

What comes next?

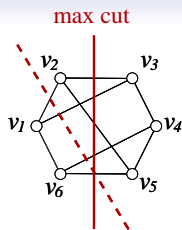
example for a hardness result:

CROSS×PLAIN→CROSS, 'all operations' is Max SNP-hard

(i.e. without the restriction $w_a = \frac{w_r + w_b}{2}$).

Max-Cut-3

- formal:
 - let $G = (V, E)$ be a graph
 - a *cut* in G is a set of edges s.t. there is a partition $V_1 \uplus V_2 = V$, where for every edge one endpoint is in V_1 , the other in V_2 .
 - *Max-Cut-3*: given graph g with degree ≤ 3 , find cut with maximal cardinality.



Theorem

Max-Cut-3 is Max-SNP-hard

Remark An optimization problem is Max-SNP-hard iff it does not have a PTAS (Polynomial Time Approximation Scheme).

A *PTAS* is an algorithm that takes an instance of a maximization problem and a parameter $\epsilon > 0$ and, in polynomial time, produces a solution that is within a factor $1 - \epsilon$ of being maximal.

Reduction of Max-Cut-3 to $CROSS \times PLAIN \rightarrow CROSS$

Reduction idea:

represent Max-Cut-3 problem as alignment problem

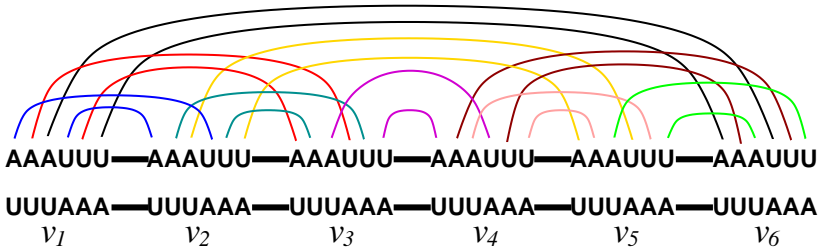
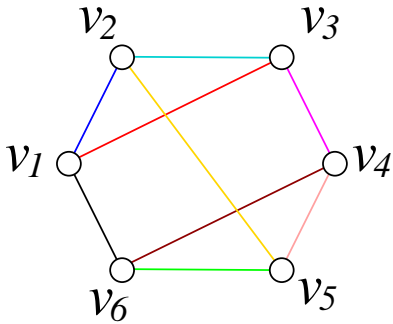
$CROSS \times PLAIN \rightarrow CROSS$ such that optimal alignment corresponds to maximum cut.

→ if Max-Cut-3 can be solved using the alignment problem, the alignment problem must also be Max-SNP-hard.

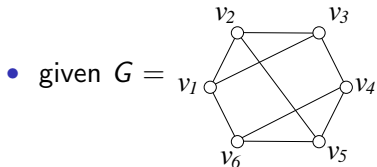
Plan

- show how to represent graph G as input of alignment problem (e.g. Sequences S_1, S_2 + structure P_1 for S_1)
- show how optimal alignment corresponds to maximum cut for G .

Representation of Graph G as Alignment Problem (Example)



Representation of Graph G as Alignment Problem (formally)

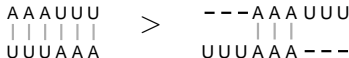


- sequences
 - $S_1 = (AAAUUU(C)^c)^{n-1}AAAUUU$, and
 - $S_2 = (UUUAAA(C)^c)^{n-1}UUUAAA$.
- the segments $AAAUUU$ in S_1 and $UUUAAA$ in S_2 correspond to the nodes
- each edge (v_i, v_j) of G corresponds to two arcs in P_1 : one connecting an A of the i -th segment with a U of the j -th segment and one connecting a U of the i -th segment with an A of the j -th segment.
- Cs are used to avoid alignment of different segments, and their number c depends on the ratio $\frac{\min(w_b, w_a, w_r)}{w_d}$
 - ← arc changes
 - ← base deletion

Correspondence of Optimal Alignment and Max Cut

Properties of Optimal Alignment

- we choose c such that every optimal alignment must match all C s
- we choose a scoring with $w_m > w_d$ and $2w_a > w_b + w_r$.
- $w_m > w_d$ implies no base mismatch:

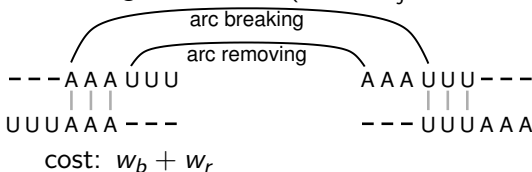


- two alignment types for each node v_i :
 - A-type:

$$\begin{array}{c}
 \text{--- A A A U U U} \\
 | | | \\
 \text{U U U A A A ---}
 \end{array}$$
 - U-type:

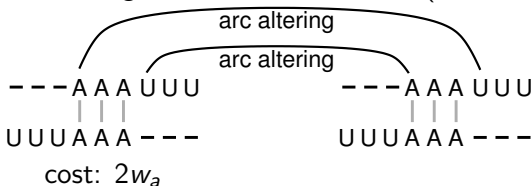
$$\begin{array}{c}
 \text{A A A U U U ---} \\
 | | | \\
 \text{--- U U U A A A}
 \end{array}$$

- A-type $:\Leftrightarrow$ node in V_1 U-type $:\Leftrightarrow$ node in V_2 .
- cost for each edge of the cut (v_i and v_j have different type)



Correspondence of Optimal Alignment and Max Cut

- cost for each edge that is not in the cut (v_i and v_j have same type)



- total cost for alignment:
 - $V_1 =$ all A-type nodes
 - $V_2 =$ all U-type nodes
 - n nodes, each degree 3 $\Rightarrow \frac{3n}{2}$ edges
 - $k := |\text{cut}(V_1, V_2)|$

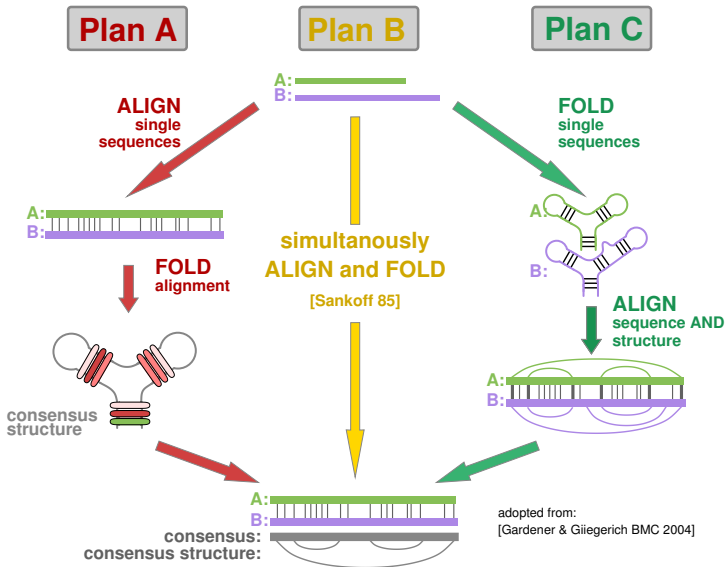
$$C = k(w_b + w_r) + \left(\frac{3n}{2} - k\right) 2w_a + n3w_d$$

assumption: $2w_a > w_b + w_r \Rightarrow > 0$

$$= 3n(w_a + w_d) - k \overbrace{(2w_a - w_b - w_r)}$$

- $\Rightarrow C$ minimal $\equiv k$ maximal
- \Rightarrow maximal cut \equiv minimal edit distance.

Approaches for Alignments of RNAs

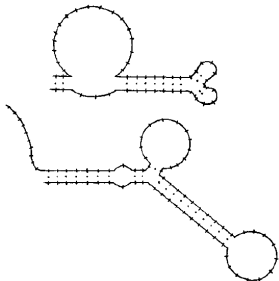


Simultaneous Alignment and Folding: Sankoff (1985)

- What do we want? What means folding into a common structure?
- First idea: preserve “shape” \equiv branching structure
- Formally: let $i_1 < i_2 \dots < i_v$ in a and $j_1 < j_2 \dots < j_w$ in b be the positions in pairs that limit multiloops or are external (*branching configuration*)

Then: structures *equivalent* (according to branching)
iff $v = w$, and $(i_f, i_g) \in P_a$ if and only if $(j_f, j_g) \in P_b$

- **finding good equivalent structures not sufficient:**



- Hence: minimize edit distance + energies (of 2 equiv. structures)

Sankoff Problem Definition

- Idea: Sankoff = Zuker Folding + Needleman/Wunsch Alignment
- IN: two sequences a and b
- find two **equivalent** structures P_a and P_b
and **compatible** alignment A of a and b
such that $Energy(a, P_a) + Energy(b, P_b) + EditDistance(A)$ minimal
- where: $Energy$ yields (loop-based) Turner free energy,
 $EditDistance$ is edit distance (base mismatch x , indel y)
- what means *compatible*?
alignment must be “consistent” with branching structure
formally: the base pairs $(i_f, i_g) \in P_a$ and $(j_f, j_g) \in P_b$ (from Def. of equivalent) must be aligned to each other

Constraints

We want to find the optimal structures + alignment with the following constraints:

constraints on the **predicted structures**:

- must be equivalent (intuitively: same kind of multiloops)

constraints on the **alignment**:

- multiloops must be aligned to their equivalent partner
- hairpin loops must be aligned to their equivalent partner
- each 2-loop (or stacking or bulge) must be aligned to exactly one other 2-loop or must be entirely aligned to a gap.

Edit distance of sub-sequences

- distance based score

x = base mismatch

y = base deletion/insertion

- $D(i, j; h, k)$ minimum sequence alignment cost between sequences $a_i \dots a_j$ and $b_h \dots b_k$.

$$D(i, j; h, k) = \min \begin{cases} D(i, j-1; h, k-1) + x & \text{if } a_j \neq b_k \\ D(i, j-1; h, k-1) & \text{if } a_j = b_k \\ D(i, j-1; h, k) + y \\ D(i, j; h, k-1) + y \end{cases}$$

- Recursion:

$$= \min \begin{cases} D(i+1, j; h+1, k) + x & \text{if } a_i \neq b_h \\ D(i+1, j; h+1, k) & \text{if } a_i = b_h \\ D(i+1, j; h, k) + y \\ D(i, j; h+1, k) + y \end{cases}$$

- Initialization: $D(i, i; h, h) = \begin{cases} x & \text{if } a_i \neq b_h \\ 0 & \text{else} \end{cases}$

Recall Zuker

- Energies: $e(s)$, where s is k -loop (or $s = \phi$ for empty structure)
- $F(i, j)$ “free”, minimum energy for subsequence $a_i \dots a_j$
- $C(i, j)$ “closed”, minimum energy for subsequence where $(i, j) \in P$
- Zuker Recursion:

$$(6) \quad C(i, j) = \min_{k \geq 1} \min_{\substack{s \text{ is a } k\text{-loop} \\ \text{closed by } (i, j)}} \left\{ e(s) + \sum_{\substack{(p, q) \\ \text{accessible} \\ \text{from } (i, j)}} C(p, q) \right\}$$

with the initial conditions $C(i, i) = \infty$, and

$$(7) \quad F(i, j) = \min \left\{ C(i, j), \min_{i \leq h < j} [F(i, h) + F(h+1, j)] \right\},$$

with the initial conditions $F(i, i) = 0$.

- Problem: (6) requires time proportional to n^{2K} where K maximum k in k -loops

Usual Simplification

- $e(s)$ for k -loops with $k \geq 3$ (multiloops)

$$e(s) = A + (k - 1)P + uQ$$

- New matrix: $G(i, j)$ for multiloops
- Recursion:

$$(9) \quad C(i, j) = \min \begin{cases} e(s), & s \text{ is the hairpin closed by } (i, j), \\ \min \{e(s) + C(p, q)\}, & s \text{ a 2-loop closed by } (i, j) \text{ with} \\ & (p, q) \text{ accessible, } u = p - i + j - q - 2 \leq U, \\ \min_{i < h < j-1} \{G(i+1, h) + G(h+1, j-1) + A\}, \end{cases}$$

where

$$(10) \quad G(i, j) = \min \begin{cases} C(i, j) + P, \\ \min_{i \leq h < j} \min \begin{cases} (G(i, h) + (j - h)Q), \\ G(i, h) + G(h + 1, j), \\ (h - i + 1)Q + G(h + 1, f), \end{cases} \end{cases}$$

and, as in (4),

$$(11) \quad F(i, j) = \min \begin{cases} C(i, j), \\ \min_{i \leq h < j} \{F(i, h) + F(h + 1, j)\}, \end{cases}$$

with initial conditions $C(i, i) = \infty$, $F(i, i) = 0$ and $G(i, i) = \infty$.

Simultaneous Alignment and Folding

- Extend definition of $D(i_1, j_1; i_2, j_2)$
 - if $i_1 > j_1$, then cost for deleting $b_{i_2} \dots b_{j_2}$.
 - if $j_2 > i_2$, then cost for deleting $a_{i_1} \dots a_{j_1}$.
- $F(i_1, j_1; i_2, j_2)$ minimum cost (sum of alignment and free energy) for $a_{i_1} \dots a_{j_1}$ and $b_{i_2} \dots b_{j_2}$.
- $C(i_1, j_1; i_2, j_2)$: minimum cost for $a_{i_1+1} \dots a_{j_1-1}$ and $b_{i_2+1} \dots b_{j_2-1}$
under condition $(i_1, j_1) \in P_a$ and $(i_2, j_2) \in P_b$

Simultaneous Alignment and Folding: Multiloop

- $G(i_1, j_1; i_2, j_2)$: matrix for multiloop alignment
- Recursion for G

$$G(i_1, j_1; i_2, j_2)$$

$$= \min \left\{ \begin{array}{l} C(i_1, j_1; i_2, j_2) + 2P + \overbrace{D(i_1, i_1; i_2, i_2)}^{\text{match } i_1 \text{ and } i_2} + \overbrace{D(j_1, j_1; j_2, j_2)}^{\text{match } j_1 \text{ and } j_2} \\ \min_{\substack{i_1 < h_1 < j_1 \\ i_2 < h_2 < j_2}} \left\{ \begin{array}{l} G(i_1, h_1; i_2, h_2) + (j_1 - h_1 + j_2 - h_2)Q \\ \quad + D(h_1 + 1, j_1; h_2 + 1, j_2), \\ G(i_1, h_1; i_2, h_2) + G(h_1 + 1, j_1; h_2 + 1, j_2), \\ (h_1 - i_1 + 1 + h_2 - i_2 + 1)Q \\ \quad + D(i_1, h_1; i_2, h_2) + G(h_1 + 1, j_1; h_2 + 1, j_2) \end{array} \right. \end{array} \right.$$

Simultaneous Alignment and Folding: “free”

- Recursion for F

$$F(i_1, j_1; i_2, j_2) = \min \begin{cases} C(i_1, j_1; i_2, j_2) + D(i_1, i_1; i_2, i_2) + D(j_1, j_1; j_2, j_2) \\ \min_{\substack{i_1 < h_1 < j_1 \\ i_2 < h_2 < j_2}} F(i_1, h_1; i_2, h_2) + F(h_1 + 1, j_1; h_2 + 1, j_2) \\ D(i_1, j_1; i_2, j_2) \end{cases}$$

- with initial conditions $C(i_1, i_1; i_2, i_2) = \infty$
and $G(i_1, i_1; i_2, j_2) = G(i_1, j_1; i_2, i_2) = \infty$

Complexity

space complexity $O(n^4)$

- constant number of matrices (C,D,F, and G)
- each of them has $O(n^4)$ entries

time complexity $O(n^6)$

- each entry of matrix D requires constant time
- each entry of F,C, and G requires $O(n^2)$ time (minimize over all h_1, h_2)
- hence: $n^4 \cdot n^2 = n^6$