Sequence Alignment

Ulf Leser

This Lecture

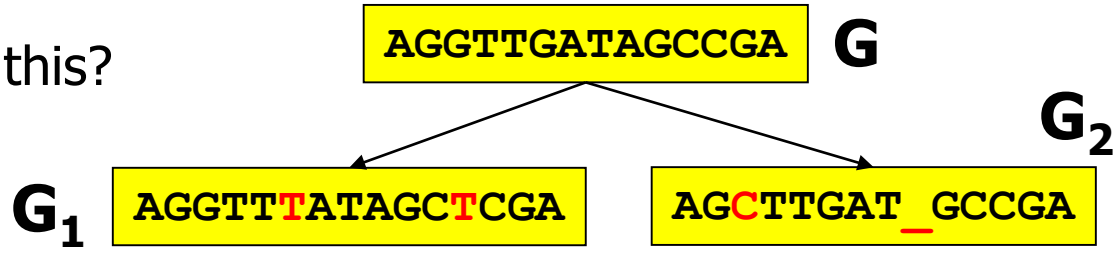
- Approximate String Matching
- Edit distance and alignment
- Computing global alignments
- Local alignment

Gene Function

- A fundamental principle of bioinformatics
  - The function of a protein depends on its physical structure
  - The physical structure depends on the protein sequence
  - The protein sequence depends on the gene sequence
  - If the sequence of two genes is only slightly different, so will be the protein sequence
  - If the sequence of two proteins is only slightly different, so will be their structure
  - If the structure of two proteins is only moderately different, they likely have the same (or at least share some) function
- Studying the sequence of genes allows the generation of hypotheses about their function

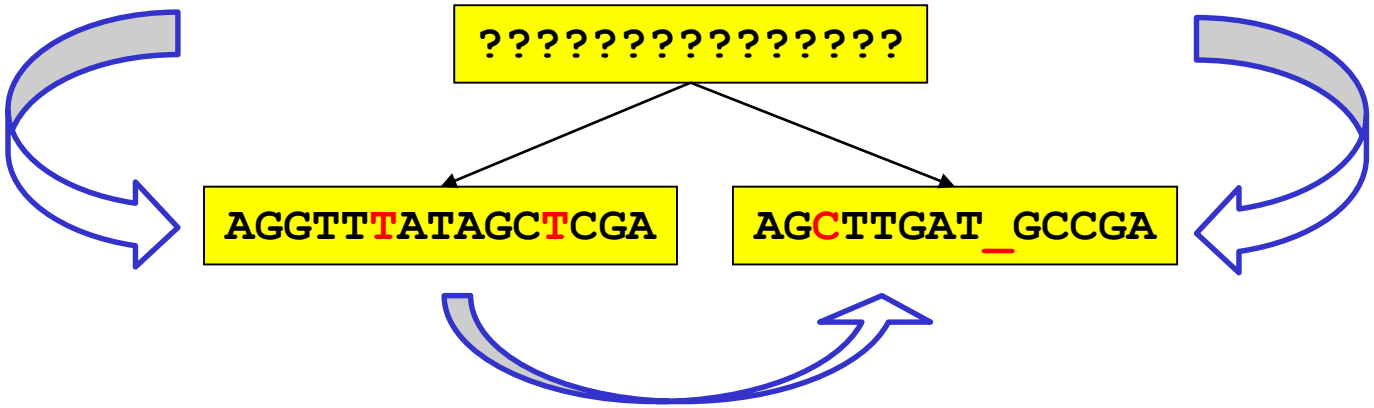
How Genes Evolve

- Evolution, sequences, and function
  - Any two species X_1, X_2 have a common ancestor A
  - Any gene G from A will undergo independent evolution in X_1 and X_2, leading to genes G_1 and G_2
  - The more similar G_1 and G_2 are, the more likely do they still have the same function (that of G)
  - For any two genes of non-trivial length, the chance that they have a very similar sequence by chance is extremely small
  - Corollary: If two genes G_1 and G_2 today are very similar, they most likely derive from the same ancestor and most likely have the same function
  - How can we quantify this?



Basic Evolutionary Events

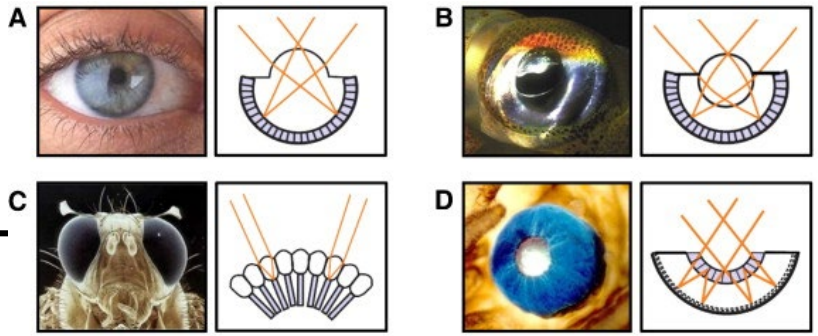
- The simplest model: Single bases can be replaced (R), inserted (I), or deleted (D) (or kept (M))
- Any changes must be explained by sequences of I, D, R
  - I.e., by singular evolutionary events accumulating over time
  - We call this an edit script
- Very simple yet quite powerful model
- One more simplification

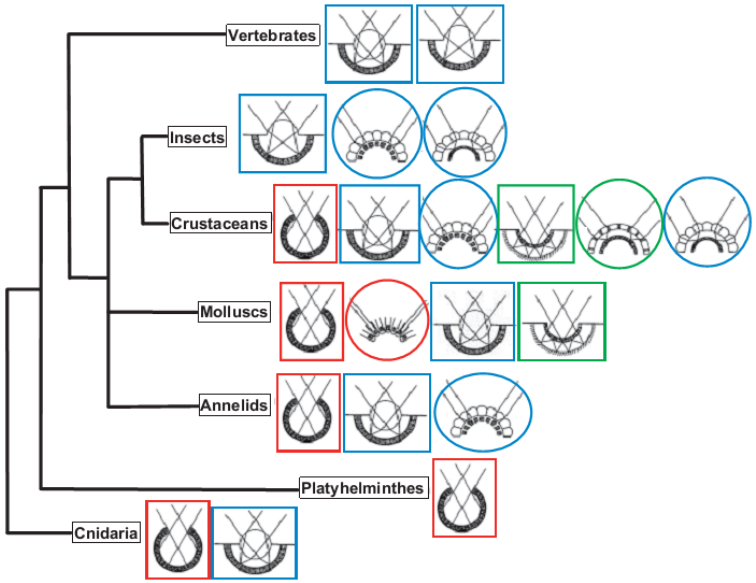


Example: Eyeless (ey)

- Family of genes identified first in Drosophila
- When activated in arbitrary cells, non-functional eyes start to grow at various places of the body
- ey is a "master gene" – controls a cascade of activations of other genes eventually leading to eye development
- Also inflicted with several other neural developments

Eyes





Red: Only shadow
Blue: Lenses etc.
Green: Mirrors

Oval: Compound eyes
Rectangle: Single chamber

Source: Treisman (2004).

- Eyes probably are an example of convergent evolution
- However, genes controlling eye development are highly conserved across a wide range of species

Homologues of “eyeless isoform D” (DM)

MFTLQPTPTAIGTVVPPWSAGTLIERLPSLEDMAHKDNVIAMRNLPCLGTAGGSGLG
GIAGKPSPTMEAVEASTASHPHSTSSYFATTYYHLTDDECHSGVNQLGGVFVGGRPL
PDSTRQKIVELAHSGARPCDISRILQVSNGCVSKILGRYYETGSIRPRAIGGSKPRVAT
AEVVSKISQYKRECPSIFAWEIRDRLLQENVCTNDNIPSVSSINRVLRNLAAQKEQQST
GSGSSSTSAGNSISAKVSVSIGGNVSNVASGSRGTLSSSTDLMQTATPLNSSESGGAS
NSGEGSEQEAIYEKLRLLNTQHAAGPGPLEPARAAPLVGQSPNHLGTRSSHPQLVHG
NHQALQQHQQQSWPPRHYSGSWYPTSLSEIPISSAPNIASVTAYASGPSLAHSLSPP
NDIESLASIGHQRNCPVATEDIHLKKELDGHQSDETGSGEGENSNGGASNIGNTEDD
QARLILKRKLQRNRTSFTNDQIDSLEKEFERTHYPDVFARERLAGKIGLPEARIQVWFS
NRRAKWRREEKLRNQRRTPNSTGASATSSSTSATASLTDSPNSLSACSSLLSGSAGG
PSVSTINGLSSPSTLSTNVNAPTLGAGIDSSESPTPIPHIRPSCTSDNDNGRQSEDCRR
VCSPCPLGVGGHQNTHHIQSNGHAQGHALVPAISPRLNFNSGSFGAMYSNMHHTAL
SMSDSYGAVTPIPSFNHSAVGPLAPPSPIPQQGDLTPSSLYPCHMTLRPPPMAPAHHH
IVPGDGGRPAGVGLGSGQSANLGASCSGSGYEVLSAYALPPPPMASSSAADSSFSAAS
SASANVTPHHTIAQESCPSPCSSASHFGVAHSSGFSSDPISPAVS…

- 250 most similar protein sequences in UniProt
  - Sequence identities all >50%,
  - All p-Values < 1E-50

This Lecture

- Approximate String Matching
- Edit distance and alignment
- Computing global alignments
- Local alignment

Edit Scripts and Edit Distances

- Definition
  - Let A, B $\in \Sigma^* = \Sigma \cup$ "_"
  - *An edit script e is a sequence of operations I, D, R, M*
  - *e is an edit script for A and B iff e(A)=B*
    - *Slightly underdetermined – which replacement? Which base to insert?*
  - *The length of an edit script is the number of I,D,R it contains*
  - *The edit distance between A and B is the length of the shortest edit script for A and B*
- Remarks
  - If we know e(A)=B, determining e' with e'(B)=A is trivial
  - The shortest edit script is not unique, but its length is
  - **MIMMMR** **IRMMMDI**
    A_TGTA **_ATGTA_**
    AGTGTC **AGTGT_C**

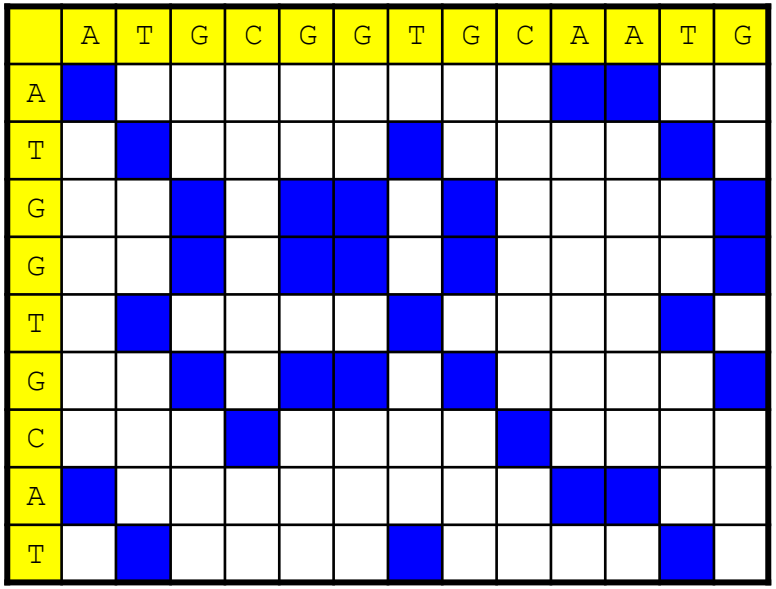
Alignment

- Edit scripts are intuitive from an evolutionary point-of-view, but somewhat clumsy from a computational point-of-view
- Definition
  - *A (global) alignment of strings A, B is an arrangement of A and B, enriched with „_" at arbitrary positions, under each other such that no column contains two „_"*
  - *The score of an alignment is the number of "_" plus the number of mismatching columns it contains*
  - *The alignment distance between A and B is the minimal score of any alignment of A and B*
- Edit distance and alignment distance are essentially identical
- Examples

A_TGT_A	**A_T_GTA**	**_AGAGAG**	**AGAGAG_**
AGTGTC_	**_AGTGTC**	**GAGAGA_**	**_GAGAGA**
Score: 3	**5**	**2**	**2**

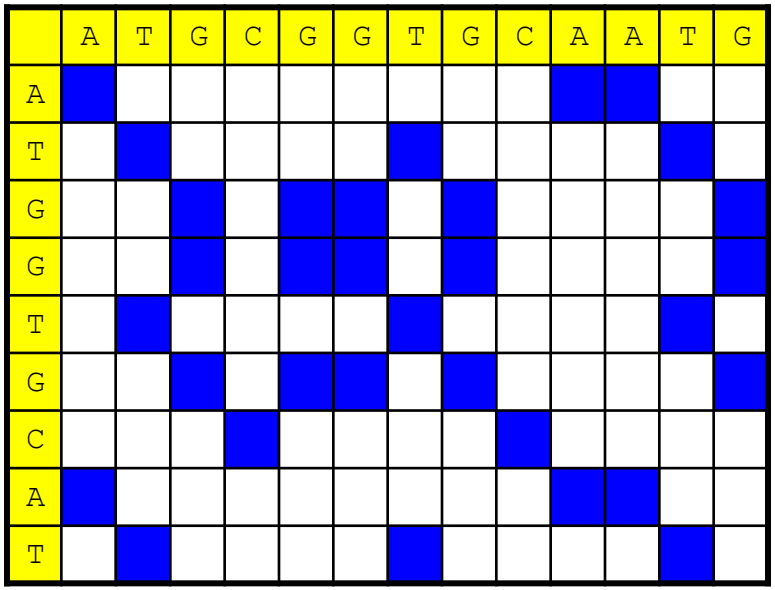
A Visual Approach: Dotplots

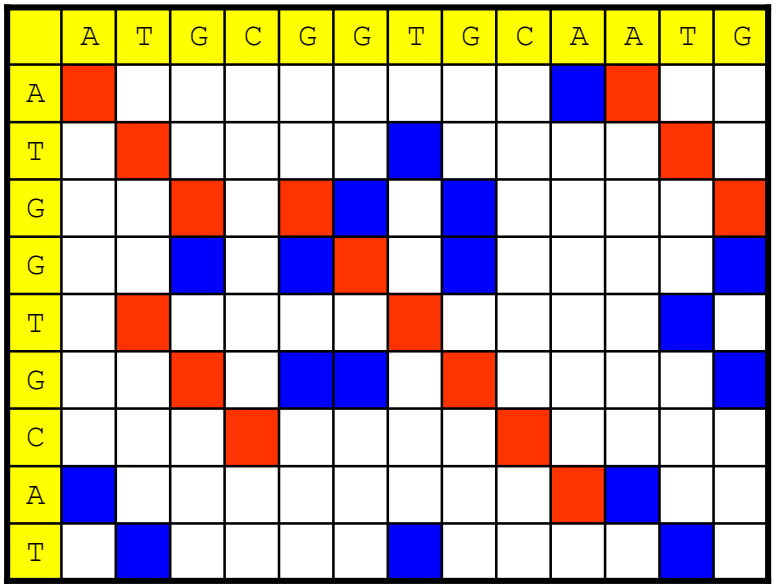
- *A dotplot of two strings A, B is a matrix M with*
  - *The i'th character in A is represented by the i'th column*
  - *The j'th character in B is represented by the j'th row*
  - *M[i,j]=1 (blue) iff A[i] = B[j]*



Dotplot and Identical Substrings

- How do identical substrings look like in a dotplot?



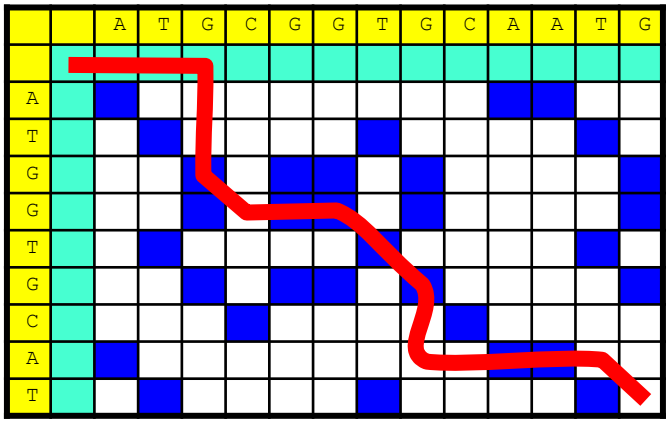


- Diagonals from up-left to down-right
  - Longest diagonal is the longest common substring

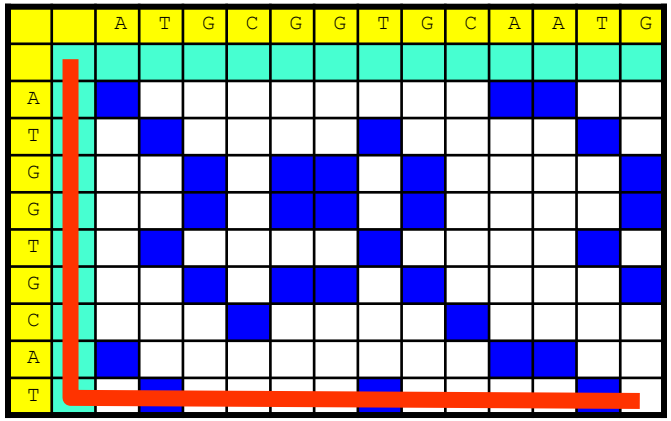
Alignments and Dotplots

- Every alignment of A, B can be uniquely mapped into a path through M
  - The path starts in the upper-left corner (coord: 0,0)
  - Go through the alignment column by column
  - Next column is "X,_" – move to the right
  - Next column is "_, X" – move down
  - Next column is "X, Y" – move right-down

```
ATG___CGGTG__CAATG
___ATGG__TGCA____T
```

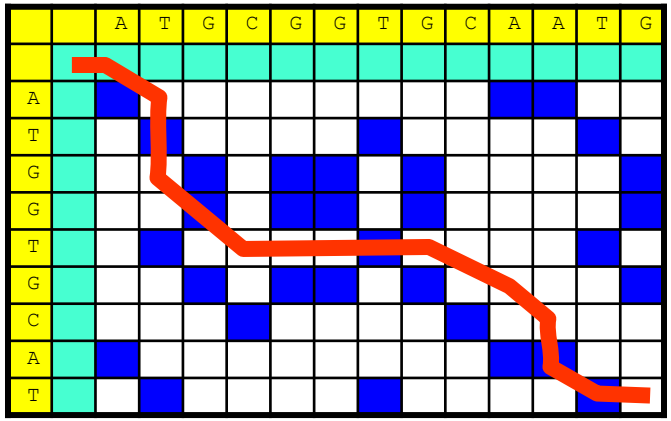



```
_____________ATGCGGTGCAATG
ATGGTGCCAT_________________
```

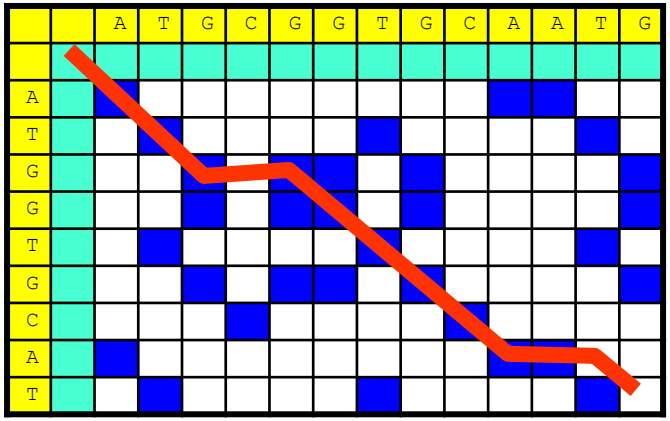


Examples

AT__GCGGTGCAA_TG
_ATGGT_____GCAT___



ATGCGGTGCAATG
ATG__GTGCA__T



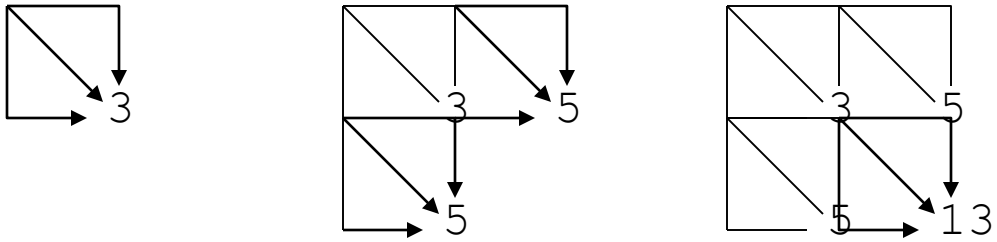
- Clearly, the number c(P) of 1's (blue cells) crossed in a diagonal step by a path P is the same as |P|-e(A,B)
- Finding the path that minimizes |P|-c(P) (or maximizes c(P)) solves the problem of computing the edit distance

This Lecture

- Approximate String Matching
- Edit distance and alignment
- Computing global alignments
- Local alignment

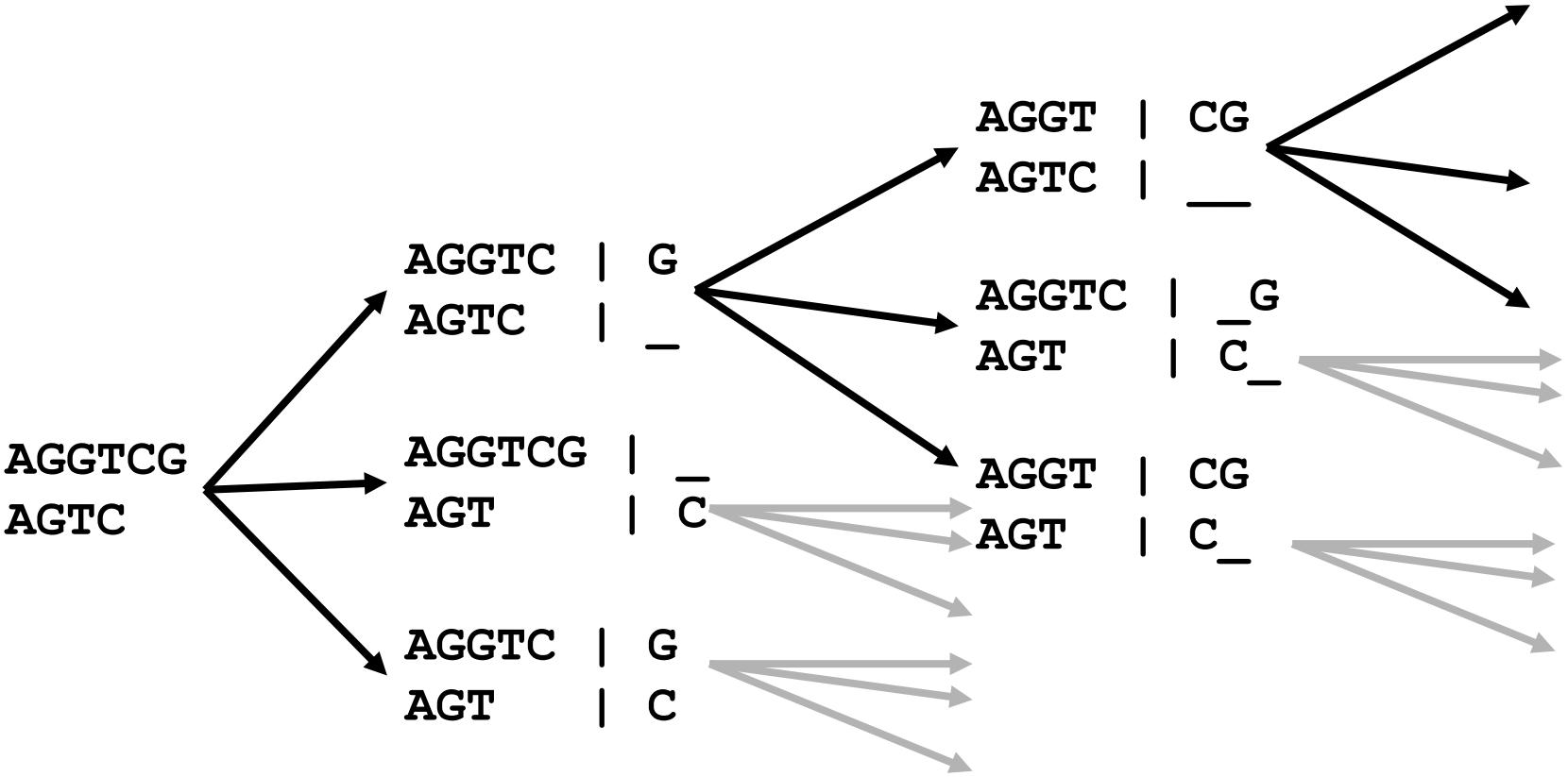
Algorithm

- How do we compute the edit distance of two strings?
- Naïve: Enumerate all paths, compute c(P) for each



- Bad news: There exist $>3^{\min(m,n)}$ paths
- Good news: We can compute e(A,B) with ~3*m*n operations

Enumerating all Paths Recursively



The naïve (recursive) Way

- Observation
  - *Let |A|=n, |B|=m*
  - *Let d(i,j)=e(A[…i], B[…j]) for 0≤i≤n and 0≤j≤m with d(i, 0)=i and d(0,j)=j*
  - *We can compute e(A,B) = d(n,m) recursively as follows*

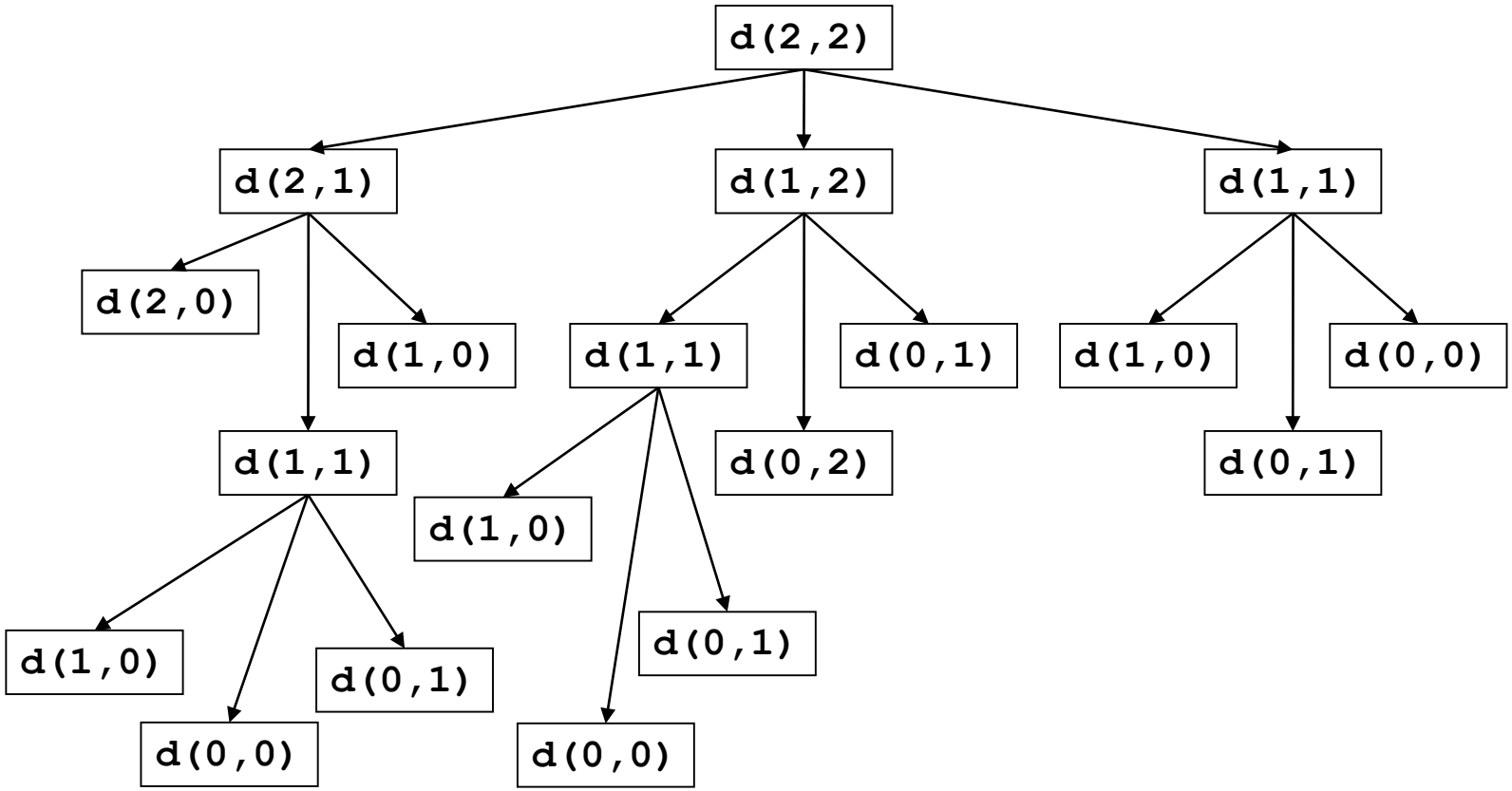
$$d(i,j) = \min \begin{cases} d(i,j-1)+1 \\ d(i-1,j)+1 \\ d(i-1,j-1)+t(i,j) \end{cases}$$

$$t(i,j) = \begin{cases} 1 : if \quad A[i] \neq B[j] \\ 0 : else \end{cases}$$

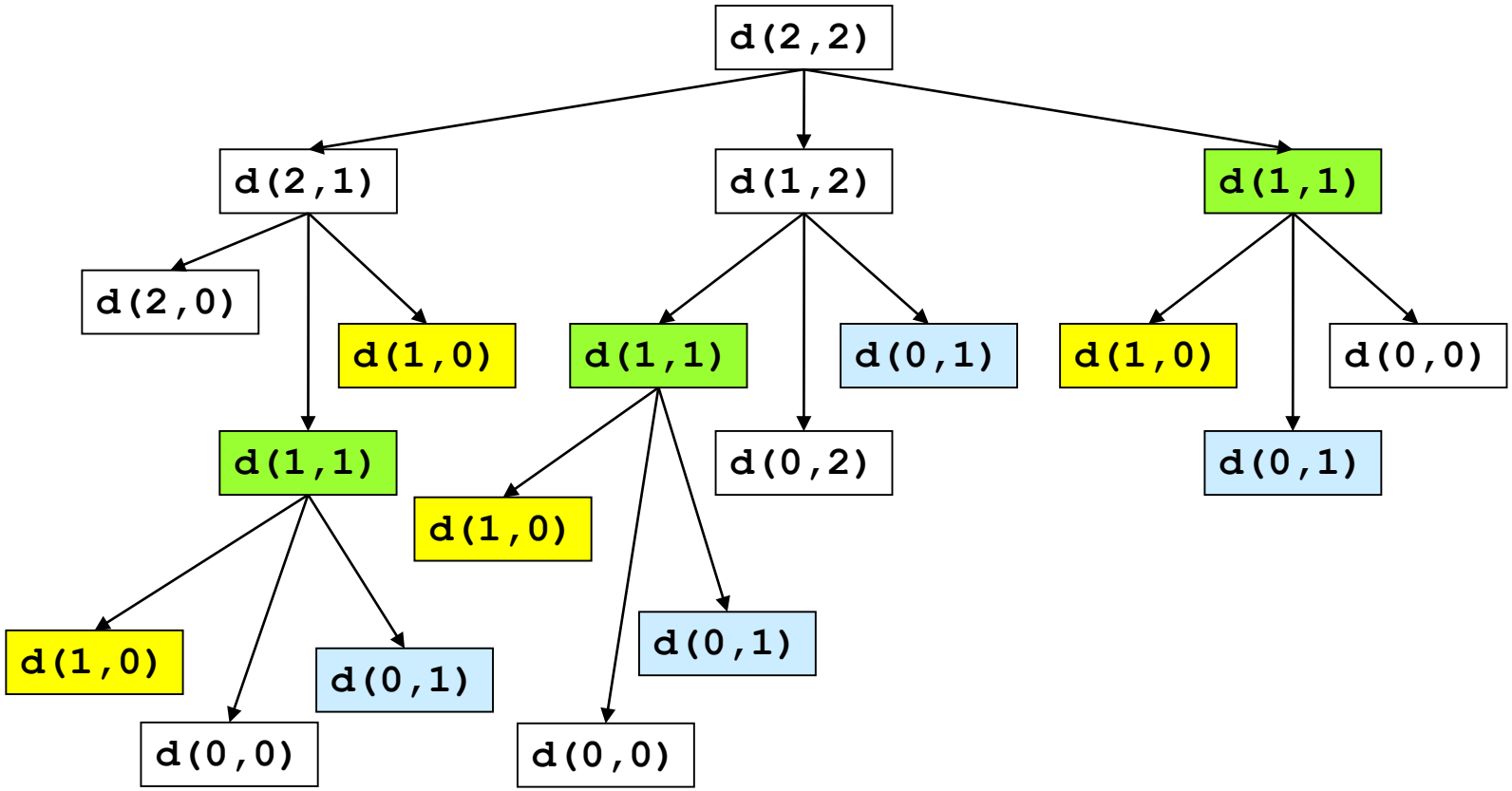
Algorithm

```
function d(i,j) {
        if (i = 0)              return j;
        else if (j = 0)         return i;
        else
                return min (  d(i,j-1) + 1,
                              d(i-1,j) + 1,
                              d(i-1,j-1) + t(A[i],B[j]));
}
function t(c1, c2) {
        if (c1 = c2)            return 0;
        else                    return 1;
}
```

What is Happening?



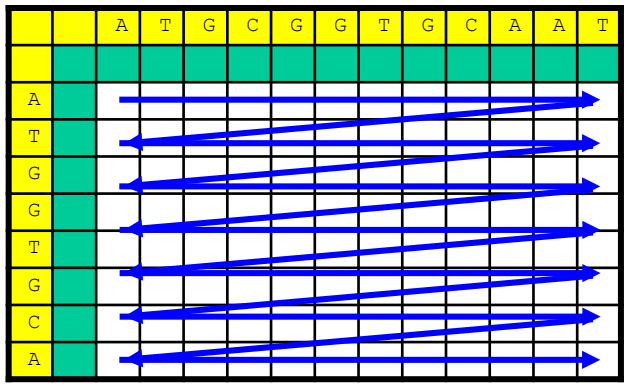
Much Redundant Computation

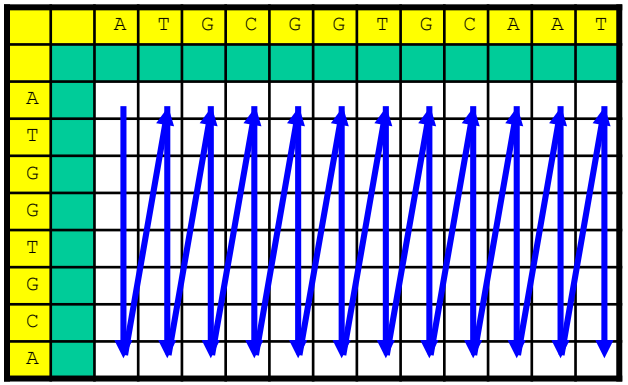


There are only ~n*m different parameter combinations

Dynamic Programming – Using a Table

- Instead of computing top-down (from n,m), we compute all different values for d(i,j) bottom-up
  - We store all values in a table
- We can immediately "compute" d(i,0) and d(0,j)
- Which values can we compute next?





Example

$$d(i,j) = \min\begin{Bmatrix} d(i,j-1)+1 \\ d(i-1,j)+1 \\ d(i-1,j-1)+t(i,j) \end{Bmatrix}$$

		A	T	G	C	G	G	T
	0	1	2	3	4	5	6	7
A	1							
T	2							
G	3							
G	4							

		A	T	G	C	G	G	T
	0	1	2	3	4	5	6	7
A	1	0						
T	2							
G	3							
G	4							

		A	T	G	C	G	G	T
	0	1	2	3	4	5	6	7
A	1	0	1	2	3	4	5	6
T	2							
G	3							
G	4							

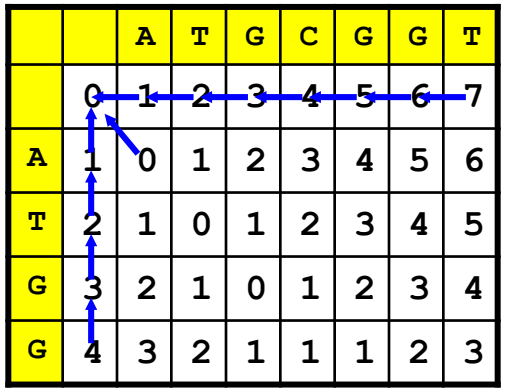
		A	T	G	C	G	G	T
	0	1	2	3	4	5	6	7
A	1	0	1	2	3	4	5	6
T	2	1	0	1	2	3	4	5
G	3							
G	4							

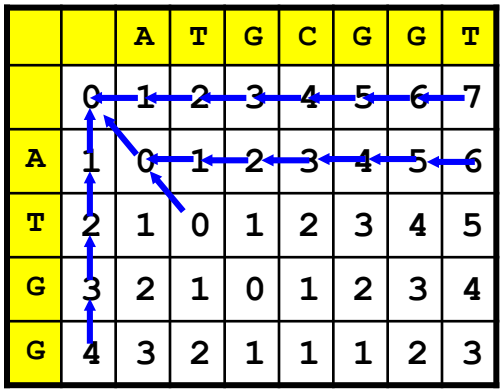
		A	T	G	C	G	G	T
	0	1	2	3	4	5	6	7
A	1	0	1	2	3	4	5	6
T	2	1	0	1	2	3	4	5
G	3	2	1	0	1	2	3	4
G	4							

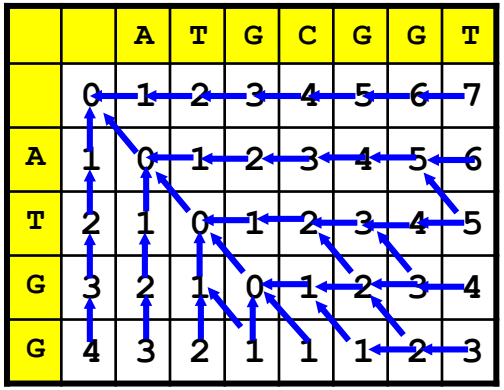
		A	T	G	C	G	G	T
	0	1	2	3	4	5	6	7
A	1	0	1	2	3	4	5	6
T	2	1	0	1	2	3	4	5
G	3	2	1	0	1	2	3	4
G	4	3	2	1	1	1	2	3

Finding the (an) optimal Alignment(s)

- Traceback
  - We find the path from back to front
  - Start at cell (n,m)
  - See which cells were used to compute d(n,m)
  - Walk any of these – finds one optimal path
  - Walking all means finding all optimal paths
- Alternative: Store pointers while filling the table







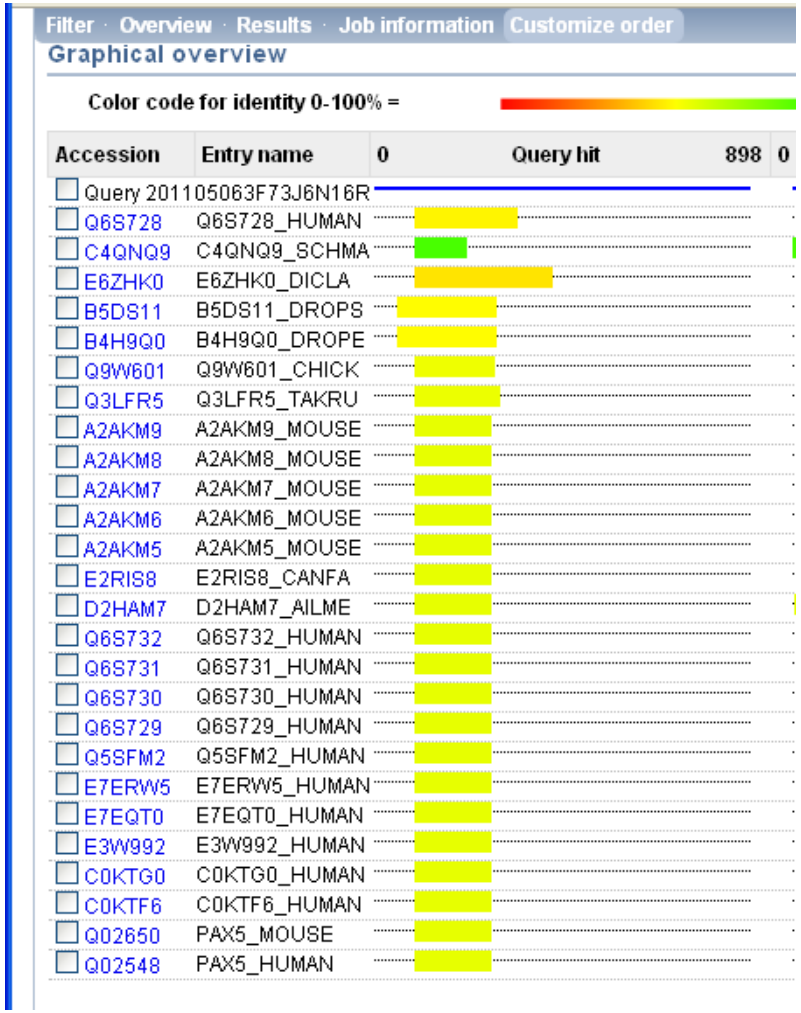
Complexity

- Building the table
  - For every d(i,j), we need to access three other cells and make some (constantly many) additions and comparisons
  - There are (m+1)*(n+1) cells
  - Thus: ~3*m*n=O(m*n) operations
- Finding one optimal alignment
  - We must walk from (n,m) to (1,1)
  - Such a path can have at most length m+n
    - We cannot go wrong!
  - Together: approximately m+n operations
- Together: O(m*n) (for m*n > m+n)

This Lecture

- Approximate String Matching
- Edit distance and alignment
- Computing global alignments
- Local alignment

Eyeless Again – a Closer Look



- The similar regions in the different homologues are not distributed randomly
- Actually, a single stretch of 128 AA, the PAX domain, is virtually unchanged in all homologues
  - Controls binding to DNA and hence regulatory effects
- Typical: Only some parts of a gene are conserved, and these carry function

Example

```
ACCCTATCGATAGCTAGAAGCTCGAAAATACCGACCAGTAT
AGGAGTCGATAATACATATAAGAGATAGAATATATTGATG
```

Zufall?

```
ACCCTATCTATA--GC-TAGAAGCTCGATAATACCGACCAGTAT-
|      ||  ||     |  ||  ||     ||  |   |    |   ||
A-GGAGTCGATCATACATATAAG-A-GATAGAATATA-TTG-ACG
```

Kein Zufall!

```
ACCCTATCGATAGCTAGAAGCTCGAAAATACCGACCAGTAT
                     |||| |||||
                AGGAGTCGATAATACATATAAGAGATAGAATATATTGATG
```

Distance or Similarity

- Until now, we computed a global distance
  - The higher e(A,B), the less similar are A and B
  - The longer A and B, the higher their distance in general
  - Different lengths are punished: $e(A,B) \geq ||A|-|B||$
- Often, we want a local similarity instead
  - If we don't compare two entities, but two strings presumably containing each one entity
  - If we have a sequence and don't know exactly where the genes are
  - If a function is associated to a motif in a protein
- We need to search for substrings $A' \in A$, $B' \in B$ which are very similar to each other
  - A' and B' should have a certain length to be interesting
  - e(A',B') does not help – optimal distance is 0 for A'=B'="'"

Sequence Similarity

- *A scoring function is a function s: $\Sigma' x \Sigma' \rightarrow$ Integer*
  - *We also call s a substitution matrix*
- *The ungapped similarity sim' of A, B wrt. s with |A|=|B|=n is defined as*

$$sim'(A,B) = \sum_{i=1}^{n} s\left(A[i], B[i]\right)$$

- *The similarity sim of A, B (wrt. s) is the highest ungapped similarity score over all alignments of A and B*
  - Higher = better; maximal similarity is n*max(s)
- We are not yet there: This still is a global similarity score

Example

$\Sigma' = \{A,C,G,T,\_\}$

	A	C	G	T	_
A	4	-2	-2	-1	-3
C		4	-1	-2	-3
G			4	-2	-3
T				4	-3

```
AC_GTC
AGGT_C
```
= -1

```
ACGTC
AGGTC
```
= 15

```
A_CGTC
AG_GTC
```
= 10

Computing sim(A, B)

- Same ideas as for edit distance
- But: We want a high similarity, not a low distance
- Thus, we can compute sim(|A|,|B|) with

$$sim(i,0) = \sum_{k=1..i} s(A[k], \_) \qquad sim(0,j) = \sum_{k=1..j} s(\_, B[k])$$

$$sim(i,j) = max \begin{cases} sim(i,j-1) + s(\_, B[j]) \\ sim(i-1,j) + s(A[i], \_) \\ sim(i-1,j-1) + s(A[i], B[j]) \end{cases}$$

Example

	A	G	T	C
A	4	-1	-1	-1
G		4	-1	-1
T			4	-1
C				4
-	-3	-3	-3	-3

Edit Distance

		A	G	G	T	C
	0	1	2	3	4	5
A	1	0	1	2	3	4
G	2	1	0	1	2	3
T	3	2	1	1	1	2
C	4	3	2	2	2	1
C	5	4	3	3	3	2

Similarity

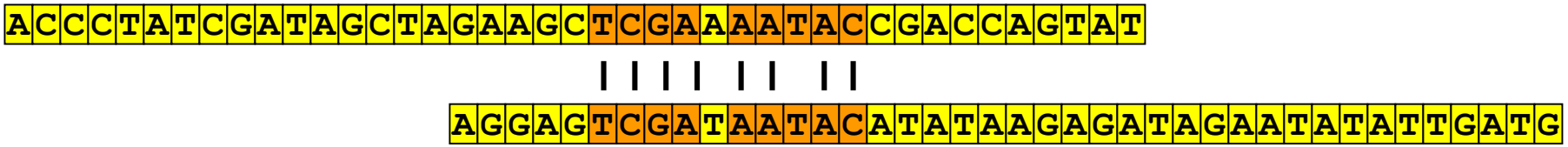
		A	G	G	T	C
	0	-3	-6	-9	-12	-15
A	-3	4	1	-2	-5	-8
G	-6	1	8	5		
T	-9					
C	-12					
C	-15					

Lokal Similarity = Local Alignment

- Definition
  - *The local similarity score sim* of A, B is defined as*

$$sim^*(A,B) = \max_{\forall A' \, substringOf \; A, B' \, substringOf \; B} \left(sim(A', B') \right)$$

- Remark
  - Inequality in length of A and B does not matter any more
  - Sounds terribly complex, but there is a neat trick



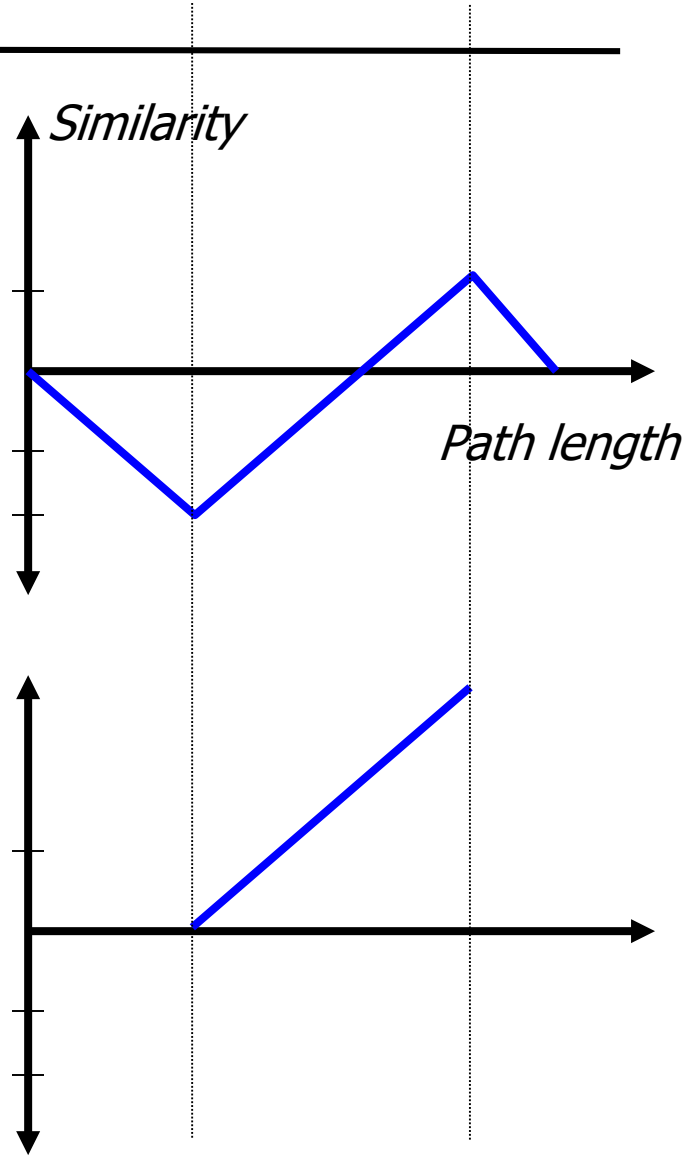
Example

Match: +1

I/R/D: -1

		A	T	G	T	G	G
	0	-1	-2	-3	-4	-5	-6
G				-1			
T					0		
G						1	
A							0

		A	T	G	T	G	G
	0	0	0				
G				1			
T					2		
G						3	
A							2



Smith-Waterman Algorithm

- Smith, Waterman: „Identification of common molecular subsequences“, J. Mol. Bio 147, 1981
- Idea
  - Note: Local paths need not span the entire strings
  - Look at a single path
  - A series of matches (positive values for scoring function s) creates a series of increasing similarity values
  - Any step with s<0 lowers the score
  - Whenever the cumulative score falls below 0, we drop this prefix
  - Instead of carrying on, we conceptually start a new local path
  - To this end, we simply set true_score=max(0,score)
  - The highest value in the matrix is the end of the best local path

Computation

- The same ideas as before
- We compute sim*(A,B) with
  - Assume ∀X: s(X,_)<0 and s(_,X)<0

$$sim(i,0) = \sum_{k=1..i} s(A[k],\_) \qquad sim(0,j) = \sum_{k=1..j} s(\_,B[k])$$

$$sim(i,j) = max \begin{cases} sim(i,j-1) + s(\_,B[j]) \\ sim(i-1,j) + s(A[i],\_) \\ sim(i-1,j-1) + s(A[i],B[j]) \\ 0 \end{cases}$$

Example

Match: +1

I/R/D: -1

		A	T	G	T	C	G
	0	-1	-2	-3	-4	-5	-6
A	-1	1	0	-1	-2	-3	-4
T	-2	0	2	1	0	-1	-2
G	-3	-1	1	3	2	1	0

```
ATGTCG
ATG___

ATGTCG
AT___G

ATGTCG
A__T_G
```

		A	T	G	T	C	G
	0	0	0	0	0	0	0
A	0	1	0	0	0	0	0
T	0	0	2	1	1	0	0
G	0	0	1	3	2	1	1

```
ATGTCG
ATG___
```

Local versus global Alignment

- Global Alignment
  - Comparison of two entire sequences
  - Use when you think the entire sequences are related
  - Interest: The differences; assumption: Relatedness
  - Example: Proteins of the same family
- Local Alignment
  - Compare uncharacterized sequences
  - Use when comparing "randomly sampled" sequences
  - Interest: Similar regions; assumptions: None
  - Often a first step before global alignment
  - Example: Find similar genes in other species genomes

Beware: Not all Events are Equal

Further Reading

- Everywhere
- Relaxed: Christianini & Hahn, Chapter 3
- Step by step: Waack, Chapter 9