Analyzing Branch Prediction Contexts Influence

Marius Oancea, Arpad Gellert, Adrian Florea, Lucian Vintan

University "Lucian Blaga" of Sibiu, Computer Science Department,
E. Cioran 4, 550025 Sibiu, Romania, E-mail: marius.oancea@ulbsibiu.ro

ABSTRACT

All present branch prediction techniques are limited in their accuracy. Our aim is to demonstrate that an important limitation cause is given by the used prediction contexts (global and local histories respectively path information). Using these dynamic contexts, some branches are unbiased and randomly shuffled, therefore unpredictable. The percentages of these branches represent a fundamental prediction limitation. For outperforming this limitation it is necessary additional more relevant context information, in order to further reduce the entropy of these branches.

KEYWORDS: Branch Prediction, Prediction Context, Unbiased Branches, SimpleScalar, SPEC Benchmarks

1 Introduction

The main aim of this paper is to analyze the present day branch prediction contexts from the point of view of their limits in predicting unbiased braches. We vary the contexts length and observed that some of dynamic contexts remained unpredictable despite of their length (32 bits of local history concatenated with 32 bits of global history). In our study we will ignore technology constraints (memory limits, predictors structures and algorithms) being only interested if the context information used by the state-of-the art predictors is or is not sufficient. The main idea is: in a perfect dynamic context all branch instances should have the same outcome. If the outcome is not the same a first solution might consists in extending the context information. Different context sizes and different context information will be studied during this article.

2 Methodology

2.1 Defining the metrics and ranges of parameters

To have a metric of that distribution we define polarization index (P) of a certain branch context as:

$$P(S_i) = \max(f_0,\ f_1) = \begin{cases} f_0, & f_0 \geq 0.5 \\ f_1, & f_0 < 0.5 \end{cases} \qquad (1)$$

where:

- $S = \{S_1,\ S_2,\ ...,\ S_k\}$ = set of distinct contexts that appear during all branch instances;

- k = number of distinct contexts, $k \le 2^p$, where p is the length of the binary context;
- $f_0 = \frac{T}{T+NT}$, $f_1 = \frac{NT}{T+NT}$, NT = number of "not taken" branch instances corresponding to context S_i, T = number of "taken" branch instances corresponding to context S_i, $(\forall) i = 1, 2, ..., k$, and obviously $f_0 + f_1 = 1$;
- if $P(S_i) = 1, (\forall) i = 1, 2, ..., k$, then the context S_i is completely biased (100%), and thus, the afferent branch is highly predictable;
- if $P(S_i) = 0.5, (\forall) i = 1, 2, ..., k$, then the context S_i is totally unbiased, and thus, the afferent branch is not predictable if the taken and not taken outcomes are shuffled.

The context information we use will be the global history (GHR), local history (LHR) and later the path information.

We will start with a GHR_Len =0 bits and we will increase it up to 32 bits. For LHR the initial value will be 16 and will be increased up to 32.

A new metric is introduced in order to measure how much prediction accuracy earned if all branches in unbiased branch context instances will be perfectly predicted. If this metric (noted T) is below 1%, by perfectly solving all branches in unbiased contexts will not provide a better prediction accuracy then T (under 1% in this case).

$$T = \frac{NUB_i}{NB_i} = 0.01 \qquad (2)$$

where NUB_i is the total number of unbiased branch context instances on benchmark i, and NB_i is the number of dynamic branches on benchmark i (therefore, the total number of branch context instances).

In our statistics we considered six SPEC2000 benchmarks [2]. For each of them only conditional branches were considered. The first 300,000,000 instructions were ignored. Next 1,000,000,000 were passed trough the statistics reporter we created using SimpleScalar framework [3].

2.2 The approach

The main idea is to start with a shorter context and make a list of unbiased randomly shuffled branches. Gradually this list is shortened by increasing the context length and reapplying the algorithm. Figure 1 presents this approach as was applied. We considered a branch context as unbiased if its corresponding P<0.95. We start to analyze to polarization using GHR_Len=0 and LHR_Len=16. We continue by counting how many dynamic branches are found in unbiased contexts (sumIk) and we divide this number by the total number of dynamic context instances founded in the current simulation session (NDyn). For sumIk/Ndyn > T the algorithm is continued until remaining number of unbiased branches becomes insignificant.

Summarizing the algorithm in pseudocode we have:

```
contextsConfigurations=16_0, 16_16, 20_20, 24_24, 28_28, 32_32
compute NDyn = count (all conditional branches)
branchesToConsider = all conditional branches
foreach ctxCfg in contextsConfigurations
      set1 = find contexts having polarization rate in [0.5-0.95)
      set2 = find contexts having polarization rate in [0.95-1.0)
```

```
      sumIK = branchCount( set1)

      persist res = sumIK/NDin
      // if res > 0.01 makes sense to continue

      branchesToConsider = branches in (set1)
      // we consider branches in set2 solved
endforeach
```

We applied the algorithm for each benchmark in the test set.

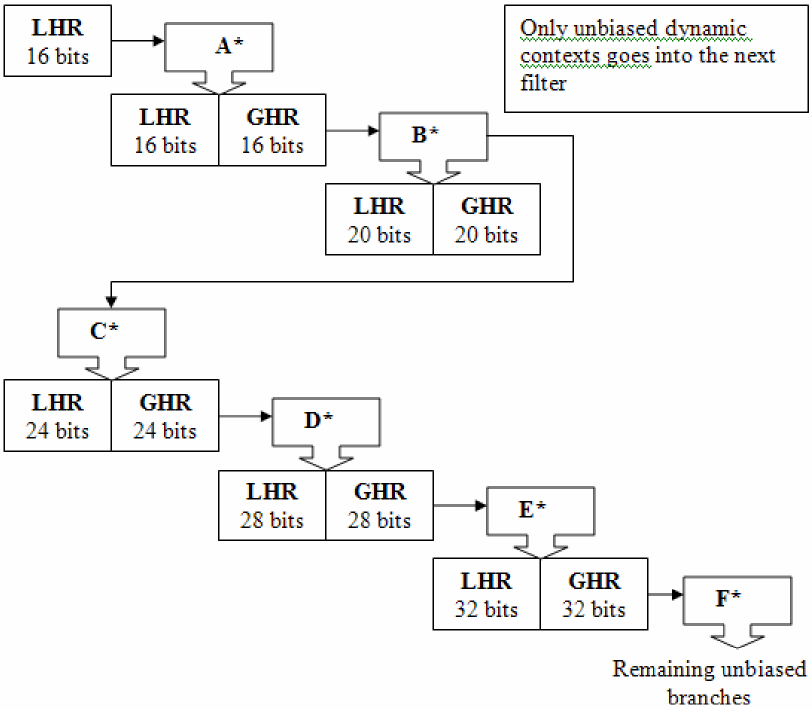

Figure 1. Finding difficult branches – Iterative approach

3 Experimental Results

In this section, we give the results of our analysis on the used SPEC 2000 benchmarks.

Algorithm step	Step formalization	Unbiased branches
A*	A* = Local16	24.56%
B*	B* = Local16Global16(A*)	12.39%
C*	C* = Local20Global20(B*)	7.80%
D*	D* = Local24Global24(C*)	5.27%
E*	E* = Local28Global28(D*)	3.71%
F*	F* = Local32Global32(E*)	2.58%

Table 1. Centralized results created by averaging the data

As we can see from the above table, for a long history (32 bits GHR and 32 bits LHR) the ultimative prediction accuracy is 97.42%. By using a different chain of steps and a more sophisticated chaining method, smaller number of unbiased context will be obtained (e.g. Instead of having B* = Local16Global16(A*) we can have B* = Local16Global4(A*) and C* = Local16Global16(B*). Therefore the chain A* B* C* D* E* F* is definitely suboptimal.

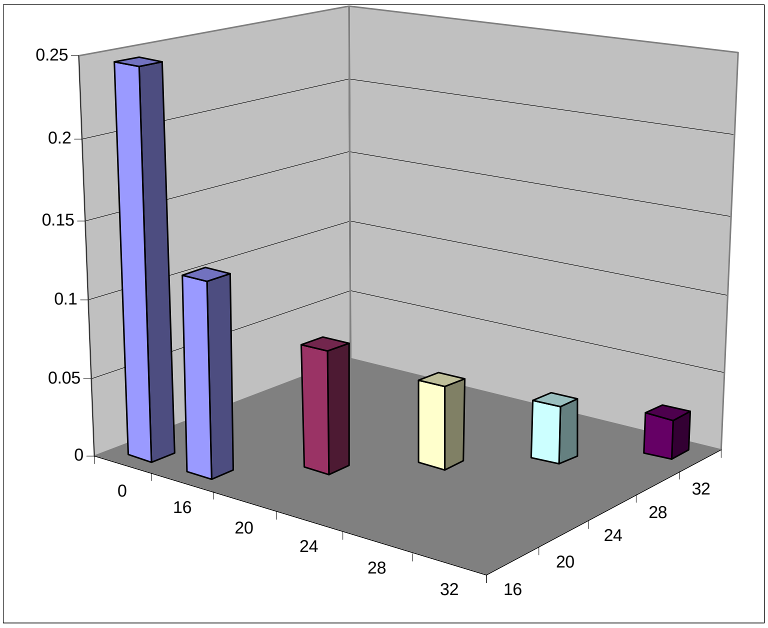

Figure 2. Evolution of averaged unbiased list size

Most of the present-day predictors cannot use very long contexts and also cannot use dynamic reconfigurable contexts length to get the full advantages of the iterative approach At this moment, our scientific hypothesis is that path information might further reduce the unbiased branches number [1]. As a consequence, we start our analysis by adding the last branch PC to the B* context. Without adding this path information the percentage of biased branches is 87.60665. After adding path of size 1 the percentage of biased contexts increased to 87.6277. The percentage of biased contexts is not significant higher that the one without any path. By increasing path length to 3, the percentage increases to 87.66439. Even increasing the path length to 20 will not help very much.

4 Conclusions

The results presented here lead us to the conclusion that the path is only useful in case of short contexts. Long contexts seem to disperse and compress very well the path information. For us, finding a new relevant context to aggressively reduce the number of unbiased shuffled branches remains an open problem.

Acknowledgments

This work was partially carried out under the HPC-EUROPA project (RII3-CT-2003-506079), with the support of the European Community - Research Infrastructure Action under the FP6 "Structuring the European Research Area" Programme.

References

[1] Nair R., Dynamic Path-Based Branch Correlation, IEEE Proceedings of MICRO-28, 1995.

[2] SimpleScalar tool set, http://www.simplescalar.com/

[3] SPEC, The SPEC benchmark programs, http://www.spec.org.