# Übungen zur Algorithmischen Bioinformatik I Blatt 0

Xiheng He

April 2021

**2. Aufgabe (10 Punkte):**

**Algorithm 1:** MSS (int[] a, int n)

```
begin
    $maxscore := 0;$    $\ell := 1;$    $r := 0$
    $rmaxscore := 0;$    $rstart := 1;$    $sstart := \ell;$
    for ( $i := 1; i \leq n; i{+}{+}$ ) do
        if ($rmaxscore > 0$) then
            $rmaxscore := rmaxscore + a[i];$
        else
            $rmaxscore := a[i]; rstart := i;$
    if ($rmaxscore > maxscore$) then
        $maxscore := rmaxscore; \ell := rstart; r := i;$
        $allScores := \{(maxscore, \ell, r)\}$
        $ssstart := \ell;$
    else if ($rmaxscore = maxscore \wedge ssstart \neq rstart$) then
        $allScores := allScores \cup \{(maxscore, rstart, i)\};$
        $sstart := rstart;$
end
```

Der Algorithmus wird linear durchlaufen, daher beträgt die Laufzeit $O(n)$

**Algorithm 2:** SMSS (int[] a, int n)

```
begin
    $maxscore := 0;$    $\ell := 1;$    $r := 0$
    $rmaxscore := 0;$    $rstart := 1;$    $sstart := \ell;$    $min\_len := +\infty$
    for ( $i := 1; i \leq n; i{+}{+}$ ) do
        if ($rmaxscore > 0$) then
            $rmaxscore := rmaxscore + a[i];$
        else
            $rmaxscore := a[i]; rstart := i;$
    if ($rmaxscore > maxscore$) then
        $maxscore := rmaxscore; \ell := rstart; r := i;$
        $allScores := \{(maxscore, \ell, r)\}$
        $min\_len := \ell - r + 1;$
        $sstart := \ell;$
    else if ($rmaxscore = maxscore$ and $sstart \neq rstart$) then
        if ($\ell - r + 1 < min\_len$) then
            $allScores := \{(maxscore, \ell, r)\}$
            $sstart := rstart;$
        else if ($\ell - r + 1 = min\_len$) then
            $allScores := allScores \cup \{(maxscore, rstart, i)\};$
            $sstart := rstart;$
end
```

Die Laufzeit beträgt weiterhin $O(n)$

**Algorithm 3:** MSS_All (int[] a, int n)

```
begin
    $(maxscore, \ell, r) :=$ MSS_DC$(a, 1, n)$;
    int[] $ss\_start$;
    int[] $ss\_end$;
    int[] $ss\_score$;
end
$(\text{int}, \text{int}, \text{int})$MSS_DC(int[] $a$, int $i, j$);
begin
    if $(i = j)$ then
        if $(a[i] > 0)$ then
            return $(a[i], i, i)$
        else
            return $(0, i, i - 1)$
    else
        $m := \lfloor \frac{i+j-1}{2} \rfloor$;
        $(s_1, i_1, j_1) :=$ MSS_DC$(a, i, m)$;
        $(s_2, i_2, j_2) :=$ MSS_DC$(a, m + 1, j)$;
        $i_3 := m$;
        $s := a[i_3]$;
        $simax := s$;
        for ( $k := i_3 - 1; k \geq i; k - -$ ) do
            $s := s + a[k]$;
            if $(s > simax)$ then
                $simax := s$;
                $i_3 := k$;
        $j_3 := m + 1$;
        $s := a[j_3]$;
        $sjmax := s$;
        for ( $k := j_3 + 1; k \leq j; k + +$ ) do
            $s := s + a[k]$;
            if $(s > sjmax)$ then
                $sjmax := s$;
                $j_3 := k$;
        $s_3 := simax + sjmax$;
        if $(len(ss\_score) \neq 0$ and $maxss\_score < maxs_1, s_2, s_3)$ then
            $ss\_start$ = new int[]; $ss\_end$ = new int[]; $ss\_score$ = new int[];
        if $(maxs_1, s_2, s_3 = s_1)$ then
            $ss\_start = ss\_start + i_1$;
            $ss\_end = ss\_end + j_1$;
            $ss\_score = ss\_score + s_1$;
            return $(s_1, i_1, j_1)$;
        if $(maxs_1, s_2, s_3 = s_2)$ then
            $ss\_start = ss\_start + i_2$;
            $ss\_end = ss\_end + j_2$;
            $ss\_score = ss\_score + s_2$;
            return $(s_2, i_2, j_2)$;
        else
            $ss\_start = ss\_start + i_3$;
            $ss\_end = ss\_end + j_3$;
            $ss\_score = ss\_score + s_3$;
            return $(s_3, i_3, j_3)$;
end
```

Die Laufzeit ist $O(nlogn)$ da es sich um Divide-Conquer Algorithmus handelt.