Speedpath Analysis Based on Hypothesis Pruning and Ranking *

Nicholas Callegari[1], Li-C. Wang[1], Pouria Bastani[2†]
[1]University of California - Santa Barbara
[2]Intel Corporation

ABSTRACT

In optimizing high-performance designs, speed limiting paths (*speedpaths*) impact the performance and power trade-off. Timing tools attempt to model and capture all such paths on a chip. Due to the high performance nature of these designs, critical paths predicted by the timing tools often do not match the actual speedpaths found on silicon chips. Early silicon data therefore is used to identify the speedpaths, and further performance optimization is carried out by pushing the delays on these paths. In this context, the paper presents a novel data mining approach that analyzes a small number of identified speedpaths against a large number of non-speedpaths. The result of this analysis for each speedpath is a set of hypotheses explaining why the path is special. These hypotheses can be used in guiding the search for the root causes, or in predicting additional paths as potential speedpaths. We demonstrate the feasibility of this approach and summarize our findings based on analysis of silicon speedpaths collected from a 65nm microprocessor.

Categories and Subject Descriptors

B.8.2 [**Hardware**]: Performance Analysis and Design Aids

General Terms

Algorithm, Performance, Reliability

Keywords

Speedpath, Timing Analysis, Data Mining

1. INTRODUCTION

For high-performance design, timing analysis and performance optimization does not stop at the first silicon. Silicon information is analyzed carefully to drive further speed and power optimization. This process is called *silicon steppings*, that involves detection and improvement of speed limiting paths (*speedpaths*). A speedpath is a path that limits the performance of a chip where the performance can be measured by observing the result of applying (functional) legacy tests. Because speedpaths can be observed at different cycles of such a test, a single chip can have multiple speedpaths [1].

Due to the nature of high performance design, speedpaths are not well-predicted by typical timing flows. The deficiency comes due to a multitude of process, design and environmental effects that are either unknown or too difficult to model and simulate accurately in the design process. Therefore, for high-performance high-volume microprocessors, it is often more (cost) effective to uncover speedpaths using actual silicon samples. In each silicon stepping, a set of speedpaths are identified where performance is further improved by pushing the delays on these paths. If causes can be established to explain the speedpaths, this information can be feedback to the design and more paths can be identified as potential speedpaths which can be fixed before they show up as speedpaths in the subsequent silicon stepping. These iterations continue until the performance is pushed to a satisfactory level, at which time design changes are frozen and mass production begins.

Each silicon stepping has significant associated manufacturing cost. Additionally, identifying and verifying silicon speedpaths demands a huge amount of engineering effort. Because of the high cost, the number of speedpaths found at each silicon steppings is typically small, for example in the 10's or 100's. Hence, once this small set of speedpaths are collected, they are treated as precious resources. One desires to uncover and utilize the information presented with these paths as much as possible for guiding performance optimization before the next silicon stepping. However, it is not always obvious what the most effective approach is to best uncover and utilize the information on a few identified speedpaths.

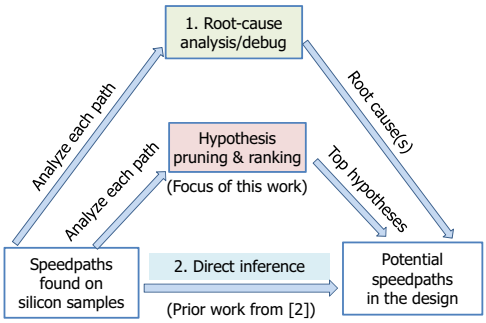

Figure 1: Three approaches to analyze speedpaths

There exists two ways by which one may use the information to find additional speedpaths. Figure 1 illustrates the two approaches: *root-cause analysis*, and *direct inference* [2]. In root-cause analysis, one tries to obtain the root cause(s) as why a path is a speedpath. The root cause(s) can then be used to identify potential speedpaths by finding all paths containing the same cause(s). While root-cause analysis is intuitive, it may not be the most desirable approach because of the usual high cost associated with it [3]. *Direct inference* offers a short-cut for finding potential speedpaths. If finding potential speedpaths is our ultimate goal, then one can simply collect all paths whose characteristics are very similar to the identified speedpaths. This is a search-by-similarity strategy and in this approach, one does not need to know the reasons behind speedpaths.

*This work supported in part by CA Micro/Intel project No. 07-079.
†This work began while Dr. Bastani was a UCSB graduate student.

DAC'09, July 26-31, 2009, San Francisco, California, USA

In [2], the authors present a direct inference method that utilizes one-class *support vector machine* (SVM) [4] to implement the search-by-similarity strategy. The result of this learning is a one-class SVM model that characterizes the speedpaths. This model can then be used to scan all other paths and rank them according to their similarities to the speedpaths. From this ranking and a proper selection criterion, potential speedpaths can be derived.

The direct inference approach is attractive because it avoids the difficulty of finding the root causes. However, it does not offer the advantage for finding guides to improve timing models and tools. Furthermore, it may not be as effective as the root-causing approach for finding all potential speedpaths. For example, a speedpath and a potential speedpath may share the same root cause and hence, they are similar if one focuses the comparison of their characteristics only on the cause. However, they may be deemed dissimilar by the direct inference because direct inference has no idea about the specific cause and utilizes the overall characteristics of the two paths as the comparison basis for deciding similarity. From this perspective, we see that the direct inference, although much more efficient, may only find a subset of the potential speedpaths.

The limitation of the direct inference approach and the high cost of the root-cause analysis approach, motivate the development of a third approach, the *hypothesis pruning and ranking* (HPR) as shown in Figure 1. In the HPR approach, we assume two sets of paths are given, a small set of speedpaths and a large set of non-speedpaths. Because the speedpath set is small, we further assume that there is a significant degree of diversity in the characteristics of those paths. Note that if this assumption is not true, one can use the approach in [2] to group paths because one-class SVM in essence produces clusters where similar paths are grouped together. The set of non-speedpaths can be found by finding all paths sensitized by the tests and do not show up as speedpaths in testing. While a speedpath set may contain tens of paths, a non-speedpath set may contain millions.

The strategy of the HPR approach is to analyze each speedpath individually against the information presented in the non-speedpath set. Conceptually, this is achieved by two steps: (1) forming a hypothesis space that contains all the hypotheses of interest, (2) pruning the impossible or unlikely hypotheses based on information presented with the non-speedpaths, and if possible further ranking the remaining hypotheses. The result of HPR approach is therefore a rank of hypotheses and this rank may be based on a partial ordering. Top hypotheses (that describe special characteristics of the speedpaths) from the rank can be used for two purposes, for guiding the search in root-cause analysis and, as that in the direct inference approach, for finding potential speedpaths.

In this paper, we present two methods to implement the HPR approach. For forming the hypothesis space, the two methods use the same idea of incorporating *features* to describe path characteristics as that introduced in prior works [2, 5]. However, for pruning and ranking, the two methods employ different techniques. The first method utilizes techniques derived from Association Rule Mining [6, 7] that is often applied to mining large business database. The second method utilizes the kernel learning concept in SVM [4] in conjunction with the rule analysis. While the first method is more intuitive to understand, we will demonstrate that the second method is much more flexible and easier to implement. Through experiments, we will demonstrate that the second method can find all top hypotheses found by the first method. The experiments are based on silicon speedpaths from a 65nm microprocessor.

The rest of the paper is organized as the following. Section 2 discusses the use of *features* to form the hypotheses space and points out the fundamental challenge in the HPR approach. Section 3 presents the basic concepts in rule mining. Section 4 discusses the concept of *confidence* and presents a way to evaluate the confidence of a hypothesis in the context of proposed hypothesis pruning and ranking. Section 5 presents the first method and Section 6 presents the second method. Section 7 presents the experimental results and Section 8 concludes.

2. FEATURES AND HYPOTHESIS SPACE

A hypothesis space is formed based on *features*. A feature can be seen as an indicator of potential concern on a path. There can be two types of features, *occurrence based* and *description based*. An occurrence based feature has a binary value. For example, a cell by itself can be a feature. Then, given a path, if the path contains the cell, the value of that feature is 1. Otherwise, the value is 0. A description based feature has a numerical value. For example, the capacitive load associated with a specific cell on a path can be a description based feature.

Given a set of n occurrence based features $F = \{f_1, \ldots, f_n\}$, the initial hypothesis space is the power set 2^F. In other words, a hypothesis is simply a combination of features. However, on a speedpath, usually not all features would appear. Suppose m features (denoted set Q) appear on the path, then the hypothesis space for the speedpath is the power set 2^Q. This is because if a feature does not appear on the path, it cannot be used to explain why the path is special. From this perspective, we see that even though one may use a large set of n features to describe all paths, on analyzing a single speedpath, the search space can be much smaller, i.e. $m \ll n$. However, even for m in the order of tens, this search space can still be too large to enumerate explicitly. Hence, in the analysis, the hypothesis space exists implicitly.

If F contains n description based features, then a hypothesis space essentially contains an infinite number of hypotheses. The intuition to see this is by linking description based features back to occurrence based features. To form a hypothesis space as the power set of the features, one needs to convert n description based features into n corresponding occurrence based features. However, there are infinite many ways for this conversion.

For example, suppose we have two description features, A and B. Suppose on a speedpath p the characteristic is described as $\{A = 0.2001, B = 0.4\}$. Suppose we have two non-speedpaths with the feature vectors described as $\{A = 0.2, B = 0.5\}$ and $\{A = 0.3, B = 0.3\}$. To explain p, one can convert A, B into two occrrence based features A_0, B_0 where A_0 has value 1 if $A = 0.2001$, $A_0 = 0$ if $A \neq 0.2001$, $B_0 = 1$ if $B = 0.3$ and $B_0 = 0$ if $B \neq 0.3$. Observe that A_0, B_0 are very specific to the path p.

Alternatively, one may think that A_0, B_0 are too specific and replace them with two new features: A_1, B_1 where $A_1 = 1$ if $A_1 \in \{A > 0.2001 \cap A < 0.2002\}$ and $B_1 = 1$ if $B_1 \in \{B > 0.35 \cap B < 0.45\}$. In the second conversion method, two ranges are used for the two features. As we can see, there are infinite many ways to decide on these two ranges to create the two occurrence based features for deriving the power set where each way would interpret the hypotheses in the hypothesis space differently. Hence the hypothesis space becomes infinite.

This simple example illustrates that in the continuous space, there are infinite many ways to partition the space to form a set of hypotheses. Hence, the fundamental challenge in hypothesis ranking is how to perform effective search in a continuous hypothesis space.

2.1 Selecting features

Usually, a speedpath is due to effects that are not accounted for in the timing flow. This can be because their rarity of occurrence along with the large additional cost to model them. Selecting a proper set of features to begin undoubtedly requires some knowledge about these unaccounted effects. Although this is an important step for the

Path	Feature				
	F1	F2	F3	F4	F5
P1	**1**	**1**	**1**	**1**	**0**
P2	1	1	1	0	1
P3	0	1	1	1	0
P4	0	1	1	1	1
P5	1	1	1	0	1
P6	0	1	1	1	0

Table 1: Sample Data

proposed approach to be effective, it should not be seen as a severe limitation for the following reasons. First, the approach does not require to select a minimal set and hence, if one is not sure about some features, the solution is simply to include them. Second, a feature is a single-order effect that can be part of the concern. Usually, it is not too difficult for an experienced designer (or a process engineer) to point out such single-order effects. For example, a design can list a set of cells that she/he feel less confident on their timing modeling. Third, a missing feature does not necessarily degrade the effectiveness of the approach significantly. For example, suppose the root cause is a combination of three features $\{A, B, C\}$ and suppose the feature set only includes A and B. This does not imply that the hypothesis $\{A, B\}$ will not be ranked high. From the viewpoint of fixing the potential speedpaths, fixing all paths with $\{A, B\}$ would have included the paths with $\{A, B, C\}$. This may lead to over-fixing, but the effectiveness with respect to the performance improvement does not suffer.

The authors in [2, 5] group features into five categories of effects: **Topological effects** that are layout dependent, **Dynamic effects** that are test patterns dependent, **Static effects** that are independent of test patterns, **Statistical effects** that are process dependent, and **Random effects** that do not belong to the previous four. From the set of effects described above, an engineer develops thus a set of features that can potentially be used to differentiate between non-speedpaths and speedpaths. In this work, based on our domain knowledge and application scenario we chose the following features:

1. Active device type in a path due to logic sensitization. These are based on four different flavors of N and P MOS transistors.
2. The predicted arc or stage delay for a particular cell in the path. This is a function of input transition time, output load and net resistance and capacitance.
3. Percent of total path delay due to the nets as opposed to cells. $\%Net = NetDelay/(TotalPathDelay)$
4. Dynamic switching activity in the region.
5. Cross coupling aggressor impact.
6. Temperature variation, using Infrared Emission Microscopy to obtain a temperature map during functional operation [8].

Note that most of the features are description based. With these features every path can now be described as a feature vector.

3. HYPOTHESIS PRUNING

To illustrate the process of hypothesis pruning, Table 1 shows a simple example with a single speedpath, P1, and five other non-speedpaths, paths P2,. . .,P6, where each path is descried with with five occurrence based features, F1-F5. The hypothesis space, i.e. the power set, is illustrated as a *concept lattice* as shown in Figure 2. Notice that in the lattice, feature F5 does not appear because it does not appear on the speedpath. In this lattice, each node represents a hypothesis which can also be called a *monomial*.

For example, $\{2\}$ denotes the single-order hypothesis {F2}. We call it "single-order" because it consists of one feature. Note that since F2 appears on all five non-speedpaths, its *coverage* by the non-speedpath set is 5. Similarly, the coverage of {F2,F3} is also 5. This is why a label "5" is on the left of the node $\{2, 3\}$.

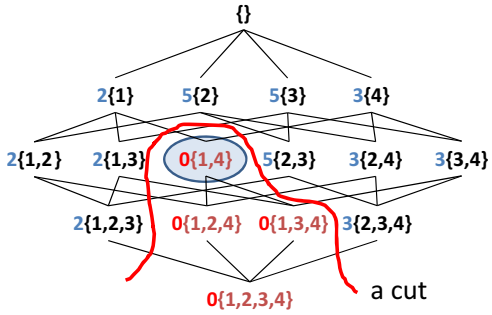

Figure 2: The lattice view of the hypothesis pruning

Because the monomial {1,4} does not exist on any of the non-speedpaths, its coverage is zero. Since $\{1, 4\}$ is not covered, all its decedents in the lattice are not covered. The four hypotheses with coverage zero are therefore the ones left after the pruning process. They are considered *consistent* with the dataset presented, i.e. not appearing on any of the non-speedpaths. The monomial $\{1, 4\}$ is a *2-order* hypothesis because it consists of two features. In general, one would like to find the lowest-order hypotheses that are consistent with the dataset and are not the decedents of any other hypothesis. This is because (1) it is easier to process the information presented in lower-order hypotheses than higher-order hypotheses, and (2) any higher-order hypothesss consistent with the dataset would have been included in its corresponding lower-order hypothesis.

Finding all the lowest-order hypotheses consistent with the dataset translates to finding a *cut* on the concept lattice. Above this cut are all nodes with non-zero coverage by the non-speedpaths. In Figure 2, the cut would separate the four nodes $\{1, 4\}$, $\{1, 2, 4\}$, $\{1, 3, 4\}$, $\{1, 2, 3, 4\}$ from the rest of the nodes.

Finding a cut in such a lattice can be challenging if the lattice size is very large. Various algorithms were proposed in association rule mining [6] for efficient finding of such a cut. For example the very first Apriori algorithm [9] is based on a breadth-first search strategy. If one expects the cut deep in the lattice (from the top), the maximum itemset algorithm MAFIA [10] that follows a depth-first search strategy, would be more efficient. The difference between association rule mining and the hypothesis pruning in our work, lies in the ways to decide *consistency*. In hypothesis pruning, this consistency is decided by checking if a hypothesis appears on any one of the non-speedpaths. In rule mining, the consistency of an itemset (an itemset corresponds to a node in the lattice) is decided by a *support-confidence* evaluation [6]. The support calculation corresponds to the consistency checking in our formulation. Several methods have been proposed for efficient calculation of the support [11]. For hypothesis pruning, the bit-set based representation proposed in [10] provides good efficiency for implementation.

4. CONFIDENCE AND ITS EVALUATION

The next step is to evaluate the confidence for hypothesis {F1,F4} (which will be same for all three higher-order hypotheses below it). The simplest way to evaluate the confidence is to calculate the ratio of the lattice space that is covered. Figure 3-(I) illustrates this concept. By assuming that the empty hypothesis is always covered, we see from Figure 2, 12 out of the 16 nodes are covered. Hence, the confidence is 75%. In evaluating all speedpaths individually, all decedents of $\{F_1, F_4\}$ would also have the same confidence. Hence, with this method, we cannot differentiate in between the hypotheses that are consistent with the dataset. We can, however, compare the confidence obtained based on one speedpath path to another.

The idea to relate confidence to the size of the non-covered subspace H can be justified with a simple probability argument as the following. For a hypothesis $h \in H$, denote the probability that h

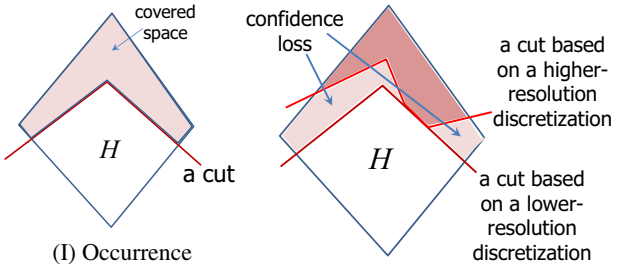

Figure 3: Confidence measured as coverage in hypothesis space

is not true as $error(h)$. We want $error(h) < \epsilon$. The probability that h is not true and h does not appear on m randomly selected non-speedpaths is $\leq (1-\epsilon)^m$. For all hypotheses in H, any one of them is not true and does not appear on m randomly selected non-speedpaths is therefore $\leq |H|(1-\epsilon)^m \leq |H|e^{-m\epsilon}$. Suppose we want this probability to be bounded by δ. Then, we can set $\delta = \frac{1}{m}ln(\frac{|H|}{\epsilon})$. In this equation $(1-\delta)$ is the confidence that is a function of $|H|$ and the error probability ϵ. We see that for a given ϵ, the larger the $|H|$ is, the smaller the $(1-\delta)$ is.

4.1 Dealing with description based features

As discussed in Section 2, for description based features, a concept lattice can be formed by converting the features into occurrence based features. Once such a conversion is decided, the resulting lattice for the speedpath is determined. Then, a cut can be found on this lattice based on the non-speedpaths. Below we discuss how the conversion method can influence the location of such a cut.

Suppose we are given with a set of features whose values fall into the range $[0, 10]$. A simple way to convert these features into occurrence based features is to divide the range into two sub-ranges of equal length: $[0, 5), [5, 10]$. Note that given a speedpath, any one of its features will fall into either one of these two sub-ranges but not both. This is also true if we, instead of dividing the range into two, divided it into four sub-ranges of equal length. Suppose in a process, we continue to divide $[0, 10]$ into into $2, 4, \ldots, 2^i$ sub-ranges of equal length to build a sequence of hypothesis lattices for a speedpath. Notice that in this lattice, the name(s) of the feature(s) labeling each node do not change. They are the features appearing on the speedpath. Hence, all the lattices have the same size. The only thing that changes is the sub-range each feature falls into. In the process, we call each way to divide the range a *discretization* method. We say that the jth discretization has a *lower resolution* than the $(j+1)$th discretization.

Given such a sequence of lattices and a non-speedpath set, we can obtain a sequence of cuts, one on each lattice. Figure 3-(II) illustrates that the cut from a *higher resolution* discretization cannot be lower than a cut from a *lower resolution* discretization. The intuition behind this is that a higher resolution discretization means a smaller sub-range specified for each feature on the speedpath. A smaller sub-range means it would be harder for feature values from the non-speedpaths to fall into that sub-range, and therefore some nodes covered before with a bigger sub-range, could become uncovered. Intuitively, what this is showing is that, a higher resolution discretization scheme would result in hypotheses that are "more specific" to the speedpath, which may reduce their confidence because of reduced coverage on its corresponding lattice.

Following the example discussed in Section 2 before, for example, a hypothesis $h_1 : \{A \in [0.2, 0.25], B \in [0.35, 0.45]\}$ would be considered more specific than $h_2 : \{A \in [0.2, 0.3], f_2 \in [0.3, 0.5]\}$ because the sub-ranges associated with the two features in h_1 are smaller. Based on our confidence evaluation scheme, the confidence of h_1 is $\leq$ the confidence of h_2.

It is clear from the discussion that if we intend to find the hypotheses with the highest confidence, then we would use the lowest-resolution discretization scheme. However, this may correspond to a cut lower in the lattice and return hypotheses with very high-orders. High-order hypotheses are less preferred because they can also be too specific to the speedpath (in the extreme case, one can say that all features together on the speedpath is the reason but this tells us little information regarding why the path is special). Hence, in searching for good hypotheses, we will try to find the ones in the low orders (such as 2nd, 3th) with the lowest discretization resolution such that in the resulting lattice, the hypotheses are consistent with the given dataset and never appear on the non-speedpaths.

5. DISCRETIZATION METHOD

The discretization strategy is commonly utilized most of the association rule mining research where items are with numerical values. The most reasonable method for discretizing continuous features is based on one's intuition, i.e. design knowledge on the features. For those features that are not so intuitive, systematic methods can be used. The three basic approaches are, equal frequency (EF), equal width (EW), and clustering which includes methods such as fuzzy clustering, K-means, and Self Organizing Map [12].

Equal width divides a range into k equal sized intervals. Equal frequency divides a range into k intervals where in each interval, the number of instances within is the same. Clustering on the other hand, tries to find best boundaries to partition a range so that most of the values would be away from the boundaries.

Figure 4 shows a histogram of the feature values based on a particular feature on both the speedpaths and the non-speedpaths. The feature is the output load of a specific gate type. In the figure, x axis shows the number of paths, and the y axis shows the feature values, both are normalized to 1. When deciding how to divide the feature values into sub-ranges, there is no easy way to determine the best number of bins to be partitioned and the location of those bins.

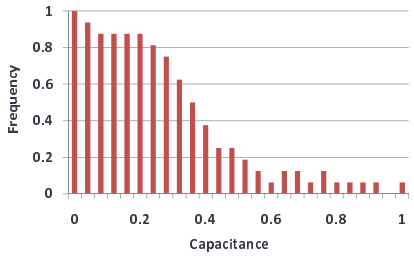

Figure 4: Normalized Capacitance Distribution

We began by identifying which features could be divided using intuition based on domain knowledge. For example, if the load of a gate is a feature, and design engineers believe that gates with a load within 5% of the maximum specified load are more likely to contain delay mismatch and should be treated separately from all other gates, it could be binned accordingly.

For the remaining features we experimented with equal width, equal frequency as well as fuzzy clustering algorithm. We discovered that because most of the features have a distribution similar to the one shown above, equal frequency is not effective. When experimented with equal width and clustering, we discovered that results are similar. Since equal width is more intuitive, we chose equal width as the main discretization method for the experiments.

Given a hypothesis order i, we tried to find the lowest resolution discretization such that there exists one or more ith-order hypotheses consistent with the dataset. This can be achieved by finding the lowest resolution discretization for each of the i feature combination such that with the discretization, the resulting hypothesis is consis-

tent with the dataset. To do this, we implemented a greedy search approach. For each i feature combination $f_1 \ldots f_i$ with corresponding value ranges $R_1 \ldots R_i$, in each step of the greedy search, one of the ith range is selected and the partitioning on the range is increased by one. Take each range R_j. Let the hypothesis based on the current discretization be h_1. Let the hypotheses after increasing its partitioning by one, be h_2. Let n_1 be the number of non-speedpaths on which h_1 appear, and n_2 be the number of non-speedpaths on which h_2 appear. The greedy is to select R_j such that its $n_1 - n_2$ is the largest. This tries to make a consistent hypothesis out of the i features by minimizing the number of partitions across all ranges.

6. MAXIMUM MARGIN METHOD

Given a fixed order, for finding the highest confidence hypotheses in that order, the discretization method tries to find the lowest resolution discretization such that one or more resulting hypotheses in that order do not appear on any non-speedpaths. Although discretization is conceptually easy to understand, we also observe that the greedy method is only a heuristic that conceptually tries to achieve the "lowest resolution."

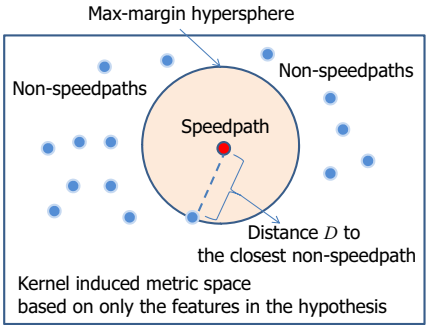

Figure 5: Find the hypothesis with maximum margin

Figure 5 presents an alternative method that is easier to implement and more robust. The method is based on finding a hypersphere with *maximum separation margin* in a *kernel* induced metric space. This concept is borrowed from a Support Vector Machine (SVM) classification algorithm [4] where a kernel function is used to map the input space onto a high dimensional kernel space (formally as Reproducing Kernel Hilbert Space (RKHS)) [4] and find a maximum margin separation hyperplane in that space.

Given a hypothesis, we use the Gaussian kernel function $k(\vec{v}, \vec{v_i})$ to evaluate the similarity between the speedpath $\vec{v}$ and each non-speedpath $\vec{v_i}$ based on *only* the features in that hypothesis. Then, for the hypothesis, we find the maximum margin hypersphere, centered at the speedpath path, to separate the speedpath from all good paths in the kernel induced metric space. Finding the hypersphere is equivalent to find the good path with the closest distance. Given a kernel $k()$, the distance between two feature vectors x, z can be calculated as $k(x,x) - 2k(x,z) + k(z,z)$ [4].

The important thing to note is that the distance is calculated based on only the feature values presented in the hypothesis. If a non-speedpath does not contain any one of the features in the hypothesis, the distance becomes infinity. Furthermore, the distance is calculated in a space induced by the kernel. Hence, if we desire to change the space (i.e. change how feature values are interpreted), we only need to change the kernel function. The work in [5] discusses the flexibility of using such a kernel. Because different feature values can meaning different things, proper interpretation of their values can be built into such a kernel. This provides the flexibility for a user to change how feature values are interpreted, for example weighting one subset of features more than another subset.

With the maximum margin method, the radius (distance) D of the hypersphere is output as the new confidence measure for the hypothesis. With this new confidence measure, on a given hypothesis order, we can rank all hypotheses in that order and output the top N hypotheses. The intuition behind using D as a confidence measure is illustrated in the simple one-dimensional picture in Figure 6.

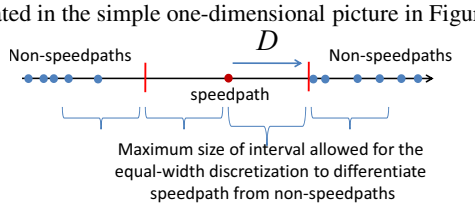

Figure 6: Linking maximum distance to the smallest number of bins in equal width discretization method

The example illustrates what maximum margin separation means in a one-dimensional Euclidean space. We see that the maximum margin D is the distance to the closest non-speedpath. Using D, one can think that it translates into the width in equal width method. This figure shows that the larger the D is, the less the number of bins can be with the equal width method, i.e. lower resolution and hence higher confidence. We see that intuitively, using D as a confidence measure concurs with the confidence definition from the perspective of a concept lattice discussed before. Because of this, we expect the discretization method and the maximum margin methods to give similar results. However, it is also noted that the one-dimensional Euclidean view does not imply the two methods are the same. When considering a multi-dimensional space and a kernel, the maximum margin method may rank hypotheses differently from the less flexible equal width discretization method.

7. EXPERIMENTAL RESULTS

For the experiments, we use the same speedpath dataset used in [2]. The design is a high performance microprocessor fabricated in a 65nm node. The dataset contains 56 speedpaths and 19000 non-speedpaths. These paths are encoded with 74 description features. In each analysis, hypotheses are ranked and derived from a speedpath. For example, by applying the maximum margin method to one speedpath and ranking all its 2nd-order hypotheses, we obtained the result in Figure 7. We see that two hypotheses stand out.

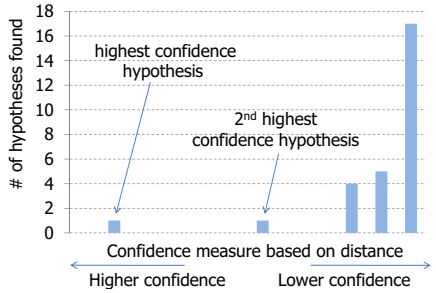

Figure 7: An example of hypothesis ranking (2nd-order)

We applied both methods, the equal width discretization (DS) method and the maximum margin (MM) method, to analyze the 56 speedpaths individually against the 19K non-speedpaths. We tried to find hypotheses in the order from 1 to 4. With the DS method, the lowest resolution discretization for each order returns one or more hypotheses. With the MM method, in each order the top N hypotheses based on their confidence ranking were selected, for $N = 1 \ldots 5$. To compare the MS method and the DS method, all hypotheses found across the 56 paths were considered together. Figure 8 shows the comparison results. The % number shown indicates in each case, how many hypotheses collected by the DS method are also in the set of top N hypotheses found by the MM method. We see that for the 1st-order, all hypotheses found by the DS method

are also found by the the MM method for $N = 1$ (and for $N =$ 2-5). For the 2nd-order, this is true for $N = 3$ and for the 3rd and 4th orders, this is true for $N = 5$. In general, we see that the MM method find all hypotheses found by the DS method with a small N.

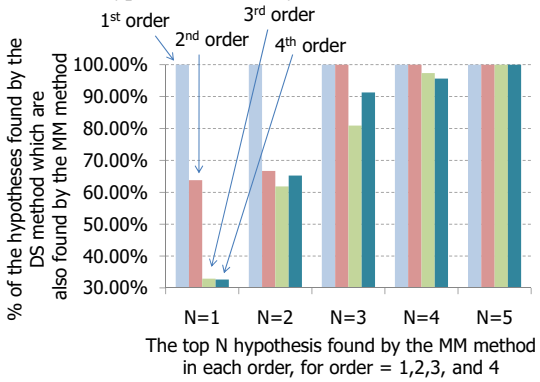

Figure 8: Discretization method vs Maximum margin method

To find potential speedpaths, we devise a voting scheme where top hypotheses found based on individual speedpaths are used to vote for the possibility that a non-speedpath is a potential speedpath. Recall that for a given hypothesis on a speedpath, its confidence is calculated as the distance to the closest non-speedpath. We normalize this distance by the order of the hypothesis and call it the *normalized confidence*. The reason for doing this is that a higher order hypothesis tends to have a larger distance because of the curse of dimensionality [4]. Normalizing the distance avoids giving preference to the higher order hypotheses. Based on normalized confidence, we can then rank all 1st-4th order hypotheses together on a speedpath. From each path, we select the top X hypotheses, and each hypothesis is the closest path in the maximum margin calculation. We add the normalized confidence of the hypothesis to that closest path as a confidence vote for the path to be a potential speedpath.

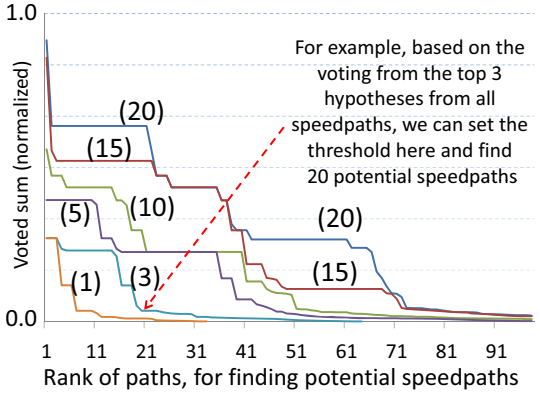

Figure 9: Finding potential speedpaths based on the voted sum from the top (X) hypotheses from all 56 speedpaths, where X is shown in the figure for 1,3,5,10,15, and 20 (using the maximum margin method).

Figure 9 shows the results for X=1,3,5,10,15, and 20. If one fix a threshold on the y-axis and says that all paths above that horizontal line are potential speedpaths, then it is intuitive to see that if more top hypotheses are used from each speedpath in the voting, more potential speedpaths will be found. In this figure, one can also select potential speedpaths by the trend of the curve, for example, if the voted sum from the voting drops to close to zero, then it is a good point to discard the paths behind. Note that by using such a voting scheme, a path is considered as a potential speedpath with higher confidence if it receives votes from more hypotheses that are also with higher confidence. This schemes also reduce the chance of letting noise in the dataset to influence the result.

7.1 Manual analysis of top hypotheses

We manually analyzed several top hypotheses (MM method). The first hypothesis was a top 2nd-order hypothesis shared by 20 speedpaths. The hypothesis consisted of a significantly high delay pushout due to coupling noise in combination with a slower than average cell that had a specific active transistor device type. It is believed the combination of these two features causes a speedpath due to inaccuracy in timing analysis for this device type when accounting for coupling noise. The second hypothesis was a top 3rd-order hypothesis shared by 10 speedpaths. The hypothesis consisted of two slower than average cells, in combination with a significantly high amount of switching activity. The third hypothesis was a top 2nd-order hypothesis shared by 3 speedpaths. This hypothesis consisted of a combination of two slower than average cells unique to the speedpaths. Additionally, unique 2nd-order and 3rd-order hypotheses were found on 21 speedpaths (no sharing with other speedpaths). Similar manual analysis on them found them also reasonable.

8. CONCLUSION

This paper presents a novel hypothesis pruning and ranking approach that analyzes a small number of identified speedpaths against a large number of non-speedpaths for explaining why the speedpaths are special. In our methodology, a hypothesis space is implicitly encoded using a set of features that represents single-order of concerns in modeling, design methodology, and/or process. When using occurrence based features, the hypothesis space is a concept lattice. Hypothesis pruning is to find a cut on this lattice. Confidence is evaluated based on lattice coverage. When using description based features, we present two methods to prune and rank hypotheses. The first method is based on value discretization and the second method is based on finding maximum separation hypersphere. We explain that the two methods are consistent and experimental results also show that they can find common hypotheses. To find potential speedpaths, we develop a voting scheme where a path is considered as a potential speedpath with high confidence if it receives more votes from high-confidence hypotheses found across all speedpaths.

9. ACKNOWLEDGMENTS

We acknowledge the following people at Intel for their contributions to our speedpath identification and root cause analysis in the form of guidance, discussions and implementation: Eli Chiprout, Kip Killpack, and Chandramouli Kashyap.

10. REFERENCES

[1] L.Lee, et al. "On Silicon-Based Speed Path Identification," *Proc. VTS*, 2005.

[2] P. Bastani, K. Killpack, L.-C. Wang, and E. Chiprout. "Speedpath prediction based on learning from a small set of examples," *Proc. DAC*, 2008.

[3] K. Killpack, C. Kashyap, E. Chiprout, "Silicon Speedpath Measurement and Feedback into EDA flows," *Proc. DAC*, 2007.

[4] N. Cristianini, and J. Shawe-Taylor, "An Introduction to Support Vector Machine," *Cambridge University Press*, 2002.

[5] P. Bastani, N. Callegari, L. Wang, and M. Abadir. "Statistical diagnosis of unmodeled systematic timing effects," *Proc. DAC*, 2008.

[6] Chengqi Zhang and Shichao Zhang. "Association Rule Mining, Models and Algorithms," *Lecture Notes in Computer Science* Vol. 2307, Springer 2002.

[7] S. Brin, R. Motwani, et al. "Dynamic itemset counting and implication rules for market basket data," *Proc. ACM SIGMOD*, 1997, pp. 255-264.

[8] D.Barton, P. Tangyunyong, J. Soden, A. Liang, and e. a. F. Low. "Infrared light emission from semiconductor devices," *ISTFA Proc.*, 1996.

[9] R. Agrawal, T. Imielinksi, and A. Swami. "Mining association rules between sets of items in large databases," *Proc. ACM SIGMOID*, 1993, pp. 207-216.

[10] D. Burdick, M.Calimlim, and J. E. Gehrke. "MAFIA: A Maximal Frequent Itemset Algorithm for Transactional Databases," *International Conference on Data Engineering*, 2001, pp. 443-452.

[11] H. Mannila, et al. Discovery of Frequent Episodes in Event Sequences," *Data Mining and Knowledge Discovery*, Vol 1, (3), 1997, pp. 259-289.

[12] M. Vannucci and V. Colla. "Meaningful discretization of continuous features for association rules mining by means of a som," *ESANN*, 2004, pp. 489-494.