부분 태그와 작은 데이터 크기에 기반한 저비용 연산결과 예측기 구조

최병수, 이동익
광주과학기술원 정보통신공학과
전화 : 062-970-2248

Cost Effective Value Prediction Microarchitecture using Partial-Tag and Narrow-Width Operands

Byung-Soo Choi and Dong-Ik Lee
Dept. of Information and Communications, K-JIST
E-mail : {bschoi,dilee}@kjist.ac.kr

Abstract

In this paper we investigate the implementation cost of value prediction methods for high performance micro-processors, and propose a new value prediction microarchitecture with low cost. After simulation, we found that the proposed microarchitecture can decrease the implementation cost by 36% to 50% and with slight performance degradation(less than 5%).

I. 서론

프로세서의 성능을 향상하는데 있어서 이론적인 한계는 명령어간 데이터 종속성에 기반한다[1]. 따라서 명령어간 데이터 종속성이 있는 명령어가 많으면 성능향상의 정도가 높지 않다. 이러한 근본적인 문제를 해결하기 위해서 최근 연산결과 예측 기법들이 소개되고 있다[2]. 이 방법들에 의하면 연산결과를 생성하는 명령어와 그 연산결과를 사용하는 명령어가 동시에 수행 가능하기 때문에 이론적으로 성능향상의 제한조건을 없앨 수 있다. 하지만 이런 이상적인 성능향상의 대가로 매우 큰 테이블을 사용해야 한다는 문제점으로 인하여 구현시 큰 문제가 되고 있다. 본 연구에서는 기존의 연산결과 예측 방법들을 비용과 성능 특성면에서 분석하고 비용을 크게 줄이면서도 성능저하를 적게하는 새로운 예측기 구현 구조에 대해서 소개한다. 시뮬레이션 결과 하드웨어 비용절감은 36%~50%정도로 매우 크면서도 성능은 기존의 방법과 비슷한 정도(5%이내의 감소)임을 알게 되었다.

II. 기존의 방법들과 비용 특성

현재까지 여러 가지 방법들이 제안되었으나 가장 많이 연구되고 있는 4 가지 방법을 기준으로 평가한다. 그들은 각각 Last[3], Stride[3], 2Level[4], Hybrid[5]방법이다.

표 1. 각 VP 별 Bit Cost 계산식

VP	Bit Cost Equation
Last	VHTentry*(Tag_bit + Value_bit + Confidence_bit)
Stride	VHTentry*(Tag_bit + State_bit + Value_bit + Stride_bit)
2level	VHTentry*(Tag_bit + LRU_bit + Value_bit*History + log2PHTentry) + PHTentry*(Confidence_bit*History)
Hybrid	VHTentry*(Tag_bit + State_bit + Stride_bit + LRU_bit + Value_bit*History + log2PHTentry) + PHTentry*(Confidence_bit*History)

Full-Tag_bit = 32(PC address) - 3(log2instruction_byte_size) - log2VHTentry, Full-Value_bit ≈ 32
Confidence_bit = 2, State_bit = 2, Stride_bit = 32, LRU_bit = (log2History) * History

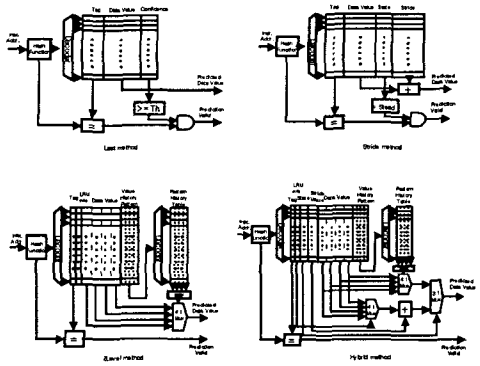


그림 1. 4 가지 연산결과 예측기 구조

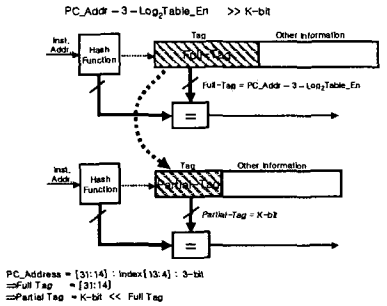


그림 2. 부분 태그를 이용한 엔트리 비용 감소

각각의 예측기 구조는 그림 1 을 참조하기 바란다. Last 방법은 하나의 연산명령어는 기존의 연산결과와 동일한 연산결과를 반복적으로 생성한다고 가정한다[3]. Stride 방법은 하나의 명령어를 기준으로 연속된 연산결과의 등차를 이용하여 가장 마지막의 연산결과와 등차의 합을 다음 명령어의 연산결과로 가정한다[3]. 2Level 방법은 하나의 명령어는 여러 개의 선택된 연산결과를 반복적으로 생성하며 그 순서가 일반적으로 일정하다는 가정하에서 연산결과를 선택적으로 예측하는 방법이다[4]. 본 연구에선 과거 4 개의 연산결과를 저장하는 것으로 가정하였다. Hybrid 는 2Level 방법이 등차를 갖는 명령어의 경우 예측이 안되는 단점을 보완하기 위해서 Stride 방법과의 연동을 활용하여 매우 높은 정확도를 확보하는 방법이다[5]. 한편 일반적으로 이러한 방법들은 매우 큰 테이블을 사용하는데, 그림 1 을 참조하면서 계산한 각각의 방법들의 비용 특성식은 표 1 과 같다. 표 1 에 따른 구현비용 순으로는 Hybrid > 2Level >> Stride >> Last 임을 알 수 있다. 한편 이러한 테이블의 크기는 매우 커서 일반적으로 명령어나 데이터 캐시의 크기보다 훨씬 크다. 따라서 본 연구에서는 이러한 비용의 문제를 효과적으로 해결하면서 성능 저하를 최소화하는 방법을 제안한다.

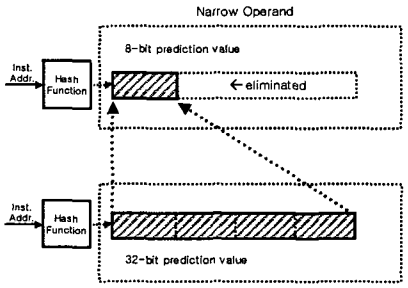


그림 3. 적은 유효비트의 활용에 따른 비용 감소

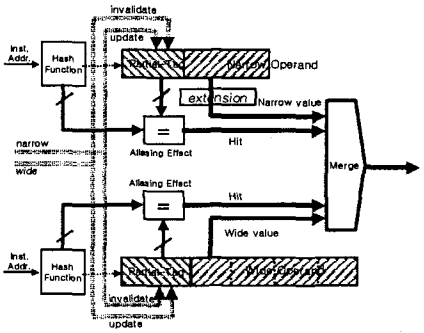


그림 4. 부분 태그와 비대칭 데이터 사이즈를 갖는 최적 구조

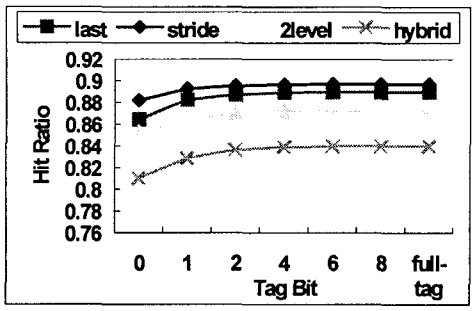
III. 최소 비용 구현 구조

3.1 태그의 최소화

일반적으로 예측값을 생성하는 테이블은 인덱스를 통해서 그 테이블의 개별적인 엔트리를 접근한다. 이때 인덱스로부터 접근되는 엔트리가 정확한 주소로 사상이 되는지를 검사하기 위해서 일반적으로 태그를 활용한다. 따라서 테이블의 엔트리는 태그와 데이터를 포함한다. 이러한 태그는 크기가 크면서도 그 변화의 폭이 매우 적다는 특성이 있다. 이러한 특성에 기반하여 연산결과 예측기에서 태그를 전부다 활용하지 않고 몇 개의 비트만 활용하는 방법이 소개되었다[6](그림 2 참조). 본 연구에서도 이 방법을 적용하여 태그부분의 비용을 최소화하도록 하였다.

3.2 데이터 공간의 최소화

한편 데이터 부분의 특성을 살펴보면 Last, 2Level 의 경우는 과거의 연산결과가 그대로 변화없이 선택되는 구조이며, Stride, Hybrid 의 경우는 등차를 이용하여 데이터가 변하는 특성을 갖고 있다. 이러한 분석에 의해서 Last,



2Level

그림 5. 부분 태그의 변화에 따른 정확도

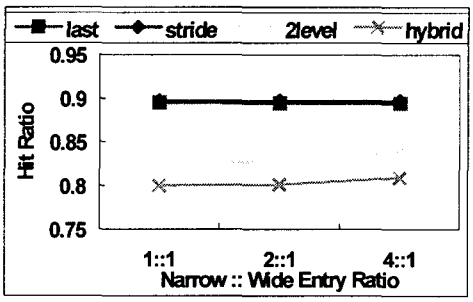
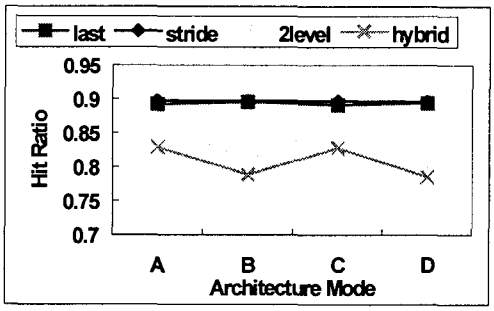


그림 6. 엔트리 비율에 따른 정확도

의 경우 한번 선정된 데이터는 거의 변화가 없으며, Stride, Hybrid 의 경우도 그 등차가 크지 않은 경우 그 변화폭이 매우 적다는 특성이 있다. 이때 하나의 연산결과가 나타내는 데이터 값을 표현하기 위한 최소비트는 항상 프로세서의 최대 정확도를 갖는 레지스터 비트와 동일하지는 않다. 예를 들어 0 ~ 255 까지는 8 비트이면 충분히 표현이 가능하기 때문에 이 경우는 데이터의 표현을 위해서 32 비트(프로세서의 기본 비트로 가정할 때) 전체를 활용할 필요가 없는 것이다. 이런 가정하에 예측기 구조에서 작은 비트를 필요로 하는 데이터와 많은 비트를 필요로 하는 데이터를 따로 분리하여 작은 비트를 사용하는 엔트리에서의 데이터 표현공간을 감소 시킴으로서 전체적인 비용절감효과가 있음을 보이는 연구결과가 소개 되었다[7](그림 3 참조).

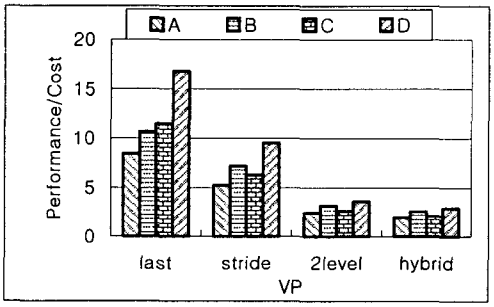
3.3 두 가지 방법의 최적결합 구조

본 연구에서는 이러한 두개의 방법들을 모두 적용하여 하나의 예측기에서 부분적 태그를 활용하는 방법과 비대칭성 데이터 표현크기를 갖는 2 개의 테이블을 갖는 구조를 제안한다. 그림 4 는 본 연구에서 제안되는 예측기 구조이다. 그림에서 태그부분이 크게 감소하였으며, 데이터 표현 테이블에서 적은 비트를 필요로 하는 엔트리에서의 데이터 표현 공간의 감소를 도식적으로 나타낸다. 동작원리는 다음과 같다.



● **예측기 참조과정** : 하나의 PC 어드레스로

그림 7. 예측기 구조의 변화에 따른 정확도



performance = hit_ratio*1000000

그림 8. 각 구조별 성능/비용

동시에 2 개의 테이블을 참조한다. 각각의 테이블에서 유효한 엔트리가 존재하는 경우, 예측값이 추출된다. 이 과정에서 하나의 PC 에 대해서 하나의 엔트리만 존재한다. 만약 작은 비트 사이즈를 나타내는 테이블에서 예측값이 추출되는 경우 원래의 비트로 확장하는 작업이 추가된다.

예측기 갱신과정 : 연산결과와 예측결과에 기반하여 테이블을 갱신한다. 연산결과의 유효 비트가 작은 비트만 필요로 하는 경우 narrow table 을 갱신하고, wide table 은 invalidate 시킨다. 만약 많은 비트를 필요로 하는 경우는 wide table 을 갱신하고, narrow table 은 invalidate 시킨다. 본 연구에서는 8-bit 를 기준으로 narrow 와 wide 로 구분하였다. 하나의 테이블을 갱신하고 다른 하나의 테이블의 엔트리를 invalidate 시키므로써 하나의 PC 에 해당하는 엔트리는 실제적으로 하나만 존재하게 된다.

이와 같이하여 태그부분과 데이터 엔트리 부분에서 비용을 동시에 감소시켜서 최소 비용을 갖는 예측기 구조를 설계할 수 있다.

IV. 시뮬레이션 결과 및 분석

4.1 시뮬레이션 방법

본 연구에서는 SPEC integer 95 의 8 개 벤치

표 2. SPEC integer 95 프로그램 및 입력 파일

Name	Input
Compress	bigtest.in
Gcc	reload1.i
Go	9stone21.in
Li	test.lsp
Perl	primes.pl primes.in
M88ksim	ctl.in
Ijpeg	specmun.ppm
Vortex	vortex.in

마크를 기준으로 평가하였다. 평가 모델은 A:{full-tag, one-mode}, B:{full-tag, two-mode}, C:{partial-tag, one-mode}, D:{partial-tag, two-mode}로 구분하였다. 이중 A 는 기존의 모델을 지칭하며, B 는 데이터 표현공간에서의 적은 비트 표현공간의 활용을, C 는 부분적인 태그만을 활용한 방법을 지칭한다. D 모델이 본 연구에서 제안된 구조에 입각한 최적화된 예측기 모델이다. 표 2 는 시뮬레이션에서 사용된 benchmark 와 입력 파일을 보여준다. 성능평가를 하기 위해서 처음 200M 개의 명령어는 무시하였고, 이후 50M 개의 명령어를 연산결과 예측 정확도를 기준으로 평가하였다.

4.2 분석

● 부분 태그 비트

그림 5 를 보면 4 비트의 부분 태그를 활용하는 경우에 원래의 full tag 를 활용하는 것과 거의 비슷한 성능을 보임을 확인할 수 있다. 결과적으로 4 비트의 부분 태그만 활용하면 되는 것을 알 수 있다.

● 엔트리 비율

two-mode 에서 두 개의 테이블에 존재하는 엔트리의 비율은 다를 수 있다. 1:1, 2:1, 4:1 의 경우를 평가한 결과 전반적으로 큰 차이가 없음을 알 수 있었다. 따라서 본 연구에서는 1:1 의 엔트리 비율을 활용하였다(그림 6 참조).

● 성능/비용 특성

결과적으로 상기 분석에 의해서 결정된 최적화된 구조는 4-bit 의 부분 태그, 1:1 의 엔트리 비율을 갖는 구조로 설정되었다. 한편 전체 엔트리의 개수는 2048 개로 한정하였다. 이런 모델을 기본으로 하여 제안된 방법과 기존의 방법들과의 성능/비용 특성을 분석한다.

우선 각 예측기 모델에서의 구조적인 변화를 평가한다. 그림 7 은 Last, Stride, 2Level, Hybrid 예측기에서의 4 가지 예측기 구현 구조의 변화에 따른 성능상의 변화를 나타낸다. 그림을 참조하면 본 연구에서 제안된 D:{partial-tag, two-mode} 모델에 의해서도 성능의 변화가 거의 없음을 알 수 있다(5%이내의 감소). 표 3 은 현재까지 결정된 여러 가지 구조적 변수들을 기준으로 A,B,C,D 구조의 비용을 계산한 것이다.

표 3. A,B,C,D 구조의 bit cost

	A	B	C	D	(A-D)/A %
Last	106,496	83,968	77,824	53,248	50%
Stride	172,032	124,928	143,360	94,208	45%
2level	354,304	256,000	325,632	225,280	36%
Hybrid	423,936	301,056	395,264	270,336	36%

VHTentry = 2048, Narrow:Wide = 1:1

Narrow_Data_bit = 8, Partial_Tag_bit = 4

그림 8 은 각각 4 가지 모델의 성능/비용 특성 변화를 도식화한 것이다. 전반적으로 제안된 모델 D(partial-tag, two-mode)는 모든 예측기 구현 구조에서 가장 높은 성능/비용 특성을 보임을 확인 할 수 있었다.

V. 결론

본 연구에서는 고성능 프로세서의 성능향상의 이론적 한계인 명령어간 종속성 문제를 해결하는 연산결과 예측 방법의 실제 구현과정에서의 높은 비용문제를 해결하는 방법에 관하여 분석하고 새로운 방법을 제안하였다. 제안된 방법은 기존의 구조에 비해서 비용은 36%~50%정도로 크게 감소시키면서도 성능 하락은 5%이내 정도로 거의 없다는 것을 확인 할 수 있었다. 이로써 고성능 프로세서의 성능향상의 이론적 한계를 극복하는 실제적인 구현방법을 찾게 되었다.

Acknowledgement

본 연구는 광주과학기술원 초고속광네트워크 연구센터를 통한 한국과학재단 우수연구센터 지원금에 의한 것입니다.

참고 문헌

[1] D.W. Wall, "Limits of Instruction-Level Parallelism," *Proc. of 4th ASPLOS*, pp.248-259, 1991.

[2] M.H.Lipasti, and J.P.Shen, "Exceeding the Dataflow Limit via Value Prediction," *Proc. of 29th MICRO*, pp.226-237, 1996.

[3] M.H.Lipasti, C.B.Wilkerson, and J.P.Shen, "Value Locality and Load Value Prediction," *Proc. of 7th ASPLOS*, pp.138-147, 1996.

[4] A. Mendelson and F.Gabby, "Speculative Execution based on Value Prediction," *Technical Report EE Dept. TR1080*, Technion-Israel Institute of Technology, Sept., 1997.

[5] K. Wang and M. Franklin, "Highly Accurate Data Prediction using Hybrid Predictors," *Proc. of 30th MICRO*, pp.281-290, 1997.

[6] Sato. T and Arita, I., "Partial Resolution in Data Value Predictors," *Proc. of ICPP*, pp.69-76, 2000.

[7] Toshinori Sato and Itsujiro Arita, "Table Size Reduction for Data Value Predictors by exploiting Narrow Width Values," *Proc. of ICS*, pp.196-205, 2000.