Lecture 10

Longest common substring problem

Algorithm	Preprocessing Time	Searching Time	Space Required
Naïve string search algorithm	None	O(mn)	None

Rabin–Karp string search algorithm	O(m)	O(n+m)
		O((n-m)m)
			Constant

Knuth–Morris–Pratt algorithm	O(m)	O(n)	O(m)

Length of a string: ***n***
Length of the pattern: ***m***
Cardinality of character set: ***k***

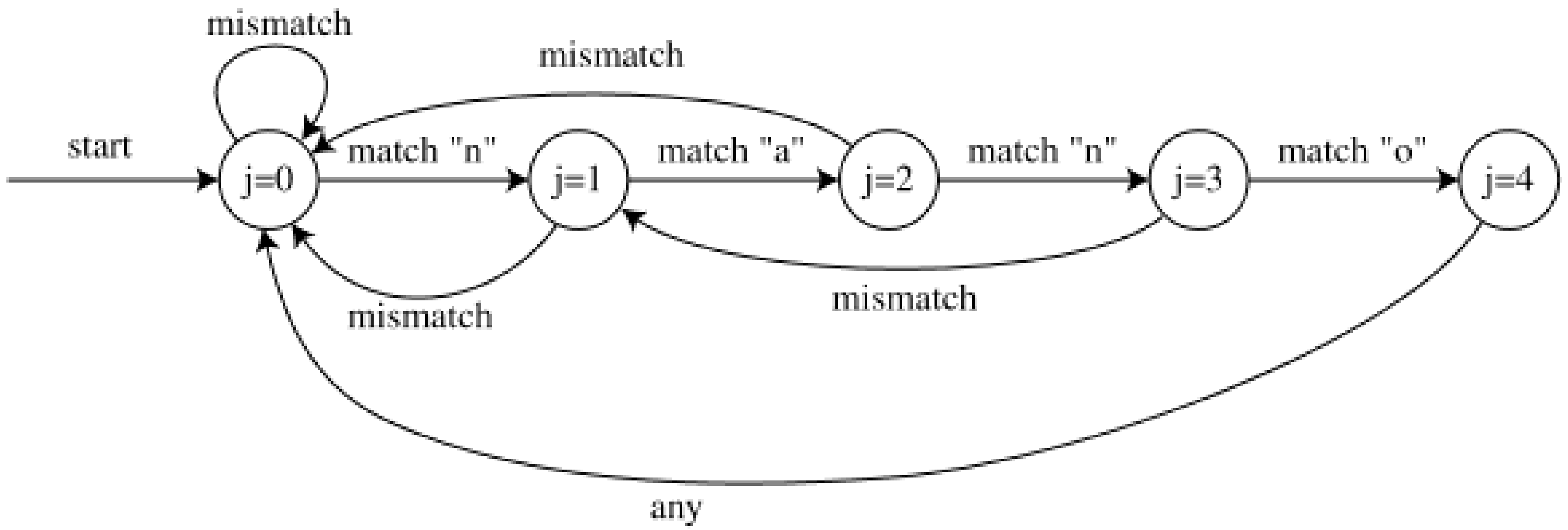
Longest common substring problem

- Naïve string search algorithm

```
for a character in txt
    for every character in pat
        break the search if txt character does not match pat character.
    If all the character of pat matches then output a match.
```

Longest common substring problem

- Knuth–Morris–Pratt (KMP) algorithm



Pattern: *nano*
Text: *banananobano*

Workout
Implement

A Rabin-Karp example

- Given T = 31415926535 and P = 26
- We choose q = 11
- P mod q = 26 mod 11 = 4

Complexity?

3	**1**	4	1	5	9	2	6	5	3	5

31 mod 11 = 9 not equal to 4

3	**1**	**4**	1	5	9	2	6	5	3	5

14 mod 11 = 3 not equal to 4

3	1	**4**	**1**	5	9	2	6	5	3	5

41 mod 11 = 8 not equal to 4

Applications

- Alpha hemoglobin and beta hemoglobin are subunits that make up a protein called hemoglobin in red blood cells. Notice the similarities between the two sequences, which probably signify functional similarity.

- Many distantly related proteins have domains that are similar to each other, such as the DNA binding domain or cation binding domain. To find regions of high similarity within multiple sequences of proteins, local alignment must be performed. The local alignment of sequences may provide information of similar functional domains present among distantly related proteins.

Homework

- Rabin–Karp string search algorithm
  - Hash function

Implement for genomics data.

Input: A gene sequence with thousands of coding regions which will code for proteins.

Problem statement: Match them against known proteins from UniProt.

```
/* pat ← pattern; M ← length of pattern
   txt ← text; N ← length of text/string
   q ← A prime number
*/
```

Steps:

```
1. Create a hash index for each possible pat ( (h*d)%q ). // h → pow(d, M-1)%q
2. Calculate the hash value of pat and first window of txt (p=(d*p + pat[i])%q; t=(d*t+txt[i])%q)
3. Slide the pattern over txt one by one
        Check the hash values of current window of txt and pat. If the hash values match
        then only check for characters one by one
                If all the character of pat matches then output a match.

        Calculate hash value for next window of txt: Remove leading digit, add trailing digit
```

Multiple alignment PSSM column

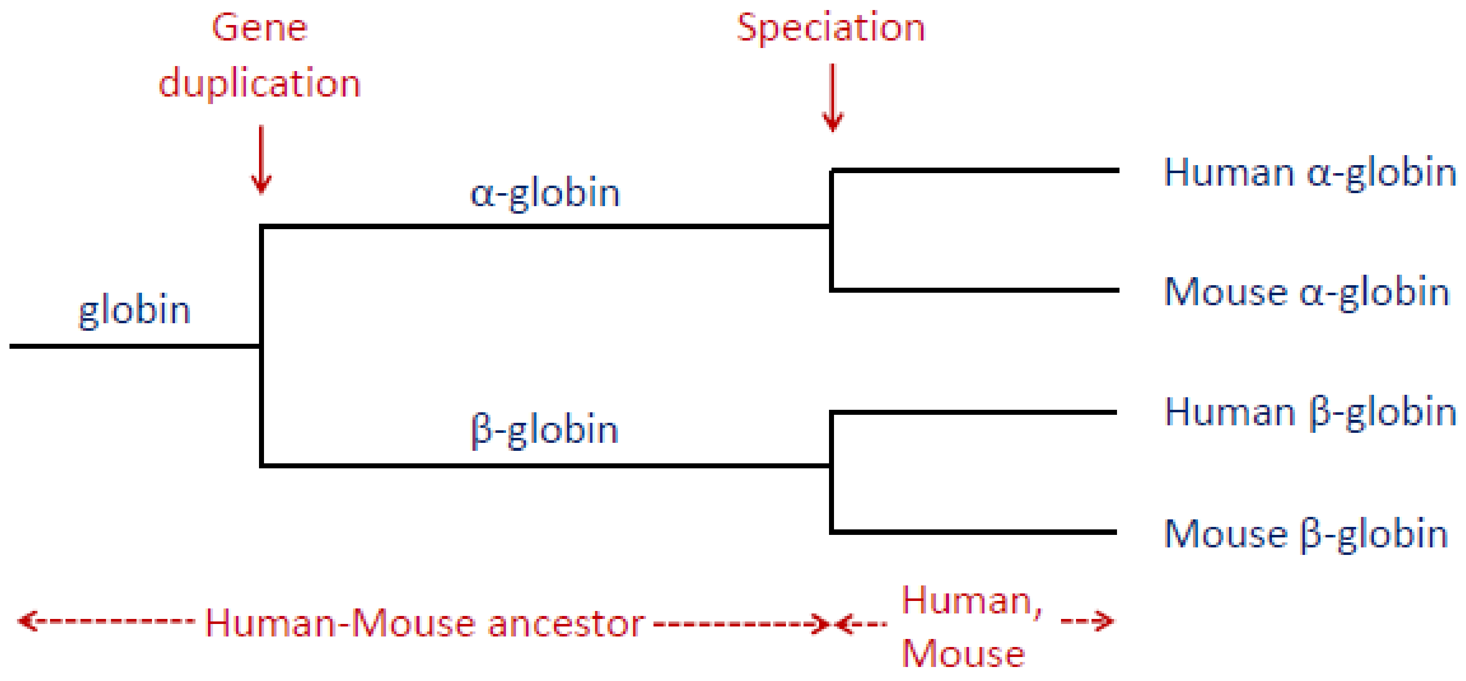
```
             140       150
RbcR    MLDSNSVDLVLMGV
LysR    WLSAQRHDLGLTET
CysB    AVSKGNADFAIATE
IlvY    KVVTGEADLAIAGK
IrgB    SDEVFEPDLIIWIE
GltC    AVRNRDIDLALLGP
OxyR    QLDSGKLDCVILAL
MleR    LLLNEEIDSSLLGS
MetR    ALQQGELDLVMTSD
AmpR    DPAAEGLDYTIRYG
TrpI    DPRRPGLDAMLWFA
NodD    RLRSGDLDFLILPD
NahR    ALQNGTVDLAVGLL
LeuO    QLRYQETEFVISYE
SyrM    LLEQGEIDVVVGQM
CatR    ALKSGRIDIAFGRI
CatM    ALKQGKIDLGFGRL
AntO    QLSQHKLEMIISDC
Svir    ELCQTNNCIVISAR
```

A	0
C	6
D	83
E	11
F	0
G	0
H	0
I	0
K	0
L	0
M	0
N	0
P	0
Q	0
R	0
S	0
T	0
V	0
W	0
Y	0

Position-based Sequence Weights

- Use to
  - reduce redundancy
  - emphasize diversity
- Based on
  - distance between a sequence and an ancestral or generalized sequence
  - weights on the diversity observed at each position in the alignment
- Application is in
  - MSA
  - Sequence searching

Digression: Orthology and Paralogy

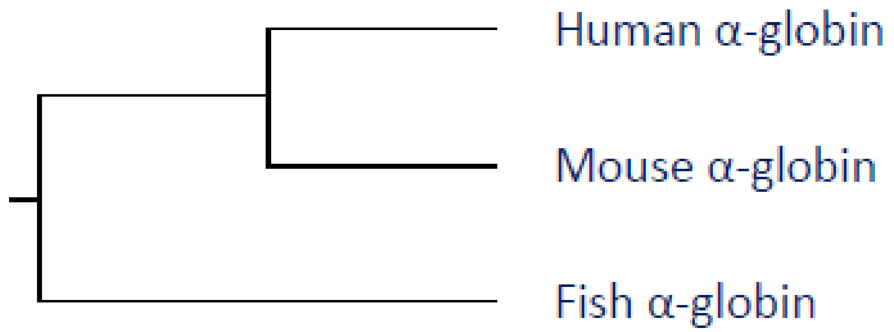


Homology: Two genes or proteins are *homologous* if they share a common ancestor.

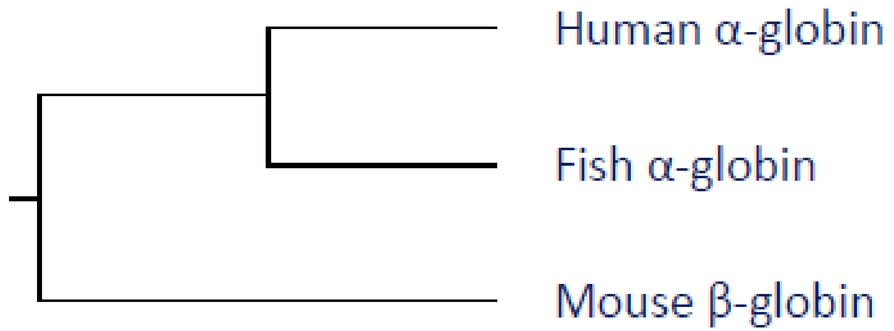
Orthology: Two genes or proteins are *orthologous* if they diverged by speciation.

Paralogy: Two genes or proteins are *paralogous* if they diverged by gene duplication.

Sequence Trees and Phylogenetic Trees



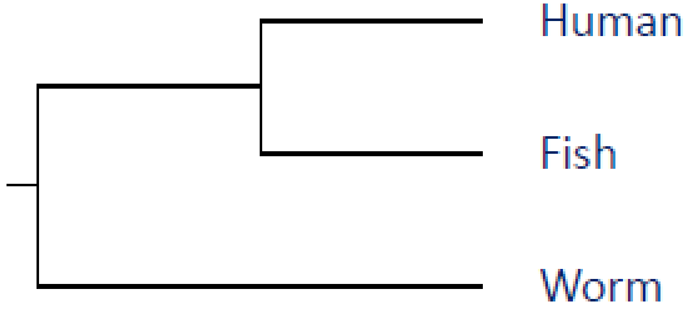
For orthologous genes or proteins, a tree reflecting sequence relationships should be congruent with the *phylogentic tree* of species relationships.



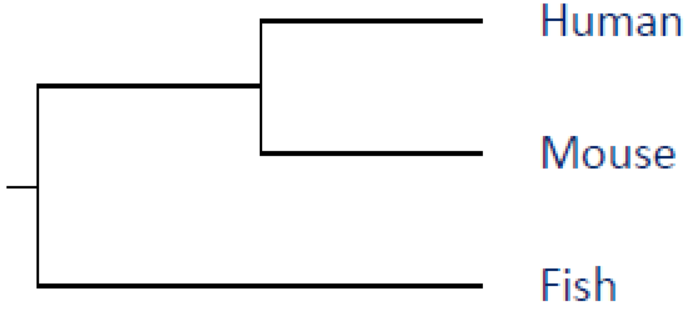
For paralogous genes or proteins, a tree reflecting sequence relationships may be incongruent with the phylogentic tree of species relationships.

Over the course of evolution, it is possible that in a particular protein family different paralogs are lost in different species. In that case there may be no set of orthologs for that family from which a valid phylogenetic tree may be reconstructed.

Weights Depend on a Set of Sequences



Given orthologous genes or proteins from these three organisms, the fish sequence should be downweighted.



Given orthologous genes or proteins from these three organisms, the fish sequence should be upweighted.

In other words, a weight is never *intrinsic* to a sequence. It is associated with a sequence only in the context of a set of other sequences.

Henikoff weights

Central idea:

Averaged over multiple-alignment columns, a sequence this is similar to others will tend to have many letters in common with those sequences.

Method:

i) For each column, divide a total weight of 1 evenly among the letter types that occur at that position, and then divide the weight assigned to each letter type evenly among the sequences that have that letter.

ii) For each sequence, sum its weights from all positions, and normalize.

Example:

Sequences	Calculation	Weight
G C G T T A G C	$\frac{1}{4}+\frac{1}{3}+\frac{1}{3}+\frac{1}{4}+\frac{1}{4}+\frac{1}{3}+\frac{1}{4}+\frac{1}{2} = 2\frac{1}{2}$	0.31250
G A G T T G G A	$\frac{1}{4}+\frac{1}{3}+\frac{1}{3}+\frac{1}{4}+\frac{1}{4}+\frac{1}{3}+\frac{1}{4}+\frac{1}{4} = 2\frac{1}{4}$	0.28125
C G G A C T A A	$\frac{1}{2}+\frac{1}{3}+\frac{1}{3}+\frac{1}{2}+\frac{1}{2}+\frac{1}{3}+\frac{1}{2}+\frac{1}{4} = 3\frac{1}{4}$	0.40625

Henikoff, S. & Henikoff, J.G. (1994) "Position-based sequence weights." *J. Mol. Biol.* **243**:574-578.

GYVGS
GFDGF
GYDGF
GYQGG

Position-based sequence weights for the alignment in Table 2A

A. Position-based residue weights

Residue	Position 1	2	3	4	5
G	1/(1*4)			1/(1*4)	1/(3*1)
Y		1/(2*3)			
F		1/(2*1)			1/(3*2)
V			1/(3*1)		
D			1/(3*2)		
Q			1/(3*1)		
S					1/(3*1)

B. Position-based sequence weights

Sequence	Position 1	2	3	4	5	Total	Normalized
GYVGS	1/(1*4)	1/(2*3)	1/(3*1)	1/(1*4)	1/(3*1)	4/3	0.267
GFDGF	1/(1*4)	1/(2*1)	1/(3*2)	1/(1*4)	1/(3*2)	4/3	0.267
GYDGF	1/(1*4)	1/(2*3)	1/(3*2)	1/(1*4)	1/(3*2)	3/3	0.200
GYQGG	1/(1*4)	1/(2*3)	1/(3*1)	1/(1*4)	1/(3*1)	4/3	0.267
Total	1	1	1	1	1	5	1.001

Each residue in each position is assigned a weight equal to $1/(r*s)$ where r is the number of different residues in the position and s is the number of times the particular residue appears in the position. The position-based weights are then added for each position in each sequence.

Advantages:

Very fast and simple.

Independent of sequence order.

Uses all information.

Disadvantages:

Ad hoc: no objective function to optimize.

Exact duplication of a sequence does *not* halve its weight. *Why?*

Work out

12AS_A:	AEKAVQVKVKAL
11AS_A:	AEKAVQVKVKAL
1IND_H:	PEKRLEWVATTL
1EZG_A:	NSQHCVKANTCT

Compute the Henikoff weight for the above sequences.

Compute a position specific scoring matrix.

Workout

- Write down the steps to compute the Henikoff weights of a number of sequences taken as input.