International Journal of Soft Computing 5 (6): 206-210, 2010
ISSN: 1816-9503

A Relearning Virtual Machine for Hot Method Prediction

Sandra Johnson and Valli Shanmugam
Department of Computer Science and Engineering, College of Engineering,
Anna University, Guindy, Chennai 600025, Tamil Nadu, India

Abstract: Predicting hot methods of a program code using machine learning algorithms in compiler optimization eliminates the overhead incurred during runtime identification. Since learning is a continuous process, the system should be able to relearn and update itself. In this study we implement this idea in a virtual machine which learns and relearns as to how hot methods can be effectively predicted in a program. This is the first attempt to make a compilation system relearn about hot method prediction after each execution. By applying relearning we are able to develop models for the prediction of the frequently called and the long running hot methods that can obtain 19 and 37% accuracies showing an improvement of 10 and 21%, respectively over the corresponding models without relearning. This is due to the ability of the model to learn from every program that enters execution and reconstruct itself.

Key words: Relearning, hot method prediction, virtual machines, SVM, frequently called hot methods, long running hot methods

INTRODUCTION

Hot method identification by profiling has long been used by compilers and virtual machines. Profiling is accurate but it incurs a lot of expensive overhead. To overcome this problem of runtime overhead, researchers are looking for innovative techniques for the identification of hot methods. Although, many compiler writers endorse the use of Machine Learning (ML) techniques in the construction of compiler heuristics, the technique has not found its application in hot method prediction. The Support Vector Machine (SVM) based hot method prediction model constructed in the previous research is a promising first step in this direction (Johnson and Valli, 2008a, b).

The performance of the predictive model is convincing enough to improve program runtime optimization and now that learning is a continuous process, we want the learnt model to relearn and update itself.

In this study we implement the effect of learning and relearning on the prediction of hot methods by a Low Level Virtual Machine (LLVM) (Lattner and Adve, 2004). The two models, one for predicting the Frequently Called Hot Methods (FCHM) and another for the Long Running Hot Methods (LRHM) are constructed using one set of benchmark programs. The hot methods in a new program from another benchmark program are predicted by the learnt system for optimization. The learnt model immediately relearns with the first set of benchmark programs along with the second one and a new predictive model is constructed. After predicting hot methods of a program, the system relearns with a new program. Thus, the predictive model keeps relearning and making new predictive models.

Literature review: Training and retraining have been used for selective learning of the predictive model for an online machine learning system (Ludl *et al*., 2008). An algorithm for lifetime learning is experimented in a neural network. Unlearning and relearning are done in two phases, a positive and a negative phase in a neural network (Carse and Oreland, 2000). A learning and a reverse learning algorithm is used in boltzman machines (Hinton and Sejnowski, 1986).

In a relearning approach to improve the classification accuracy in the k-nearest neighbor classifiers, it has been shown that the relearning combined with ensemble computing works better than relearning and other conventional methods (Ishii *et al*., 2008a, b). Application of relearning to improve the classification accuracy of the k-nearest neighbor classification algorithm is also seen in the research on Nearest Neighbor Classification by Relearning (Ishii *et al*., 2009).

SYSTEM OVERVIEW

Figure 1 shows the system overview of the learning and the relearning hot method predictive system. The relearnt hot method predictive models for the frequently called and the long running hot methods are constructed using the SVM-based model (Johnson and Valli, 2008a, b).

Corresponding Author: Sandra Johnson, Department of Computer Science and Engineering, College of Engineering, Anna University, Guindy, Chennai 600025, Tamil Nadu, India

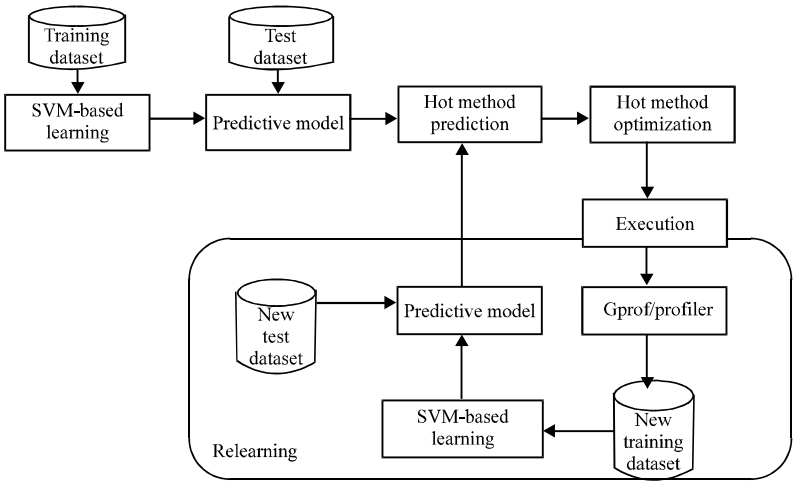

Fig. 1: Relearning virtual machine-system architecture

These predictive models are used to predict hot methods in programs which are optimized before execution. The feedback through profiling and gprof tools from the execution of the program is used to evaluate the prediction accuracy of the models. The feedback obtained is also used by the relearning system in the reconstruction of the predictive models. Thus, after the prediction, optimization and execution of every new program, the predictive model is reconstructed by the relearning system.

The predictive model is trained with all the sets of programs in a benchmark suite. That is the feature vector set is constructed using one full benchmark suite. A total of ten features are used in developing the FCHM predictive model whereas twenty nine features are used for the LRHM predictive model. These feature sets represent the effective feature sets constructed in the previous work using the sequential backward elimination process coupled with a knock-out strategy (Johnson and Valli, 2008a, b).

Once trained, both the predictive models are used to predict the hot methods in a new benchmark suite. This new prediction experience is compared with the actual hot methods during execution to evaluate the Hot Method Prediction Accuracy (HMPA). The system now starts its relearning process and constructs a new training set by appending the training feature vector constructed from the new benchmark program. The new training data set is then used to construct two new predictive models, one for the FCHM and another for the LRHM. The system unlearns and relearns in the process of making a new predictive model. The new predictive model could be used for predicting any new benchmark program. Thus, the model learns from every new program that enters the system for execution. This research is the first attempt to apply relearning in virtual machines. The limitations of the system are:

- An offline relearning happens throughout the lifetime of the system and this makes it difficult for real time and online systems to use the system
- The system unlearns and then relearns instead of incremental learning or updating the learnt system. This kind of relearning after unlearning consumes time but it can be done as a background process in an online system

LEARNING AND RELEARNING BY THE PREDICTIVE MODEL

The ten and twenty nine static features that are used in the construction of the respective predictive models for the FCHM and the LRHM are collected from each method by an offline static analysis of the LLVM's bytecode. These features form the feature vectors that are accumulated in the training data set file. The feature vector is labeled +1 for the hot methods and -1 for the cold methods. Profiling and gprof tool are used, respectively for the identification of the FCHM and LRHM.

EVALUATION METHODOLOGY

The effects of learning and relearning of the virtual machine are evaluated on six different combinations of three benchmark suites namely, SPEC (2000), UTDSP (Lee, 1998) and Mediabench (Lee *et al*., 1997). The predictive model constructed using one benchmark suite is used to predict the hot methods of the other two benchmark suites. The prediction accuracies obtained are

the outcome of the initial learning on the first benchmark suite. Next, the predictive models are subjected to a relearning process. The relearning system is evaluated by using various combinations of the three benchmark suites. After the initial learning of the predictive models by one of the benchmark suites, programs from another benchmark suite are used as testing programs for hot method prediction. The predicted hot methods are optimized and then executed. The prediction results are compared with the actual hot methods generated by a profiler for FCHM and the gprof tool for LRHM. Next, the test benchmark programs are added to the existing set of training benchmark programs and the predictive models are retrained. Thus new predictive models are constructed with the old and the new benchmark programs. When a new program enters the system for execution, the existing predictive model predicts the hot methods in the program and after execution, the predictive models are reconstructed using the additional information obtained from the new program. The training and evaluation methodology adopted in the present study, trains the predictive model in one benchmark suite and tests using another benchmark suite which is an improvement over the leave-one-out evaluation strategy within a benchmark suite used in the previous research (Johnson and Valli, 2008a, b).

RESULTS OF RELEARNING OF THE PREDICTIVE MODELS

Figure 2-4 show the LRHM prediction accuracies obtained for the individual programs of the benchmark suites using the initial training prior to relearning. For instance Fig. 2 shows HMPA% obtained on the SPEC benchmark using the predictive models trained by either UTDSP or Mediabench. The observations of initial learning are presented as HMPA% without relearning in Table 1. In the results with FCHM predictive model, the prediction accuracies are small and the observations of the initial model are shown in Table 2 as HMPA% without relearning.

Relearning of FCHM: Table 2 shows the data on the performance of the relearning predictive models for the FCHM. It is seen that with the models for the FCHM, the accuracy of prediction is low before and after relearning compared with the prediction accuracies of 68 and 16% got in the previous research where the UTDSP and SPEC benchmark programs are evaluated using the leave-one-out methodology within a benchmark suite (Johnson and Valli, 2008a, b). However, the relearning system shows a consistent prediction with five out of six benchmark combinations indicating overall improvement. When individual predictive models are considered, the model

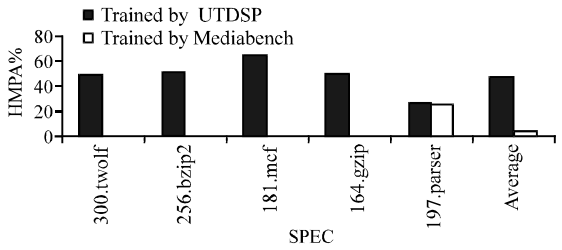

Fig. 2: Hot method predictions on SPEC benchmark suite by models trained by UTDSP and Mediabench

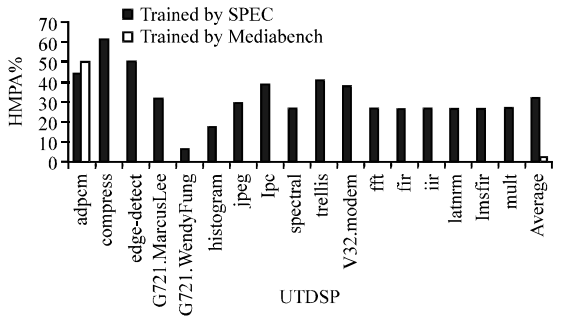

Fig. 3: Hot method predictions on UTDSP benchmark suite by models trained by SPEC, Mediabench

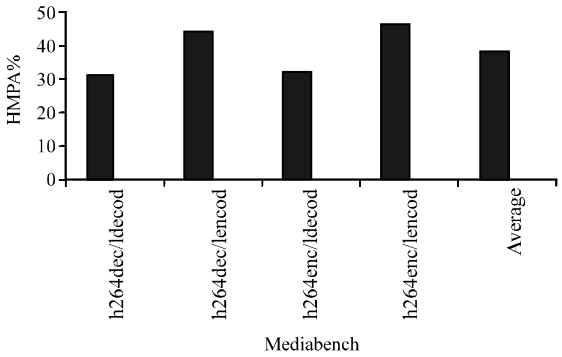

Fig. 4: Hot method predictions on Mediabench benchmark suite by models trained by SPEC, UTDSP

built by Mediabench can predict SPEC with 19% HMPA and UTDSP with 0%. The system when relearnt first by using UTDSP and later by SPEC benchmark suites can achieve 26 and 12%, respectively. It is an overall 10% improvement over prediction prior to relearning.

Relearning of LRHM: Table 1 shows a similar data on the relearning experience of the LRHM predictive model. It is seen that the highest LRHM prediction accuracies of 48 and 38% are obtained, respectively on benchmark suites SPEC and Mediabench when the model is initially learnt by UTDSP. Even though the same model predicts LRHM with a reasonably high prediction accuracy percentage of

Table 1: HMPA% and its improvement on relearning with various benchmark combination sequences for LRHM predictive model

Benchmark for initial learning	HMPA% without relearning		Relearning			Relearning			Overall	
	Benchmark	HMPA%	Benchmark 1	HMPA%	Improvement	Benchmark 2	HMPA%	Improvement	HMPA%	Improvement
UTDSP	SPEC	48	SPEC	31	-17	Media bench	31	-7	31	-12
	Media bench	38	Media bench	17	-21	SPEC	1	-47	9	-34
SPEC	UTDSP	32	UTDSP	63	31	Media bench	11	11	37	21
	Media bench	0	Media bench	11	11	UTDSP	0	-32	6	-11
Mediabench	SPEC	5	SPEC	31	26	UTDSP	31	28	31	27
	UTDSP	3	UTDSP	55	52	SPEC	11	6	33	29

Table 2: HMPA% and its improvement on relearning with various benchmark combination sequences for FCHM predictive model

Benchmark for initial learning	HMPA% without relearning		Relearning			Relearning			Overall	
	Benchmark	HMPA%	Benchmark 1	HMPA%	Improvement%	Benchmark 2	HMPA%	Improvement%	HMPA%	Improvement%
UTDSP	SPEC	6	SPEC	0	-6	Media bench	0	0	0	-3
	Media bench	0	Media bench	0	0	SPEC	11	5	6	3
SPEC	UTDSP	2	UTDSP	20	18	Media bench	0	0	10	9
	Media bench	0	Media bench	0	0	UTDSP	25	23	13	12
Media bench	SPEC	19	SPEC	10	-9	UTDSP	23	23	17	7
	UTDSP	0	UTDSP	26	26	SPEC	12	7	19	10

31% after relearning with SPEC as the first benchmark and Mediabench as the second, the system's overall prediction accuracy decreases by 17 and 7%, respectively for the SPEC and Mediabench benchmarks. The LRHM predictive model can achieve a maximum HMPA of 48 and 32% prior to relearning for the UTDSP and SPEC benchmark programs while the previous research has shown 86 and 48% on the same benchmarks when the leave-one-out methodology is used (Johnson and Valli, 2008a, b). In this research, we train the predictive model in one benchmark and test it on a different benchmark suite.

The highlight of the performance of the predictive model for LRHM is a model initially built by SPEC which can obtain a prediction accuracy of 32% on UTDSP and 0% on Mediabench prior to relearning. With the first relearning experience on UTDSP, it can achieve a HMPA of 63% and with subsequent relearning on Mediabench, it gives 11% leading to an overall HMPA of 37%. It is a 21% improvement over the system without relearning. Although, the model is initially trained with Mediabench and later relearnt in SPEC-UTDSP order and UTDSP-SPEC order shows the highest overall improvement of 27 and 29%, respectively the SPEC trained model when relearnt in UTDSP-Mediabench order gives the highest and the most consistent HMPA%. Unlike the high performing LRHM predictive model which invariably yields better prediction accuracies, the predictions of the FCHM model are small. Nevertheless, the FCHM predictive model also performs fairly well on relearning. It may be concluded that the hot method prediction keeps improving with machine relearning.

CONCLUSION

The research is an attempt to construct a relearning virtual machine which learns every time a new program enters the system for execution. The HMPA percentages obtained with some of the relearnt predictive models are very impressive clearly indicating that the models could achieve a reasonable improvement of 10 and 21%, respectively for the frequently called and the long running hot methods when relearnt using different combinations of SPEC, UTDSP and Mediabench benchmarks, over the systems without relearning. These results confirm the effective predictability of hot methods by machine learnt models previously observed in the research. The online systems can do the relearning as a background process. To address the problem of unlearning that precedes relearning, future research on relearning systems will concentrate on updating the predictive model instead of a routine unlearning and relearning.

REFERENCES

Carse, B. and J. Oreland, 2000. Evolution and learning in neural networks: Dynamic correlation, relearning and thresholding. Adaptive Behav., 8: 297-312.

Hinton, G.E. and T.J. Sejnowski, 1986. Learning and Relearning in Boltzmann Machines. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations, Rumelhart, D.E., J.L. McClelland and PDP Research Group, MIT Press, London, England.

Ishii, N., T. Yamada and Y. Bao, 2008a. Improved accuracy by relearning and combining distance functions. Lecture Notes Comput. Sci., 5178: 926-933.

Ishii, N., T. Yamada and Y. Bao, 2008b. Text classification by relearning and ensemble computation. Stud. Comput. Intell., 149: 217-226.

Ishii, N., Y. Hoki, Y. Okada and Y. Bao, 2009. Nearest neighbor classification by relearning. Lecture Notes Comput. Sci., 5788: 42-49.

Johnson, S. and S. Valli, 2008a. An approach to predict hot methods using support vector machines. Proceedings of the 16th International Conference on Advanced Computing and Communication, Dec. 14-17, Chennai, pp: 27-31.

Johnson, S. and S. Valli, 2008b. Hot method prediction using support vector machines. Ubiquitous Comput. Commun. J., 3: 37-37.

Lattner, C. and V. Adve, 2004. LLVM: A compilation framework for lifelong program analysis and transformation. Proceedings of the International Symposium on Code Generation and Optimization, March 20-24, Chicago, IL, USA., pp: 75-86.

Lee, C., 1998. UTDSP benchmark suite. http://www.eecg.toronto.edu/~corinna/DSP/infrastructure/UTDSP.html.

Lee, C., M. Potkonjak and W.H. Mangione-Smith, 1997. MediaBench: A tool for evaluating and synthesizing multimedia and communication systems. Proceedings of the 30th International Symposium on Microarchitecture, Dec. 1-3, Research Triangle Park, NC. USA., pp: 330-335.

Ludl, M.C., A. Lewandowski and G. Dorffner, 2008. Adaptive machine learning in delayed feedback domains by selective relearning. Applied Artificial Intell., 22: 543-557.

SPEC, 2000. The standard performance evaluation corporation. http://www.specbench.org.