Introduction to Algorithms	Problem Set 4
CS 4820 Spring 2016	Lillyan Pan

1 Problem 1

1.1 Algorithm

Assume the input is a sorted list of numbers from 0 to $n-1$.

1. If the list contains two elements or less then return the list.
2. Else divide the list into two halves based on parity:
   - A contains only even numbers
   - B contains only odd numbers
3. Recursively split resulting lists based on the parity of the resulting numbers when
   - if elements are even, divide through by 2
   - if elements are odd, subtract each element by 1 and divide through by 2
4. Build new list M by
   - multiplying through the left split list by 2
   - multiplying through the right split list by 2 and add 1 to each element
   - append the right split list to the end of the left split list
   - Repeat until all split lists are merged
5. Return M

1.2 Runtime

The height of the recursion tree is bounded by $log\ n$ as each iteration divides the list length by 2. Each iteration takes at most $O(n)$ time to process the numbers' parity. Hence, the total running time for the algorithm is $O(nlog\ n)$.

1.3 Correctness

Lemma 1. *For numbers ordered $p_0, p_1, ..., p_{n-1}$, if p_i and p_k have different parity, they cannot be part of an equal gap triple with $i < j < k$.*

Proof of Lemma 1. Given p_i and p_k have different parity, then if p_j is even, the gaps $p_j - p_i, p_k - p_j$ will be of different parities, as subtraction between two even numbers results in an even number and subtraction between numbers of different parities results in an odd number. Similarly, if p_j is odd, the gaps $p_j - p_i, p_k - p_j$ will be of different parities by the same reasoning. Because $p_j - p_i, p_k - p_j$ are of different parities, p_i, p_j, p_k cannot form an equal gap triple. Hence, the lemma is proven as necessary.

Lemma 2. *The above algorithm correctly sorts the input list such that if there is a triple* $i < j < k$, $p_j - p_i \neq p_k - p_j$, *where numbers are ordered* $p_0, p_1, ..., p_{n-1}$.

Proof of Lemma 2. We will prove Lemma 2 using a proof by induction.

If $n \leq 2$, by definition of a gap triple, a gap triple cannot occur.

Now, consider $n = 2^{k-1}$ for some $k > 2$. Assume this merges lists of length 2^{k-2}, that preserves the property that the resulting list contains no gap triples. For list L of length $n = 2^k$, the previous split of lists of length 2^{k-1} can be mapped to list L by multiplying through the left split list by 2, multiplying through the right split list by 2 and add 1 to each element, and then appending the right split list to the end of the left split list. By the inductive hypothesis, the split lists of length 2^{k-1} contain no gap triples. Because the two lists' elements are of different parities, list L contains no gap triples.

For $n = 2^k$, the last step merges two lists of even and odd numbers. By Lemma 1, if p_i is from one list, and p_k is from the other list of a different parity, then p_i, p_j, p_k cannot form an equal gap triple.