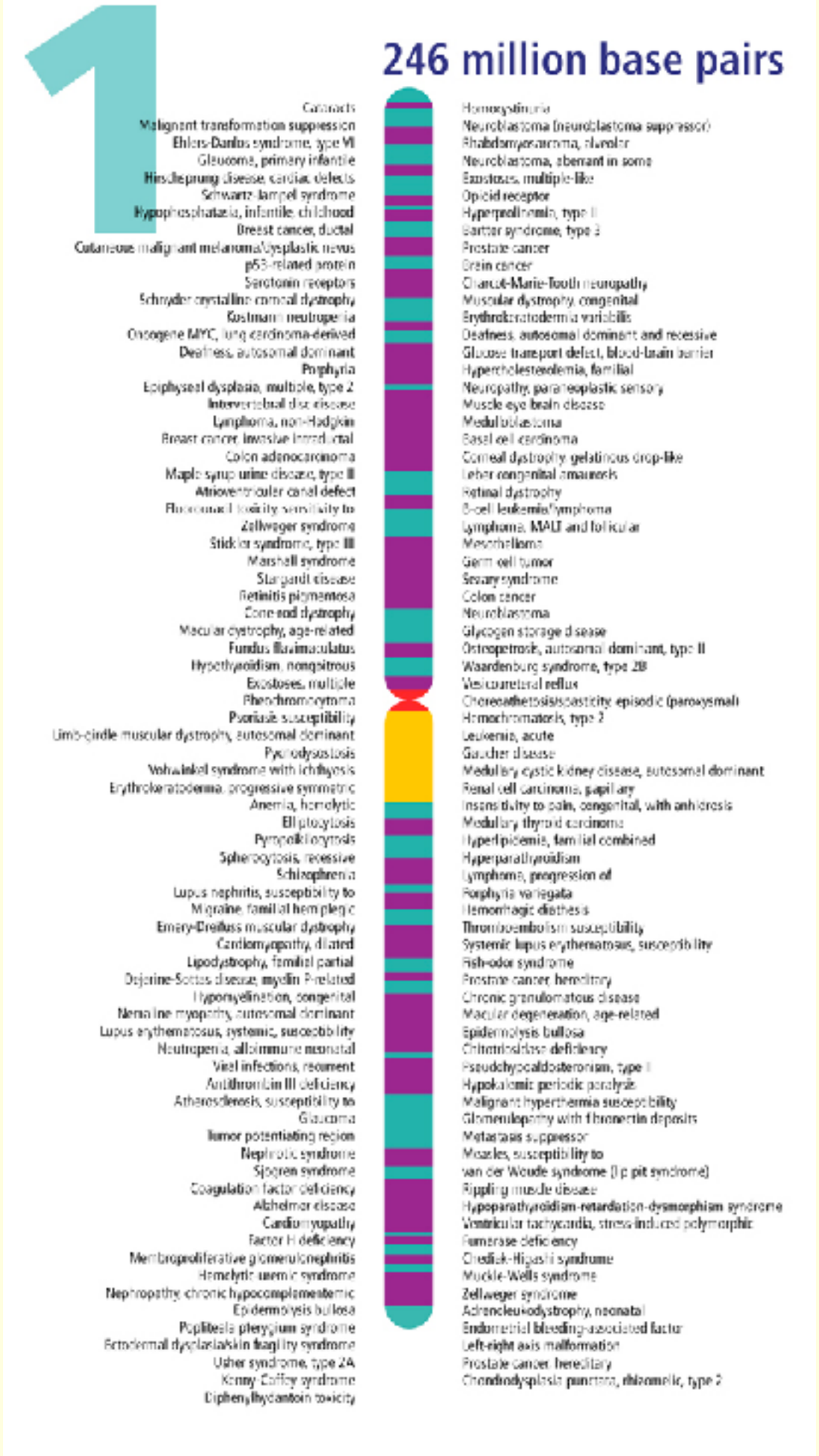
CMPS 6630: Introduction to Computational Biology and Bioinformatics

Gene Prediction

Now What?


0Mbp
50Mbp
100Mbp
150Mbp
200Mbp
250Mbp
1
3233
70
2
3592
73
3
3200
91
4
2741
46
5
2039
45
6
1570
47
7
1483
54
8
2443
35
9
1432
41
10
2041
59
11
2701
43
12
2149
66
13
1062
64
0Mbp
50Mbp
100Mbp
14
347
21
15
1552
28
16
1125
58
17
1279
42
18
1123
31
19
723
57
20
381
8
21
4
6
22
15
56
X
531
17
Y
51
34



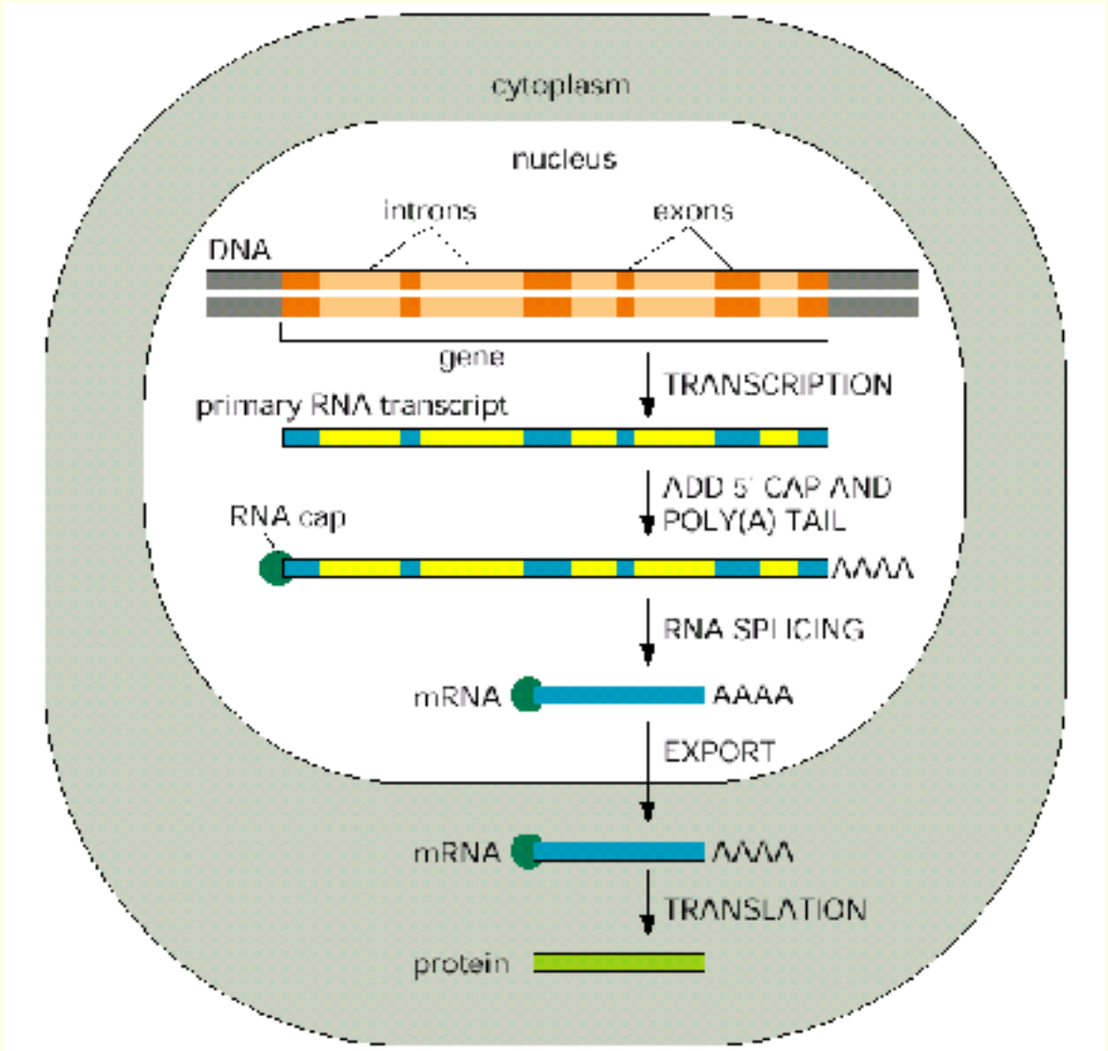
Suppose we want to annotate a genome according to genetic traits.

Given a genome, where are the genes?

Given a gene, where on the genome did it come from?

Finding Genes

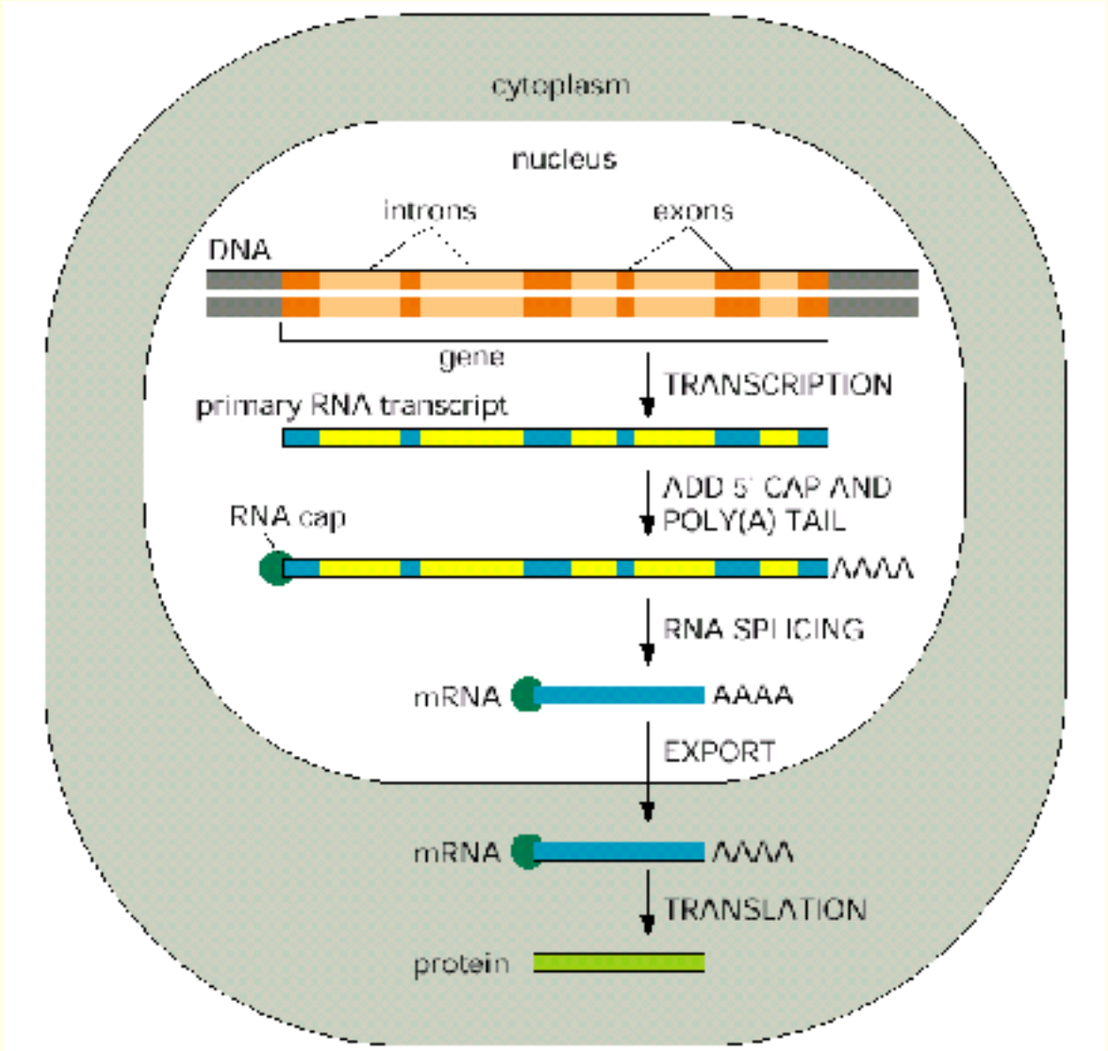
Given a strand of mRNA, can we just look for Met and “STOP” codons?



	U	C	A	G	
U	UUU = Phe UUC = Phe UUA = Leu UUG = Leu	UCU = Ser UCC = Ser UCA = Ser UCG = Ser	UAU = Tyr UAC = Tyr UAA = Stop UAG = Stop	UGU = Cys UGC = Cys UGA = Stop UGG = Trp	U C A G
C	CUU = Leu CUC = Leu CUA = Leu CUG = Leu	CCU = Pro CCC = Pro CCA = Pro CCG = Pro	CAU = His CAC = His CAA = Gln CAG = Gln	CGU = Arg CGC = Arg CGA = Arg CGG = Arg	U C A G
A	AUU = Ile AUC = Ile AUA = Ile AUG = Met	ACU = Thr ACC = Thr ACA = Thr ACG = Thr	AAU = Asn AAC = Asn AAA = Lys AAG = Lys	AGU = Ser AGC = Ser AGA = Arg AGG = Arg	U C A G
G	GUU = Val CUC = Val GUA = Val GUG = Val	GCU = Ala GCC = Ala GCA = Ala GCG = Ala	GAU = Asp GAC = Asp GAA = Glu GAG = Glu	GGU = Gly GGC = Gly GGA = Gly GGG = Gly	U C A G

Finding Genes

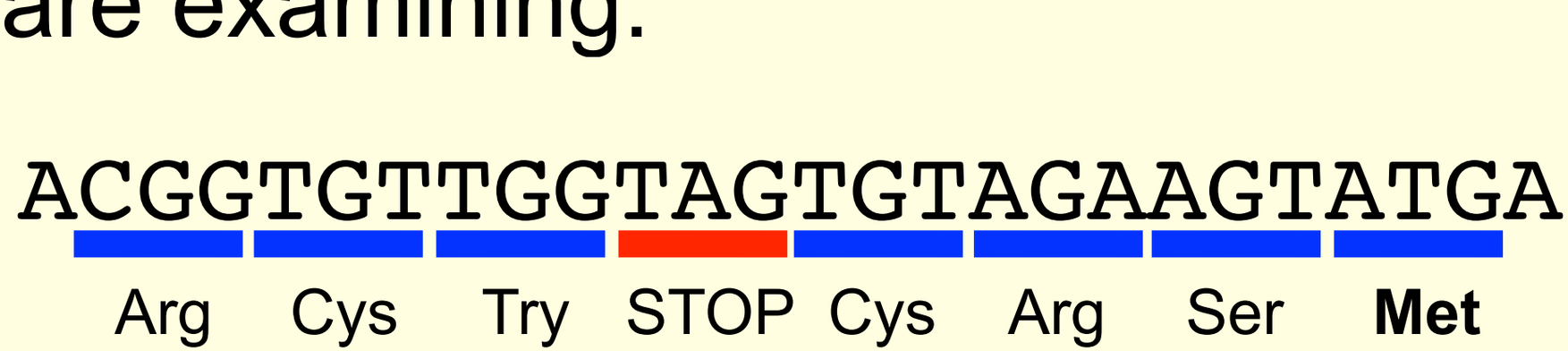
Given a strand of mRNA, can we just look for Met and “STOP” codons?



	U	C	A	G	
U	UUU = Phe UUC = Phe UUA = Leu UUG = Leu	UCU = Ser UCC = Ser UCA = Ser UCG = Ser	UAU = Tyr UAC = Tyr UAA = Stop UAG = Stop	UGU = Cys UGC = Cys UGA = Stop UGG = Trp	U C A G
C	CUU = Leu CUC = Leu CUA = Leu CUG = Leu	CCU = Pro CCC = Pro CCA = Pro CCG = Pro	CAU = His CAC = His CAA = Gln CAG = Gln	CGU = Arg CGC = Arg CGA = Arg CGG = Arg	U C A G
A	AUU = Ile AUC = Ile AUA = Ile AUG = Met	ACU = Thr ACC = Thr ACA = Thr ACG = Thr	AAU = Asn AAC = Asn AAA = Lys AAG = Lys	AGU = Ser AGC = Ser AGA = Arg AGG = Arg	U C A G
G	GUU = Val CUC = Val GUA = Val GUG = Val	GCU = Ala GCC = Ala GCA = Ala GCG = Ala	GAU = Asp GAC = Asp GAA = Glu GAG = Glu	GGU = Gly GGC = Gly GGA = Gly GGG = Gly	U C A G

Open Reading Frames

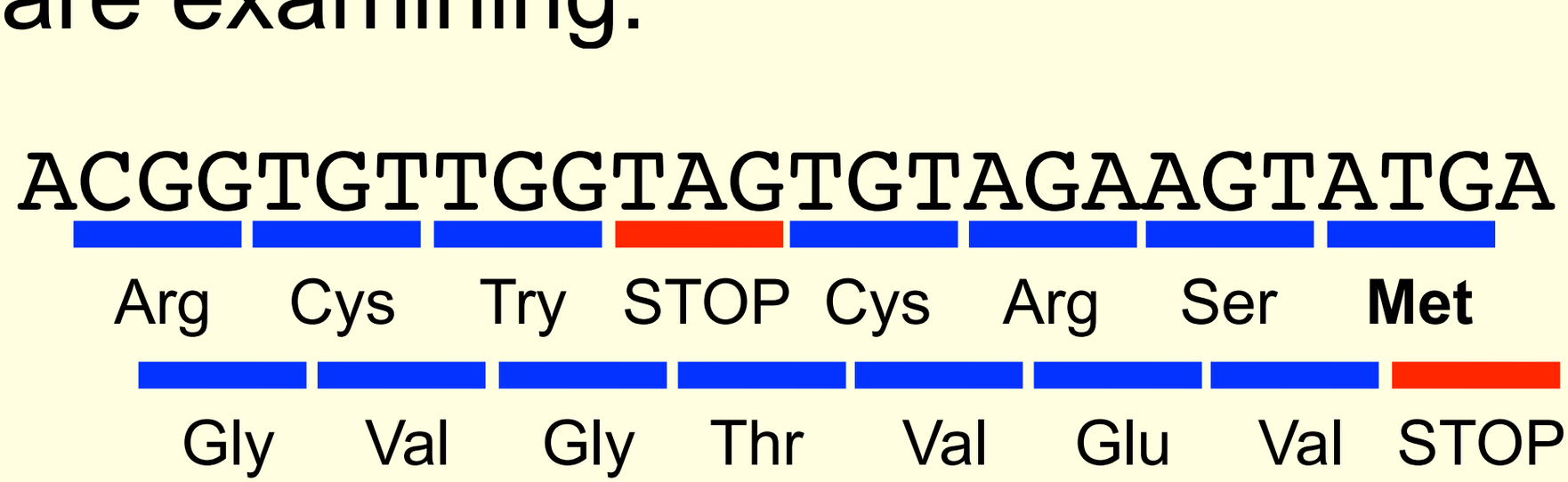
We could identify coding regions by searching for Met and STOPs. Suppose we are examining:



When is a base-triple truly a STOP codon?

Open Reading Frames

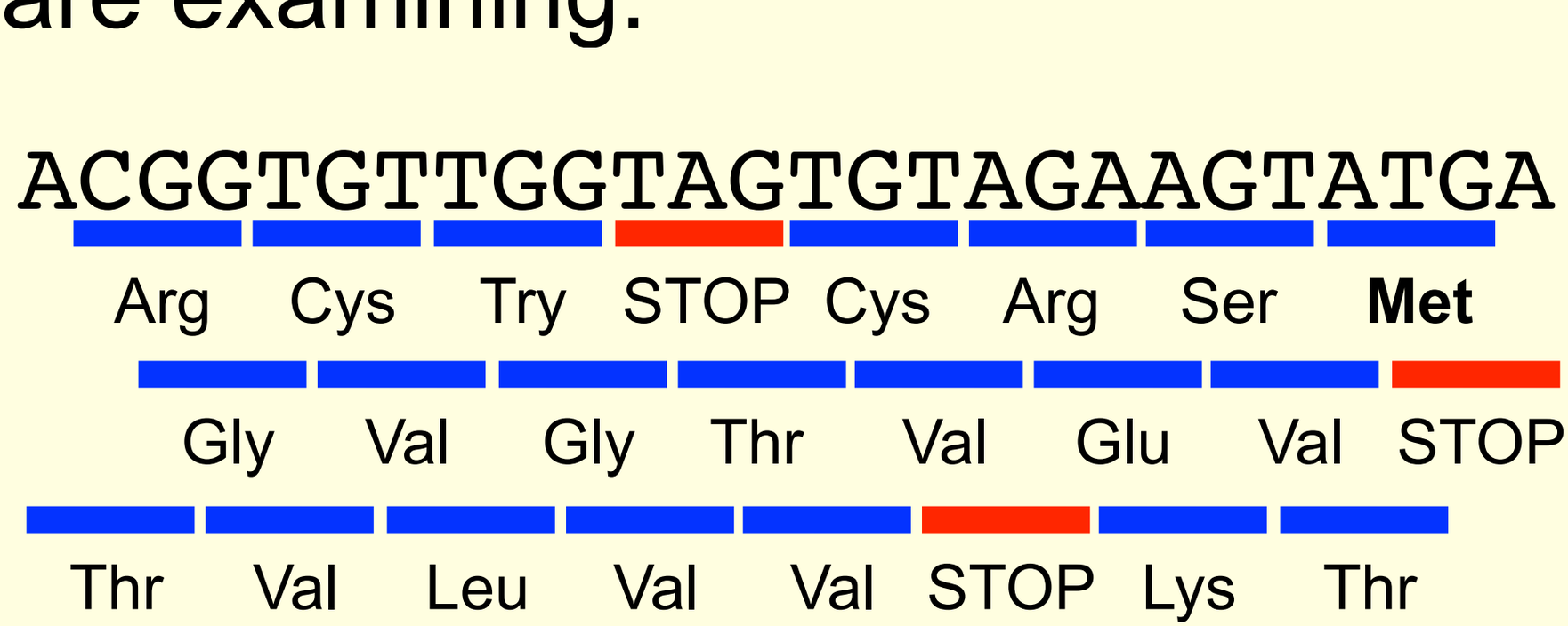
We could identify coding regions by searching for Met and STOPs. Suppose we are examining:



When is a base-triple truly a STOP codon?

Open Reading Frames

We could identify coding regions by searching for Met and STOPs. Suppose we are examining:



<u>Frame shifts</u> can change the protein sequence being coded.

“ORF” Detection

- Codons must have a functional pattern in a coding region; in a random sequence how often would we see a STOP?

- Given a window of DNA in a genome, can we assess the likelihood that it is a coding region (given a particular frameshift)?

- In known genes, Arg is 12x more likely to be coded by CGC than AGG.

CODON USAGE IN *E. COLI* GENES[1]

	Codon	Amino acid[2]	%[3]	Ratio[4]	Codon	Amino acid	%	Ratio	Codon	Amino acid	%	Ratio	Codon	Amino acid	%	Ratio	
U	**UUU**	**Phe (F)**	1.9	0.51	**UCU**	**Ser (S)**	1.1	0.19	**UAU**	**Tyr (Y)**	1.6	0.53	**UGU**	**Cys (C)**	0.4	0.43	**U**
	UUC	**Phe (F)**	1.8	0.49	**UCC**	**Ser (S)**	1.0	0.17	**UAC**	**Tyr (Y)**	1.4	0.47	**UGC**	**Cys (C)**	0.6	0.57	**C**
	UUA	**Leu (L)**	1.0	0.11	**UCA**	**Ser (S)**	0.7	0.12	**UAA**	**STOP**	0.2	0.62	**UGA**	**STOP**	0.1	0.30	**A**
	UUG	**Leu (L)**	1.1	0.11	**UCG**	**Ser (S)**	0.8	0.13	**UAG**	**STOP**	0.03	0.09	**UGG**	**Trp (W)**	1.4	1.00	**G**
C	**CUU**	**Leu (L)**	1.0	0.10	**CCU**	**Pro (P)**	0.7	0.16	**CAU**	**His (H)**	1.2	0.52	**CGU**	**Arg (R)**	2.4	0.42	**U**
	CUC	**Leu (L)**	0.9	0.10	**CCC**	**Pro (P)**	0.4	0.10	**CAC**	**His (H)**	1.1	0.48	**CGC**	**Arg (R)**	2.2	0.37	**C**
	CUA	**Leu (L)**	0.3	0.03	**CCA**	**Pro (P)**	0.8	0.20	**CAA**	**Gln (Q)**	1.3	0.31	**CGA**	**Arg (R)**	0.3	0.05	**A**
	CUG	**Leu (L)**	5.2	0.55	**CCG**	**Pro (P)**	2.4	0.55	**CAG**	**Gln (Q)**	2.9	0.69	**CGG**	**Arg (R)**	0.5	0.08	**G**
A	**AUU**	**Ile (I)**	2.7	0.47	**ACU**	**Thr (T)**	1.2	0.21	**AAU**	**Asn (N)**	1.6	0.39	**AGU**	**Ser (S)**	0.7	0.13	**U**
	AUC	**Ile (I)**	2.7	0.46	**ACC**	**Thr (T)**	2.4	0.43	**AAC**	**Asn (N)**	2.6	0.61	**AGC**	**Ser (S)**	1.5	0.27	**C**
	AUA	**Ile (I)**	0.4	0.07	**ACA**	**Thr (T)**	0.1	0.30	**AAA**	**Lys (K)**	3.8	0.76	**AGA**	**Arg (R)**	0.2	0.04	**A**
	AUG	**Met (M)**	2.6	1.00	**ACG**	**Thr (T)**	1.3	0.23	**AAG**	**Lys (K)**	1.2	0.24	**AGG**	**Arg (R)**	0.2	0.03	**G**
G	**GUU**	**Val (V)**	2.0	0.29	**GCU**	**Ala (A)**	1.8	0.19	**GAU**	**Asp (D)**	3.3	0.59	**GGU**	**Gly (G)**	2.8	0.38	**U**
	GUC	**Val (V)**	1.4	0.20	**GCC**	**Ala (A)**	2.3	0.25	**GAC**	**Asp (D)**	2.3	0.41	**GGC**	**Gly (G)**	3.0	0.40	**C**
	GUA	**Val (V)**	1.2	0.17	**GCA**	**Ala (A)**	2.1	0.22	**GAA**	**Glu (E)**	4.4	0.70	**GGA**	**Gly (G)**	0.7	0.09	**A**
	GUG	**Val (V)**	2.4	0.34	**GCG**	**Ala (A)**	3.2	0.34	**GAG**	**Glu (E)**	1.9	0.30	**GGG**	**Gly (G)**	0.9	0.13	**G**
	U				**C**				**A**				**G**				

[1] The data shown in this table is from the Arabidopsis Research Companion on the World Wide Web (//weeds/mgh.harvard.edu). Codon frequencies for many other bacteria can be found at http://morgan.angis.su.oz.au/Angis/Tables.html.

[2] The letter in parenthesis represents the one-letter code for the amino acid.

[3] % represents the average frequency this codon is used per 100 codons.

[4] Ratio represents the abundance of that codon relative to all of the codons for that particular amino acid.

Using known mRNA, we can compute the likelihood that a stretch of DNA is coding (given the frame shift).

	U			C			A			G	
U	UUU	Phe	57	UCU	Ser	16	UAU	Tyr	58	UGU	Cys	45
	UUC	Phe	43	UCC	Ser	15	UAC	Tyr	42	UGC	Cys	55
	UUA	Leu	13	UCA	Ser	13	UAA	Stp	62	UGA	Stp	30
	UUG	Leu	13	UCG	Ser	15	UAG	Stp	8	UGG	Trp	100
C	CUU	Leu	11	CCU	Pro	17	CAU	His	57	CGU	Arg	37
	CUC	Leu	10	CCC	Pro	17	CAC	His	43	CGC	Arg	38
	CUA	Leu	4	CCA	Pro	20	CAA	Gln	45	CGA	Arg	7
	CUG	Leu	49	CCG	Pro	51	CAG	Gln	66	CGG	Arg	10
A	AUU	Ile	50	ACU	Thr	18	AAU	Asn	46	AGU	Ser	15
	AUC	Ile	41	ACC	Thr	42	AAC	Asn	54	AGC	Ser	26
	AUA	Ile	9	ACA	Thr	15	AAA	Lys	75	AGA	Arg	5
	AUG	Met	100	ACG	Thr	26	AAG	Lys	25	AGG	Arg	3
G	GUU	Val	27	GCU	Ala	17	GAU	Asp	63	GGU	Gly	34
	GUC	Val	21	GCC	Ala	27	GAC	Asp	37	GGC	Gly	39
	GUA	Val	16	GCA	Ala	22	GAA	Glu	68	GGA	Gly	12
	GUG	Val	36	GCG	Ala	34	GAG	Glu	32	GGG	Gly	15

Using known mRNA, we can compute the likelihood that a stretch of DNA is coding (given the frame shift).

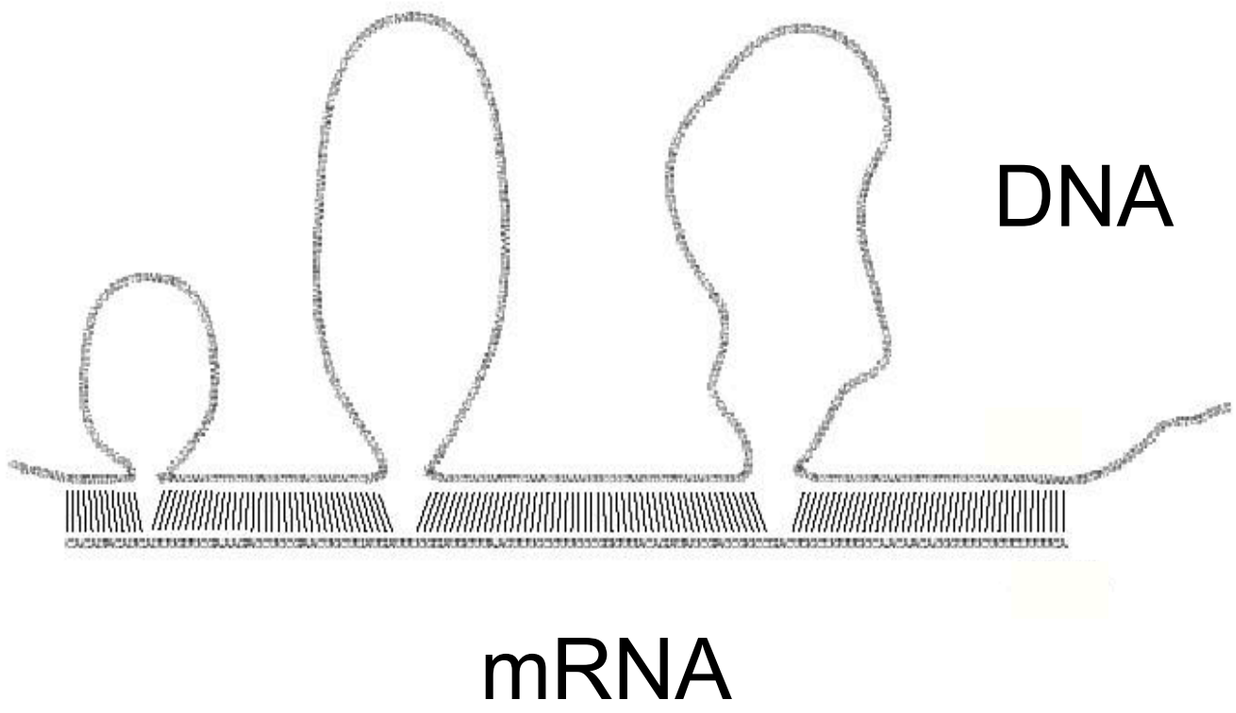
Codon Statistics

- Suppose we have known codon frequencies $F^* = f_1^*, f_2^*, \ldots, f_{61}^*$.

- For our unknown sequence, calculate codon usages F_0, F_1, F_2 for each possible frame shift.

- Compute $\arg\max_i \delta(F_i, F^*)$, for an appropriately chosen cost function δ (Euclidean, KL-distance, etc).

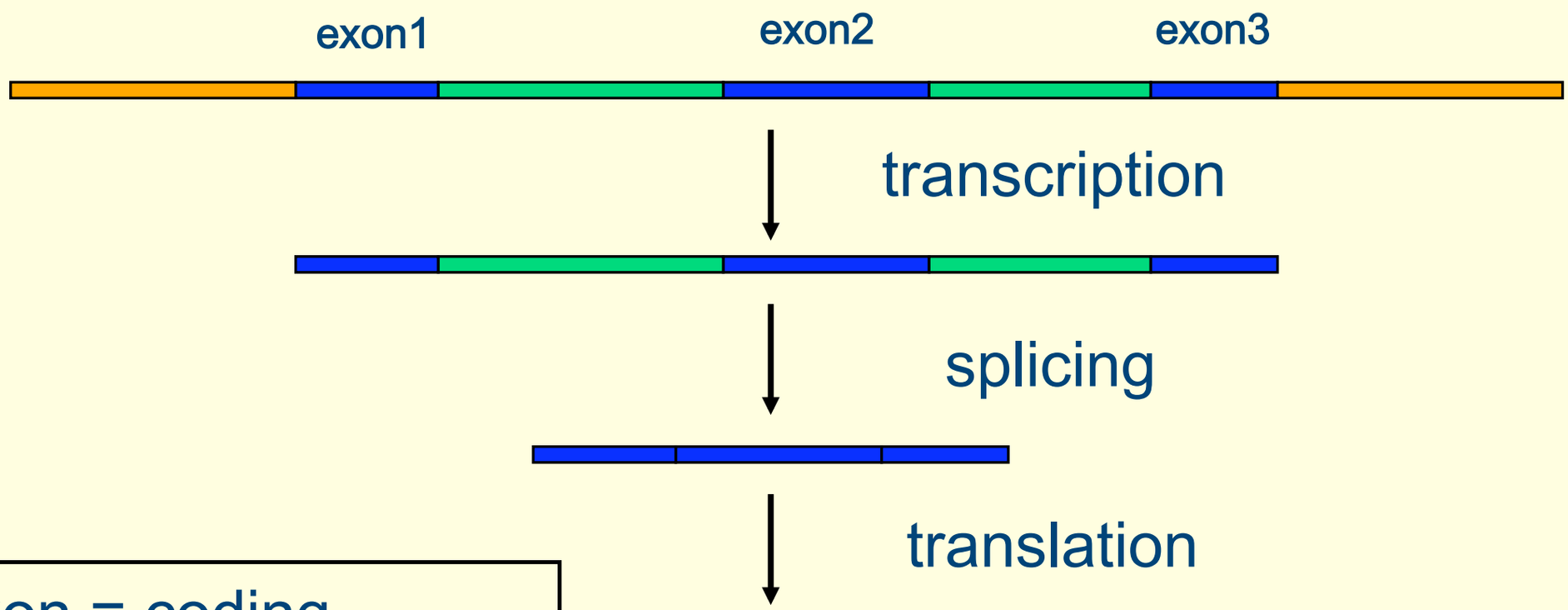
Gene Splicing



Sharp and Roberts (1977) hybridized the mRNA for a viral protein to its corresponding "gene" and showed that transcription can be "spliced".

So given a genomic sequence, we need to identify fragmented exonic components (with or without mRNA).

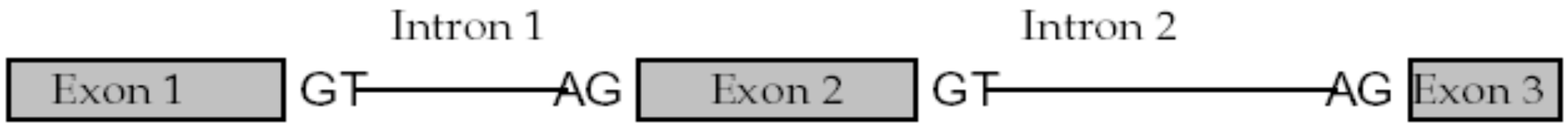
Introns and Exons



exon = coding
intron = non-coding

About 5% of a genomic sequence is exonic, while the rest is intronic (some say it is “junk”). Prokaryotes don’t have exons!

Splicing Signals

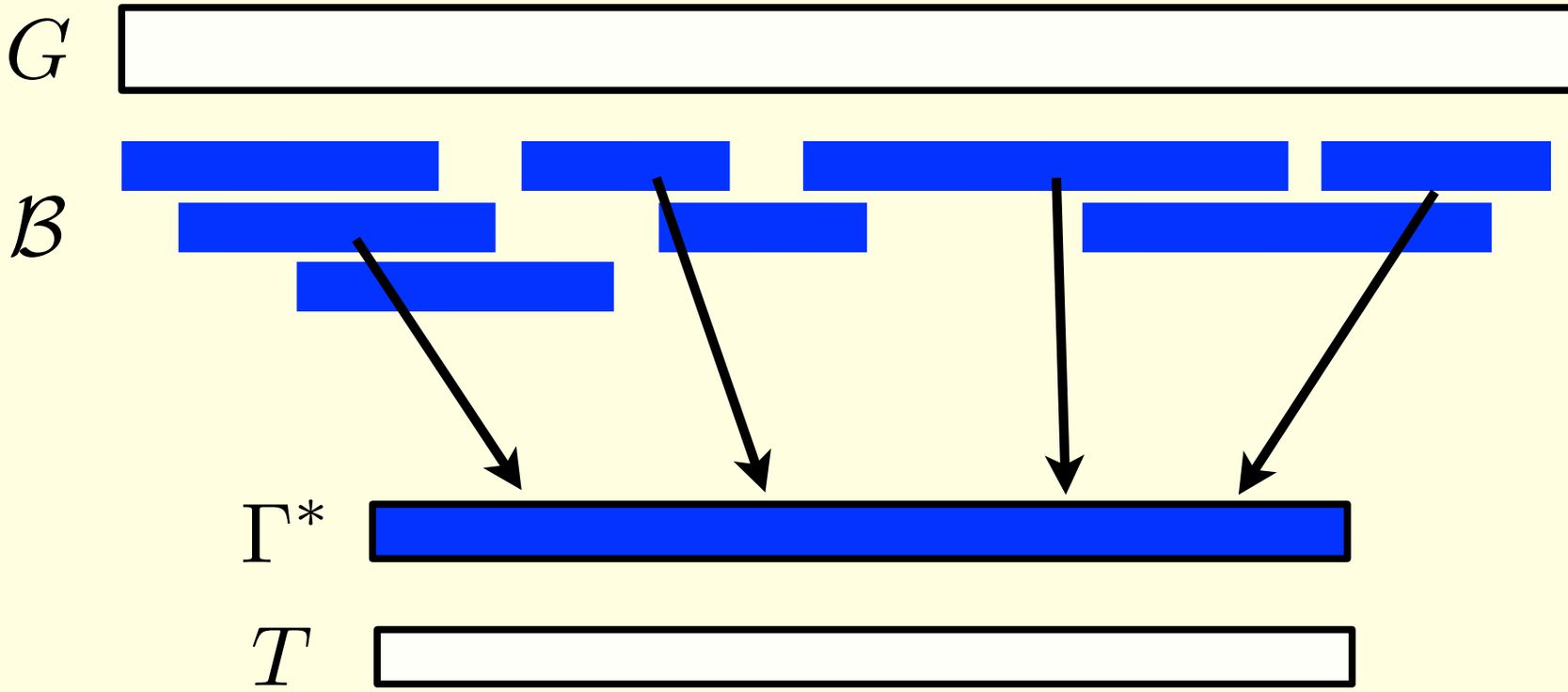


We can attempt to perform a “spliced alignment” by using a known homologous gene.

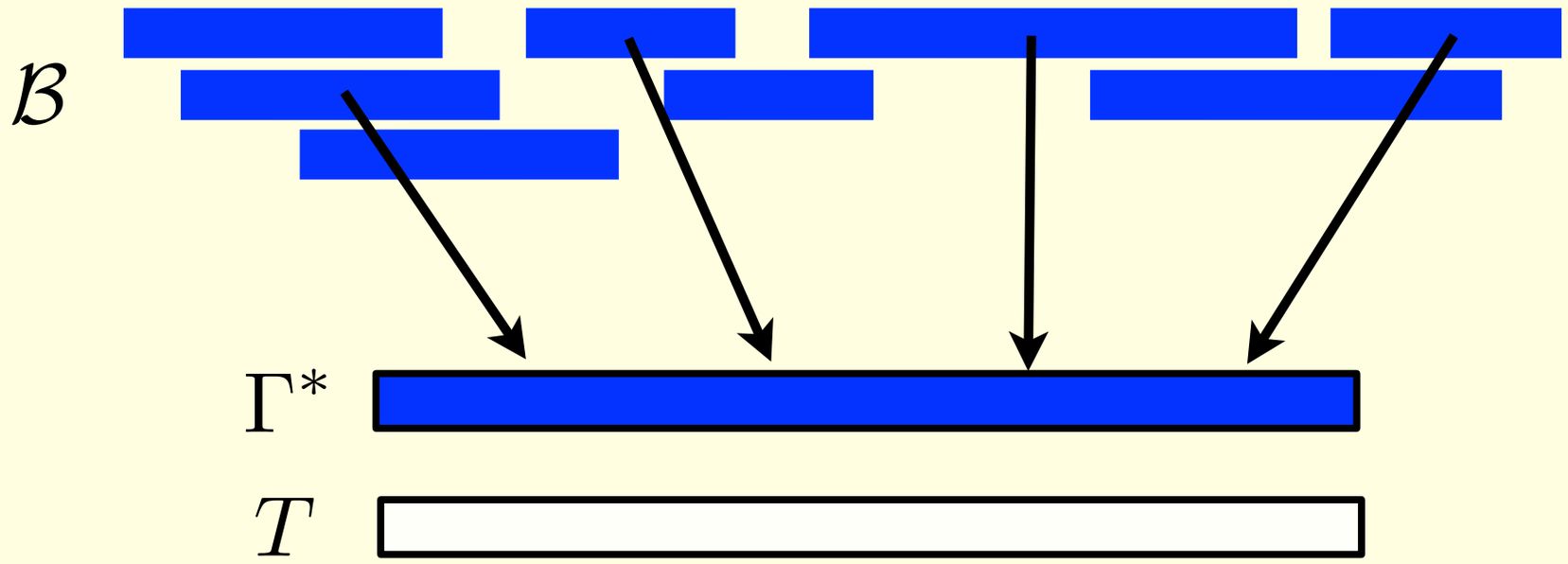
Statistical methods for gene detection attempt to detect the “transition” between splice sites by comparing the distributions of codons on either side of an AG or GT pair.

Spliced Alignment

Given a DNA sequence G, a target sequence T, and a candidate set of exons $\mathcal{B}$, which chain of non-overlapping exons Γ^* maximizes the alignment score $cost(\Gamma^*, T)$.



Spliced Alignment



Suppose we had a candidate chain Γ ending in a block B.

.

We want: $\Gamma^* = \arg\max_B S(length(B), length(T), B)$

We can find the optimal spliced alignment using dynamic programming. (How quickly?)

Gelfand *et al.*

$$S(i, j, k) = \max_{\text{all chains } \Gamma \text{ containing block } B_k} s(\Gamma^*(i), T(j)).$$

$S(i, j, k)$ can be easily computed by dynamic programming as described below.

Let $\mathcal{B}(i) = \{k: last(k) < i\}$ be the set of blocks ending (strictly) before position i in G. The following recurrence computes $S(i, j, k)$ for $1 \leq i \leq n$, $1 \leq j \leq m$, and $1 \leq k \leq b$:

$$S(i,j,k) = \max \begin{cases} S(i-1,j-1,k) + \Delta(g_i, t_j), & \text{if } i \neq first(k) \\ S(i-1,j,k) + \Delta_{indel}, & \text{if } i \neq first(k) \\ \max_{l \in \mathcal{B}(first(k))} S(last(l), j-1, l) + \Delta(g_i, t_j), & \text{if } i = first(k) \\ \max_{l \in \mathcal{B}(first(k))} S(last(l), j, l) + \Delta_{indel}, & \text{if } i = first(k) \\ S(i,j-1,k) + \Delta_{indel}. \end{cases}$$

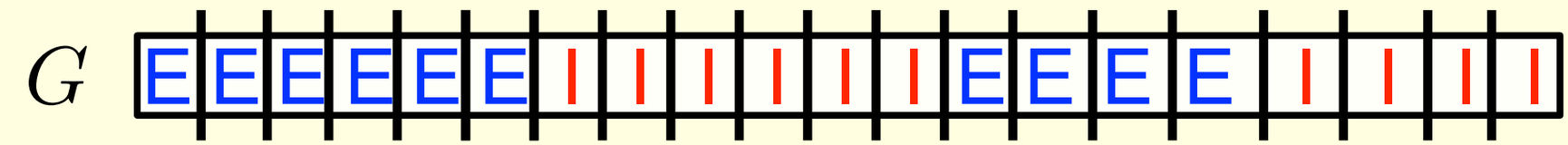
[1]

After computing the three-dimensional table $S(i,j,k)$, the score of the optimal spliced alignment can be found as

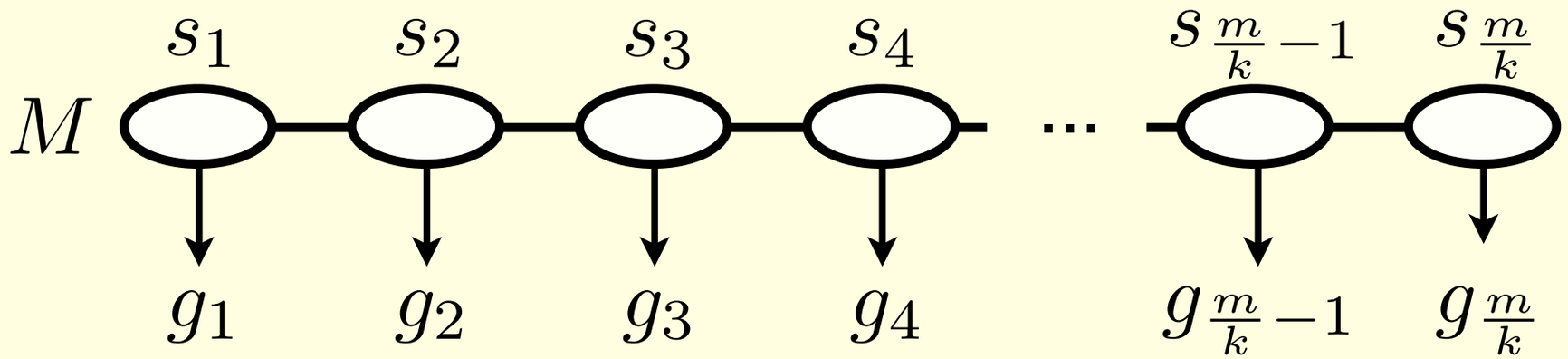
$$\max_k S(last(k), m, k).$$

Hidden Markov Models

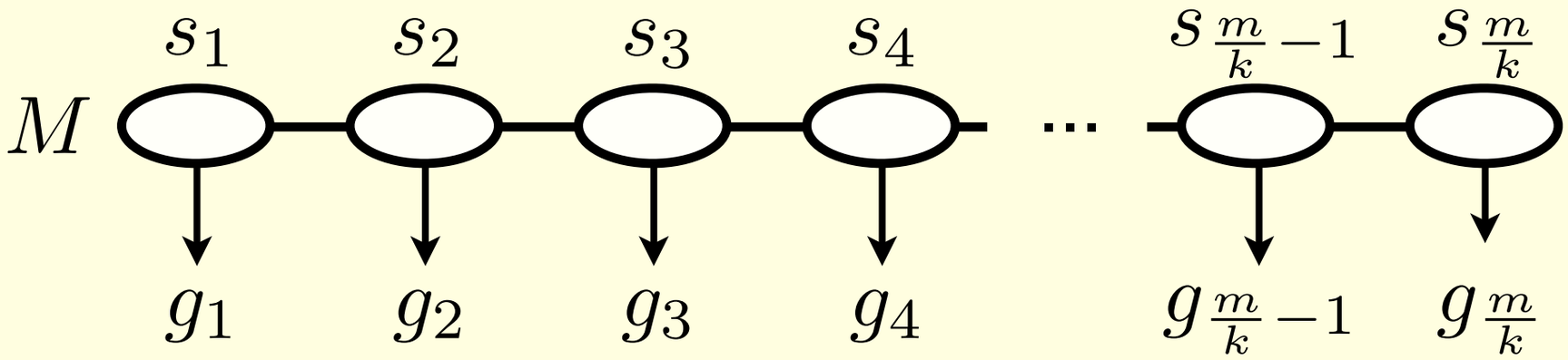
Suppose that we associate blocks of length k in G with two possible states, “intron” and “exon”.



Starting with a prior and conditional likelihoods for each block, can we find the most likely set of exons for G?



Hidden Markov Models



We are given $\Pr[s_i]$ and $\Pr[s_i \mid s_{i-1}]$ as input.

Our goal is to find $\arg \max_{(s_1, s_2, \ldots, s_{\frac{m}{k}})} \Pr[s_1, s_2, \ldots, s_{\frac{m}{k}}]$

This can be done using the Viterbi algorithm, which is essentially a dynamic programming method.

Combinatorial Gene Regulation

- A differential gene expression (e.g., microarray, HTS) experiment showed that when gene X is knocked out, 20 other genes are not expressed

- **How can one gene have such drastic effects?**

Regulatory Proteins

- Gene X encodes regulatory protein, a.k.a. a ***transcription factor*** (TF)

- The 20 unexpressed genes rely on gene X's TF to induce transcription

- A single TF may regulate multiple genes

Regulatory Regions

- Every gene contains a regulatory region (RR) typically stretching 100-1000 bp upstream of the transcriptional start site

- Located within the RR are the ***Transcription Factor Binding Sites*** (TFBS), also known as ***motifs***, specific for a given transcription factor

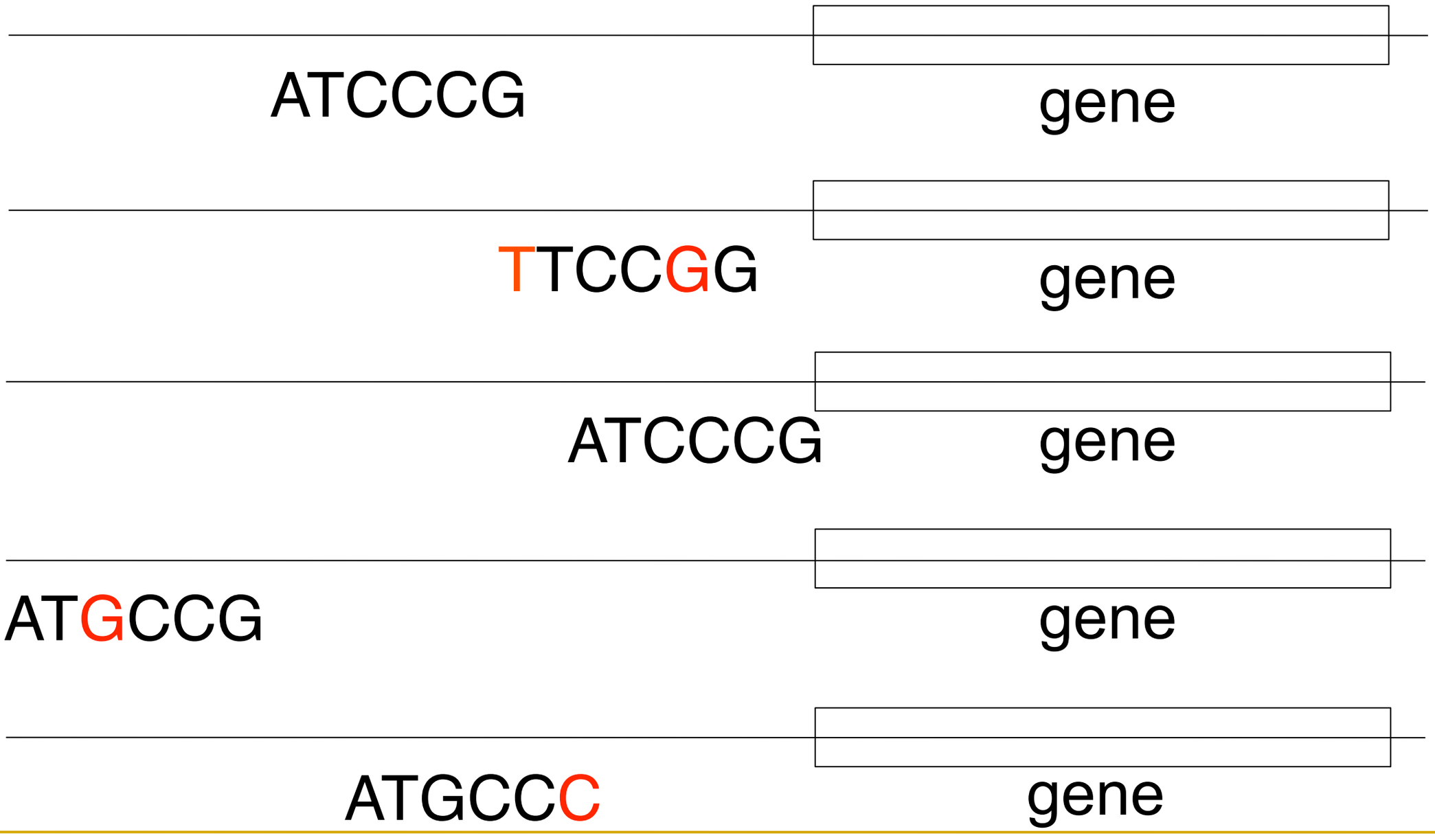
- TFs influence gene expression by binding to a specific location in the respective gene's regulatory region - TFBS

Transcription Factor Binding Sites

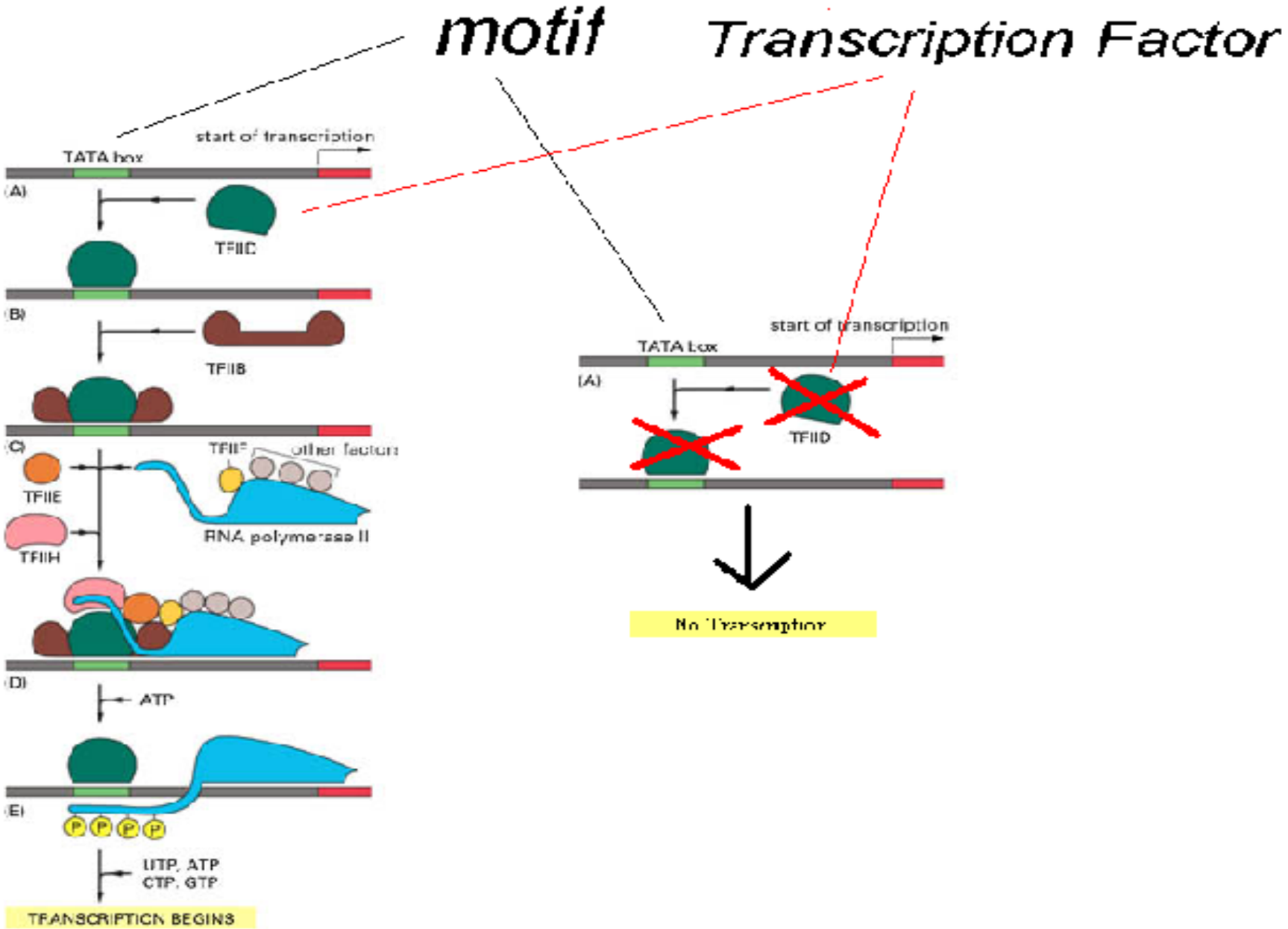
- A TFBS can be located anywhere within the Regulatory Region.

- TFBS may vary slightly across different regulatory regions since non-essential bases could mutate

Motifs and Transcriptional Start Sites



Transcription Factors and Motifs



Motif Logo

- Motifs can mutate on non important bases
- The five motifs in five different genes have mutations in position 3 and 5
- Representations called *motif logos* illustrate the conserved and variable regions of a motif

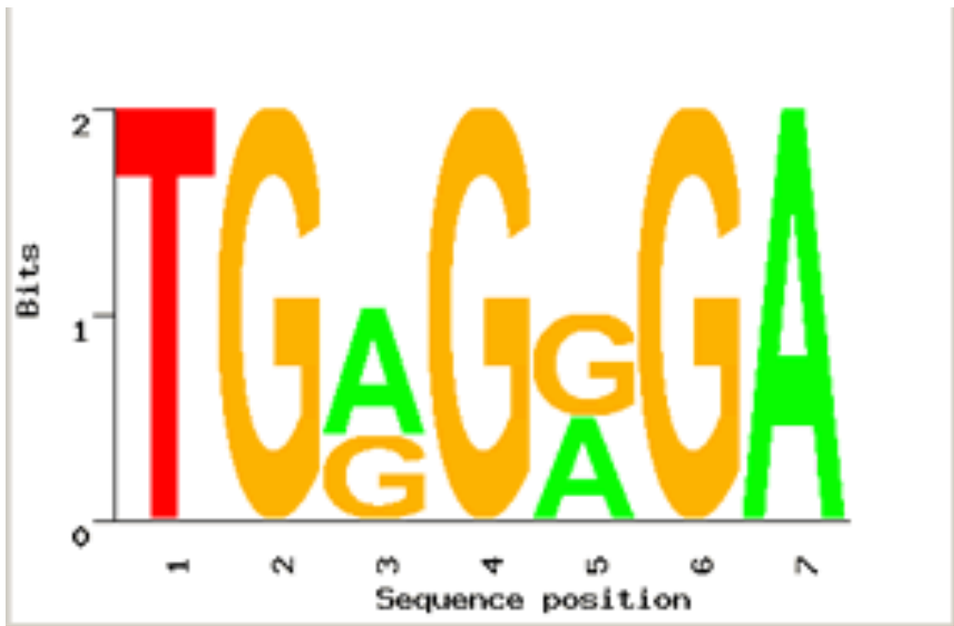
TGGGGGA

TGAGAGA

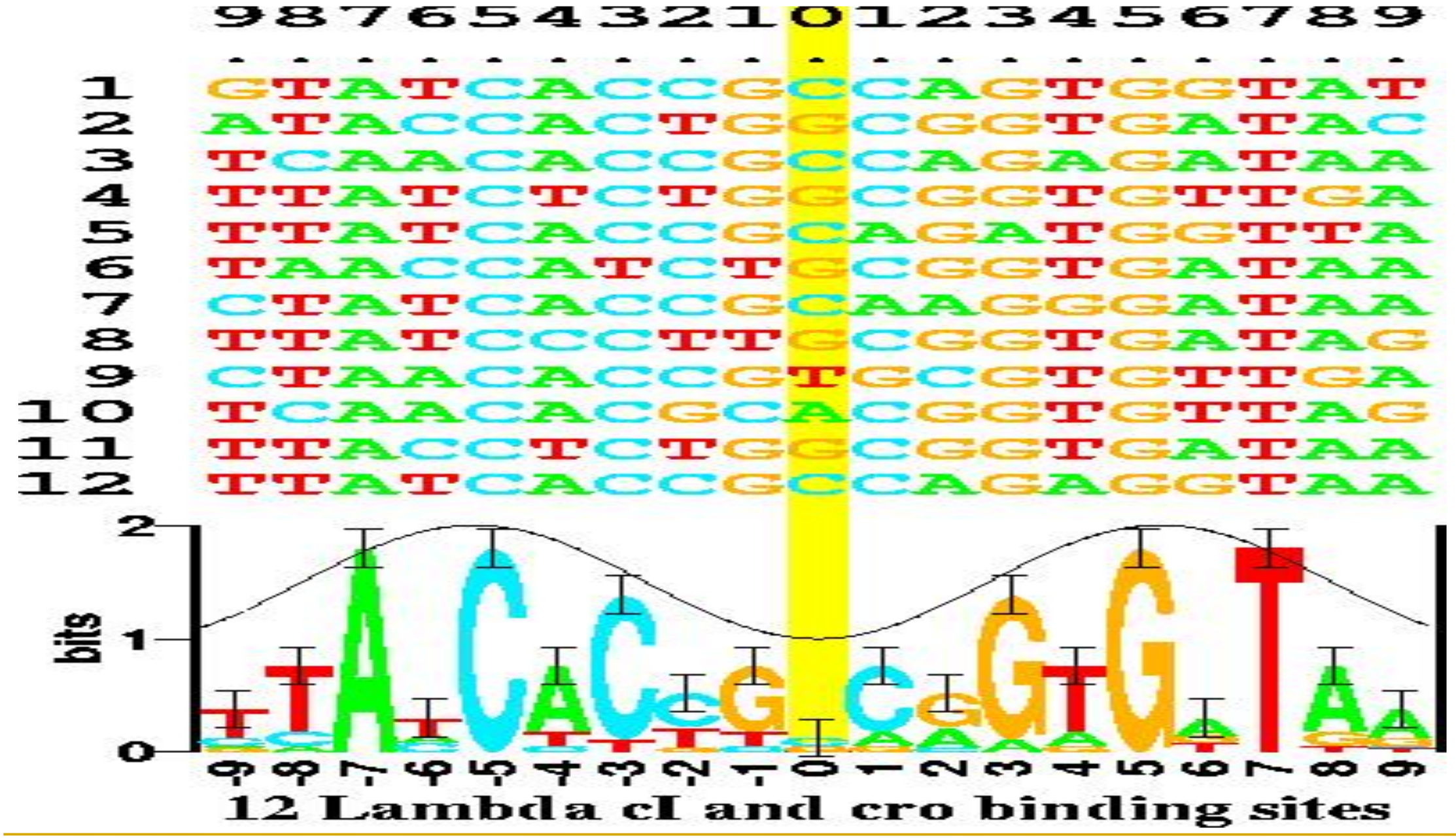
TGGGGGA

TGAGAGA

TGAGGGA



Motif Logos: An Example



(http://www-lmmb.ncifcrf.gov/~toms/sequencelogo.html)

Identifying Motifs

- Genes are turned on or off by regulatory proteins
- These proteins bind to upstream regulatory regions of genes to either attract or block an RNA polymerase
- Regulatory protein (TF) binds to a short DNA sequence called a motif (TFBS)
- So finding the same motif in multiple genes' regulatory regions suggests a regulatory relationship amongst those genes

Identifying Motifs: Complications

- We do not know the motif sequence
- We do not know where it is located relative to the genes start
- Motifs can differ slightly from one gene to the next
- How to discern it from “random” motifs?

DNA Motifs

Small conserved regions of DNA can regulate transcription, but how do we find them?

```
CCTGATAGACGCTATCTGGCTATCCACGTACGTAGGTCCTCTGTGCGAATCTATGCGT
AGTACTGGTGTACATTTGATACGTACGTACACCGGCAACCTGAAACAAACGCTCAGAA
AAACGTACGTGCACCCTCTTTCTTCGTGGCTCTGGCCAACGAGGGCTGATGTATAAGA
GTAAGTCATAGCTGTAACTATTACCTGCCACCCCTATTACATCTTACGTACGTATACA
ACGCGTCATGGCGGGGTATGCGTTTTGGTCGTCGTACGCTCGATCGTTAACGTACGTC
```

Given a set of n sequences, can we find a shared substring of length k ?

DNA Motifs

Small conserved regions of DNA can regulate transcription, but how do we find them?

CCTGATAGACGCTATCTGGCTATCC**ACGTACGT**AGGTCCTCTGTGCGAATCTATGCGT
AGTACTGGTGTACATTTGAT**ACGTACGT**ACACCGGCAACCTGAAACAAACGCTCAGAA
AA**ACGTACGT**GCACCCTCTTTCTTCGTGGCTCTGGCCAACGAGGGCTGATGTATAAGA
GTAAGTCATAGCTGTAACTATTACCTGCCACCCCTATTACATCTT**ACGTACGT**ATACA
ACGCGTCATGGCGGGGTATGCGTTTTGGTCGTCGTACGCTCGATCGTTA**ACGTACGT**C

ACGTACGT

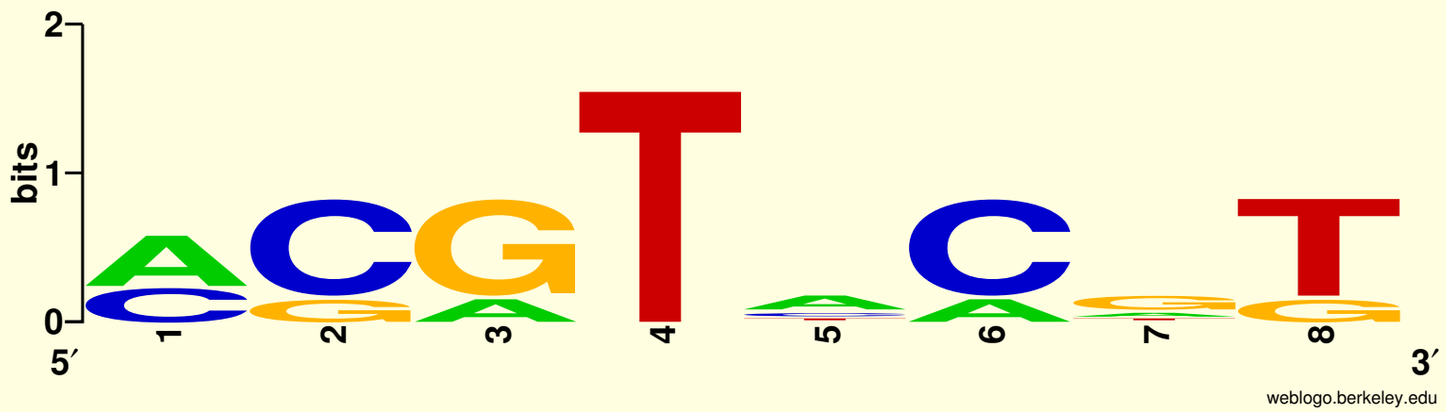
Given a set of n sequences, can we find a shared substring of length k ?

DNA Motifs

Small conserved regions of DNA can regulate transcription, but how do we find them?

CCTGATAGACGCTATCTGGCTATCC**AGGTACTT**AGGTCCTCTGTGCGAATCTATGCGT
AGTACTGGTGTACATTTGAT**CCATACGT**ACACCGGCAACCTGAAACAAACGCTCAGAA
AA**ACGTTAGT**GCACCCTCTTTCTTCGTGGCTCTGGCCAACGAGGGCTGATGTATAAGA
GTAAGTCATAGCTGTAACTATTACCTGCCACCCCTATTACATCTT**ACGTCCAT**ATACA
ACGCGTCATGGCGGGGTATGCGTTTTGGTCGTCGTACGCTCGATCGTTA**CCGTACGG**C

AGGTACTT
CCATACGT
ACGTTAGT
ACGTCCAT
CCGTACGG

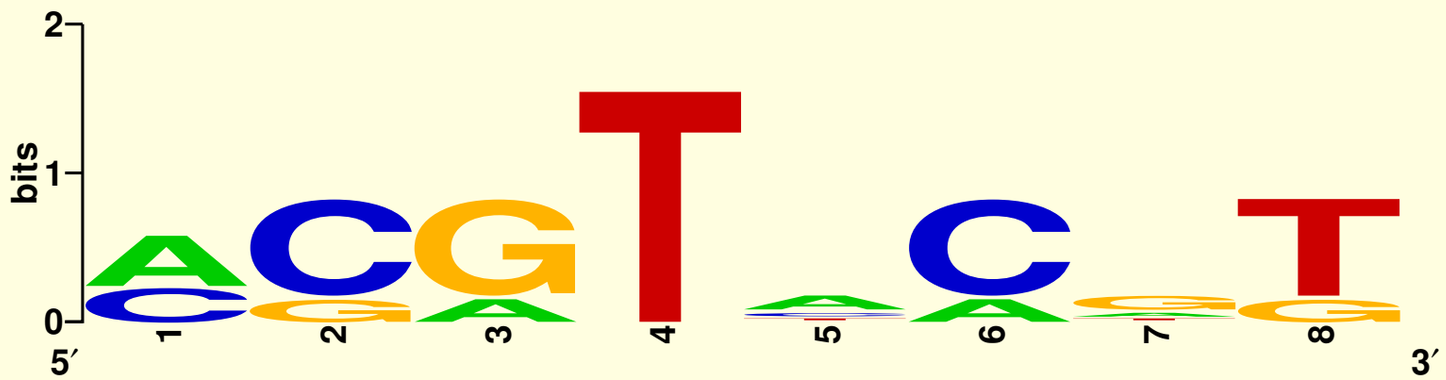


DNA Motifs

Given a set of n sequences of length m,
what is the <u>consensus</u> substring of length k ?

CCTGATAGACGCTATCTGGCTATCCAGGTACTTAGGTCCTCTGTGCGAATCTATGCGT
AGTACTGGTGTACATTTGATCCATACGTACACCGGCAACCTGAAACAAACGCTCAGAA
AAACGTTAGTGCACCCTCTTTCTTCGTGGCTCTGGCCAACGAGGGCTGATGTATAAGA
GTAAGTCATAGCTGTAACTATTACCTGCCACCCCTATTACATCTTACGTCCATATACA
ACGCGTCATGGCGGGGTATGCGTTTTGGTCGTCGTACGCTCGATCGTTACCGTACGGC

AGGTACTT
CCATACGT
ACGTTAGT
ACGTCCAT
CCGTACGG



DNA Motifs

Given a set of n sequences of length m, what is the consensus substring of length k ?

If we knew where each motif started, then we just need to compute:

$$score(s_1, s_2, \ldots, s_n) = \sum_{i=0}^{k-1} best(\{g_j[s_j + i] \mid j = 1, 2, \ldots, n\})$$

What if we tried all possible starting points?

$(m - k)^n$ possible pairings of substrings!

A Motif Finding Analogy

- The Motif Finding Problem is similar to the problem posed by Edgar Allan Poe (1809 – 1849) in his *Gold Bug* story

The Gold Bug Problem

- Given a secret message:

```
53++!305))6*;4826)4+.)4+);806*;48!8`60))85;]8*:+*8!83(88)5*!;
46(;88*96*?;8)*+(;485);5*!2:*+(;4956*2(5*-4)8`8*; 4069285);)6
!8)4++;1(+9;48081;8:8+1;48!85;4)485!528806*81(+9;48;(88;4(+?3
4;48)4+;161;:188;+?;
```

- Decipher the message encrypted in the fragment

Hints for The Gold Bug Problem

- Additional hints:
  - The encrypted message is in English
  - Each symbol correspond to one letter in the English alphabet
  - No punctuation marks are encoded

The Gold Bug Problem: Symbol Counts

- Naive approach to solving the problem:
  - Count the frequency of each symbol in the encrypted message
  - Find the frequency of each letter in the alphabet in the English language
  - Compare the frequencies of the previous steps, try to find a correlation and map the symbols to a letter in the alphabet

Symbol Frequencies in the Gold Bug Message

- **Gold Bug Message**:

Symbol	8	;	4	)	+	*	5	6	(	!	1	0	2	9	3	:	?	`	-	]	.
Frequency	34	25	19	16	15	14	12	11	9	8	7	6	5	5	4	4	3	2	1	1	1

- **English Language**:

e t a o i n s r h l d c u m f p g w y b v k x j q z

Most frequent → Least frequent

The Gold Bug Message Decoding: First Attempt

- By simply mapping the most frequent symbols to the most frequent letters of the alphabet:

```
sfiilfcsoorntaeuroaikoaiotecrntaeleyrcooestvenpinelefheeosnlt
arhteenmrnwteonihtaesotsnlupnihtamsrnuhsnbaoeyentacrmuesotorl
eoaiitdhimtaecedtepeidtaelestaoaeslsueecrnedhimtaetheetahiwfa
taeoaitdrdtpdeetiwt
```

- The result does not make sense

The Gold Bug Problem: l-tuple count

- A better approach:
  - Examine frequencies of *l*-tuples, combinations of 2 symbols, 3 symbols, etc.
  - "The" is the most frequent 3-tuple in English and ";48" is the most frequent 3-tuple in the encrypted text
  - Make inferences of unknown symbols by examining other frequent *l*-tuples

The Gold Bug Problem: the ;48 clue

- Mapping "the" to ";48" and substituting all occurrences of the symbols:

```
53++!305))6*the26)h+.)h+)te06*the!e`60))e5t]e*:+*e!e3(ee)5*!t
h6(tee*96*?te)*+(the5)t5*!2:*+(th956*2(5*h)e`e*th0692e5)t)6!e
)h++t1(+9the0e1te:e+1the!e5th)he5!52ee06*e1(+9thet(eeth(+?3ht
he)h+t161t:1eet+?t
```

The Gold Bug Message Decoding: Second Attempt

- Make inferences:

```
53++!305))6*the26)h+.)h+)te06*the!e`60))e5t]e*:+*e!e3(ee)5*!t
h6(tee*96*?te)*+(the5)t5*!2:*+(th956*2(5*h)e`e*th0692e5)t)6!e
)h++t1(+9the0e1te:e+1the!e5th)he5!52ee06*e1(+9thet(eeth(+?3ht
he)h+t161t:1eet+?t
```

- "thet(ee" most likely means "the tree"
  - Infer "(" = "r"

- "th(+?3h" becomes "thr+?3h"
  - Can we guess "+" and "?"?

The Gold Bug Problem: The Solution

- After figuring out all the mappings, the final message is:

AGOODGLASSINTHEBISHOPSHOSTELINTHEDEVILSSEATWENYONEDEGRE
ESANDTHIRTEENMINUTESNORTHEASTANDBYNORTHMAINBRANCHSEVENT
HLIMBEASTSIDESHOOTFROMTHELEFTEYEOFTHEDEATHSHEADABEELINE

FROMTHETREETHROUGHTHESHOTFIFTYFEETOUT

The Solution (cont'd)

- Punctuation is important:

A GOOD GLASS IN THE BISHOP'S HOSTEL IN THE DEVIL'S SEA,
TWENY ONE DEGREES AND THIRTEEN MINUTES NORTHEAST AND BY NORTH,
MAIN BRANCH SEVENTH LIMB, EAST SIDE, SHOOT FROM THE LEFT EYE OF
THE DEATH'S HEAD A BEE LINE FROM THE TREE THROUGH THE SHOT,
FIFTY FEET OUT.

Solving the Gold Bug Problem

- Prerequisites to solve the problem:
  - Need to know the relative frequencies of single letters, and combinations of two and three letters in English
  - Knowledge of all the words in the English dictionary is highly desired to make accurate inferences

DNA Motifs

Given a set of n sequences of length m, what is the consensus substring of length k ?

If we knew where each motif started, then we just need to compute:

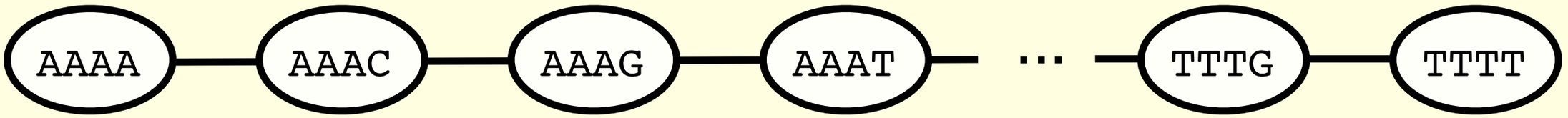
$$score(s_1, s_2, \ldots, s_n) = \sum_{i=0}^{k-1} best(\{g_j[s_j + i] \mid j = 1, 2, \ldots, n\})$$

What if we tried all possible k-mers?

$4^k \cdot n(m-k)$ comparisons

Branch and Bound

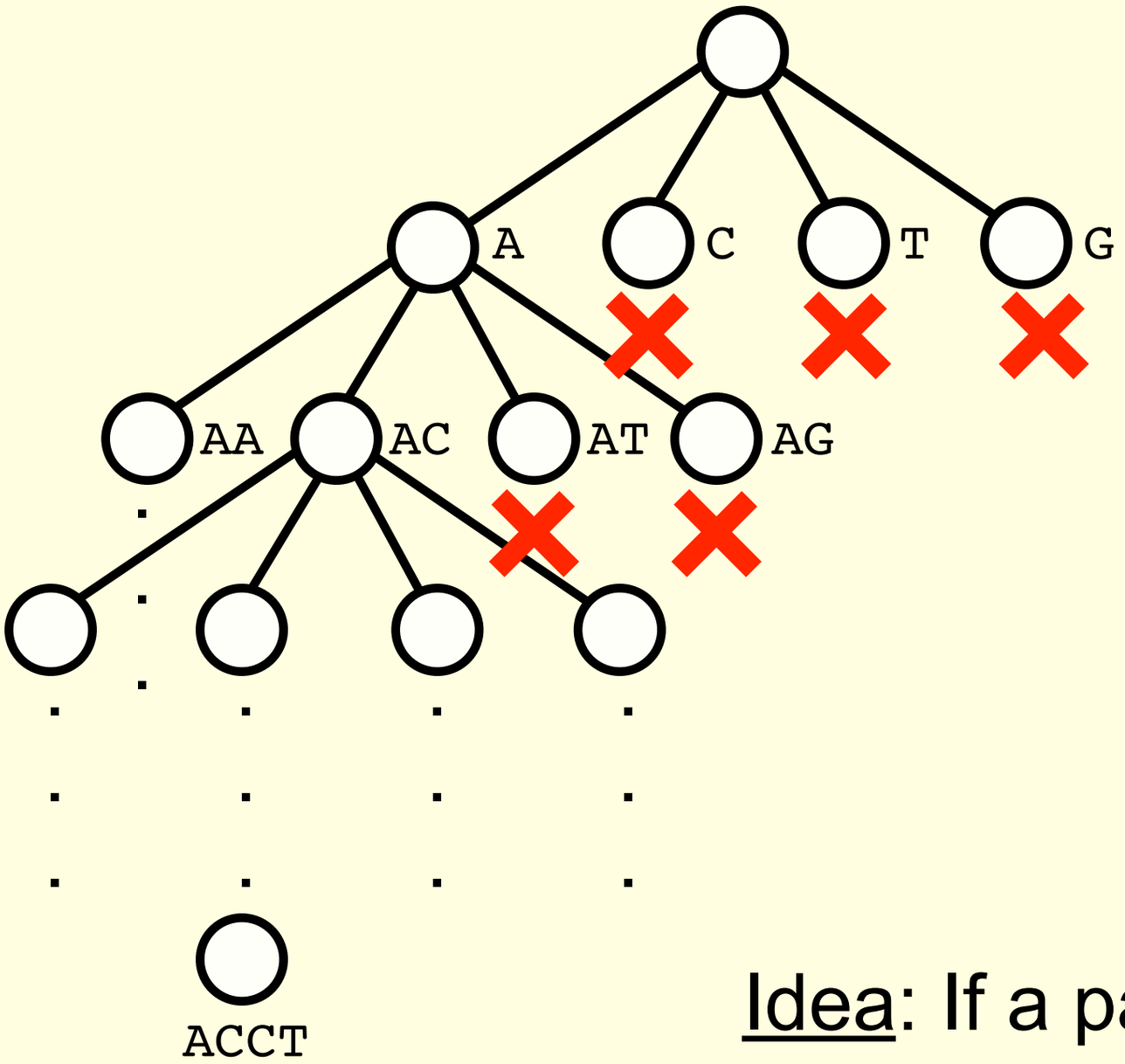
Can we improve our sequential search strategy?



What if we looked at shorter segments?

If a k-mer has a “bad” prefix, then we can eliminate all k-mers with that prefix.

Branch and Bound



We can draw the search as a tree where each level corresponds to a prefix of the k-mer we want.

We must establish bounds on the “score” of a motif given its prefix.

Idea: If a particular k-mer cannot be improved upon by a different prefix, eliminate that subtree from the search.