EVALUATING EFFECT OF OPTIMIZATION LEVEL ON VALUE PREDICTABILITY [a]

Kiichi Sugitani, Akihiko Hamano, Toshinori Sato and Itsujiro Arita
Department of Artificial Intelligence, Kyushu Institute of Technology
E-mail:{kiichi,akihiko,tsato,arita}@mickey.ai.kyutech.ac.jp

The practice of speculation in resolving data dependences based on value prediction has been studied as a means of extracting more instruction level parallelism. There are many studies on value prediction mechanisms with high predictabilities. However, to the best of our knowledge, the relationship between compiler optimization level and value predictability has not been investigated. In this paper we evaluate value predictability of several binaries which are compiled with different optimization levels. Detailed simulations reveal that value prediction is still effective for highly optimized binaries.

1 Introduction

In order to improve performance, modern microprocessors rely on exploiting instruction level parallelism (ILP). Recently, the practice of speculation in resolving data dependences has been studied as a means of extracting more ILP. An outcome of an instruction is predicted by value predictors [1, 2]. The instruction and its dependent ones can be executed simultaneously, thereby exploiting ILP aggressively. There are many studies on value prediction mechanisms with high predictabilities and on reducing their hardware cost. However, to the best of our knowledge, the relationship between compiler optimization level and value predictability has not been investigated. Someone might expect that sophisticated optimizations obviate value prediction. Therefore, in this paper we evaluate value predictability of several binaries which are compiled with different optimization levels.

2 Evaluation Methodology

We use a functional simulator for evaluating value predictability. We implemented the simulator using the SimpleScalar tool set (ver.2.0) [3]. The SimpleScalar/PISA instruction set architecture (ISA) is based on MIPS ISA. In this paper, we investigate last-value predictor [1] and hybrid predictor [2]. The former represents simple predictors and the latter does complex ones. The configuration of the hybrid predictor is based on [2].

Seven programs from eight SPEC95 CINT benchmarks are used for this study. The input files are modified so that evaluation time is practical. Each program is executed to completion. The binaries evaluated in this study are

[a] This work is supported in part by the grants from Japan Society for the Promotion of Science and from Fukuoka Industry, Science & Technology Foundation.

distributed by University of Michigan. They were compiled by GNU GCC (version 2.7.2.3) and MIRV [4]. For each compiler, three optimization levels, `-O0`, `-O1`, and `-O2`, were performed. In general, MIRV `-O0` executes considerably less instructions than GCC `-O0`, because MIRV has graph coloring allocation and copy propagation in `-O0` while GCC has no register allocation in `-O0` [4]. For the remaining optimization levels, the two compilers have almost equal abilities.

3 Simulation Results

In this section, we present simulation results. We use predictability, prediction coverage, and prediction accuracy as metrics for evaluation. We define the predictability as the number of instructions that are (correctly and incorrectly) predicted by a value predictor over that of all register-writing instructions. The prediction coverage is defined as the number of instructions correctly predicted over that of all register-writing instructions. The prediction accuracy is the percentage of instructions correctly predicted over all predicted instructions.

Figure 1 shows simulation results, when the last-value predictor is utilized to GCC binaries. The horizontal and vertical axes denote the optimization levels and the percentage respectively. Each bar, divided into two parts, indicates the predictability and the prediction coverage. The lower part (black) indicates the percentage of the instruction whose data value is correctly predicted. The upper part (gray) indicates the percentage that is mispredicted. That is, the lower part is the prediction coverage and the sum of the two parts is the predictability. For each group, bars from left to right indicate the results when the value predictor has 1024-, 2048-, 4096-, and 8192-entry tables, respectively. We can find the followings. First, in the case of `129.compress`, both the predictability and the prediction coverage are considerably reduced when the optimization level changes from `-O0` to `-O1`. This might mean that relatively high optimizations obviate the value prediction. Second, different from the previous observation, both the predictability and the prediction coverage are slightly improved as the optimization level becomes higher in the cases of the remaining programs. This denies the first investigation. Third, the efficiency of the value prediction changes little when the optimization level changes. Therefore, we can confirm that the conservative last-value predictor is still effective for highly optimized binaries.

Figure 2 shows simulation results, when the hybrid predictor is utilized to GCC binaries. The characteristics similar to the last-value predictor case is observed. However, two programs should be mentioned. In the case of `129.compress`, the difference in the predictability and the prediction coverage between `-O0` and `-O1` is considerably smaller than the last-value predictor case. In the case of `134.perl`, both the predictability and the prediction coverage

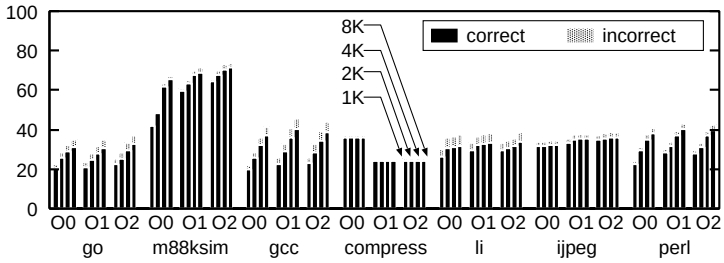

Figure 1: Influence of optimization in GCC on last-value predictors

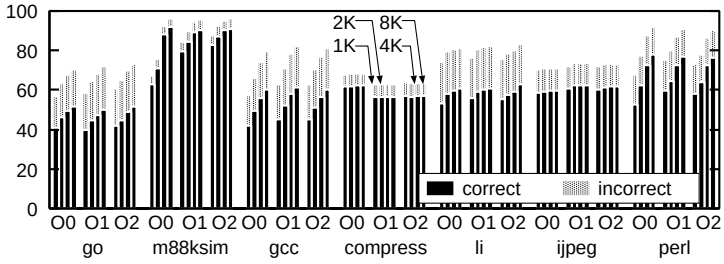

Figure 2: Influence of optimization in GCC on hybrid predictors

are slightly reduced as the optimization level becomes higher. However, these observations does not require the different conclusions from the last-value predictor case. In general, both the predictability and the prediction coverage hardly changes when different optimization levels are considered. And thus, the aggressive hybrid predictor is also effective for highly optimized binaries.

Figure 3 shows simulation results, when the last-value predictor is utilized to MIRV binaries. The different characteristics from GCC binaries can be observed. First, we can find that in general both the predictability and the prediction coverage are greater in the case of MIRV than in the case of GCC. This can be observed throughout programs and optimization levels. Second, it is generally found that both the predictability and the prediction coverage decrease slightly as the optimization level becomes higher. This difference from the GCC case can be found especially in the case of `099.go`. However, the decrease is not as large as that can be found in GCC `129.compress`. Thus, we can conclude that the conservative last-value predictor is still effective for MIRV binaries.

Figure 4 shows simulation results, when the hybrid predictor is utilized to MIRV binaries. The characteristics similar to the last-value predictor case is observed, while `099.go` shows considerably different behavior. Both the predictability and the prediction coverage, which are greater than in the GCC

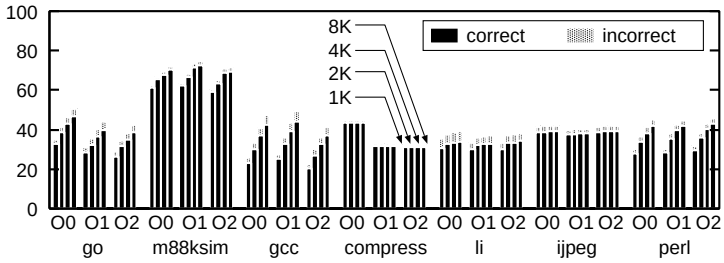

Figure 3: Influence of optimization in MIRV on last-value predictors

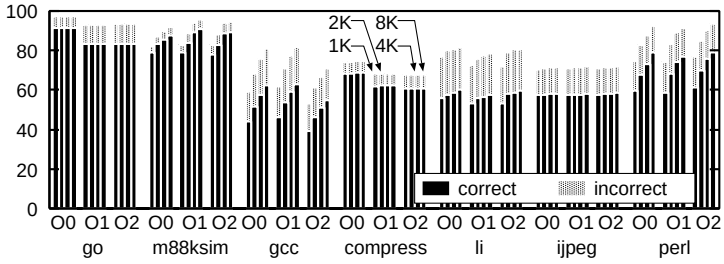

Figure 4: Influence of optimization in MIRV on hybrid predictors

case, are reduced slightly when higher optimization levels are applied. In addition, for `099.go` both the predictability and the prediction coverage are considerably reduced when the optimization level changes from `-O0` to `-O1`. However, we can conclude that the aggressive hybrid predictor is also effective for highly optimized binaries.

4 Conclusion

This paper evaluated value predictability of GCC and MIRV binaries which are compiled with different optimization levels. From detailed simulation results, we have found that both for GCC and MIRV the value prediction is still effective for binaries compiled with every optimization level.

References

1. M.H.Lipasti, J.P.Shen, "Exceeding the dataflow limit via value prediction", *29th Int'l Symp. on Microarchitecture*, 1996.
2. K.Wang, M.Franklin, "Highly accurate data value prediction using hybrid predictors", *30th Int'l Symp. on Microarchitecture*, 1997.
3. D.Burger, T.M.Austin, "The SimpleScalar tool set, version 2.0", *ACM SIGARCH Computer Architecture News*, vol.25, no.3, 1997.
4. M. Postiff, et al., "The MIRV SimpleScalar/PISA compiler", *Technical Report CSE-TR-421-00*, University of Michigan, 2000.