CS 4820, Spring 2017 Homework 5, Problem 1

Name: Yuxiang Peng

NetID: yp344

Collaborators: jl3455, zl542

1. We are going to design an D&C algorithm for solving this question. As we can summarize from the question, we can only use the method of judging two cards equivalent or not. We divide all n cards into two parts of equal size. The first set contains n/2 cards and the second set contains the rest. We keep dividing these two sets in similar approaches thus we get 4 n/4 sets. By dividing the set, we find that the card which have more than n/2 equivalent cards must at least have m/2 equivalent cards in any one of our divided set of a size of m. Then we need to traverse all the cards and to count the number of all equivalent cards towards this card. If we finally find a card meet the requirements, we return this card. If not, we keep looking for the card until all the sets are read. The base statuses are about the set of size 1 and size 2. If we do not have a smallest divided size of size 1, it means the size of the overall set is even. The opposite case is the overall size is odd. When we have a smallest divided size of set of 1, it can be the possible solution. While for the smallest divided size of 2, unless the two cards are equivalent, we return either one of the card which can be the possible solution.

   The designed algorithm is as follows:

   Set C;

   FindModeCard(C)

   If the size of the set is 1, return this card;

   If the size of the set is 2

      If the two cards are equivalent, return either one of the card;

      Else return null;

   C1 = the first half the cards C;

   C2 = the rest of the cards C;

   Solu1 = FindModeCard(C1);

   Solu2 = FindModeCard(C2);

   If(Solu1 != null)

      Traverse all the cards to find if the Solu1 is the card that has more than n/2 equivalent cards in the overall set C; If it is, return Solu1;

If(Solu1 != null)

Traverse all the cards to find if the Solu1 is the card that has more than n/2 equivalent cards in the overall set C; If it is, return Solu2;

Return;

Running Time:

Since we divide the overall problem into 2 equal size sub-problems and the operating time is the traverse time * O(n). We get the formula T(n) = 2*T(n) + C*O(n). According to the piazza and the chapter, the running time is O(nlogn)

Proof:

Let us assume that we have a card Cn that has more than n/2 equivalent cards in the overall set which does not meet the requirement, that is to say it is not returned in any one of the smallest divided set. If we get m sets and the average size of the set is 2, the number of the equivalent cards of Cn including itself is less than m*2/2=m. Because m*2 = n, that is to say the number is less than n/2. It is impossible. So if we have the answer Cn, it must at least have more than size/2 equivalent cards in any one of the divided set. Then if we traverse the overall set, we can easily test whether it is the answer or not. It proves the correctness of the algorithm.