Understanding and Improving the Energy Efficiency of Hybrid Mobile Applications

William G.J. Halfond

University of Southern California

Together with Ding Li, Jiaping Gui, Angelica Huyen Tran, Shuai Hao, and Ramesh Govindan

Work supported by NSF grant 1321141

Energy Measurement Goals

1. Accurate
2. Fast
3. Fine-grained
4. Lightweight

Measurement & Estimation Tools

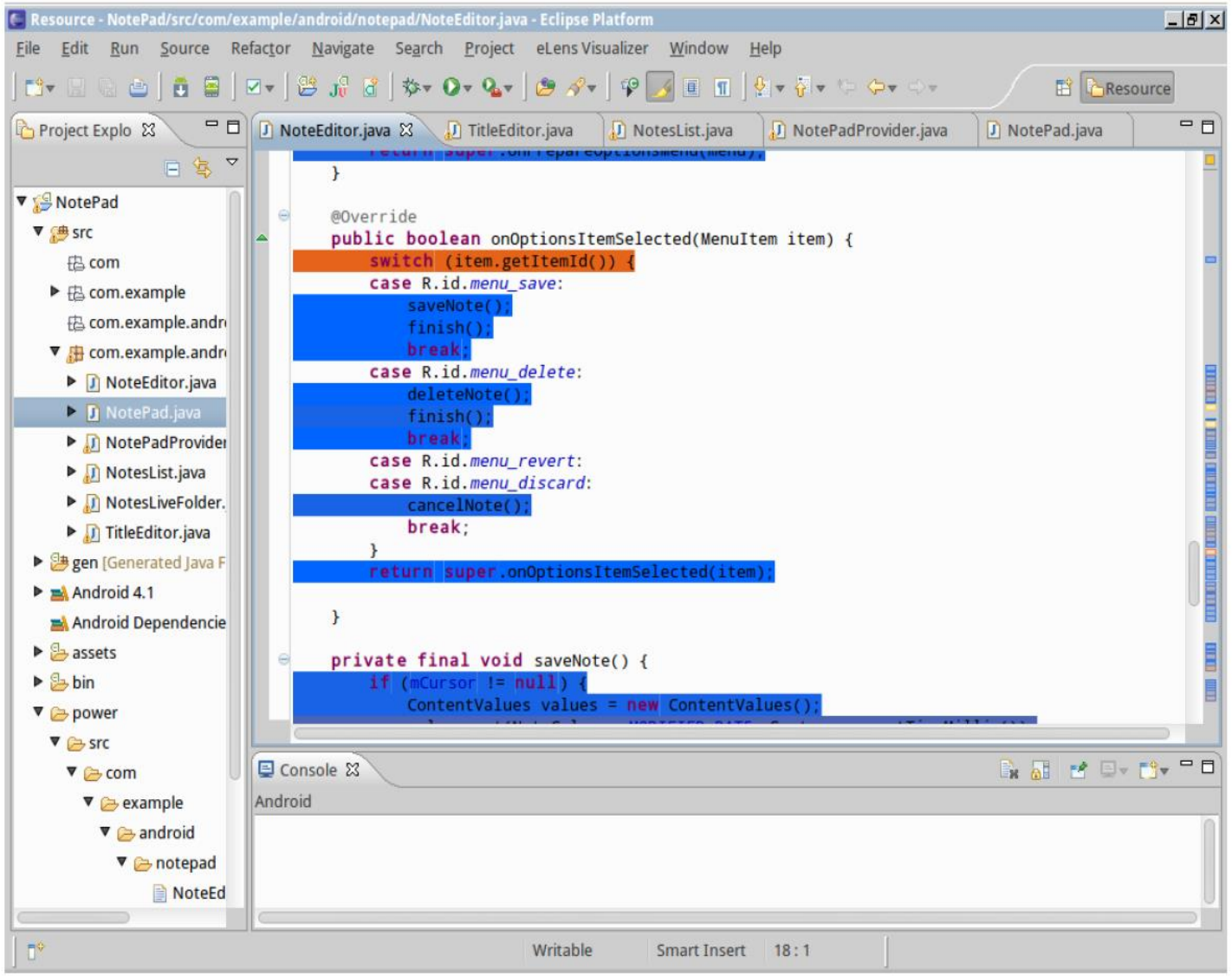
eLens – estimate energy consumption

- Combines program analysis and per-instruction cost modeling
- Accurate to within 8.8% of ground truth
- But creating models is hard!

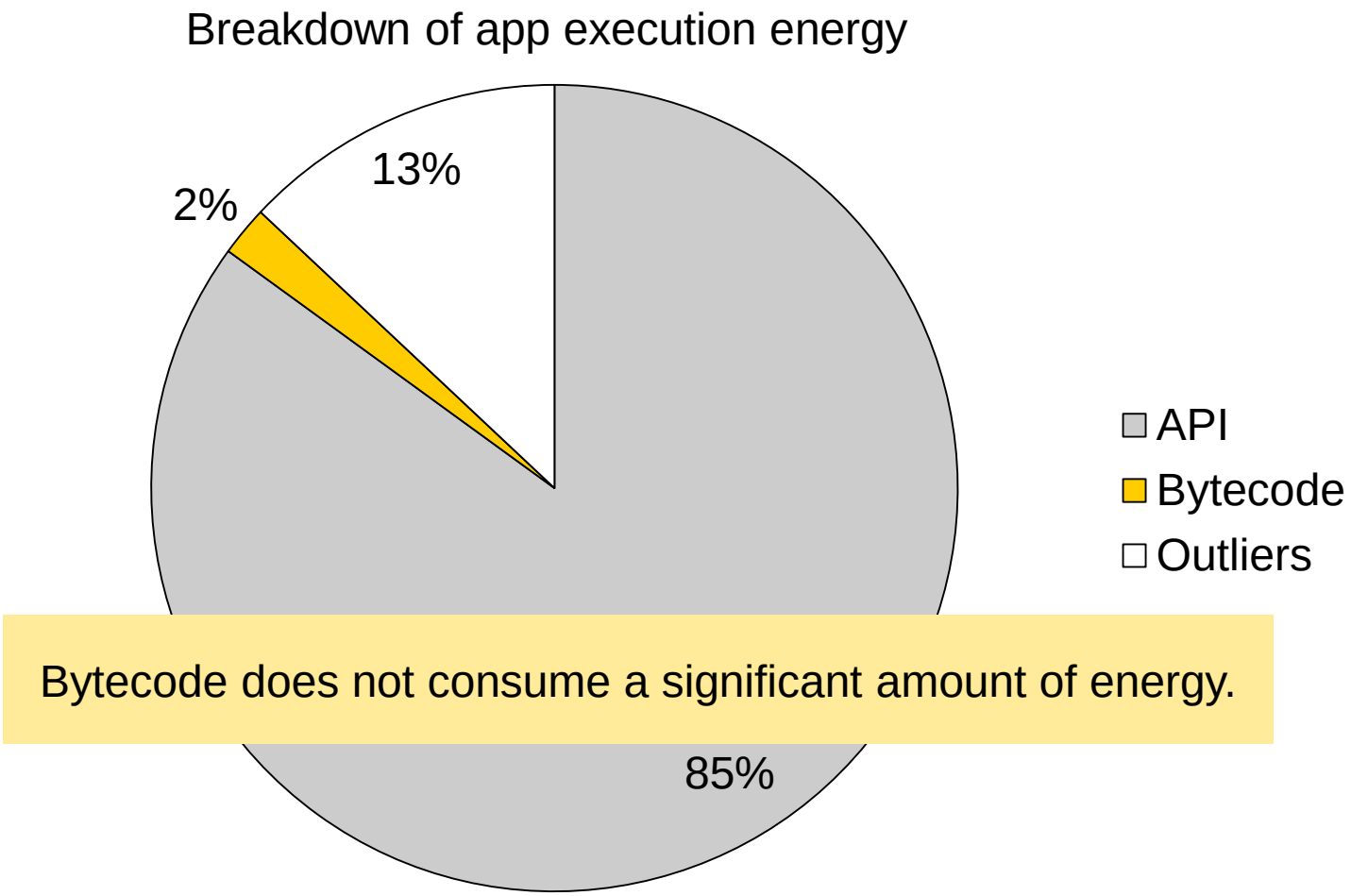
vLens – visualize energy consumption

- Combines program analysis, statistical analysis, and coarse-grained measurements
- Accurate to within 10% of ground truth

Insight into Code-level Energy Usage

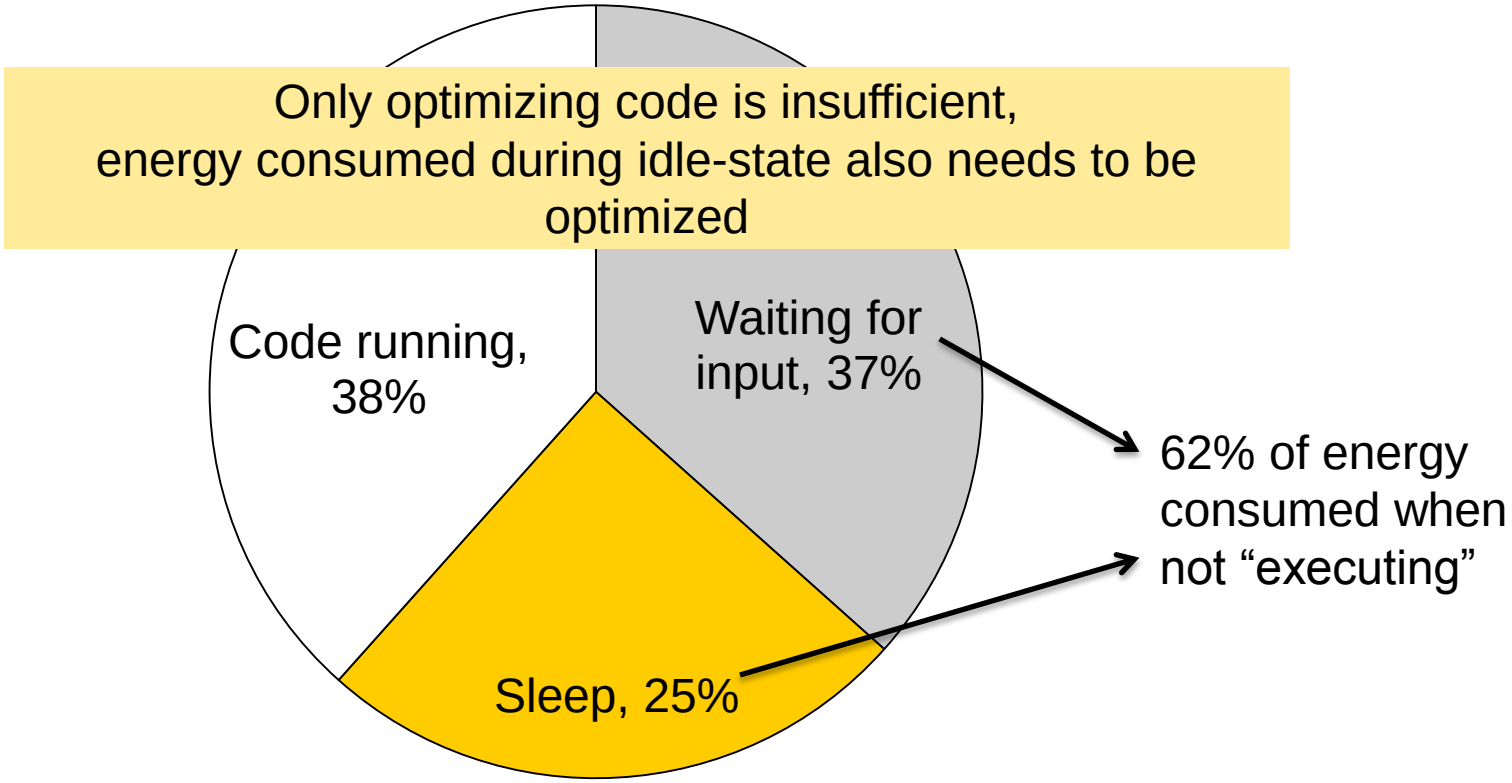


Energy Consumption of System APIs vs. Bytecode vs. Outliers



Bytecode does not consume a significant amount of energy.

Energy Consumed by the Idle State of An Application

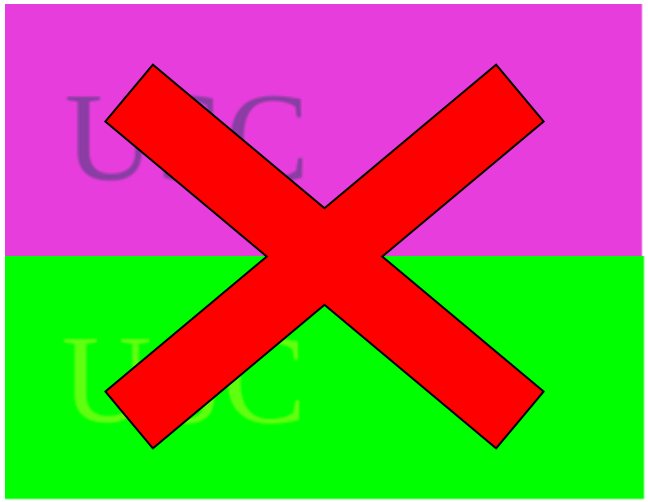


Smartphone Display: OLED

- Popular technology for smartphone displays
- More energy efficient than prior technologies
- Different energy consumption patterns

Display Oriented Techniques

- Dim the display
  - Good start, but more can be done
- Invert colors:



Goal

Automatically transform the implementation of a web application so that the web pages it generates consume less energy, but maintain aesthetics, when displayed on an OLED smartphone.

Challenges

1. Iden plication
   – Nec hat can be g
2. Dete in the web
   – Imp re
3. Tran scheme
   – Mai esthetics)

```
1  public void print_html()
2  {
3   print("<body bgcolor=\"white\" style=\"color:black;\">");
4   println("<table><tr>");
5   int a=1;
6   if(a==0){
7     println("<td>hi</td>");
8   }
9   else{
10    println("<td style=\"background-color:red; color:yellow
          ;\">ha</td>")
11  }
12  for(int i=0;i<2;i++){
13    println("<td style=\"background-color:green; color:blue
          ;\">usc</td>");
14  }
15  println("</tr></table>");
16  println("</body>");
17 }
```

Approach Overview

1. Compute the set of generated HTML pages
2. Determine visual relationships in pages
   - Example: adjacent and contained

Phase 1

3. Identify colors that have visual relationships
4. Solve for a new color scheme
   - Is more energy efficient
   - Maintains similar color differences

Phase 2

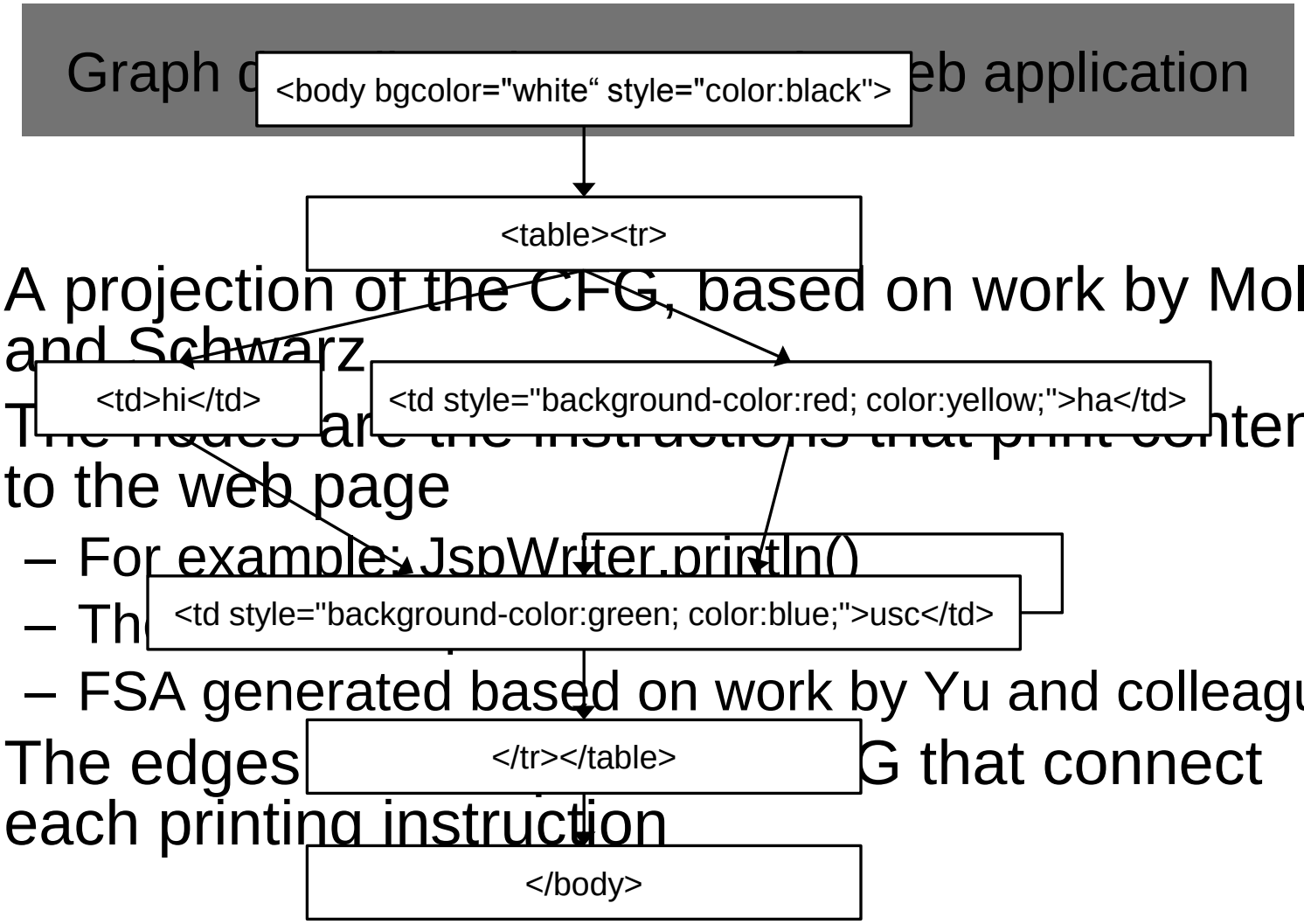
5. Rewrite application to use new color scheme

Phase 3

Phase 1: HTML Output Analysis

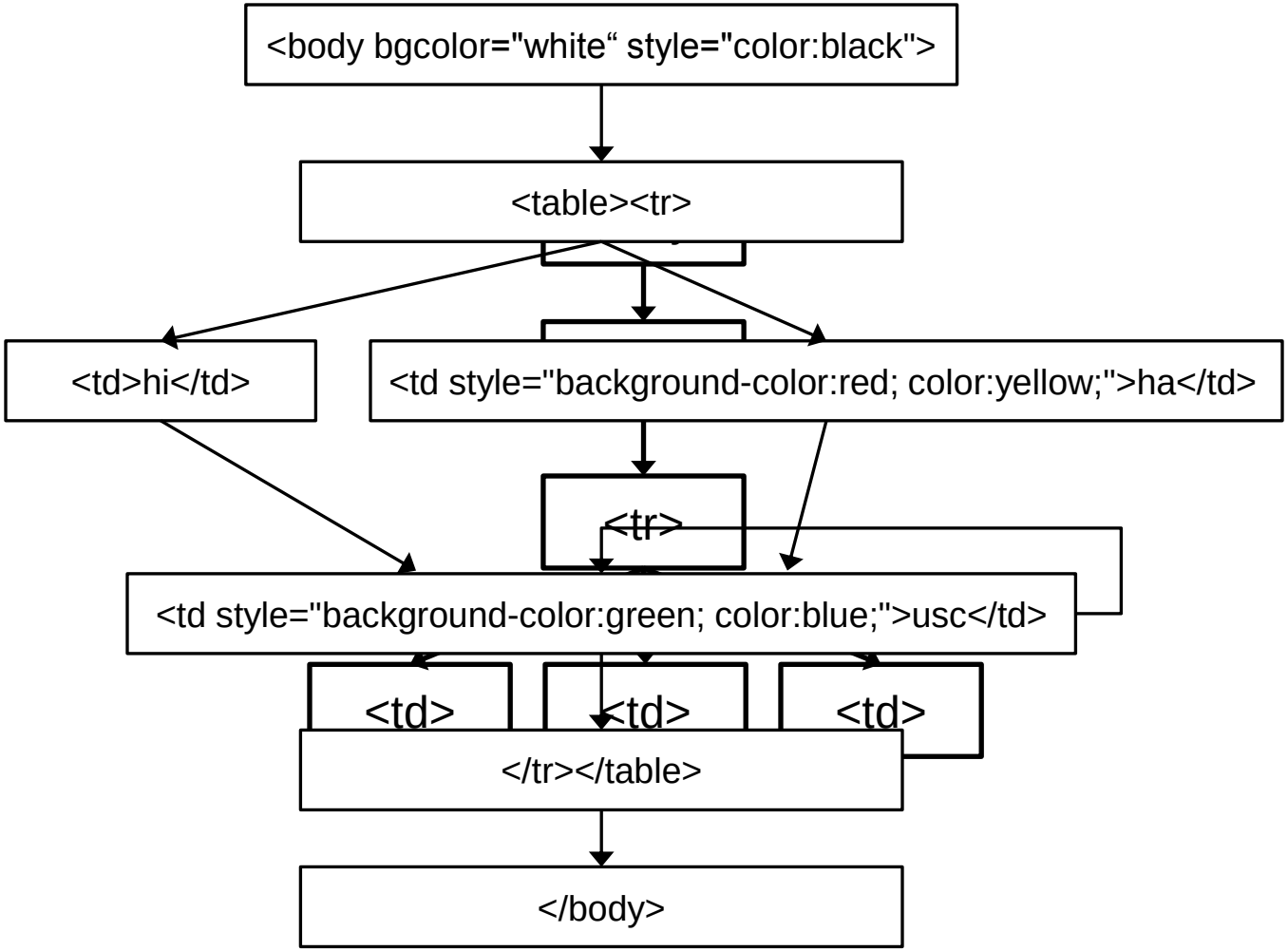
A. Compute the set of HTML pages that could be generated by the application at runtime
B. Determine visual relationships among HTML elements in the pages
   - Example: adjacent and contained

Phase 1A: HTML Output Graph



- A projection of the CFG, based on work by Moller and Schwarz
- The nodes are the instructions that print content to the web page
  - For example: JspWriter.println()
  - Th
  - FSA generated based on work by Yu and colleagues
- The edges G that connect each printing instruction

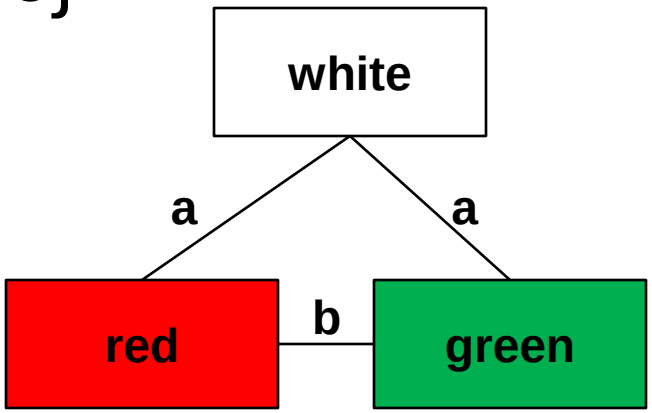
Phase 1B: Visual Relationship Graph



2: Color Transformation

Color Conflict Graph (CCG)

- Shows visual relationships of colors in a page
- BCCG: weights are in {a,b,c}
  - a>b>c>0
  - a: parent-child
  - b: siblings
  - c: everything else

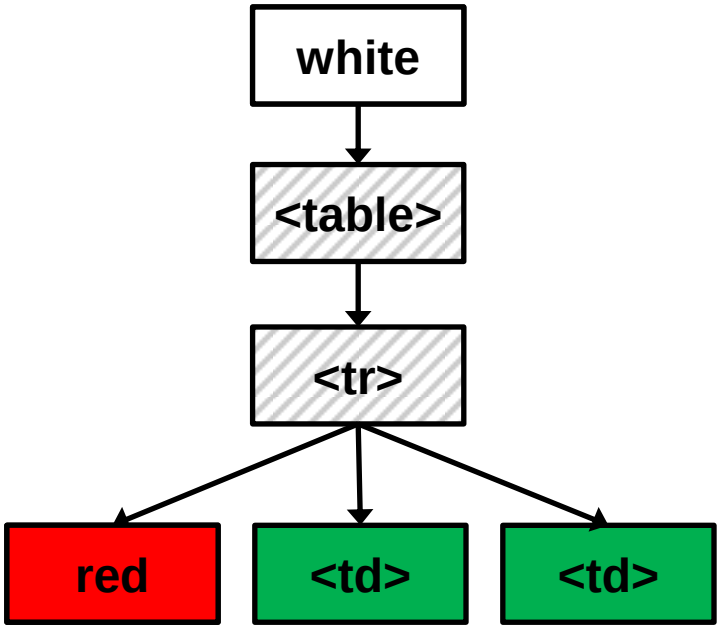


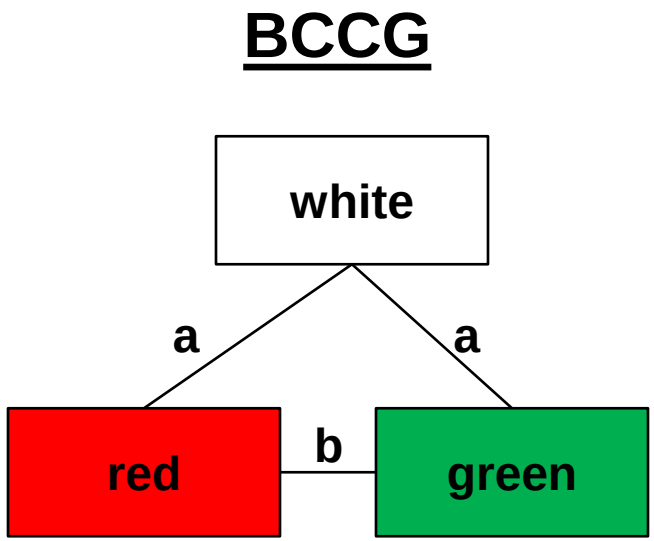
2: Color Transformation

Building the Color Conflict Graph

1. Basic unit is color definition (CD)
   - CSS based
   - HTML based
2. Perform reachability analysis over visual relationship graph
3. “Reaching CDs” define edges in CCG

2: Color Transformation





BCCG: weights are in {a,b,c}, a>b>c>0

a: parent-child

b: siblings

c: everything else

2: Color Transformation

Generate the color transformation scheme (CTS)

1. Let $S = \langle C_0, C_1, C_2, \ldots, C_k \rangle$ nodes of the CCG
2. Let S' be the new coloring, where C_0 =black
3. Compute S' that results in similar color differences as in S, i.e. minimize:

$$\sum_{i=0}^{k} \sum_{j=0}^{k} w_{ij} \left| Dist(C_i, C_j) - Dist(C_i', C_j') \right|$$

4. Optimization problem is NP-Hard, use simulated annealing to approximate optimal solution

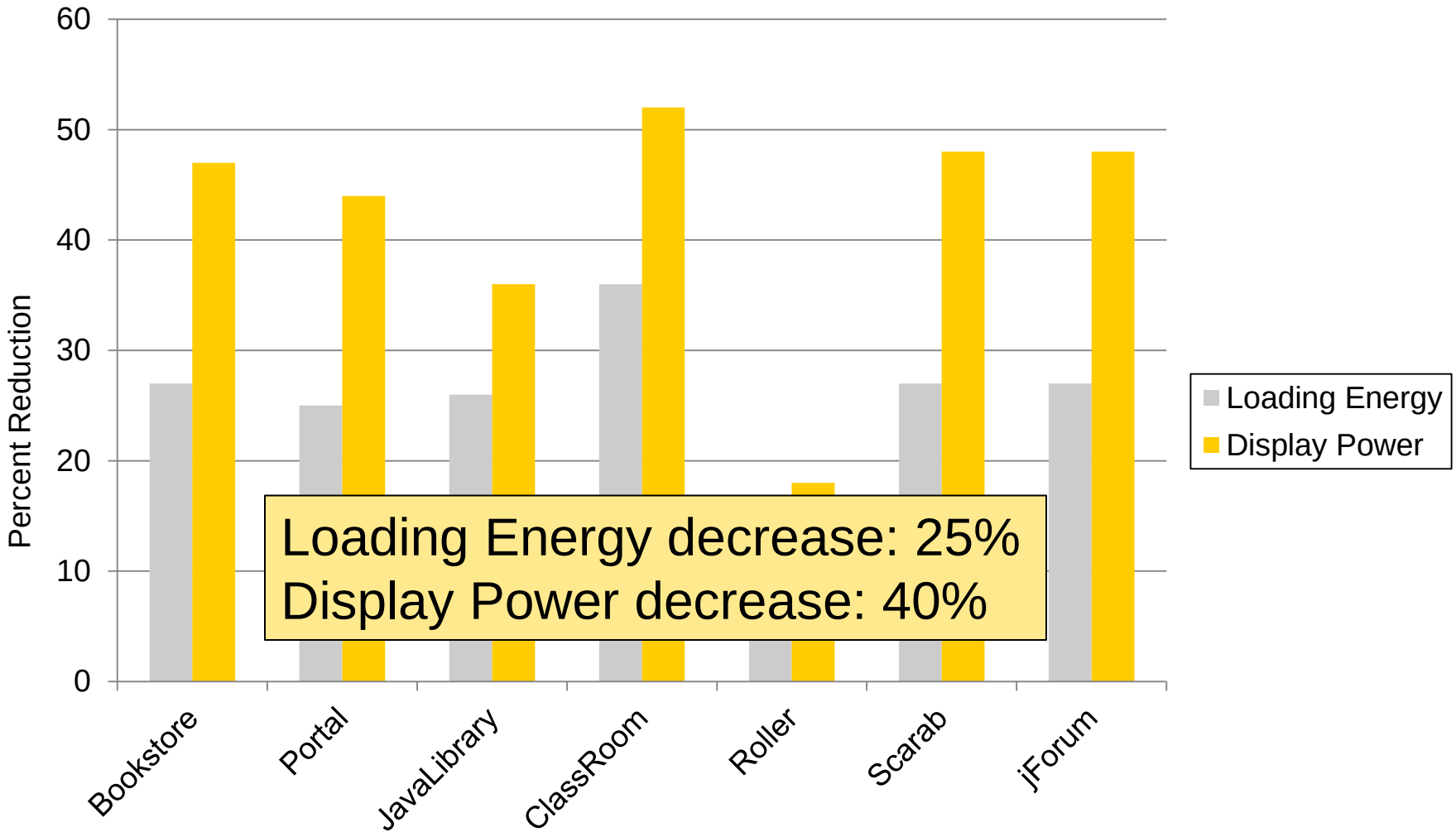
Phase 3: Output Modification

1. Dynamically generated HTML pages
   - Insert instrumentation to replace HTML printing instructions
   - Replace original colors with new colors
2. Template based frameworks
   - Use CSS parser to identify entries to be replaced
   - Replace entries by rewriting CSS and HTML

Evaluation

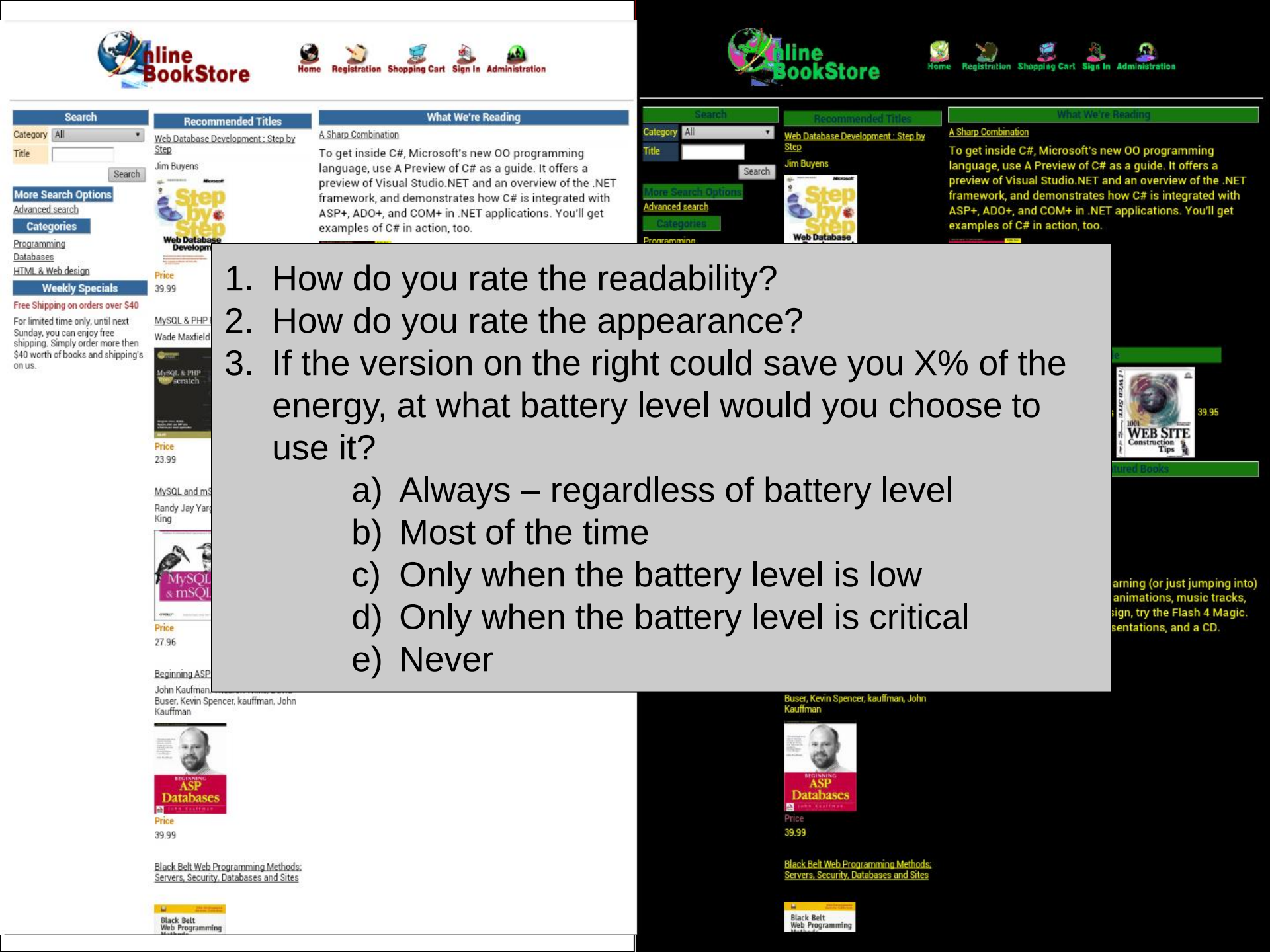
- **RQ 2:** How much energy is saved by the transformed web pages?
- **RQ 3:** To what degree do users accept the appearance of the transformed web pages?

RQ2: Energy Savings



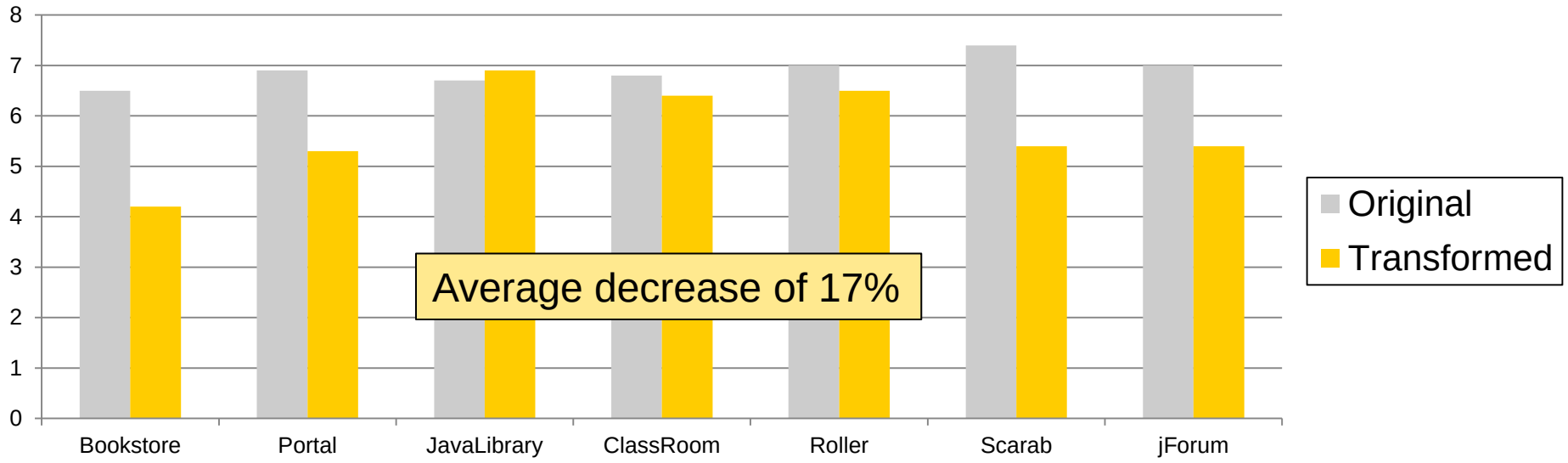
RQ3: User Acceptance

Users asked to rate before/after color transformation produced by our approach

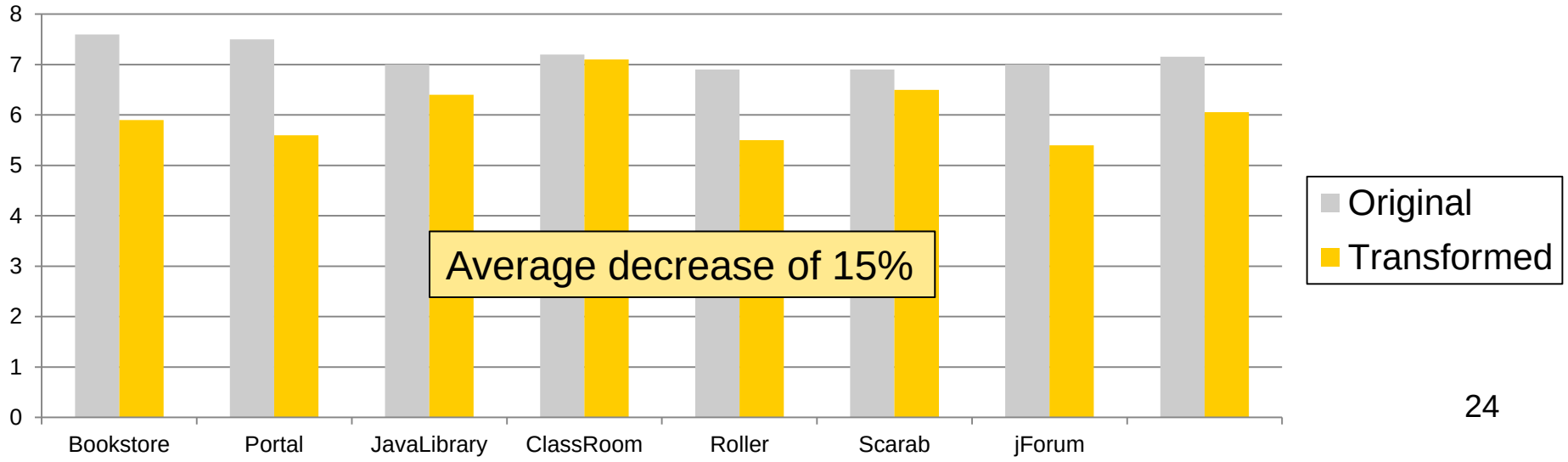


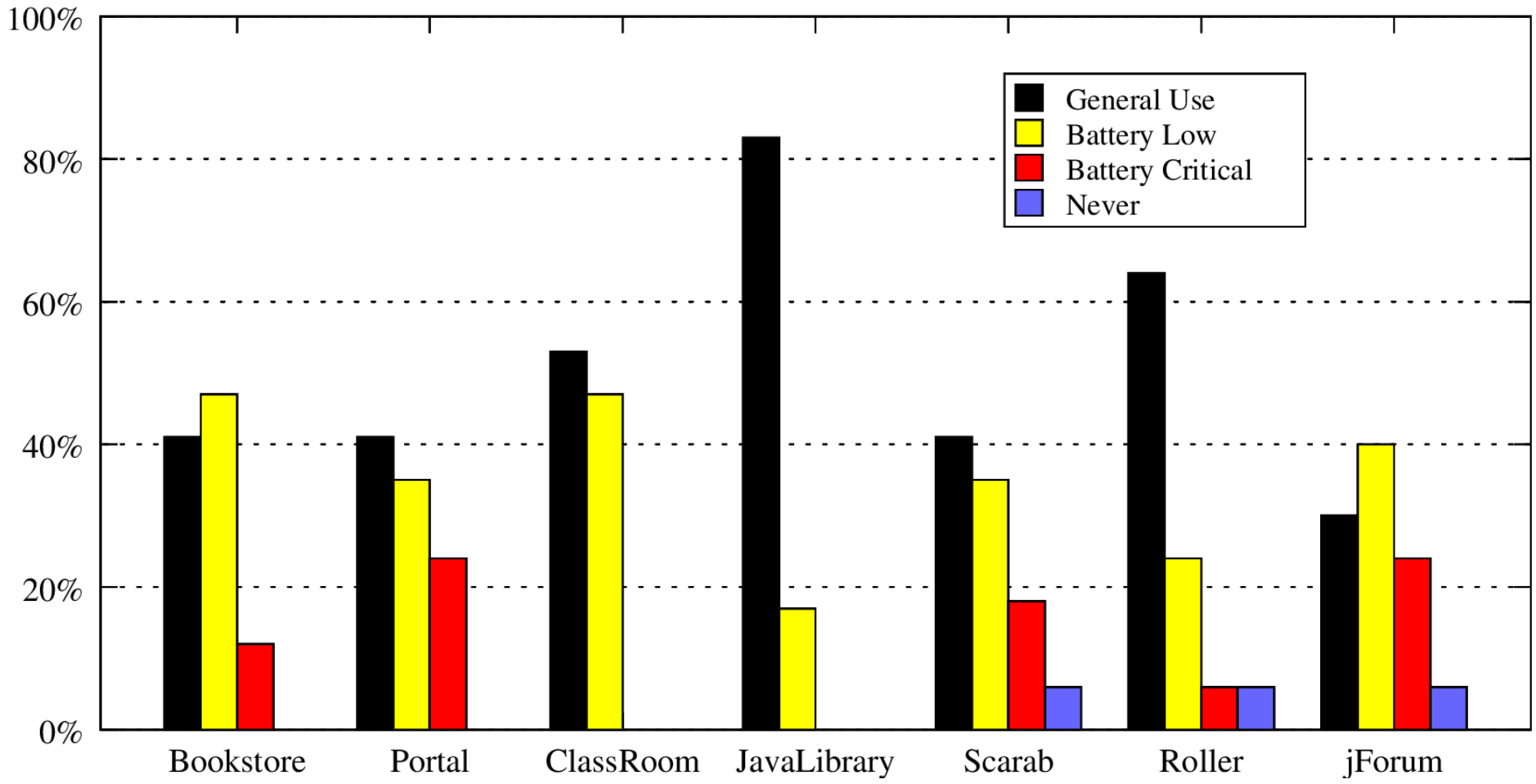
1. How do you rate the readability?
2. How do you rate the appearance?
3. If the version on the right could save you X% of the energy, at what battery level would you choose to use it?
    a) Always – regardless of battery level
    b) Most of the time
    c) Only when the battery level is low
    d) Only when the battery level is critical
    e) Never

Attractiveness



Readability





- 60% choose transformed app for general usage
- 97% choose transformed app for battery critical

Summary

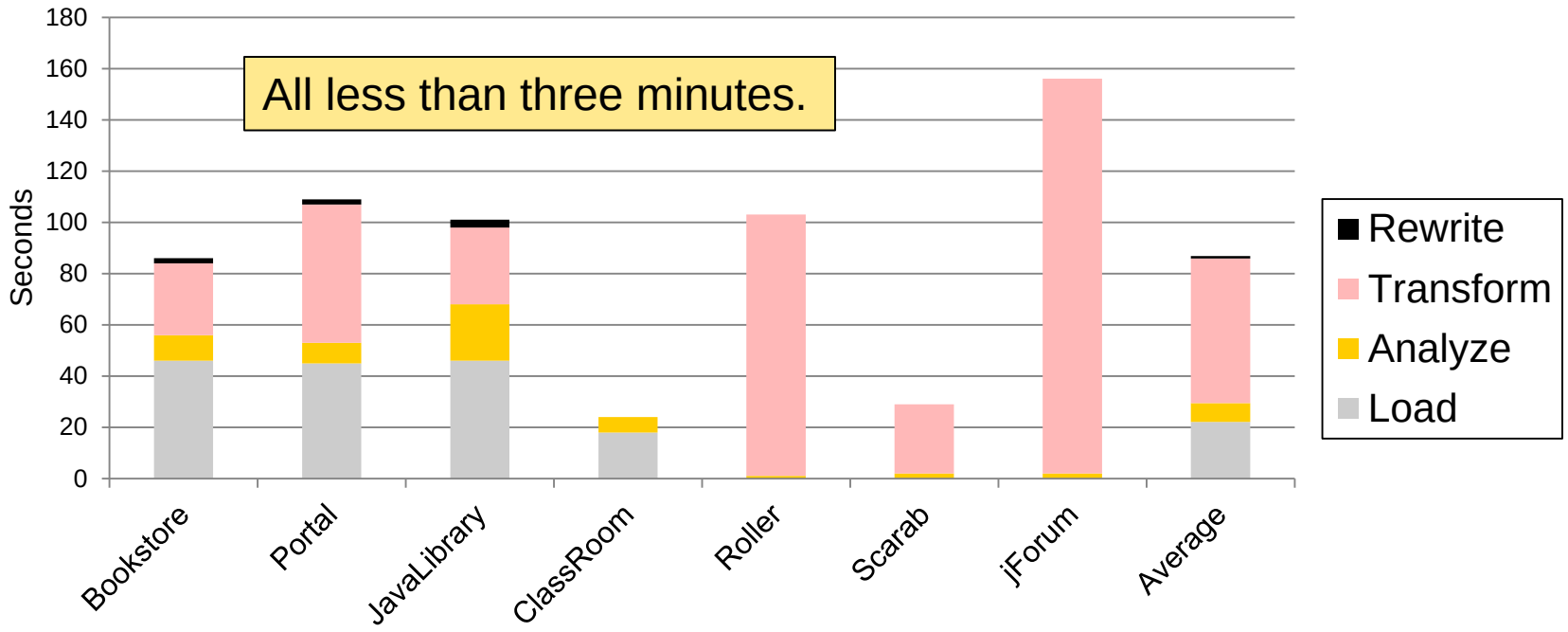
- **Visualize energy consumption**
  - eLens: program analysis + cost models
  - vLens: program analysis + regression analysis
- **Understand energy consumption**
  - Idle state energy consumption is significant
  - Display is a major part of this
- **Change energy consumption**
  - Automatically rewrite web pages so they use more energy efficient color scheme

Subject Applications

Name	Framework	SLOC
Bookstore	JSP	24,305
Portal	JSP	21,393
JavaLibrary	JSP & Servlet	73,468
ClassRoom	JSP	5,127
Roller	JSP & Struts	154,065
Scarab	Velocity & Turbine	145,435
jForum	Velocity	31,841

- Four embed color information in HTML, three use CSS
- Three heavily use JavaScript in the user interface
- Three use Model-View-Controller style

RQ1: Time Cost



- Most of the load time was Soot processing
- Load times varies because some apps use templates
- Transform time varies based on complexity of HTML page structure