

Homework 4

Thomas Kim tsk389 51835

David Munoz dam2989 51840

CS331 Algorithms and Complexity

Problem Q1. Given n samples and unambiguous `ImgComp`, design a divide and conquer algorithm which can report whether more than 50% of samples belong to a single class. Prove correctness and runtime.

main

```

1:  $S' = \text{signal-to-noise}(S)$ 
2: if  $\frac{|S'|}{|S|} > .5$  then
3:   return "YES"
4: else
5:   return "NO"
6: end if

```

signal-to-noise(S)

```

1: Let  $S := \{s_1, s_2, \dots, s_n\}$  be the set of images used as input to ImgComp
2: Let  $A := \{s_1, s_2, \dots, s_{\frac{n}{2}}\}, B = S - A$ 
3: if  $|A| = |B| = 1 \wedge \text{ImgComp}(s_1, s_2) = \text{"YES"}$  then
4:   return  $\{s_1, s_2\}$ 
5: else if  $|A| = |B| = 1 \wedge \text{ImgComp}(s_1, s_2) = \text{"NO"}$  then
6:   return  $\{s_1\}$ 
7: else
8:    $A' := \text{signal-to-noise}(A)$ 
9:    $B' := \text{signal-to-noise}(B)$ 
10:   $A'' := A'$ 
11:   $B'' := B'$ 
12:  for  $\forall b_i \in B$  do
13:    if  $\text{ImgComp}(a_1, b_i) = \text{"YES"}$  then
14:       $B'' := B'' \cup \{b_i\}$ 
15:    end if
16:  end for
17:  for  $\forall a_i \in A$  do
18:    if  $\text{ImgComp}(b_1, a_i) = \text{"YES"}$  then
19:       $A'' := A'' \cup \{a_i\}$ 
20:    end if
21:  end for
22:  return  $\max(A'', B'')$ 
23: end if

```

Proof of correctness

Proof. Claim: If more than 50% of S belongs to one set C , $a \in A' \wedge a \in C \vee b \in B' \wedge b \in C$ and `signal-to-noise(S)` will return C

Base case: $|S| = 4, |A| = |B| = 2$

Suppose more than 50% of S belongs to some category C . This means $|C| > \frac{S}{2} \implies |C| > 2$.

By the pigeonhole principle (either A or B will get a matching pair because at least 3 out of 4 match and there are 2 subsets), $|A'| = 2 \vee |B'| = 2$.

Without loss of generality, suppose $|A'| = 2$. If A' does not contain some $a \in C$, then $|C| \leq |S - A'| = 2 \implies$ contradiction.

Given the correctness of A' and/or B' , lines 12, 17, 22 will return some set $R \mid ||R| \geq |A| \implies R = C$

Induction:

Suppose $a \in A' \wedge b \in B' \wedge A' \cap C = \{\} \wedge B' \cap C = \{\}$

Claim: If C exists, $|A'| \geq \frac{|A|}{2} \vee |B'| \geq \frac{|B|}{2}$

Suppose the opposite was true. The maximal subset $S' \in S$ such that S' contains matching elements would be bounded by $|S'| < \frac{|A|}{2} + \frac{|B|}{2} = \frac{|S|}{2}$

Without loss of generality, suppose $\text{signal-to-noise}(A)$ returned $A' \mid |A'| \geq \frac{|A|}{2} = \frac{|S|}{4}$

Note that A' must contain some element $a \in C$, because if it didn't, $|C| \leq 2 \cdot \frac{|A|}{2} \implies |C| \leq \frac{|S|}{4}$

Note that $A - A' \cup C = \{\}$, because if some $a \in A \wedge a \notin A' \in C \wedge a' \in A' \in C, a \in A'$

By line 12, $A'' = C$, because the algorithm loops through B and by the previous note, $A - A' \cup C = \{\}$

Finally, if the largest subset $S' \subset S \mid \frac{|S'|}{|S|} \leq \frac{1}{2}$, the algorithm will return false by line 5 of main and lines 13 and 18 of $\text{signal-to-noise}(S)$.

□

Proof of Runtime

Proof. $\text{signal-to-noise}(S)$ creates 2 subproblems of size $\frac{n}{2}$ by lines 2, 8, 9

$\text{signal-to-noise}(S)$ calls ImgComp n times by lines 12, 17, since $|A'| \leq |A| = \frac{n}{2} \wedge |B'| \leq |B| = \frac{n}{2}$

Therefore, the recurrence for $\text{signal-to-noise}(S)$ is $T(n) = 2T(\frac{n}{2}) + \theta(n)$

By master theorem, $f(n) = \theta(n) = \theta(n^1 \log^0(n)) \implies T(n) = \theta(n \log(n))$

□

Problem Q2. $\text{FindLocalMinimum}(G)$

- 1: Define $\text{quad}(v)$ to be (the set of vertices in the same quadrant of G as v) unioned with (any elements of B which border the previous set).
- 2: Let B, C, T, TC be defined on G as specified in the problem.
- 3: $\text{minVert} \leftarrow \min_{v \in S \cup T} F(v)$
- 4: **if** $\text{minVert} \in S \vee \text{minVert} \in TC$ **then**
- 5: return minVert
- 6: **else**
- 7: return $\text{FindLocalMinimum}(\text{quad}(\text{minVert}))$
- 8: **end if**

Lemma 1: For all such grids G , there exists a local minimum.

Proof. There are a finite number of squares each with unique weight w_i .

Let the set $W = \{w_1, w_2, \dots, w_i\}$

By the well-ordering principle, there exists a least element of W .

By definition, this least element is less than each of its neighbors, because it is less than every other weight in the set.

Thus, it is a local (global) minimum.

□

Base case: Note that there are no bases cases by lemma 1.

Proof of correctness:

Proof. Suppose there is no local minimum when $S' \cup T' = G'$, where G' is the subgrid in the final iteration.

Take v_c to be the center vertex of G' .

Note that in the previous iteration, v_c must have been in T , since it couldn't have been in $S \cup TC$ and

$S' \cup T' = G'$ This means v_c is the smallest element in G'
 This means v_c is a local minimum.

□

Proof of runtime:

Proof. The recurrence associated with this algorithm is $T(n) = T(\frac{n}{2}) + \theta(n)$

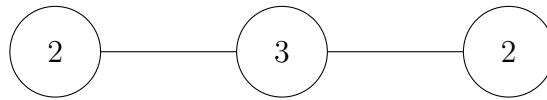
Expanding this recurrence, $T(n) = T(\frac{n}{4}) + c\frac{n}{2} + n$

$$= \sum_{i=1}^n c \frac{n}{2^i}$$

Therefore the total runtime is $\theta(n)$

□

Problem Q3(a).



In this example, the middle node will be chosen and the two side nodes discarded for a total weight of 3.

If the middle node was discarded and the two side nodes chosen, the total weight would have been $4 > 3$.

Problem Q3(b).



In this example, the maximum weight independent set includes nodes v_1 and v_4 , but 1 is odd and 4 is even.

Problem Q3(c). Let the set of independent vertices $S_i \subset \{v_1, v_2, v_3, \dots, v_i\}, i \leq n$ have some weight W_i

Let each vertex v_i have some weight w_i

Suppose S_i includes vertex v_i . S_i can therefore not include v_{i-1} , so $W_i = W_{i-2} + w_i$

Suppose S_i does not include v_i . $W_i = W_{i-1}$

Base case(s): $W_0 = 0, W_1 = w_1$

Algorithm: $W_i = \max(W_{i-1}, W_{i-2} + w_i)$

return W_n

Memoization: W_j only has to be computed once for each unique j .

Proof of correctness:

Proof. Suppose W_n is not the maximum weight.

This means either W_{i-1} or W_{i-2} wasn't maximized.

This continues, with some W_{i-k} not being maximized.

However, W_0 and W_1 are always maximized by definition.

Thus, $W_{i-k}, k = i$ must be maximized.

This contradicts the original assumption, since each step is guaranteed to maximize the weight if the previous step is maximized.

□

Proof of runtime:

Proof. W_0, W_1, \dots, W_n will all be calculated exactly once due to memoization.

The time required to calculate W_i is $O(1)$, since it is a single comparison and a single addition.

Therefore, the total complexity is $O(n)$.

□