Objectives

• Tools for Finding Concerns
  ➢ Dora

Review

• What is a problem for developers to perform maintenance tasks?
• What are some tools to address this problem?
  ➢ What approach do the tools take?
• What are the tools’ strengths and limitations?

“Exploring the neighborhood with Dora to expedite software maintenance”
Gibson Hill, Pollock, Vijay-Shankar
ASE 2007

Relevant Neighborhood

DORA

Reviewing Paper

• Problem?
  ➢ Goals
• Approach?
  ➢ Benefits

Program Exploration for Maintenance

eBay auction sniping (bidding) program has bug in add auction event trigger

• Exploration Task: Locate code related to ‘add auction’ trigger
• Starting point: DoAction() method, from prior knowledge
  ➢ Handles all user-triggered events

Running Example Scenario

Source: Hill ASE 2007
Source: Hill ASE 2007
Looking for: ‘add auction’ trigger

- DoAction() has 40 callees
- Only 2/40 methods relevant
- Locates locally relevant items
- But too many irrelevant

And what if you wanted to explore more than one edge away?

Looking for: ‘add auction’ trigger in 1902 methods (159 files, 23KLOC)
• Use lexical information from comments & identifiers
• Search with query ‘add*auction’
• 91 query matches in 50 methods
• Only 2/50 methods relevant
+ Locates globally relevant items
- But too many irrelevant

Dora gets it right...
Looking for: ‘add auction’ trigger
✓ Structural: guide exploration from starting point
✓ Lexical: prunes irrelevant edges

Software Maintenance:
Dora to the rescue
• Developers spend more time finding and understanding code than actually fixing bugs [Kersten & Murphy 2005, Ko et al. 2005]
• Critical need for automated tools to help developers explore and understand today’s large & complex software

⇒ Key Contribution: Automated tools can use program structure and identifier names to save the developer time and effort

Dora the Program Explorer

The Dora Approach
Prune irrelevant structural edges from seed
1. Obtain set of methods one call edge away from seed
2. Determine each method’s relevance to query
   Calculate lexical-based relevance score
3. Prune low-scored methods from neighborhood, using threshold
4. Recursively explore

Dora comes from exploradora, the Spanish word for a female explorer.
Calculating Relevance Score:

**Term Frequency**
- Score based on number of occurrences of query terms in the method
- Intuition: The more query terms in a method, the more likely it is relevant

**Query:** 'add auction'

- Intuition: The more query terms in a method, the more likely it is relevant

**Score based on number of occurrences of query terms in the method**

**Intuition:** The more query terms in a method, the more likely it is relevant

**Query:** 'add auction'

**Source:** Hill ASE 2007

---

**Calculating Relevance Score:**

**Inverse Document Frequency**
- Use inverse document frequency (idf)
- Intuition: Highly weight terms that appear in few documents/methods
- Terms appearing all over program not good discriminators
- Don’t separate relevant from irrelevant methods

**Source:** Hill ASE 2007

---

**Calculating Relevance Score:**

**TF-IDF**
- Score based on method query term frequency (idf)
- Multiplied by natural log of inverse document frequency (idf)

**Query:** 'add auction'

**Intuition:** The more query terms in a method, the more likely it is relevant

**Score based on method query term frequency (idf)**

**Intuition:** The more query terms in a method, the more likely it is relevant

**Score based on method query term frequency (idf)**

**Source:** Hill ASE 2007

---

**Calculating Relevance Score:**

**What about location?**
- Weigh term frequency (tf-idf) based on location:
  - Method name more important than body
  - Method body statements normalized by length

**Source:** Hill ASE 2007

---

**Dora’s Relevance Score**

**Factors**
- \( \sum \text{tf-idf for each query term in the method name} \)
- \( \sum \text{tf-idf for each query term in the method body} \)
- The number of statements in the method
- Binary methods: a library?
- How to determine weights?
  - Applied logistic regression
  - Trained on methods from 9 concerns in previous concern location tool evaluation [Shepherd et al. 2007]

**Source:** Hill ASE 2007

---

**Logistic Regression**

**Train on methods with known relevance to a concept**

**Factors**
- \( \sum \text{tf-idf for each query term in the method name} \)
- \( \sum \text{tf-idf for each query term in the method body} \)
- The number of statements in the method
- Binary methods: a library?
- How to determine weights?
  - Applied logistic regression
  - Trained on methods from 9 concerns in previous concern location tool evaluation [Shepherd et al. 2007]

**Train on methods with known relevance to a concept**

**Factors**
- \( \sum \text{tf-idf for each query term in the method name} \)
- \( \sum \text{tf-idf for each query term in the method body} \)
- The number of statements in the method
- Binary methods: a library?
- How to determine weights?
  - Applied logistic regression
  - Trained on methods from 9 concerns in previous concern location tool evaluation [Shepherd et al. 2007]

**Train on methods with known relevance to a concept**

**Factors**
- \( \sum \text{tf-idf for each query term in the method name} \)
- \( \sum \text{tf-idf for each query term in the method body} \)
- The number of statements in the method
- Binary methods: a library?
- How to determine weights?
  - Applied logistic regression
  - Trained on methods from 9 concerns in previous concern location tool evaluation [Shepherd et al. 2007]

**Train on methods with known relevance to a concept**

**Factors**
- \( \sum \text{tf-idf for each query term in the method name} \)
- \( \sum \text{tf-idf for each query term in the method body} \)
- The number of statements in the method
- Binary methods: a library?
- How to determine weights?
  - Applied logistic regression
  - Trained on methods from 9 concerns in previous concern location tool evaluation [Shepherd et al. 2007]

**Train on methods with known relevance to a concept**

**Factors**
- \( \sum \text{tf-idf for each query term in the method name} \)
- \( \sum \text{tf-idf for each query term in the method body} \)
- The number of statements in the method
- Binary methods: a library?
- How to determine weights?
  - Applied logistic regression
  - Trained on methods from 9 concerns in previous concern location tool evaluation [Shepherd et al. 2007]

**Train on methods with known relevance to a concept**

**Factors**
- \( \sum \text{tf-idf for each query term in the method name} \)
- \( \sum \text{tf-idf for each query term in the method body} \)
- The number of statements in the method
- Binary methods: a library?
- How to determine weights?
  - Applied logistic regression
  - Trained on methods from 9 concerns in previous concern location tool evaluation [Shepherd et al. 2007]

**Train on methods with known relevance to a concept**

**Factors**
- \( \sum \text{tf-idf for each query term in the method name} \)
- \( \sum \text{tf-idf for each query term in the method body} \)
- The number of statements in the method
- Binary methods: a library?
- How to determine weights?
  - Applied logistic regression
  - Trained on methods from 9 concerns in previous concern location tool evaluation [Shepherd et al. 2007]

**Train on methods with known relevance to a concept**

**Factors**
- \( \sum \text{tf-idf for each query term in the method name} \)
- \( \sum \text{tf-idf for each query term in the method body} \)
- The number of statements in the method
- Binary methods: a library?
- How to determine weights?
  - Applied logistic regression
  - Trained on methods from 9 concerns in previous concern location tool evaluation [Shepherd et al. 2007]
Example:

Dora explores ‘add auction’ trigger

Scores from DoAction() seed:

- Identified as relevant with 0.5 threshold
  - DoAdd() (0.93)
  - DoPasteFromClipboard() (0.60)

- With only one false positive
  - DoSave() (0.52)


“Source Code Exploration with Google”

- Problem?
  - Limitations of state of the art
  - Goals
- Approach?
  - Benefit?
- Evaluation?
  - Results?

Experimental Evaluation: Design

- Gold Set: 8 concerns from 4 Java programs, manually mapped by 3 independent developers [Robillard et al. 2007]
- Compare 4 exploration techniques: 1 structural, 3 lexical + structural
  - Structural: Suade [Robillard 2005]
    - Automatically generates exploration suggestions from seed set
    - Elements that have few connections outside the seed set are more relevant
    - Uses caller/callee & field def-use information to make recommendations
  - Lexical + Structural: Dora (sophisticated)
  - Lexical + Structural: boolean AND (naïve)
  - Lexical + Structural: boolean OR (naïve)

Experimental Evaluation: Design

- Gold Set: 8 concerns from 4 Java programs, manually mapped by 3 independent developers [Robillard et al. 2007]
- Compare 4 exploration techniques: 1 structural, 3 lexical + structural
- Measures: Precision (P), Recall (R), & F Measure (F)
  - Precision (P): \( P = \frac{TP}{TP + FP} \) (Are the results returned actually relevant?)
  - Recall (R): \( R = \frac{TP}{TP + FN} \) (How close are the returned results to the gold set?)
  - F Measure (F): \( F = \frac{2PR}{P+R} \) (High when P & R are similarly high)


**Results: All Concerns**

- Dora outperforms Suade with statistical significance (α = 0.05)
- Dora, OR, and Suade perform significantly better than AND
- Dora and Suade not significantly different from OR (α = 0.05)
  - OR > Suade, p = 0.43
  - Dora > OR, p = 0.033
  - Dora > Suade, p = 0.0037
- Dora achieves 100% P & R (exact gold set) for 25% of the data—more than any other technique

---

**Results: By Concern**

- Overall trend also seen for most concerns
- Exceptions: 9 & 12
  - AND had much higher precision
  - Relevant methods contained both query terms

---

**Experimental Evaluation: Result Summary**

- Does an integrated lexical- and structural-based approach (Dora) outperform a purely structural approach (Suade)?
  - Dora outperforms Suade with statistical significance (α = 0.05)
  - Is a sophisticated lexical scoring technique required, or are naive lexical scoring techniques sufficient to identify the relevant neighborhood?
    - Although not statistically significant, Dora outperforms OR
    - Dora, Suade, & OR outperform AND (α = 0.05)
  - Integrated lexical- and structural-based approaches can outperform purely structural, but not all lexical scoring mechanisms are sufficient to do so

---

**“Source Code Exploration with Google”**

- Problem?
  - Limitations of state of the art
  - Goals
- Approach?
  - Benefit?
- Evaluation?
  - Results?
- Limitations?
- Conclusions

---

**Conclusion**

→ Integrated lexical- and structural-based approaches outperform purely structural ones

---

Limitations

- Relies on method, variable names
- Alternate relevance scores?
- How easy is it to find the seed/starting point?
Future Work

- Automatically find starting seeds
- Use more sophisticated lexical information
  - Synonyms, topic words (currency, price related to bidding)
  - Abbreviation expansion
- Evaluate on slicing

Dora Installation

1. From the Eclipse menu bar, select Help | Software Updates | Available Software
2. Select “Add Site…”
3. Enter in the URL, “http://www.cis.udel.edu/~gibson/dora/download” and hit OK
4. Select the Dora plug-in and hit Install..., then click Next
5. Accept the terms of the license agreement and click Finish
6. Restart your workspace
7. Go to Window | Preferences | Dora and make sure the ‘dot:’ preference points to your installation
   - set to /usr/bin/dot by default.
   - which dot

Open Discussion

- What do you think of this tool compared to the others we discussed on Friday?
- Would you use the tools differently? Under different circumstances?

Why So Much Time on Search?

- Seemed like it was solved
  - We have “Find”, “Grep”
  - Problems caused by larger code bases
- Different perspectives on a problem/solution
  - Be aware of what authors are telling you and what they aren’t
- Current research
  - NLP tools do not work for software-specific synonyms

Merging Software

- Who has experience with CVS? With Subversion?

Reminder: Midterm Wednesday

- 15% of grade
- Focus on UNIX commands, Bash scripting
  - UNIX philosophy
  - Reading and writing Unix commands
  - Understand purpose of various tools
- Software tools
  - What can they do?
- Tool types we’ve covered so far
  - Build tools
  - Search/navigation tools