Übungen zur Algorithmischen Bioinformatik I

Blatt 4

Xiheng He

April 2021

1. Aufgabe (10 Punkte):

```
SumExist(int[] S, int x, int n)
begin
    int l = 0;
    int r = n − 1;
    /* If x = sum loop ends, l must be smaller than r                    */
    while (l ≤ r) do
        if (S[l] + S[r] == x) then
            return true;
        /* if x < sum, reduce the sum until equal to x                   */
        else if (S[l] + S[r] > x) then
            r − −;
        /* if x > sum, increase the sum until equal to x                 */
        else
            l + +;
        return false;
end
```

Analyse:

Eingabegröße: $n = |S|$

Laufzeit: Vergleich benötigt genau ein mal Pro Rekursion und nach Vergleich wird entweder l um 1 erhöht oder r um 1 reduziert. Egal wie l oder r sich verändert, die Kosten der arithmetischen Operationen beträgt jeweils l und $n-1-r$. Damit ist die gesamte Kosten $l+n-1-r$ wobei $l-r \leq 0$ und While-Schleife kann maximal n mal durchlaufen. Somit ist die Laufzeit $O(n)$.

Korrektheit: Der Algorithmus wird am Anfang mit dem Summe der ersten und letzten beiden Elementen und x vergleichen. Dazu gibt es drei Fälle: Wenn $x = S[0] + S[n-1]$ wird *true* zurückgegeben; Wenn $x > S[0] + S[n-1]$ bedeutet es dass die Summe erhöht werden muss, dann wird ein Element großer als das kleineste Element $S[0]$ benötigt. Anderfalls wird ein kleinere Element $< S[n-1]$ benötigt. Ist ein großer Element oder große Summe nötig, muss das nach rechts gesucht werden, sonst nach links. Damit wird l steigen und r verringert bis $l = r$. Jedes Mal wenn sich der Pointer l oder r bewegt checken wir ob $S[l] + S[r] == x$. Wenn solche zwei Elemente am Ende noch nicht gefunden wurden, dann gibt überhaupt nicht.