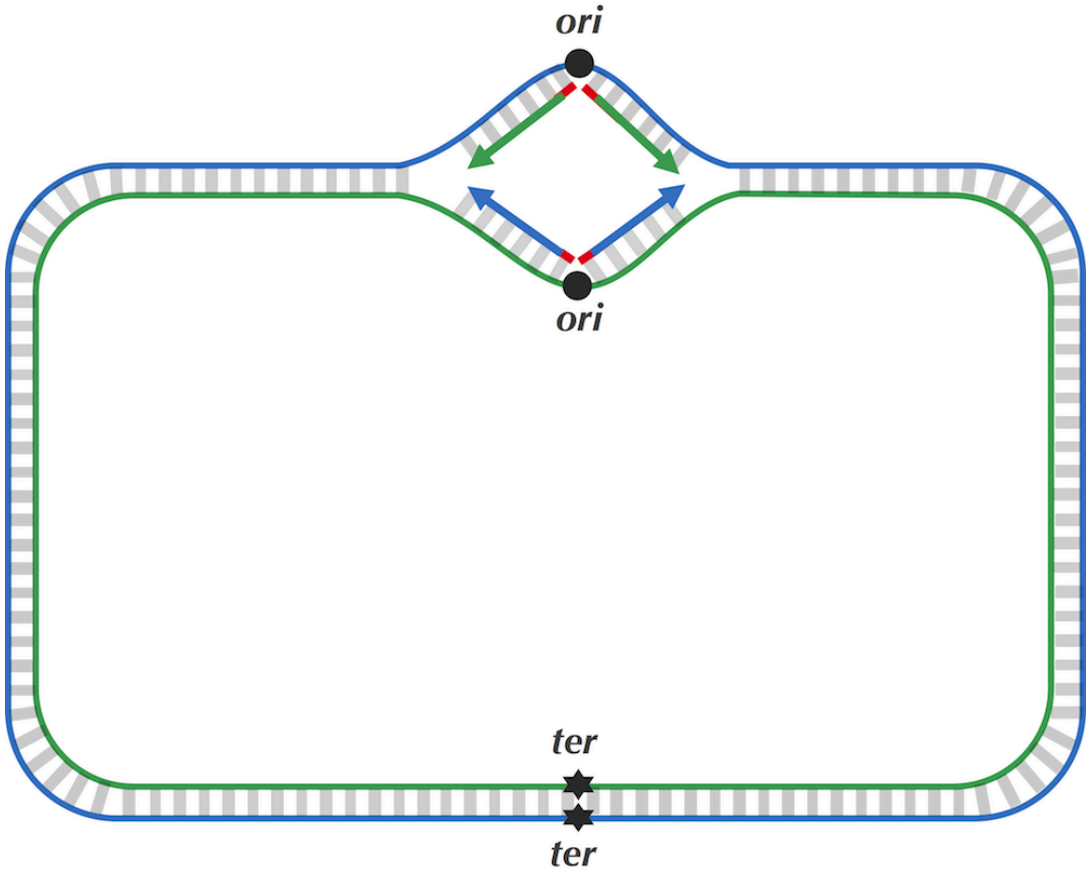
CMSC 423:
Finding Biological Signals

Part 2

Problem: Finding Hidden Messages in the Replication of Origin



- **Input**:

A string *Text* (representing the replication origin of a genome).

- **Output**:

A hidden message in *Text*.

DnaA: protein that binds to a short segment within the *ori* to begin replication

***DnaA* box**: where *DnaA* binds, the "hidden" message within the *ori*

Finding Hidden Messages

- Look for deviations from what is expected
- Random DNA strings do not have long “parts” that repeat nearby each other
- Key idea: Find k-mers that are more frequent than expected
  - *k*-mer: string of length *k* (for example, ACTAT is a 5-mer)

Pseudocode for fining the number of occurrences of a given *k*-mer

```
PatternCount(Text, Pattern)
    count ← 0
    for i ← 0 to |Text| - |Pattern|
        if Text(i, |Pattern|) = Pattern
            count ← count + 1
    return count
```

```
PatternCount(Text, Pattern)
    count ← 0
    for i ← 0 to |Text| − |Pattern|
        if Text(i, |Pattern|) = Pattern
            count ← count + 1
    return count
```

PatternCount(ACTGACTCCCACCCC, ACT)

$|Text| = 15$

$|Pattern| = 3$

Text	A	C	T	G	A	C	T	C	C	C	A	C	C	C	C
i															
Count															

```
PatternCount(Text, Pattern)
    count ← 0
    for i ← 0 to |Text| − |Pattern|
        if Text(i, |Pattern|) = Pattern
            count ← count + 1
    return count
```

PatternCount(ACTGACTCCCACCCC, ACT)

$|Text| = 15$

$|Pattern| = 3$

Text	A	C	T	G	A	C	T	C	C	C	A	C	C	C	C
i	0														
Count	1														

```
PatternCount(Text, Pattern)
    count ← 0
    for i ← 0 to |Text| − |Pattern|
        if Text(i, |Pattern|) = Pattern
            count ← count + 1
    return count
```

PatternCount(ACTGACTCCCACCCC, ACT)

$|Text| = 15$

$|Pattern| = 3$

Text	A	C	T	G	A	C	T	C	C	C	A	C	C	C	C
i	0	1													
Count	1	1													

```
PatternCount(Text, Pattern)
    count ← 0
    for i ← 0 to |Text| − |Pattern|
        if Text(i, |Pattern|) = Pattern
            count ← count + 1
    return count
```

PatternCount(ACTGACTCCCACCCC, ACT)

Text *Pattern*

|*Text*| = 15

|*Pattern*| = 3

Text	A	C	T	G	A	C	T	C	C	C	A	C	C	C	C
i	0	1	2	3	4										
Count	1	1	1	1	2										

```
PatternCount(Text, Pattern)
    count ← 0
    for i ← 0 to |Text| − |Pattern|
        if Text(i, |Pattern|) = Pattern
            count ← count + 1
    return count
```

PatternCount(ACTGACTCCCACCCC, ACT)

Text

Pattern

$|Text| = 15$

$|Pattern| = 3$

STOP

Text	A	C	T	G	A	C	T	C	C	C	A	C	C	C	C
i	0	1	2	3	4	5	6	7	8	9	10	11	12		
Count	1	1	1	1	2	2	2	2	2	2	2	2	2		

and Think

Write pseudocode that finds the number of occurrences of all *k*-mers in a string.

and Think

Write pseudocode that finds the number of occurrences of all *k*-mers in a string.

```
FindAllPatterns(Text, k)
    Count <- an array of length |Text|- k+1
    for i <- 0 to |Text| - k
        Pattern <- Text(i, k)
        Count(i) <- PatternCount(Text, Pattern)
```

FindAllPatterns(ACTGACTCCCACCCC, 3)

Pattern												
i												
Count												

Text	**A**	**C**	**T**	**G**	**A**	**C**	**T**	**C**	**C**	**C**	**A**	**C**	**C**	**C**	**C**

FindAllPatterns(ACTGACTCCCACCCC, 3)

Pattern	ACT											
i	0											
Count	2											

Text	A	C	T	G	A	C	T	C	C	C	A	C	C	C	C
i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Count	1	1	1	1	2	2	2	2	2	2	2	2	2		

FindAllPatterns(ACTGACTCCCACCCC, 3)

Pattern	ACT	CTG										
i	0	1										
Count	2	1										

Text	A	C	T	G	A	C	T	C	C	C	A	C	C	C	C
i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Count	0	1	1	1	1	1	1	1	1	1	1	1	1		

FindAllPatterns(ACTGACTCCCACCCC, 3)

Pattern	ACT	CTG	TGA	GAC	ACT	CTC	TCC	CCC	CCA	CAC	CCC	CCC
i	0	1	2	3	4	5	6	7	8	9	10	11
Count	2	1	1	1	2	1	1	3	1	1	3	3

Problem: Finding the Most Frequent Words

- **Input**:

A string *Text* and an integer *k*.

- **Output**:

All most frequent *k*-mers in *Text*.

- RIGOROUSLY DEFINED COMPUTATIONAL PROBLEM

and Think

Can a string have multiple most frequent *k*-mers?

If no, explain why. If yes, give an example.

and Think

Can a string have multiple most frequent *k*-mers?

If no, explain why. If yes, give an example.

Yes. ATGATGATG

k=2 **AT: 3, TG: 3**, GA: 2

Problem: Finding the Most Frequent Words

```
FrequentWords(Text, k)
    FrequentPatterns ← an empty set
    Count ← an array of length |Text|-k+1
    for i ← 0 to |Text| - k
        Pattern ← Text(i, k)
        Count(i) ← PatternCount(Text, Pattern)
    maxCount ← maximum value in array Count
    for i ← 0 to |Text| - k
        if Count(i) = maxCount
            add Text(i, k) to FrequentPatterns
    remove duplicates from FrequentPatterns
    return FrequentPatterns
```

FrequentWords(ACTGACTCCCACCCC, 3)

Pattern	ACT	CTG	TGA	GAC	ACT	CTC	TCC	CCC	CCA	CAC	CCC	CCC
i	0	1	2	3	4	5	6	7	8	9	10	11
Count	2	1	1	1	2	1	1	3	1	1	3	3

Max Count = 3

FrequentPatterns = {CCC, ~~CCC~~, ~~CCC~~}