Energy Consumption Tools Pack

Leandro Fontoura Cupertino, Georges DaCosta,
Amal Sayah, Jean-Marc Pierson

IRIT – Toulouse Institute of Computer Science Research
UPS – University of Toulouse (Paul Sabatier)

GreenDays @ Luxembourg – January 28-29th 2013

Outline

Introduction

Motivation

- Why to measure and monitor the power consumed by an application?

Introduction

Motivation

- Why to measure and monitor the power consumed by an application?
- It changes over time according to the hardware used.
- Example: screensavers

Technology	CRT	LCD/LED
Lifetime	Enhance	Same
Monitor Energy	Save	Same
CPU Energy	Consume	Waste
GPU Energy	Consume	Waste

Introduction

Motivation

Power consumption on a node/PC is **application dependent**

Introduction

Motivation

Power consumption on a node/PC is **application dependent**

1. Data Centers Resource Management
   - DCs usually operates below full load
   - Workload classes differ depending on DC type

App Monitor → App Profiler → Resource Manager → Energy Efficiency

Introduction

Motivation

Power consumption on a node/PC is **application dependent**

1. Data Centers Resource Management
   - DCs usually operates below full load
   - Workload classes differ depending on DC type

   App Monitor → App Profiler → Resource Manager → Energy Efficiency

2. Software development
   - Power consumption of a same functionality varies according to
     - ★ Libraries
     - ★ Coding patterns
     - ★ Compilers
     - ★ Computer Architecture

Energy Consumption Tools Pack[2]

- Energy consumption library (libec)
  - Power sensors and estimators
  - Core library used in ectools
- Data Acquisition tool (ecdaq)
  - Collect data during benchmark execution
  - Create new power estimators
- Data Monitoring tool (ectop/ganglia plugin)
  - Lightweight monitor of power estimators and sensors
  - Monitor power consumption of application
  - Compare power estimators in any environment
- Energy profiler (valgreen) [1]
  - Software development

[1] Cupertino, L. F., DaCosta, G., Sayah, A., and Pierson, J.-M. Valgreen: an Applications Energy Profiler. SCSE'2013. To appear in March 2013.

[2] Avaiable under GPL3 licence.

Energy Consumption Library

Sensors

Definition: "a device that detects or measures a physical property and records, indicates, or otherwise responds to it" (Oxford dictionary)

- Direct measure (hardware):
  - E.g. wattmeter: measure node's electric power in watts
- Logical estimator (software):
  - Require one or more hardware sensors
  - E.g. process' wattmeter: estimates the process' power

Energy Consumption Library

Sensors

Hardware sensors[3]:

- Performance Counters
- ACPI Powermeter
- Grid'5000 PDU
- Networking

Software sensors:

- CPU Usage
- Memory Usage
- Inverse CPU PE
- MinMax CPU PE

[3] Testbed: notebook, G5K, RECS 510

Energy Consumption Library

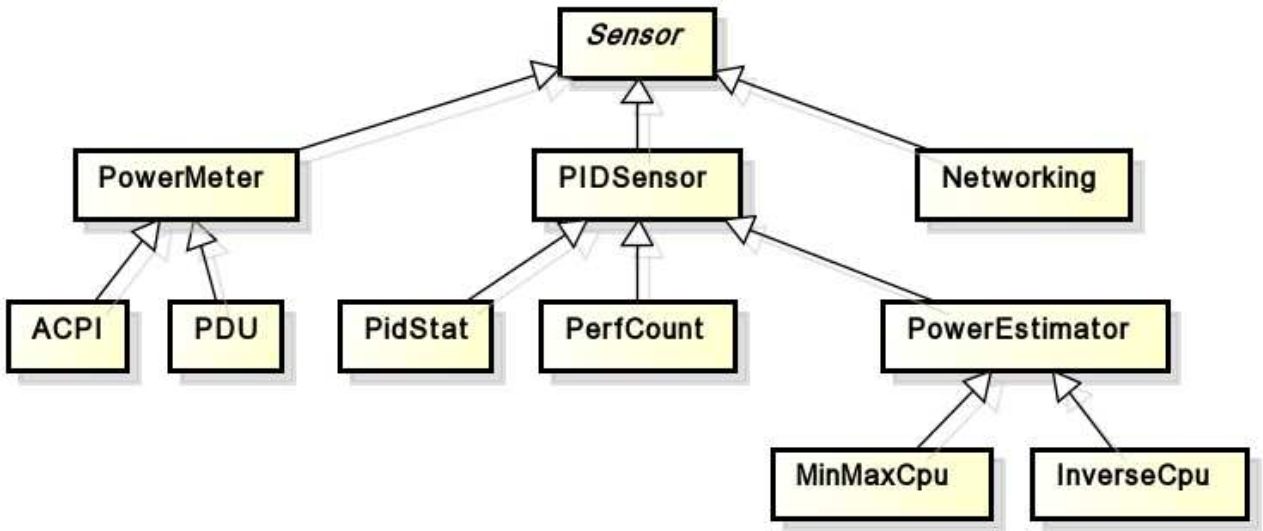
Sensors

Hardware sensors[3]:

- Performance Counters
- ACPI Powermeter
- Grid'5000 PDU
- Networking

Software sensors:

- CPU Usage
- Memory Usage
- Inverse CPU PE
- MinMax CPU PE



[3] Testbed: notebook, G5K, RECS 510

Energy Consumption Library

Application Power Estimators

- MinMax CPU

$$P_{pid} = (P_{max} - P_{min}) \times \frac{t_{cpu}^{pid}}{t_{cpu} + t_{idl}} + \frac{P_{min}}{|PR_t|}$$

- MinMax2 CPU

$$P_{pid} = \begin{cases} (P_{max} - P_{min}) \times \frac{t_{cpu}^{pid}}{t_{cpu}+t_{idl}} + \frac{P_{min}}{|PR_a|} & \text{if } pid \text{ is active} \\ 0 & \text{otherwise} \end{cases}$$

- Inverse CPU

$$P_{pid} = P_{PM} \times \frac{t_{cpu}^{pid}}{t_{cpu}}$$

Energy Consumption Library

Application Power Estimators

- Auto adaptive
  - Parameterization
    - ★ Statistical
    - ★ Metaheuristic
  - Function approximation
    - ★ Machine learning
- Workload dependent

How to Create a new sensor

libec

demos/ex1/MySensor.h

```
#ifndef MYSENSOR_H__
#define MYSENSOR_H__

#include <libec/sensor/SensorPid.h>

class MySensor : public cea::PIDSensor
{
public:
  MySensor();

  cea::sensor_t getValue(pid_t pid);

  void update(pid_t pid);
};

#endif
```

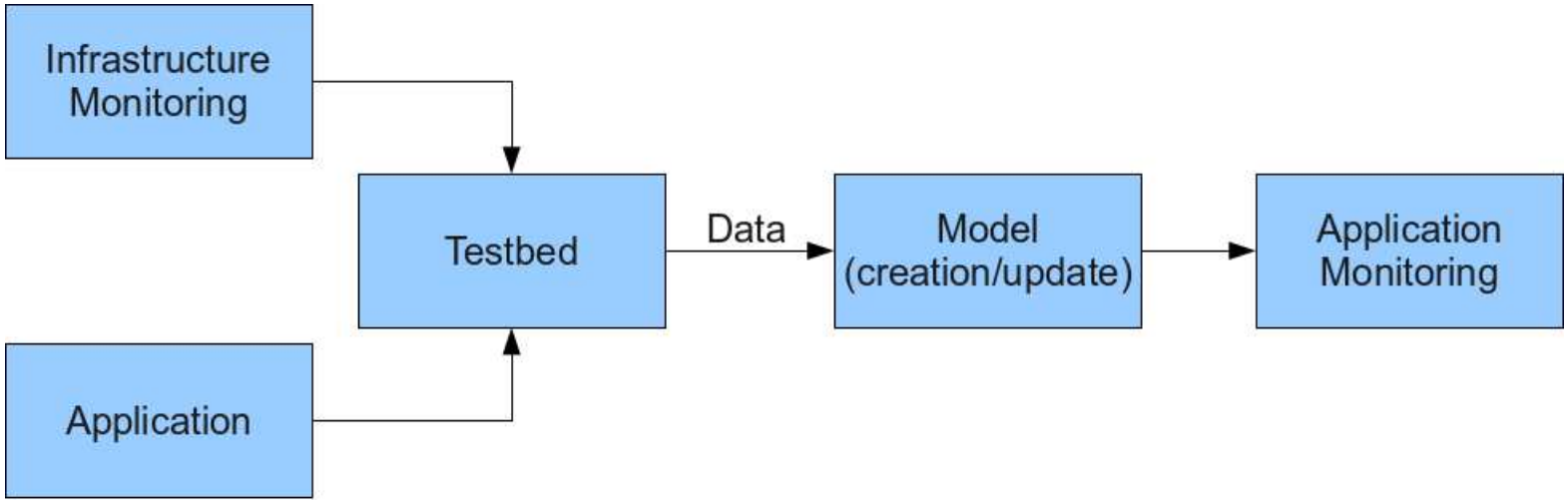
demos/ex1/MySensor.cpp

```
#include "MySensor.h"
#include <libec/tools/Tools.h>

MySensor::MySensor() {
  _name = "MySensor";
  _alias = "MS";
  _type = cea::U64;
  _isActive = true;
}

cea::sensor_t MySensor::getValue(pid_t pid) {
  update(pid);
  return _cValue;
}

void MySensor::update(pid_t pid) {
  _cValue.U64 = pid * cea::Tools::rnd(1, 10);
}
```

ecdaq

Data Acquisition Tool

- Environment



- Input: command line benchmark
- Output: gnuplot compatible file

ectop

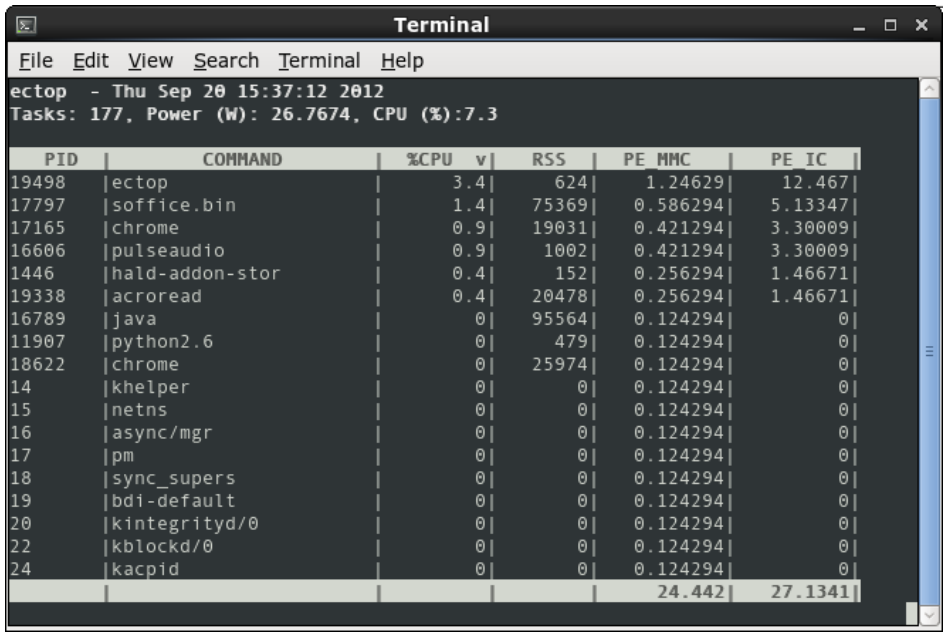
Data Monitoring Tool

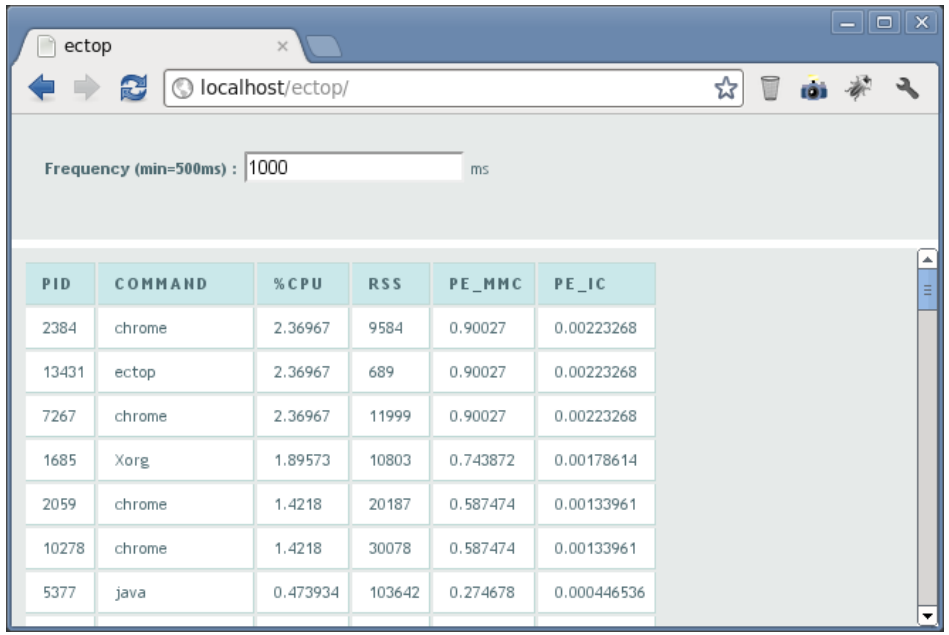
- Functionality
  - Sort data by columns (ascending/descendig)
  - Show accumulated values (sum bar)
  - Pan view
- Lightweight
  - Memory: 3Kb
  - CPU[4]:
    - ★ 0.3% (unsorted)
    - ★ 0.7% (sorted/sum bar)
    - ★ 2.0% (sorted/sum bar/html)
- Documentation available [html]
- Testbed: notebook, Grid5000, RECS Server Duo 510

[4] Data for a single core from top application for a Intel(R) Core(TM)2 Duo CPU T8300 @ 2.40GHz

ectop

Data Monitoring Tool





How to add a power estimator

ectop

src/ectop/main.c

```
...

void addSensors(MonitorEctop* m)
{
  m->addPowerMeter(new AcpiPowerMeter());

  m->addCpuSensor(new PidStat(PidStat::CPU_USAGE));

  m->addSensor(new PidStat(PidStat::CPU_USAGE));
  m->addSensor(new MinMaxCpu(new CpuElapsedTime(), 22, 55));
  m->addSensor(new MinMaxCpu2(new CpuElapsedTime(), 22, 55));
  m->addSensor(new InverseCpu(new AcpiPowerMeter(), new CpuElapsedTime()));
}

...
```

Valgreen

Energy Profiler

- Functionality
  - Modular power model
  - Power model calibration
  - Accurated energy profiler

- Default power model

$$P_{pid} = w_0 + w_1 \sum_{cpu} \frac{t_{cpu}^{pid}}{t_{cpu} + t_{idl}} + w_2 \frac{1}{|\mathbf{PR}|}, \quad \forall cpu \in \mathbf{C}$$

$$P_{node} = \sum_{pid} P_{pid}, \quad \forall pid \in \mathbf{PR}$$

$$P_{node} = w_0 |\mathbf{PR}| + w_1 \sum_{cpu} \frac{t_{cpu}}{t_{cpu} + t_{idl}} + w_2.$$

Valgreen

Energy Profiler

Calibration Process

- Reference value
  - Powermeter for data centers
  - ACPI for portable devices
- Issue: time between updates
  - Fine grained sampling
- Learning methodology
  - Filter data to be fitted
  - Linear Regression

Valgreen

Experimental Setup

- Applications: Linux stress microbenchmarks / LAPACK
- Power sampling frequency: 4Hz (1/250ms)
- Testbed: Dell Latitude D830 laptop
  - Processor: Intel Core 2 Duo T8300 at 2.40GHz
  - Memory: 2GB DDR2-667 SDRAM
  - Display: 15.4 Wide Screen WUXGA LCD
  - Disk: 120GB ST9120823ASG
- Hyperthreading: disabled
- DVFS: disabled

Valgreen

Calibration Experiment

- Calibration methodology: linear regression
- User's parametrization: $w_0 = 0$, $w_1 = P_{max} - P_{idl}$ and $w_2 = P_{idl}$

$$P_{pid} = w_0 + w_1 \sum_{cpu} \frac{t_{cpu}^{pid}}{t_{cpu} + t_{idl}} + w_2 \frac{1}{|\mathbf{PR}|}, \quad \forall cpu \in \mathbf{C}$$

Valgreen

Calibration Experiment

- Calibration methodology: linear regression
- User's parametrization: $w_0 = 0$, $w_1 = P_{max} - P_{idl}$ and $w_2 = P_{idl}$

$$P_{pid} = w_0 + w_1 \sum_{cpu} \frac{t_{cpu}^{pid}}{t_{cpu} + t_{idl}} + w_2 \frac{1}{|\mathbf{PR}|}, \quad \forall cpu \in \mathbf{C}$$

- Results

	w_0	w_1	w_2	MSE	R^2
calibrated	-1.13	17.16	32.60	8.91	0.9259
user defined	0.00	23.00	22.00	62.64	0.6637

Table: Comparison between calibrated and predefined parameters.

Valgreen

Power Sampling Frequency Experiment

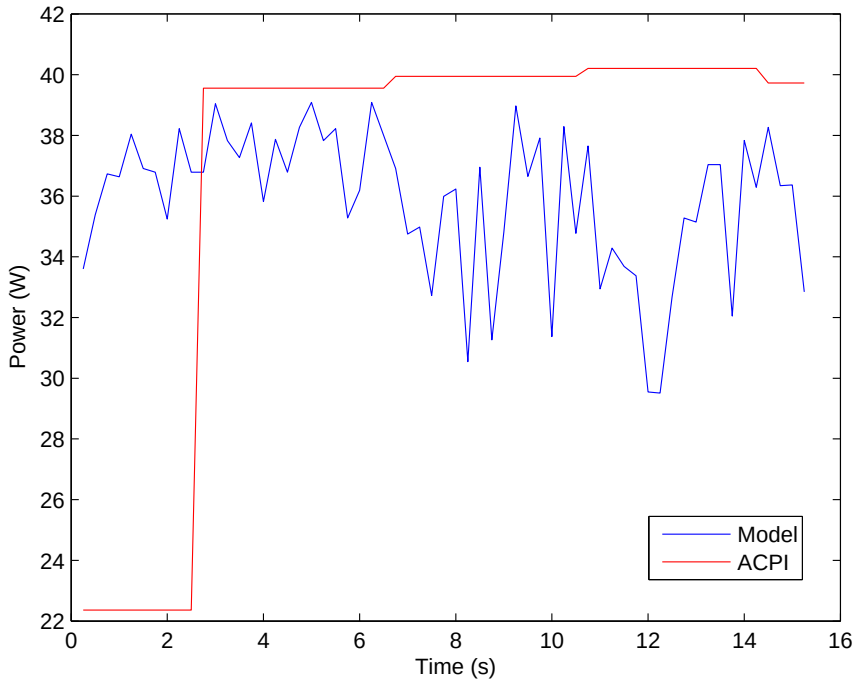


Figure: Comparison between ACPI power meter and the linear model

Valgreen

Compiler Optimization Experiment

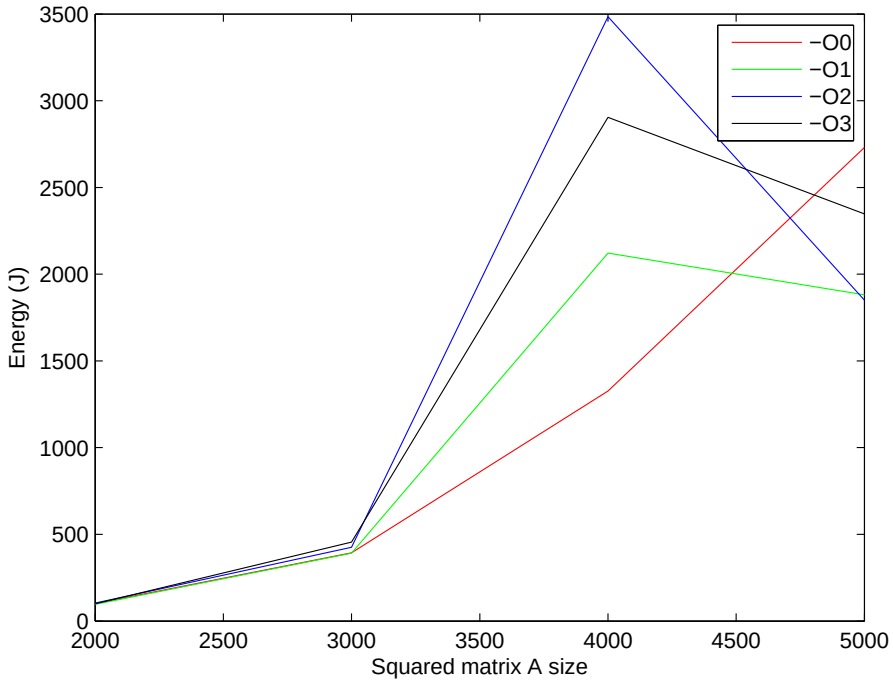


Figure: Energy dissipated to solve a linear system of equations using lapack

Conclusions and Future work

Conclusions

- Comparison of models can be done on real time
- Total energy consumption of application can be estimated in a specific hardware

Future work

- Implement new sensors and power estimators
- Develop auto adaptive power models

Acknowledge

This work was performed within the CoolEmAll[5] project.

- Members of the CoolEmAll consortium

[5] CoolEmAll (INFSO-ICT-288701) is a European Union co-funded project. ICT Theme: FP7-ICT-2011-7

Thank you for your attention.

fontoura@irit.fr
www.irit.fr/~Leandro.Fontoura-Cupertino