CM122/222 Bioinformatics Algorithms

Discussion Section 1A

Shuwen Qiu
janetqiu@cs.ucla.edu

Reminder

Due Apr 13th noon

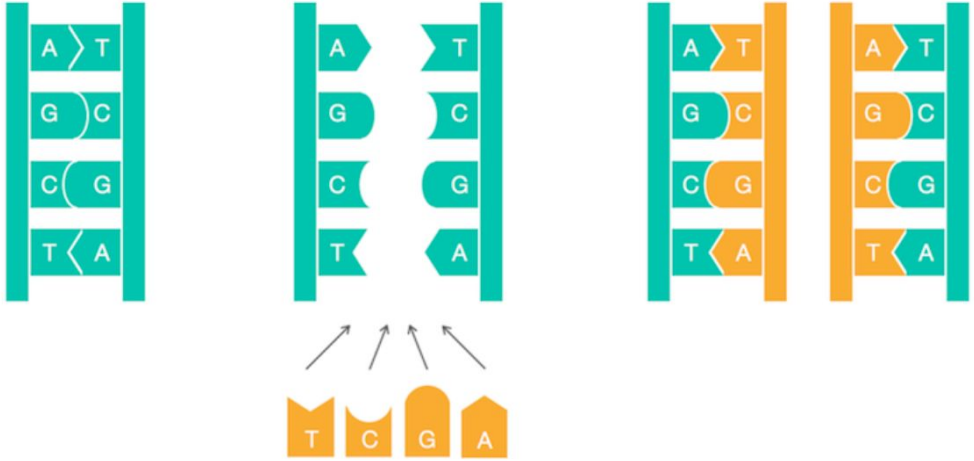
- HW1
- Paper1 Questions

DNA Replication

Nucleotides

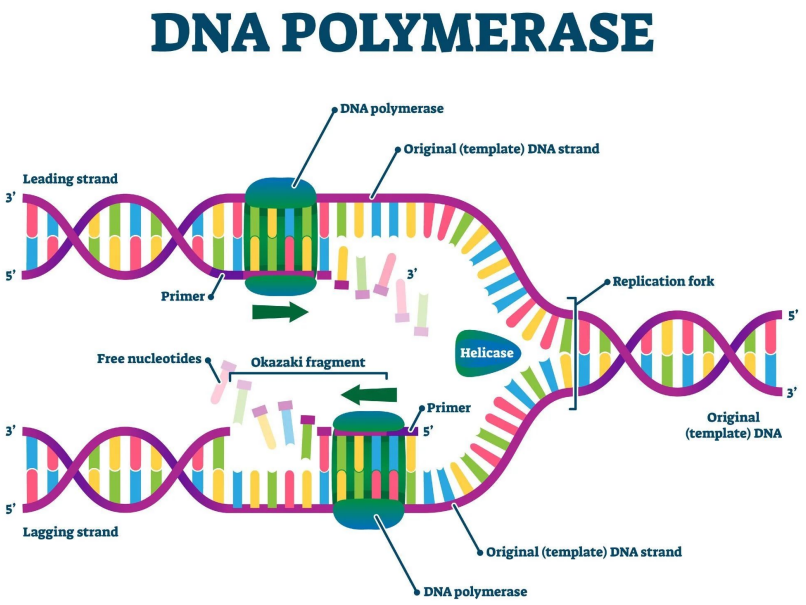
adenine (A) <-> thymine (T)

cytosine (C) <-> guanine (G)



Where in the Genome Does DNA Replication Begin

- Replication origin (ori): where replication begins
- DNA polymerases: copy machines
- Why finding ori important:
  - understanding how cells replicate
  - Gene therapy



Mandal, Ananya. (2022, December 30). What is DNA Polymerase?. News-Medical. Retrieved on April 06, 2023 from https://www.news-medical.net/life-sciences/What-is-DNA-Polymerase.aspx.

DnaA Boxes

DnaA boxes: short segment within the ori

DnaA: mediate the initiation of replication that binds with DnaA

initiation of replication is mediated by DnaA, a protein that binds to a short segment within the ori known as a DnaA box

Finding DnaA Boxes

Decipher “hidden messages” in “The Gold-Bug”

```
53‡‡†305))6·;4826)4‡.)4‡);806·;48†8^60))85;161;:‡·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(‡?34;48)4‡;1‡(;:188;‡?;
```

```
53‡‡†305))6·THE26)H‡.)H‡)TE06·THE†E^60))E5T161T:‡·E
†E3(EE)5·†TH6(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(‡?3HTHE)H‡T1‡(T:1EET‡?T
```

48 → THE
; → T
4 → H
8 → E

Counting “words”

DnaA boxes: common patterns showing in high frequency?

Example:

Any high frequent nucleotide substring?

ACAACTATGCATACTATCGGGAACTATCCT

Counting “words”

Task: find the most frequent substring in ori

K-mer: a string of length k

Task 1: Given a k-mer pattern, count the frequency that it appears as a substring of Text: Count(Text, Pattern)

Big-O Notation

How we measure an algorithm efficiency?

O(1) describes an algorithm that will always execute in the same time (or space) regardless of the size of the input data set

```
bool IsFirstElementNull(IList<String> elements)
{
    return elements[0] == null;
}
```

https://robbell.io/2009/06/a-beginners-guide-to-big-o-notation

Big-O Notation

O(N) describes an algorithm whose performance will grow linearly and in direct proportion to the size of the input data set.

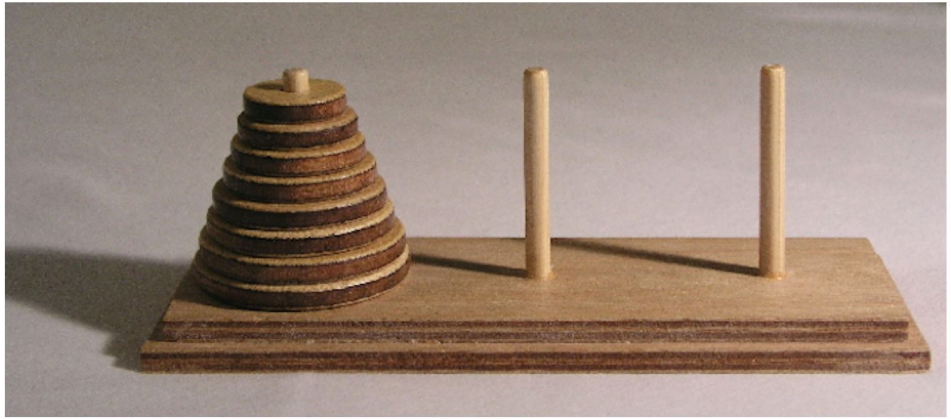
Examples for O(N^2)?

```
bool ContainsValue(IEnumerable<string> elements, string value)
{
    foreach (var element in elements)
    {
        if (element == value) return true;
    }
    return false;
}
```

Big-O Notation

O(2^N) denotes an algorithm whose growth doubles with each addition to the input data set.

Recursion: The tower of Hanoi

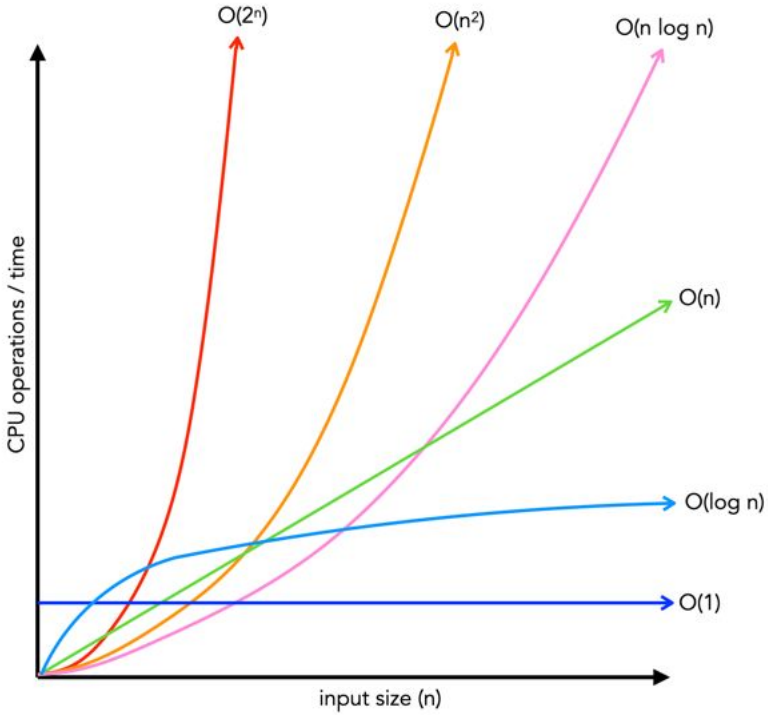


Move(n, left, right) = Move(1, left, middle) + Move(n-1, left, right) + Move(1, middle, right)

Big-O Notation

O(logN): Doubling the size of the input data set has little effect on its growth

craftofcoding.wordpress.com



Credited to Luke

Counting “words”

9-mer: bacterial DnaA boxes are usually nine nucleotides long

a 9-mer appearing three or more times in a randomly generated DNA string of length 500 is approximately 1/1300 → how?

Frequent Words in *Vibrio cholerae*

k	3	4	5	6	7	8	9
count	25	12	8	8	5	4	3
***k*-mers**	tga	atga	gatca	tgatca	atgatca	atgatcaa	atgatcaag
			tgatc				cttgatcat
							tcttgatca
							ctcttgatc

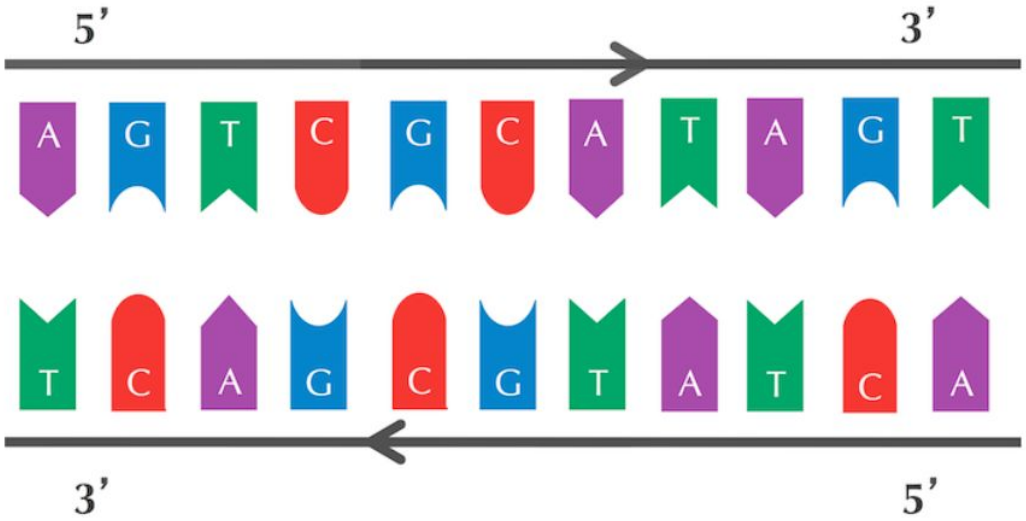
Probabilities of Patterns in a String

1. Compute Pr(7, 3, “01”, 2)
2. Compute Pr(N, A, k-mer Pattern, t)
3. Compute Pr(N, A, k, t): some k-mer appearing t or more times in a string of N
4. Pr(500, 4, 9, 3)

Counting Strings and Complimentary Strings

Direction: 5’ → 3’

Task 2: reverse complement of string



Counting Strings and Complimentary Strings

Count(Pattern) + Count($Pattern_{RC}$) is ever higher

```
atcaatgatcaacgtaagcttctaagcATGATCAAGgtgctcacacagtttatccacaac
ctgagtggatgacatcaagataggtcgttgtatctccttcctctcgtactctcatgacca
cggaaagATGATCAAGagaggatgatttcttggccatatcgcaatgaatacttgtgactt
gtgcttccaattgacatcttcagcgccatattgcgctggccaaggtgacggagcgggatt
acgaaagcatgatcatggctgttgttctgtttatcttgttttgactgagacttgttagga
tagacggtttttcatcactgactagccaaagccttactctgcctgacatcgaccgtaaat
tgataatgaatttacatgcttccgcgacgatttacctCTTGATCATcgatccgattgaag
atcttcaattgttaattctcttgcctcgactcatagccatgatgagctCTTGATCATgtt
tccttaaccctctattttttacggaagaATGATCAAGctgctgctCTTGATCATcgtttc
```

Finding ‘ori’

How to locate ‘ori’?

- Finding locations where high frequent k-mer happens
- Clump: a k-mer that appears many times within a short interval of the genome
- (L, t)-clump
  - For a given window of a string of length L, some k-mers appear >= t times
  - `gatcagcataagggtccC`**`TGCA`**`A`**`TGCA`**`TGACAAGCC`**`TGCA`**`GTtgttttac`
  - `TGCA forms a (25, 3)-clump`
- Task 3:
  - **Input**: A string *Genome*, and integers *k*, *L*, and *t*.
  - **Output**: All distinct *k*-mers forming (*L*, *t*)-clumps in *Genome*.

Finding ‘ori’

- Task 3:
  - **Input**: A string *Genome*, and integers *k*, *L*, and *t*.
  - **Output**: All distinct *k*-mers forming (*L*, *t*)-clumps in *Genome*.
- Subtasks:
  - Counting frequency of all possible k-mers in a Genome of length L

```
Clump(Genom, k, L, t)
res = []
for pos in 0, |Genom| - L
    window = Genom[pos: pos+L]
    FreqTable = CountFreq(k, window)
    for k-mer in FreqTable:
        if FreqTable[k-mer] >= t:
            res.append(k-mer)
return res.
```

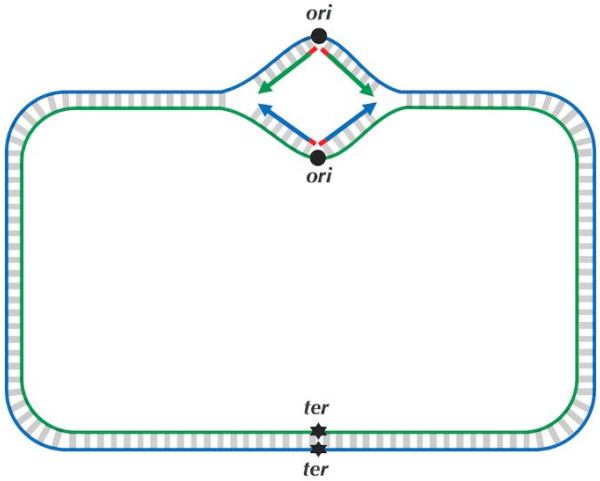
Finding ‘ori’

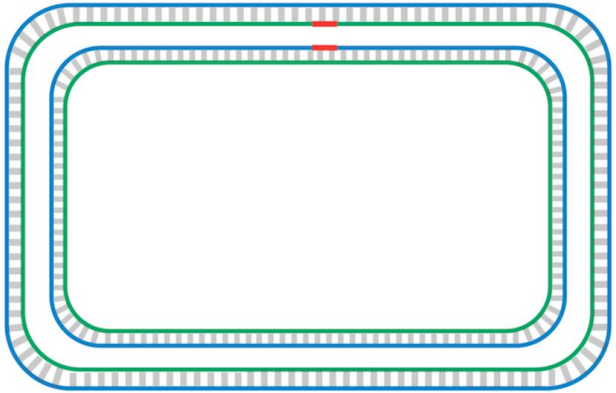
The number of clump locations can be huge

- Looking more into DNA replication

Replicate DNA

- Replication is bidirectional
- Primer: where replication starts
- the strands completely separate at the replication terminus (ter)

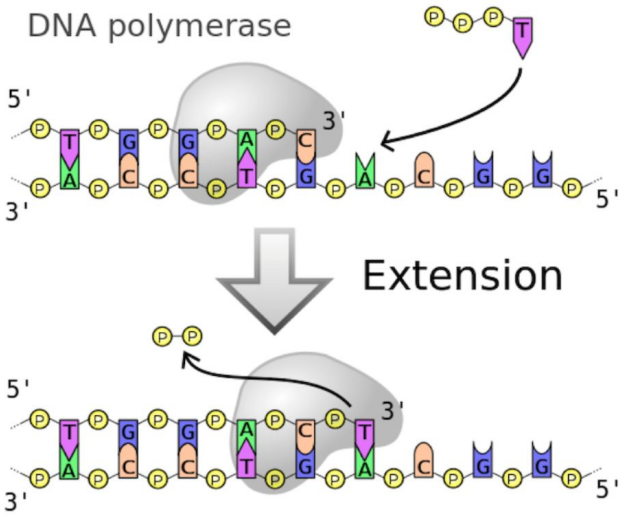




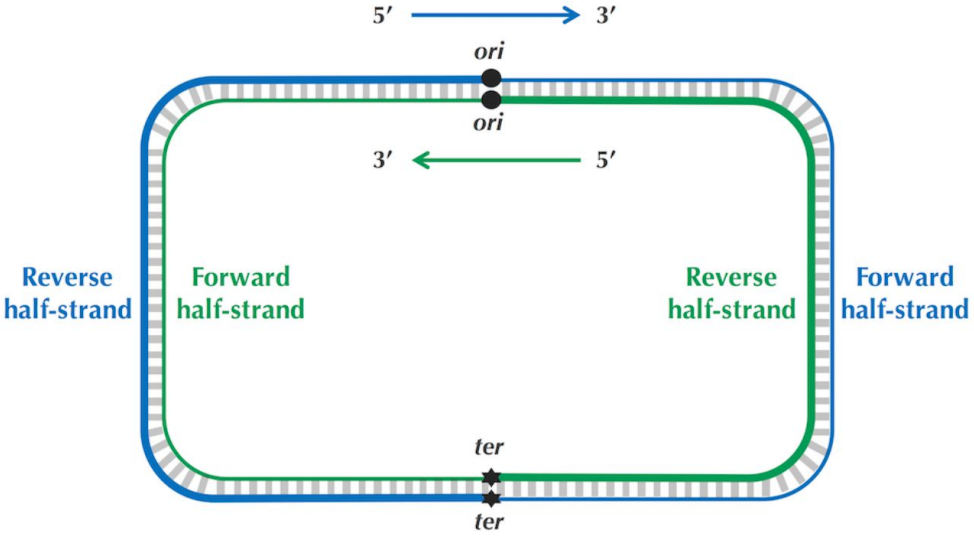
Asymmetry of Replication

DNA polymerases are unidirectional: 3’ → 5’

What about forward half-strand?

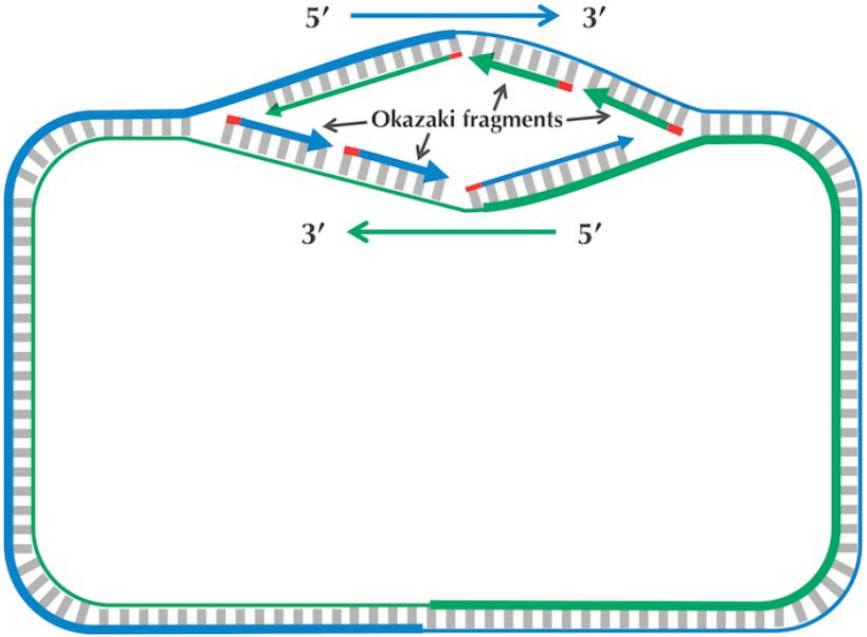


Courtesy Madeleine Price Ball



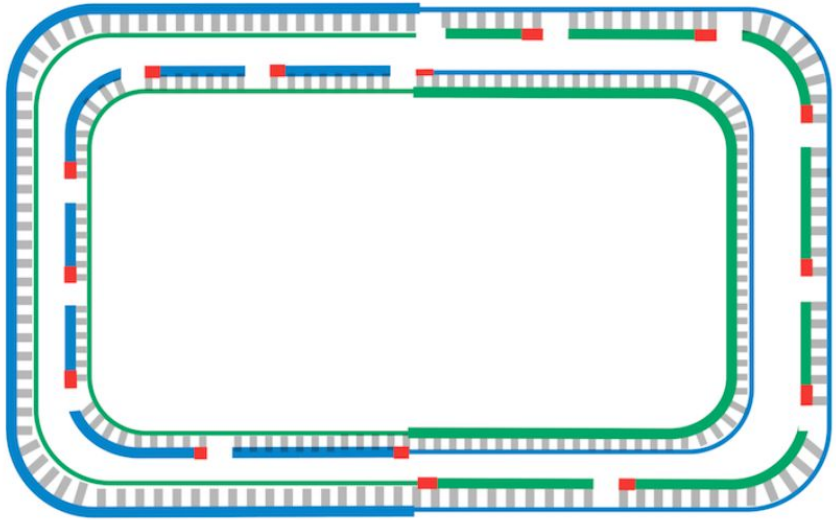
Okazaki fragments

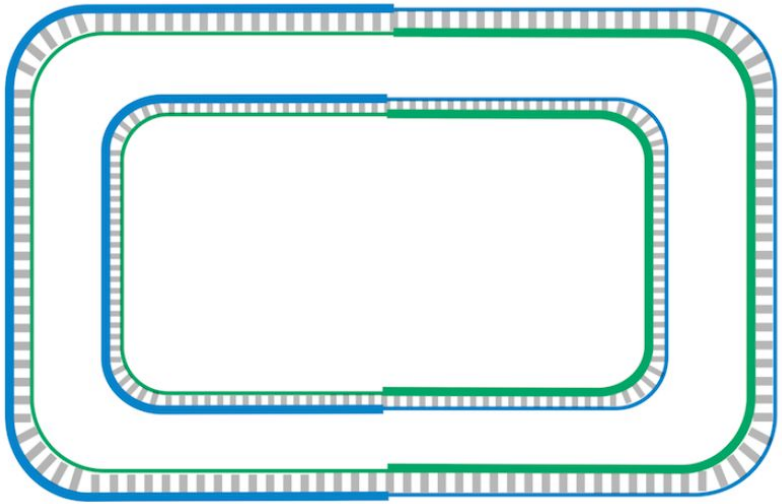
- Reverse half-strand: progresses continuously
- forward half-strand: wait until the another 2,000 nucleotides or so opened. It then requires a new primer to begin synthesizing



Okazaki fragments

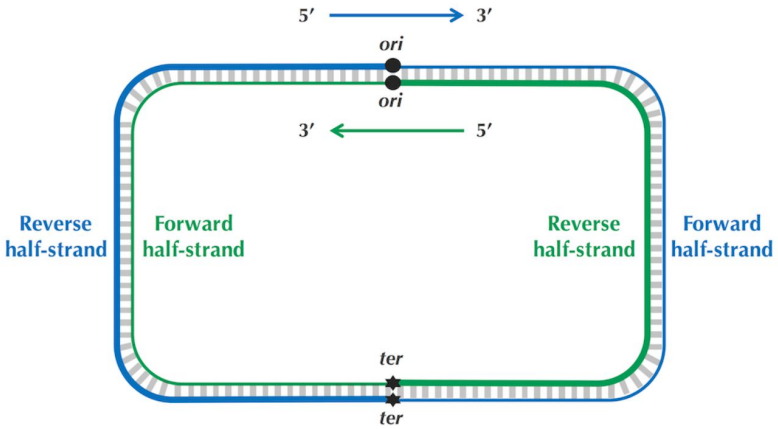
consecutive Okazaki fragments are sewn together by an enzyme called **DNA ligase**

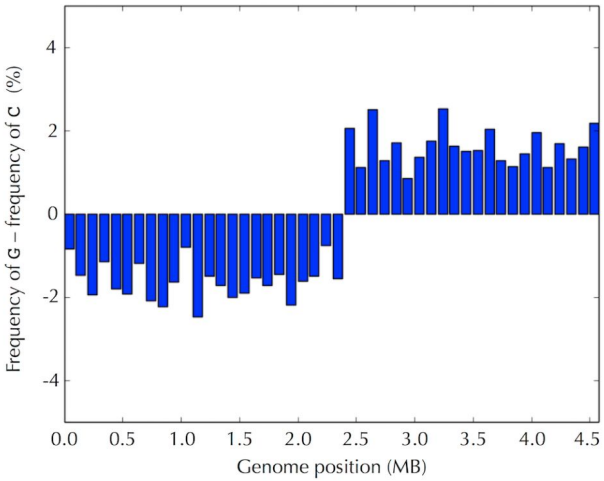




Peculiar Statistics of the Forward and Reverse Half-Strands

- The difference between the frequencies of **guanine** and **cytosine** across the 46 fragments of the *E. coli* genome
- Starting from ter. → the reverse half-strand, → ori. → the forward half-strand.





Deamination

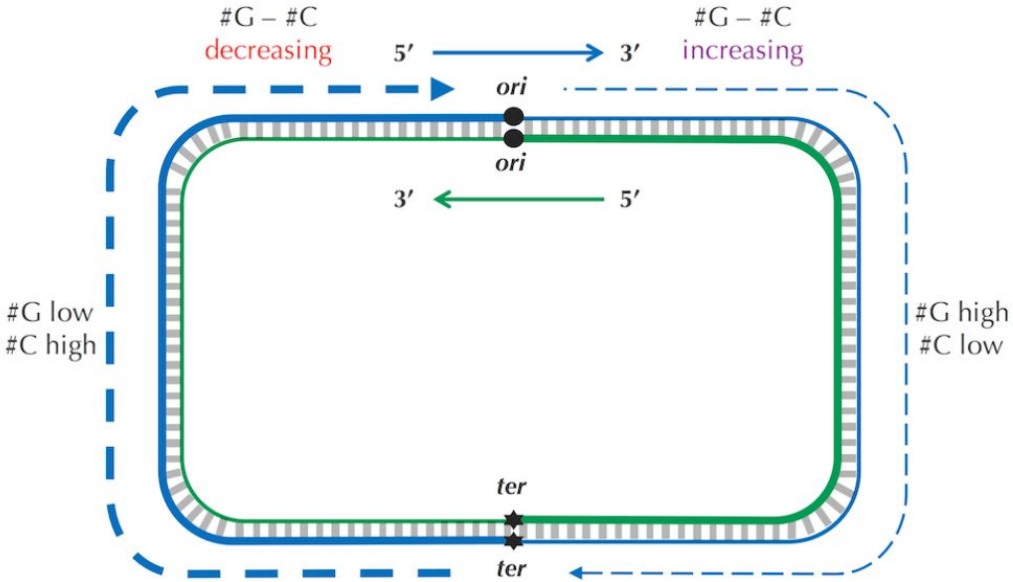
How does this difference happen?

- A reverse half-strand proceeds quickly, it lives double-stranded for most of its life.
- A forward half-strand spends a much larger amount of its life single-stranded, waiting to be used as a template for replication.
- Deamination: (C) has a tendency to mutate into (T)

	#C	#G	#A	#T
Entire strand	427419	413241	491488	491363
Reverse half-strand	219518	201634	243963	246641
Forward half-strand	207901	211607	247525	244722
Difference	+11617	-9973	-3562	+1919

The Skew Diagram

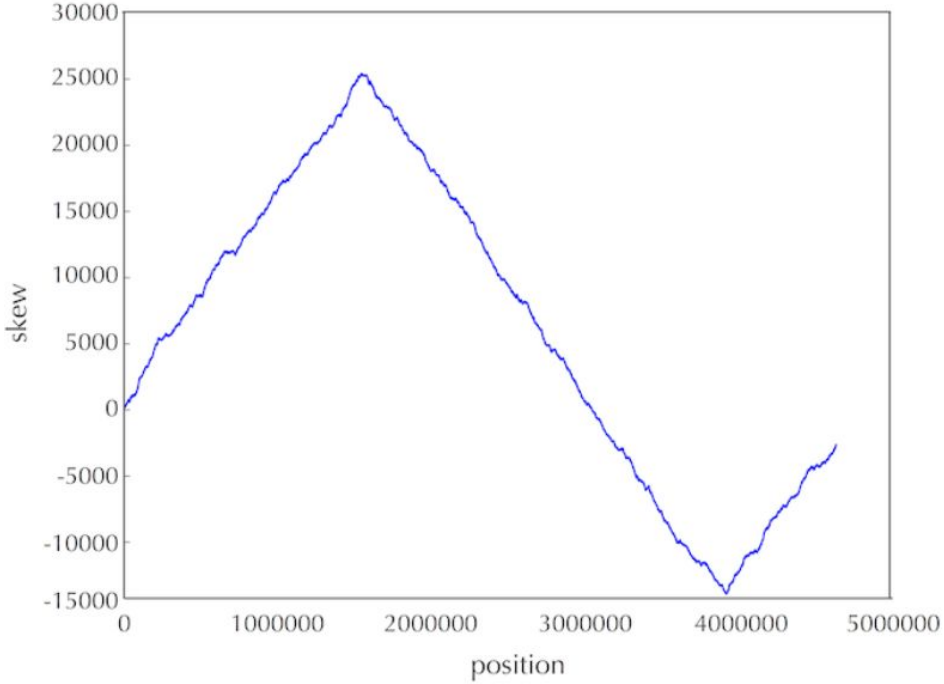
How to locate ori?



The Skew Diagram

Task 4: locate minimum #G-#C position in strings

$Skew_i$(Genome): the difference between the total number of occurrences of G and the total number of occurrences of C in the first i nucleotides of Genome.



```
def MinSkew(Genome).
    diff = 0.  diffmin = 0
    pos_vecs = [ ]
    for pos from 0 to |Genome|:
        if Genom[pos] == 'c'
            diff = diff - 1
        if - - - - 'G'
            diff = diff + 1
        if diff == diffmin:
            add pos to pos_vecs
    if diff < diffminimal:
        diffmin = diff
        pos_vecs = [pos]
```

Finding DnaA Boxes in ori

No 9-mer happens 3 or more times in ori?

```
aatgatgatgacgtcaaaaggatccggataaaacatggtgattgcctcgc
ataacgcggtatgaaaatggattgaagcccgggccgtggattctactcaa
ctttgtcggcttgagaaagacctgggatcctgggtattaaaaagaagatc
tatttatttagagatctgttctattgtgatctcttattaggatcgcactg
ccctgtggataacaaggatccggcttttaagatcaacaacctggaaagga
tcattaactgtgaatgatcggtgatcctggaccgtataagctgggatcag
aatgaggggttatacacaactcaaaaactgaacaacagttgttctttgga
taactaccggttgatccaagcttcctgacagagttatccacagtagatcg
cacgatctgtatacttatttgagtaaattaacccacgatcccagccattc
ttctgccggatcttccggaatgtcgtgatcaagaatgttgatcttcagtg
```

Finding DnaA Boxes in ori

ATGATCAAC and CATGATCAT, which differ from ATGATCAAG and CTTGATCAT in only a single nucleotide

```
atcaATGATCAACgtaagcttctaagcATGATCAAGgtgctcacacagtttatccacaac
ctgagtggatgacatcaagataggtcgttgtatctccttcctctcgtactctcatgacca
cggaaagATGATCAAGagaggatgatttcttggccatatcgcaatgaatacttgtgactt
gtgcttccaattgacatcttcagcgccatattgcgctggccaaggtgacggagcgggatt
acgaaagCATGATCATggctgttgttctgtttatcttgttttgactgagacttgttagga
tagacggtttttcatcactgactagccaaagccttactctgcctgacatcgaccgtaaat
tgataatgaatttacatgcttccgcgacgatttacctCTTGATCATcgatccgattgaag
atcttcaattgttaattctcttgcctcgactcatagccatgatgagctCTTGATCATgtt
tccttaaccctctattttttacggaagaATGATCAAGctgctgctCTTGATCATcgtttc
```

Counting “words” with Mismatches

Task 5: Hamming Distance

A	T	G	A	C	C	G
A	A	T	A	C	G	G

The Hamming distance is 3

The Hamming distance measures the number of mismatches between corresponding positions in two sequences (assuming they have equal length)

Credited to Luke

Counting “words” with Mismatches

Task 6: $Count_d$(Text, Pattern): the total number of occurrences of Pattern in Text with at most d mismatches

- Relation to task 1?

Task 7: Find the most frequent k-mers with mismatches <= d in a string.

- Subtask: find the most frequent k-mers in a string.
- Subtask: generate all possible neighbors of a Pattern with <=d mismatches

Generating Neighbors(Pattern, d)

- Generating Neighbors(Pattern, 1)?
- If we know Neighbors(Suffix(Pattern), d) how does it help us construct Neighbors(Pattern, d)?

```
def GenNeighbors(Pattern, d):
    If  d == 0:
        return Pattern
    If len(Pattern) == 1:
        return [A, G, C, T]
    New_neighbors = [ ]
    Neighbors = GenNeighbors(Pattern[1:], d)
    for neighbor in Neighbors:
        If hammingDist(neighbor, Pattern[1:]) == d:
            new_neighbors.append(Pattern[0] + neighbor)
        Else:
            for symbol in [A, T, C, G]:
                new_neighbors.append(symbol + neighbor)
```

```
def MostKmer(Text, k, d)
    freqTable = { }
    for pos from 0 to |Text| - k:
        k-mer = Text[pos:pos+k]
        neighors = GenNeighbor(k-mer, d)
        for pattern in neghors
            if pattern in FreqTable:
                FreqTable[pattern] + 1
            else:
                FreqTable[pattern] = 1
    MaxinFreq in FreqTable.
```

Finding DnaA Boxes in ori

- Try a small window either starting, ending, or centered at the position of minimum skew
- Still having other high frequent 9-mers serving as “hidden messages” for other functions

```
aatgatgatgacgtcaaaaggatccggataaaacatggtgattgcctcgc
ataacgcggtatgaaaatggattgaagcccgggccgtggattctactcaa
ctttgtcggcttgagaaagacctgggatcctgggtattaaaaagaagatc
tatttatttagagatctgttctattgtgatctcttattaggatcgcactg
cccTGTGGATAAcaaggatccggcttttaagatcaacaacctggaaagga
tcattaactgtgaatgatcggtgatcctggaccgtataagctgggatcag
aatgaggggTTATACACAactcaaaaactgaacaacagttgttcTTTGGA
TAActaccggttgatccaagcttcctgacagagTTATCCACAgtagatcg
cacgatctgtatacttatttgagtaaattaacccacgatcccagccattc
ttctgccggatcttccggaatgtcgtgatcaagaatgttgatcttcagtg
```