Employing Variable Cross-Reference Prediction and Iterative Dispatch to Raise Dynamic Branch Prediction Accuracy

Po-Jen Chuang*, Yue-Ter Liau, Young-Tzong Hsiao and Yu-Shian Chiu

Department of Electrical Engineering, Tamkang University, Tamsui, Taiwan 251, R.O.C.

Abstract

To improve branch prediction accuracy for the two-level adaptive branch predictor, two schemes – dealing respectively with the prediction and dispatch parts, are presented in this paper. *The proposed VCR prediction scheme* is able to achieve desirable prediction accuracy, with reasonably low time complexity and no extra hardware cost, by variably cross-referring traces in the PHT to make predictions. *The Iterative dispatch approach* utilizes the PHT history to do dispatching for an additional layer of pattern history which helps providing more information for making better predictions. To attain desirable prediction accuracy at reduced cost, a combined predictor formed by the proposed VCR scheme and the optimal PPM algorithm is also considered. Extensive trace-driven simulation runs have been conducted to evaluate the performance of our proposed schemes and other predictors. As the results indicate, our proposed schemes compare favorably in most of the situations in terms of prediction accuracy.

Key Words: Branch History, Dynamic Branch Prediction, Performance Evaluation, Prediction Accuracy, Trace-driven Simulation, Two-level Adaptive Branch Predictor

1. Introduction

Branch prediction is important in maintaining processor performance. As high prediction accuracy ensures better performance, raising prediction accuracy becomes essential [1]. A number of new schemes (such as [2–9]) are built to lift up prediction accuracy for the two-level adaptive branch predictor [10]. These prediction schemes, such as the 2-bit counter [2], the Markov predictor [5, 11], the PPM algorithm [5], and the gshare [3], agree [4], bi-mode [7], YAGS [8] and DHLF predictors [9], deal with either the prediction part or the dispatch part or with both parts of the two-level predictor, each with its own advantages and disadvantages. For instance, *the 2-bit counter* and *the Markov predictor* – though generally used in branch prediction – are unable to provide adequate prediction accuracy for high performance processors. *The PPM algorithm* is able to yield remarkable prediction accuracy but the involved complexity is considerably high. The gshare, agree, bi-mode and YAGS predictors aim to reduce the so-called pattern history table (PHT) interference (happening when the outcome of a branch interferes with the subsequent prediction of a completely unrelated branch) in order to lift up the prediction accuracy. The DHLF predictor tries to dynamically find the amount of history that performs best for each code and input data at execution time.

The goal of this paper is to improve branch prediction accuracy at reasonably low cost. A new prediction scheme dealing with the prediction part is first established. The proposed scheme is called the *variable cross-reference* (VCR) prediction scheme because it is able to make desirable predictions (in terms of prediction accuracy, time complexity and hardware cost) by variably cross-referring to traces in the PHTs. Different from previous schemes, the VCR scheme involves the *loop his-*

*Corresponding author. E-mail: pjchuang@ee.tku.edu.tw

tory which, existing in most programs and easily observable in small programs, is helpful in elevating the prediction accuracy. Also presented is a*n iterative* dispatch approach which involves some structural changes in the dispatch part of the branch prediction. In our new design, the PHT functions as an intermediary index tag stage and the branch history in each PHT entry is used as an index tag indexing to an entry in the corresponding sub-PHT at an additional stage. The branch history in the indexed sub-PHT entry is then used for prediction. In this way the iterative approach helps divide information into more classes to reduce PHT interference and to enhance prediction accuracy accordingly. Besides, a new combined predictor formed by bringing the proposed VCR scheme and the optimal PPM algorithm together is also introduced to attain desirable performance with reduced complexity.

Extensive trace-driven simulation runs using the SPEC CINT95 benchmarks [12] have been conducted to evaluate and compare the performance of our proposed schemes and other related schemes. The collected results show that *the proposed VCR scheme* yields lower misprediction rates, i.e., higher prediction accuracy, than the 2-bit counter and the agree predictor – schemes dealing with the prediction part, and under certain conditions it produces even superior performance over the optimal PPM algorithm at much less cost. *The proposed iterative dispatch approach* is shown to outperform the gshare and DHLF predictors – schemes dealing with the dispatch part. When compared with schemes dealing with both the prediction and dispatch parts, such as the bi-mode and YAGS predictors, the VCR scheme depicts superior performance over the two predictors (at no extra hardware cost), except in some situations of a certain benchmark where the performance of the VCR scheme may not be as satisfactory as that of the two predictors (whose performance have been obtained with extra hardware and cost). It is also shown that the VCR prediction scheme can readily work with schemes dealing with the dispatch part, and the iterative dispatch approach can be handily incorporated with schemes dealing with the prediction part – to enhance performance especially for schemes with lower prediction accuracy. Simulation results also exhibit that the combined PPM-VCR predictor is able to make optimal prediction at lowered cost.

2. Background and Previous Works

The two-level adaptive branch predictor [10] uses two levels of branch history information to make predictions (Figure 1). Prediction is made according to the branch behavior for *the history of the last k branches encountered* and *the last s occurrences of the specific pattern of these k branches*. Two major data structures, the branch history register (BHR) and the pattern history table (PHT), are employed to record the two levels of information. The information is collected at run-time by updating the contents of the BHR and the bits in the entries of the PHT. The update is made following the branch outcomes.

As depicted in the Figure 1, the structure of a two-level adaptive branch predictor can be divided into two parts – the *dispatch* part (where the information is dispatched from the BHR to the PHT through some dispatch mechanism) and the *prediction* part (where the content of the addressed PHT entry is used to predict the branch outcome through some prediction decision function).

2.1 Schemes Dealing with the Prediction Part

The 2-bit Counter

The 2-bit counter approach [2] uses a 2-bit saturating

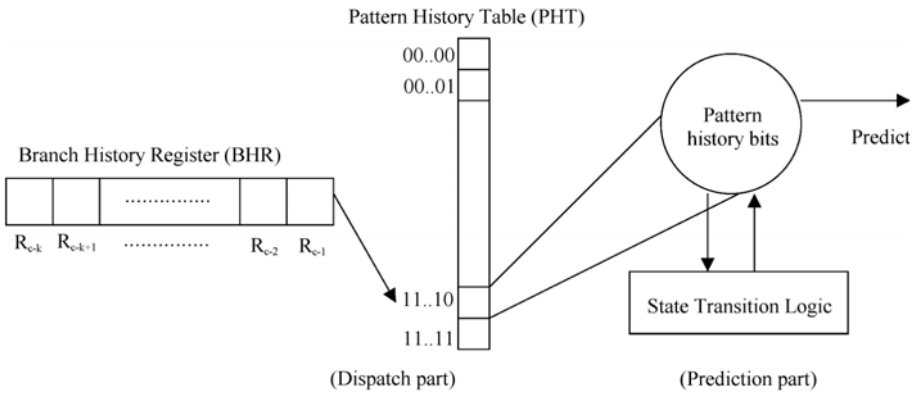

Figure 1. A two-level adaptive branch predictor.

up-down counter to predict which path the branch will take according to the PHT. In the two-level adaptive branch prediction, the 2-bit counter scans the data of a certain entry in the PHT. It is increased or decreased by 1 when a branch is taken or not taken. If the counter value is 2 or greater, a branch is predicted "taken", otherwise it is predicted "not taken". The 2-bit counter involves not much complexity, but its prediction accuracy can not meet the need of high performance processors.

The Markov Predictor

A Markov predictor of order j predicts the next bit based on the j immediately preceding bits and the simple Markov chain [11]. As specified in [5], the transition probabilities are proportional to the observed frequencies of a 1 or 0 that occurs given that the predictor is in a particular state (the bit pattern is associated with the state). The predictor builds the transition frequency by recording the number of times a 1 or 0 occurs in the (j+1)th bit that follows the j-bit pattern. The chain is thus built for prediction. To predict a branch outcome, the predictor simply uses the j immediately preceding bits (outcomes of previous branches) to index a state and predicts the next bit to correspond to the most frequent transition out of that state. Note that the prediction accuracy of the Markov predictor can be impaired with too long or too short PHTs: If the PHT is too long, the outdated information may affect prediction accuracy; if too short, it may cause zero frequency counts, result in incomplete Markov chains, and thus provide inadequate information for prediction. Meanwhile it is also unlikely for the predictor to predict with equal frequency counts.

The Prediction by Partial Matching (PPM) Algorithm

The zero frequency situation of the Markov predictor can be improved by the Prediction by Partial Matching (PPM) algorithm [5]. The basis of a PPM algorithm of order m is a set of m+1 Markov predictors. The PPM uses the m immediately preceding bits to search a pattern in the highest order Markov predictor. If the search succeeds (i.e., if the pattern with a non-zero frequency count appears in the input sequence), the PPM will predict the next bit using the mth order Markov predictor. If the pattern is not found, the PPM will use the m-1 immediately preceding bits to search the next (m-1)th order Markov predictor. Whenever a search fails, the PPM reduces the pattern by one bit and uses it to search in the next lower order Markov predictor until the pattern is found or until searching in the 0th order Markov predictor. Thus, by recording and updating information through variably referring the data in the PHT, the PPM algorithm is able to improve the zero frequency situation in a Markov predictor. The prediction accuracy achieved by the PPM algorithm is remarkable, but the involved overhead (to keep necessary information) is also conspicuous.

The Agree Predictor

Since the number of PHT entries is finite, it is likely that two unrelated branches in the instruction stream are mapped to the same PHT entry by the predictor's indexing function. The situation is known as the PHT interference [4] because the outcome of one branch will interfere with the subsequent prediction of another completely unrelated branch. The Agree predictor [4] attempts to reduce the PHT interference by taking advantage of the biased behavior. It attaches a biasing bit to each branch in the Branch Target Buffer (BTB) according to the branch direction just before the biasing bit is written into the BTB. The biasing bit, which predicts the most likely outcome of the branch, will stay the same until the branch is replaced in the BTB by another branch. The PHT records "agreeing" or "not-agreeing" the biasing bit and a 2-bit counter is used to predict whether or not the branch will go in the direction indicated by the biasing bit. The counter will be incremented if the branch's direction agrees with the biasing bit or be decremented if it disagrees.

2.2 Schemes Dealing with the Dispatch Part

The Gshare Predictor

To reduce the PHT interference, the gshare [3] predictor tries to use the PHT entries more effectively by XOR-ing the k-bit BHR with the lower k bits of the branch address to generate the index into the PHT. The introduction of address bits into the index effectuates useful distribution across all PHT entries, but the resulted interference reduction is relatively limited in contrast to indexing by the BHR [8].

The DHLF Predictor

As each code requires a specific amount of branch history to give the best results, the DHLF (Dynamic History-Length Fitting) predictor [9] tries to dynamically find the amount of history that performs best for each code and input data at execution time. The DHLF predictor works on the basis of monitoring the mispredictions

during program execution and changing the history length accordingly. During the execution of the program the number of mispredictions for each interval (consisting of a fixed number of consecutive dynamic branches) is computed using a fixed history length. At the end of each interval the history length to be used at the next interval is determined based on the current number of mispredictions and the minimum number of mispredictions encountered so far. If the current number of mispredictions is less than or equal to the minimum number of mispredictions, the history length is not changed for the next interval. If it is greater, the history length will be changed to the one corresponding to the minimum number of mispredictions or toward to it by increasing or decreasing (by one) the current history length.

2.3 Schemes Dealing with both the Prediction and Dispatch Parts

The Bi-mode Predictor

As it is uncertain if the behavior of a branch will correspond to its bias when the branch is first introduced to the BTB, the bi-mode predictor [7] helps eliminate mischosen fixed biases by dynamically choosing the branches' biases. There are three PHTs in the bi-mode predictor: The choice PHT, the taken direction PHT and the not-taken direction PHT. A branch is first indexed by its address to the choice PHT. The choice PHT then chooses one of the direction PHTs that is indexed by BHR xor-ed with the branch address. The selected direction PHT makes the final prediction and gets updated. The choice PHT will also be updated unless it gives a prediction contradicting the branch outcome while the selected direction PHT gives the correct prediction. The choice PHT functions like the biasing bit in the agree predictor, only that it can dynamically choose the bias (the biasing bit in the agree predictor is fixed).

The YAGS Predictor

To reduce the amount of unnecessary information in the PHT, the YAGS (Yet Another Global Scheme) predictor [8] stores in the direction PHTs only the instances when the branch does not comply with its bias. To identify those instances in the direction PHTs, tags (the least significant bits of the branch address) are added to each entry and are referred to as direction caches. When a branch occurs in the instruction stream, the choice PHT is accessed. If it indicates "taken," the "not taken" cache will be accessed to check if the prediction does not agree with the bias (a special case). If there is a miss in the "not taken" cache, the choice PHT is used for prediction. If there is a hit in the "not taken" cache, it provides the prediction. A similar process is activated in the "taken" cache when the choice PHT indicates "not taken." The choice PHT is addressed and updated like the choice PHT in the bi-mode predictor. The "not taken" cache will be updated when a prediction from it is used or when the choice PHT indicates "taken" while the branch outcome is "not taken" (the mentioned special case). The same process applies to the "taken" cache.

3. The Proposed Schemes

To attain desirable prediction accuracy with reduced overhead for the two-level adaptive branch prediction, we propose in this paper a new prediction scheme and an iterative dispatch approach. The proposed prediction scheme is also combined to work with the PPM algorithm to maintain optimal prediction accuracy at decreased cost.

3.1 The Variable Cross-Reference Prediction Scheme

Different from previous schemes dealing with the prediction part, our proposed scheme involves the *loop history* which, existing in most programs and easily observable in small programs, can execute a large quantity of branch instructions. We believe recognizing the loop property can be of significant use in elevating the accuracy of branch prediction.

In the PHT, i.e., the second level of the two-level dynamic predictor, a prediction scheme is used to predict the outcome of a branch according to the sequence of branch outcomes (taken or not taken – represented by a single bit 1 or 0) in the addressed PHT entry. Our proposed prediction scheme operates as follows. The sequence of outcomes in the addressed PHT entry is first divided into two parts that are equal in length, i.e., number of outcomes. (If there is an odd number of outcomes in the PHT entry, the "least recent" outcome could be ignored.) The two parts are then cross-referred to see if they match with each other. If both parts are completely the same, we assume the same outcome will repeat again (according to the loop history) and thus predict the coming branch outcome to be the first outcome in the first part (also the first outcome in the second part). If the two parts do not match, ignore the next two "least recent"

outcomes in the sequence and again divide the remaining outcomes into two equal-length parts. Check the two parts: If they are the same, predict the outcome to be the first outcome of the first part; if not, repeat the above *referring* process until a match for the two parts is located. In case no match is found when the number of referred outcomes is reduced to only one in each part, employ some other scheme (such as the 2-bit counter or 0th Markov predictor) to assist the prediction.

Figure 2 demonstrates the operation of the proposed scheme. The sequence of branch outcomes in the addressed PHT, assumed to be $R_{c-s}R_{c-s+1} \cdots R_{c-1}$ with s (the number of outcomes in each PHT entry) being an even number, is first divided into two equal-length parts, i.e., $R_{c-s}R_{c-s+1} \cdots R_{c-s/2-1}$ and $R_{c-s/2}R_{c-s/2+1} \cdots R_{c-1}$. The two parts are then compared. If they match each other, that is, if

$$R_{c-s} = R_{c-s/2},\ R_{c-s+1} = R_{c-s/2+1},\ \cdots \text{ and } R_{c-s/2-1} = R_{c-1},$$

the coming branch outcome is predicted to be $R_c = R_{c-s}$. If they do not match each other, ignore the two "least recent" outcomes R_{c-s} and R_{c-s+1} and divide the remaining outcomes into two new parts $R_{c-s+2}R_{c-s+3} \cdots R_{c-s/2}$ and $R_{c-s/2+1}R_{c-s/2+2} \cdots R_{c-1}$. Check again. If the two parts match, that is, if

$$R_{c-s+2} = R_{c-s/2+1},\ R_{c-s+3} = R_{c-s/2+2},\ \cdots \text{ and } R_{c-s/2} = R_{c-1},$$

our prediction will be $R_c = R_{c-s+2}$. If they do not match, ignore the next two "least recent" outcomes R_{c-s+2} and R_{c-s+3}, and again divide the remaining outcomes into $R_{c-s+4}R_{c-s+5} \cdots R_{c-s/2+1}$ and $R_{c-s/2+2}R_{c-s/2+3} \cdots R_{c-1}$. If the two parts match each other, the coming branch outcome is predicted to be $R_c = R_{c-s+4}$. If they do not match, repeat the same comparison process (by ignoring the next two "least recent" outcomes at each comparison attempt). Prediction can be made whenever a match is found by this variable cross-reference. (Note that in our scheme the two parts under comparison are with variable, not fixed, lengths.) If eventually only one outcome is left in each part and they still do not match each other, the prediction is handed over to a 2-bit counter or a 0th Markov predictor. Featured by such a variable cross-reference process – which needs no extra hardware at all, the proposed prediction scheme is called the Variable Cross-Reference (VCR) scheme.

Figure 3 further illustrates the VCR scheme. As it shows, there are 11 bits (01010101101) in the addressed PHT entry. Ignore the most significant bit, i.e., the "least recent" branch outcome, and divide the remaining 10 bits into two equal-length parts 10101 and 01101. Compare the two parts. As there is no match, ignore the next two "least recent" bits and cross-refer the newly divided two parts 1010 and 1101. Since there is no match, ignore

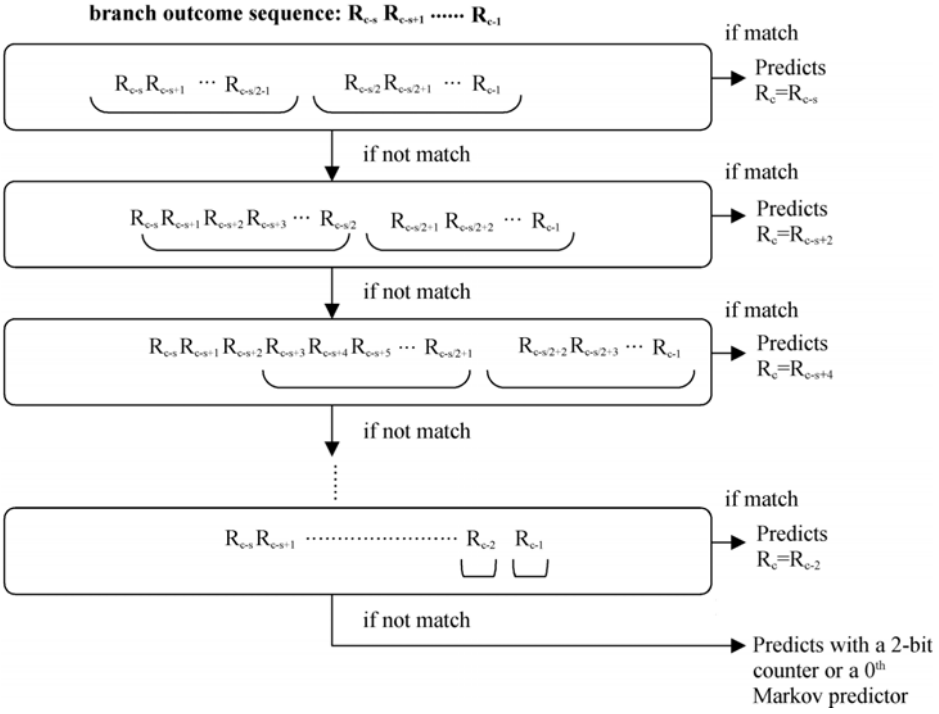

Figure 2. Prediction flowchart of our VCR scheme.

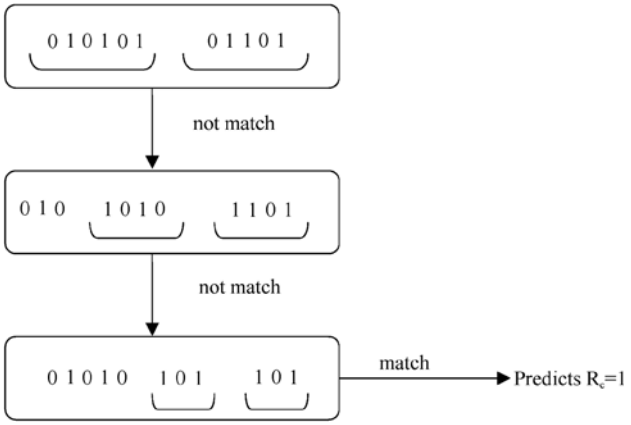

Figure 3. Example of our VCR scheme.

the next two "least recent" bits and divide the remaining bits (101101) into two new parts 101 and 101. With the two parts matching each other, we thus predict the coming branch outcome to be '1' (i.e., the most significant bit in both parts).

3.2 The Iterative Dispatch Approach

To enhance branch prediction accuracy, we also present a new approach that involves some structural changes in the dispatch part. In our new design, the PHT functions as an intermediary index tag stage and the branch history in each PHT entry is used as an index tag indexing to an entry in the corresponding sub-PHT at an additional stage. The branch history in the indexed sub-PHT entry is then used for prediction. For instance, if the bits in the branch history register (BHR) are $R_{c-k}R_{c-k+1} \cdots R_{c-1}$, it will address an entry in the PHT. However, the history bits in the addressed PHT entry are used not for prediction but as an index tag indexing the corresponding sub-PHT at the additional stage. Suppose the length of the PHT entry is m bits, we can index to a corresponding sub-PHT with 2^m entries in the same way as indexing the BHR to the PHT, and have 2^{m+k} sub-PHT entries in total. Predictions are then made by referring to the bits in the sub-PHT entries. The BHR, PHT and sub-PHTs will update their contents – after the outcome of each branch turns out – to lead the sub-PHTs for future predictions. In this way the "traces of traces" are referred to as a kind of information to improve prediction accuracy. That is, this iterative dispatch approach utilizes the PHT history to do dispatching for an additional layer of pattern history and the information can hence be further divided into 2^m classes, providing more information and less PHT interference than employing only the traditional PHTs in making predictions.

Figure 4 exhibits the structure of our proposed iterative dispatch approach on a two-level adaptive branch predictor. As shown here, the PHT exists between the BHR and the sub-PHTs as an intermediary index tag stage. Data in the BHR are first classified by the intermediary index tag stage (the PHT) which is much shorter than the entire sub-PHTs. Based on the behavior of the branch outcomes in a sub-PHT entry, predictions are then made. (Note that due to such a structural change, the number of

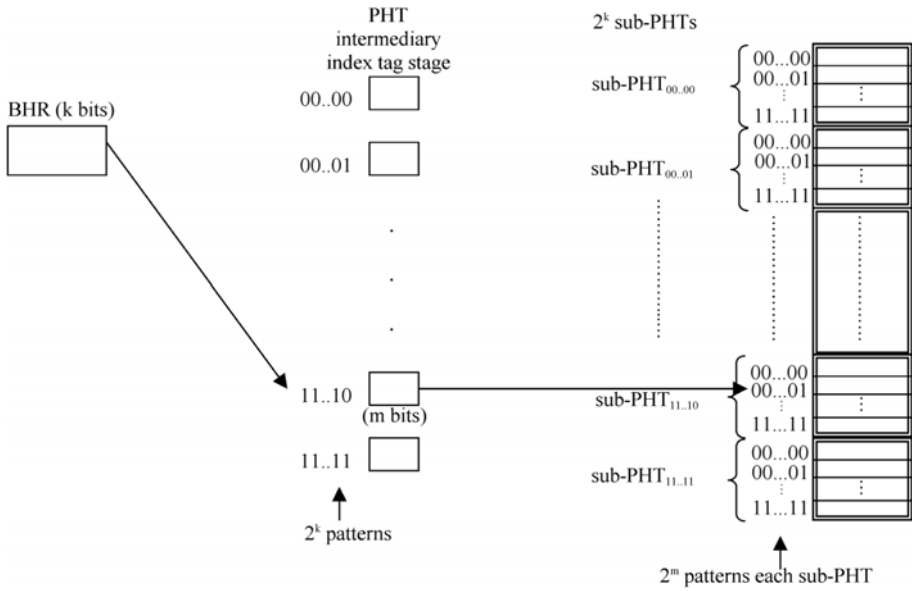

Figure 4. Structure of the iterative dispatch.

table entries increases and so does the needed warm-up time.) Assume the length of the BHR is k bits. It can arddress the PHT with 2^k tags and each tag indexes to an entry of the corresponding sub-PHT with 2^m entries. Before a prediction is made, an entry (i.e., an index tag) in the PHT is addressed according to the bits $R_{c-k}R_{c-k+1} \cdots R_{c-1}$ in the BHR. The index tag then addresses an entry in the corresponding sub-PHT (say sub-PHT$_x$, $0 \leqq x \leqq 2^k-1$). A prediction is finally made by referring to the bits in the indexed sub-PHT entry. If the branch result is R_c, it is then shifted into the BHR and the bits in the BHR are updated as $R_{c-k+1}R_{c-k+2} \cdots R_{c-1}R_c$. The PHT entry and the indexed sub-PHT entry are also updated by the bit R_c.

As mentioned, the iterative dispatch approach is designed to assist predictors in elevating prediction accuracy. Take the proposed VCR predictor as an example. When encountered with the sequence of branch outcome 10110101, the PPM algorithm, Markov predictor and 2-bit counter will predict the next bit to be 1, while the VCR scheme will predict it to be 0. In fact, the sequence displays a loop history of 1011 with an extra 0 in the middle – a situation which may lead the VCR scheme to wrong predictions. For situations like this, the iterative dispatch approach can be brought in to help as demonstrated in Figure 5. Assuming $m = 1$, we first initialize the intermediary index tag stage to be 0 and the sequence of branch outcome to be 10110101. After the BHR encounters the first two bits 10 and makes the prediction, shift the branch outcome (i.e., 1) into both entry 10 of the PHT and entry 0 of the corresponding sub-PHT (i.e., sub-PHT$_{10}$). Now the newly updated information of both the addressed PHT entry and the indexed sub-PHT entry becomes 1. Then based on the branch outcome for the next 2 bits 01, the content of the PHT entry 01 and the indexed sub-PHT$_{01}$ entry are also updated with the outcome 1. As the original content of the PHT entry 01 is 0, we update entry 0, instead of entry 1, of sub-PHT$_{01}$. The content is now updated to 1. Such a shifting and updating process is repeated following every two bits of the sequence until 01 is again shifted into the BHR. With the content of the PHT entry 01 being updated to 1, entry 1 of sub-PHT$_{01}$ will be updated accordingly. When the end bits of sequence 01 is shifted into the BHR, the referred PHT will be entry 0 of sub-PHT$_{01}$ because the content of the updated PHT entry 01 is 0 (due to the branch outcome after the last 2-bit sequence 01 being 0). As the content of entry 0 of sub-PHT$_{01}$ is 1, the VCR scheme will thus predict the branch outcome to be 1, like the other schemes. The proposed iterative dispatch approach is shown through simulation results to work not only for the VCR scheme but also for other schemes, especially for schemes with lesser performance, such as the 2-bit counter (to be discussed in later sections).

3.3 The PPM-VCR Predictor

It has been maintained that a single predictor able to record larger quantity of trace data proves to be the most effective predictor, but the performance of a combined predictor is usually better than that of a single predictor at the same hardware cost [6]. A combined predictor is composed of at least two single predictors which simultaneously make predictions when a branch occurs. A selector is employed to evaluate the prediction performance of each (single) predictor. Based on previous outcomes, the selector will check and choose the predictor most likely to make the correct prediction for the current

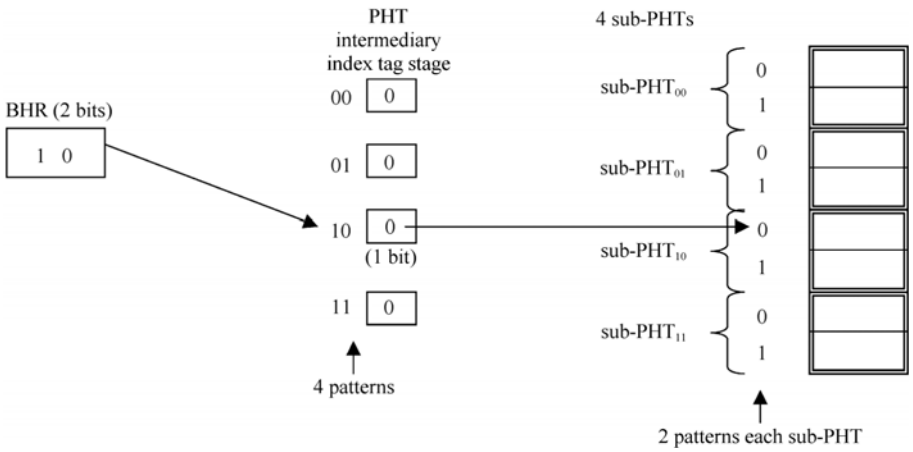

Figure 5. Example of the iterative dispatch.

branch. Branch prediction outcomes made by each predictor are recorded in a 2-bit counter that updates itself with each new result. Following the continually updated data, the 2-bit counter is able to decide a better predictor and select it for predicting the incoming branch.

As mentioned, the PPM algorithm achieves remarkable prediction accuracy at the cost of substantial complexity, whereas the proposed VCR scheme depicts quite satisfying prediction accuracy with much less complexity. Indeed the VCR scheme performs even better than the optimal PPM algorithm under certain conditions, such as during the warm-up period and with shorter PHTs. (When the PHTs are short, "referring" tends to yield the same probability for "taken" and "not taken" of the branch, making the PPM algorithm unable to make correct predictions. The VCR scheme is free of such limitations. It can make fast and correct predictions whenever the cross-reference finds a match.) We are thus interested in combining the two prediction schemes together to see if performance of the combined predictor can be strengthened with reduced complexity. For the combined PPM-VCR predictor, we choose not to use a 2-bit counter as the priority selector considering the performance and overhead of the two prediction schemes. Instead, the priority selector for the PPM algorithm is expanded into an (n-1)-bit counter (the length of the PHT is 2^n) and that for the VCR scheme is set to be a 1-bit counter. When making predictions, employ the PPM algorithm to do the job if it displays larger priority; otherwise, employ the VCR scheme. The priority selectors are updated with each new prediction result for future predictions.

4. Performance Evaluation

Extensive trace-driven simulation runs using four SPEC CINT95 benchmarks [12] – vortex, perl, m88ksim and gcc – are conducted to evaluate performance of the proposed schemes and other schemes. The SimpleScalar Toolset [13] is used to generate and capature address traces. Prediction accuracy is the performance measure of interest, but for more informative presentation, misprediction rates (one minus prediction accuracy) are presented in the following discussions, as in [7]. Note that the Markov predictor is not included in this simulation because of its prediction limitations for zero and equal frequencies, and the PPM algorithm adopted here is the optimal one, i.e., with its best performance. The misprediction rates are collected under various BHR lengths (2~7 bits) and PHT lengths (8~256 bits). However, we present only the misprediction rates collected under PHT lengths = 8~256 bits with BHR length = 7 bits, and under BHR lengths = 2~7 bits with PHT length = 256 bits, due to limited space.

Comparison among predictors dealing with the prediction part

Depicted in Figure 6(a) are the misprediction rates for the 2-bit counter, the PPM algorithm, the agree predictor and the VCR scheme resulting from running the four SPEC CINT benchmarks under PHT lengths = 8~256 bits with BHR length = 7 bits (a similar performance trend can be found with any of the BHR lengths). As exhibited, the performance of our VCR scheme yields constantly lower misprediction rates than the 2-bit counter and the agree predictor. In fact, the proposed scheme outperforms even the optimal PPM algorithm at shorter PHTs, such as 8 bits, in some benchmarks. This is because with shorter PHTs, "referring" for the PPM algorithm tends to yield the same probability for "taken" and "not taken" of the branch, making the algorithm unable to predict correctly. By contrast, misprediction rates for the PPM algorithm at longer PHT lengths are apparently lower than that for the 2-bit counter, the agree predictor and the VCR scheme. However, it should be pointed out that the high performance of the PPM algorithm is achieved at substantial cost because our simulation adopts the largest predictable PHT length – 256 bits, which makes the PPM algorithm use the 255th PPM predictor or 256 Markov predictors to predict branches.

Figure 6(b) presents the misprediction rates of these schemes under BHR lengths = 2~7 bits with PHT length = 256 bits. The results also exhibit a similar trend as what is shown in Figure 6(a), i.e., our VCR scheme always yields lower misprediction rates than the 2-bit counter and the agree predictor, and sometimes even matches the optimal PPM algorithm.

Comparison among predictors dealing with the dispatch part

Performance of the gshare predictor, the DHLF predictor and our iterative dispatch approach is illustrated in Figure 7(a) where the misprediction rates are collected under various PHT lengths with BHR length = 7 bits. The figures show that misprediction rates obtained from the four benchmarks are always lower for our iterative dispatch approach than for the other two schemes. This is

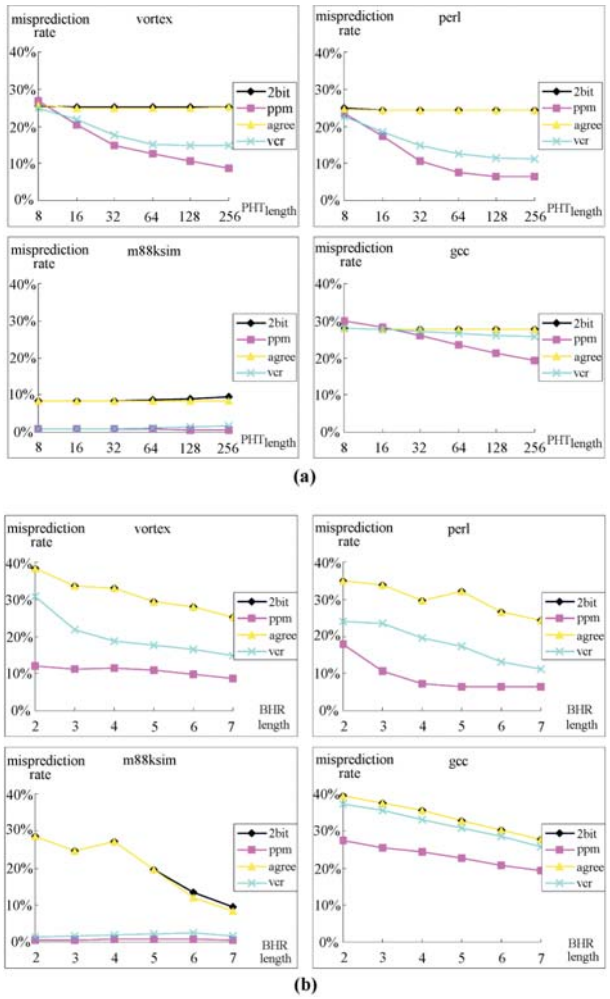

Figure 6. Misprediction rates for schemes dealing with the prediction part.

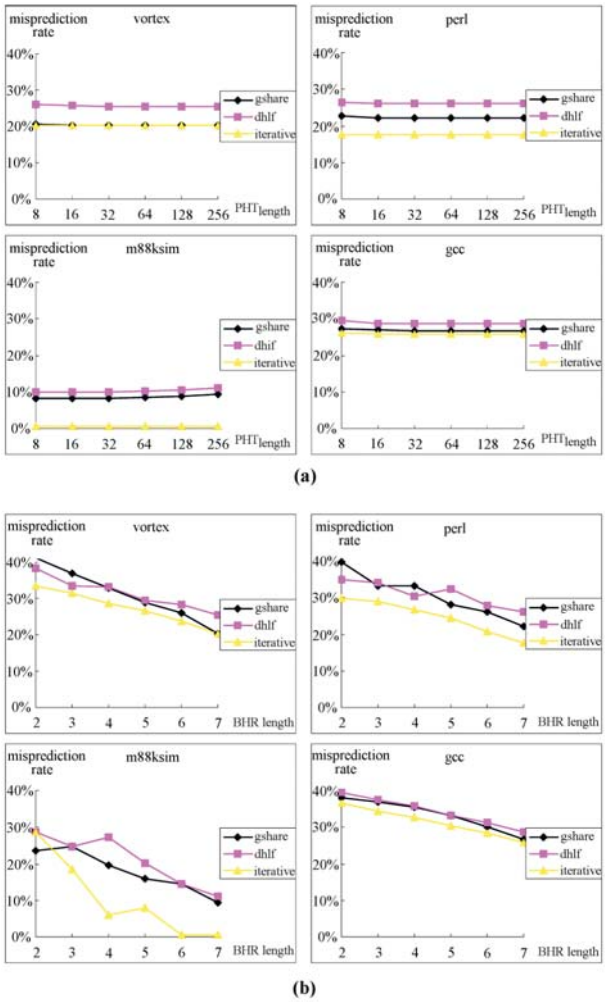

Figure 7. Misprediction rates for schemes dealing with the dispatch part.

because the iterative dispatch approach utilizes the PHT history to do dispatching for an additional layer of pattern history and by dividing the information into more classes it is able to provide more information and reduce PHT interference in making predictions. Similar results can also be learned from Figure 7(b) which depicts the misprediction rates under various BHR lengths with PHT length = 256 bits (collected simulation results show that performance of the three schemes follows almost the same trend with any of the PHT lengths.)

Comparison between our VCR scheme and the bi-mode and YAGS predictors

Figure 8 provides the misprediction rates for the bi-mode predictor, the YAGS predictor (predictors that deal with both the prediction and dispatch parts) and our VCR scheme under (a) various PHT lengths with BHR length = 7 bits and (b) various BHR lengths with PHT length = 256 bits. As we can see, the overall performance of the VCR scheme excels that of the other 2 schemes, except in benchmark gcc where the VCR scheme falls behind the bi-mode predictor and also the YAGS predictor in some situations – with slight differences. Actually in more practical situations, such as with longer PHT or BHR lengths, the performance of our VCR scheme compares favorably, at no extra cost, to the other two predictors which need more extra hardware and cost, such as two extra direction PHTs and doubled predictions in the choice and direction PHTs for the bi-mode predictor.

Effect of incorporating our schemes with other predictors

The VCR scheme, as mentioned, is ready to work with schemes that deal with the dispatch part, while the

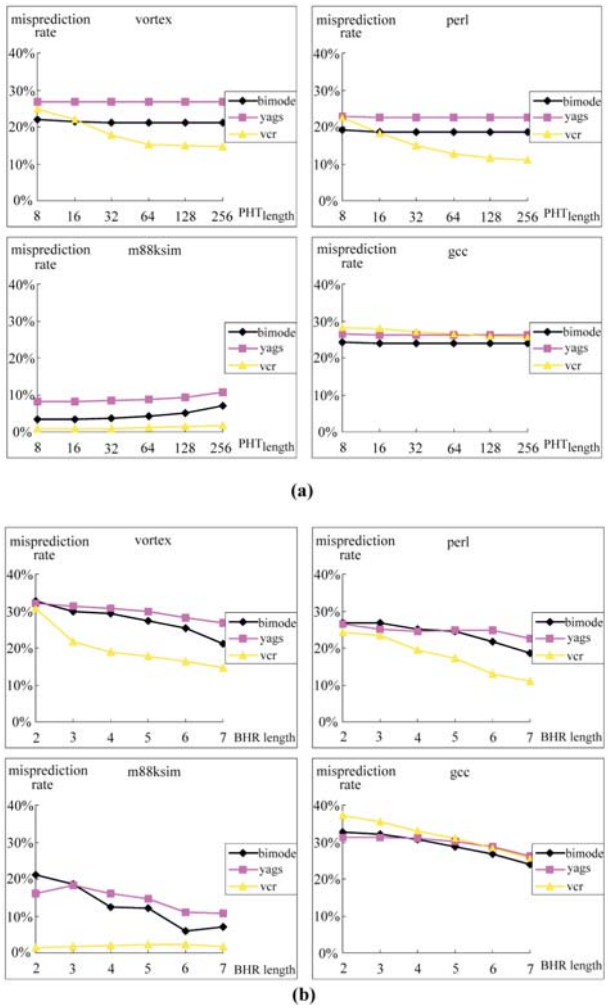

Figure 8. Misprediction rates for our VCR scheme and schemes dealing with both parts.

iterative dispatch approach can be handily incorporated with schemes dealing with the prediction part – to enhance prediction accuracy especially for schemes with lesser performance. The effect (performance gain) of incorporating the proposed schemes with other approaches is also presented by simulation results which indicate different degrees of performance gain for different incorporated schemes. To save space, Figure 9 presents only the performance of the 2-bit counter (scheme with low prediction accuracy) and the PPM algorithm (scheme with high prediction accuracy) – with and without our iterative dispatch approach. Figures 9(a) and 9(b) show significant performance gain for the 2-bit counter with the iterative dispatch approach, while Figures 9(c) and 9(d) exhibit a slightly enhanced PPM algorithm with the same dispatch approach. The varied performance gain for the two predictors when incorporated with the iterative dispatch approach results from the fact that the PPM algorithm is already an optimal prediction scheme by itself while the 2-bit counter alone yields relatively low overall prediction accuracy and hence leaves more room for improvement.

Performance of the PPM-VCR predictor

As mentioned in Section 3, the VCR scheme works better and faster than the PPM algorithm under shorter PHTs while the PPM algorithm performs more desirably at longer PHTs. To make the most of their advantages, we bring the two schemes together to form a combined predictor and conduct performance comparison under various PHT lengths with BHR length = 7 bits and various BHR lengths with PHT length = 256 bits. The result shows the PPM-VCR predictor performs as well as or better than the PPM algorithm alone. This is especially significant when the potentially reducible complexity due to the VCR scheme is taken into account. (Figure presentation is omitted due to limited space.)

Discussions

It is interesting to see from the performance comparison in Figures 6 and 8 that the proposed VCR scheme outperforms the existing schemes quite obviously in some benchmarks, like m88ksim, and less obviously in some other benchmarks, like gcc. Recall that the VCR scheme, which distinguishes itself from previous schemes by taking advantages of the loop history, divides the sequence of outcomes in the addressed PHT entry into two parts that are equal in length. The two parts are then cross-referred to see if they match each other. If they do, the coming branch outcome is predicted to be the first outcome in either part; if they don't, repeat the same process by ignoring the two "least recent" outcomes at each comparison attempt until a match is located. If a match is located and prediction is made at earlier comparison attempts, the matched two parts are of larger lengths (indicating the program's history carries "bigger loop" property), and vice versa. As prediction may be made upon matched parts of different lengths, we are interested to see the percentages, along with the misprediction rates, of branch prediction made upon varied matched parts. For any benchmarks, it is found through simulation that misprediction rates are always lower (i.e., prediction accuracies are higher) when predictions are made at earlier comparison attempts (i.e., upon longer matched parts – parts with more bits). Take m88ksim and gcc as an exam-

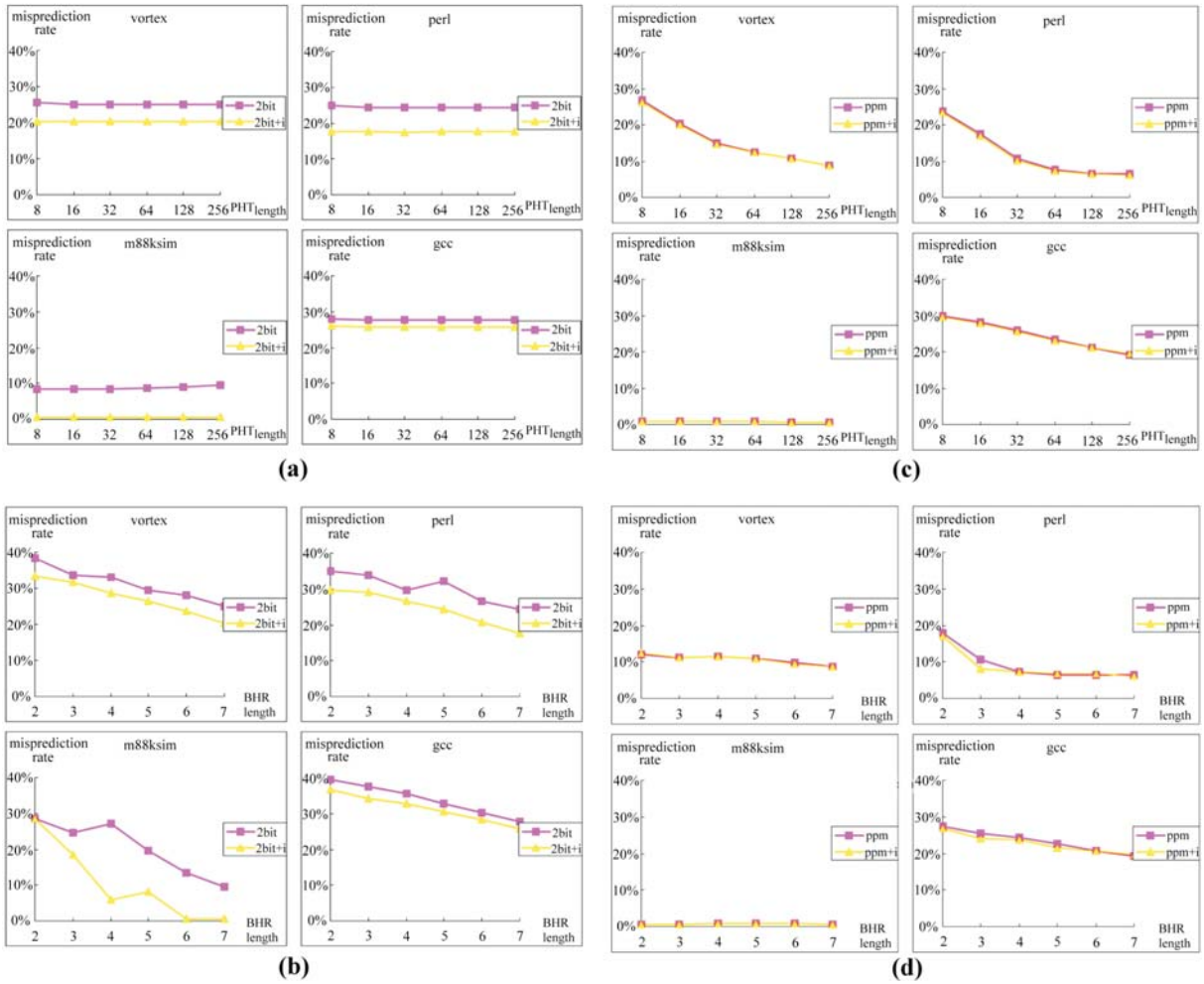

Figure 9. Incorporating our iterative approach with the 2-bit counter & PPM algorithm.

ple. For m88ksim, most of the predictions are made after only a few comparison attempts (i.e., with much longer matched parts), while the situation for gcc is quite reversed. This explains our previous performance comparison results. It also pinpoints the fact that our VCR scheme works even better for programs with history carrying "bigger loop" property.

If no match can be located even when the number of referred outcomes is reduced to only one in each compared part. In this case, prediction can be handed over to some other scheme, like the 2-bit counter adopted in our previous simulation. The "last bit" prediction is also a feasible alternative especially when reducing complexity is concerned. If the above non-matched situation happens, the "last bit" prediction will predict instantly the coming branch outcome to be the last bit, i.e., the "most recent" bit in the PHT entry. The performance of our VCR scheme using the last bit prediction in non-matched situations has also been simulated. It turns out that prediction accuracies for our VCR scheme using the 2-bit counter and the last bit are almost the same (e.g. with only about 1% accuracy difference for gcc).

5. Conclusion

To improve branch prediction accuracy, a variable cross-reference (VCR) prediction scheme and an iterative dispatch approach are proposed in this paper. The proposed VCR scheme can be easily implemented and is able to yield desirable prediction accuracy for a high performance processor at low cost. To further enhance prediction accuracy, an iterative dispatch approach is provided. The approach utilizes the PHT history to do dispatching for an additional layer of pattern history which helps providing more information for making better predictions. It is shown that the proposed VCR scheme and

iterative dispatch approach can handily work with other predictors to fortify performance. A PPM-VCR predictor is also presented to demonstrate the advantages of a combined predictor.

Performance of the proposed schemes and other prediction schemes is simulated (by conducting trace-driven simulation runs using four SPEC CINT95 benchmarks) for evaluation and comparison. The results show that the overall performance of our VCR scheme compares favorably to other schemes, such as the 2-bit counter and the agree predictor due to its variable cross-reference to the traces in the PHT. With much less complexity, the VCR scheme even outperforms the optimal and yet complicated PPM algorithm under some conditions. When compared with the bi-mode and YAGS predictors – which deal with both the prediction and dispatch parts of the two-level predictor and require extra hardware and cost, the VCR scheme still produces better performance in most of the situations. Simulation results show that the proposed iterative dispatch approach outperforms the gshare and DHLF predictors – schemes dealing with the dispatch part. It is also show that the iterative dispatch approach can lift prediction accuracy for different schemes, especially for schemes with lesser performance, such as the 2-bit counter. On the other hand, performance of the PPM-VCR combined predictor reveals slight degrees of improvement over the optimal PPM algorithm. The performance gain alone may not appear significant enough, but the potentially reducible complexity (due to the VCR scheme) is appealing.

References

[1] Boggs, D. et al., "The Microarchitecture of the Intel Pentium 4 Processor on 90 nm Technology," *Intel Technology Journal,* Vol. 8, Feb. (2004).

[2] Yeh, T.-Y. and Patt, Y. N., "Alternative Implementations of Two-Level Adaptive Branch Prediction," *Proc. 19th Annual Int'l Symp. on Computer Architecture*, May, pp. 124–134 (1992).

[3] McFarling, S., "Combining Branch Predictors," *Technical Report, TN-36,* Digital Western Research Laboratory, June (1993).

[4] Sprangle, E., Chappell, R. S., Alsup, M. and Patt, Y. N., "The Agree Predictor: A Mechanism for Reducing Negative Branch History Interference," *Proc. 24th Annual Int'l Symp. on Computer Architecture,* May, pp. 284–291 (1997).

[5] Chen, I.-C. K., Coffey, J. T. and Mudge, T. N., "Analysis of Branch Prediction via Data Compression," *Proc. 7th Int'l Conf. on Architectural Support for Programming Languages and Operating Systems,* Oct., pp. 128–137 (1996).

[6] Sechrest, S., Lee, C.-C. and Mudge, T., "Correlation and Aliasing in Dynamic Branch Predictors," *Proc. 23rd Annual Int'l Symp. on Computer Architecture,* May, pp. 22–32 (1996).

[7] Lee, C.-C., Chen, I.-C. K. and Mudge, T. N., "The Bi-Mode Branch Predictor," *Proc. 30th Int'l Symp. on Microarchitecture,* Dec., pp. 4–13 (1997).

[8] Eden, A. N. and Mudge, T., "The YAGS Branch Prediction Scheme," *Proc. 31st Int'l Symp. on Microarchitecture,* Dec., pp. 69–77 (1998).

[9] Juan, T., Sanjeevan, S. and Navarro, J. J., "Dynamic History-Length Fitting: A Third Level of Adaptivity for Branch Prediction," *Proc. 25th Annual Int'l Symp. on Computer Architecture,* May, pp. 155–166 (1998).

[10] Yeh, T.-Y. and Patt, Y. N., "Two-Level Adaptive Branch Prediction," *Proc. 24th annual Int'l Symp. on Microarchitecture,* Nov., pp. 51–61 (1991).

[11] Ross, S. M., *Introduction to Probability Models,* London, United Kingdom: Academic Press (1985).

[12] SPEC CPU'95, Technical Manual, Aug. (1995).

[13] Burger, D. and Austin, T. M., "The SimpleScalar Tool Set, Version 2.0," *Univ. of Wisconsin-Madison CS Dept. Technical Report #1342,* June (1997).

Manuscript Received: Sep. 15, 2006
Accepted: Mar. 13, 2007