ISSN No. 0976-5697

Volume 2, No. 1, Jan-Feb 2011

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

A Novel Approach for Branch Prediction using SVM

Pradipta Mishra
HiTech Institute of Technology
Bhubaneswar,Orissa,India
pradipta.system@gmail.com

Asis Kumar Tripathy*
NM Institute of Engineering and Technology
Bhubaneswar,Orissa,India
asistripathy@gmail.com

Abstract: Branch prediction accuracy is a crucial parameter in determining the amount of parallelism that can be exploited. Current advanced branch prediction techniques are able to attain correct predictions in most cases. Conventionally, the profiling information is used to predict the behavior of a branch. However, they are unsuccessful in predicting specific conditions like non linear branching and looping. This prevents them from achieving full accuracy. SVM have an inherent capability to analyze inputs to form meaningful relationships among them.

Keywords: Branch prediction, SVM

I. INTRODUCTION

Correct branch prediction is important especially in the age of super-pipelined superscalar processors. Highly accurate prediction is required to reduce the number of penalty cycles in the multiple instruction issue processors. Branch prediction is an essential part of modern micro architectures. Rather than stalling when a branch is encountered, a super pipelined processor uses branch prediction to speculatively fetch and execute instructions along the predicted path. Machine learning techniques offer the possibility of further improving the performance by increasing prediction accuracy.In our approach, we consider branch prediction as a specific problem belonging to the pattern recognition and in particular to the use of SVM based learning technique to predict the branches. The SVM have several attractive properties. They are simpler to implement and tune, their training is faster, and they are computationally inexpensive. In this Thesis, we investigate the behavior of the branches and loops.We capture the various characteristics of the branches and loops and feed them to a SVM to correctly predict the general branches and branches in the loop.

A. Branch Prediction

Originally branch predictors relied on static prediction schemes such as taken or not-taken and compiler help in determining branch outcome. Several dynamic branch prediction architectures have evolved due to heavy research in these areas. Dynamic branch predictors include one-bit table of counters, 2-bit table of counters and two level implementations of the counters to account for global history. Several implementations of predictors based on other ideas such as perceptrons [2] are emerging and may allow the fundamental limit of control dependences on ILP to be decreased.

Correct branch prediction is important especially in the age of super-pipelined superscalar processors. Highly accurate prediction is required to reduce the number of penalty cycles in the multiple instruction issue processors. The number of penalty cycles decides the performance loss for processors whose pipeline depth and instruction issue rate are high. Branch prediction is an essential part of modern micro architectures. Rather than stalling when a branch is encountered, a super pipelined processor uses branch prediction to speculatively fetch and execute instructions along the predicted path. Machine learning techniques offer the possibility of further improving the performance by increasing prediction accuracy.

B. SVM:

First of all working with neural networks [13] for supervised and unsupervised learning showed good results while used for learning applications. Multilayer perceptron (MLP)[12] uses feed forward and recurrent networks. The properties of MLPs include universal approximation of continuous nonlinear functions, learning with input-output patterns and also involve advanced network architectures with multiple inputs and outputs.

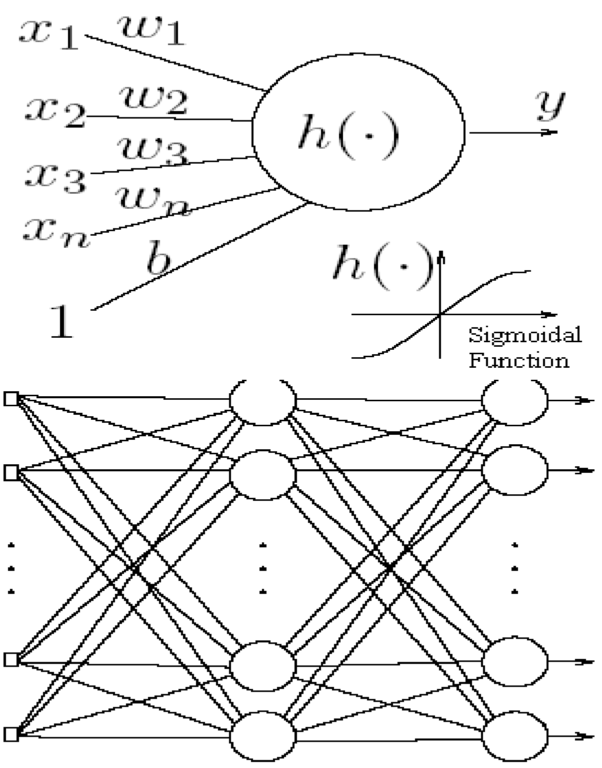

Figure 1[a.Simple Neural Network, 1 b.Multilayer Perceptron.]

These are simple visualizations just to have an overview as how neural network looks like.

II. LITERATURE SURVEY

Several attempts are made for successful implementation of branch prediction. Both static and dynamic branch prediction methods are useful. S. Patil, N. B.Anne, U. Thirunavukkarasu and E. E. Regentova[1] proposed a new branch prediction algorithm. Study of simple loop structures reveals certain parameters that give information about their characteristics. Danie A. Jimenez and Calvinlin [2] introduced a new branch predictor that uses neural learning techniques—the perceptron in particular—as the basic prediction mechanism. Burch [3] proposed a case study on static and dynamic branch prediction that shows merits and demerits of both the methods. Gummaraju and Franklin[4] proposed three methods to implement dynamic branch prediction in multithreaded processor that shows significant improvement in performance.

In his paper Hoogerbrugge [5] shows the improved results for a VLIW Processor using dynamic branch prediction. Steven, Anguera, Egan, Steven and Vintan [6] proposed a paper Improvement of Dynamic Branch Prediction using Neural Networks .Loh[7] proposed a case study simulation Difference Between Academia and Industry and he presented a comparative Study of two differene simulate on framework one for Academy and other for Industry.H. Loh[7] present a simple Divide and Conquer Approach for Neural class branch prediction. He chooses to design a new predictor that is quite different from previous neural approaches but still takes advantage of the same phenomena of deep history branch correlation. Neural Class branch predictor's able to achieve phenomenal prediction rate and IPC performance using only simple PHT structures.Ribas and Goncalves[8] shows work on Branch Prediction performance using Two-level perceotron Table. This work evaluates the performance of branch predictors based on multiplier perceptrons organized in two level tables. Guy III and Haggard [9] gives a paper high performance Branch predication. In this paper GHT-Logic architecture did not use a PHT and thus is much easier and smaller to implementation.Quinones and parcerisa [10] proposed a different approach ,which predicts the out comes of branches by predicting there guarding predicates.The predications are made for every predicate definition and store until the predicate is used by some branch.Daniel A. Jim´enezy Heather L. Hansonz Calvin Lin [11]These Boolean formulas provide a compact encoding of a class of functions that is expressive enough to perform branch prediction yet concise enough to be encoded in branch instructions.

III. PROBLEM STATEMENT: BRANCH PREDICTION

Small improvements in accuracy can have a large impact on performance; decreasing the misprediction rate from, say, 5% to 4% can decrease the execution time of a typical program by as much as 14%. Here, we propose a novel branch prediction scheme using SVM which can achieve a very better accuracy. We assume that all predictions are being made in parallel to the instruction decode (ID) stage of the processor pipeline. The actual prediction is generated by a SVM. The 80386 instruction set is considered in our study.

IV. BRANCH PREDICTION USING SVM

SVM are a family of algorithms remarkably efficient in many real-world applications. A growing interest for SVM in bioinformatics has emerged recently and resulted in powerful methods for various tasks.

A SVM basically learns how to classify objects $x \in X$ into two classes $\{-1, +1\}$ from a set of labeled training examples $\{x1 . . . xm\}$.

The resulting classifier is based on the decision function:

$f(x) = \sum_{i=1}^{m} \lambda i K(xi,\ x)$ where x is any new object to be classified, K(., .) is a so-called kernel function and the coefficients $\{\lambda 1, . . . , \lambda m\}$ are learned during training by solving a constrained optimization problem.

The kernel K can be considered as a dot product between objects, or more precisely between the images of the objects after a mapping to a highdimensional Hilbert space. As a result it defines the metric properties of the space of objects, namely the "size" of each object and the "angle" between any two objects.

Let's takes some MIPS Assembly language code

```
Beq    $at,   $0,    L                    pcsrc=1   Branches
Add        $V1, $0,    $0                 to  pc+4+(offset*4)
Add    $v1,   $v1,    $v                           pcsrc=0
continues to
J            somewhere                           pc+4
L:  add      $v1, $v0, $v0
```

A. Fetch the instruction like beq $at, $0, offset from memory
B. Read the source register from the register file
C. Compare the values by subtracting them in the ALU
D. If the subtraction result is 0 the source operands were equals and the pc should be loaded with the target address, PC+4+(offset*4)
E. E. Otherwise the branches should not be taken and the PC should just be incremented PC +4 to fetch the next instruction

Algorithm

Correctly classify all training data

Let take w =weight each instruction set and x_i=address of instruction set

$wx_i+b>=1$ if $y_i=+1$ for Pcsrc=1

$wx_i+b>=-1$ if $y_i=-1$ for Pcsrc=0

$y_i\ (wx_i+b)>=1$ for all i

Maximize margine Same as $\underline{1/2w^t w}$

We can formulate a Quadratic Optimization Problem and solve for w and b

Minimize $\phi(w)=\underline{1/2w^t w}$

Subject to $y_i(wx_i+b)>=1$ Pcsrc =1 for all i

This equation is represented all branch instructions present in the instruction sample .

When a branch is encountered:

A.The branch address is hashed to index i, to access SVM[i]

B. SVM[i]'s weights, SVM $_\alpha$ [i], are fetched into an vector register of floating point weights, $\alpha = (\alpha\ 0 . . . \alpha\ m)$.

C. The dot products ki = Ker (PHT, SVM $_\alpha$[i]) for i € {1 . . .m} are calculated in parallel. There are many simple kernels available for which this is a fast computation.

D. The value of y is computed as the dot product of SVM $_{\alpha}$[i]and the global history register.
E. The results of the multiplications and the bias, y = b0 +$\sum^{m}_{i=1}$ K (PHT, SVM $_{\alpha}$[i]) are summed.
F. The branch is predicted not taken when y is negative, or taken otherwise. Once the actual outcome of the branch becomes known, the training algorithm uses this outcome and the value of y to update the weights.
G. The final prediction is the sign of y.

The predictor stores the value of i, for later training of SVM[i].

A SVM is a learning device that takes a set of input values and combines them with a set of weights (which are learned through training) to produce an output value. In our predictor, each weight represents the degree of

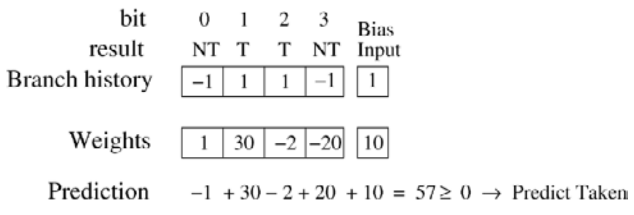

Figure. 12

The SVM prediction mechanism. The prediction is the sign of the dot product of the branch history and the Svm weights. The taken branches (T) in the branch history are represented as 1's, and not taken branches (NT) are represented as >=1's. The bias weight represents the bias of the branch independent of branch history, so its input bit is hardwired to 1. Correlation between the behavior of a past branch and the behavior of the branch being predicted. Positive weights represent positive correlation, and negative weights represent negative correlation. To make a prediction, each weight contributes in proportion to its magnitude in the following manner. If its corresponding branch was taken, we add the weight; otherwise we subtract the weight. If the resulting sum is positive, we predict taken; otherwise we predict not taken. To make this solution work, the branch history uses 1 to represent taken and >=1 to represent not taken. The perceptrons are trained by an algorithm that increments a weight when the branch outcome agrees with the weight's correlation and decrements the weight otherwi

V. PREDICTION USING SVM

Total sample Train sample Validation Test sample Prediction

4 1:0 2:1	4 1:0 2:1			
8 1:4 2:1	8 1:4 2:1	44 1:40 2:1		
12 1:8 2:4	12 1:8 2:4	48 1:44 2:2	64 1:60 2:2	63.99998
16 1:12 2:8	16 1:12 2:8	52 1:48 2:1	68 1:64 2:2	67.999978
20 1:16 2:1	20 1:16 2:1	56 1:52 2:1	72 1:68 2:8	71.999976
24 1:20 2:1	24 1:20 2:1	60 1:56 2:1	76 1:72 2:4	75.999974
28 1:24 2:2	28 1:24 2:2		80 1:76 2:1	79.999972
32 1:28 2:2	32 1:28 2:2			
36 1:32 2:4	36 1:32 2:4			
40 1:36 2:8	40 1:36 2:8			
44 1:40 2:1				
48 1:44 2:2				
52 1:48 2:1				
56 1:52 2:1				

VI. CONCLUSIONS

In this paper a novel branch predictor that uses SVM learning techniques has been introduced. The SVM in particular is used as the basic prediction mechanism with suitably formulated inputs. The ability of SVM to use long history lengths is a key advantage .These long history lengths lead to better accuracy. Weakness of perceptron in the single layer form is its inability to learn linearly inseparable functions. This is a limitation of existing branch predictors using perceptrons. Another strength of SVM is its ability to learn linearly inseparable functions. The training of the learning network is relatively easy, when the SVM is used. Further no local optimal solution evolves, unlike in neural networks. It scales relatively well to high dimensional data and the trade-off between classifier complexity and error can be controlled explicitly.

VII. REFERENCE

[1] S. Patil, N. B.Anne, U. Thirunavukkarasu, E. E. Regentova technical report Department of Electrical and Computer Engineering University of Nevada, Las Vegas
[2] Daniel A. Jimenez, Stephen W. Keckler, and Calvin Lin. Then impact of delay on the design of branch predictors.In Proceedings of the 33th Annual International Symposium on Microarchitecture, December 2000.
[3] Yu Wang & Lei Chen [2] technical report Department of Computer Science Univer sity of California, Davis.
[4] Burch, "A case staudy of Static and Dynamic Branch Prediction", International conference, IEEE, 1997
[5] Gummaraju and Franklin, "Branch Prediction in multi-Threaded Processors"International Conference on

parallel Architecture and Compilation Technique, IEEE, 2000

[6] Hoogerbrugge , "Dynamic branch prediction for a VLIW Processor", International Conference,IEEE,2000

[7] Steven, Anguera, Egan, Steven and Vintan "Dynamic Branch Prediction Using Neural Network",International Journal IEEE,2000

[8] Loh,"Simulation Difference between Academic and Industry: A Branch Prediction case staudy", IEEE, 2000

[9] H.Loh, "A Simple Divide and Conqure Approach for Neural Class Branch Prediction", International Conference on parallel Architecture, IEEE.2005

[10] Ribas and Goncalves ,"Evalute branch prediction Using Two-level Perceptron Table",IEEE,20068.

[1]] Guy III and Haggard, "High Performance Prediction"International journal, IEEE, 1996

[12] Fundamentals of Neural Networks Architectures, Algorithms, and Application Laurene Fau sett Prentice Hall International 1994

[13] Vikramaditya Jakkula technical report Department of Computer Science of School of EECS, Washington State University, Kernel machine, www.Kernel-machine.org.