제21회 한국정보처리학회 춘계학술발표대회 논문집 제11권 제1호 (2004. 5)

신경망을 이용한 분기 예측의 개선

곽종욱, 김주환, 전주식
서울대학교 전기 컴퓨터 공학부
e-mail : {leoniss, if10001, csjhon}@panda.snu.ac.kr

A Novel Approach to Improve Branch Prediction Accuracy by Neural Network Information

Jong Wook Kwak, Ju-Hwan Kim, Chu Shik Jhon
Dept. of Electrical Engineering and Computer Science, Seoul National University

요 약

파이프라인과 슈퍼스칼라 방식이 일반화된 시스템 구조 하에서, 분기 명령어는 시스템 전체적인 성능에 중요한 영향을 미친다. 특히 분기 예측이 실패했을 경우, 잘못된 분기 예측으로 인한 페널티가 발생한다는 점에서 분기 예측의 정확도에 대한 중요성은 크다고 할 수 있다. 본 논문에서는 분기 예측의 정확도를 높이기 위해서, 분기 예측과 관련된 신경망을 구축하여 이를 통해 분기 예측에 필요한 각 요소별 가중치의 변화를 분석하고, 이를 분기 예측에 새롭게 반영하고자 한다. 본 논문에서는 이를 위해 실행 구동 방식의 시뮬레이터인 *SimpleScalar* 를 통하여 모의 실험을 수행하였으며, 실험 결과 본 논문에서 제시한 새로운 기법이 기존의 일반적인 이단계 적응형 분기 예측 기법이나 *gshare* 기법에 비하여 더 우수한 결과를 보였다.

1. 서론

컴퓨터 구조적인 측면에서 좀 더 높은 단계의 성능 향상을 추구하기 위한 노력과 연구가 최근까지 많이 진행되어 왔다. 특히 고성능의 컴퓨터 시스템에 있어서 명령어 단위의 병렬성(ILP, Instruction Level Parallelism)에 대한 추구는 시스템 성능 향상을 위한 필수 조건으로 부각되어 왔다. 이와 같은 ILP 를 향상시키기 위한 노력으로, 더욱 세분화된 파이프라인(Pipeline)의 사용과 아울러 다양한 형태의 슈퍼스칼라(Superscalar) 방식들이 제안되었다. 또한 프로세서의 입장에서 뿐만 아니라 메모리적 측면에서도 다양한 형태의 캐쉬 구조 개선에 관한 연구 논문이 많이 발표되었다. 한편, 시스템상에서 수행되는 작업(Workload)들 역시 기존과 비교하여 더욱 다양화되고, 또한 이를 지원하는 컴파일러나 운영체제 등의 발전에 따라 각 응용 프로그램의 수행 패턴 역시 점차 최적화 되어가는 방향으로 변화되어 가리라 예상할 수 있다[1].

이와 같은 세분화된 파이프라인과 다수개의 명령어들을 동시에 이슈되는 슈퍼스칼라 방식, 그리고 이를 지원하는 최적화된 캐쉬 메모리 구조와 컴파일러, 운영체제와 같은 다양한 형태의 시스템 지원 요소 등이 결합된 고성능 시스템에서의 분기 예측 기법(Branch Prediction Scheme)의 정확도는 시스템의 성능 향상에 결정적인 영향을 미친다고 할 수 있다. 이러한 최적화된 시스템 구조와 변화되고 다양화되는 응용 프로그램의 관계 하에서, 각 분기 명령(Branch Instruction)의 Taken/NotTaken(이하 T/NT) 여부에 영향을 미치는 요소들도, 요소 자체가 바뀌거나 영향을 미치는 정도가 변화 되리라 예상할 수 있다. 따라서 본 논문에서는 신경망(Neural Network)의 구성을 통한 도출 정보를 바탕으로, 이를 분석하여 새로운 형태의 분기 예측 기법을 소개하고자 한다. 또한 각각의 요소들이 분기의 T/NT 여부에 미치는 영향의 정도를 분석하고자 한다.

이하 본 논문의 구성은 다음과 같다. 2 장에서는 본 논문에서 이용하는 신경망과 관련된 배경지식에 대한 설명과 아울러 기존의 대표적인 분기 예측 기법에 대한 소개를 한다. 그리고 3 장에서는 본 논문에서 제시하고자 하는 신경망의 정보를 이용하는 새로운 형태의 분기 예측 기법을 소개하며, 4 장에서는 이에 대한 성능을 평가하고 그 결과를 분석한다. 끝으로 5 장에서는 본 논문의 결론 및 향후 연구 과제를 제시하고

본 논문을 끝맺는다.

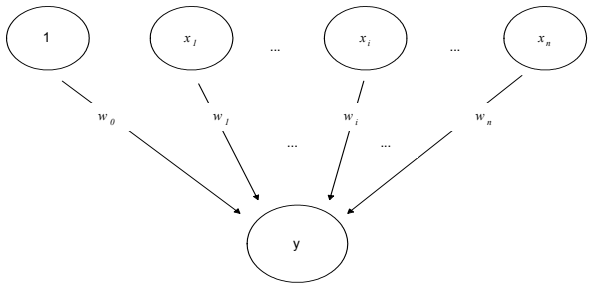
2. 관련 연구

이 절에서는 본 논문과 관련된 배경지식으로서의 신경망에 대한 언급과, 기존의 분기예측 기법에 대한 소개를 한다.

2.1 신경망

인공 지능의 한 분야로서의 신경망은, 다양한 응용 분야에 이용되어 왔으며, 특히 패턴의 인식(Pattern Recognition), 분류(Classification)[2], 이미지 이해(Image Understanding) 등은 그 대표적인 활용 분야라 할 수 있다[3]. 본 논문의 연구 주제인 분기 예측 기법 역시 각각의 분기들을 Taken 분기와 NotTaken 분기로 이를 이분하는 분류(Classification)에 기반을 두는 신경망의 응용예이라 할 수 있다.

이러한 신경망의 주요 구성 요소중의 하나로서 Perceptron 이 있다. Perceptron 의 초기 개념은 1962 년 두뇌의 기능을 연구하기 위해서 최초로 제안되었으며[4], 현재 다양한 형태의 Perceptron 들이 존재하고 있다[5].



(그림 1) Perceptron 모델

그림 1 에 기본적인 Perceptron 모델이 도시되어 있다. 각각의 입력값 $x_1, \ldots, x_n$ 은 해당되는 연결선(Edge)을 통해 각각 $w_1, \ldots, w_n$ 의 가중치를 가지며, 이를 y 의 입력으로 전파(Propagation)한다. 이를 통한 Perceptron y 의 출력은 일반적으로 다음과 같이 계산된다.

$$y = w_0 + \sum_{i=1}^{n} x_i w_i$$

2.2 분기 예측

최근 제시되는 분기 예측 방식은 크게 *bimodal* 기법과 *gshare* 기법으로 구분 지어 볼 수 있다. *bimodal* 방식이라 불리는 기존의 기법은 분기 예측을 위해서 별도의 분기 예측 테이블(Branch Predictor Table)을 사용한다. 이때 해당 테이블의 각 요소들은 해당 분기 명령어의 주소와 함께 일반적으로 2-bit Saturation Counter 를 가지고 있으며, 해당 분기의 주소값에 대한 Counter 값을 이용하여 분기 예측을 수행한다. *gshare* 기법은 1993 년 Western Research Laboratory 에서 제안한 기법으로[6], Yeh and Patt 의 이단계 적응형 분기 예측 기법(2 Level Adaptive Training Branch Prediction)[7]의 변형이라 할 수 있다. 이러한 *gshare* 기법에서는 분기 예측에 있어서 분기 명령어의 PC 값과 함께, *bimodal* 기법과는 달리 추가적인 Global Branch History 값을 사용한다.

한편, 신경망을 통한 분기 예측의 성능 향상과 관련된 기존의 연구는 주로 정적 분기 예측(Static Branch Prediction)에 국한되는 경향이 있었다[8]. 하지만 최근들어 신경망을 통한 다양한 형태의 동적 분기 예측(Dynamic Branch Prediction) 기법들이 소개되고 있으며[9][10], 그리고 이는 대부분 Yeh and Patt 의 이단계 적응형 분기 예측 기법의 변형을 시도한 것들이라 할 수 있다. 본 논문에서 제시하고자 하는 "신경망을 이용한 분기 예측의 개선" 기법 역시 이단계 적응형 분기 예측 기법의 변형 가운데 하나로서, 이와 같은 변형에 대한 힌트로서 신경망의 도출 결과를 이용한 방식이라 할 수 있다.

3. 신경망의 정보를 이용한 분기 예측 기법의 개선

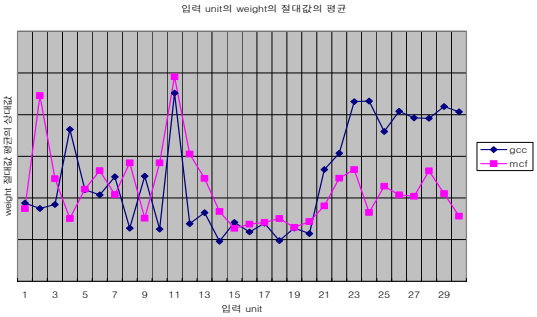
이 절에서는 분기 예측에 있어서 이에 영향을 미치는 다양한 요소들에 대한 가중치의 정도를 파악하기 위한 신경망의 구성 및 이에 대한 실험 결과를 소개하며, 이를 통한 새로운 형태의 개선된 *gshare* 기법을 제안한다.

3.1 신경망의 구성

본 논문에서 사용하는 신경망의 구성은 다음과 같다. 신경망의 입력은 총 30 개로, 가장 최근의 Global Branch History 10 개, Branch Direction History 10 개, 그리고 분기 명령어의 상위 PC 값 10bit 를 사용하였다. 여기서 Branch Direction History 는 분기의 T/NT 여부에 영향을 미치는 정도를 조사하기 위해서 본 논문에서 추가적으로 사용한 요소로써, 각각의 분기 명령어마다 1bit 씩을 할당하여, 전방 분기(Forward Branch)이면 1, 후방 분기(Backward Branch)이면 0 을 기록한다. 또한 입력 Perceptron 30 개와는 별도로, 내부(Hidden) Perceptron 을 3 단계로 구분하여 각각 60, 30, 10 개씩 총 100 개를 사용하였으며, 끝으로 출력 Perceptron 1 개와 추가적으로 초기 Bias 용 Perceptron 1 개를 사용한다. 내부 Perceptron 의 구성 설정은 다음과 같다. 우선 1 단계 Perceptron 60 개와 2 단계 30 개, 그리고 2 단계 30 개와 3 단계 10 개 사이는 완전 연결(Fully Connected)로서, $60\times30+30\times10$ 개의 링크를 사용하며, 3 단계 10 개와 출력 Perceptron 사이에는 10×1 개의 링크를 사용한다. 특히 입력 Perceptron 과 1 단계 내부 Perceptron 과의 연결은 3 가지 입력 요소 중 분기 명령어의 PC 는 다른 요소와의 상관 관계가 미미하다고 판단되어, Global Branch History 10 개와 Branch Direction History 10 개에서 내부 Perceptron 1 단계 60 개 중 40 개에만 완전 연결로 연결하여 20×40 개의 링크를 사용하고, 분기 명령어의 PC 값 10 개의 입력은 별도로 내부 1 단계 60 개 가운데 나머지 20 개로의 연결만을 설정하여 10×20 개의 링크를 사용하였다. 끝으로 Bias Perceptron 에서 내부 Perceptron 과 출력 Perceptron 으로의 연결에 101 개의 링크를 사용하였다.

신경망의 학습을 위해 전파(Propagation)에 사용되는

식은 *sigmoid function* 을 사용하였으며, 후방 전파(Back Propagation)를 위해서는, 각 Perceptron 의 *delta* 값을 계산하여 이를 통해 가중치의 변화를 조정하는 방식을 사용하였다. 이와 같이 주어진 신경망에서 각 입력 Perceptron 의 가중치 절대값의 상대적인 차이를 나타낸 내용이 그림 2 에 나타나 있다.



(그림 2) 입력 Perceptron 의 가중치의 변화량

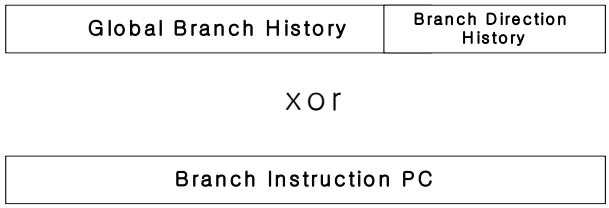
사용된 응용은 SPEC CINT2000[11]에서 제공하는 응용 프로그램들 중 일부이다. 앞서 언급한 바와 같이, 1 번부터 10 번까지의 입력은 Global Branch History 이며, 1 번에 가까울수록 최근의 분기를 뜻한다. 11 번부터 20 번까지의 입력은 Branch Direction History 로써, 이 역시 11 번에 가까울수록 최근의 분기를 뜻한다. 끝으로 21 번부터 30 번까지의 입력은 분기 명령어 PC 의 상위 비트들로써, 21 번에 가까울수록 하위비트(LSB)이며, 30 번에 가까울수록 상위비트(MSB)이다.

그림 2 에서 보여주듯이, 분기 명령어의 PC 값은 상위 혹은 하위 비트의 특별한 구분 없이 대체적으로 높은 가중치를 가진다는 것을 알 수 있다. 반면 Global Branch History 와 Branch Direction History 는 최근의 값일수록 높은 가중치를 가지며, 최근에서 점차 멀어질수록 대체적으로 가중치가 줄어드는 경향을 가진다는 것을 알 수 있다. 특히 주목할만한 사항은, Global Branch History 의 경우는 Branch Direction History 와 비교하여 상대적으로 다양한 가중치의 변화 정도를 보이지만 대체로 비슷한 값을 가지며 천천히 줄어드는 경향을 보이는 반면, Branch Direction History 는 최근의 값일수록 매우 높은 값을 가지며, 이 값은 최근에서 멀어질수록 빠르게 줄어드는 경향이 있는 것을 알 수 있다. 또한 이때의 최근 Branch Direction History 의 가중치 값은 상대적으로 다른 여타의 입력보다 월등히 높은 값을 가진다는 것을 알 수 있다. 이와 같은 신경망의 구성 하에서 나타난 특징을 기반으로 하여, 본 논문에서는 분기 예측에 필요한 각 구성요소들의 영향력의 정도를 반영하여 새로운 형태의 *gshare* 기법을 제안하고자 한다.

3.2 개선된 형태의 *gshare*

기존의 *gshare* 기법은 Global Branch History 와 분기 명령어의 PC 정보만을 사용하여 분기 예측 테이블을 Indexing 하는 것에 비하여, 본 논문에서는 그림 3 과 같이 Branch Direction History 를 추가하여 새로운 방식으로 분기 예측 테이블을 Indexing 하고자 한다.

그림 3 에서 보여지듯이 제안된 새로운 기법은 기존의 *gshare* 에서 Global Branch History 만이 차지하던 부분을 Global Branch History 와 Branch Direction History 를 조합하여 사용하는 방식으로, 이를 분기 명령어의 PC 값과 *xor* 를 수행하여 최종적으로 분기 예측 테이블을 Indexing 한다.



(그림 3) Branch Direction History 를 포함하는 *gshare*

4. 시스템 성능 분석

본 논문에서 제안한 기법을 평가하기 위해서 모의실험을 수행하였다. 모의실험에는 프로그램 실행 구동방식(Execution Driven)의 시뮬레이터인 *SimpleScalar*[12]를 사용하였다.

4.1 모의 실험 환경

본 논문에서 사용된 모의 실험 환경은 다음과 같다. 기본적인 실험 환경은 *SimpleScalar* Tool Set 중, 분기 예측과 관련된 구동형 시뮬레이터인 *bpred* 를 사용하였으며, 이에 적용된 응용 프로그램은 SPEC CINT2000 에서 제공하는 벤치마크 프로그램 중에서 임의로 선택된 4 가지 이다. 사용된 응용의 세부 사항은 테이블 1 과 같다.

<표 1> 응용 프로그램

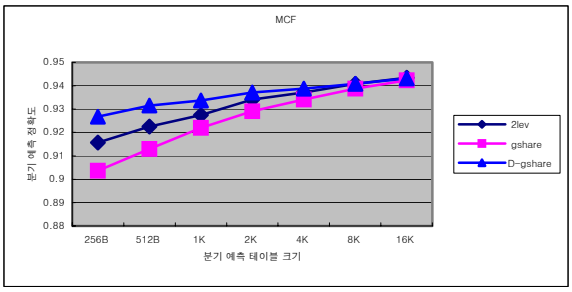
Benchmark	Description
mcf	Combinatorial Optimization
gzip	Compression
gcc	C Programming Language Compiler
vpr	FPGA Circuit Placement and Routing

4.2 실험 결과

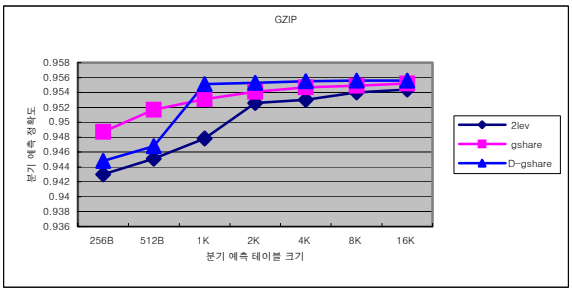
분기 예측과 관련된 정책이 바뀌었을 때의 가장 분명한 비교 수치는 분기 예측의 정확도(Branch Prediction Accuracy)이다. 모의실험을 통해 *mcf, gzip, gcc, vpr* 의 각 응용 프로그램별 분기 예측의 정확도를 나타낸 내용이 그림 4 에서 그림 7 에 나타나 있다. 이때 분기 예측 테이블의 크기는 256byte 에서 16Kbyte 까지 변화를 주었으며, 분기 예측의 비교 대상은 기본적인 이단계 적응형 분기 예측 기법(2lev)과 그리고 이를 변형한 *gshare* 기법(gshare), 끝으로 본 논문에서 제시한 Branch Direction History 가 포함된 새로운 형태의 *gshare*(D-gshare) 기법이다.

그림 4 부터 그림 7 에서 보여 지듯이, 각 응용 프로그램별 분기 예측의 정확도에 있어서 약간의 차이는 존재하지만, 일반적으로 이단계 적응형 분기 예측 기법과 *gshare* 기법은 비슷한 양상을 보인다는 것을 알

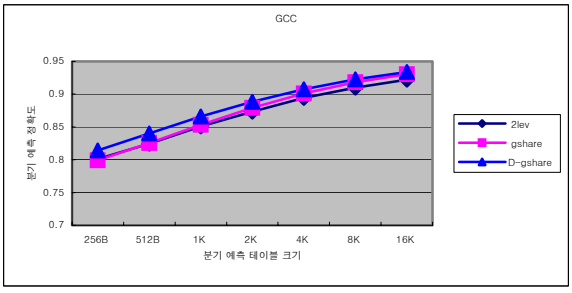
수 있다. *mcf* 와 *vpr* 의 경우는 근소한 차이로 이단계 적응형 분기 예측 기법이 우수하며, 반대로 *gcc* 와 *gzip* 은 *gshare* 기법이 성능면에서 우수하다는 것을 알 수 있다. 이와 비교하여, 본 논문에서 제시된 Branch Direction History 가 포함된 *gshare* 기법은 모든 응용의 다양한 분기 예측 테이블의 크기 하에서, 대부분 보다 더 정확한 분기 예측의 결과를 보인다는 것을 알 수 있다. 그리고 이러한 결과는, 3.1 절에서 제시된 결과를 바탕으로 분기의 예측에 영향을 미친 각 요소별 가중치를 반영하여, 보다 더 높은 가중치 요소를 분기 예측 기법에 적용시킨 결과라 하겠다.



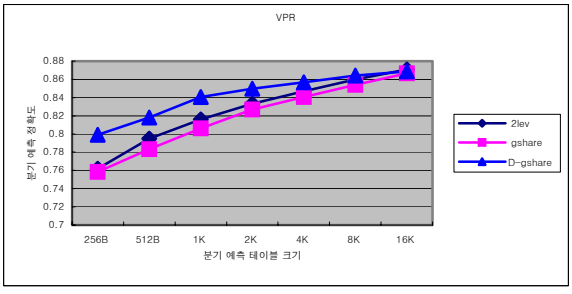
(그림 4) *mcf* 의 분기 예측 정확도



(그림 5) *gzip* 의 분기 예측 정확도



(그림 6) *gcc* 의 분기 예측 정확도



(그림 7) *vpr* 의 분기 예측 정확도

5. 결론

시스템의 성능 향상을 위한 프로세서 기술과 캐시 메모리 기술의 상호 관계 속에서, 분기 예측의 정확도를 높이는 기법은 시스템 성능 향상의 필수적인 요소로 자리 잡았다. 본 논문에서는 이러한 분기 예측의 정확도를 높이기 위하여, 기존의 시스템에서 일반적으로 사용되는 이단계 적응형 분기 예측 기법과 *gshare* 기법의 새로운 대안을 제시하였다. 새로운 방식은 신경망에 근거하여 도출된 결과를 바탕으로, 분기 예측에 영향을 미치는 각 요소 별로의 가중치의 변화를 분석하여, 높은 영향을 미치는 요소를 분기 예측에 새롭게 적용한 기법이라 할 수 있다. 실험 결과, 제안된 새로운 기법은 대부분의 경우에 있어서 기존의 방법과 비교하여 우수한 성능을 나타내었다.

본 논문의 향후 연구과제로는 다음과 같은 사항을 들 수 있다. 우선, 보다 더 다양한 응용 프로그램을 통한 추가적인 실험이 필요하며, 비교 대상을 더욱 확대하여 *bimodal* 기법이나, *comb* 기법과의 비교도 확인하여 이들과의 성능상의 차이점을 분석해 보아야 할 것이다.

참고문헌

[1] J. L. Hennessy and D.A Patterson, Computer Architecture : A Quantitative Approach, Second Edition, Morgan Kaufmann Publishers, Inc, 1996.

[2] L. Faucett, Fundamentals of Neural Networks : Architectures, Algorithms and Applications. Prentice-Hall, Englewood Cliffs, NJ, 1994

[3] Arun D. Kulkarni, Artificial Neural Networks for Image Understanding. Van Nostrand Reinhold, 1993

[4] Rosenblatt, F. Principles of Neurodynamics : Perceptrons and the Theory of Brain Mechanisms. Spartan, New York, 1962

[5] Block, H. D., The Perceptron : A Model for Brain Functioning. Rev. Mod. Phy. 34, 123-135, 1962

[6] McFarling, S., Combining branch predictors. Tech. Rep. TN-36m, Digital Western Research Lab., June, 1993

[7] Yeh, T. Y. and Patt, Y. N., Two-level adaptive branch prediction. In Proceedings of the 24th ACM/IEEE International Symposium on Microarchitecture, 51-61, 1991

[8] Brad Calder, D. Grunwald, M. Jones, D. Lindsay, J. Martin, M. Mozer, and B. Zorn. Evidence-based static branch prediction using machine learning. ACM Transactions on Programming Languages and Systems, 19(1), 1997

[9] Jimenez, D. A. and Lin, C., Dynamic branch prediction with percetprons. In Proceedings of the 7th International Symposium on high Performance Computer Architecture, 197-206, 2001

[10] G. Steven, et al., Bynamic Branch Prediction using Neural Networks, In Proceedings of the Euromicro Symposium on Digital Systems Design, IEEE, 2001

[11] SPEC CPU2000 Benchmarks, http://www.specbench.org

[12] D. Burger, T. M. Austin, and S. Bennett, "Evaluating future micro-processors: the SimpleScalar tool set", Tech. Report TR-1308, Univ. of Wisconsin-Madison Computer Sciences Dept., 1997.