of the standard normal distribution. A rough estimate of n is given by $n \approx c/(\alpha c_1 + c_2)$ in the fixed-budget case and $n \approx h_{n_0}(\alpha)\big(z_\delta/(\varepsilon\theta_{n_0})\big)^2$ in the fixed-precision case. The latter formula is obtained by first setting the half-width of the above confidence interval equal to $\varepsilon\theta$ and solving for n, and then approximating θ and $h_n(\alpha)$ using a small pilot run comprising n_0 composite-model replications (using RC). The above procedure can be refined in many ways. For example, in the fixed-budget scenario, one might set $n \approx \beta c/(\alpha c_1 + c_2)$, where $\beta \in (0,1)$ is chosen to ensure that, with a specified probability, the budget is not exceeded.

5 SETTING THE REPLICATION FRACTION

To use the RC technique, the user needs to compute α^*, which requires knowledge of the statistics $\mathscr{S} = (c_1, c_2, V_1, V_2)$. However, costs, variances, and covariances are not observed until the simulation is run, leading to a seemingly intractable situation. Two observations point toward a solution of this problem.

1. There is some tolerance for error in estimating $\mathscr{S}$. Errors in the estimates of $\mathscr{S}$ might result in slightly suboptimal simulation performance, but estimation of θ will still be correct. Indeed, the analysis of Section 4 shows that the simulation results themselves are asymptotically valid for any value of α.
2. A composite modeling system such as Splash is oriented toward re-use of models, and important performance or statistical characteristics of a model can be stored as part of the model's metadata (SADL files in the case of Splash).

Thus, for example, the cost of executing pilot runs to estimate the statistics in $\mathscr{S}$ can be amortized over multiple model executions. Moreover, as the model is used in production runs, its behavior can be observed and used to continually refine the statistics in $\mathscr{S}$, resulting in continually improving performance.

The problem is roughly analogous to the query-optimization problem in relational database systems. In this latter setting, the user specifies the desired query result, and the query-optimizer component of the database system determines a low cost execution plan for the query, based on statistics stored in the system catalog. This problem has been studied for decades—see, for example, Cormode et al. (2012) and Haas et al. (2009)—and many of the ideas developed there can potentially be adapted to the current setting. In the remainder of this section, we outline some possible approaches toward estimating the statistics needed for the RC method.

Recall the FR-PHI healthcare model that was mentioned in Section 1, and first consider the problem of estimating c_1, the expected execution time of the FR model. One version of this model is a simple multidimensional lognormal random walk, which takes as input three drift parameters and a small covariance matrix comprising six "volatility" parameters. Thus the challenge is to determine the function $c_1(x)$, where x represents the vector of input parameters. To this end, we can view c_1 as an expected simulation response and use observations of model execution times (from production runs or exploratory pilot runs or both) to build a metamodel. This metamodel can be stored along with the model (e.g., in a SADL file) and used to optimize simulation runs as needed. A broad range of metamodeling techniques are available, ranging from simple polynomial regression models to sophisticated Gaussian process models—see, e.g., Kleijnen (2007) and Salemi et al. (2013)—allowing trade-offs between metamodel cost, complexity, and accuracy. (The computational burden associated with computing statistics for the RC method should not outweigh the cost savings from using RC.)

To estimate the remaining statistics c_2, V_1, and V_2, which represent expected outputs of the PHI model (or functions of such outputs), we again view these quantities as functions of their input values, where in this case the input value corresponds to the random output of the FR model. The complications here are that the output of the FR model is (i) random and (ii) a time series. Thus, for example, we have $V_1 = \mathrm{Var}[X_{2;0,0}] = \int_{x\in\mathscr{X}} V_1(x)\,dF_1(x)$, where $V_1(x) = \mathrm{Var}[X_{2;0,0} \mid X_{1;0} = x]$, the set $\mathscr{X}$ is the space of possible realizations of the time series $X_{1;0}$ output by the FR model, and F_1 is the distribution of $X_{1;0}$. One practical approach to obtaining estimates of V_1 from prior observations of model executions is as follows. First, use dimensionality-reduction techniques to obtain a compact (low dimensional) representation of the output time series. Many such methods have been proposed in the literature, ranging from a representation via piecewise-linear approximations (Chen et al. 2007) to more complex schemes such as recursive feature elimination on common principal components (Yang and Shahabi 2007). Next, construct a multi-dimensional histogram of the reduced values, comprising buckets $B_1, B_2, \ldots, B_n$. Section 3.5 in Cormode et al. (2012) contains a fairly recent survey of methods for maintaining multidimensional histograms; several of these methods appear to work well, especially when the dimensionality is not too high (hence the previous dimensionality-reduction step). Associated with each bucket B_i is the bucket frequency f_i, as well as v_i, the value of $V_1(x_i)$ for a representative point x_i whose reduced representation lies in B_i. We can then approximate V_1 as $\widehat{V}_1 = \sum_{i=1}^{n} f_i v_i$. In some settings, V_1 might also depend on a real-valued vector z of additional input parameters that might include inputs to M_1 or M_2 or both. In this case we can use a metamodeling approach, as for c_1, to obtain a function $\widehat{V}_1(z)$. The remaining statistics c_2 and V_2 are handled analogously.

6 CONCLUSION

Problems with simulation efficiency are compounded for composite models. Our new result-caching technique is designed to exploit the composite structure of such models to boost simulation efficiency. The method can complement more traditional efficiency-improvement techniques and can potentially yield large improvements in simulation performance. The techniques are applicable beyond Splash to any composite simulation environment where component models communicate by reading and writing datasets, for example, the Nimrod system of Bethwaite et al. (2010).

This paper represents a first step in developing the result-caching technique; much work remains. We are currently evaluating the RC ideas experimentally and developing both exact and approximate

RC schemes for stochastic composite models in which the data flows can be represented as a directed acyclic graph (DAG). In these more complex settings, we expect that the replication fractions often will not be amenable to closed-form solution, and so we are investigating exact and approximate numerical techniques. In this regard, we observe that the correlation structure between model outputs is similar to that which arises in the study of sampling-based estimation techniques for database "join" queries (Haas et al. 1996), so prior analytical methodology may be applicable. An important aspect of this problem is the identification of a good set of statistics to maintain on composite models in order to allow estimation of the replication fractions, as discussed in Section 5. We are currently trying to identify the best techniques to use in practice. Moreover, taking a cue from the query-optimization literature, it may also be possible to develop sequential execution procedures that start in "naive mode" and gradually switch to more optimal execution plans as the component models' behavior is observed and the statistics $\mathscr{S}$ refined. Another key challenge is handling mutually dependent models in which data flows in both directions. We are currently investigating some "fixed point" and "perturbation" approaches to this problem, which must be solved prior to development of efficiency-improvement techniques in this broader setting. Overall, the goal is to move toward broader practical application of composite modeling based on loose coupling of component models, thereby facilitating interdisciplinary modeling and simulation of highly complex systems.

A PROOF OF THEOREM 1

Assume without loss of generality that $\theta = 0$, since in the general case we can consider the shifted random variables $X_{2;j,i} - \theta$. Set $W_{n,j} = \sum_{i=0}^{r_{n,j}-1} X_{2;j,i}$ for $n \geq 0$ and $j \in [0..m_n)$, where $r_{n,j}$ is defined as in (1). (We denote by $[a..b)$ the set of integers $\{a, a+1, \ldots, b-1\}$ and by $[a..b]$ the set $\{a, a+1, \ldots, b\}$.) According to these definitions, $\theta_n = (1/n)\sum_{j=0}^{m_n-1} W_{n,j}$, where the random variables $\{W_{n,j}\}_{j\in[0..m_n)}$ are mutually independent for each n with common mean equal to 0. Set $V_{n,j} = \mathrm{Var}[W_{n,j}] = E[W_{n,j}^2] = r_{n,j}V_1 + r_{n,j}(r_{n,j}-1)V_2$ for $n \geq 0$ and $j \in [0..m_n)$. Observe that $\sup_{n,j} r_{n,j} \leq r_\alpha + 1$, so that $\sup_{n,j} V_{n,j} \leq (r_\alpha+1)V_1 + r_\alpha(r_\alpha+1)V_2$. If $E[X_{2;0,0}^2] < \infty$, then $V_1, V_2 < \infty$ and the $V_{n,j}$'s are not only finite, but are uniformly bounded as well.

Next, define $S_{n,0} = \sigma_{n,0}^2 = 0$, and set $S_{n,k} = \sum_{j=0}^{k-1} W_{n,j}$ and $\sigma_{n,k}^2 = \mathrm{Var}[S_{n,k}] = \sum_{j=0}^{k-1} V_{n,j}$ for $k \in [1..m_n]$. Also set $\sigma_n^2 = \sigma_{n,m_n}^2$ and $S_n = S_{n,m_n}$, and define the process $\{\mathscr{U}_n(t)\}_{0\leq t\leq 1}$ for $n \geq 0$ by setting $\mathscr{U}_n(t) = \sum_{k=0}^{m_n-1}(S_{n,k}/\sigma_n)I(\sigma_{n,k}^2/\sigma_n^2 \leq t < \sigma_{n,k+1}^2/\sigma_n^2)$ for $t \in [0,1)$ and $\mathscr{U}_n(1) = S_n/\sigma_n$, where I is an indicator function. The process $\mathscr{U}_n$ has piecewise-constant sample paths.

We establish the CLT in (7) by establishing both the SLLN

$$C(t)/t \xrightarrow{\text{a.s.}} \alpha c_1 + c_2, \tag{10}$$

where $C(t) = C_{\lfloor t \rfloor}$, and the functional central limit theorem (FCLT)

$$n^{-1/2}\sigma_n \mathscr{U}_n \Rightarrow \sqrt{h(\alpha)}B, \tag{11}$$

where the stochastic process B is a standard Brownian motion on $[0,1]$ and $\Rightarrow$ denotes weak convergence on the space $D[0,1]$; see Billingsley (1999). The desired result then follows from Theorem 1 in Glynn and Whitt (1992). (Note that the function $\sigma_n\mathscr{U}_n/n$ corresponds to the function $\mathscr{Y}_\varepsilon$ in Glynn and Whitt (1992); see Remark 4 in Glynn and Whitt (1992).)

Lemma 1 Suppose that $c_1 + c_2 < \infty$. Then (10) holds.

Proof. It suffices to show that $C_n/n \xrightarrow{\text{a.s.}} \alpha c_1 + c_2$. Suppose that $1/\alpha$ is not an integer, so that $r_\alpha = \lfloor 1/\alpha \rfloor < 1/\alpha$. Observe that $\lim_{n\to\infty} m_n/n = \lim_{n\to\infty}\lceil \alpha n \rceil/n = \alpha$, so that $\lim_{n\to\infty} n/m_n \to 1/\alpha$ and hence $\lfloor n/m_n \rfloor = r_\alpha$ for sufficiently large n. Equations (1) and (2) then imply that $r_{n,j} = r_\alpha + I_{n,j}$, where

$$I_{n,j} = \begin{cases} 1 & \text{if } j < n - m_n r_\alpha; \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

Thus, for sufficiently large n, we have $\sum_{i=0}^{r_{n,j}-1} T_{2;j,i} = U_j + I_{n,j}T_{2;j,r_\alpha}$, where $U_j = \sum_{i=0}^{r_\alpha - 1} T_{2;j,i}$, so that $C_n = \sum_{j=0}^{m_n-1} T_{1;j} + \sum_{j=0}^{m_n-1} U_j + \sum_{j=0}^{n-r_\alpha m_n-1} T_{2;j,r_\alpha}$. Observe that the random variables $\{U_j\}_{j\ge 0}$ are i.i.d., and their common mean $r_\alpha c_2$ is finite under the assumption of the lemma. Similarly, the random variables $\{T_{1;j}\}_{j\ge 0}$ are i.i.d. with finite mean c_1, and we can apply the SLLN for i.i.d. random variables to obtain

$$\frac{C_n}{n} = \frac{m_n}{n}\frac{1}{m_n}\sum_{j=0}^{m_n-1} T_{1;j} + \frac{m_n}{n}\frac{1}{m_n}\sum_{j=0}^{m_n-1} U_j + \frac{n-r_\alpha m_n}{n}\frac{1}{n-r_\alpha m_n}\sum_{j=0}^{n-r_\alpha m_n-1} T_{2;j,r_\alpha}$$
$$\xrightarrow{\text{a.s.}} \alpha c_1 + \alpha r_\alpha c_2 + (1-\alpha r_\alpha)c_2 = \alpha c_1 + c_2.$$

Now suppose that $1/\alpha$ is an integer, so that $r_\alpha = \lfloor 1/\alpha \rfloor = 1/\alpha$. Because $\lfloor n/m_n \rfloor = \lfloor n/\lceil \alpha n \rceil \rfloor \le \lfloor n/(\alpha n) \rfloor = 1/\alpha$ for all n and $\lim_{n\to\infty} n/m_n \to 1/\alpha$ as before, it follows that, for sufficiently large n, we have $\lfloor n/m_n \rfloor = r_\alpha$ if r_α divides n and $\lfloor n/m_n \rfloor = r_\alpha - 1$ otherwise. In this case we can write $\sum_{i=0}^{r_{n,j}-1} T_{2;j,i} = \widehat{U}_j + \widehat{I}_{n,j}T_{2;j,r_\alpha}$, where $\widehat{U}_j = \sum_{i=0}^{r_\alpha-2} T_{2;j,i}$ and

$$\widehat{I}_{n,j} = \begin{cases} I\big(j < n-(r_\alpha - 1)m_n\big) & \text{if } r_\alpha \text{ does not divide } n; \\ 1 & \text{otherwise.} \end{cases} \tag{13}$$

It follows that $C_n = \sum_{j=0}^{m_n-1} T_{1;j} + \sum_{j=0}^{m_n-1} \widehat{U}_j + \sum_{j=0}^{n-(r_\alpha-1)m_n-1} T_{2;j,r_\alpha}$, so that $C_n/n \xrightarrow{\text{a.s.}} \alpha c_1 + \alpha(r_\alpha - 1)c_2 + [1-\alpha(r_\alpha-1)]c_2 = \alpha c_1 + c_2$. □

Remark 1 An almost identical argument shows that $\theta_n/n \xrightarrow{\text{a.s.}} 0$ as $n\to\infty$.

The next step is to establish the FCLT in (11). To this end, we prove a couple of preliminary results, starting with an asymptotic analysis of $\sum_{j=0}^{m_n-1} W_{n,j}^2$ and its expected value σ_n^2.

Lemma 2 If $E[X_{2;0,0}^2] < \infty$, then $(1/m_n)\sum_{j=0}^{m_n-1} W_{n,j}^2 \xrightarrow{\text{a.s.}} h(\alpha)/\alpha$ and $\sigma_n^2/m_n \to h(\alpha)/\alpha$ as $n\to\infty$.

Proof. First suppose that $1/\alpha$ is not an integer. Arguing as in Lemma 1, we can write $W_{n,j} = G_j + I_{n,j}X_{2;j,r_\alpha}$ for sufficiently large n, where $G_j = \sum_{i=0}^{r_\alpha-1} X_{2;j,i}$ and $I_{n,j}$ is defined in (12). Thus

$$\frac{1}{m_n}\sum_{j=0}^{m_n-1} W_{n,j}^2 = \frac{1}{m_n}\sum_{j=0}^{m_n-1} G_j^2 + \frac{2}{m_n}\sum_{j=0}^{m_n-1} I_{n,j}H_j + \frac{1}{m_n}\sum_{j=0}^{m_n-1} I_{n,j}X_{2,j,r_\alpha}^2, \tag{14}$$

where $H_j = \sum_{i=0}^{r_\alpha-1} X_{2;j,i}X_{2;j,r_\alpha}$ and we have used the fact that $I_{n,j}^2 = I_{n,j}$. The finite-moment assumption of the lemma implies that $V_1, V_2 < \infty$, so that $\mathrm{E}[G_j^2] = r_\alpha V_1 + r_\alpha(r_\alpha-1)V_2 < \infty$. Moreover, the G_j's are i.i.d., so that the first term on the right side of (14) converges to $r_\alpha V_1 + r_\alpha(r_\alpha - 1)V_2$ almost surely as $n\to\infty$. For the second term, observe that

$$\frac{1}{m_n}\sum_{j=0}^{m_n-1} I_{n,j}H_j = \frac{1}{m_n}\sum_{j=0}^{n-r_\alpha m_n-1} H_j = \frac{n-r_\alpha m_n}{m_n}\left(\frac{1}{n-r_\alpha m_n}\sum_{j=0}^{n-r_\alpha m_n-1} H_j\right).$$

The H_j's are i.i.d. with $\mathrm{E}[H_j] = r_\alpha V_2 < \infty$, so that the second term on the right side of (14) converges to $2(\alpha^{-1} - r_\alpha)r_\alpha V_2$ almost surely. A similar argument shows that the third term converges to $(\alpha^{-1} - r_\alpha)V_1$ almost surely, so that $m_n^{-1}\sum_{j=0}^{m_n-1} W_{n,j}^2 \xrightarrow{\text{a.s.}} r_\alpha V_1 + r_\alpha(r_\alpha-1)V_2 + (\alpha^{-1} - r_\alpha)(V_1 + 2r_\alpha V_2) = h(\alpha)/\alpha$.

Now suppose that $1/\alpha$ is an integer. Again arguing as in Lemma 1, we can write $W_{n,j} = \widehat{G}_j + \widehat{I}_{n,j}X_{2;j,r_\alpha-1}$ for sufficiently large n, where $\widehat{G}_j = \sum_{i=0}^{r_\alpha-2} X_{2;j,i}$ and $\widehat{I}_{n,j}$ is defined in (13). Set $\widehat{H}_j = \sum_{i=0}^{r_\alpha-2} X_{2;j,i}X_{2;j,r_\alpha-1}$

and observe that, for n sufficiently large,

$$\begin{aligned}\frac{1}{m_n}\sum_{j=0}^{m_n-1} W_{n,j}^2 &= \frac{1}{m_n}\sum_{j=0}^{m_n-1}\widehat{G}_j^2 + \frac{2}{m_n}\sum_{j=0}^{m_n-1}\widehat{I}_{n,j}\widehat{H}_j + \frac{1}{m_n}\sum_{j=0}^{m_n-1}\widehat{I}_{n,j}X_{2,j,r_\alpha-1}^2 \\ &\xrightarrow{\text{a.s.}} [(r_\alpha-1)V_1 + (r_\alpha-1)(r_\alpha-2)V_2] + [(r_\alpha-1)V_2] + V_1 \\ &= r_\alpha V_1 + r_\alpha(r_\alpha-1)V_2.\end{aligned}$$

Using the fact that $r_\alpha = 1/\alpha$, it is easy to verify that the rightmost expression is equal to $h(\alpha)/\alpha$, establishing the first assertion of the lemma.

To prove the second assertion, set $Q_j = (r_\alpha+1)^2\sum_{i=0}^{r_\alpha} X_{2,j,i}^2$ for $j \ge 0$. With probability 1, we have for any $n \ge 0$ that $W_{n,j}^2 = \big(\sum_{i=0}^{r_{n,j}-1} X_{2;j,i}\big)^2 \le \big(r_{n,j}\max_{0\le i<r_{n,j}}|X_{2;j,i}|\big)^2 \le (r_\alpha+1)^2\max_{0\le i\le r_\alpha} X_{2,j,i}^2 \le Q_j$. Set $\Gamma_n = (1/m_n)\sum_{j=0}^{m_n-1} W_{n,j}^2$ and $\Gamma_n^* = (1/m_n)\sum_{j=0}^{m_n-1} Q_j$ for $n \ge 0$. Observe that the Q_j's are i.i.d. with $E[Q_j] = (r_\alpha+1)^3 V_1 < \infty$, so that $\lim_{n\to\infty}\Gamma_n^* = \mathrm{E}[Q_1]$ almost surely. Moreover, $\mathrm{E}[\Gamma_n^*] \equiv \mathrm{E}[Q_1]$ for $n \ge 0$, so that $\lim_{n\to\infty}\mathrm{E}[\Gamma_n^*] = \mathrm{E}[\lim_{n\to\infty}\Gamma_n^*]$, which implies (Chung 2001, Th. 4.5.4) that the sequence $\{\Gamma_n^*\}_{n\ge 0}$ is uniformly integrable, that is, $\lim_{x\to\infty}\mathrm{E}[\Gamma_n^* I(\Gamma_n^* > x)] = 0$. Because $0 \le \Gamma_n \le \Gamma_n^*$ for all n, it follows that $\{\Gamma_n\}_{n\ge 0}$ is also uniformly integrable. Since $\Gamma_n \xrightarrow{\text{a.s.}} h(\alpha)/\alpha$ by the first assertion of the lemma, Theorem 4.5.4 in Chung (2001) implies that $\mathrm{E}[\Gamma_n] \to h(\alpha)/\alpha$, that is, $\sigma_n^2/m_n \to h(\alpha)/\alpha$. □

Remark 2 It follows immediately from Lemma 2 that $(m_n/n)\big[(1/m_n)\sum_{j=0}^{m_n-1} W_{n,j}^2\big] \xrightarrow{\text{a.s.}} \alpha\big[h(\alpha)/\alpha\big] = h(\alpha)$. Since $\theta_n \xrightarrow{\text{a.s.}} 0$ as $n \to \infty$ by Remark 1, it follows that $h_n(\alpha) \xrightarrow{\text{a.s.}} h(\alpha)$, where $h_n(\alpha)$ is defined as in (8). Thus $h_n(\alpha)$ is a strongly consistent estimator of $h(\alpha)$.

We next establish a "Lindeberg condition" that will be sufficient for the FCLT to hold.

Lemma 3 If $E[X_{2;0,0}^2] < \infty$, then

$$\lim_{n\to\infty}\frac{1}{\sigma_n^2}\sum_{j=0}^{m_n-1}\mathrm{E}\left[W_{n,j}^2 I(|W_{n,j}| \ge \varepsilon\sigma_n)\right] = 0 \tag{15}$$

for any $\varepsilon > 0$.

Proof. Fix $\varepsilon > 0$ and define $\{Q_j\}_{j\ge 0}$ as above.Recall that the Q_j's are i.i.d. with finite mean, so that, for a given value of n and $j \in [0..m_n)$,

$$\mathrm{E}\left[W_{n,j}^2 I(|W_{n,j}| \ge \varepsilon\sigma_n)\right] = \mathrm{E}\left[W_{n,j}^2 I(W_{n,j}^2 \ge \varepsilon^2\sigma_n^2)\right] \le \mathrm{E}\left[Q_j I(Q_j \ge \varepsilon^2\sigma_n^2)\right] = \mathrm{E}\left[Q_1 I(Q_1 \ge \varepsilon^2\sigma_n^2)\right].$$

Because $h(\alpha) > 0$ by assumption, Lemma 2 implies that $\lim_{n\to\infty}\sigma_n^2 = \infty$, so that $I(Q_1 \ge \varepsilon^2\sigma_n^2) \xrightarrow{\text{a.s.}} 0$. Thus, because Q_1 is nonnegative with $\mathrm{E}[Q_1] < \infty$, we have $\lim_{n\to\infty}\mathrm{E}\left[Q_1 I(Q_1 \ge \varepsilon^2\sigma_n^2)\right] = 0$ by the dominated convergence theorem (Chung 2001, p. 44). Again using the fact that $h(\alpha) > 0$, so that $1/h(\alpha) < \infty$, we have, using Lemma 2, $\lim_{n\to\infty}(1/\sigma_n^2)\sum_{j=0}^{m_n-1}\mathrm{E}\left[W_{n,j}^2 I(|W_{n,j}| \ge \varepsilon\sigma_n)\right] \le \lim_{n\to\infty}(m_n/\sigma_n^2)\mathrm{E}\left[Q_1 I(Q_1 \ge \varepsilon^2\sigma_n^2)\right] = \big(\alpha/h(\alpha)\big)\cdot 0 = 0$, proving the result. □

Proof of Theorem 1. In light of (10) and of Theorem 1 in Glynn and Whitt (1992), it suffices to establish the FCLT in (11). Because $\{W_{n,j}\}_{n\ge 0, j\in[0..m_n)}$ is a triangular array of row-wise independent random variables such that $\sigma_n^2 < \infty$ for $n \ge 0$ and (15) holds for all $\varepsilon > 0$, an appeal to the FCLT for such arrays (Billingsley 1999, p.147) shows that $\mathscr{U}_n \Rightarrow B$. Using Lemma 2, we have $n^{-1/2}\sigma_n\mathscr{U}_n = (\sigma_n^2/m_n)^{1/2}(m_n/n)^{1/2}\mathscr{U}_n \Rightarrow \big(h(\alpha)/\alpha\big)^{1/2}\cdot\alpha^{1/2}\cdot B = \sqrt{h(\alpha)}B$, proving (11). □

Remark 3 For $x = \{x(t)\}_{t\in[0,1]}$, denote by π_1 the projection that maps x to $x(1)$. By applying π_1 to (11) via the continuous mapping theorem (Billingsley 1999, p. 134), we obtain (9).

REFERENCES

Altintas, I., C. Berkley, E. Jaeger, M. Jones, B. Ludascher, and S. Mock. 2004. "Kepler: An Extensible System for Design and Execution of Scientific Workflows". In *Proc. 16th Intl. Conf. Scientific Statist. Database Mgmt. (SSDBM)*, 423–424.

Bethwaite, B., D. Abramson, F. Bohnert, S. Garic, C. Enticott, and T. Peachey. 2010. "Mixing Grids and Clouds: High-Throughput Science Using the Nimrod Tool Family". In *Cloud Computing: Principles, Systems and Applications*, 219–237. London: Springer.

Billingsley, P. 1999. *Convergence of Probability Measures*. second ed. New York: Wiley.

Chen, Q., L. Chen, X. Lian, Y. Liu, and J. X. Yu. 2007. "Indexable PLA for Efficient Similarity Search". In *Proc. 33rd Intl. Conf. Very Large Data Bases (VLDB)*, 435–446.

Chung, K. L. 2001. *A Course in Probability Theory*. third ed. San Diego: Academic Press.

Cormode, G., M. N. Garofalakis, P. J. Haas, and C. Jermaine. 2012. "Synopses for Massive Data: Samples, Histograms, Wavelets, Sketches". *Foundations and Trends in Databases* 4 (1–3): 1–294.

Dean, J., and S. Ghemawat. 2004. "MapReduce: Simplified Data Processing on Large Clusters". In *Proc. 6th Symp. Operating Sys. Design Implementation (OSDI)*, 137–150.

Fox, B. L., and P. W. Glynn. 1990. "Discrete-Time Conversion for Simulating Finite-Horizon Markov Processes". *SIAM J. Appl. Math* 50 (5): 1457–1473.

Glynn, P. W., and W. Whitt. 1992. "The Asymptotic Efficiency of Simulation Estimators". *Oper. Res.* 40 (3): 505–520.

Haas, P. J., N. C. Barberis, P. Phoungphol, I. Terrizzano, W.-C. Tan, P. G. Selinger, and P. P.Maglio. 2012. "Splash: Simulation Optimization in Complex Systems of Systems". In *Proc. 50th Allerton Conf. on Commun. Control Comput.*, 414–425.

Haas, P. J., I. F. Ilyas, G. M. Lohman, and V. Markl. 2009. "Discovering and Exploiting Statistical Properties for Query Optimization in Relational Databases: A Survey". *Statist. Anal. Data Mining* 1 (4): 223–250.

Haas, P. J., J. F. Naughton, S. Seshadri, and A. N. Swami. 1996. "Selectivity and Cost Estimation for Joins Based on Random Sampling". *J. Comput. Syst. Sci.* 52 (3): 550–569.

Hammersley, J. M., and D. C. Handscomb. 1964. *Monte Carlo Methods*. Chapman and Hall.

Kleijnen, J. P. C. 2007. *Design and Analysis of Simulation Experiments*. Springer.

Park, H., T. Clear, W. B. Rouse, R. C. Basole, M. L. Braunstein, K. L. Brigham, and L. Cunningham. 2012. "Multi-Level Simulations of Health Delivery Systems: A Prospective Tool for Policy, Strategy, Planning and Management". *Service Science* 4 (3): 253–268.

Salemi, P., J. Staum, and B. L. Nelson. 2013. "Generalized Integrated Brownian Fields for Simulation Metamodeling". In *Proceedings of the 2013 Winter Simulation Conference*, edited by R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, 543–554. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Tan, W. C., P. J. Haas, R. L. Mak, C. A. Kieliszewski, P. G. Selinger, P. P. Maglio, S. Glissmann, M. Cefkin, and Y. Li. 2012. "Splash: A Platform for Analysis and Simulation of Health". In *Proc. ACM Intl. Health Informatics Symp. (IHI)*, 543–552.

Yang, K., and C. Shahabi. 2007. "An Efficient K Nearest Neighbor Search for Multivariate Time Series". *Inf. Comput.* 205 (1): 65–98.

AUTHOR BIOGRAPHIES

PETER J. HAAS is a Research Staff Member at the IBM Almaden Research Center and a Consulting Professor of Management Science and Engineering at Stanford University. His email address is phaas@us.ibm.com and his web page is http://researcher.watson.ibm.com/researcher/view.php?person=us-phaas.