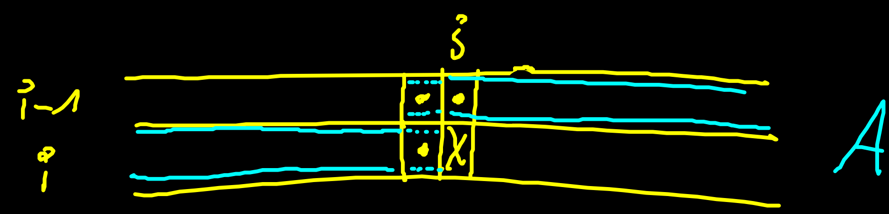
7a) OBdA $n \leq m$

Idee: Alg. aus VL, aber nur letzte Zeile aufheben.



procedure SCORE(S,T) {
    A := SCORELINES(S,T)
    return A[n]
}

procedure SCORESLINES(S,T) {
    A := init Array[1..n]
    for j = 1, ..., n
        A[j] := j·g
    for i = 1, ..., m
        match := (i-1)·g ;

for $j = 1, \dots, n$ {

$$\text{new} := \min \begin{cases} \text{match} + p(S_i, T_j) \\ A[j] + g \\ g + \begin{cases} A[j-1] &, j > 1 \\ i \cdot g &, j = 1 \end{cases} \end{cases}$$

match := $A[j]$

$A[j]$:= new

}

}

return A

}

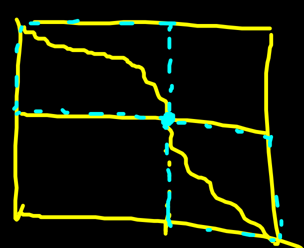
Laufzeit: berechnen $m \cdot n$ Werte $\rightsquigarrow \Theta(nm)$

Speicher: Hilfsvariablen + Array der Länge n

$\rightsquigarrow \Theta(n)$

b)

$\lfloor \frac{m}{2} \rfloor = i^*$



j^*

$\rightsquigarrow$ rekursiv auf (S_{1,i^*}, T_{1,j^*})

und $(S_{i^*+1,n}, T_{j^*+1,m})$

und Ergebnisalignments konkat.

aus Schnittpunkt eines „optimalen Pfades" mit Zeile i^*.

Ergebnis kodieren als Wort über $\{I, D, M\}$

procedure ALIGN(S,T) // $n = |T|$, $m = |S|$

if $(m = 0)$ {

```
    return I^n
  }
  else if (n = 0) {
    return D^m
  }
  else if (m = 1) {
    i := argmin_{1 ≤ i ≤ n} p(S_i, T_1)
    return I^{i-1} M I^{n-i}
  }
  else {
    i* := ⌊m/2⌋
    A := ScoresLines(S_{1,i*}, T)
    B := ScoresLines(S^R_{i*,m}, T^R)
    j* := argmin_{0 ≤ j ≤ n} A[j] + B[n-j]
    return ALIGN(S_{1,i*}, T_{1,j*})
         + ALIGN(S_{i*+1,m}, T_{j*+1,n})
}
```

Speicherbedarf: • A, B recyclen

$\rightsquigarrow \Theta(n)$

• Stack: Rekursionstiefe

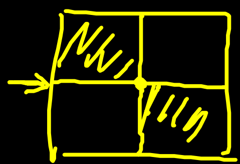
$O(\log_2 m)$

$\rightsquigarrow O(n + \log_2 m)$

Ergebnis

• $O(n+m)$

$\rightsquigarrow O(n+m)$ insgesamt



Zeit: $O\left(nm + \frac{1}{2}nm + \frac{1}{4}nm + \dots\right)$

$\subseteq O(nm \cdot 2)$

Referenzen:

1) Hirschberger (D&C für LCS)
2) Myers, Miller (1988)
   Optimal alignments in linear space.

8| i) $\in NP$: klar, denn: gegeben ein Alignment rechnet man die Score in Zeit $O(L \cdot k)$ aus (spaltenweise, WHILE-Modell)

$\rightsquigarrow$ Zertifikat in Polyzeit (det) verifizierbar. $\square_{i)}$

ii) $\in NP$-hard:

gzz: Dec-(0,1)-Sh-SSeq-Problem $\le_p$ MASP

Reduktionsfunktion:

$$I_{DSSP}\left(\{S_1, \dots, S_k\} \in \{0,1\}^*, \; m \in \mathbb{N}\right)$$

$$\overset{f}{\longmapsto} \left(\{S_1, \dots, S_k\}, \delta, d\right)$$

mit
$$p(0,0) = p(1,1) = g = 1,$$
$$p(0,1) = p(1,0) = m\binom{k}{2} + 1,$$
$$d = m\binom{k}{2}.$$

zz: I_{DSSP} ist Ja-Instanz für DSSP $\iff$ $f(I_{DSSP})$ ist Ja-Instanz für MASP

„$\Rightarrow$" gegeben $S_1, \dots, S_k$, m, sodass es eine Superseq. w mit Länge $t \leq m$ gibt, baue Alignment:

$$\left.\begin{array}{l} w \\ S_1 \\ \vdots \\ S_k \end{array}\right\} \text{ mit } - \text{ passend auffüllen (geht, da } w \text{ SS)}$$

mit Kosten $t\binom{k}{2} \leq m\binom{k}{2}$

$\leadsto$ $f(I_{DSSP})$ ist Ja-Inst. $\square$ $\Rightarrow$

„$\Leftarrow$" Haben also MA mit Score $\leq d \leq m\binom{k}{2}$

$\leadsto$ keine Mismatches,
Alignment hat Länge $\leq m$

$\leadsto$ eindeutiger Consensus ist SS
mit Länge $\leq m$
(Laufzeit f ist poly., klar) $\square$ $\Leftarrow$

$\square$

9) Sei $v_1, \dots, v_n$ ein Pfad in G mit
$s = v_1$, $w = v_n$, $w \in V$ bel.

$$g^*(v_1, \dots, v_n) = \sum_{i=1}^{n-1} g^*(v_i, v_{i+1})$$

$$= \sum_{i=1}^{n-1} \Big(g(v_i, v_{i+1}) + \xi(v_{i+1}) - \xi(v_i$$

$$= \sum_{i=1}^{n-1} g(v_i, v_{i+1}) + \xi(v_n) - \xi(v_1)$$

$$= g(v_1, \dots, v_n) + \underbrace{\xi(w) - \xi(s)}_{=: c}$$

Da c unabh. von dem konkreten Pfad ist, liefert Minim. gemäß g bzw. g^* einen Pfad mit gleichen (minimalen) Kosten. $\square$