Predicting x86 Program Runtime for Intel Processor

Behram Mistree, Hamidreza Hakim Javadi, Omid Mashayekhi
bmistree, hrhakim, omidm@stanford.edu

Introduction

In this Project we try to predict the runtime for Intel processor x86 loop free programs.

As a reference, we use a tool called IACA provided by Intel that given a set of loop free instructions would return back the number of cycles it takes to run them over the processor. In fact this black box resembles the internal structure of an Intel processor and takes into account the effects of parallelism and pipelining.

Our goal is to simulate the behaviour of this black box with machine learning techniques.

Feature Collection

We need to capture the read/write conflicts between the instructions. This way we can capture whether instructions can run in parallel or not.

```
program 1 :
movl $0x40, %rax
addl  $0x40, %rbx, %rbx
```

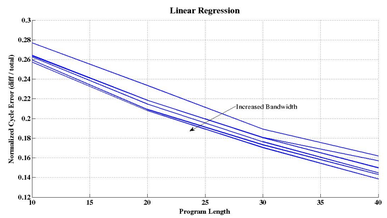
```
program 2 :
movl $0x40, %rax
addl  $0x40, %rax, %rax
```

$$\phi(I_{m+1},\tau,P)=\begin{bmatrix} C(I_{m+1-\tau})\times\mathbf{1}\{I_{m+1-\tau}\rightarrow I_{m+1}\} \\ C(I_{m+2-\tau})\times\mathbf{1}\{I_{m+2-\tau}\rightarrow I_{m+1}\} \\ \vdots \\ C(I_m)\times\mathbf{1}\{I_i\rightarrow I_{m+1}\} \\ C(I_{m+1}) \end{bmatrix}$$

Fixed Length Programs

- Linear Regression

Collect the features as explained for each instruction for programs with length L, and learn number of cycles it takes to run the program from the aggregate feature vector containing all the features from all the instructions.

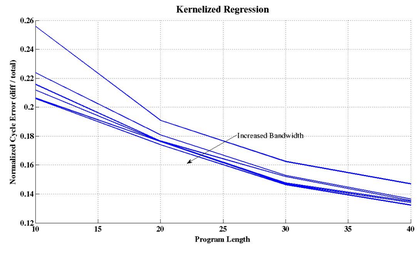


$\gamma^{pre} = (X^TX + \lambda I)^{-1}X^TY$

- Kernel Regression

In order to capture the non linear dependencies between feature vectors and labels we tried a kernelized regression with exponential kernel.

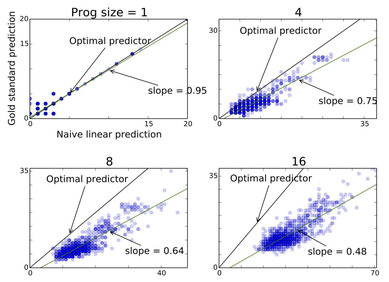
K: Training Kernel Matrix
$c = (K+ \lambda I)^{-1}Y$
$k(x^{new})_i = k(x^{new},x_i)$
$\gamma^{pre} = c^Tk(x^{new})$



Variable Length Programs

- Naive Generative Model

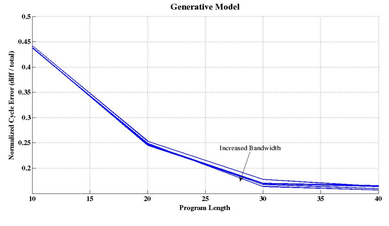
Simply for each program add the number of cycles it takes to run each instruction independently. Since we are not considering any possible parallelism the prediction is far from reality



- Generative Model

For a set of programs with length (bw+1) learn the difference in number of cycles that takes to run the program w and w/o the lat instruction based on the feeatures collected for the last instruction.

Then for a new program the predicted runtime would be the sum of predicted cycles for each instruction.



- Implementation and data collection
  - we have written a couple of thousand lines python code to collect the feature vectors based on read write conflicts for a series of instruction.
  - To evaluate under real circumstances, we have disassembled gcc and gdb and used them as input data for testing and training set.
  - There are a series of matlab scripts for feature collection and learning algorithms, for future extensions and reusability.