Problem Comprehension & Metaprogramming

How do we program, and how can computers help?

by Kenny Harvey
Agenda

• What is programming (in a general sense)?
• How do we comprehend problems?
• Metaprogramming
• Topics for future work
What is programming (in a general sense)?

• “Writing computer programs” – circular/loose

• “Describing a solution to a computer, so it may solve a problem”

• Requires translation from the logical definitions of problems, through the abstract concepts in which humans think, to the electrical signals computers understand
What is programming (in a general sense)?

Describing a solution to a computer, so it may solve a problem.

(Comprehension) → (Solution) → (Description)
What is programming (in a general sense)?


- Understanding (Comprehension)
- Method-finding (Solution)
- Coding (Description)

Metaprog. Most PL
Agenda

• What is programming (in a general sense)?
• How do we comprehend problems?
• Metaprogramming
• Topics for future work
How do we comprehend problems?

No formal definitions, but empirically:

• **Pattern-recognition** – Brooks (1977) \[^1\]
  
  • “Genius is having seen it before” – Prof. Aho

• **Verbal/Numerical WM** – Siegmund (2014) \[^2\]

• **Levels of abstraction**
How do we comprehend problems?
How do we comprehend problems?

```
1000001101111111011111
1100000000000001110101
000001001000000110100
0101111111100000000001
```
How do we comprehend problems?

837DFC00 7504 8345FC01
How do we comprehend problems?

```assembly
cmpx $0; -4(%rbp)
jne .L2
addl $1; -4(%rbp)
```
How do we comprehend problems?

```cpp
if(!x) ++x;
```
How do we comprehend problems?

Secondary Notation – Schrepfer (2009) \[3\]

```c
if( !x )
    ++x;
```

How can we bridge this gap?

Hint: We’re programmers!
Agenda

• What is programming (in a general sense)?
• How do we comprehend problems?
• Metaprogramming
• Topics for future work
Metaprogramming

• Even more circular definitions
• “Describing a solution to a computer, so it may solve describing solutions to computers, so they may solve problems”
• C macros, JITs, Lex/YACC
• Not just Comprehend $\rightarrow$ Solve $\rightarrow$ Describe…
Metaprogramming

• Task is to comprehend, solve, and describe:
  • Comprehension
  • Solution
  • Description

<table>
<thead>
<tr>
<th>Comprehend</th>
<th>Comprehension</th>
<th>Solution</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solve</td>
<td>?</td>
<td>Abstraction</td>
<td>Linguistic WM</td>
</tr>
<tr>
<td></td>
<td>ML?</td>
<td>Meta Lang.</td>
<td>Grammars</td>
</tr>
</tbody>
</table>
Metaprogramming

• Let’s revisit our abstraction
  • Does it suggest comprehension?
  • Can we improve?

```cpp
if(!x)
    ++x;
```
Metaprogramming
Metaprogramming

My Package: increment x if unset

If (low) Condition

++

Re Package

x

My Package

x
Metaprogramming

My Package
increment x if unset
Demo

Agenda

• What is programming (in a general sense)?
• How do we comprehend problems?
• Metaprogramming
• Topics for future work
Topics for Future Work

• Semester project: logic database, autocomplete
• More software engineering tools
  • Tags for “Done”, “Tested”, could produce “Taint” system
  • Auto-generate tests
• Easier searching / browsing
• Background Cog. Psych. literature for UI
• Mobile coding!
References


