2014 Peking University Young Scientists Symposium on Informatics

A Memory Resource Prediction Method Based on Machine Learning

Gaoxiang Zhang

School of Electronics Engineering and Computer Science, Peking University
1100012899@pku.edu.cn

Abstract

✓**The para-virtualization architecture of Xen results in the incapability of acquiring memory demand. The method equivalent to software simulation is widly used but gains high overhead. In this paper, we propose a low cost method which uses machine learning to build the memory resource prediction model via getting performance counter. This method can integrate with other optimizing approach and reduce the overhead.**

Introduction

✓Virtualization enables multiple operating systems to run on their own *virtual machine* (VM) separately and efficiently for users. One of the challenges is effective management of shared physical resources, which has a premise of the precise and low-cost measurement of resources.

✓As a representation of para-virtualization, Xen modifies the kernel of operating system and realizes a light software layer namely *virtual machine monitor* (VMM). Owing to no host OS participation in resource management, VMs are almost directly upon hardware like real systems, leading to high performance. However such fidelity causes the VMM is unaware of resource demand of VMs and thus unable to conduct management.

✓To be informed of the usage, the wide-spread method [1] is to simulate the page scheduling process in real time. However, this method results in high overhead. Our work is motivated by the algorithms in machine learning. We hope to use machine learning to predict memory resource precisely and efficiently.

Proposed

✓Feature and Algorithm Selection

- Low cost performance monitoring counter (PMC) that the number is limited and is most related to the variation trend of working set size (WSS):
  L1 cache accesses, L2 cache misses, L3 cache misses, TLB misses
- Regression model is trained by Lasso Algorithm.

✓Limitations of existing methods

- System rss stat: low accuracy
- Pin: quite high overhead
- Original MRC based WSS tracking [1]: high overhead
- 'Intermittent Memory Tracking' [2]: only good to long and flat phases

✓Main Idea

To prevent excessive tracking, our idea is to intermittently open tracking and dynamically change the model, then we use the sub-model to predict the WSS in a sub region. Because of the stable in-phase character, this idea is more suitable. Tracking interval is the key parameter. The first strategy sets a fixed interval. The second strategy uses dynamic interval. With an initial interval, when meeting a tracking point, the algorithm will test the gap between real data and predict data. If the gap is too large, the algorithm will decrease the interval in order to collect more neighbor data to correct the model. Or the gap is small enough, which means the model is stable, so the frequent tracking is not useful and the interval will increase.

✓ Self Adaptive Algorithm

```
SELF_ADAPTIVE_ALGORITHM(trainx,trainy):
1   new testx[], testy[], result[], model
2   set warm_up, interval, threshold_max, threshold_min, delta
3   initialization
4   for i <- 0 to warm_up
5      do testy[i] = trainy[i]
6           testx[i] = trainx[i]
7          result[i] = trainy[i]
8   model = Lasso(testx,testy)
9   for i <- warm_up to length(trainx)
10     do result[i]=model.predict(trainx[i])
11          if i%interval = 0
12             then do if error(trainy[i],result[i])>threshold_max
13                  then do interval = interval - delta
14             else if error(trainy[i],result[i])<threshold_min
15                  then do interval = interval + delta
16             model = Lasso(testx,testy)
```

✓Optimization

Firstly, the negative data should go to zero. Secondly, we judge a 'noise' if its value is far larger or smaller than its neighbor, these noises should be eliminated and go to the mean value of its neighbor. Thirdly, we directly regard data on tracking as the prediction result but not regression value.
It is obvious that neighbor data may affect more on prediction, so we add the weight of closer data by simply setting a limited FIFO auxiliary set with latest data. When a training data arrives, it will be added to both training set and this auxiliary set, with the old data in auxiliary dequeuing (if needed). And the regression use both sets to train the model.

Evaluation

✓Evaluation criteria

$$Tracking\ rate = \frac{tracking\ intervals}{total\ intervals} \times 100\%$$

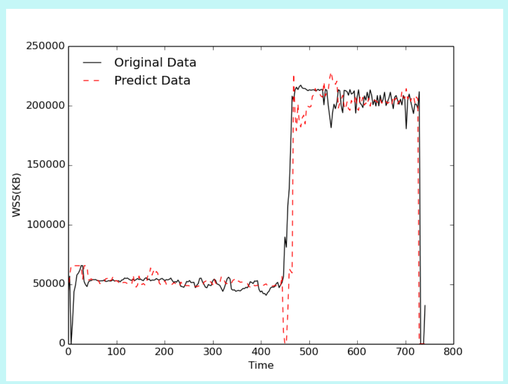
$$Error\ rate = \frac{|predict\ value\ - real\ value|}{max(predict\ value, real\ value, 1)} \times 100\%$$

✓Testing Sets

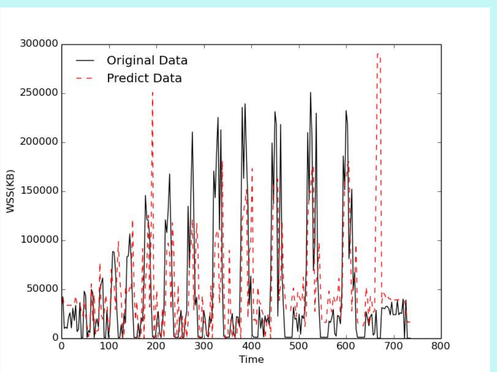
All 29 standard benchmarks in SPEC CPU 2006
Manmade stage program

✓Experimental Results

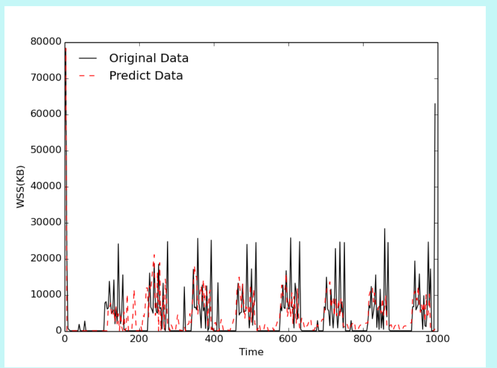
Benchname	Error Rate	Tracking Rate
400.perlbench	31.4%	8.1%
401.bzip2	43.2%	18.8%
403.gcc	30.0%	7.4%
410.bwaves	27.0%	11.1%
416.gamess	10.9%	5.4%
429.mcf	18.5%	6.3%
433.milc	6.0%	4.6%
434.zeusmp	16.3%	5.4%
435.gromacs	1.1%	6.1%
436.cactusADM	5.5%	4.4%
437.leslie3d	0.6%	4.9%
444.namd	29.3%	15.6%
445.gobmk	24.9%	9.1%
447.dealⅡ	49.5%	21.1%
450.soplex	9.0%	6.0%
453.povray	5.4%	9.2%
454.calculix	39.4%	22.2%
456.hmmer	20.2%	11.2%
458.sjeng	3.3%	4.6%
459.GemsFDTD	1.0%	4.1%
462.libquantum	1.5%	5.0%
464.h264ref	27.6%	10.2%
465.tonto	42.4%	21.4%
470.lbm	0.4%	6.2%
471.omnetpp	5.8%	4.5%
473.astar	18.9%	5.5%
481.wrf	13.8%	5.2%
482.sphinx3	5.1%	6.0%
483.xalancbmk	15.5%	6.4%
Mean of SPEC	9.60%	7.61%
Mean of rss	34.20%	N/A
Stage Program	12%	5.20%



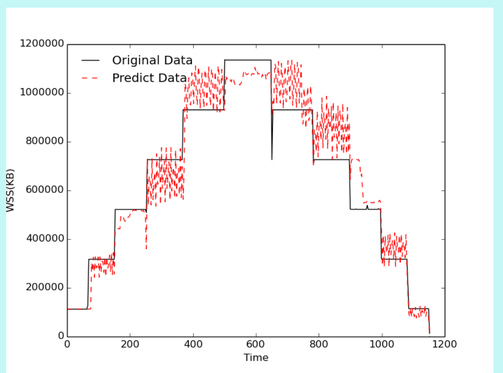
450 with stable phases



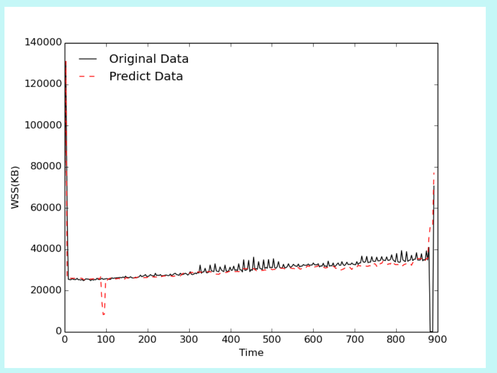
447 with wavy wss



465 with periodic wavy phases



Stage Program



482 with slope phase

Conclusion

✓**We propose an approach using machine learning to reduce the frequency of original tracking and acquire relatively precise working set size value. It overcomes the defects of existing methods with the assist of raw data and Lasso algorithm. By testing on SPEC, we verify that our method can synthetize existing optimizing techniques to gain much better performance.**

Reference

[1] Mattson RL, Gecsei J, Slutz D, Traiger IL. Evaluation techniques for storage hierarchies. IBM System Journal. 1970;9(2):78–117.
[2] Zhao W, Jin X, Wang Z, Wang X, Luo Y, Li X. Low cost working set size tracking. In Proceedings of the 2011 annual conference on USENIX Annual Technical Conference. 2011.

Acknowledgement

✓This work is supported by the Peking University Principal Undergraduate Research Foundation (URTP2013PKU004).
✓My sincere thanks go to my advisors: Dr. Yingwei Luo, Dr. Xiaolin Wang and Dr. Zhenlin Wang.
✓Thank all members in GIS lab for their help.