A run-time generic decision framework for power and performance management on mobile devices

Martin Peres, Mohamed Aymen Chalouf, Francine Krief

LaBRI, University of Bordeaux

11/12/2014

Summary

Introduction

Modern devices have multiple ways of doing the same thing

- Multiple network interfaces (4G/Wifi/Ethernet);
- Multiple processor types (ARM's BIG.Little).

Problem: How to select the most efficient processor/interface?

- Each way can become the most efficient one;
- Static configurations cannot yield the best efficiency.

Introduction

Migrations/handovers possible

- BIG.Little processors: task migration in a few ms;
- Network: Live sockets migration possible using MPTCP.

Solution

- Need for a run-time decision engine for power management.

Challenges

- Need real-time requirements;
- Need to be able to compare arbitrary hardware models.

Summary

State of the art for run-time decision engines

Decision engines are (among others) used in

- Multi-homed devices;
- Workload-consolidations in data centres;
- Dynamic Voltage/Frequency Scaling for CPUs/GPUs.

Decision engines are mostly implemented with

- High-level languages such as Matlab;
- Linear-optimisation frameworks;
- Genetic algorithms.

Problems: Approaches hard to bound in

- CPU and RAM usage;
- Execution time.

State of the art for run-time decision engines

PISA

- Provides application-agnostic optimisation algorithms;
- Defines an interface between the user and the algorithms;
- Is not a framework and does not export debugging helpers.

Proposition: RTGDE provides

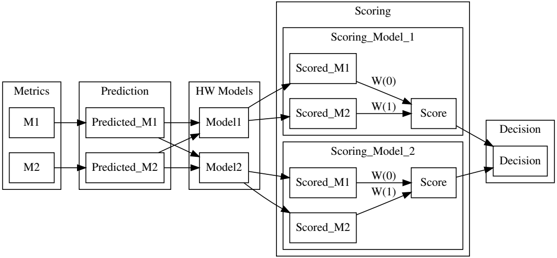
- A generic flowgraph for run-time decision making;
- A framework around this flowgraph written in C;
- A way to compare different pieces of hardware;
- Debugging helpers to understand decisions.

Summary

RTGDE : A generic Run-Time Generic Decision Engine

Abstract the decision process with RTGDE

- RTGDE proposes a 5-staged flowgraph:
  - Metrics: Describe the current state of the system;
  - Predictions: Foresee the evolution of the state of the system;
  - HW models: Propose a configuration and its impact;
  - Scoring: Evaluate the impact of the model on predictions;
  - Decision: Select the most efficient HW model.

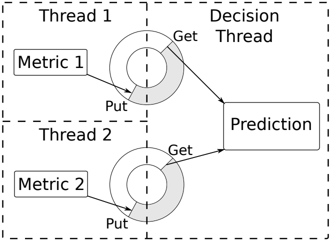


Decision Flowgraph

RTGDE : A generic Run-Time Generic Decision Engine

Metrics

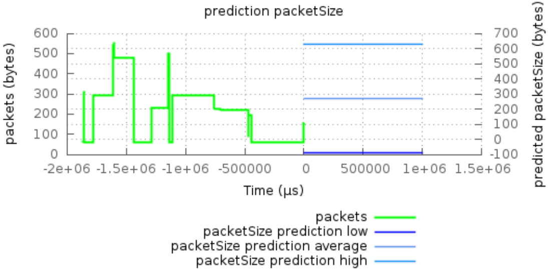
- Should represent accurately the state of the system;
- Should have an asynchronous data submission.



RTGDE : A generic Run-Time Generic Decision Engine

Predictions

- Predict the evolution of a metric or create a new one;
- Can take multiple metrics as an input;
- Can specify a constraint that the HW model should enforce;
- Store its output into a graph instead of a formula;
- Generate 3 outputs: The min., average and max. predictions.



RTGDE : A generic Run-Time Generic Decision Engine

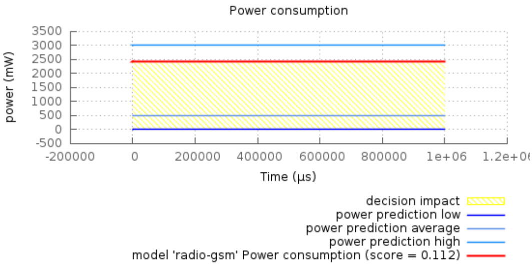
HW models

- Model the behaviour of a piece of hardware (Example: a NIF);
- Take as an input all the predictions/constraints;
- Propose a configuration for the HW modelled;
- Evaluate its impact on all the predictions/constraints;

RTGDE : A generic Run-Time Generic Decision Engine

Score

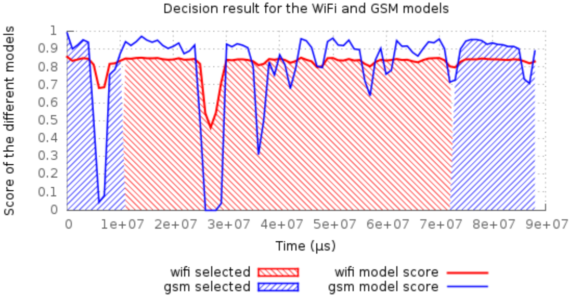
- Takes as an input the output of the HW models;
- Computes the score (between 0 and 1) of a HW model by scoring its impact on predictions. Default scoring method:
  - $score = \frac{\sum_{i=0}^{n} S_m(i)*W(i)}{\sum_{i=0}^{n} W(i)}$; $W(i)$: Weight of the prediction i
  - $S_m(i) = \frac{\int_{t=0}^{T} p_i(t).dt}{T}$; T: Timespan of the prediction
    $P_i(t)$: Probability of insufficient performance for prediction i



RTGDE : A generic Run-Time Generic Decision Engine

Decision

- Takes the output of all HW models and their scores;
- Calls a user-defined callback with the result of the decision;
- Can be implemented using a Finite State Machine (FSM).



Summary

Evaluations of the framework

Genericity of the framework

- Implements a generic flowgraph for decision-making;
- Demonstrated through 2 scenarios:
  - Network interface selection;
  - BIG.Little processor type selection.

Low RAM usage

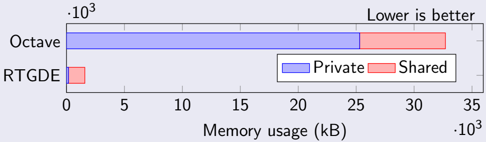


Figure: Minimum memory usage when using RTGDE or Octave to implement a decision engine

Evaluations of the framework

Low CPU usage

- Test performed on an Intel i7 860;
- Execution time of the processor selection flowgraph:
  - 43.95 μs (std = 13.19);
  - 0.004395% of the CPU time.
- Matlab's performance is lower than C++ [Andrews2012].

Real-time worthiness

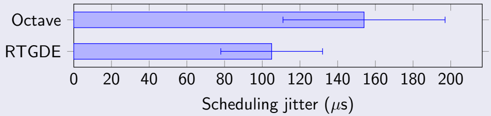


Figure: Jitter in the scheduling of decision process on Linux. Data collected during 30 seconds when the system was idle.

Summary

Conclusion

Advantages of the RTGDE framework

- Can execute multiple flowgraphs in parallel in real time;
- Ease debugging and introspection;
- Allow comparing the output of two models having the same interface $\Rightarrow$ Enable self-optimisation in heterogeneous devices.

Limits

- All the flowgraphs must be independent;

Future work

Future work

- Check the genericity of the flowgraph in more scenarios;
- Implement more prediction and selection algorithms;
- Write a website to allow researchers to share their HW models.

Thank you for listening! Any questions?