Optimization: Logic Optimization

CPSC 501: Advanced Programming Techniques
Fall 2022

Jonathan Hudson, Ph.D
Assistant Professor (Teaching)
Department of Computer Science
University of Calgary

Wednesday, November 23, 2022

UNIVERSITY OF CALGARY

Code Tuning Logic

Code Tuning

- Guidelines:
  - Save each version of your code using version control
  - Use the profiler to find a bottleneck
  - Tune the bottleneck, using just one technique
  - Measure the improvement
    - If none, revert to the prior version
  - Repeat until desired performance is achieved

Stop when found

Logic Techniques - Found

- Logic techniques
  - Stop testing when answer found
    - E.g.

```
boolean negFound = false;
for (int i = 0; i < input.length; i++) {
    if (input[i] < 0) {
        negFound = true;
    }
}
```

Logic Techniques – Found (cont'd)

- Is better as:

```
boolean negFound = false;
for (int i = 0; i < input.length; i++) {
    if (input[i] < 0) {
        negFound = true;
        break;
    }
}
```

Frequency

Logig Techniques - Frequency

- Order tests by frequency in switch and if-else structures
  - E.g.

```
if ((c == '+') || (c == '-')) {
    processMath(c);
} else if ((c >= '0') && (c <= '9')) {
    processDigit(c);
} else if ((c >= 'a') && (c <= 'z')) {
    processLetter(c);
}
```

Logig Techniques – Frequency (cont'd)

- Since letters are more common, is better as:

```
if ((c >= 'a') && (c <= 'z')) {
    processLetter(c);
} else if ((c >= '0') && (c <= '9')) {
    processDigit(c);
} else if ((c == '+') || (c == '-')) {
    processMath(c);
}
```

Switch

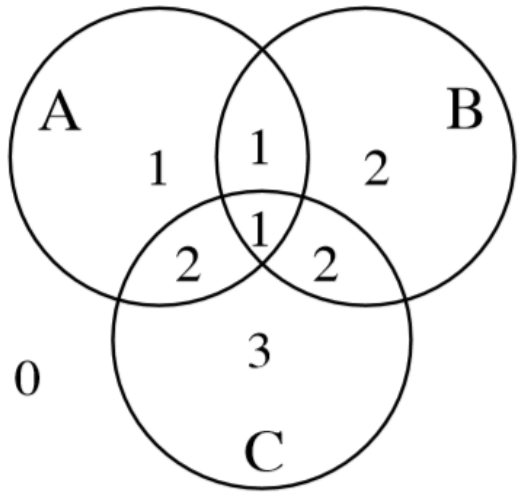
Logic Techniques - Switch

- Substitute switch statement for if-else construct, or vice-versa
  - In Java, an if-else construct can be faster than a switch …
  - But in C switch often optimized to a lookup table
  - Beware of assumptions (compiler optimizations, caching, predictive code execution mean this is variable sometimes)
  - Often the longer you run modern code on CPU with simple decisions the more difference gets ‘cached’ or ‘predictive branched’ out of differences

Lookup Table

Logic Techniques – Lookup Table

- Substitute table lookups for complicated expressions
  - E.g.



Logic Techniques – Lookup Table (cont'd)

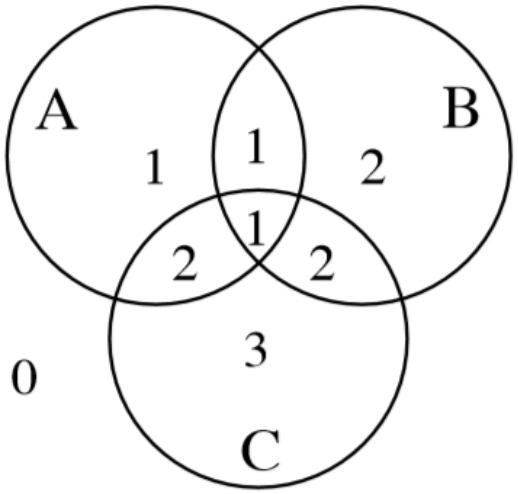
- Can be implemented using complicated logic:

```
if ((a && !c) || (a && b && c)) {
    category = 1;
} else if ((b && !a) || (a && c && !b)) {
    category = 2;
} else if (c && !a && !b) {
    category = 3;
} else {
    category = 0;
}
```

Logic Techniques – Lookup Table (cont'd)

- But is faster with a lookup table:

```
static int[][][] categoryTable = new int[2][2][2];
static {
    categoryTable[0][0][0] = 0;
    categoryTable[1][0][0] = 1;
    categoryTable[0][1][0] = 2;
    categoryTable[0][0][1] = 3;
    categoryTable[1][1][0] = 1;
    categoryTable[0][1][1] = 2;
    categoryTable[1][0][1] = 2;
    categoryTable[1][1][1] = 1;
}
```




```
category = categoryTable[a][b][c];
```

Lazy evaluation

Logic Techniques – Lazy Evaluation

- Use lazy evaluation
  - E.g. A 5000-entry table could be generated when the program starts
    - But if only a few entries are ever used, it may be better to compute values as needed, and then store them in the table
    - i.e. Cache them for further use

Onward to …
loop optimization.

Jonathan Hudson
jwhudson@ucalgary.ca
https://pages.cpsc.ucalgary.ca/~jwhudson/

UNIVERSITY OF CALGARY