CTiB Project 1, Merging

Chester Henry Charlton, Viktor Konsgaard, (Chongming Chen)

2022/09/26

1

TERMINATES: We implement a while loop conditioned on at least one index being less than the length of at least one of the lists (x or y). Since at least one of these indices increases by 1 each loop, it is guaranteed to terminate when the value of the index reaches the value of the (finite) list.

CORRECT: it will generate the correctly sorted list since at any given step the algorithm takes the smallest element and adds it to our output list by using the fact that both of the lists are already sorted. So since it starts at the beginning of each list and each of those are the respective smallest elements in those lists, comparing those and selecting the smaller for the first element of our output list results in the globally smallest element. At the next iteration, the same process occurs but ignoring the previously selected "smallest element" and so we continue to select the smallest extant elements from both lists. This would NOT work if the lists were unsorted.

2

To determine the order of our merge-sort algorithm, we count the primitive operations to generate a function of complexity with respect to the input size. These counts are denoted as comments. We then use this function on further calculations on complexity.

```
i, j = 0, 0 # 2
z = [] # 1
while i < len(x) and j < len(y): # 3n + 3m
    if x[i] < y[j]: # 3n
        z.append(x[i]) # n
        i += 1 # n
    else: # m
        z.append(y[j]) # m
        j += 1 # m
z += y[j:] # 1
z += x[i:] # 1
return z # not counted (output)
```

The total primitive operations are $8n + 6m + 5$. Let $f(n, m) = 8n + 6m + 5$ and $g(n, m) = n + m$. To show that $f(n, m) \in g(n, m)$, we must show that there exists some real constant $c > 0$ such that for some $n_0, m_0 \geq 1$ it is true that $f(n, m) \leq c \cdot g(n, m)$ for every integer $n \geq n_0$.

$$\begin{aligned} f(n, m) &\leq c \cdot g(n, m) \\ c &= 8 \\ 8n + 6m + 5 &= 8n + 8m \\ m &= \frac{5}{2} \end{aligned}$$

Therefore our condition is satisfied for $(m_0, n_0, c) = (\frac{5}{2}, 8, 8)$. So $f(n, m) = 8n + 6n + 5 \in O(n + m)$, which is sufficient to say that our merge algorithm is also in $O(n + m)$ since the afformentioned function represents the algorithm's complexity.