ARPE

Analyzing the relationship between parameters and effectors

Martina Maggio, Henry Hoffmann
Lund University, University of Chicago
Motivation

Controlling computing systems is messy

Often you hear “modeling that system is too complicated, therefore...”
Motivation

And it’s actually true

Depending on the model level, creating a model for a computing system might be really complicated
Motivation

Example:

Suppose we want to model power consumption of a software piece given the datasheet power consumption of the instruction set. We need to know that the software code translates into a set of atomic operations.
Motivation

This might be really “complicated”

However, often it is much easier to grasp the relationship between the variable that can be changed and some final effect
Motivation

This is a first step to respond to the question of how to automatically control the system.
Outline

- Methodology
- Experimental evaluation
- Conclusion and future work
Methodology

ideally:

something that generates one or more models

what we can change

what we can measure
Methodology

what we can change

what we can measure

CG
Methodology

what we can change

CG

what we can measure

configuration generator
Configuration generator

Selects a set of configurations given the parameter list and their values

Automatically generates scripts to execute experiments on the system to build the model
Methodology

what we can change

CG

DC

data collector

what we can measure
Data collector

Executes experiments on the system collecting relevant data automatically
Methodology

what we can change

CG  DC  DA

what we can measure

data analyzer
Data analyzer

Given the collected data generated many different models of the system and selects the less “error prone”
Data analyzer

- Single parameter linear regression to explain the relationship between one parameter and the “measured” effect
- Multiple parameters multivariate linear regression to build a comprehensive model
Execute **tomcat** from the DaCapo benchmark suite 50 times per configuration

- Parameters: number of threads (1 to 32), iterations (1 to 10)

- Model the real execution time of one instance

8 seconds to process the data, low errors
Experimental results

- Execution time
- Energy consumed
Execution time

- Comparison of models for execution time given performance counter or heartbeats
We measure the number of instructions retired per second with OProfile and the number of heartbeats retired per second for instrumented applications: swaptions and x264 from the PARSEC benchmark suite.
<table>
<thead>
<tr>
<th></th>
<th>Heart Rate</th>
<th>Instructions Retired</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Set</td>
<td>0.00027 [s]</td>
<td>0.00034 [s]</td>
</tr>
<tr>
<td>Test Set</td>
<td>0.00030 [s]</td>
<td>0.04130 [s]</td>
</tr>
<tr>
<td></td>
<td>Heart Rate</td>
<td>Instructions Retired</td>
</tr>
<tr>
<td>----------------</td>
<td>------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><strong>Training Set</strong></td>
<td>0.00494 [s]</td>
<td>0.00072 [s]</td>
</tr>
<tr>
<td><strong>Test Set</strong></td>
<td>0.00353 [s]</td>
<td>0.00072 [s]</td>
</tr>
</tbody>
</table>
Results insight:

There is no unique parameter to predict execution time, for some benchmark the heart rate does a better job, for others the number of instructions retired is more insightful.

Future work might be: how do we distinguish?
Energy consumption

Do we really need hardware power sensors or can we infer the power consumption of CPU-intensive applications based on the CPU consumption parameter?
Energy consumption

We measure the execution time, the power consumption and the CPU consumption of sunflow, treadbeans (from the DaCapo benchmark suite) and lookbusy
Energy consumption

![Energy consumption graphs]

- **Energy consumption in joules (J)**
  - Axes: #threads, energy [J]
  - Chart 1: Energy consumption increases with the number of threads.

- **CPU usage**
  - Axes: #threads, cpu [%]
  - Chart 2: CPU usage remains constant across different thread counts.

- **Power consumption**
  - Axes: #threads, power [W]
  - Chart 3: Power consumption increases with the number of threads.

- **Elapsed time**
  - Axes: #threads, elapsed time [s]
  - Chart 4: Elapsed time increases with the number of threads.

**Table 5:** Error (in joules) using different sensors, computing all the possible resources to compute and terminate it is confirmed that the race to idle approach allocation time nor the amount of consumed energy decreases. A model that can predict energy consumption is obviously more accurate given power information. Alternatively, APRE also identifies a model that is comparable in terms of accuracy to map the application. For example, h2 and xalan falls into the category where there is an optimal point in terms of energy consumption, while lusearch is similar to sunflow. The case of tradebeans is completely different. The model built using power consumption produces an error about 1% of the energy value. The model built with the CPU consumption produces an error of about 0.5%.

For more details, see the data in Table 5. The data for the benchmark is shown in Table 5. The error data is reported in Table 5. The model built using power consumption is obviously more accurate given power information. Alternatively, APRE also identifies a model that can predict the energy consumption with an error of about 0.5% of the energy value. The model built using power consumption is obviously more accurate given power information. Alternatively, APRE also identifies a model that can predict the energy consumption with an error of about 0.5% of the energy value.
Energy consumption

![Energy Consumption Graphs]

Figure 6: Energy results for lookbusy.

In that case, the analysis phase took 0.002 seconds. In this case, using the two models results in almost identical errors, therefore they can be both considered reliable. The results confirm what previously seen with the execution of real applications. The error percentage in this case is even more limited, and it is around 0.1%.

To summarize the contributions of this set of experiments, we used APRE to find that using power consumption for energy prediction is ideal. However, if one does not have any power consumption sensor, it is possible to use the percentage of CPU consumed to have an estimate of the energy consumed by the benchmark. With the entire set of benchmarks we experienced errors in the range 0.1–10%.

4.3 Map-Reduce applications

The last experiment uses the Apache Hadoop Map-Reduce framework. Many parameters can be set when launching a map-reduce experiment. Among these, we focus on the number of mappers and the number of reducers. We use mrbench in a single node installation of Hadoop. We collect data about elapsed time for the benchmark execution varying the number of mappers and reducers.

First, we fix the number of reducers and use APRE to build a model from the number of mappers to the execution time. Then we use APRE to build a model that takes as input the number of reducers and estimates the elapsed time, keeping the number of mappers constant. Next, we use APRE to combine the two models and identify a best configuration candidate. We run a final experiment when we simultaneously change the number of mappers and reducers and we use APRE to perform multivariate linear regression and build the model from the two parameters.

The range of variation of both the parameters—the number of mappers and reducers—is from 1 to 5, which is reasonable given the execution on a single node. Every experiment with a fixed number of workers is repeated keeping this number fixed in all the possible combinations. Therefore in the first experiment the number of reducers is fixed to 1 and the number of mappers varies in the whole range. In the second experiment the number of reducers is 2 and so on. The number of input lines to be processed is 10. For each experiment, the number of iterations is 10. When the number of reducers is in the set \([1, 2, 3]\), the best value is reached when two mappers are activated. If the number of reducers is 4 or 5, the number of mappers that ensures the fastest execution is 1. Whichever is the number of mappers, the best configuration is always reached when only one reducer is spawned. The multivariate linear regression reports an error of 1.6165 [s] when building the model with both the number of mappers and reducers. Empirically, the best configuration is two mappers and one reducer. Figure 7 shows the average data over the experiment.

Concluding, we wanted to show how APRE can be used when dealing with Map-Reduce parameters. In line with the current research trends, we also recorded the CPU consumption within the execution. Since the experiment is on the same benchmark, intuitively, the same usage pattern appears. One could use APRE to collect data and analyze the similarity of different CPU traces.

5 Conclusion and future work

In this paper we presented APRE, a tool for automating large experimental campaigns involving multiple parameters and to derive models from the obtained data. We developed the tool starting from our needs and we introduced different situations in which it proved useful.
## Errors

<table>
<thead>
<tr>
<th></th>
<th>power &amp; time</th>
<th>CPU &amp; time</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunflow</td>
<td>16.10 [J]</td>
<td>287.90 [J]</td>
</tr>
<tr>
<td>treadbeans</td>
<td>59.99 [J]</td>
<td>126.03 [J]</td>
</tr>
<tr>
<td>lookbusy</td>
<td>20.62 [J]</td>
<td>19.48 [J]</td>
</tr>
</tbody>
</table>
Note that all the sensors used are already coded in the released version, for example the tool automatically measures and stores CPU consumption during the benchmark execution without human intervention.
Conclusion

- We have developed a tool to build models based on the execution of applications on specific hardware.
- The tool is given a parameter list and reasonable values or intervals and executes the application with different configuration producing equation based models.
Future work

- Automatic controller generation, based on models
- Exploration of diagnostic features given the output of the tool - is self healing possible?
- Workload diversity checker
Questions?

Thanks for the attention

http://github.com/martinamaggio/arpe