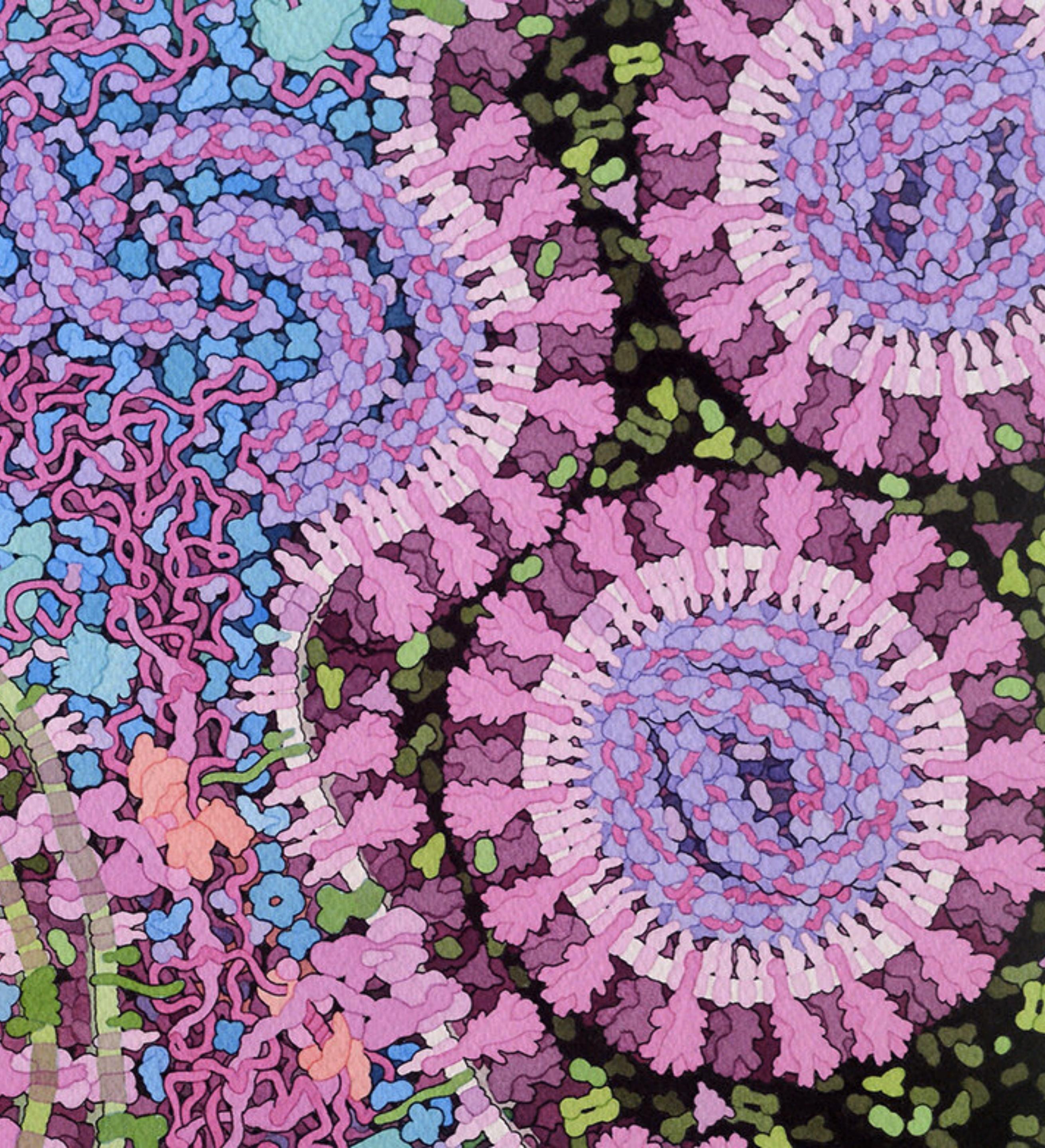


بسم الله الرحمن الرحيم

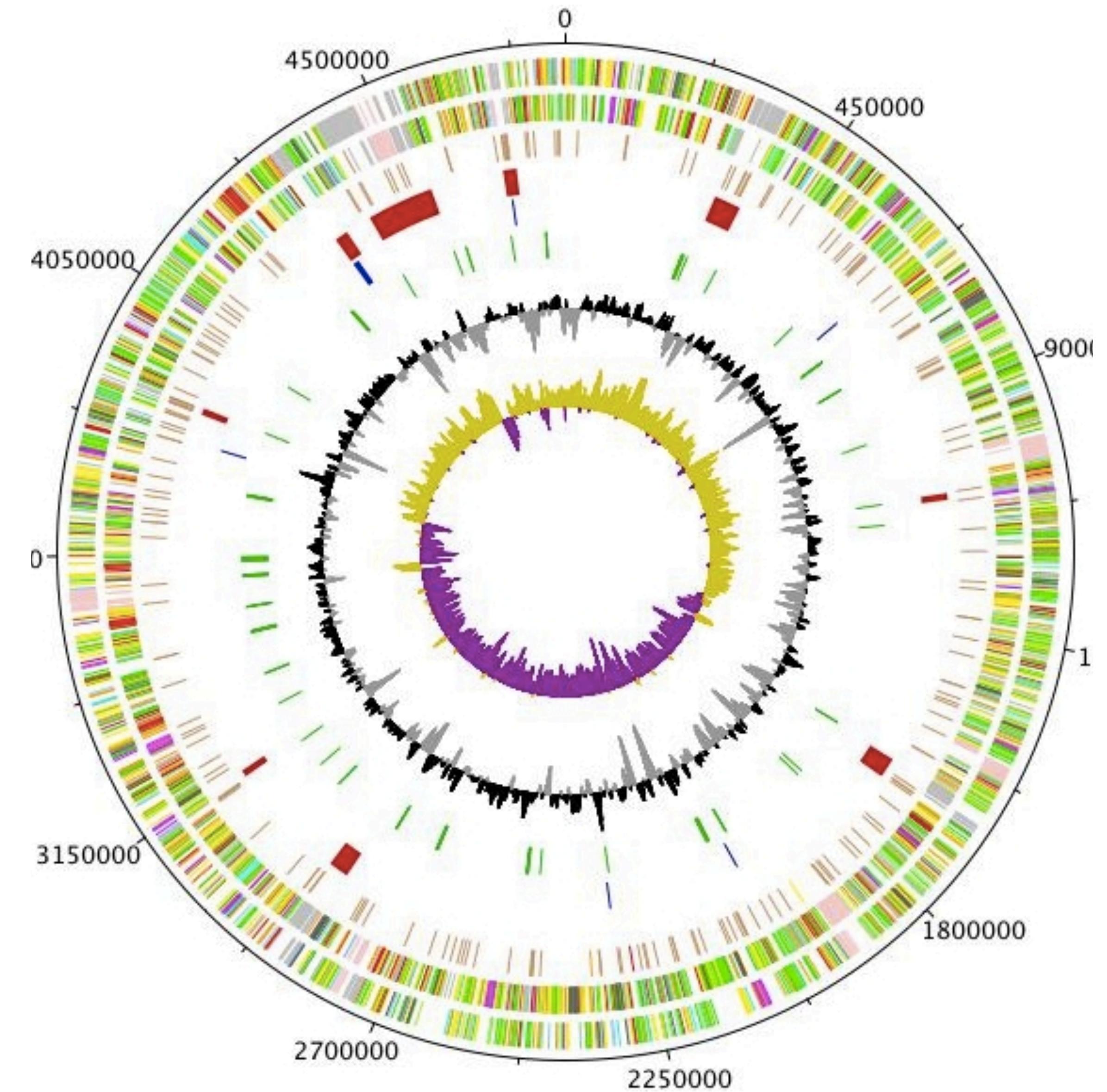
# ڙنو ميڪ محاسباتي

جلسه ۲ : هم ترازي توالي

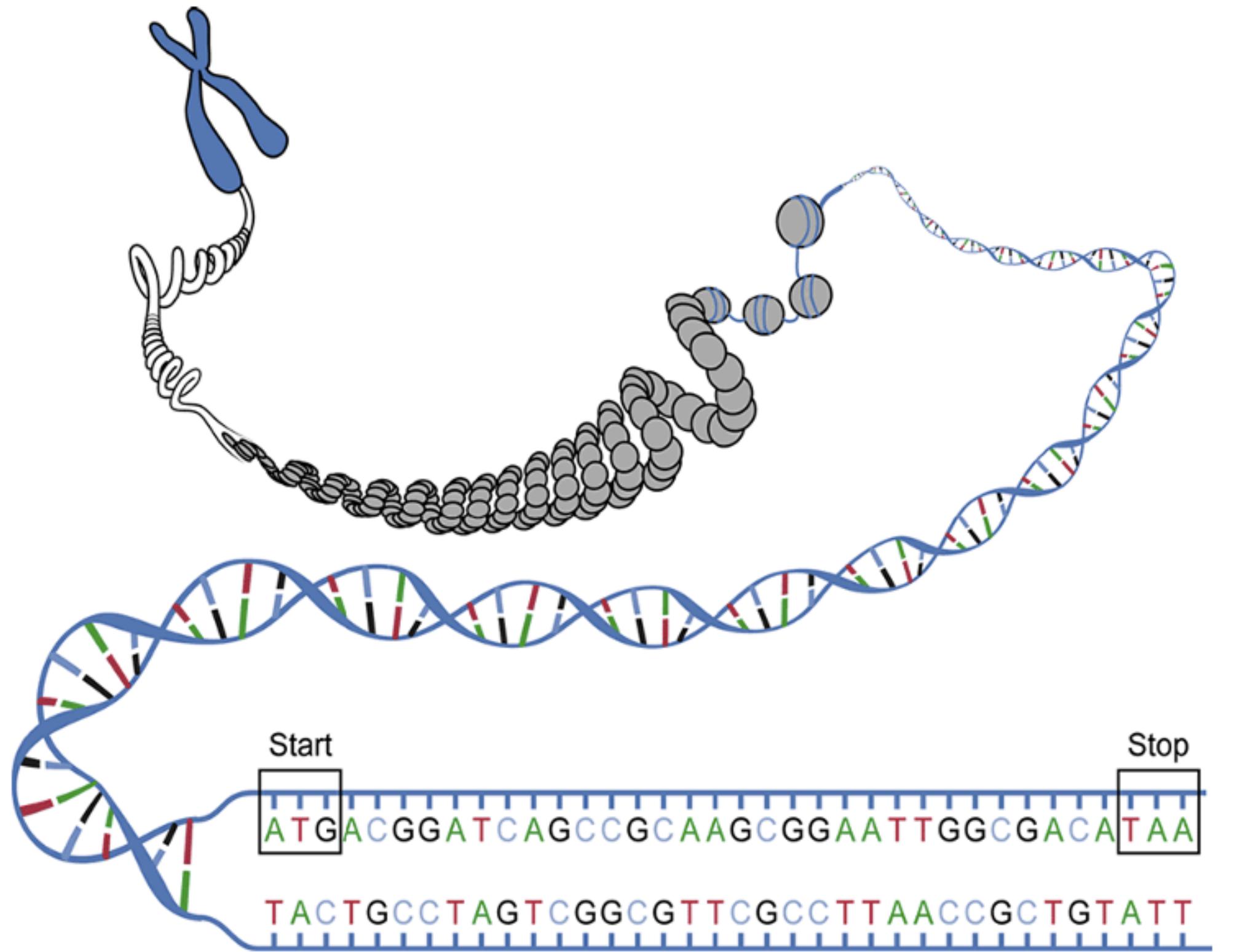
ترم پايزز ۱۴۰۱-۱۴۰۰



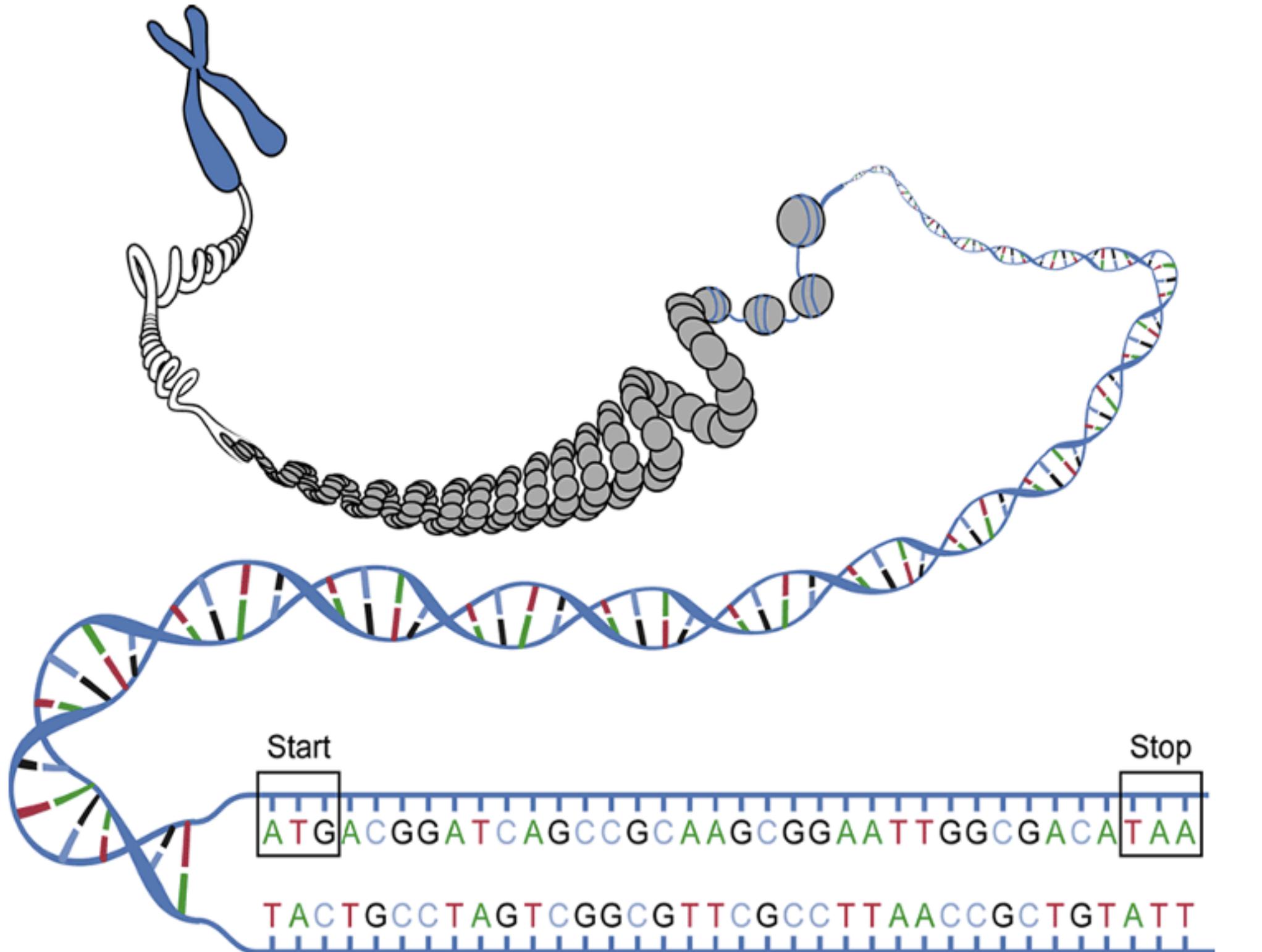
# رشته‌ها



# DNA رشته‌های

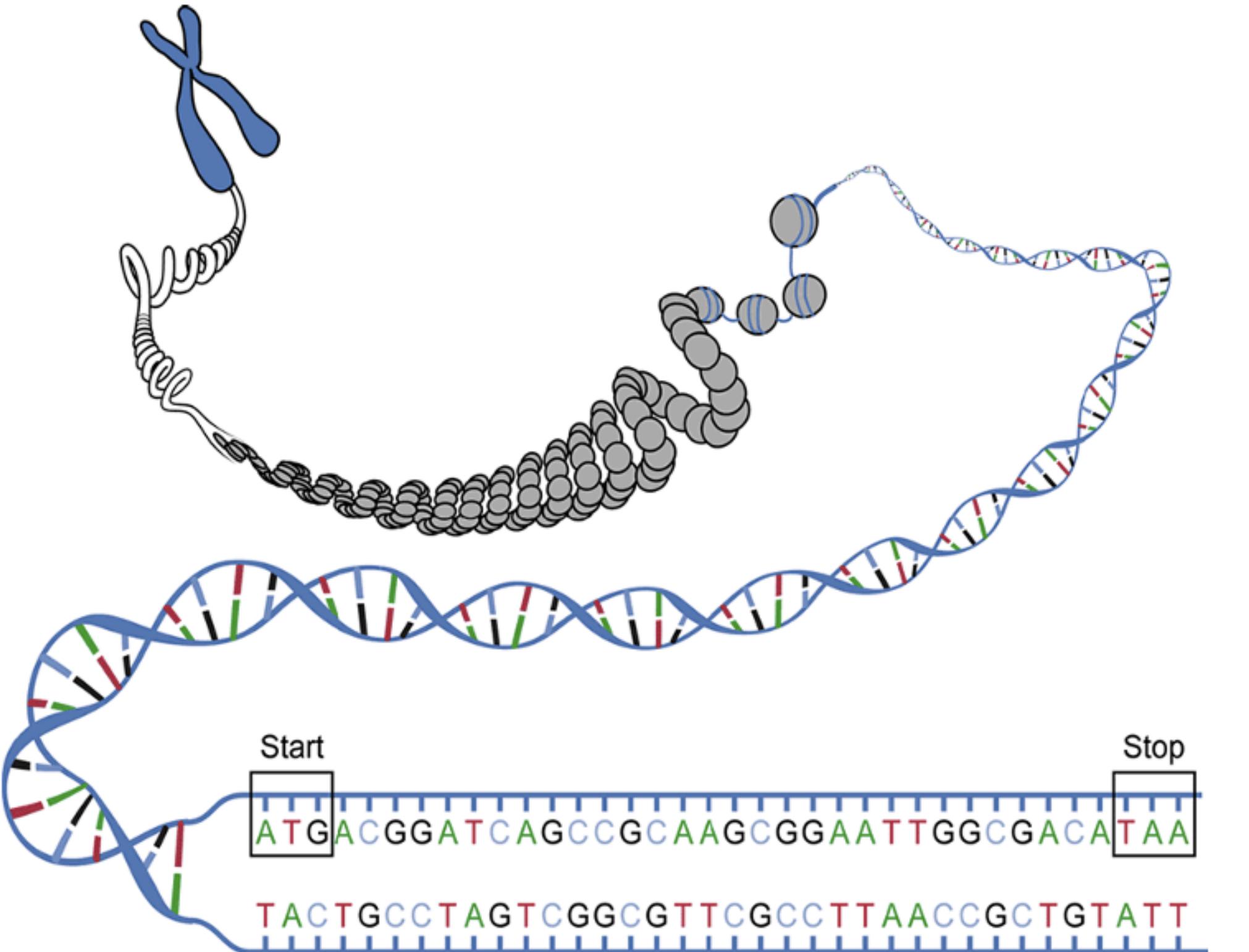


# DNA رشته‌های



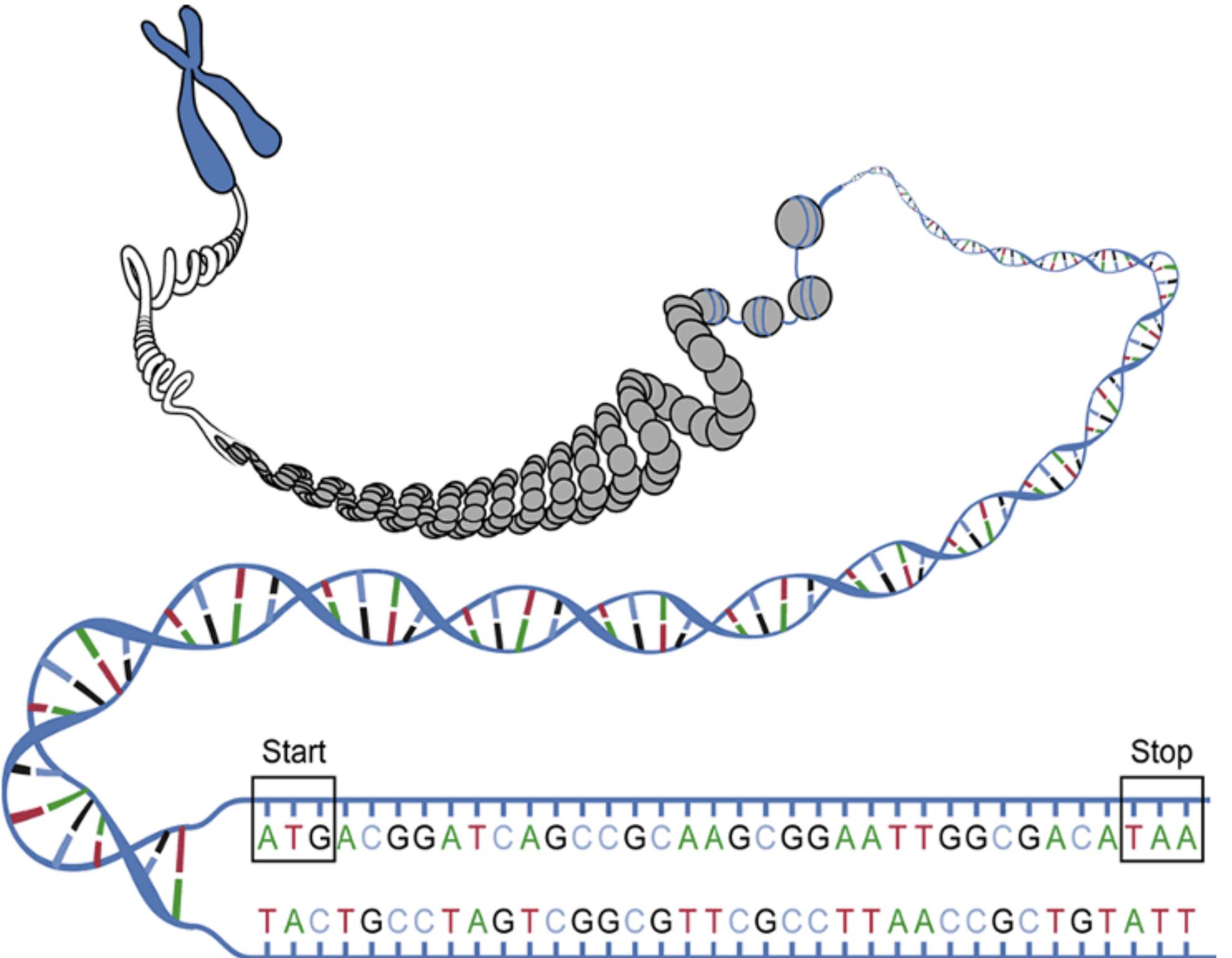
- توالی‌های ژنومی
- یک رشته از الفبای {A, T, C, G}

# DNA رشته‌های



- توالی‌های ژنومی
- یک رشته از الفبای {A, T, C, G}
- ژنوم انسان = ۳ میلیارد (هر مجموعه کروموزومی)

# DNA رشته‌های

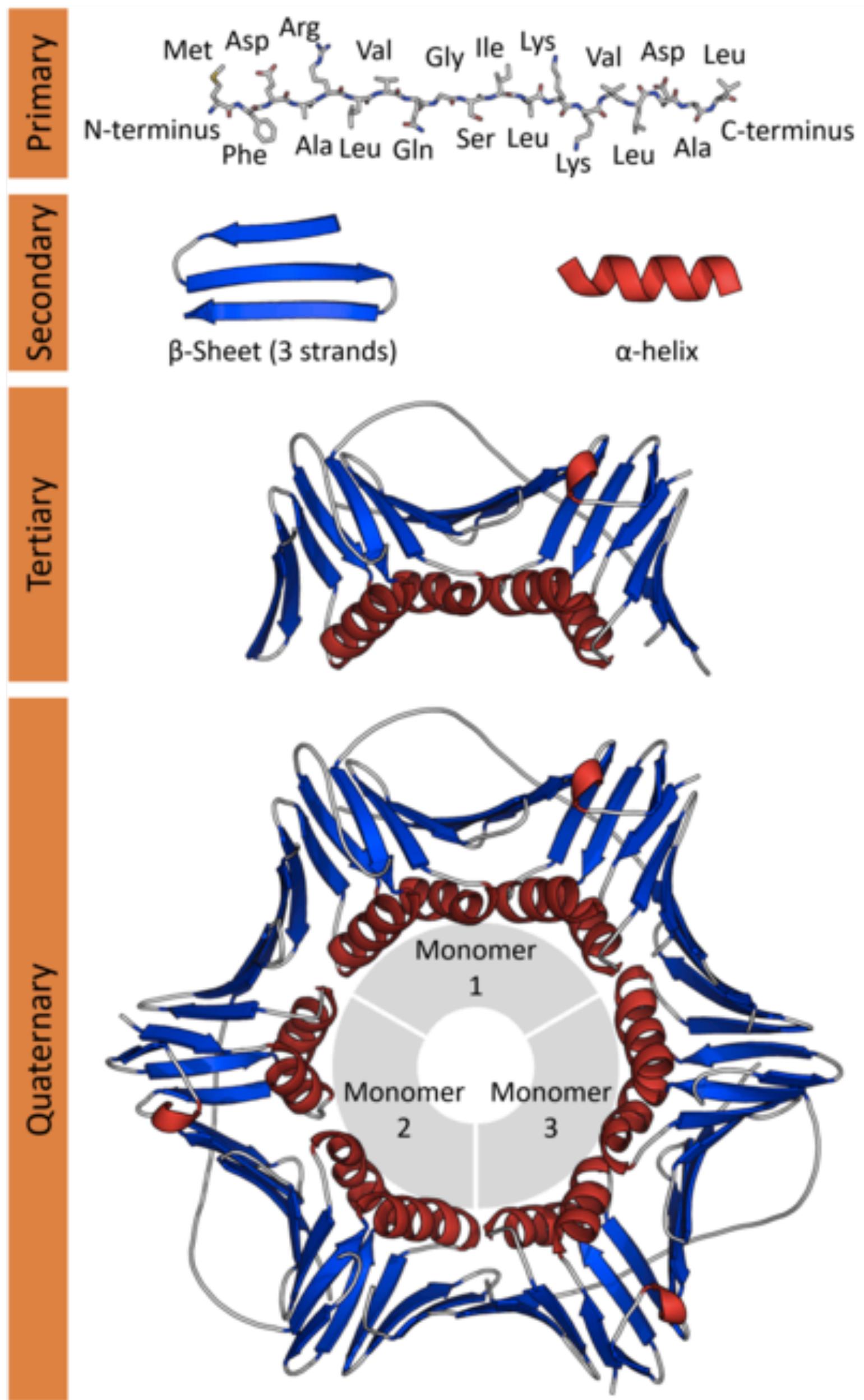


- توالی‌های ژنومی
- یک رشته از الفبای {A, T, C, G}
- ژنوم انسان = ۳ میلیارد (هر مجموعه کروموزومی)

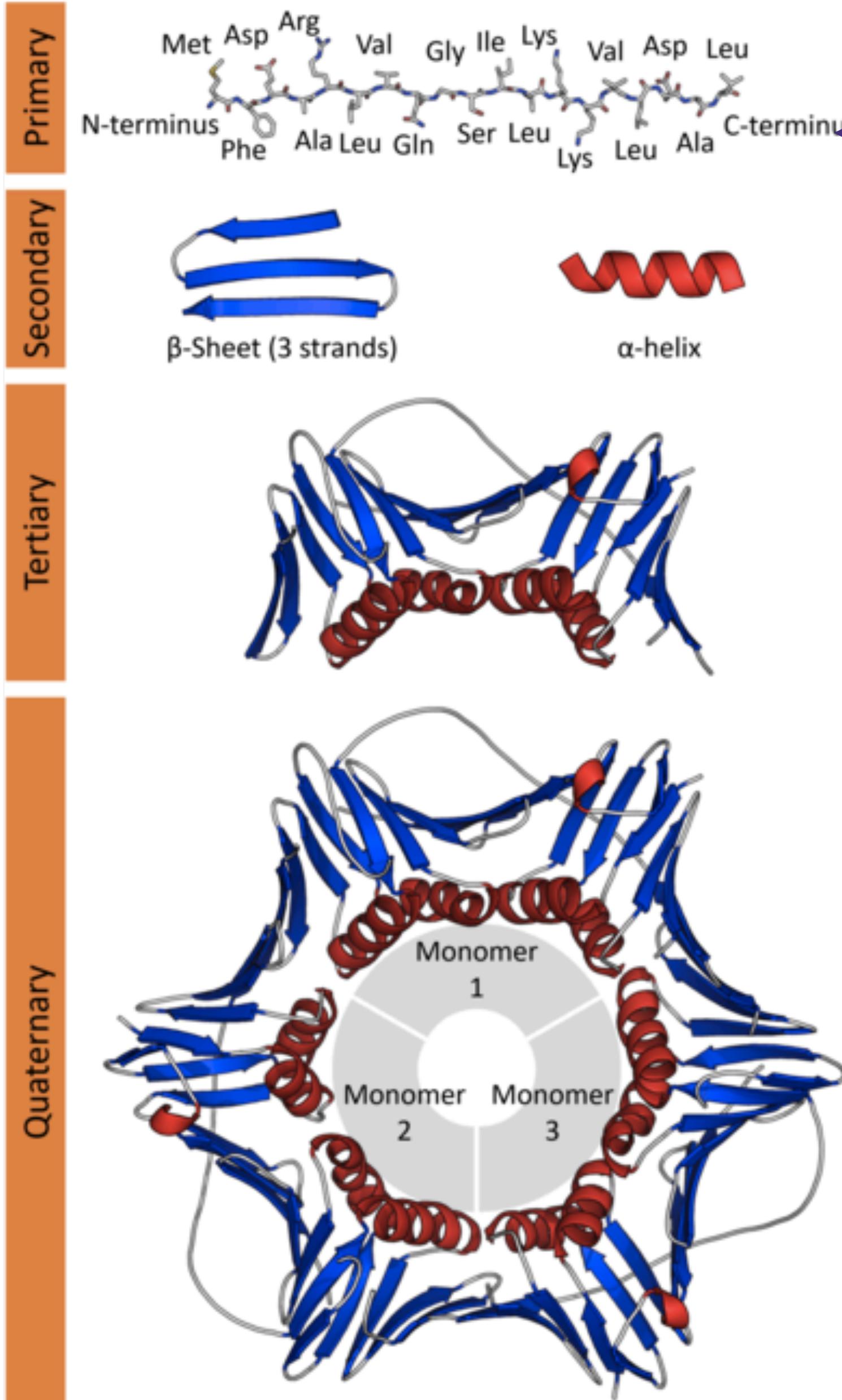
مثال

1 atcttggctc acaggggacg atgtcaagct cttcctggct ctttcctcagc cttgttgctg  
61 taactgctgc tcagtccacc attgaggaac aggccaagac atttttggac aagtttaacc  
121 acgaagccga agacctgttc tatcaaagtt cacttgcttc ttggaattat aacaccaata  
181 ttactgaaga gaatgtccaa aacatgaata atgctgggga caaatggtct gccttttaa  
241 aggaacagtc cacacttgcc caaatgtatc cactacaaga aattcagaat ctcacagtca

# رشته‌های پروتئینی



# رشته‌های پروتئینی



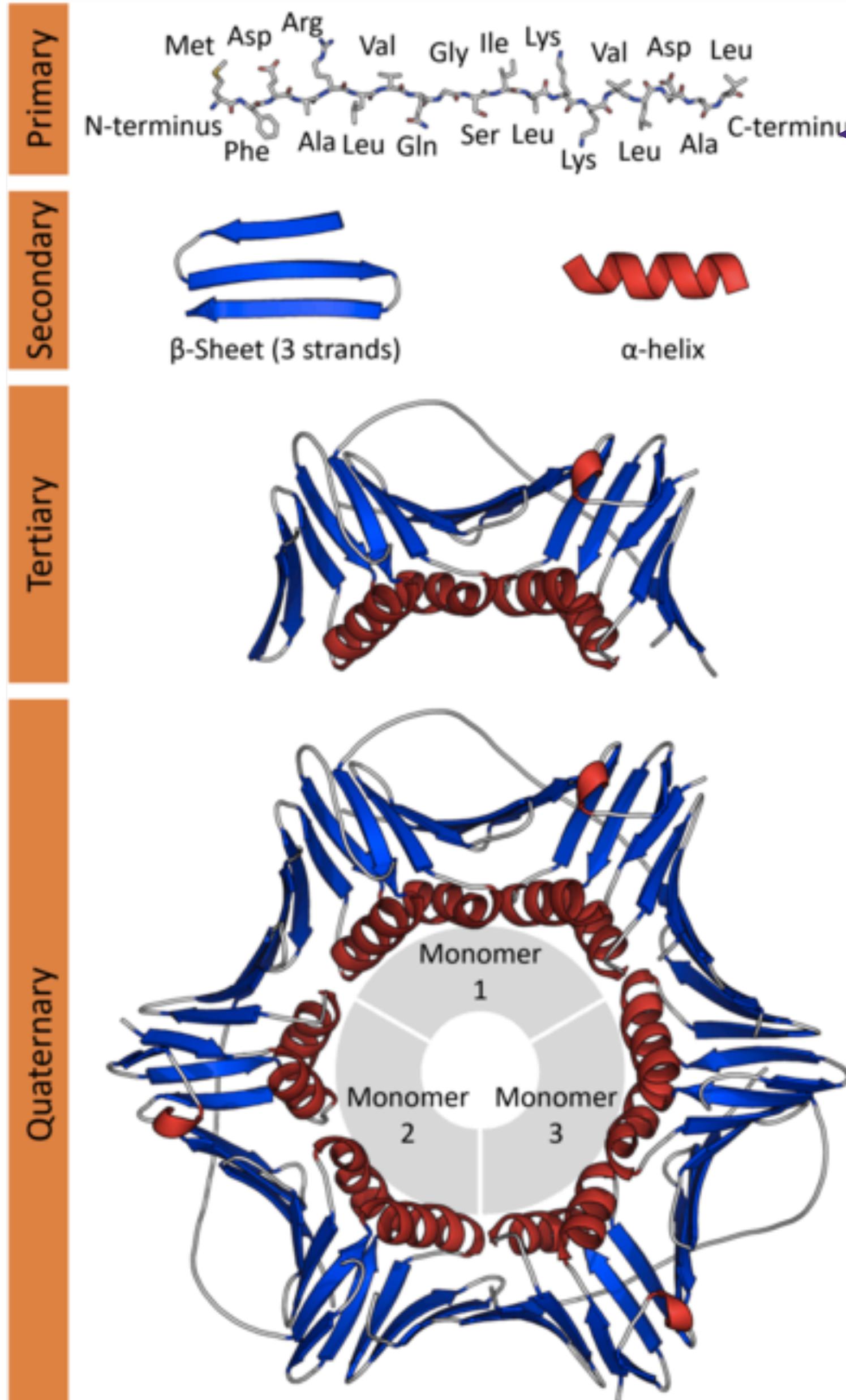
Amino acid	3- and 1-	
	3	1
<a href="#">Alanine</a>	Ala	A
<a href="#">Arginine</a>	Arg	R
<a href="#">Asparagine</a>	Asn	N
<a href="#">Aspartate</a>	Asp	D
<a href="#">Cysteine</a>	Cys	C
<a href="#">Glutamine</a>	Gln	Q
<a href="#">Glutamate</a>	Glu	E
<a href="#">Glycine</a>	Gly	G
<a href="#">Histidine</a>	His	H
<a href="#">Isoleucine</a>	Ile	I
<a href="#">Leucine</a>	Leu	L
<a href="#">Lysine</a>	Lys	K
<a href="#">Methionine</a>	Met	M
<a href="#">Phenylalanin</a>	Phe	F
<a href="#">Proline</a>	Pro	P
<a href="#">Serine</a>	Ser	S
<a href="#">Threonine</a>	Thr	T
<a href="#">Tryptophan</a>	Trp	W
<a href="#">Tyrosine</a>	Tyr	Y
<a href="#">Valine</a>	Val	V

# رشته‌های پروتئینی

● توالی پروتئینی = یک رشته از الفبای

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

● ۲۰ حرفی



Amino acid	3- and 1-	
	3	1
<u>Alanine</u>	Ala	A
<u>Arginine</u>	Arg	R
<u>Asparagine</u>	Asn	N
<u>Aspartate</u>	Asp	D
<u>Cysteine</u>	Cys	C
<u>Glutamine</u>	Gln	Q
<u>Glutamate</u>	Glu	E
<u>Glycine</u>	Gly	G
<u>Histidine</u>	His	H
<u>Isoleucine</u>	Ile	I
<u>Leucine</u>	Leu	L
<u>Lysine</u>	Lys	K
<u>Methionine</u>	Met	M
<u>Phenylalanin</u>	Phe	F
<u>Proline</u>	Pro	P
<u>Serine</u>	Ser	S
<u>Threonine</u>	Thr	T
<u>Tryptophan</u>	Trp	W
<u>Tyrosine</u>	Tyr	Y
<u>Valine</u>	Val	V

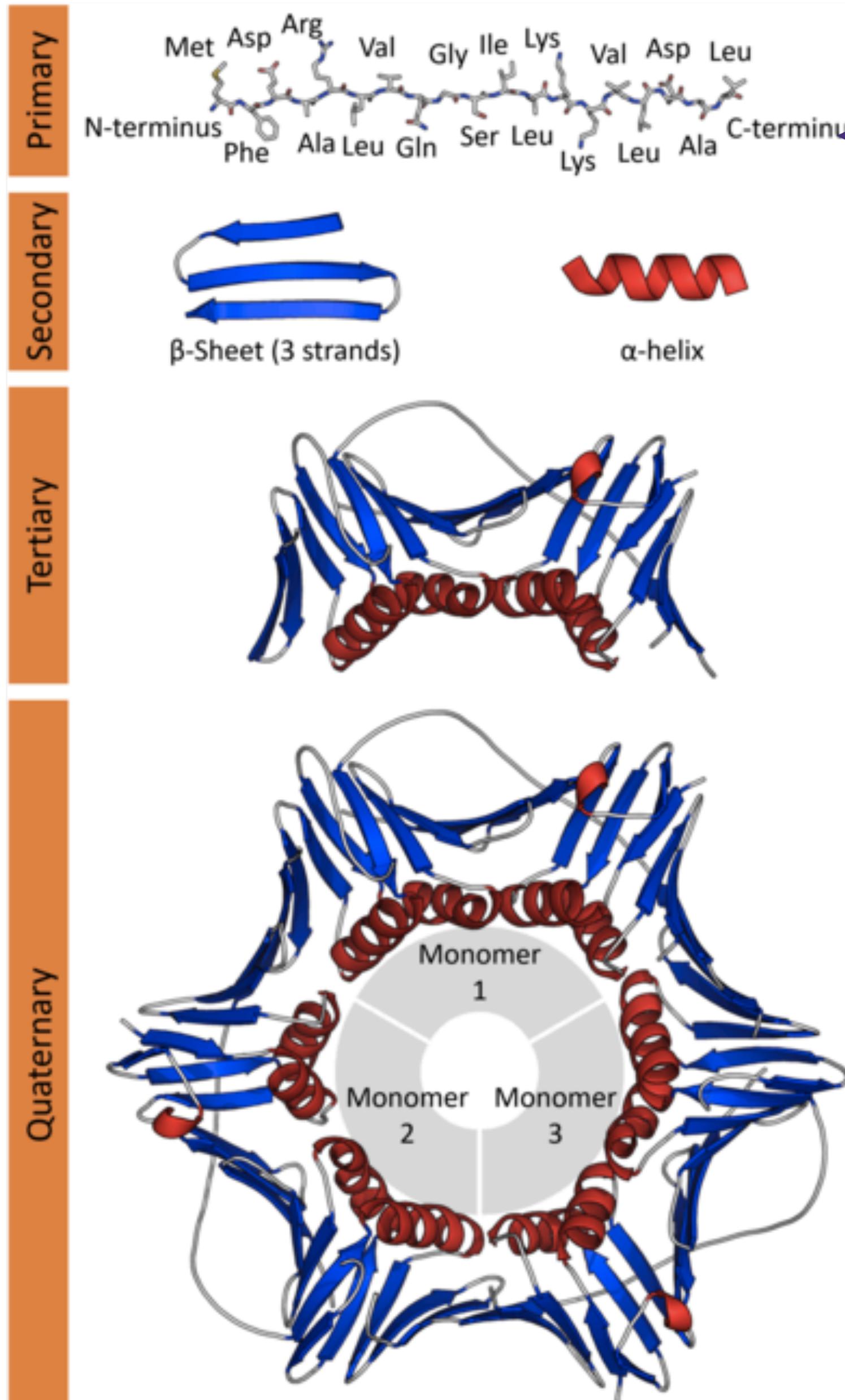
# رشته‌های پروتئینی

● توالی پروتئینی = یک رشته از الفبای

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

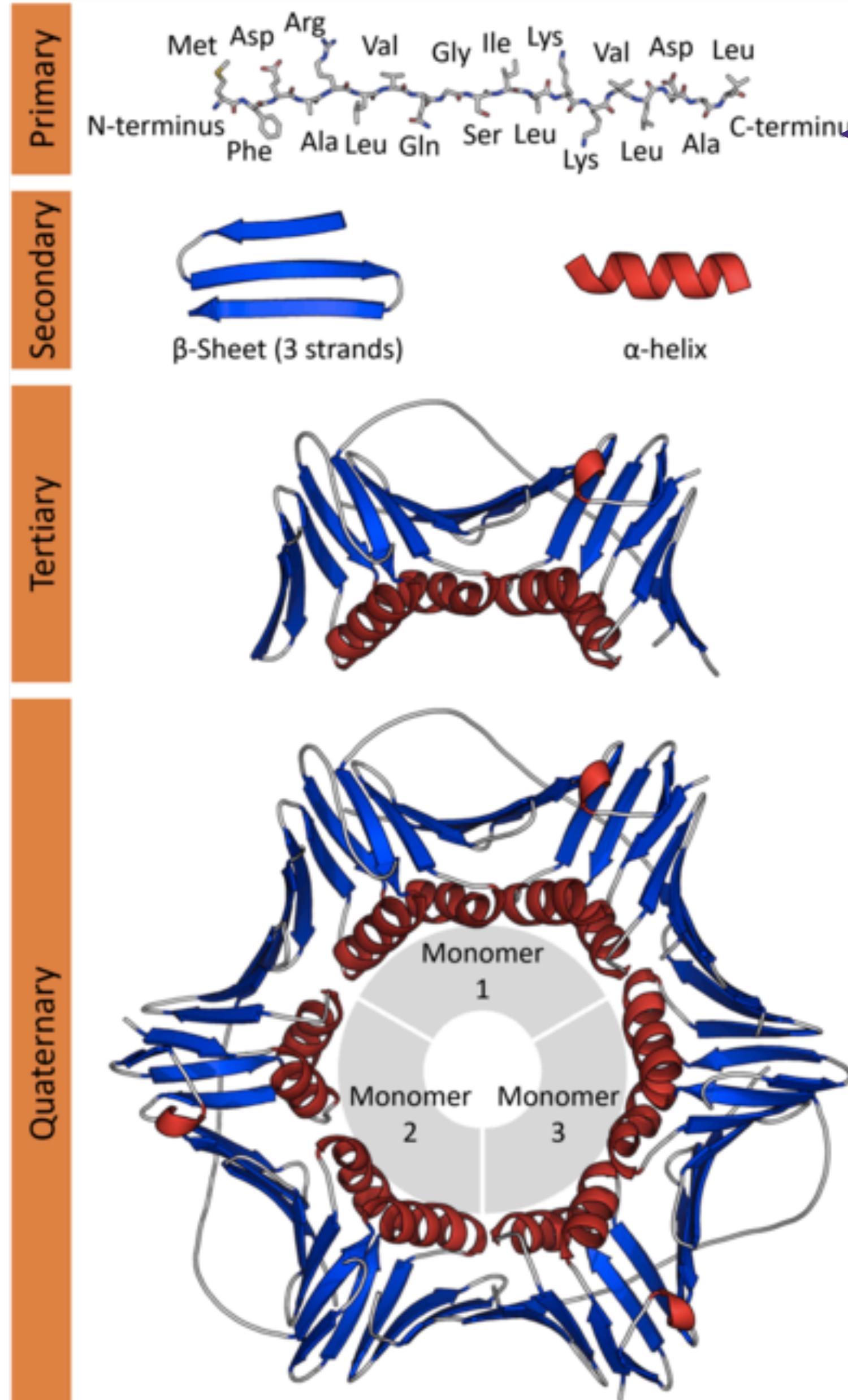
● ۲۰ حرفي

● طول > ۲۰۰۰



Amino acid	3- and 1-	
	3	1
<u>Alanine</u>	Ala	A
<u>Arginine</u>	Arg	R
<u>Asparagine</u>	Asn	N
<u>Aspartate</u>	Asp	D
<u>Cysteine</u>	Cys	C
<u>Glutamine</u>	Gln	Q
<u>Glutamate</u>	Glu	E
<u>Glycine</u>	Gly	G
<u>Histidine</u>	His	H
<u>Isoleucine</u>	Ile	I
<u>Leucine</u>	Leu	L
<u>Lysine</u>	Lys	K
<u>Methionine</u>	Met	M
<u>Phenylalanin</u>	Phe	F
<u>Proline</u>	Pro	P
<u>Serine</u>	Ser	S
<u>Threonine</u>	Thr	T
<u>Tryptophan</u>	Trp	W
<u>Tyrosine</u>	Tyr	Y
<u>Valine</u>	Val	V

# رشته‌های پروتئینی



Amino acid	3- and 1-	
	3	1
<a href="#">Alanine</a>	Ala	A
<a href="#">Arginine</a>	Arg	R
<a href="#">Asparagine</a>	Asn	N
<a href="#">Aspartate</a>	Asp	D
<a href="#">Cysteine</a>	Cys	C
<a href="#">Glutamine</a>	Gln	Q
<a href="#">Glutamate</a>	Glu	E
<a href="#">Glycine</a>	Gly	G
<a href="#">Histidine</a>	His	H
<a href="#">Isoleucine</a>	Ile	I
<a href="#">Leucine</a>	Leu	L
<a href="#">Lysine</a>	Lys	K
<a href="#">Methionine</a>	Met	M
<a href="#">Phenylalanin</a>	Phe	F
<a href="#">Proline</a>	Pro	P
<a href="#">Serine</a>	Ser	S
<a href="#">Threonine</a>	Thr	T
<a href="#">Tryptophan</a>	Trp	W
<a href="#">Tyrosine</a>	Tyr	Y
<a href="#">Valine</a>	Val	V

● توالی پروتئینی = یک رشته از الفبای

ARNDQCQEGHIL  
KMFPSTWYV

● ۲۰ حرفي

● طول > ۲۰۰۰

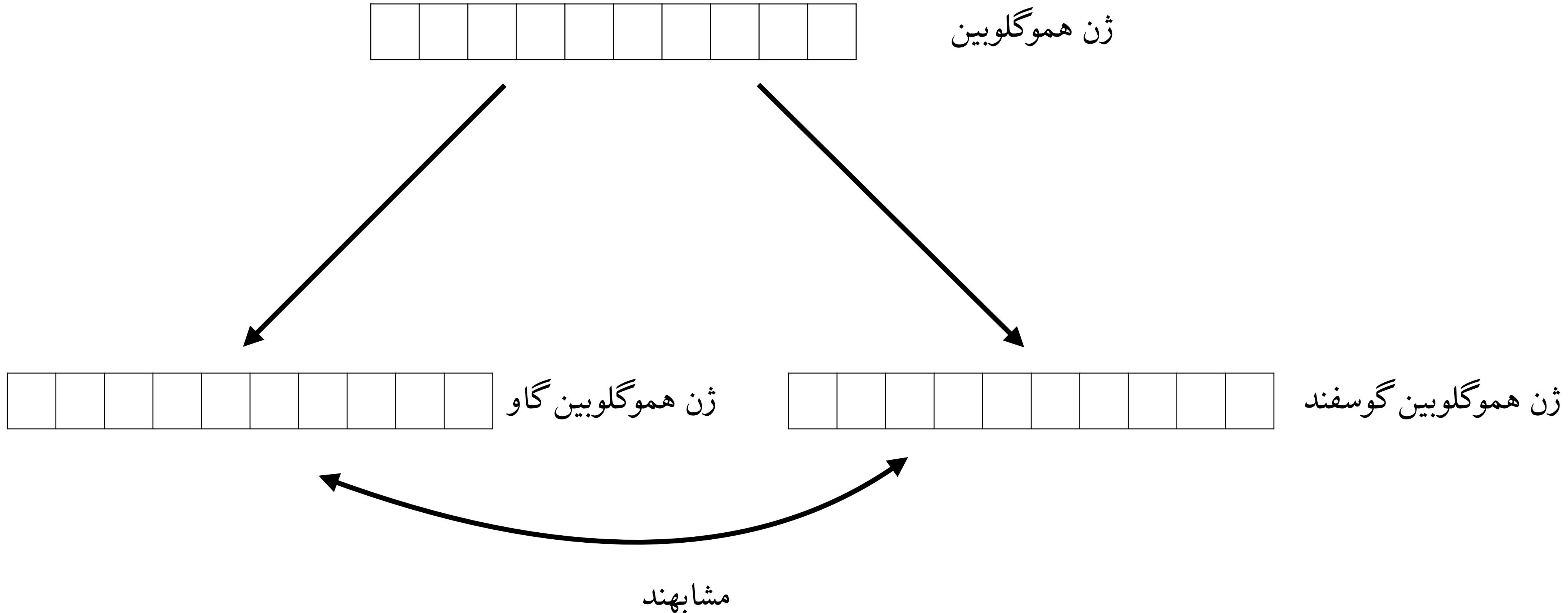
● مثال:

MSSSWLLLSSLVAVTAAQS  
TIEEQAKTFLDKFNHEAE  
DLFYQSS

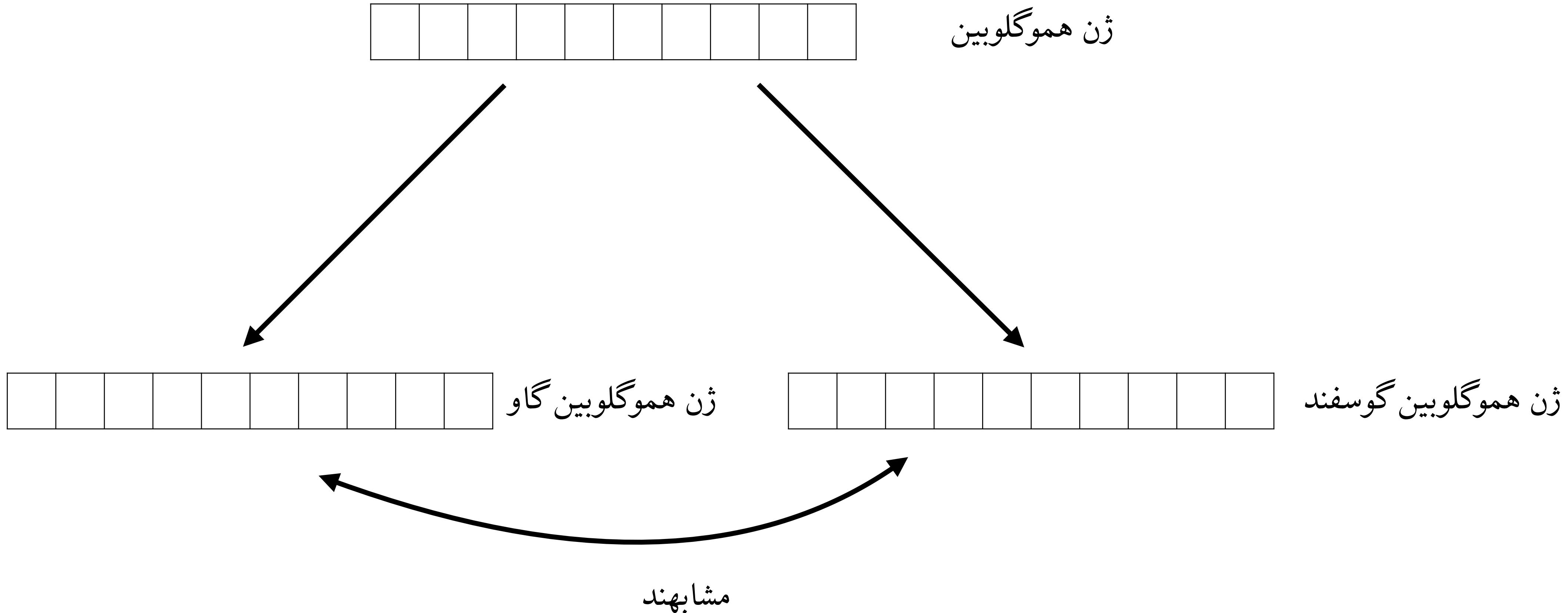
# هم ترازی



# چرا هم ترازی مهم است؟



# چرا هم ترازی مهم است؟



● شباخت => تکامل، شباخت => عملکرد یکسان، کمی شباخت => فاصله تکاملی، ...

# مسئله فاصله ویرایشی

- ورودی: دو رشته  $S$  و  $T$
- عملیات:
  - الف) حذف یک حرف
  - ب) اضافه کردن یک حرف
  - ج) تغییر یک حرف به حرف دیگر
- خروجی: کمترین عملیات لازم برای تبدیل  $S$  به  $T$

# مسئله فاصله ویرایشی

● مثال:

$S = ACAAT\ CC$  ●

$T = AGCAT\ GC$  ●

● ورودی: دو رشته S و T

● عملیات:

● الف) حذف یک حرف

● ب) اضافه کردن یک حرف

● ج) تغییر یک حرف به حرف دیگر

● خروجی: کمترین عملیات لازم برای تبدیل S به T

# مسئله فاصله ویرایشی

● مثال:

$S = \text{ACAAT CC}$  ●

$T = \text{AGCAT GC}$  ●

● روش تبدیل ۱: سه تا تغییر

● ورودی: دو رشته S و T

● عملیات:

● الف) حذف یک حرف

● ب) اضافه کردن یک حرف

● ج) تغییر یک حرف به حرف دیگر

● خروجی: کمترین عملیات لازم برای تبدیل S به T

# مسئله فاصله ویرایشی

● مثال:

$S = ACAAT\ CC$  ●

$T = AGCAT\ GC$  ●

● روش تبدیل ۱: سه تا تغییر

● روش تبدیل ۲: اضافه کردن G، حذف A دوم،  
تغییر C به G

● ورودی: دو رشته S و T

● عملیات:

● الف) حذف یک حرف

● ب) اضافه کردن یک حرف

● ج) تغییر یک حرف به حرف دیگر

● خروجی: کمترین عملیات لازم برای تبدیل S به T

# مسئله هم ترازی ساده

● مثال:

$S = ACAAT\ CC$  ●

$T = AGCAT\ GC$  ●

● ورودی: دو رشته  $S$  و  $T$

● خروجی:

● یک جدول

● سطر اول  $S$  که تعدادی «\_» به آن اضافه شده

● سطر اول  $T$  که تعدادی «\_» به آن اضافه شده

s1	-	s2	s3	s4	-	-	-	s5	-
-	T1	T2	T3	-	-	T4	T5	T6	T7

● دارای بیشترین ستون‌های مساوی

# مسئله هم ترازی ساده

● مثال:

S = ACAAT CC ●

T = AGCAT GC ●

● مثال هم ترازی:

S = A-CAATCC

T = AGCA-TGC

● سطر اول S که تعدادی «\_» به آن اضافه شده

● سطر اول T که تعدادی «\_» به آن اضافه شده

s1	-	s2	s3	s4	-	-	-	s5	-
-	T1	T2	T3	-	-	T4	T5	T6	T7

● دارای بیشترین ستون‌های مساوی

# «مسئله فاصله ویرایشی ساده»

S = A-CAATCC

T = AGCA-TGC

● قضیه: اگر امتیاز همترازی بهینه  $a^*$  باشد و هزینه روش ویرایش بهینه  $e^*$  باشد داریم:

$$e^* = |T_a| - a^*$$

● ایده اثبات:

● هر همترازی  $\iff$  روش ویرایش

$$e = |T_a| - a$$

● هر روش ویرایش  $\iff$  همترازی

$$a \geq |T_a| - e$$

سرتاسری

# مسئله هم ترازی

● ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  
 $\delta : (\Sigma \cup \{-\})^2 \rightarrow \mathbb{R}$

● خروجی:

● یک جدول

● سطر اول  $S$  که تعدادی «-» به آن اضافه شده

● سطر اول  $T$  که تعدادی «-» به آن اضافه شده

s1	-	s2	s3	s4	-	-	-	s5	-
-	T1	T2	T3	-	-	T4	T5	T6	T7

● دارای بیشترین امتیاز

● امتیاز = جمع امتیاز ستون‌ها (بر اساس  $\delta$ )

سرتاسری

$\delta$

# مسئله هم ترازی

	-	A	C	G	T
-	-1	-1	-1	-1	-1
A	-1	2	-1	-1	-1
C	-1	-1	2	-1	-1
G	-1	-1	-1	2	-1
T	-1	-1	-1	-1	2

مثال:

ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف

$$\delta : (\Sigma \cup \{ - \})^2 \rightarrow \mathbb{R}$$

$S = \text{ACAAT CC}$

خروجی:

$T = \text{AGCAT GC}$

یک جدول

سطر اول  $S$  که تعدادی «-» به آن اضافه شده

سطر اول  $T$  که تعدادی «-» به آن اضافه شده

s1	-	s2	s3	s4	-	-	-	s5	-
-	T1	T2	T3	-	-	T4	T5	T6	T7

دارای بیشترین امتیاز

امتیاز = جمع امتیاز ستون‌ها (بر اساس  $\delta$ )

$\delta$

# مسئله هم ترازی

	-	A	C	G	T
-	-1	-1	-1	-1	-1
A	-1	2	-1	-1	-1
C	-1	-1	2	-1	-1
G	-1	-1	-1	2	-1
T	-1	-1	-1	-1	2

مثال:

$$S = \text{ACAAT CC}$$

$$T = \text{AGCAT GC}$$

مثال هم ترازی:

$$S = \text{A-CAATCC}$$

$$T = \text{AGCA-TGC}$$

ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف

$$\delta : (\Sigma \cup \{-\})^2 \rightarrow \mathbb{R}$$

خروجی:

یک جدول

سطر اول  $S$  که تعدادی «-» به آن اضافه شده

سطر اول  $T$  که تعدادی «-» به آن اضافه شده

s1	-	s2	s3	s4	-	-	-	s5	-
-	T1	T2	T3	-	-	T4	T5	T6	T7

دارای بیشترین امتیاز

امتیاز = جمع امتیاز ستون‌ها (بر اساس  $\delta$ )

$\delta$

# مسئله هم ترازی

	-	A	C	G	T
-	-1	-1	-1	-1	-1
A	-1	2	-1	-1	-1
C	-1	-1	2	-1	-1
G	-1	-1	-1	2	-1
T	-1	-1	-1	-1	2

مثال:

$$S = \text{ACAAT CC}$$

$$T = \text{AGCAT GC}$$

مثال هم ترازی:

$$S = \text{A-CAATCC}$$

$$T = \text{AGCA-TGC}$$

امتیاز: ???

ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف

$$\delta : (\Sigma \cup \{-\})^2 \rightarrow \mathbb{R}$$

خروجی:

یک جدول

سطر اول  $S$  که تعدادی «-» به آن اضافه شده

سطر اول  $T$  که تعدادی «-» به آن اضافه شده

s1	-	s2	s3	s4	-	-	-	s5	-
-	T1	T2	T3	-	-	T4	T5	T6	T7

دارای بیشترین امتیاز

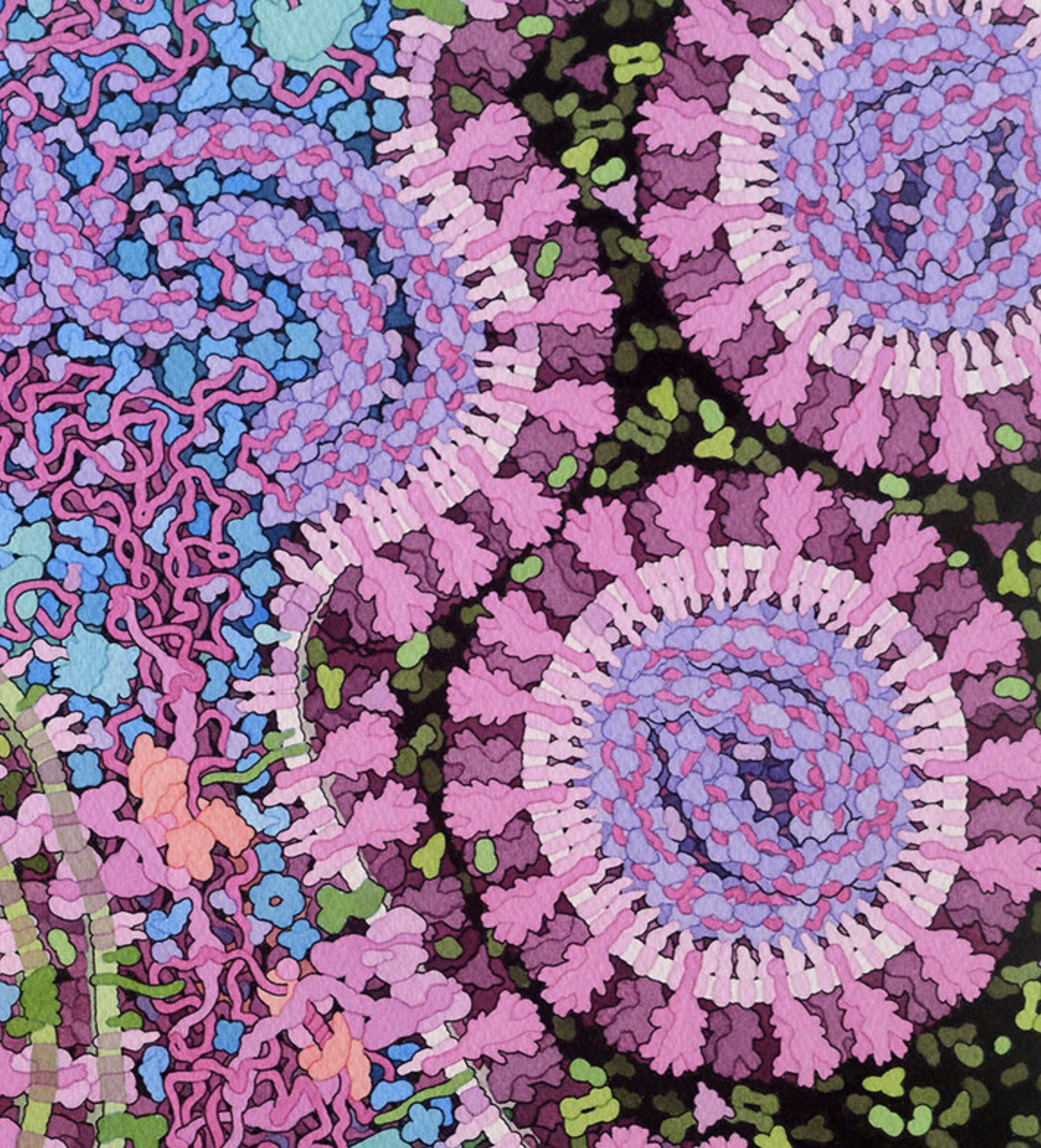
امتیاز = جمع امتیاز ستون‌ها (بر اساس  $\delta$ )

بسم الله الرحمن الرحيم

# ڙنو ميڪ محاسباتي

جلسه ۳: هم ترازي توالي (۲)

ترم پايز ۱۴۰۱-۱۴۰۰



# الگوريتم Needleman-Wunsch

● ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  $\delta : (\Sigma \cup \{ - \})^2 \rightarrow \mathbb{R}$

# الگوريتم Needleman-Wunsch

- ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  $\delta : (\Sigma \cup \{-\})^2 \rightarrow \mathbb{R}$
- روش: برنامهنویسی پویا

# الگوريتم Needleman-Wunsch

- ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  $\delta : (\Sigma \cup \{-\})^2 \rightarrow \mathbb{R}$
- روش: برنامهنویسی پویا
- تعریف  $V(i, j)$ : بیشترین امتیاز هم ترازی بین  $S[1..i]$  و  $T[1..j]$

# الگوريتم Needleman-Wunsch

- ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  $\delta : (\Sigma \cup \{ - \})^2 \rightarrow \mathbb{R}$
- روش: برنامهنویسی پویا
- تعریف ( $V$ ): بیشترین امتیاز هم ترازی بین  $S[1..i]$  و  $T[1..j]$
- رابطه بازگشتی:

$$V(i, j) = \max \begin{cases} V(i - 1, j - 1) + \delta(S[i], T[j]) & \text{match/mismatch} \\ V(i - 1, j) + \delta(S[i], -) & \text{delete} \\ V(i, j - 1) + \delta(-, T[j]) & \text{insert} \end{cases}$$

# الگوريتم Needleman-Wunsch

- ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  $\delta : (\Sigma \cup \{ - \})^2 \rightarrow \mathbb{R}$
- روش: برنامهنویسی پویا
- تعریف ( $V$ ): بیشترین امتیاز هم ترازی بین  $S[1..i]$  و  $T[1..j]$
- رابطه بازگشتی:

$$V(i, j) = \max \begin{cases} V(i - 1, j - 1) + \delta(S[i], T[j]) & \text{match/mismatch} \\ V(i - 1, j) + \delta(S[i], -) & \text{delete} \\ V(i, j - 1) + \delta(-, T[j]) & \text{insert} \end{cases}$$

- حالت پایه:
  - $V(0, 0) = 0$
  - $V(0, j) = V(0, j - 1) + \delta(-, T[j])$  Insert  $j$  times
  - $V(i, 0) = V(i - 1, 0) + \delta(S[i], -)$  Delete  $i$  times

# مثال لـNeedleman-Wunsch

$S = \text{AC AAT C C}$

$T = \text{AGCATGC}$

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

$\delta$

	-	A	C	G	T
-	-1	-1	-1	-1	-1
A	-1	2	-1	-1	-1
C	-1	-1	2	-1	-1
G	-1	-1	-1	2	-1
T	-1	-1	-1	-1	2

# محاسبه خود بهترین هم ترازی؟

$T = \text{AGCATGC}$

$S = \text{AC AAT CC C}$

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

# الگوريتم Needleman-Wunsch

- ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  $\delta : (\Sigma \cup \{ - \})^2 \rightarrow \mathbb{R}$
- روش: برنامه‌نویسی پویا
- تعریف  $V(i, j)$ : بیشترین امتیاز هم‌ترازی بین  $S[1..i]$  و  $T[1..j]$
- رابطه بازگشتی:

$$V(i, j) = \max \begin{cases} V(i - 1, j - 1) + \delta(S[i], T[j]) & \text{match/mismatch} \\ V(i - 1, j) + \delta(S[i], -) & \text{delete} \\ V(i, j - 1) + \delta(-, T[j]) & \text{insert} \end{cases}$$

- حالت پایه:  $V(0, 0) = 0$
- $V(0, j) = V(0, j - 1) + \delta(-, T[j])$  Insert  $j$  times
- $V(i, 0) = V(i - 1, 0) + \delta(S[i], -)$  Delete  $i$  times

زمان اجرا:

# الگوریتم Needleman-Wunsch

- ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  $\delta : (\Sigma \cup \{ - \})^2 \rightarrow \mathbb{R}$
- روش: برنامه‌نویسی پویا
- تعریف  $V(i, j)$ : بیشترین امتیاز هم‌ترازی بین  $S[1..i]$  و  $T[1..j]$
- رابطه بازگشتی:

$$V(i, j) = \max \begin{cases} V(i - 1, j - 1) + \delta(S[i], T[j]) & \text{match/mismatch} \\ V(i - 1, j) + \delta(S[i], -) & \text{delete} \\ V(i, j - 1) + \delta(-, T[j]) & \text{insert} \end{cases}$$

- حالت پایه:
  - $V(0, 0) = 0$
  - $V(0, j) = V(0, j - 1) + \delta(-, T[j])$  Insert  $j$  times
  - $V(i, 0) = V(i - 1, 0) + \delta(S[i], -)$  Delete  $i$  times

زمان اجرا:  
 $O(mn)$

# الگوریتم Needleman-Wunsch

- ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  $\delta : (\Sigma \cup \{ - \})^2 \rightarrow \mathbb{R}$
- روش: برنامه‌نویسی پویا
- تعریف  $V(i, j)$ : بیشترین امتیاز هم‌ترازی بین  $S[1..i]$  و  $T[1..j]$
- رابطه بازگشتی:

$$V(i, j) = \max \begin{cases} V(i - 1, j - 1) + \delta(S[i], T[j]) & \text{match/mismatch} \\ V(i - 1, j) + \delta(S[i], -) & \text{delete} \\ V(i, j - 1) + \delta(-, T[j]) & \text{insert} \end{cases}$$

- حالت پایه:
  - $V(0, 0) = 0$
  - $V(0, j) = V(0, j - 1) + \delta(-, T[j])$  Insert  $j$  times
  - $V(i, 0) = V(i - 1, 0) + \delta(S[i], -)$  Delete  $i$  times

# الگوريتم Needleman-Wunsch

زمان اجرا:  
 $O(mn)$

حافظه:

ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  $\delta : (\Sigma \cup \{ - \})^2 \rightarrow \mathbb{R}$

روش: برنامهنویسی پویا

تعريف  $V(i, j)$ : بیشترین امتیاز هم ترازی بین  $S[1..i]$  و  $T[1..j]$

رابطه بازگشتی:

$$V(i, j) = \max \begin{cases} V(i - 1, j - 1) + \delta(S[i], T[j]) & \text{match/mismatch} \\ V(i - 1, j) + \delta(S[i], -) & \text{delete} \\ V(i, j - 1) + \delta(-, T[j]) & \text{insert} \end{cases}$$

حالت پایه:  $V(0, 0) = 0$

$V(0, j) = V(0, j - 1) + \delta(-, T[j])$  Insert  $j$  times

$V(i, 0) = V(i - 1, 0) + \delta(S[i], -)$  Delete  $i$  times

# الگوريتم Needleman-Wunsch

زمان اجرا:  
 $O(mn)$

حافظه:  
 $O(mn)$

- ورودی: دو رشته  $S$  و  $T$  و ماتریس مشابهت حروف  $\delta : (\Sigma \cup \{ - \})^2 \rightarrow \mathbb{R}$
- روش: برنامهنویسی پویا
- تعریف  $V(i, j)$ : بیشترین امتیاز هم ترازی بین  $S[1..i]$  و  $T[1..j]$
- رابطه بازگشتی:

$$V(i, j) = \max \begin{cases} V(i - 1, j - 1) + \delta(S[i], T[j]) & \text{match/mismatch} \\ V(i - 1, j) + \delta(S[i], -) & \text{delete} \\ V(i, j - 1) + \delta(-, T[j]) & \text{insert} \end{cases}$$

- حالت پایه:
  - $V(0, 0) = 0$
  - $V(0, j) = V(0, j - 1) + \delta(-, T[j])$  Insert  $j$  times
  - $V(i, 0) = V(i - 1, 0) + \delta(S[i], -)$  Delete  $i$  times

# بهترین‌های هم‌ترازی دقیق

● الگوریتم Needleman-Wunsch زمان  $O(mn)$  حافظه  $O(mn)$

# بهترین‌های هم‌ترازی دقیق

- الگوریتم Needleman-Wunsch زمان  $O(mn)$  حافظه  $O(mn)$
- بهترین الگوریتم که می‌شناسیم  $O(n^2/\log^2 n)$
- روش ۴ - روس

# بهترین‌های هم‌ترازی دقیق

- الگوریتم Needleman-Wunsch زمان  $O(mn)$  حافظه  $O(mn)$ :
- بهترین الگوریتم که می‌شناسیم  $O(n^2/\log^2 n)$
- روش ۴-روس
- کران پایین  $\Omega(n \log n)$  زمان لازم است.
- (قدیمی): حداقل

# بهترین‌های هم‌ترازی دقیق

- الگوریتم Needleman-Wunsch زمان  $O(mn)$  حافظه  $O(mn)$ :
- بهترین الگوریتم که می‌شناسیم  $O(n^2/\log^2 n)$
- روش ۴-روس
- کران پایین  $\Omega(n \log n)$  زمان لازم است.
- (قدیمی): حداقل هیچ الگوریتمی در زمان  $O(n^{2-\delta})$  برای مسئله وجود ندارد، در صورت درست بودن SETH.
- (اخیراً): به ازای هر  $0 < \delta$  هیچ الگوریتمی در زمان  $O(n^{2-\delta})$  برای مسئله وجود ندارد، در صورت درست بودن SETH.

# کمتر کردن حافظه

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

حافظه الگوریتم هم ترازی =  $O(mn)$

مثلا برای دو زنوم؟

# کمتر کردن حافظه

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

- حافظه الگوریتم هم ترازی =  $O(mn)$
- مثلا برای دو زنوم؟
- (ایده اول) نگهداری آخرین سطر مشکل: خود هم ترازی؟

# کمتر کردن حافظه

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

- حافظه الگوریتم هم ترازی =  $O(mn)$

- مثلا برای دو زنوم؟

- ایده اول) نگهداری آخرین سطر

- مشکل: خود هم ترازی؟

- ایده دوم) یافتن قدم میانی

● تعریف ماتریس  $[j]: M[i,j]$  بهینه  $V[i,j]$  از کدام خانه از ستون  $n/2$  از  $V$  به دست آمده؟

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

$M[i-1,x] = M[i,j]$

اگر  $V[i-1,x]$  از روی  $V[i,j]$  پر شده بود

اگر  $x, i=n/2+1$

اگر  $M[i-1,x] : i > n/2+1$

● تعریف ماتریس  $[j]: M[i,j]$  بهینه  $V[i,j]$  از کدام خانه از ستون  $n/2$  از  $V$  به دست آمده؟

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

$M[i-1,x] = M[i,j]$

اگر  $V[i-1,x]$  از روی  $V[i,j]$  پر شده بود

اگر  $x, i=n/2+1$

اگر  $M[i-1,x] : i > n/2+1$

● تعریف ماتریس  $[j]: M[i,j]$  بهینه  $V[i,j]$  از کدام خانه از ستون  $n/2$  از  $V$  به دست آمده؟

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

$$M[i-1,x] = M[i,j]$$

اگر  $V[i-1,x]$  از روی  $V[i,j]$  پر شده بود

اگر  $x, i=n/2+1$

اگر  $M[i-1,x] : i > n/2+1$

<=> بدون هزینه اضافی می دانیم در بهترین هم رდیفی

$T[1..M[n,m]]$  به  $S[1..n/2]$  منطبق شده و

$T[M[n,m]+1..m]$  به  $S[n/2+1..n]$  منطبق شده

● تعریف ماتریس  $[j]: M[i,j]$  بهینه  $V[i,j]$  از کدام خانه از ستون  $n/2$  از  $V$  به دست آمده؟

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

$$M[i-1,x] = M[i,j]$$

اگر  $V[i-1,x]$  از روی  $V[i,j]$  پر شده بود

اگر  $x, i=n/2+1$

اگر  $M[i-1,x] : i > n/2+1$

<=> بدون هزینه اضافی می دانیم در بهترین هم رდیفی

$T[1..M[n,m]]$  به  $S[1..n/2]$  منطبق شده و

$T[M[n,m]+1..m]$  به  $S[n/2+1..n]$  منطبق شده

● تعریف ماتریس  $[j]: M[i,j]$  بهینه  $V[i,j]$  از کدام خانه از ستون  $n/2$  از  $V$  به دست آمده؟

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

$$M[i-1,x] = M[i,j]$$

اگر  $V[i-1,x]$  از روی  $V[i,j]$  پر شده بود

اگر  $x, i=n/2+1$

اگر  $M[i-1,x] : i > n/2+1$

<=> بدون هزینه اضافی می دانیم در بهترین هم رდیفی

$T[1..M[n,m]]$  به  $S[1..n/2]$  منطبق شده و

$T[M[n,m]+1..m]$  به  $S[n/2+1..n]$  منطبق شده

● تحلیل زمان اجرا:

● تحليل زمان اجرا:

$$T(n, m) =$$

● تحلیل زمان اجرا:

$$T(n, m) = O(mn)$$

● تحلیل زمان اجرا:

$$T(n, m) = O(mn) + T(n/2, x) + T(n/2, m - x)$$

● تحلیل زمان اجرا:

$$T(n, m) = O(mn) + T(n/2, x) + T(n/2, m - x)$$

● تحليل زمان اجرا:

$$\begin{aligned} T(n, m) = & O(mn) + \cancel{T(n/2, x)} + \cancel{T(n/2, m - x)} \\ & + O(n/2 \times x) + O(n/2 \times (m - x)) + T(n/4, a_1) + T(n/4, a_2) + T(n/4, a_3) + T(n/4, a_4) \end{aligned}$$

● تحلیل زمان اجرا:

$$T(n, m) = O(mn) + T(n/2, x) + T(n/2, m - x)$$

$$+ O(n/2 \times x) + O(n/2 \times (m - x)) + T(n/4, a_1) + T(n/4, a_2) + T(n/4, a_3) + T(n/4, a_4)$$

$$T(n, m) = O(mn) + O(mn/2) + O(mn/4) + \dots$$

● تحليل زمان اجرا:

$$T(n, m) = O(mn) + T(n/2, x) + T(n/2, m - x)$$

$$+ O(n/2 \times x) + O(n/2 \times (m - x)) + T(n/4, a_1) + T(n/4, a_2) + T(n/4, a_3) + T(n/4, a_4)$$

$$T(n, m) = O(mn) + O(mn/2) + O(mn/4) + \dots$$

$$= O(mn)$$

الگوریتم هم دیفی با زمان  $O(m \cdot n)$  و حافظه  $O(m+n)$

● تحلیل زمان اجرا:

$$T(n, m) = O(mn) + T(n/2, x) + T(n/2, m - x)$$

$$+ O(n/2 \times x) + O(n/2 \times (m - x)) + T(n/4, a_1) + T(n/4, a_2) + T(n/4, a_3) + T(n/4, a_4)$$

$$T(n, m) = O(mn) + O(mn/2) + O(mn/4) + \dots$$

$$= O(mn)$$

# الگوریتم یکی از دانشجویان

- Align( $S[1..m]$ ,  $T[x..x+2^y]$ , given  $V[0..m,x]$ ):
  - Fill matrix  $V[0..m, x..x+2^y]$
  - Keep  $V[0..m, x+\{2^0, 2^1, \dots, 2^{\{y-1}\}]$  in memory
  - Call Align( $S[1..m]$ ,  $T[x+2^i..2^{\{i+1}\}]$ , given  $V[0..m, 2^i]$ ) for  $I = 0 .. y-1$

$$S(n) = O(mn) + \sum_{i=0}^{\log n - 1} S(2^i)$$

$$Q(n) := \sum_{i=0}^{\log n} S(2^i)$$

$$Q(n) := \sum_{i=0}^{\log n} S(2^i) = S(n) + \sum_{i=0}^{\log n - 1} S(2^i) = O(mn) + 2 \sum_{i=0}^{\log n - 1} S(2^i) = O(mn) + 2Q(n/2)$$

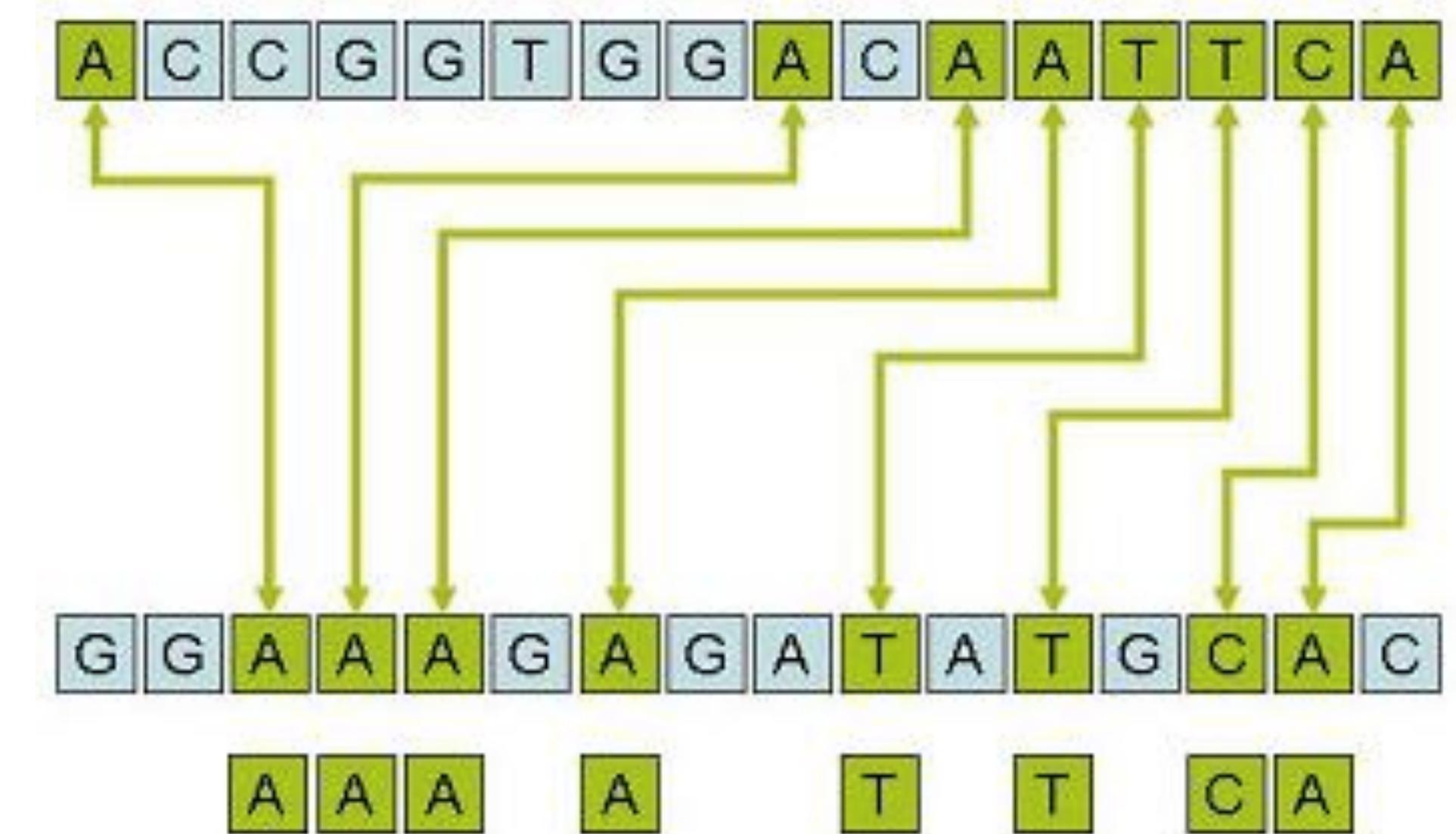
$$Q(n) = O(mn \log n)$$

$$S(n) = O(mn \log n)$$

زمان اجرا =  $S(n)$

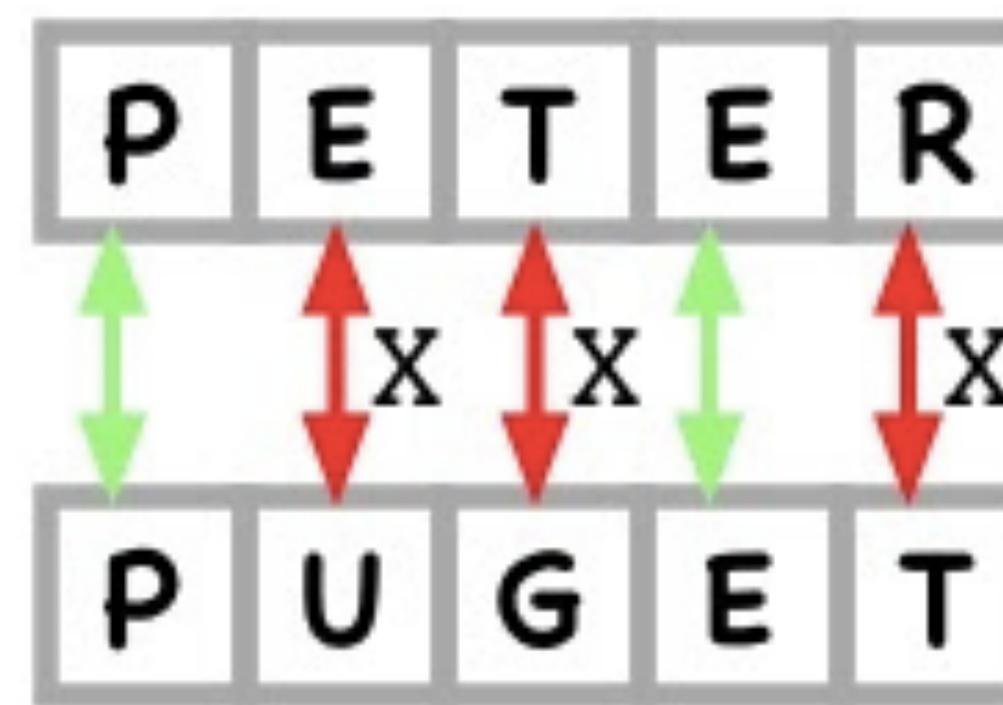
# با هم ترازی می‌توان مسئله‌های خاصی را حل کرد

- طول بزرگ‌ترین زیردنباله مشترک

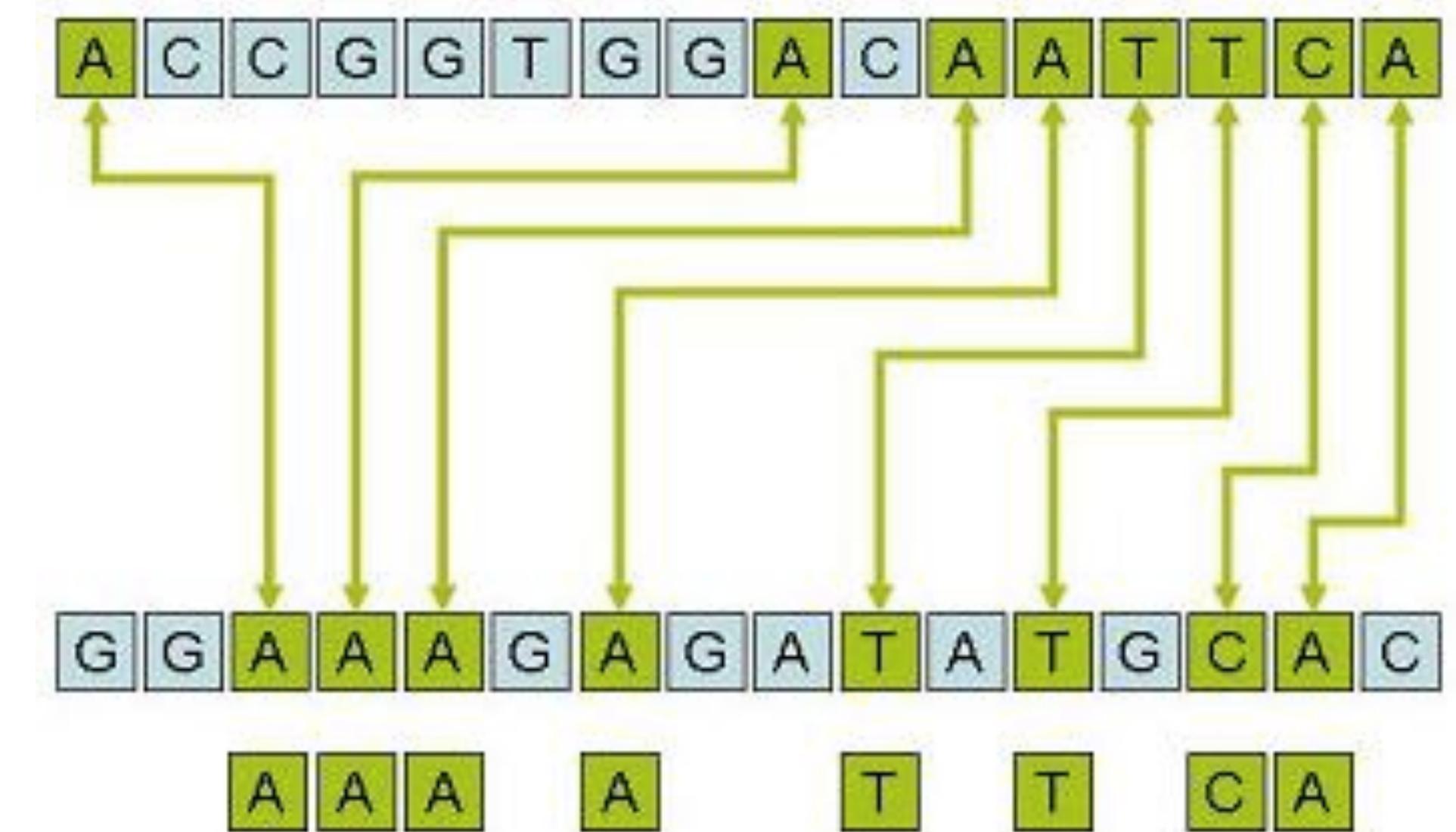


# با هم ترازی می‌توان مسئله‌های خاصی را حل کرد

● فاصله همینگ



● طول بزرگ‌ترین زیردنباله مشترک



# أنواع هم ترازي

• هم ترازي سرتاسری

# أنواع همترازى

- همترازى سرتاسری
- همترازى موضعی
- زیر دنباله‌هایی از  $S$  و  $T$  بیابیم که همترازی آن‌ها بیشترین امتیاز را داشته باشد
- الگوریتم؟

# الگوریتم هم ترازی موضعی

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

$$V(i, j) = \max \begin{cases} 0 & \text{align empty strings} \\ V(i - 1, j - 1) + \delta(S[i], T[j]) & \text{match/mismatch} \\ V(i - 1, j) + \delta(S[i], -) & \text{delete} \\ V(i, j - 1) + \delta(-, T[j]) & \text{insert} \end{cases}$$

# هزینه پرش

خطی

ATCCGAACATCCAATCGAAGC  
A---G--CATGCAAT-----

# هزینه پرش

ATCCGAACATCCAATCGAAGC  
A---G--CATGCAAT-----

خطی

آفین

$$g(q) = h + qs$$

# هزینه پرش

ATCCGAACATCCAATCGAAGC  
A---G--CATGCAAT-----

خطی ●

$$g(q) = h + qs \bullet$$

محدب ●

$$g(q) = a \log q + b \bullet$$

# تابع هزینه را از کجا پیاوریم؟

- برای DNA :  
مثال ۵+ برای انطباق
- مثال ۴ - برای عدم انطباق
- مثال ۱ - برای پرس  
تفاوت بین A-G و C-T

# تابع هزینه را از کجا پیاوریم؟

- برای DNA:
- مثلا ۵+ برای انطباق
- مثلا ۴- برای عدم انطباق
- مثلا ۱- برای پرس
- تفاوت بین A-G و C-T
- روش خوبی است؟

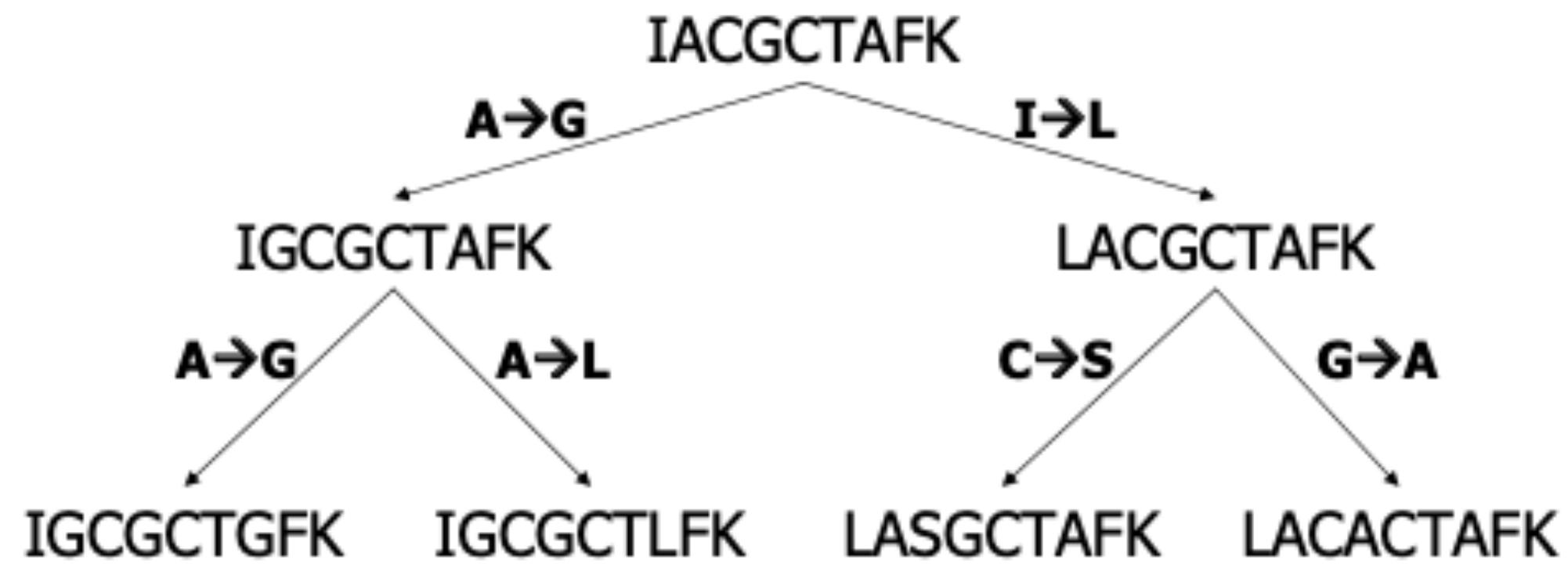
# تابع هزینه را از کجا پیاوریم؟

- برای DNA:
- مثلا ۵+ برای انطباق
- مثلا ۴- برای عدم انطباق
- مثلا ۱- برای پرس
- تفاوت بین A-G و C-T
- روش خوبی است؟
- برای پروتئین؟

# ماتریس امتیاز :PAM

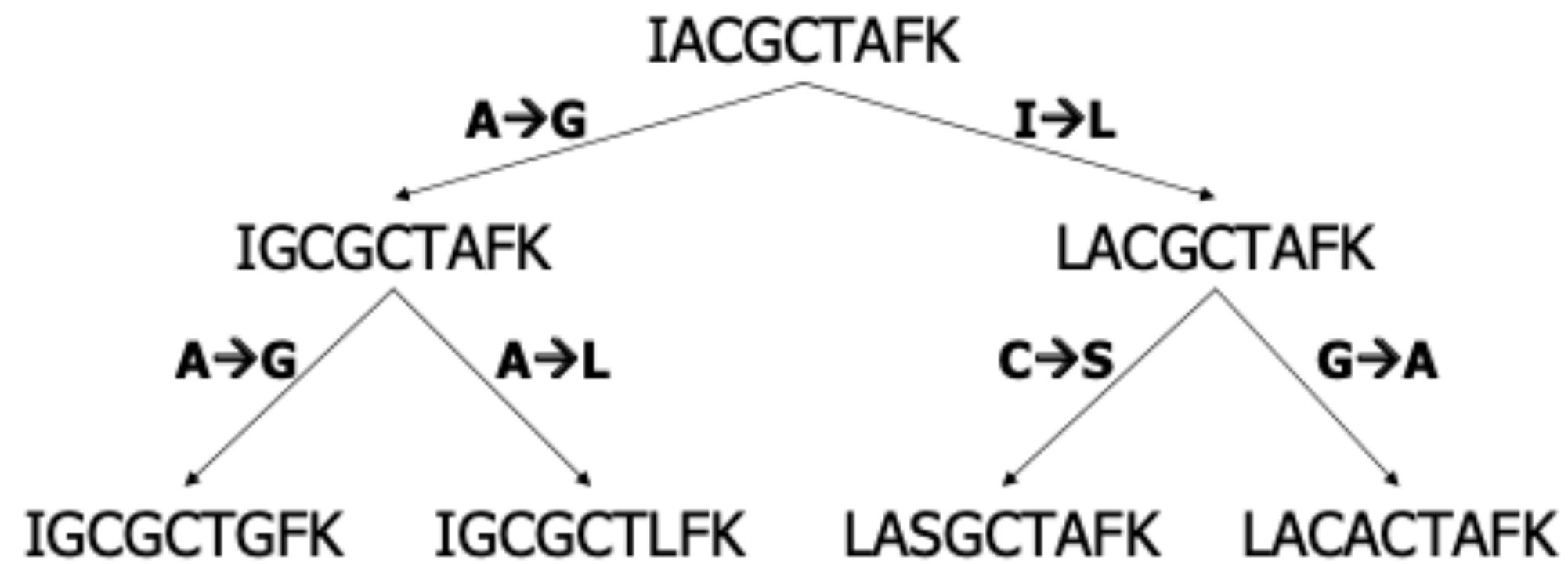
- ورودی:
- تعدادی رشته خیلی شبیه (تمام جهش‌ها را مشاهده می‌کنیم)
- درخت تبارزایی روی آن‌ها

# :PAM امتیاز ماتریس



- ورودی:
- تعدادی رشته خیلی شبیه (تمام جهش‌ها را مشاهده می‌کنیم)
- درخت تبارزایی روی آنها

# :PAM امتیاز ماتریس

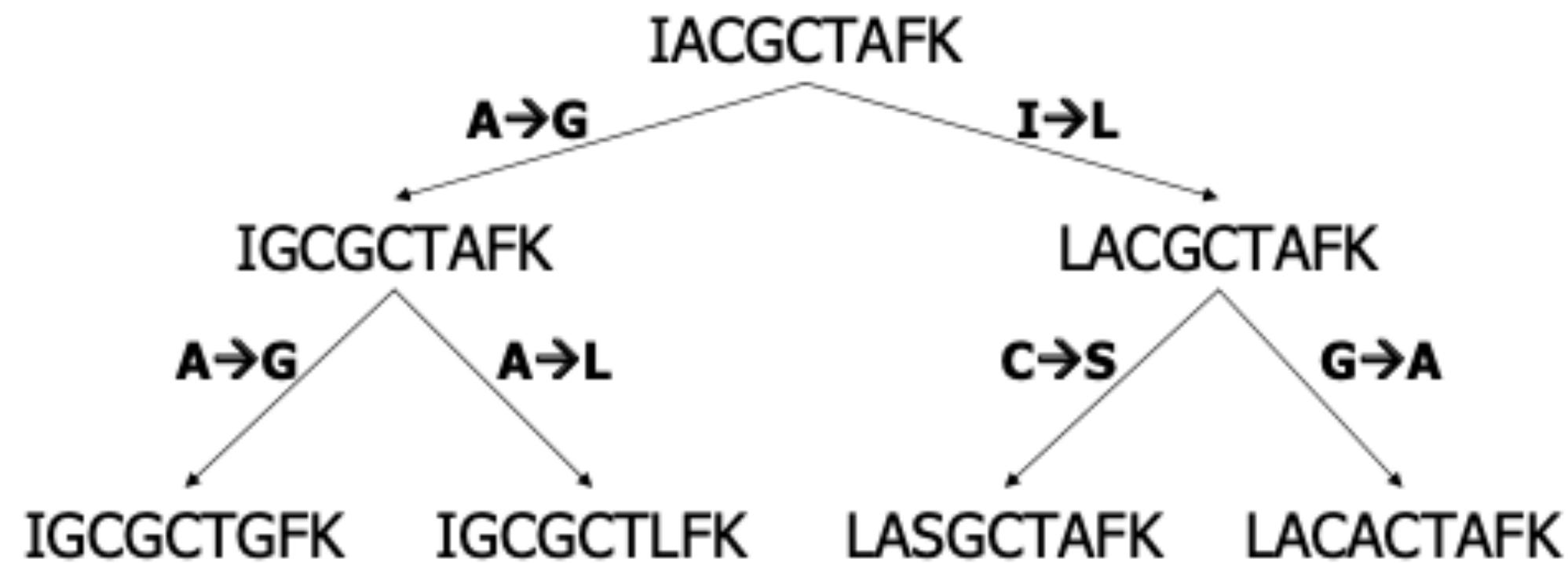


- ورودی:
- تعدادی رشته خیلی شبیه (تمام جهش‌ها را مشاهده می‌کنیم)
- درخت تبارزایی روی آنها

●  $M(i,j)$  ماتریس احتمال تبدیل  $i$  به  $j$ :

● فرض:

# :PAM ماتریس امتیاز



کنیم)

ورودی:

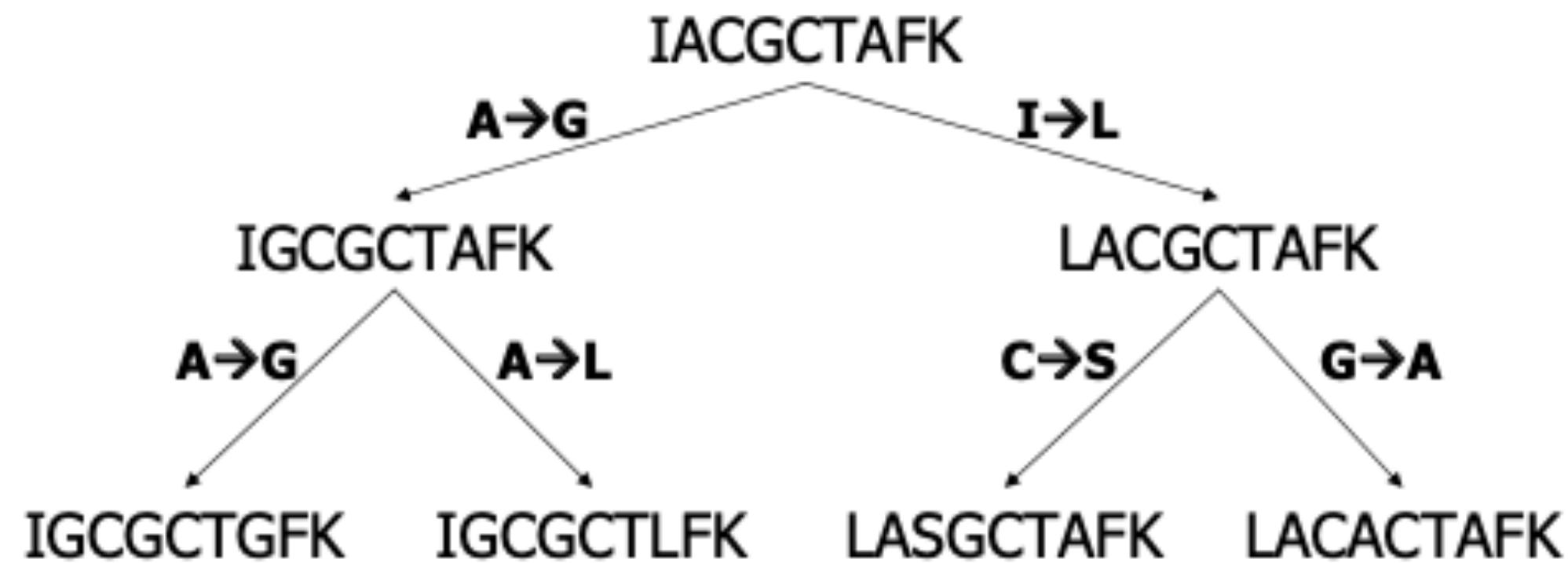
تعدادی رشته خیلی شبیه (تمام جهش‌ها را مشاهده می‌کنیم)

درخت تبارزایی روی آن‌ها

فرم:  $M(i,j)$  ماتریس احتمال تبدیل  $i$  به  $j$ :

تعداد  $j$  در تمام رشته‌ها  $n(j)$

# :PAM ماتریس امتیاز



تعدادی رشته خیلی شبیه (تمام جهش‌ها را مشاهده می‌کنیم)

تعداد جهش‌های بین  $i$  و  $j$

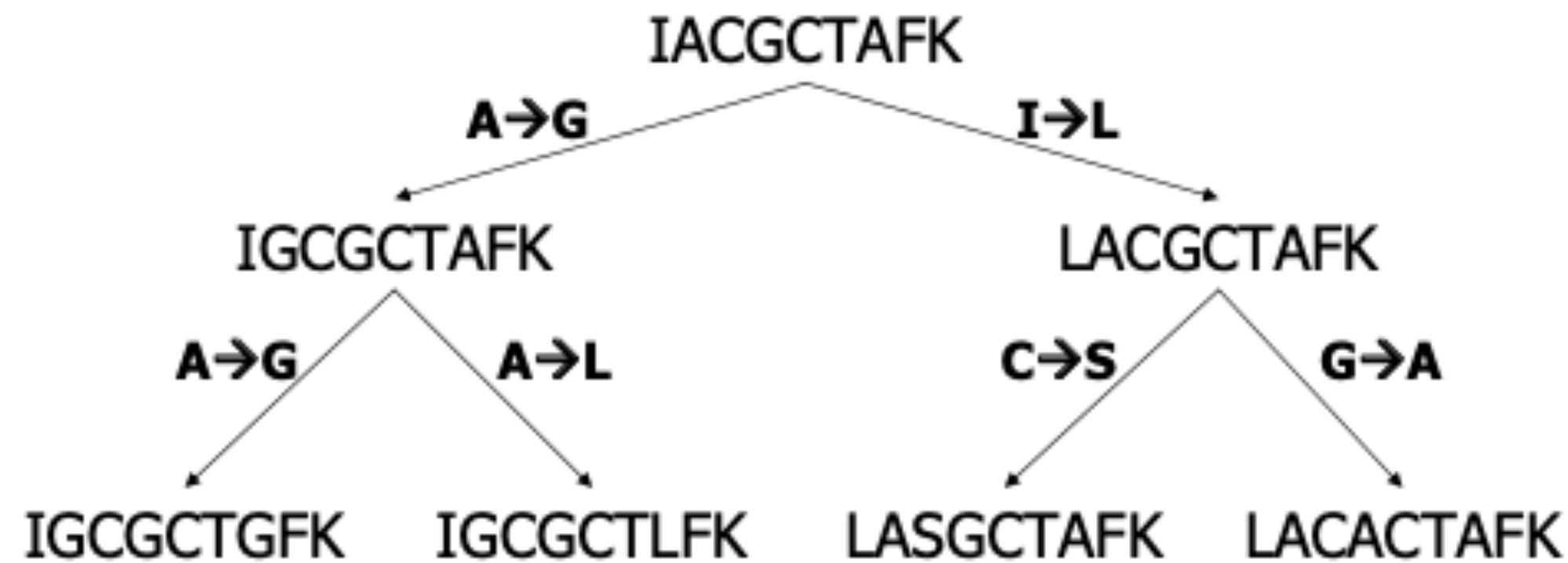
$$m(j) = \frac{\sum_{i=1, i \neq j}^{20} A(i, j)}{n(j)}$$

ماتریس احتمال تبدیل  $i$  به  $j$ :  $M(i,j)$

فرض:

تعداد  $j$  در تمام رشته‌ها  $n(j)$

# :PAM ماتریس امتیاز



جاهش

تعداد جاهش‌های بین  $i$  و  $j$

ماتریس احتمال تبدیل  $i$  به  $j$ :  $M(i,j)$

فرض:

$$m(j) = \frac{\sum_{i=1, i \neq j}^{20} A(i, j)}{n(j)}$$

تعداد  $j$  در تمام رشته‌ها  $n(j)$

$$M(i, j) = \lambda A(i, j) \frac{m(j)}{\sum_{i=1, i \neq j}^{20} A(i, j)}$$

کنترل نرخ جاهش

تعداد جهش‌های بین  $i$  و  $j$

$$m(j) = \frac{\sum_{i=1, i \neq j}^{20} A(i, j)}{n(j)}$$

کنترل نرخ جهش

$$M(i, j) = \lambda A(i, j) \frac{m(j)}{\sum_{i=1, i \neq j}^{20} A(i, j)}$$

تعداد  $j$  در تمام رشته‌ها  $n(j)$

تعداد جهش‌های بین  $i$  و  $j$

کنترل نرخ جهش

$$m(j) = \frac{\sum_{i=1, i \neq j}^{20} A(i, j)}{n(j)}$$

تعداد  $j$  در تمام رشته‌ها  $n(j)$

$$M(i, j) = \lambda A(i, j) \frac{m(j)}{\sum_{i=1, i \neq j}^{20} A(i, j)}$$

$$M(j, j) = 1 - \sum_{i=1, i \neq j}^{20} M(i, j)$$

تعداد جهش‌های بین  $i$  و  $j$

کنترل نرخ جهش

$$m(j) = \frac{\sum_{i=1, i \neq j}^{20} A(i, j)}{n(j)}$$

تعداد  $j$  در تمام رشته‌ها  $n(j)$

$$M(i, j) = \lambda A(i, j) \frac{m(j)}{\sum_{i=1, i \neq j}^{20} A(i, j)}$$

$$M(j, j) = 1 - \sum_{i=1, i \neq j}^{20} M(i, j)$$

$$0.99 = 1 - \lambda \sum_{j=1}^{20} A(i, j) \frac{m(j)}{\sum_{i=1, i \neq j}^{20} A(i, j)}$$

PAM<sub>1</sub>

تعداد جهش‌های بین i و j

$$m(j) = \frac{\sum_{i=1, i \neq j}^{20} A(i, j)}{n(j)}$$

کنترل نرخ جهش

تعداد j در تمام رشته‌ها n(j)

$$M(i, j) = \lambda A(i, j) \frac{m(j)}{\sum_{i=1, i \neq j}^{20} A(i, j)}$$

$$M(j, j) = 1 - \sum_{i=1, i \neq j}^{20} M(i, j)$$

$$0.99 = 1 - \lambda \sum_{j=1}^{20} A(i, j) \frac{m(j)}{\sum_{i=1, i \neq j}^{20} A(i, j)}$$

$$\text{PAM}_n(i, j) = \log \frac{f(j)M_n(i, j)}{f(i)f(j)} = \log \frac{f(j)M^n(i, j)}{f(i)f(j)} = \log \frac{M^n(i, j)}{f(i)}$$

:PAM<sub>n</sub>

نسبت «تبديل i به j» به «تصادفی کنار هم بودن i و j»

# ماتریس :BLOSUM

IACGCTAFK  
IGCGCTAFK  
LACGCTAFK  
IGCGCTGFK  
IGCGCTLFK  
LASGCTAFK  
LACACTAFK

● ورودی: هم رده‌ی فی چندگانه

IACGCTAFK  
IGCGCTAFK  
LACGCTAFK  
IGCGCTGFK  
IGCGCTLFK  
LASGCTAFK  
LACACTAFK

# ماتریس :BLOSUM

● ورودی: هم رده‌ی چندگانه

نسبت مشاهده a و b به کل  
مشاهده‌ها

$$\delta(a, b) = \frac{1}{\lambda} \ln \frac{O_{a,b}}{f_a f_b}$$

تصادفی کنار هم بودن a و b

# ماتریس :BLOSUM

IACGCTAFK  
IGCGCTAFK  
LACGCTAFK  
IGCGCTGFK  
IGCGCTLFK  
LASGCTAFK  
LACACTAFK

● ورودی: هم رده‌ی چندگانه

نسبت مشاهده a و b به کل  
مشاهده‌ها

$$\delta(a, b) = \frac{1}{\lambda} \ln \frac{O_{a,b}}{f_a f_b}$$

تصادفی کنار هم بودن a و b

$$O_{A,G} = 23/189$$

$$f_A = 9/63$$

$$f_G = 10/63 \quad E_{A,G} = 0.0227.$$

# ماتریس BLOSUM: (تمیزکاری)

- BLOSUM 62:

AVAAA  
AVAAA  
AVAAA  
AVLAA  
VVAAL



$AV[A_{0.75}L_{0.25}]AA$   
VV $AAL$

# رابطه دو گروه ماتریس

BLOSUM 80  $\approx$  PAM 1 ●

BLOSUM 62  $\approx$  PAM 120 ●

BLOSUM 45  $\approx$  PAM 250 ●

سؤال؟

