MODIFYING COMPILER PARSING TO IMPROVE THE USABILITY OF BOA: A DOMAIN-SPECIFIC LANGUAGE AND INFRASTRUCTURE FOR ULTRA-LARGE-SCALE SOFTWARE REPOSITORY DATA MINING

SAMBHAV P. SRIRAMA
AGENDA

• Status Quo
• What is Boa?
• Barriers
• The Solution
• My Specific Work
STATUS QUO

• Online repositories

• Collaborative spaces

• Revision-control systems
WHAT IS BOA?

• Domain-specific language
  • Query-language for non-experts

• Data mining infrastructure
  • Contains software projects
BARRIERS

• Three big barriers to entering this field:
  • Requires expert knowledge of data mining
  • Requires an infrastructure to data mine (which can be expensive to construct)
  • Can take days to execute due to inefficiencies
THE SOLUTION

• Boa offers:
  • Infrastructure
    • Efficiently executes jobs
    • Scalable
  • Web-based UI and compiler
  • Easily learnable query language
    • Designed for higher level abstractions
MY SPECIFIC WORK: THE BOA COMPILER

• Compilers have two parts: a lexer and a parser
  • Lexer chops up code into tokens
  • Parser processes the tokens to determine intent

• Fixed using JavaCC framework
  • Implemented deep error recovery
EXAMPLE: AVERAGE CHURN RATES

```python
# what are the churn rates for all projects
p: Project = input;
counts: output mean[string] of int;

visit(p, visitor {
    before node: Revision -> counts[p.id] += len(node.files);
});
```

p: Projectttttt = input;

^
REFERENCES

- boa.cs.iastate.edu/
- http://design.cs.iastate.edu/papers/ICSE-14