Refactoring for Energy Efficiency: A Reflection on the State of the Art



Gustavo Pinto   Francisco Soares   Fernando Castor

{ghlp, fmssn, castor}@cin.ufpe.br

Infra x Apps

Infra x Apps

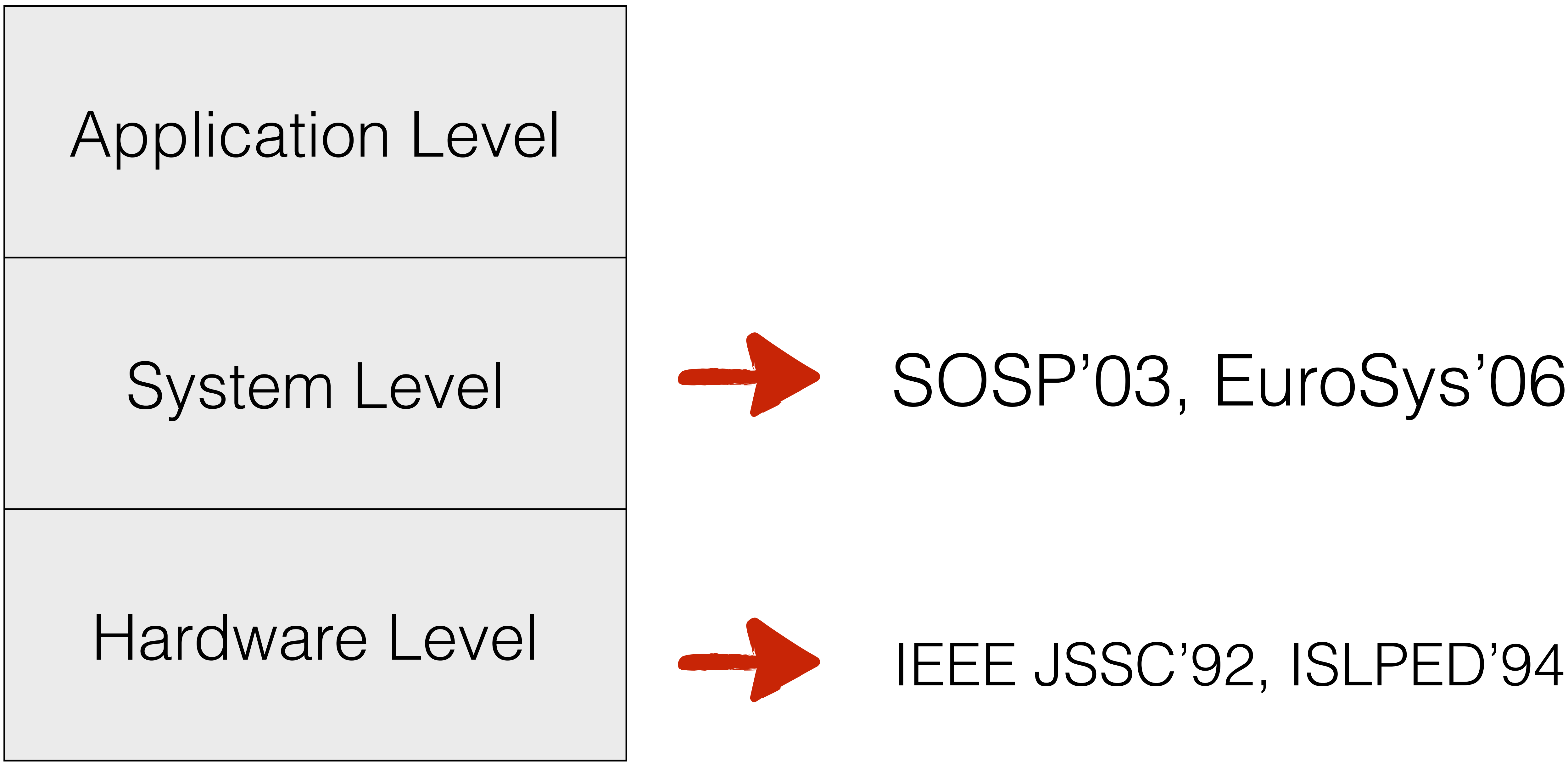
Application Level

System Level

Hardware Level

IEEE JSSC'92, ISLPED'94

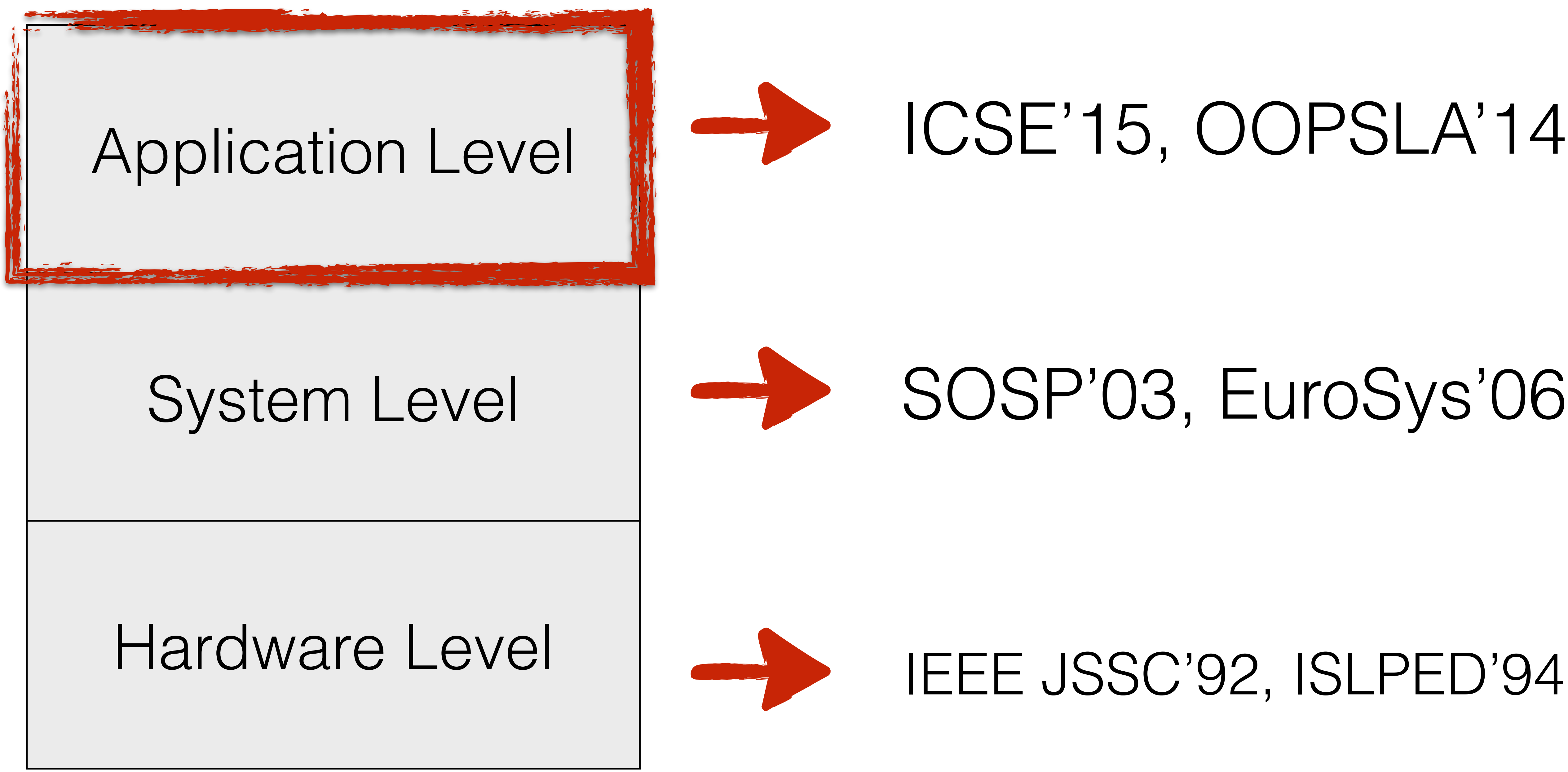
Infra x Apps


Application Level
System Level
Hardware Level
SOSP’03, EuroSys’06
IEEE JSSC’92, ISLPED’94

Infra x Apps

Application Level	→ ICSE'15, OOPSLA'14
System Level	→ SOSP'03, EuroSys'06
Hardware Level	→ IEEE JSSC'92, ISLPED'94

Infra x Apps


Application Level
ICSE’15, OOPSLA’14
System Level
SOSP’03, EuroSys’06
Hardware Level
IEEE JSSC’92, ISLPED’94

What is the problem?

I have no idea on how to improve this parallel code to be more energy efficient :(

What is the problem?

I have no idea on how to improve this parallel code to be more energy efficient :(

Is there any tool that can help us to improve our system to consume less energy?

There is a lack of tools for

- Measurement
- Identify opportunities
- Refactoring & Reengineering
- Testing & debugging

G. Pinto, F. Castor, and Y. D. Liu. Mining questions about software energy consumption. In MSR, pages 22–31, 2014.

There is a lack of tools for

- Measurement
- Identify opportunities
- Refactoring & Reengineering
- Testing & debugging

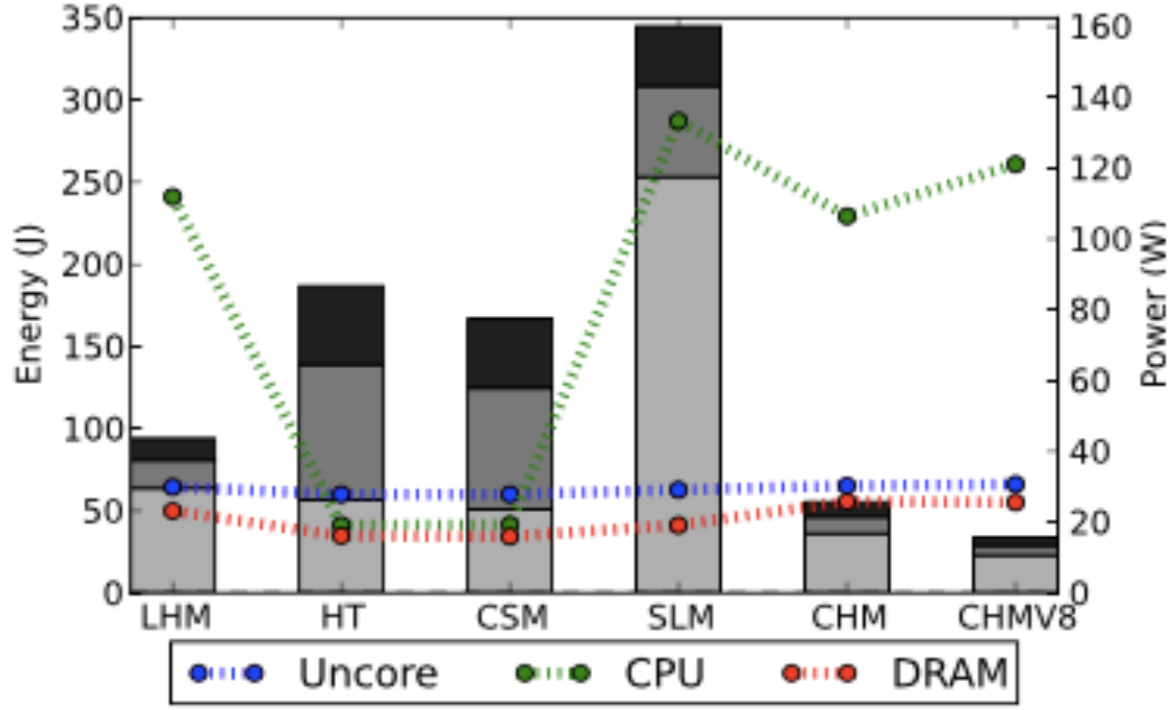
G. Pinto, F. Castor, and Y. D. Liu. Mining questions about software energy consumption. In MSR, pages 22–31, 2014.

We are not the only ones!

“This research agenda argues software reengineering tools and techniques, like static and dynamic program analysis, and systematic code transformations like refactoring, can be used to obtain more energy efficient applications.”

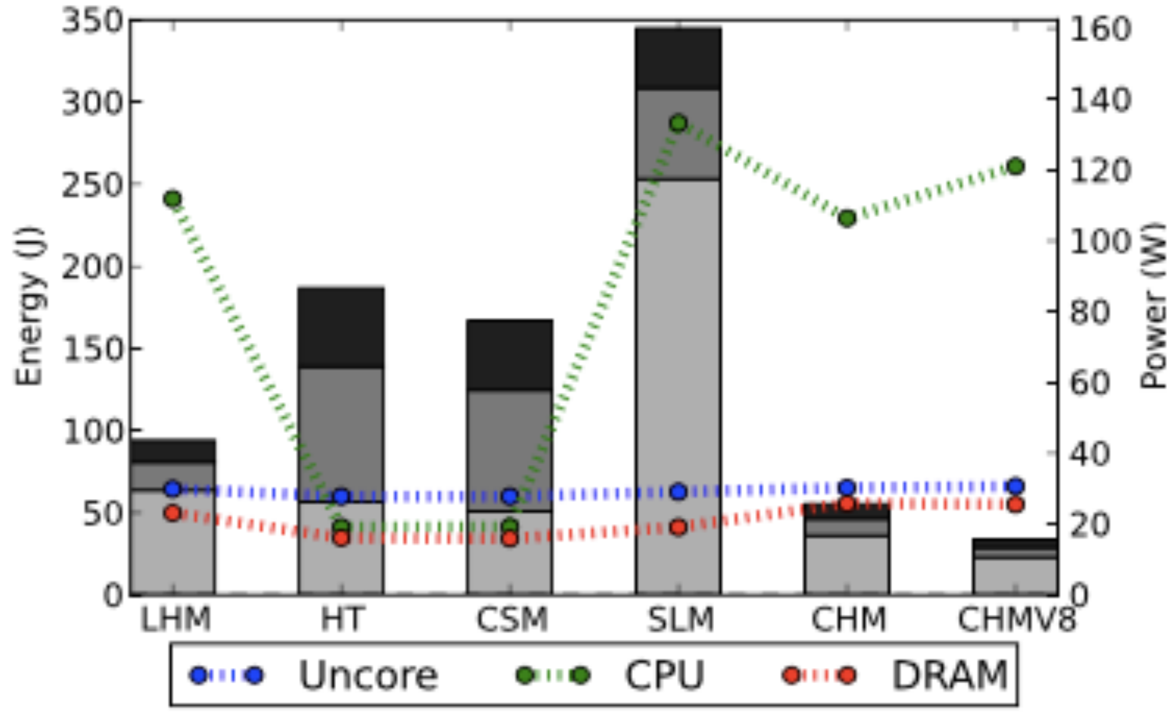
J. Jelschen, M. Gottschalk, M. Josefiok, C. Pitu, and A. Winter. Towards applying reengineering services to energy-efficient applications. In CSMR, pages 353–358, 2012.

Can Refactoring be used to improve the energy efficiency of a software system?



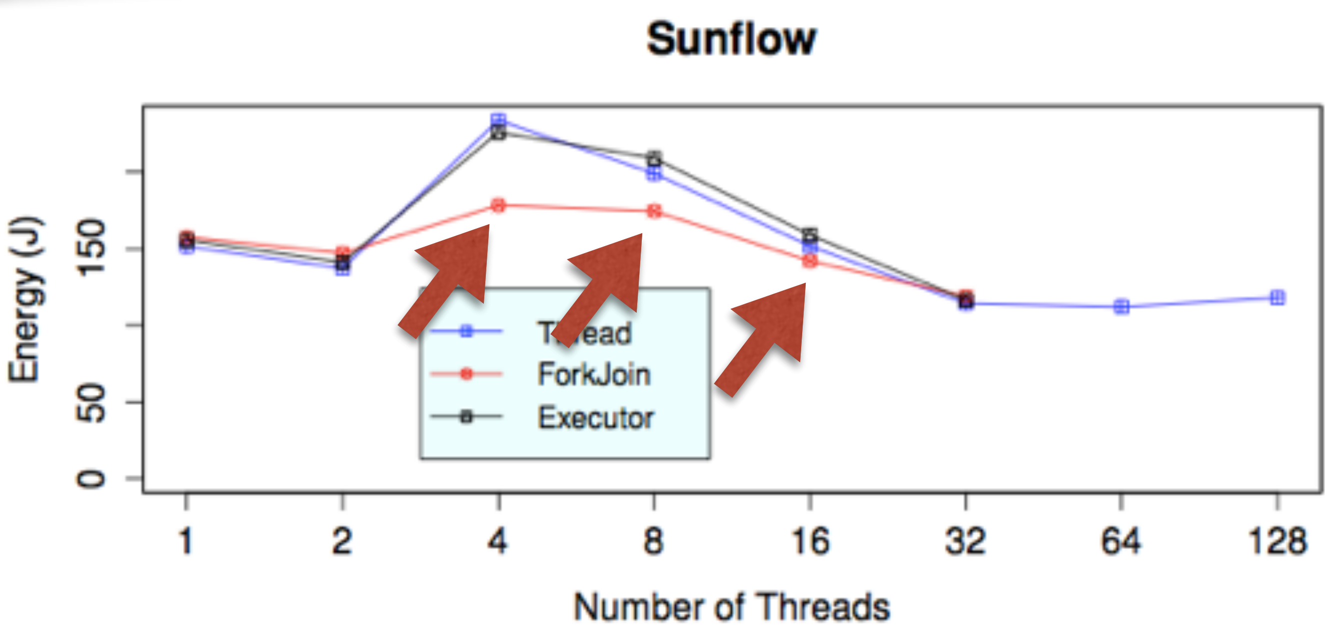
Changing a Map implementation

G. Pinto, K. Liu, F. Castor, and Y. Liu. A comprehensive study on the energy efficiency of java thread-safe collections. Journal of Systems and Software, 2015.



Changing a Map implementation

G. Pinto, K. Liu, F. Castor, and Y. Liu. A comprehensive study on the energy efficiency of java thread-safe collections. Journal of Systems and Software, 2015.



Changing a thread management construct

G. Pinto, F. Castor, and Y. D. Liu. Understanding energy behaviors of thread management constructs. In OOPSLA, pages 345–360, 2014.

Can software energy consumption research be instantiated in refactorings?

OOPSLA
CONFERENCE

International Conference on
Software
Maintenance

ISSTA

ECOOP

CSMR

OOPSLA
CONFERENCE

International Conference on
Software
Maintenance

ISSTA

ECOOP

C S M R

“Power” or “Energy”

OOPSLA CONFERENCE

International Conference on Software Maintenance

ISSTA

ECOOP

CSMR

“Power” or “Energy”

Making Web Applications More Energy Efficient for OLED Smartphones

Ding Li, Angelica Huyen Tran, William G. J. Halfond
Department of Computer Science
University of Southern California
Los Angeles, California, USA
{dingli, tranac, halfond}@usc.edu

ABSTRACT

A smartphone's display is one of its most energy consuming components. Modern smartphones use OLED displays that consume more energy when displaying light colors as opposed to dark colors. This is problematic as many popular mobile web applications use large light colored backgrounds. To address this problem we developed an approach for automatically rewriting web applications so that they generate more energy efficient web pages. Our approach is based on program analysis of the structure of the web application implementation. In the evaluation of our approach we show that it can achieve a 40% reduction in display power consumption. A user study indicates that the transformed web pages are acceptable to users with over 60% choosing to use the transformed pages for normal usage.

Categories and Subject Descriptors

OLED screens [36] are increasingly popular in different smartphones, such as the Samsung Galaxy, Sony Xperia, and LG Optimus series. These screens are more energy efficient than previous generation displays, but also have very different energy consumption patterns. In particular, darker colors, such as black, require less energy to display than lighter colors, such as white. Unfortunately, many popular and widely used web applications use light-colored backgrounds. This means that, for many web application, there is a significant opportunity to improve the battery life of smartphones by improving the color usage of a web application's pages.

Researchers and engineers have long recognized the need to reduce a smartphone's display energy. A well-known and widely used smartphone technique is to dim the display to conserve energy [15]. For example, when the smartphone is idle. This technique is useful, but there is room for additional improvement by exploiting the OLED screen's unique

Green Streams for Data-Intensive Software

Thomas W. Bartenstein and Yu David Liu
SUNY Binghamton
Binghamton, NY13902, USA
{tbarten1, davidL}@binghamton.edu

***Abstract*—This paper introduces GREEN STREAMS, a novel solution to address a critical but often overlooked property of data-intensive software: energy efficiency. GREEN STREAMS is built around two key insights into data-intensive software. First, energy consumption of data-intensive software is strongly correlated to data volume and data processing, both of which are naturally abstracted in the stream programming paradigm; Second, energy efficiency can be improved if the data processing components of a stream program coordinate in a “balanced” way, much like an assembly line that runs most efficiently when participating workers coordinate their pace. GREEN STREAMS adopts a standard stream programming model, and applies Dynamic Voltage and Frequency Scaling (DVFS) to coordinate the pace of data processing among components, ultimately achieving energy efficiency without degrading performance in a parallel processing environment. At the core of GREEN STREAMS is a novel constraint-based inference to abstract the intrinsic relationships of data flow rates inside a stream program, which uses linear programming to minimize the frequencies – hence the energy consumption – for processing components while still maintaining the maximum output data flow rate. The core**

have been proposed to address energy efficiency, through design patterns [12], [13], programming language designs [14], [15], [16] and compiler and runtime optimizations [17], [18], [19], but none has focused on data-intensive software. This is unfortunate because the root cause of energy consumption for data-intensive software is often a combination of high-volume data processing and complex data flows, distinctive traits not sufficiently addressed by solutions built around control-flow-centric models.

In this paper, we propose GREEN STREAMS, a novel energy-efficient solution that addresses data-intensive software. At its essence, GREEN STREAMS is an energy-efficient “twist” to standard stream programming models [4], [20], [21]. Stream programming is a general-purpose paradigm where software is composed as a *stream graph*, where nodes of the graph are data processing components called *filters*, and edges of the graphs are data flows called *streams*. Compared with control-flow-centric models (*e.g.* Java and C), the streaming model exposes

OOPSLA
CONFERENCE

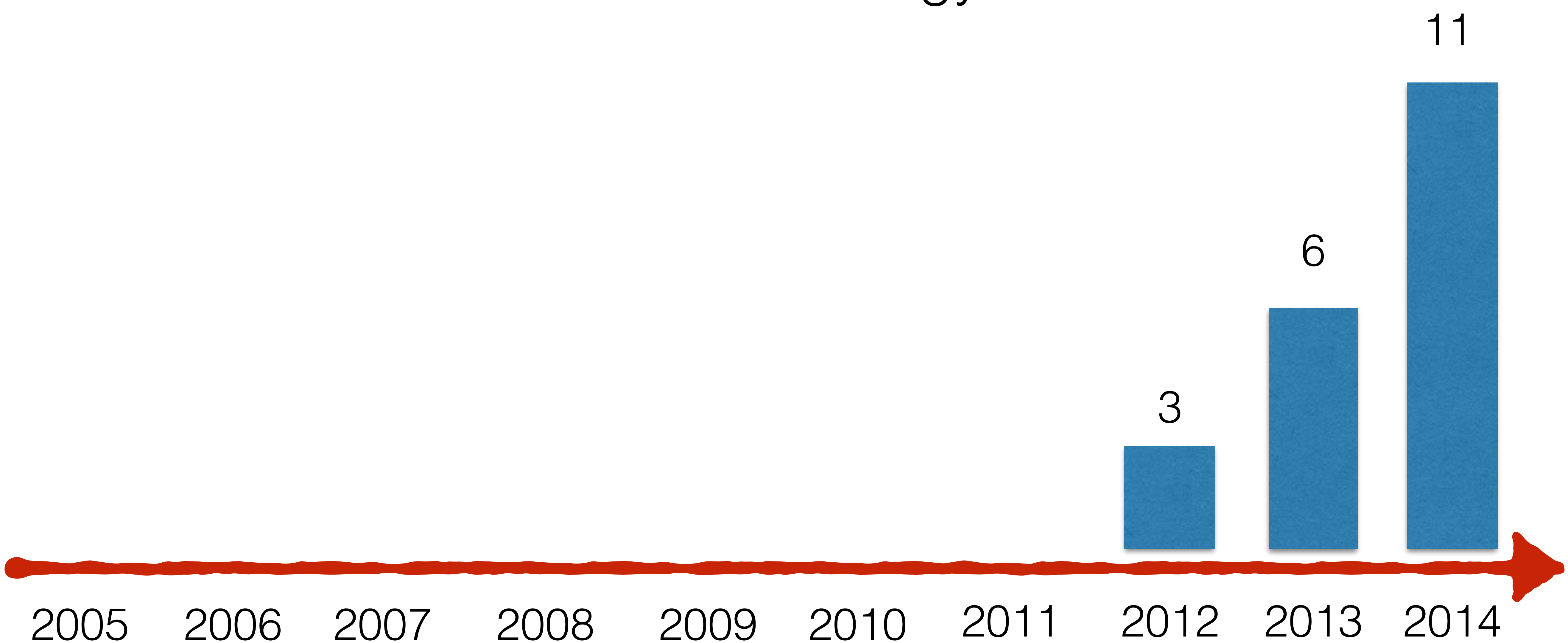
International Conference on
Software
Maintenance

ISSTA

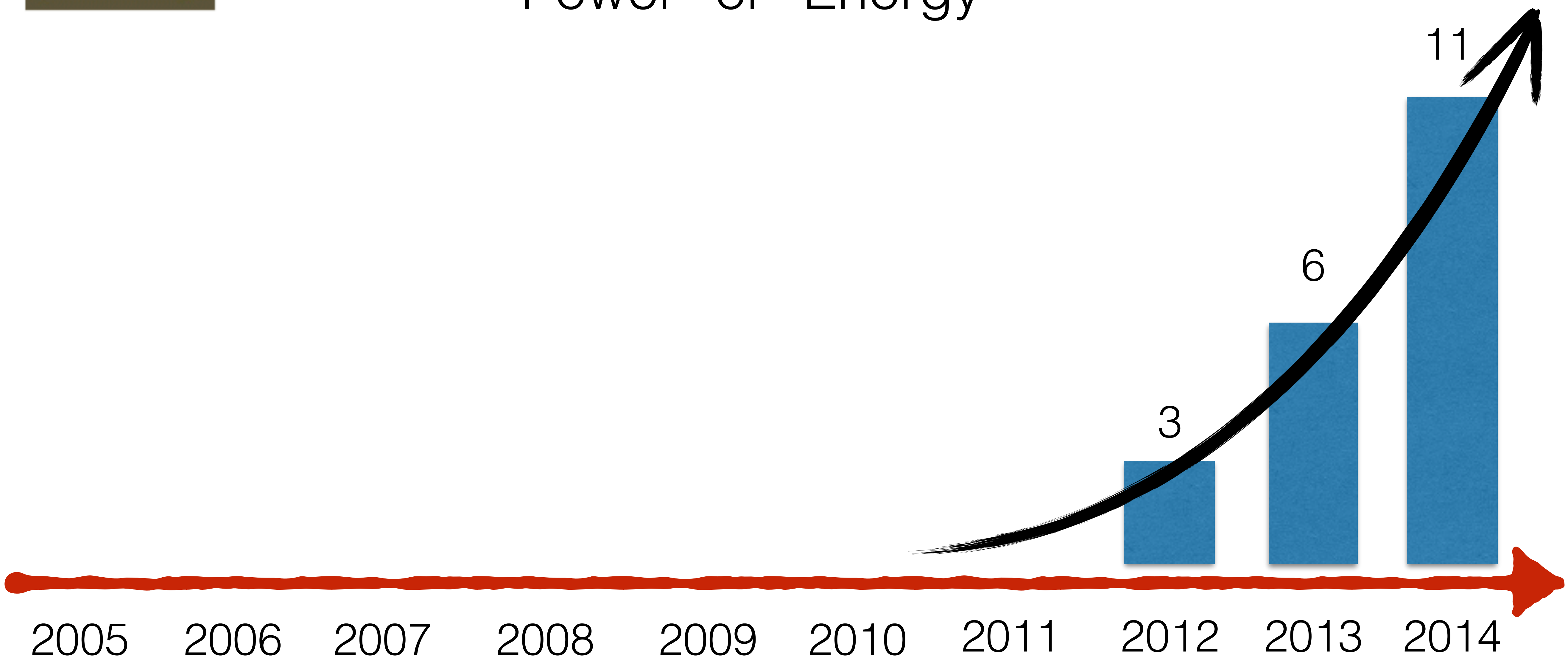
ECOOP

CSMR

“Power” or “Energy”



“Power” or “Energy”



20 Selected Papers

19 Selected Papers

No case study

17 Selected Papers

Do not change the source code

14 Selected Papers

Little to do with refactoring

Research Question

- What are the **opportunities**, and their inherent **challenges**, to derive new **refactorings** focusing on improving the **energy efficiency** of a software system?

Mobile Apps

User Interfaces

- An average of **40% of energy saving** when using darker instead of lighter colors

Challenges

- Color can be dynamically generated
- Analyze different scattered files

D. Li, A. H. Tran, and W. G. J. Halfond. Making web applications more energy efficient for OLED smartphones. In ICSE, pages 527–538, 2014.

Mobile Apps

CPU Offloading:

- Can reduce the overall energy consumption of a mobile application by **up to 50%**

Challenges:

- Decide when to refactor;
- Setup the cloud environment;

Y. Kwon and E. Tilevich. Reducing the energy consumption of mobile applications behind the scenes. In ICSM, pages 170–179, 2013.

Concurrent/Parallel Programming

Excessive Copy Chain

- **Energy saving of 15.38%** when the data is shared instead of copied

```java
import static Arrays.*;
class Task extends RecursiveAction{
  public Task (User[] u) {}
  protected void compute() {
    if (u.length < N) { local(u); }
    else {
      int split = u.length / 2;

      User[] u1 = copyOfRange(u, 0,
          split);
      User[] u2 = copyOfRange(u,
          split, u.length);

      invokeAll(new Task(u1),
                new Task(u2));
    }
  }
}
```

Challenges

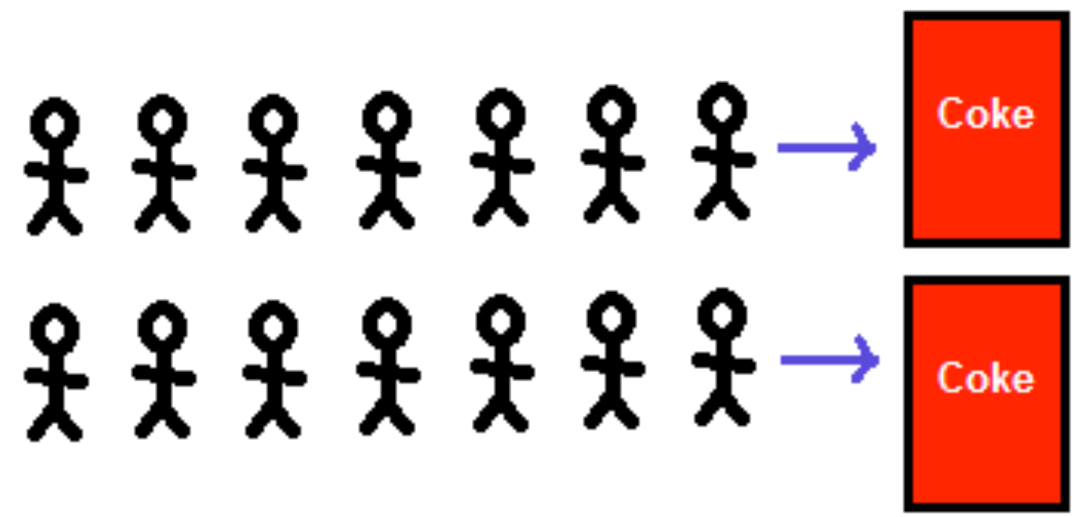
- Identify the copy pattern;

G. Pinto, F. Castor, and Y. D. Liu. Understanding energy behaviors of thread management constructs. In OOPSLA, pages 345–360, 2014.

Concurrent/Parallel Programming

Embrace Parallelism

- Parallel solutions can **save up to 80%** of energy consumption



Parallel: 2 queues, 2 vending machines

Challenges

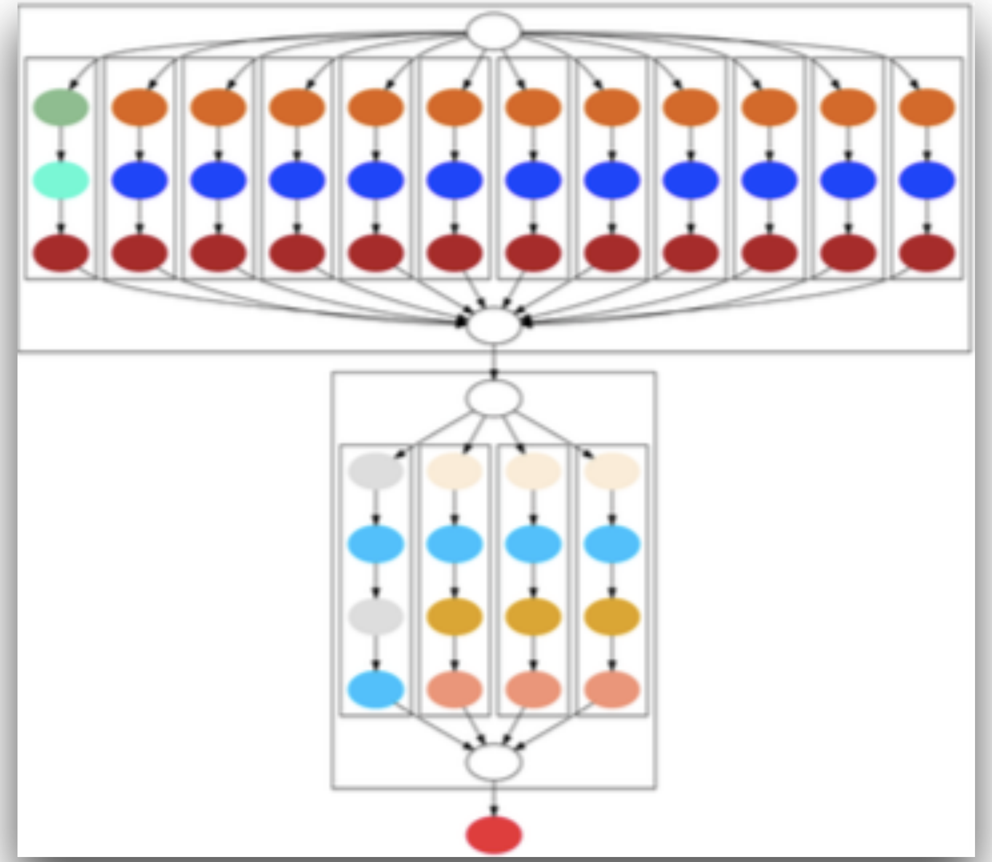
- Not all kinds of problems can be fully parallelizable;
- Energy efficiency can degrade as the user embraces multi-core CPUs;

M. Kambadur and M. A. Kim. An experimental survey of energy management across the stack. In OOPSLA, pages 329–344, 2014.

DVFS Techniques

Stream Programming:

- An average CPU **energy saving of 28%**



Challenges:

- There is no prior support for refactoring for stream programming;

T. W. Bartenstein and Y. D. Liu. Green streams for data-intensive software. In ICSE, pages 532–541, 2013.

DVFS Techniques

Energy Types:

- An energy saving of **30% up to 50%** on CPU

```
phases { graphics <cpu main; main <cpu math; }
modes {hifi <: full; lofi <: hifi; }
class Main {
  main () {
    Recognizer rz = new Recognizer();
    View  v = adapt (new View());
    while(true) {
      Gesture g = processInput();
      int result = rz.recognize(g);
      v.paintOverlay(adapt[graphics] g, result);
    }
  }
  Gesture processInput()
  { ...
    return new Gesture@phase(math)();
  }
}
```

Challenges:

- When to refactor;
- Take care of new language constructs;

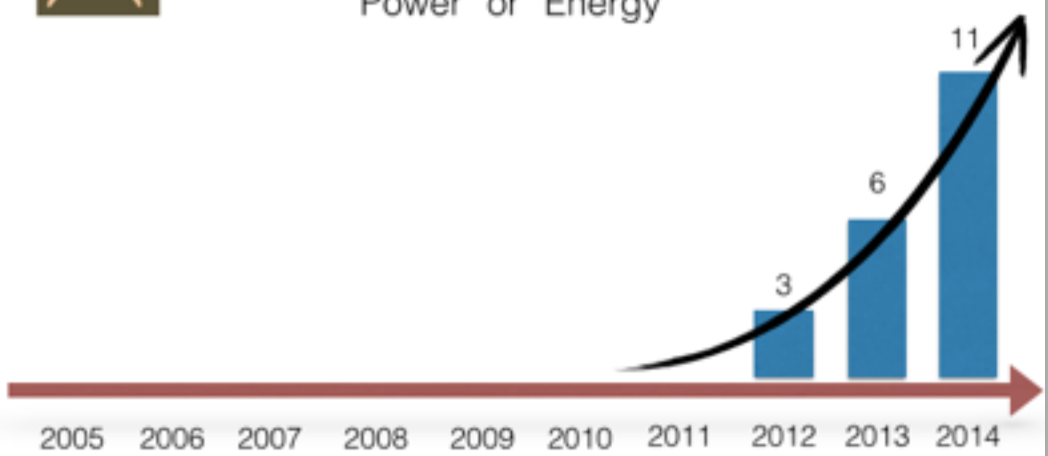
M. Cohen, H. S. Zhu, E. E. Senem, and Y. D. Liu. Energy types. In OOPSLA, pages 831–850, 2012.

What is next?

- Implement some refactorings
- Integrate the refactoring in existing IDEs
- Evaluate the effectiveness of the refactoring
    - Controlled experiment with practitioners

Refactoring for Energy Efficiency: A Reflection on the State of the Art

"Power" or "Energy"



9

Mobile Apps

User Interfaces

- An average of 40% of energy saving when using darker instead of lighter colors

Challenges

- Color can be dynamic generated
- Analyze different scattered files

D. Li, A. H. Tran, and W. G. J. Halfond. Making web applications more energy efficient for OLED smartphones. In ICSE, pages 527–538, 2014.

15

Concurrent/Parallel Programming

Excessive Copy Chain

- Energy saving of 15.38% when the data is shared instead of copied

```
import static Arrays.*;
class Task extends RecursiveAction{
  public Task (User[] u) {}
  protected void compute() {
    if (u.length < N) { local(u); }
    else {
      int split = u.length / 2;

      User[] u1 = copyOfRange(u, 0,
         split);
      User[] u2 = copyOfRange(u,
         split, u.length);

      invokeAll(new Task(u1),
                new Task(u2));
    }
  }
}
```

Challenges

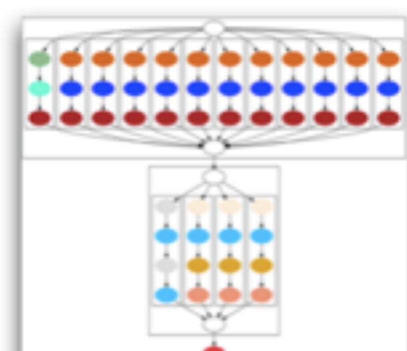
- Identify the copy pattern;

G. Pinto, F. Castor, and Y. D. Liu. Understanding energy behaviors of thread management constructs. In OOPSLA, pages 345–360, 2014.

DVFS Techniques

Stream Programming:

- An average CPU energy saving of 28%.



Challenges:

- There is no prior support for refactoring for stream programming

T. W. Bartenstein and Y. D. Liu. Green streams for data-intensive software. In ICSE, pages 532–541, 2013.

19