Most Common 3Path Algorithm

Gray Meeks

My approach to analyzing an algorithm begins with laying out the absolute minimum amount of work that must be done. The below list highlights the bare minimum work that MUST be put in to the most common 3path algorithm to get the answer:

- Read every line in the log
- Organize entirety of log entries by user
- Count frequency of each 3path

To me, this particular problem screams symbol table / dictionary. The idea of storing a value based on a unique key (and doing so efficiently as dictionaries do) really enables a clean and effective solve for this algorithm. My approach to the problem was as follows:

UserDictionary: dictionary whose key is user and value is a list of paths

PathDictionary: dictionary whose key is a path and value is the path's frequency

```
For each line in log:
    1) Update UserDictionary based on new line
    2) If a new 3path has occurred:
        a. Update PathDictionary to reflect new frequency
        b. If frequency beats stored "best", overwrite best
```

This approach not only gets the answer, but it does so in a manner that uses appropriate data structures and minimizes the amount of work done working along the way rather than taking a more iterative, naïve approach. The last major benefit this approach provided was that, with one major change (added for loop) and a couple minor changes (numeric changes), I was able to easily convert my original solution from most common 3path to most common 'n'path where n is a user defined path length.