Efficient Monte Carlo based Incremental Statistical Timing Analysis

Vineeth Veetil, Dennis Sylvester, David Blaauw
EECS Department, University of Michigan, Ann Arbor, MI - 48109
tvvin,dennis,blaauw@eecs.umich.edu

Abstract

Modeling and accuracy difficulties exist with traditional SSTA analysis and optimization methods. In this paper we describe methods to improve the efficiency of Monte Carlo-based statistical static timing analysis. We propose a Stratification + Hybrid Quasi Monte Carlo (SH-QMC) approach to reduce the number of samples required for Monte Carlo based SSTA. Our simulations on benchmark circuits up to 90K gates show that the proposed method requires 23.8X fewer samples on average to achieve comparable accuracy in timing estimation as a random sampling approach. Results on benchmark circuits also show that when SH-QMC is performed with multiple parallel threads on a quad core processor, the approach is faster than traditional SSTA with comparable accuracy. SH-QMC scales better than traditional SSTA with circuit size. We also propose an incremental approach to recompute a percentile delay metric after ECO. The results show that on average only 1.2% and 0.8% of original samples need to be evaluated for exact recomputation of the 95th percentile and 99th percentile delays, after sample size reduction using SH-QMC.

1. Introduction

Process parameter variations have taken on increasing importance in nanometer-scale CMOS. Rather than using simple corner models that capture worst-case behavior at the device level (and lead to large guard bands), modern CAD tools are moving towards a more probabilistic view of circuit timing behavior. In replacing corner models, there are two primary approaches that incorporate process parameter uncertainty in timing analysis. The first is to perform statistical static timing analysis (SSTA) by modeling gate delay as a function of process parameters and propagating these distribution functions to compute the distribution of circuit delay [1,2]. We refer to these approaches as traditional SSTA. In traditional SSTA it has proven challenging to efficiently model skewness in the arrival time distribution which results from non-linearity of the gate delays and the maximum function. Also, a number of modeling issues are still in early stages of development, such as combined analysis of large interconnect structures driven by non-linear drivers, coupling events, and modeling of transparent latches. While some progress has been made in addressing these issues [1-4], it is expected that a fully mature traditional SSTA tool capable of performing timing sign-off may not be widely available in the near future. The second approach is Monte Carlo based SSTA, which involves selection of samples of the process variation space to obtain statistical distributions of circuit timing behavior. The application of Monte Carlo (MC) for statistical timing was discussed in [5], where it was shown that Monte Carlo based SSTA is accurate even in scenarios with high dimensionality and non-standard distributions in the process variation space, where traditional SSTA has difficulties. However, there are two main difficulties with this approach. First, the standard MC approach of random selection of samples in the process variation space requires too many samples for sufficient accuracy, resulting in high runtime cost. Second, there is no work to show the applicability of MC based SSTA for incremental statistical timing analysis. In this work, we address both concerns.

Standard techniques to reduce the sample size for MC based approaches exist in statistics literature and are called variance reduction techniques. The application of these techniques for parametric yield estimation has been analyzed in literature [6-9]. In [6], a Latin Hypercube approach for parametric yield estimation is proposed. In [7], mixture importance sampling for statistical SRAM design and analysis is proposed. The approach in [8] uses the control variates technique in conjunction with importance sampling for timing yield estimation. However, while several approaches are reviewed, no results are presented. In [9], the authors propose to use Quasi Monte Carlo Analysis for yield estimation. However, it is not clear how this approach can be extended to systems with large number of dimensions (variables) which is often the case with process variation. Also, these approaches do not focus on the specific problem of using MC as an alternative to traditional SSTA for timing analysis. Variance reduction relies heavily on information about the system [11], hence it is important to adapt it specifically to timing analysis. To the best of our knowledge this work is the first to directly study variance reduction aimed at improving the efficiency of MC-based SSTA with an accurate process variation model considering intra-die variation with spatial correlation [2] and uncorrelated random variation.

ECO(Engineering Change Order) and synthesis tools require incremental timing analysis techniques for fast recomputation of circuit delay with small changes in the design. To meet time to market, designers need tools capable of performing fast incremental timing analysis, and such tools need to incorporate process variations. While incremental techniques for traditional SSTA exist in literature [1], the lack of such techniques has been a major drawback for MC based approaches to SSTA. We address the specific problem of recomputing a percentile delay metric after incremental circuit sizing. To the best of our knowledge, this work is the first to address incremental timing analysis in MC based SSTA.

This paper has two main contributions. First, we introduce a new approach for variance reduction in MC based SSTA, Stratified Sampling + Hybrid Quasi Monte Carlo (SH-QMC). In SH-QMC, we propose to use circuit timing criticality information for sample size reduction. We use information about the criticality of variables to the circuit delay to order them. For the most critical variables, we then employ techniques that achieve high accuracy with few samples. For the less critical variables, we use techniques that are effective for problems of higher dimensionality. The proposed approach is implemented and tested on benchmark circuits with sizes up to 90,000 gates, and compared to a random sampling approach for selecting samples in the process variation space. In general SH-QMC shows large speedups relative to the random sampling approach: 23.8X on average and upto 44X on the benchmarks studied. Our results also show that the number of samples required does not increase with the number of gates in the circuit. Additionally, when SH-QMC is implemented with multiple threads on a quad core processor, it is faster than traditional SSTA for comparable accuracy. We also observe that the performance of SH-QMC scales better than traditional SSTA with circuit size.

Second, we propose a technique to recompute a percentile delay metric after incremental circuit sizing, where individual gates are resized. In this technique, we use information local to the resized gate to prune out most of the samples, leaving only a few samples to be reevaluated. Our results for the incremental computation of the 95th percentile and 99th percentile delays of benchmark circuits show that on average only 1.2% and 0.8% of original samples need to be evalu-

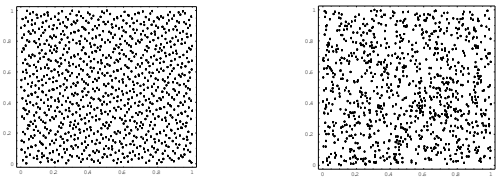

Figure 1. Quasi random and pseudo random sequences.

ated for exact recomputation, even after sample size reduction using SH-QMC.

This paper is organized as follows. Section 2 discusses the applicability of existing variance reduction approaches in statistics to the statistical timing analysis domain. Section 3 presents our work on variance reduction for MC based SSTA. In Section 4, we propose our approach to incremental statistical timing analysis. We present detailed results in Section 5 and conclude with Section 6.

2. Variance Reduction Approaches for Statistical Timing

MC based statistical timing involves selecting samples of the process variation space to obtain statistical distributions of circuit delay. This is mapped to the standard mathematical problem of MC, which is to estimate the integral of a function, using samples in its domain. There are standard techniques for variance reduction of MC, which include Quasi Monte Carlo techniques, Latin Hypercube sampling, stratified sampling, importance sampling and control variates. In this section, we briefly discuss their applicability to the statistical timing analysis framework.

2.1 Quasi Monte Carlo

The standard MC method addresses the problem of approximating the integral of a function $f(x)$ over the s-dimensional hypercube $C^s = [0, 1)^s$, where x represents a point in an s-dimensional space. The MC estimate of the integral $\bar{f}$ is given by the arithmetic mean of f_i, which are values of the function $f(x)$ evaluated at n samples distributed throughout the hypercube. The Koksma-Hlawka inequality relates the error bound of a method to numerically estimate an integral using a sequence of samples, to a mathematical measure of uniformity for the distribution of the points, called "discrepancy" [10]. This inequality suggests that we should use a sequence with the smallest possible discrepancy to evaluate the function in order to achieve the smallest possible error bound. Such sequences constructed to reduce discrepancy are called Low Discrepancy Sequences (LDSs). Quasi monte carlo techniques are characterised by their use of LDSs to generate samples. LDSs are deterministic sequences, in other words there is no randomness in their generation. Intuitively, these sequences are well dispersed through the domain of the function, minimizing any gaps and/or clustering of points. Figure 1 illustrates that quasi random sequences generate samples with lower discrepancy compared to pseudo random sequences (sequences with properties similar to "truly" random sequences). Sobol[12], Faure and Niederreiter[9] are LDSs that have been studied extensively. In this work, we consider Sobol sequences, which are known to be simple to construct and more resistant to the *pattern dependency* issue (mentioned below), compared to the other sequences. Interested readers can refer to [12] for a construction of the Sobol sequence, and [13] for an implementation.

In the context of statistical timing analysis, Quasi Monte Carlo techniques have been studied in [9]. The author notes that LDSs are imperfect and as the number of dimensions in the problem increases, there is degraded uniformity. This effect is especially significant among the higher coordinates of LDSs, which show undesirable patterns as opposed to the low discrepancy pattern in Figure 1. This phenomenon is referred to as *pattern dependency*. The author suggests that in timing analysis the lower coordinates of Sobol sequences, which have no significant pattern dependencies, be assigned to the important variables in the sampling procedure. Therefore, a concept of criticality of variables in timing analysis needs to be defined, which can be used to sort the variables in the order of their decreasing importance.The coordinates of the Sobol sequence can then be assigned to variables in this order. We present a technique for ordering the variables based on their criticality to circuit delay in the statistical timing framework.

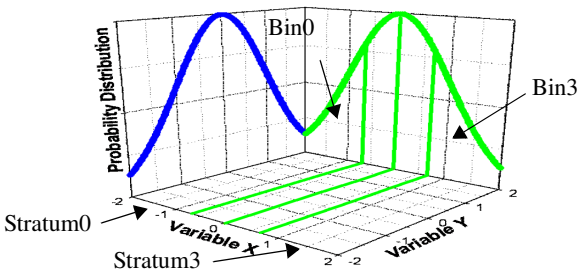

Figure 2. Stratification of a 2D space. Variable X is divided into 4 bins, thus dividing the sample space into 4 strata.

A very important related point is that effectiveness of Sobol sequences cannot be guaranteed beyond a certain number of dimensions, even with the above technique. Hence, in this work, we use Quasi Monte Carlo techniques in conjunction with stratified sampling and Latin Hypercube Sampling (LHS), such that Sobol sequences sample only limited number of dimensions of the problem. The next two subsections provide a brief overview of stratified sampling and LHS.

2.2 Stratified sampling

Stratified sampling is a technique to partition the sample space into mutually exclusive strata, and then sample using any of the known variance reduction techniques within each [11]. The stratification method in this work is illustrated for a 2D example in Figure 2, where random variable X is divided into 4 equal probability bins (X is equally likely to fall in any of the 4 bins), whereas random variable Y is not binned. This method is adopted when X is critical to the function value to be estimated, whereas Y is not. In this way, the 2D space is partitioned into 4 strata as shown in the figure. Throughout the work, we use 'bin' to refer to regions in individual variables, and 'strata' to refer to partitions in the nD space, where n is the dimensionality. In general in multidimensional space, 1 or more variables are binned, and the permutations of bins across variables define strata. In the case of timing analysis, the timing behavior of the circuit is more sensitive to the critical variables by selection and these variables are binned. Therefore within strata the timing behavior exhibits lower variation and is easier to estimate. The technique leads to accuracy with few samples, however cannot be used over very large dimensions since the number of strata increases exponentially with the number of variables binned, an issue when dealing with large number of components which may be "critical", as in statistical timing analysis.

2.3 Latin Hypercube Sampling

Latin Hypercube sampling is a technique in variance reduction which deals with multidimensional systems [14-16]. This technique tries to sample each variable involved uniformly by dividing the variable into equal probability bins. The samples from bins in variables are combined across dimensions to obtain faster convergence than random sampling. This is in contrast with taking all permutations of the bins across variables to define strata, and then sampling within each stratum as in stratified sampling described above. This means

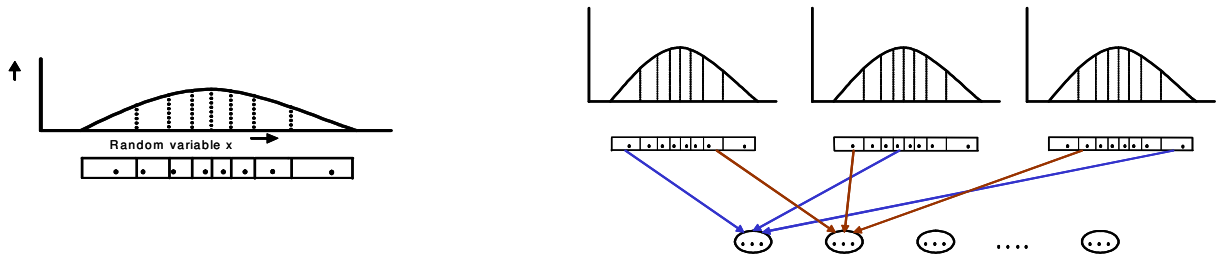

Figure 3. Latin Hypercube Sampling (a) Divide each variable in 8 equal probability bins and sample in bins. (b) Combine randomly to form 8 triplets

that LHS can deal with large dimensions, however with a moderate rate of convergence compared to full stratification.

The LHS procedure is illustrated in Figure 3. Each random variable is divided into equal probability bins. One sample is generated within each bin. Such samples are combined across variables to obtain Latin Hypercube samples. This is the procedure to obtain k samples, where k is the number of bins per variable. To obtain mk number of samples, we repeat the LHS procedure m times.

Two other techniques that have been studied for application to integrated circuit yield estimation are importance sampling and control variates. In general, these methods require more detailed information about the circuit. For literature in statistics about the method, refer to [11]. More work is required to establish the effectiveness of these approaches for use in the modern integrated circuit design process.

3. Smart sampling based on timing criticality

In this section, we first describe our process variation model and then go on to discuss our smart sampling approach.

3.1 Process variation model

Our process variation model is based on [2] which takes into account intra-die spatially correlated variation by partitioning the die into $n * n$ grids and assuming identical parameter variations within a grid. Therefore, each source of variation is represented by a set of random variables for all grids. For example, transistor gate length variation is represented by a set of random variables for all grids and the set is of multivariate normal distribution with a covariance matrix R_{Lg}. Principal component analysis is performed on these correlated random variables to obtain a set of principal components. Similarly, principal components are obtained for other sources of variation. Let $p_i : i=1,...,m$ be the principal components of all global sources of variation. In addition to these global sources of variation, we have an independent random variable Δr to account for random variation at the gate level. The delay for a gate is expressed as a linear combination of principal components of p_i's and Δr:

$$d = d_0 + k_1 \times p_1 + \ldots + k_m \times p_m + k_{m+1} \times \Delta r \qquad (1)$$

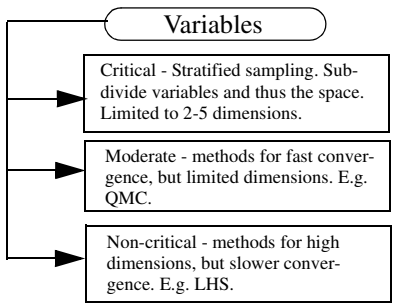

Figure 4. Using timing criticality, variables are ordered. Variance reduction techniques are applied where most effective.

where d_0 is the gate delay mean, k_i: $i=1,...,m$ are the coefficients for the principal components. p_i's and Δr are independent unit normal random variables after suitably scaling their coefficients.

3.2 Stratification+Hybrid Quasi Monte Carlo(SH-QMC)

In our smart sampling approach SH-QMC, we propose to use circuit timing criticality information to reduce the sample size for MC based statistical timing analysis. In the previous subsection, we have defined the variables representing process parameter variation. In our proposed approach, we order these variables based on their criticality to the circuit delay using a timing criticality parameter P_{crit} defined in the next subsection. We then apply Quasi Monte Carlo (QMC), stratified sampling and LHS to variables based on their convergence property and the ability to handle multiple variables (dimensions) as illustrated in Figure 4. The topmost critical variables guide the stratified sampling approach, which leads to faster convergence. Only the top 2-5 variables are used to guide stratification since the number of strata increases exponentially with the number of variables as explained in Section 2.2. QMC method is then employed on the topmost to moderately critical variables for its fast convergence properties. However, QMC can exhibit pattern dependencies with large number of variables, so only a limited number of variables are sampled using QMC. On the non-critical variables, we use Latin Hypercube Sampling which is applicable for large number of variables, but has slower convergence to an accurate result.

The method is illustrated in Figure 5 using a 5 variable example. As mentioned before, variables are ordered as critical, moderately critical and non-critical. The two most critical variables $r1$ and $r2$ are divided into 4 bins each (Figure 5a). A stratum is defined as a set of points in the 5D space restricted to one bin each in $r1$ and $r2$, but unrestricted in $r3$, $r4$ and $r5$. The total number of strata is 16, arising from 4 by 4 permutations of the bins. Figure 5b illustrates one particular stratum which we use to explain the remaining steps. In this stratum, points are restricted to bin 2 in $r1$ and bin 3 in $r2$. As shown in Figure 5c, QMC method based on Sobol sequence is used to sample $r1$, $r2$ and $r3$ in the stratum and LHS is applied to $r4$ and $r5$. Note that since we are only sampling within the stratum, samples of r1 and r2 are restricted to the respective bins. QMC generates triplets as shown in the figure. For performing LHS, $r4$ and $r5$ are divided into 8 bins each and one value is selected from each bin as in Figure 5c. 8 LHS pairs are generated by randomly picking from $r4$ and $r5$ in one step of LHS. Two LHS pairs are shown in Figure 5d. Next, the LHS pairs are combined with the QMC triplets to generate our final samples. The procedure is repeated: LHS pairs are generated again in $r4$ and $r5$, and QMC triplets are generated in the other 3 variables. These are then combined as before. After generating the samples in this stratum, we move to the next stratum and repeat our steps. In this manner, we generate samples in all 16 strata.

As mentioned in Section 2.1, among the variables on which QMC is employed, the lower coordinates of LDSs are assigned to the more critical variables. The order of criticality here is again decided using the parameter P_{crit}.

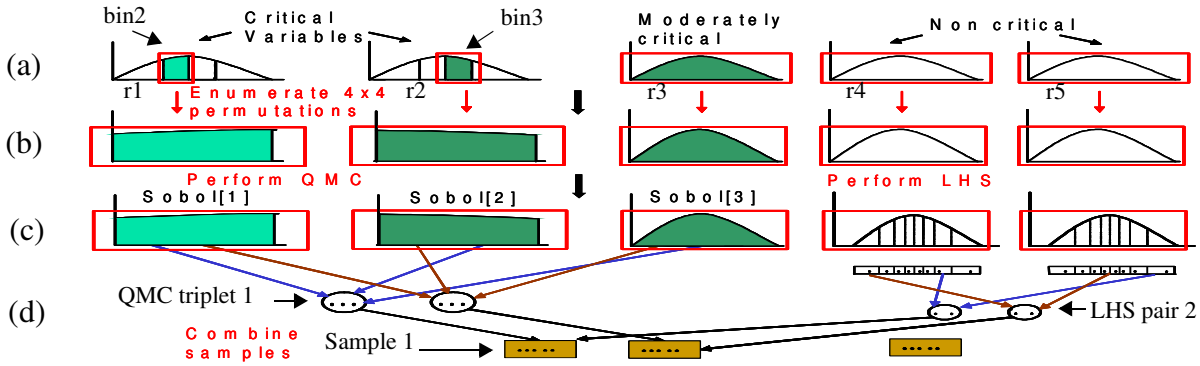

Figure 5. Stratified Latin Hypercube Sampling (a) Ordering variables based on timing criticality. (b) One of 16 strata in the sample space. (c) QMC triplets and LHS pairs. (d) These are combined to obtain final samples.

3.3 Timing criticality P_{crit}

To order the principal components, we employ a timing criticality metric P_{crit}. To compute P_{crit}, we perform static timing analysis on the nominal circuit to identify critical paths within a slack of s% of worst-case arrival time, where s is a parameter. This STA run is performed under nominal process conditions. Now, each grid is assigned a weight equal to the number of gates falling in any of the potential critical paths. Let w^i_g be the weight of the i^{th} grid. The weight of the j^{th} principal component is given by

$$w_j = \sum_i \left(w^i_g \times k_{ij} \right) \qquad (2)$$

where k_{ij} is the coefficient of the j^{th} principal component in the i^{th} grid variation. This empirical technique leads to fast computation of P_{crit} with sufficient accuracy to guide our proposed SH-QMC.

4. Incremental Evaluation of a Percentile Delay

ECO and synthesis tools require efficient incremental timing analysis techniques for fast recomputation of circuit delay with small changes in the design, while also accounting for process variation. In MC based SSTA there is a lack of incremental capability to date. In this section, we present an approach for the incremental evaluation of a specific percentile delay of a circuit with a small change in circuit sizing. We illustrate the approach for the case of single gate sizing in this work. However, the approach can be extended to the case of simultaneous multiple gate sizing. The key intuition is that if the samples for SH-QMC on circuit C are reused for C' (C with gate g sized), then most samples need not be reevaluated to recompute the x^{th} percentile delay; only those samples that have a circuit arrival time 'close' enough to the x^{th} percentile delay of C need to be reevaluated. An upperbound on change in circuit arrival time of a sample from C to C' can be determined from a local bound computation involving only a few gates connected to the gate g being resized. This bound can be used to prune out a majority of the samples, leaving us with a few that need to be reevaluated. Further speedup can be achieved with established techniques for incremental STA on the samples selected for reevaluation.

4.1 Algorithm

We perform timing analysis on an original circuit C using our SH-QMC approach and store the samples for the process variation space and the corresponding circuit arrival time in memory. Our approach for the recomputation of a specific percentile delay using the stored samples is illustrated in Figure 6. For each sample, a bound on change in circuit arrival time from C to C' (C with gate g sized) is obtained as explained in Section 4.2. Each sample has a positive bound and negative bound for either direction of change. The samples are sorted in the order of increasing circuit arrival time for C. In Figure 6a, the samples are represented by points on the circuit arrival time distribution curve. They are visited in the decreasing order of arrival time starting from the x^{th} percentile value t_x. A sample k is selected for reevaluation if its arrival time for circuit C and the positive bound for k add up to exceed t_x. For example, in Figure 6a, sample i is pruned out since its positive bound is not large enough to cross t_x. However, sample i-1 is reevaluated as it has a large enough upper bound to cross t_x. As illustrated in Figure 6b, the arrival time for i-1 is recomputed. Sample i-1 is updated with this value of arrival time which shifts t_x to the right. Next sample i-2 is reevaluated, however the arrival time value obtained is less than t_x, so t_x does not change. Sample i-2 is also updated with the recomputed arrival time value. After considering all samples to the left of t_x, we visit the samples to the right. The criterion for reevaluating a sample here is that its arrival time for C and the negative bound for the sample should add up to less than t_x. After this step, we repeat the procedure and visit samples to the left of the updated t_x. Samples reevaluated earlier are not visited again. The termination criterion is that there are no

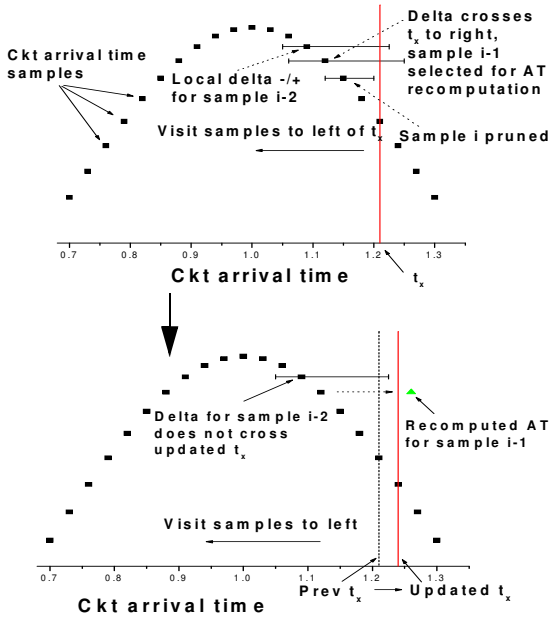

Figure 6. (a) Samples are visited in decreasing order of circuit arrival time, starting from the x^{th} percentile (t_x). Samples with delta crossing t_x are selected, others pruned. (b) Recomputation of circuit arrival time is performed at the selected sample and t_x is updated.

samples to the left or right of t_x which satisfy the criterion for reevaluation. The final value of t_x is the x^{th} percentile delay of C'.

The justification for reuse of samples is that our metric to guide SH-QMC P_{crit} (Section 3.3) is measured at the grid level in our process variation model, so within reasonable ECO changes the timing criticality of the circuit does not change to significantly alter our metric P_{crit}. In particular, we are only concerned about the relative ordering of variables based on P_{crit}. Therefore with single gate sizing, the samples are still accurate. For cases where there is significant design change, SH-QMC is performed again to generate new samples. As mentioned the samples for C are stored in memory. Our results on the benchmarks studied demonstrate that the number of samples for SH-QMC that gives sufficient accuracy is 80 for the largest circuits. Therefore, we need to store 80 samples for each gate. In general, if the number of samples required is much higher, the memory overhead could be significant. Section 3.1 defined the variables to model process variation, which are the principal components for all sources of variation and an independent random component at the gate level. Now, it is enough to store samples for these components, as the device parameters can be retrieved using the values of components. Storing samples for the principal components incurs negligible memory overhead. In the case of the independent random component, instead of storing all samples of the component for all the gates, we store the initial 'seed' value for the pseudorandom number generator. Note that for STA, gate delays are propagated in the topological order. This offset in the topological order along with the 'seed' value is provided to the pseudorandom number generator which reproduces the random numbers while incremental analysis is performed.

4.2 Computing circuit arrival time bound for samples

We compute the maximum possible increase and decrease in the circuit arrival time for each sample of circuit C using local gate delay change information when gate g is sized. Define sets $Fi(g)$ of fanin gates of g, $FoFi(g)$ of fanouts of gates in $Fi(g)$ and $Fo(g)$ of fanout gates of g. We select subpaths that are candidates for obtaining the bounds in circuit arrival time and evaluate the change in delay of these subpaths when g is sized. Every subpath starting from an input pin of a gate in $Fi(g)$ and ending in an output pin of a gate in either $Fo(g)$ or $FoFi(g)$ is a candidate for this evaluation. Some such subpaths could have more than one gate in $Fi(g)$. We assume that delay change is significant only in the gates in the three sets defined above, therefore only these gates affect the change in subpath delay. Now, we obtain bounds for circuit arrival time change for a sample S as follows. Let $P(g)$ be the set of all candidate subpaths. $t_S(p)$ and $t'_S(p)$ are delays for subpath p in sample S before and after sizing gate g, respectively. Then the negative and positive bounds are given by:

$$delta\_neg(g,S) = min\{t_S'(p) - t_S(p) \forall p \in P(g), 0\} \quad (3)$$

$$delta\_pos(g,S) = max\{t_S'(p) - t_S(p) \forall p \in P(g), 0\}. \quad (4)$$

In other words, we find the maximum and minimum values of the change in delay of candidate subpaths. As gate delay change is assumed to be significant only in the local subcircuit (set of gates belonging to $Fi(g)$, $Fo(g)$ and $FoFi(g)$), the computational overhead is low. In our algorithm in Section 4.1, we only need either of $delta\_neg$ or $delta\_pos$ for most samples. A $delta\_neg$ or $delta\_pos$ computation for a sample involves gate delay computation and propagation in the local subcircuit twice, one each before and after gate sizing. Therefore, the cost of arrival time bound computation across all the samples for the percentile delay recomputation is approximately twice that of performing Monte Carlo analysis on the local subcircuit with smart samples. The runtime for this is negligible compared to that of a single STA run for most practical circuits.

Table 1. Comparison of random sampling, LHS based and SH-QMC approaches based on sample size required to achieve target accuracy on benchmarks studied. The last two columns show the speedup of LHS and SH-QMC respectively, over random sampling.

Circuit	No of gates	RS count	LHS count	SH-QMC count	LHS speedup	SH-QMC speedup
C432	256	1120	1120	240	1	4.7
C499	544	1760	1360	40	1.29	44
C880	500	1760	1440	80	1.22	22
C1908	603	1440	960	80	1.50	18
C2670	780	1600	1200	80	1.33	20
C3540	1163	2320	1440	160	1.61	14.5
C5315	1692	2160	1120	80	1.93	27
C6288	3834	1840	880	80	2.09	23
C7552	2152	3040	1280	80	2.38	38
VD1	14503	1360	800	80	1.70	17
VD2	34082	2000	880	80	2.27	25
USB	32898	2240	1200	80	1.87	28
ETHER	57327	2080	1600	80	1.30	26
VGA	90831	2000	800	80	2.50	25

5. Results

Our simulation results are based on a 90nm industrial technology library. In our implementation we only consider channel length variation as a source of process variation for simplicity. However, this is not a limitation of our approach, which is general and can handle all sources of variation. The inter-die spatially correlated intra-die and uncorrelated random components of channel length variation are considered. The overall standard deviation is 10% of nominal channel length. This amount of process variation increases absolute variability, but more importantly serves to highlight the accuracy comparison of the techniques considered. The number of grids in the spatial correlation model for individual circuits is varied linearly with post-placement area starting from 2 by 2 for the smallest circuit to 16 by 16 for the largest circuit. This corresponds to a grid area of approximately 40µm by 40µm for all the circuits. We compare our proposed SH-QMC approach with random sampling and LHS based techniques. Simulations are performed on ISCAS85 benchmark circuits [17], and 5 large circuits. These are Viterbi Decoder 1(VD1), Viterbi Decoder 2(VD2), USB2.0 Core (USB), Ethernet MAC Core (ETHER) and VGA Controller Core (VGA), with gate counts varying from approximately 15,000 to 90,000. We perform synthesis and APR on all the circuits using commercial tools.

Our comparisons are based on the error in estimating statistical moments of arrival time distribution for a given method w.r.t the moments from a golden of 40,000 Monte Carlo runs. Consider for

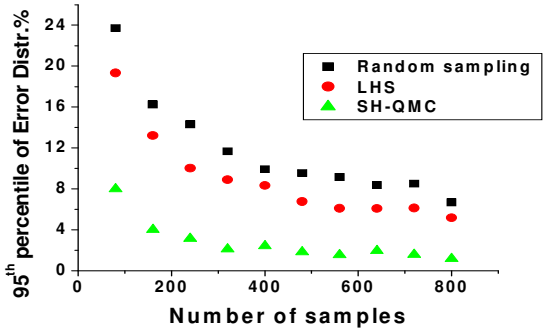

Figure 7. Error comparison of random sampling, LHS and SH-QMC for a VGA circuit (90831 gates) w.r.t. golden of MC count 40,000.

example a given trial MC_1 of size 100 samples. This gives a circuit arrival time distribution. From this, moments μ_1 and σ_1 (mean arrival time and standard deviation in arrival time) are obtained and error (magnitude of deviation from the golden) calculated for both. From repeated trials (each of 100 samples in this example), we get 2 distributions for error. The nature of the error distributions show the efficiency of the technique. For example, as we increase the number of samples from 100 to 200 in the above example and repeat the experiments, the error distribution is expected to get tighter and closer to zero. In particular, the 95th percentile of the error gets closer to zero and we use this value as a criterion to compare different techniques. The minimum number of samples required by a technique such that the 95th percentile of error distribution is less than 5% for both mean arrival time and standard deviation of arrival time is our performance metric for the technique.

Table 1 compares the number of samples required for random sampling, an LHS-based technique and our proposed SH-QMC approach. The proposed approach achieves on an average 23.8X reduction (lowest 4.7X up to 44X) in number of samples w.r.t random sampling, whereas LHS achieves a modest improvement of 1.7X on average (lowest 1X up to 2.5X). The improvements are consistent across the benchmark circuits studied. In Section 3.3, we mention that critical paths are identified within a slack of s% for computing timing criticality P_{crit}. We investigated the sensitivity of the results to the parameter s and found that varying s from 1-5% showed no changes in the number of samples required to meet the stated accuracy objective, indicating that the proposed technique is stable with respect to this parameter. Figure 7 visually presents the 95th percentile of error of random sampling, LHS and SH-QMC for our largest circuit VGA (90831 gates) w.r.t. the golden model. Though we have two error distributions (corresponding to mean and standard deviation of arrival time), our simulations show that the error in estimating standard deviation always dominates the error in mean. The error plotted in Figure 7 is therefore for the standard deviation of arrival time.

Table 2 compares the runtime of SH-QMC and traditional SSTA. For both mean and standard deviation of arrival time the error for SH-QMC in the table is the average absolute deviation from their values in the golden model; for traditional SSTA this is the error w.r.t the golden. The golden model is MC with 40,000 samples. One drawback of Monte Carlo techniques in general is that every time an experiment is performed, the error w.r.t golden is different. This means that the error in one particular MC experiment is sometimes higher than the average value mentioned. However, the 95th percentile of the absolute error distribution is still less than 5% for all the circuits in the table. This translates to an error of 3-7ps in absolute time for different circuits, which is a reasonable target for the given process technology. All our simulations were performed on a single Quad Core processor. For SH-QMC, we perform two different experiments, in one we spawn 4 threads to use the parallelism in the Quad Core machine, and in the other we run a single thread on the

Table 2. Runtime comparison of SHQMC with SSTA. AT = circuit delay

Circuit	No of gates	Mean AT Error(%)		σ AT Error (%)		SSTA Runtime(s)	SH-QMC Runtime(s)	
		SSTA	SH-QMC	SSTA	SH-QMC		Multi thread	Single thread
VD1	14503	1.56	0.08	2.43	1.80	0.92	0.83	2.9
VD2	34082	1.66	0.34	2.37	2.12	3.79	2.42	8.7
USB	32898	1.36	0.53	3.48	1.85	4.37	4.22	14.2
ETHER	57327	0.35	0.05	1.8	2.3	8.18	6.2	19.9
VGA	90831	0.40	0.08	0.03	1.80	9.93	6.85	22.1

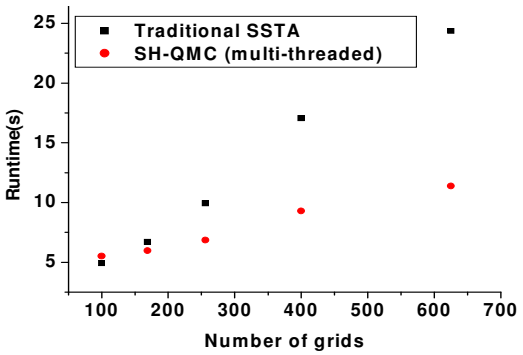

Figure 8. Performance comparison of traditional SSTA with multi-threaded SH-QMC for VGA circuit (90831 gates) as function of number of grids in process variation model.

machine. The former uses the parallelism in sample evaluation, which is straightforward in MC methods but not true of traditional SSTA. Parallelizing traditional SSTA is non-trivial and would incur runtime cost. We consider circuits with more than 10,000 gates for meaningful runtime comparisons. SH-QMC with multi-threading performs better than traditional SSTA in runtime. Also, further speedup in SH-QMC can be achieved in a straightforward manner using parallel processing on more than one processor. Figure 8 compares the performance of traditional SSTA with SH-QMC for the VGA circuit as a function of number of grids in the process variation model. This clearly illustrates that SH-QMC scales better than traditional SSTA. Figure 9 is a typical case comparison of efficiency in estimating a high percentile statistic in arrival time distribution obtained from our approach w.r.t a traditional SSTA approach. The error in estimating the 99th percentile arrival time for SH-QMC is better than traditional SSTA at more than 72 samples for the USB2.0 Core circuit (32898 gates) considered. In general, our approach estimates the 99th percentile arrival time better than traditional SSTA for all benchmark circuits studied at a low number of samples. Figure 10 compares the probability distribution curve of arrival time of the USB circuit for SH-QMC (96 samples) and a traditional SSTA approach, w.r.t the golden. Our technique captures the mean arrival time (marked with vertical lines) and the overall shape of the distribution better than the traditional SSTA approach.

Table 3 presents our results for the incremental evaluation of the 95th percentile and 99th percentile delay after a gate size change using our approach in Section 4. In our experiments, we select 100 gates at random for a given circuit. Each gate is sized up individually and the percentile delays recomputed. Our simulations show that on average only 1.2% and 0.8% of samples need to be reevaluated for exact recomputation of the 95th percentile and 99th percentile delays after sample size reduction using SH-QMC.

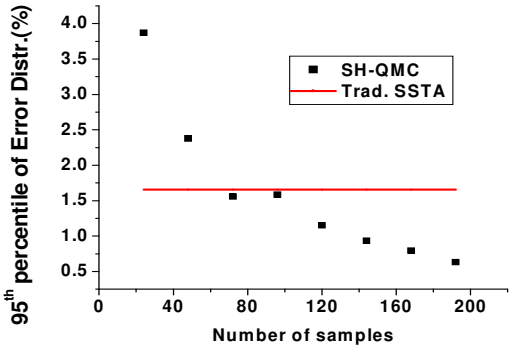

Figure 9. Comparison of 99th percentile error of SH-QMC vs. traditional SSTA w.r.t golden of 40,000 MC count for USB circuit.

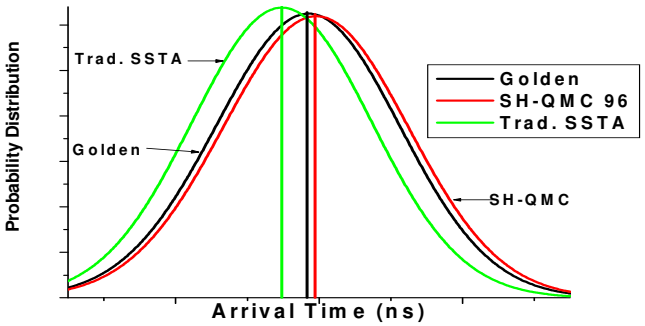

Figure 10. Arrival time distribution of SH-QMC (96 samples) and traditional SSTA w.r.t golden(40,000 MC) for USB circuit.

6. Conclusions

This paper presents a Stratification + Hybrid Quasi Monte Carlo (SH-QMC) approach to improve the efficiency of MC based statistical static timing analysis. The proposed approach uses easily computable timing criticality information, and achieves on average 23.8X and upto 44X reduction in the number of samples required for timing estimation compared to a random sampling approach. With multi-threading on a quad core processor for SH-QMC, the approach is faster than traditional SSTA for comparable accuracy. Also, further speedup of SH-QMC is straightforward using parallel processing across machines. In addition, SH-QMC scales better than traditional SSTA with circuit size. Our approach estimates the 99th percentile arrival time better than traditional SSTA for benchmark circuits studied using only a low number of samples. We proposed an incremental approach to recompute a percentile delay metric after ECO. The results show that on average only 1.2% and 0.8% of original samples need to be evaluated for exact recomputation of the 95th percentile and 99th percentile delays after ECO.

References

[1] C. Visweswariah, et al., "First-Order Incremental Block-Based Statistical Timing Analysis," *Proc. Design Automation Conference*, pp 331-336, 2004.

[2] H. Chang, and S.S.Sapatnekar, "Statistical Timing Analysis Considering Spatial Correlations using a Single Pert-Like Traversal," *Proc. International Conference on Computer-Aided Design*, pp. 621-625, 2003.

[3] K. Chopra, et al., "Parametric yield maximization using gate sizing based on efficient statistical power and delay gradient computation", *Proc. International Conference on Computer Aided Design*, pp 1023-1028, 2005.

[4] F. N. Najm,N. Menezes, "Statistical timing analysis based on a timing yield model," *Proc. Design Automation Conference*, pp. 460-465, 2004.

[5] L. Scheffer, "The Count of Monte Carlo," *TAU*, 2004.

[6] M. Keramat and R. Kielbasa, "Worst Case Efficiency of LHSMC Yield Estimator of Electrical Circuits." *Proc. ISCAS*, v. 3, pp. 1660 - 1663, 1997.

[7] R. Kanj, R. Joshi, and S. Nassif, "Mixture Importance Sampling and Its Application to the Analysis of SRAM Designs in the Presence of Rare Failure Events," *Proc. Design Automation Conference*, pp. 69-72, 2006.

[8] S. Tasiran and A. Demir, "Smart Monte Carlo for Yield Estimation," *TAU*, 2006.

[9] A. Singhee and R.A. Rutenbar, "From Finance to Flip Flops: A Study of Fast Quasi-Monte Carlo Methods from Computational Finance Applied to Statistical Circuit Analysis", *Proc. ISQED*, pp. 685-692, 2007.

[10]E. Hlawka,"Funktionen von beschrankter Variation in der Theorie der Gleichverteilung", *Ann. Mat. Pura Appl.*, 54, pp 325-333., 1961.

[11]R.Y. Rubinstein, Simulation and the Monte Carlo Method, John Wiley & Sons, Inc., 1981.

[12]I.M.Sobol, "The Distribution of Points in a Cube and the Approximate Evaluation of Integrals", *USSR Comp. Math and Math. Phys.,* 7(4), pp. 86-112, 1967.

[13]P. Bratley, B. Fox, "Algorithm 659: Implementing Sobol's Quasirandom Sequence Generator", *Trans. on Mathematical Software*, 14(1), pp 88-100, 1988.

[14]M.D.McKay, W.J.Conover and R.J.Beckman,"A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code",*Technometrics,*21,pp 239-245,1979.

[15]M. Stein, "Large Sample Properties of Simulations Using Latin Hypercube Sampling," *Technometrics*, 29,pp 143-151, 1987.

[16] W.L. Loh, On Latin Hypercube Sampling, *The Annals of Statistics,* Vol. 24, No. 5, 2058-2080, 1996.

[17]F. Brglez and H. Fujiwara, "Neutral netlist of ten combinational benchmark circuits and a target translator in FORTRAN", Special session on ATPG and fault simulations, *Proc. ISCAS*, pp. 695-698,1985.

Table 3. Performance of incremental evaluation of 95th and 99th percentile delay with gate size change for SH-QMC with 80 samples. AT= Arrival Time

Circuit	No of gates	Avg. Incremental evaluations per gate		Avg. Incremental evaluations per gate/sample size(%)	
		95th percentile AT	99th percentile AT	95th percentile AT	99th percentile AT
C432	256	0.72	0.61	0.90	0.76
C499	544	0.755	0.735	0.94	0.92
C880	500	0.73	0.545	0.91	0.68
C1908	603	0.94	0.525	1.18	0.66
C2670	780	1.015	0.7	1.27	0.88
C3540	1163	0.57	1.01	0.71	1.26
C5315	1692	1.235	0.565	1.54	0.71
C6288	3834	0.545	0.505	0.68	0.63
C7552	2152	1.19	0.985	1.49	1.23
VD1	14503	1.515	0.51	1.89	0.64
VD2	34082	0.54	0.515	0.68	0.64
USB	32898	1.625	0.57	2.03	0.71
ETHER	57327	0.96	0.535	1.20	0.67
VGA	90831	0.84	0.505	1.05	0.63