CS 466
Introduction to Bioinformatics
Lecture 1

Mohammed El-Kebir

August 27, 2018

In case of an emergency…

Emergencies can happen anywhere and at any time, so it's important that we take a minute to prepare for a situation in which our safety could depend on our ability to react quickly. Take a moment to learn the different ways to leave this building. If there's ever a fire alarm or something like that, you'll know how to get out and you'll be able to help others get out. Next, figure out the best place to go in case of severe weather – we'll need to go to a low-level in the middle of the building, away from windows. And finally, if there's ever someone trying to hurt us, our best option is to run out of the building. If we cannot do that safely, we'll want to hide somewhere we can't be seen, and we'll have to lock or barricade the door if possible and be as quiet as we can. We will not leave that safe area until we get an Illini-Alert confirming that it's safe to do so. If we can't run or hide, we'll fight back with whatever we can get our hands on. If you want to better prepare yourself for any of these situations, visit police.illinois.edu/safe. Remember you can sign up for emergency text messages at emergency.illinois.edu.

Course Staff

Instructor:

- Mohammed El-Kebir (melkebir)
- Office hours: Mondays, 3:15-4:15pm

Developing combinatorial algorithms for studying all stages of cancer progression.

TA:

- Anusri Pampari (pampari2)
- Office hours: TBD

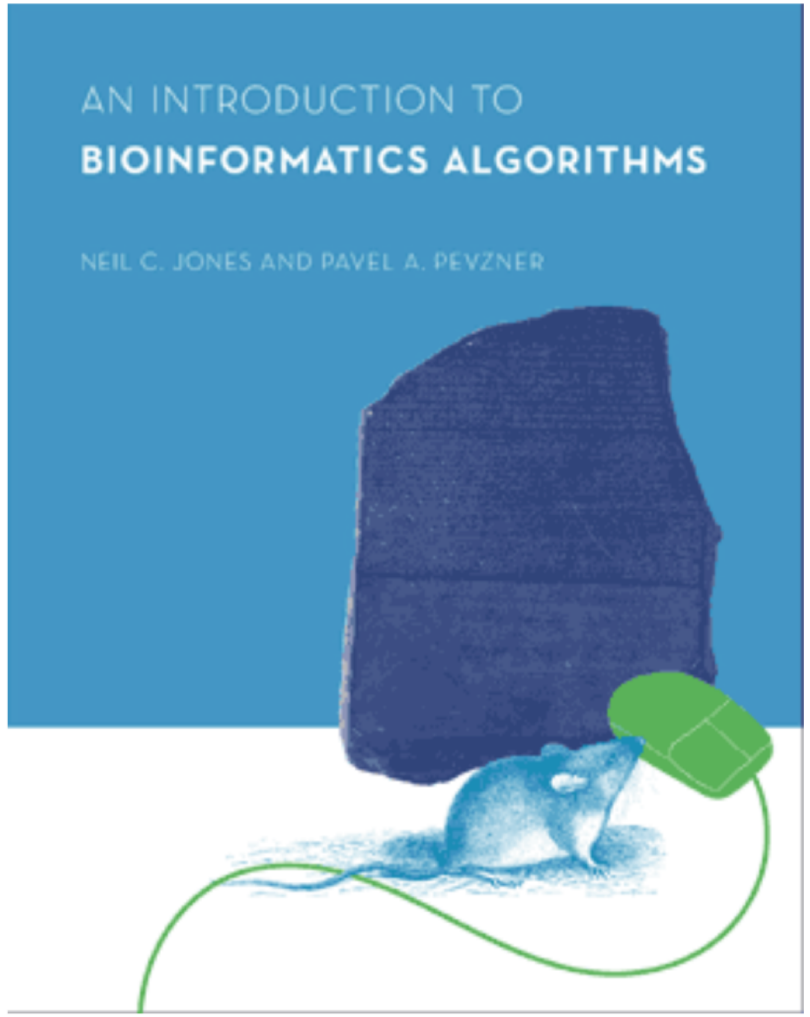
Course Organization

Course website:
www.el-kebir.net/teaching/cs466

Syllabus:
- Prerequisites: CS 225 and its prerequisites
- Textbook

Grading:
- 5 written/programming assignments
- Midterm
- Final
- Research project

Piazza: (please sign up)
- https://piazza.com/class#fall2018/cs466



Course Objectives

Learn:

- Learn underlying ideas of common algorithms in bioinformatics.
- Learn to translate a biological problem into a computational problem.
- Learn to read scientific papers, propose and conduct independent research.

Not learn:

- Will not learn to run popular bioinformatics packages.
- Will not learn how to program.

Homework Assignments

- 5 homework assignments
- Each homework assignment is a combination of written/programming exercises
- LaTeX highly recommended for homework assignments
- Please use Python for programming exercises

Late policy:

- Students may request one 3-day extension in the semester for full credit
- Late submission within 3 days 80%

Primer on Molecular Biology

Molecular Biology is the field of **biology** that studies the composition, structure and interactions of cellular **molecules** – such as nucleic acids and proteins – that carry out the **biological** processes essential for the cell's functions and maintenance.

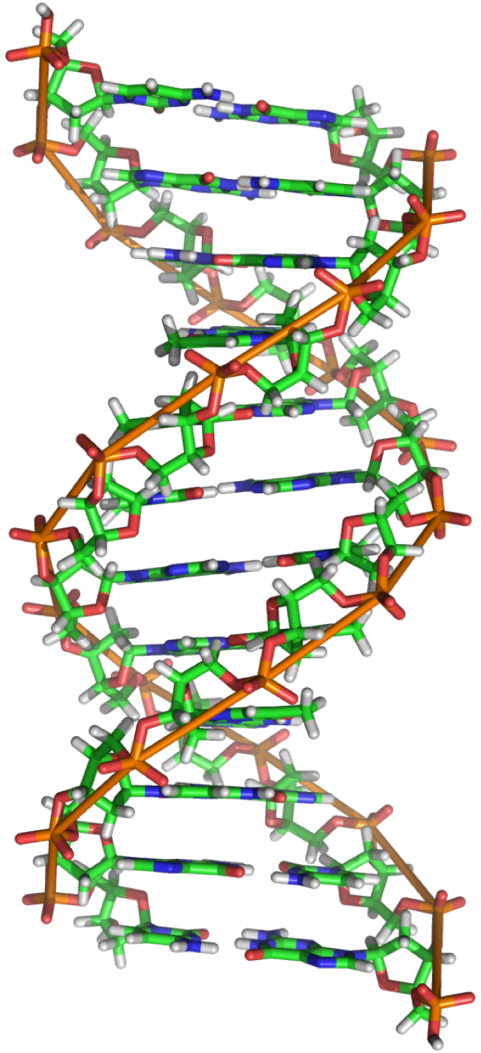
https://www.nature.com/subjects/molecular-biology

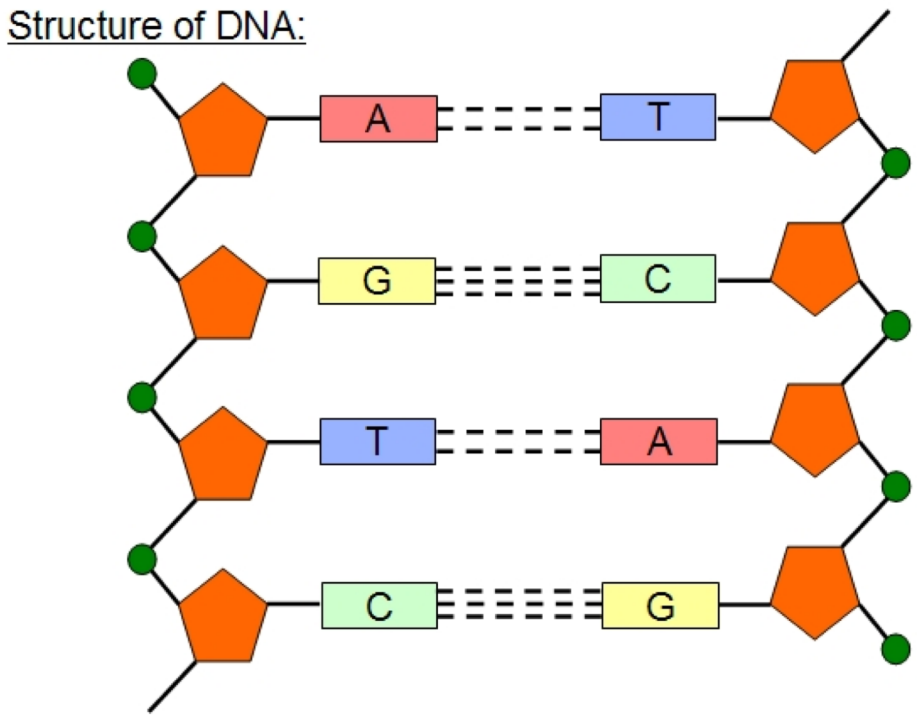
Cellular molecules:

1. DNA
2. RNA
3. Protein

DNA

Each strand composed of sequence of covalently bonded **nucleotides** (**bases**).





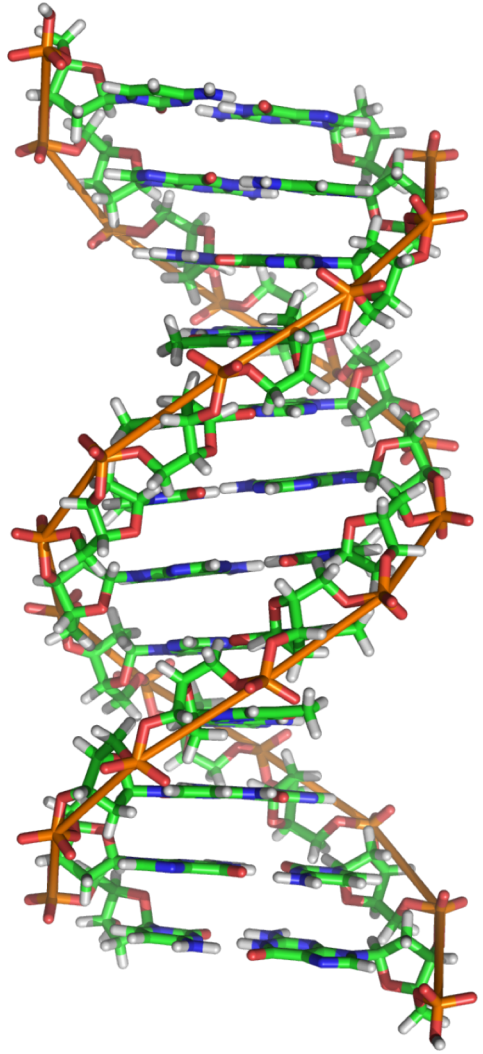
Four nucleotides:

A (adenine)
C (cytosine)
T (thymine)
G (guanine)

A ←→ T, C ←→G Watson-Crick base-pairing

DNA

Each strand composed of sequence of covalently bonded **nucleotides** (**bases**).



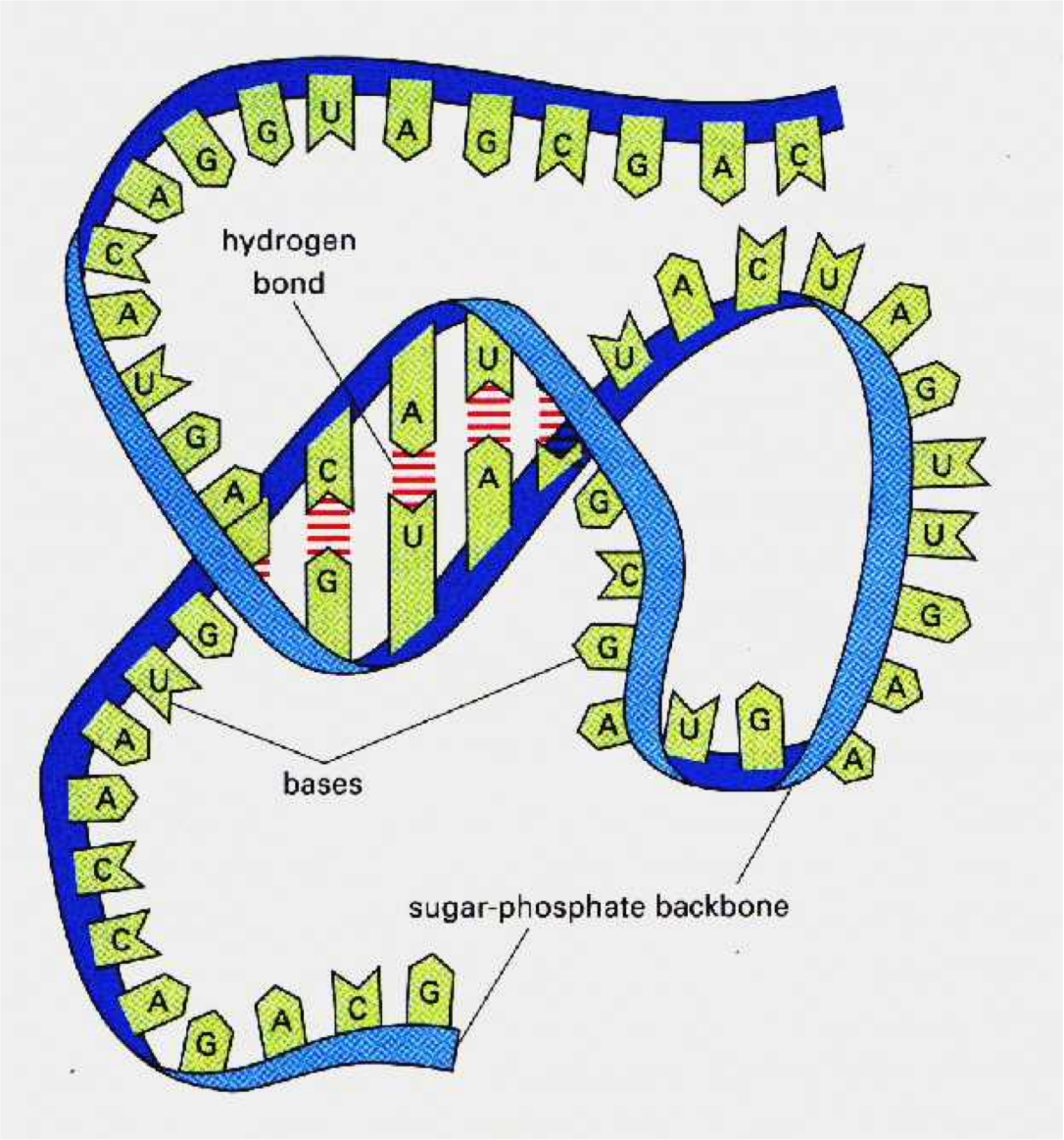
```
5' ...ACGTGACTGAGGACCGTG...  3'
   ...|||||||||||||||||||...
   ...TGCACTGACTCCTGGCAC...
3'                           5'
```

Pair of strings from 4 character alphabet

```
5' ...ACGTGACTGAGGACCGTG
      CGACTGAGACTGACTGGGT
      CTAGCTAGACTACGTTTTA
      TATATATATACGTCGTCGT
      ACTGATGACTAGATTACAG
      TGATTTTAAAAAAATATT... 3'
```

Single string from 4 character alphabet

RNA



- **Single-stranded**
  - A (adenine)
  - C (cytosine)
  - U (uracil)
  - G (guanine)

- Can fold into **structures** due to base complementarity.

  A ←→ U, C ←→G

- Comes in many flavors:

  mRNA, rRNA, tRNA, tmRNA, snRNA, snoRNA, scaRNA, aRNA, asRNA, piwiRNA, etc.

Protein

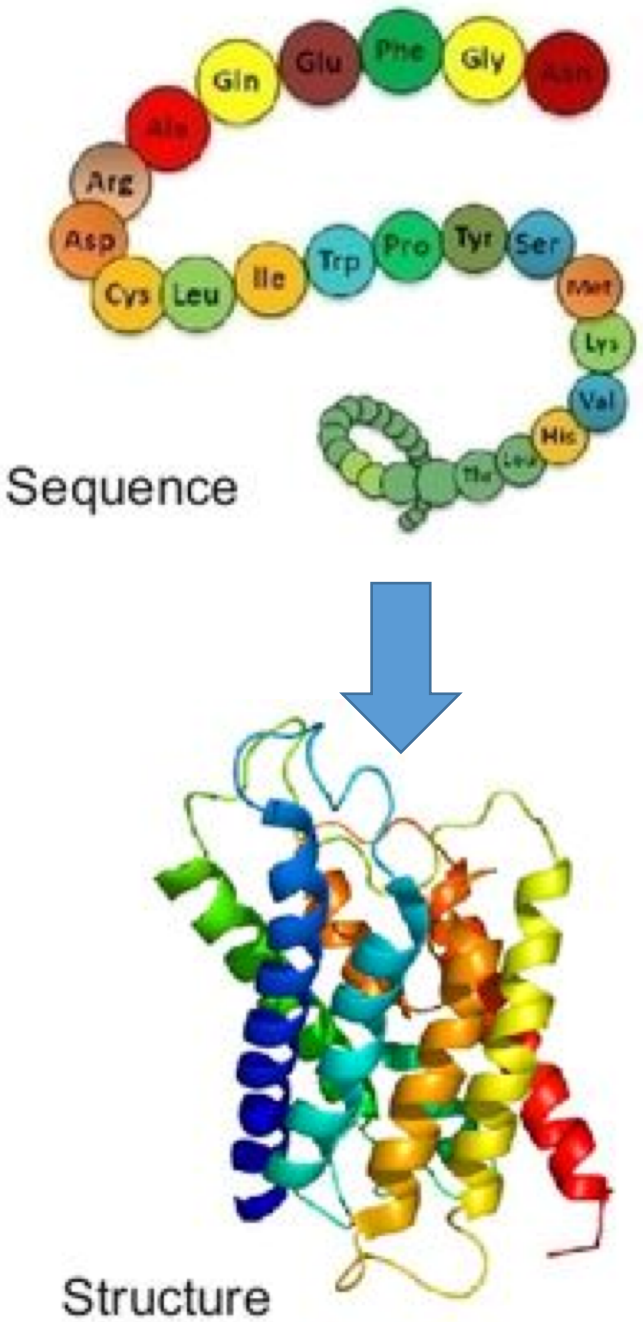
- String of amino acids: 20 letter alphabet

…DTIGDWNSPSFFGIQLVSSVHT
TLWYRENAFPVLGGFSWLSWFNW
HNMGYYYPVYHIGYPMIRCGTHL
VPMQFAFQSIARSFALVHWNAPM
VLKINPHERQDPVFWPCLYYSVD
IRSMHIGYPMIRCYQA…

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

Protein

- String of amino acids: 20 letter alphabet
- Folds into 3D structures to perform various functions in cells



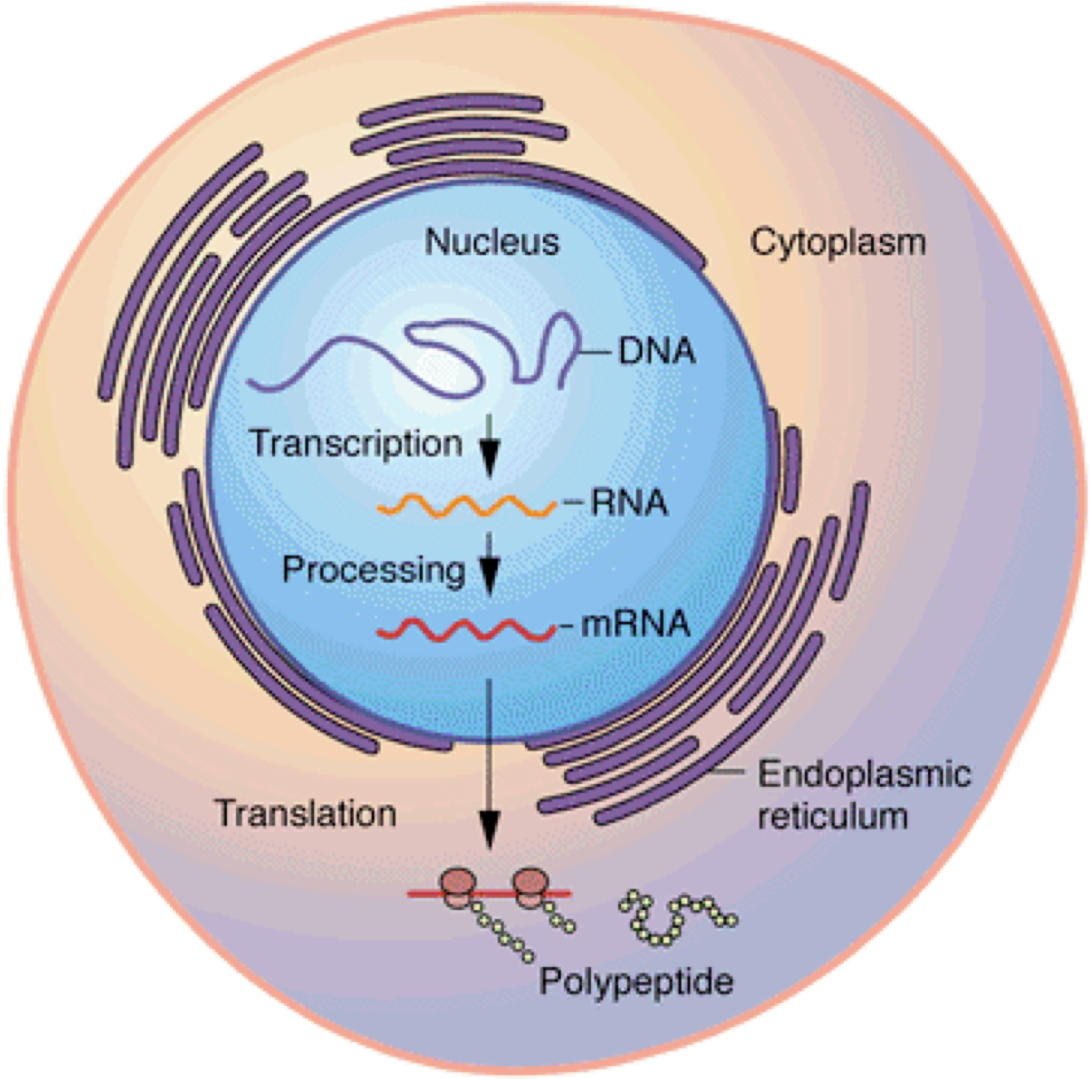
Primer on Molecular Biology

Three fundamental molecules:

1. **DNA**
   Information storage.

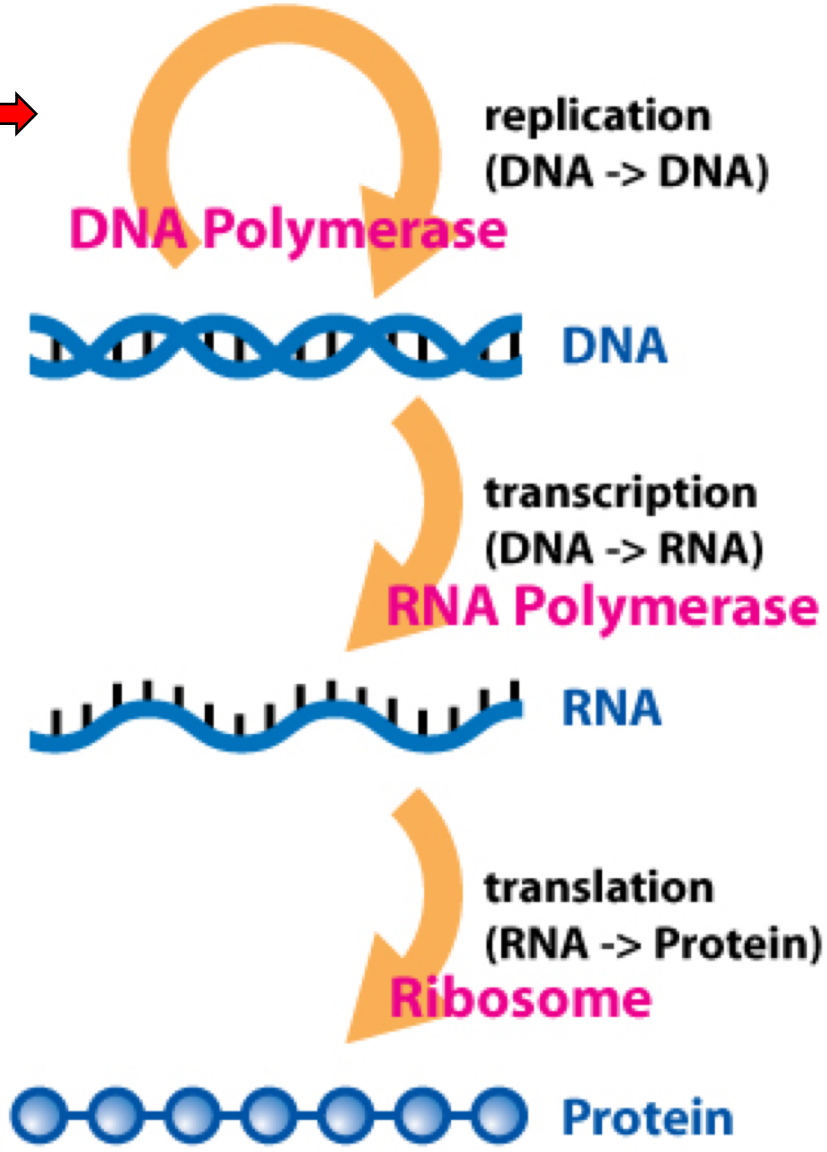
2. **RNA**
   Old view: Mostly a “messenger”.
   New view: Performs many important functions.

3. **Protein**
   Perform most cellular functions (biochemistry, signaling, control, etc.)



Central Dogma of Molecular Biology

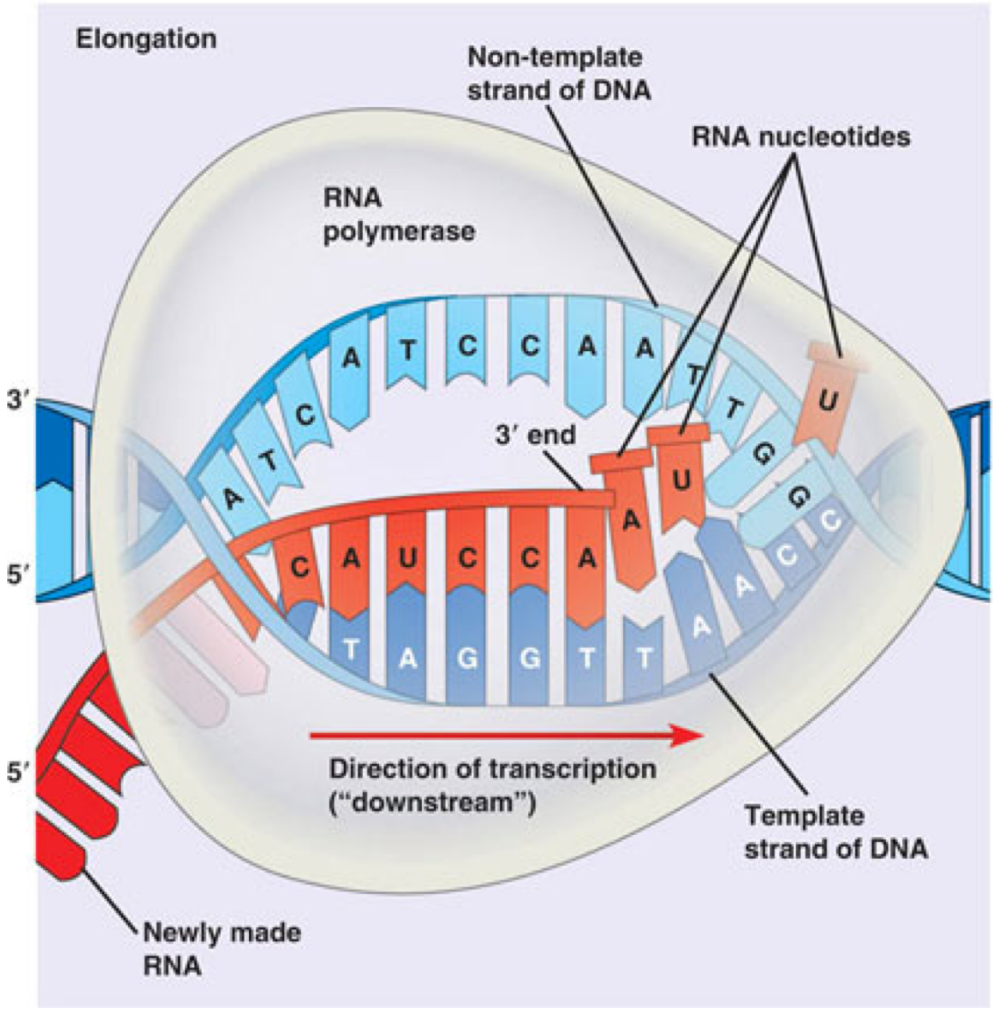
Start here



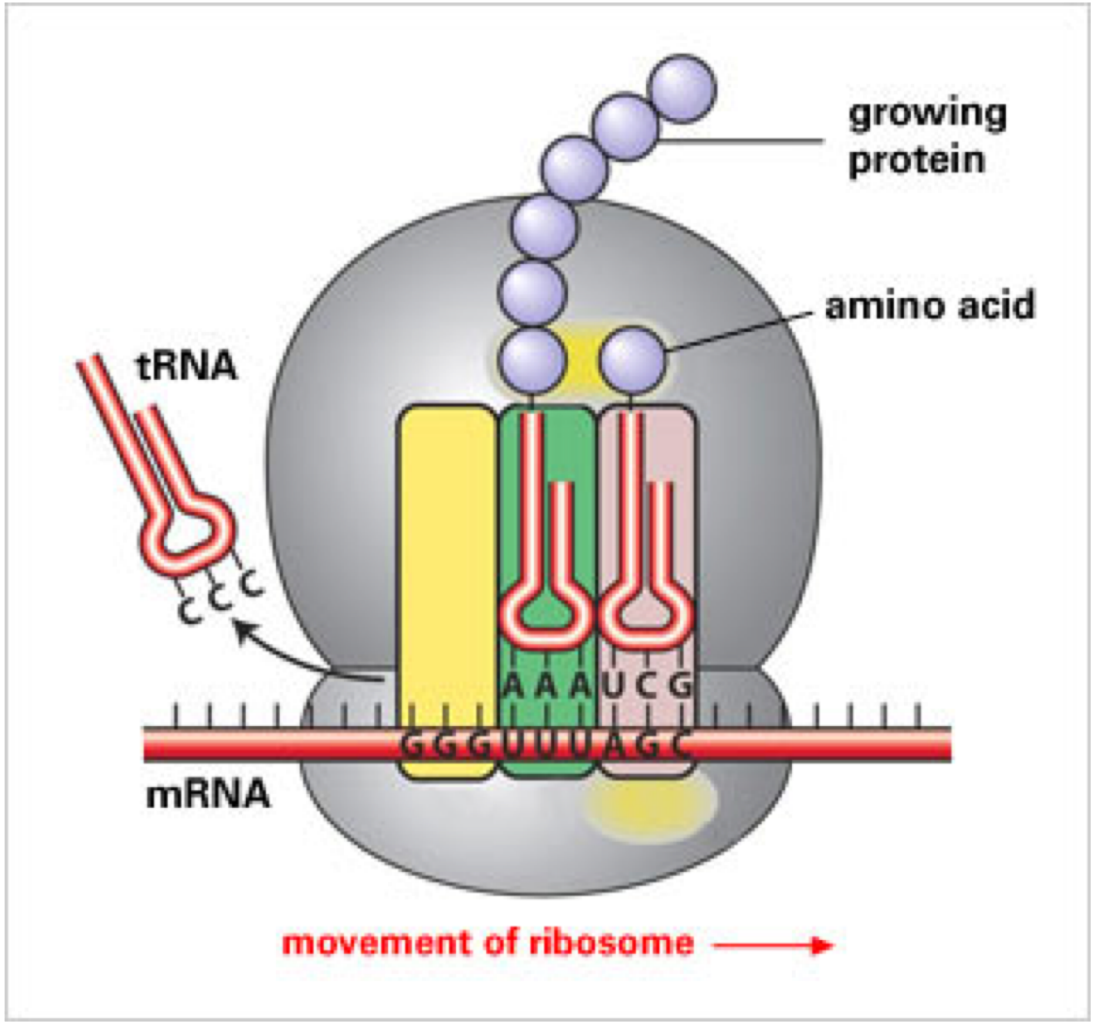
DNA → RNA → Protein:
The process by which cells "read" the genome

First proposed by Francis Crick in 1956.

Transcription and Translation

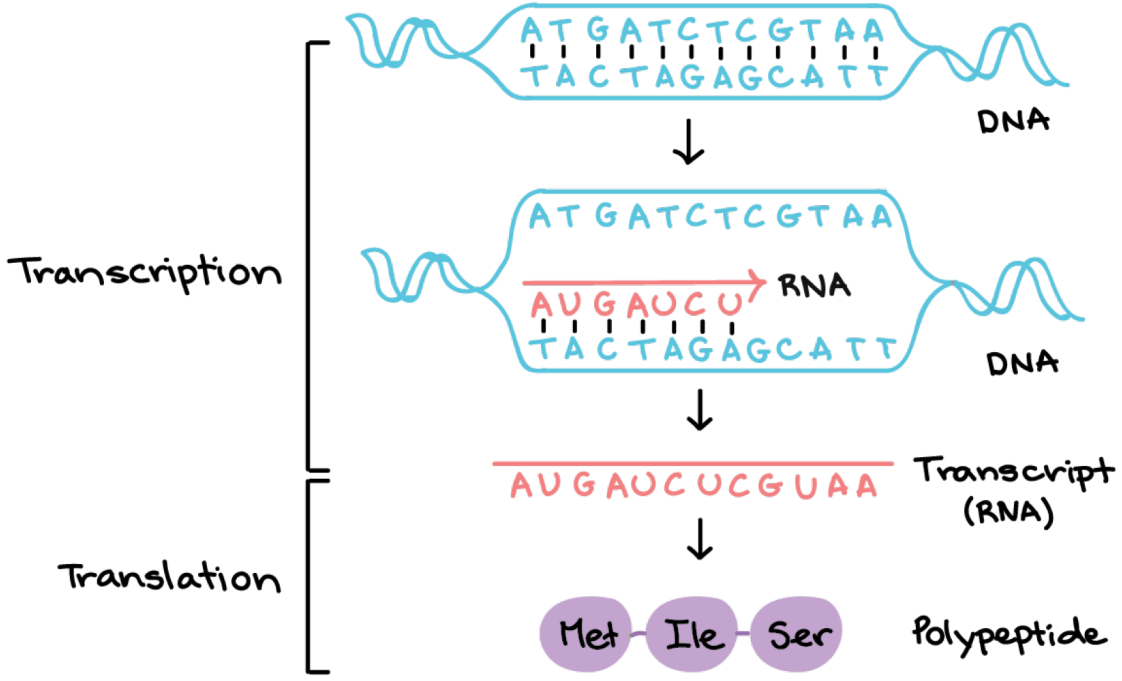


http://dna-rna.net/wp-content/uploads/2011/08/rna-transcription2.jpg

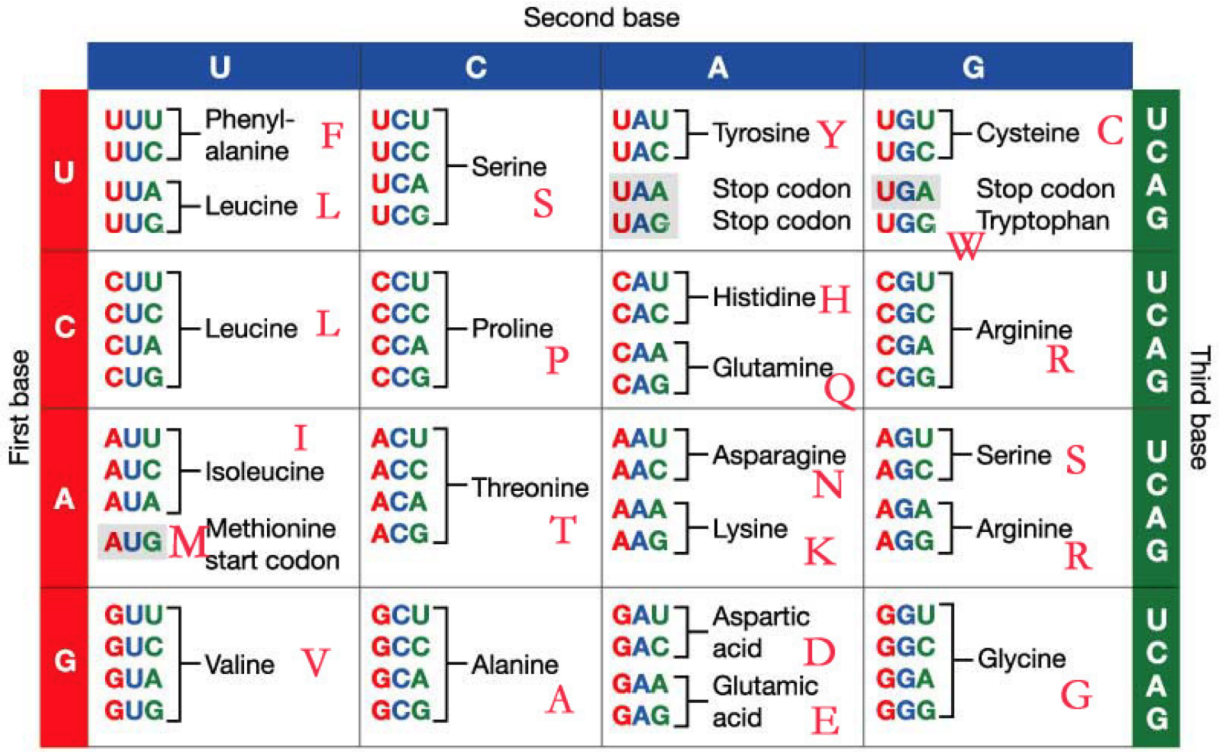


http://www.frontiers-in-genetics.org/en/pictures/translation_1.jpg

Transcription and Translation



https://www.khanacademy.org/science/biology/gene-expression-central-dogma/transcription-of-dna-into-rna/a/overview-of-transcription



http://bioinfo.bisr.res.in/project/crat/pictures/codon.jpg

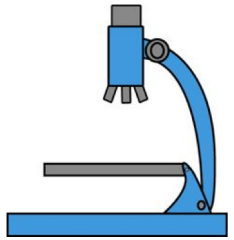
What is Computational Biology/Bioinformatics?

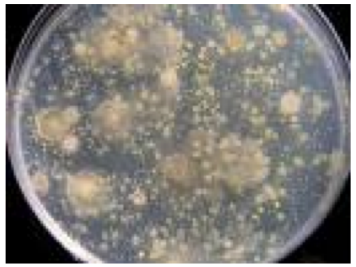
Computational biology and **bioinformatics** is an interdisciplinary field that develops and applies **computational methods** to analyze large collections of biological data, such as genetic sequences, cell populations or protein samples, to make new predictions or **discover new biology**.

https://www.nature.com/subjects/computational-biology-and-bioinformatics

Technology and Bioinformatics are Transforming Biology

Until late 20th Century





Hypothesis Generation and Validation

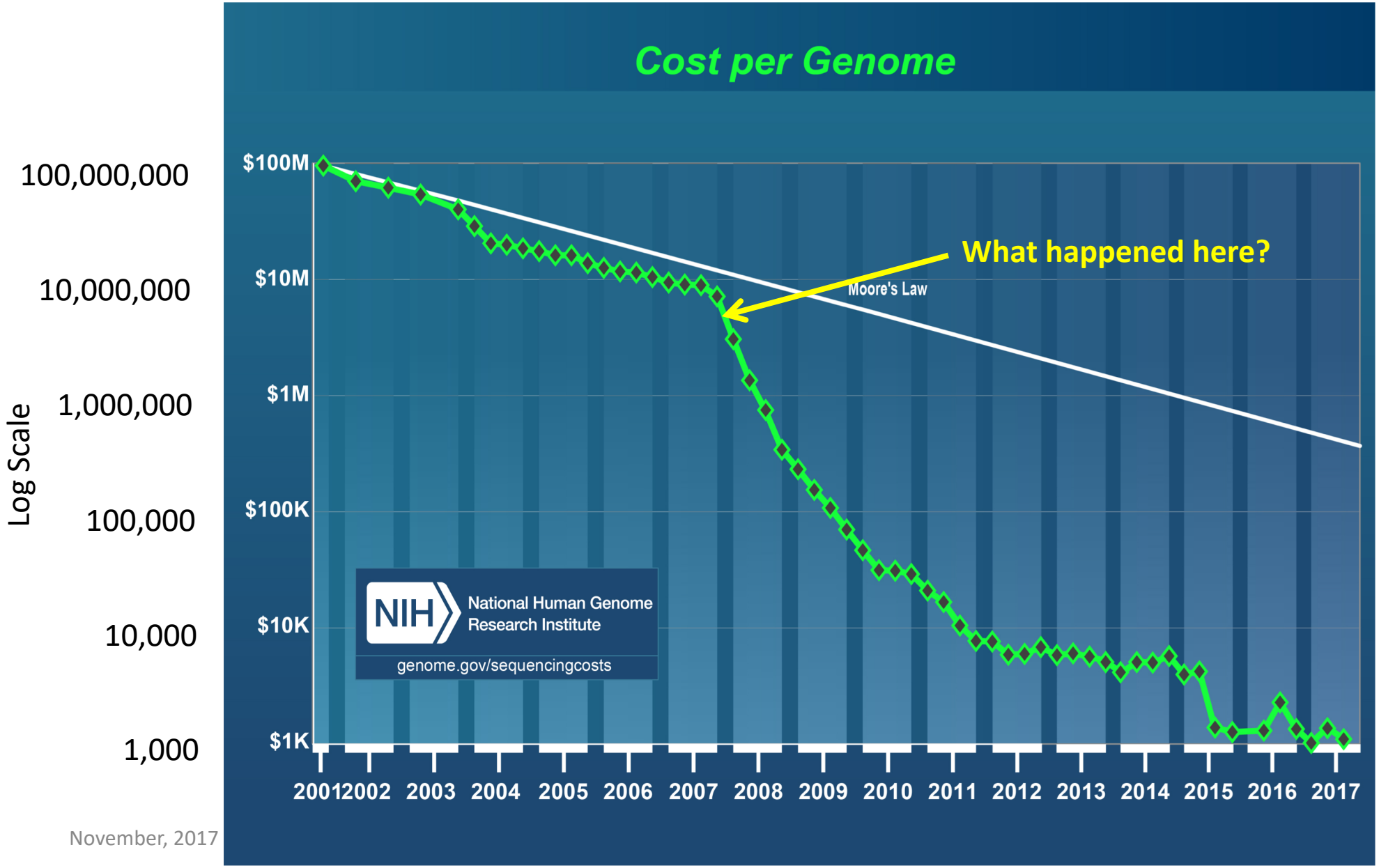
21th Century and Beyond

Algorithms

Hypothesis Generation and Validation

High throughput technologies

A Deluge of Data



A Deluge of Data

1000 Genomes
A Deep Catalog of Human Genetic Variation

adsp
Alzheimer's Disease Sequencing Project

1000 Plant Genomes

Autism Genome 10K

T1K
Fish-T1K
Transcriptomes of 1000 Fishes

GENOME 10K

100k genomes project

International Cancer Genome Consortium

HMP
NIH HUMAN MICROBIOME PROJECT

NIH
THE CANCER GENOME ATLAS
National Cancer Institute
National Human Genome Research Institute

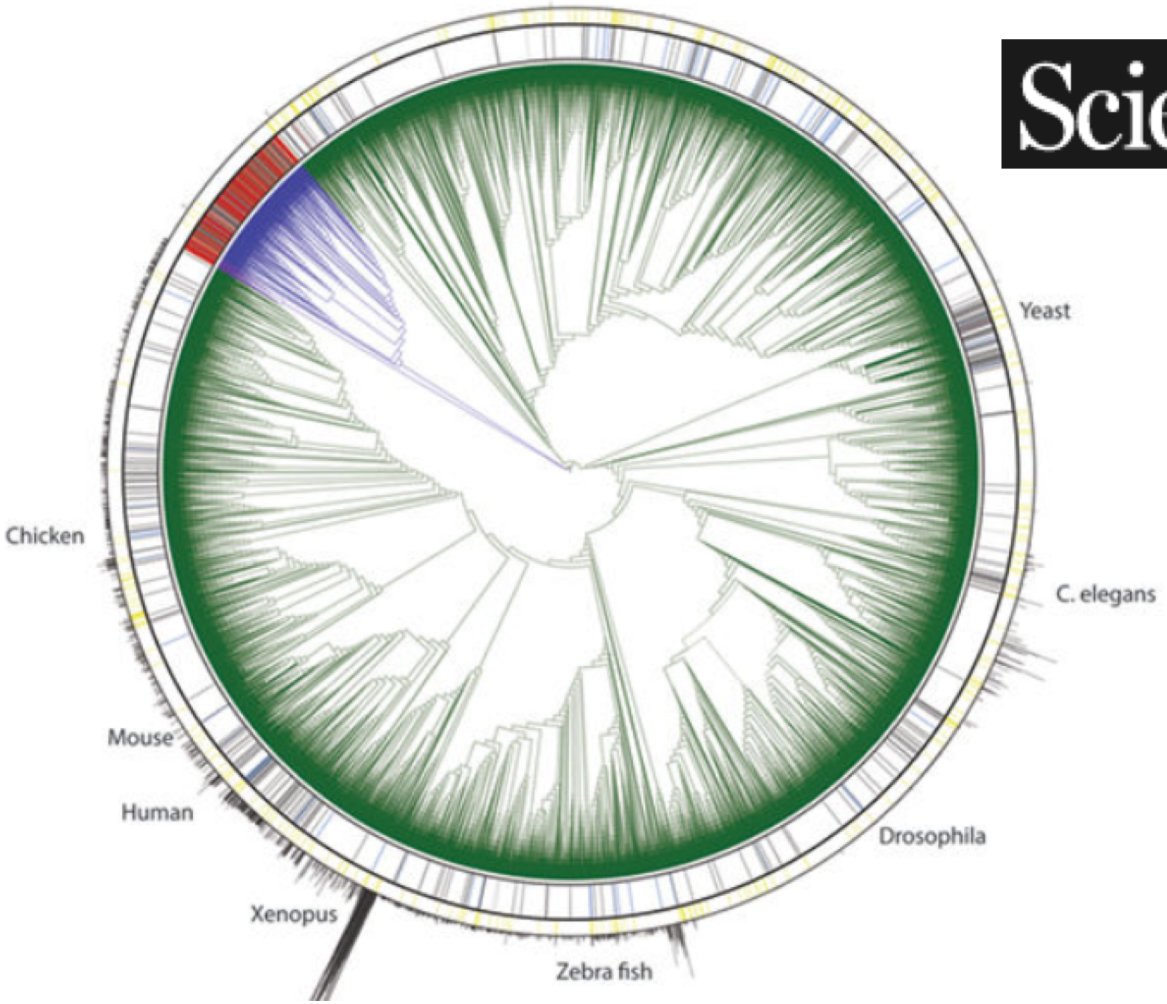
A Deluge of Data

Biologists propose to sequence the DNA of all life on Earth

By **Elizabeth Pennisi** | Feb. 24, 2017 , 1:15 PM

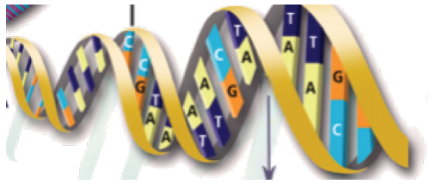
Science

Outer ring color scheme:
Red: Completed genome
Light Blue: Low resolution genome

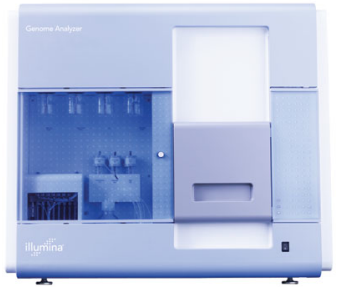


Question: What does it mean that we can sequence a genome?

No technology exists that can sequence a complete (human) genome from end to end!



Genome
Millions -billions nucleotides



Next-generation DNA sequencing

... CATTCAGTAG ...

... AGCCATTAG ...

... GGTAGTTAG ...

... GGTAAACTAG ...

... TATAATTAG ...

... CGTACCTAG ...

10-100's million noisy ***reads***
Reads: 30-1000 nucleotides

Making sense of this data absolutely requires the use and development of **algorithms**!

Why Study Computational Biology?

Interdisciplinary

- Biology
- Computer Science
- Mathematics
- Statistics

= FUN!

Why choose just 1?

Best Jobs	Worst Jobs
1. Actuary	200. Newspaper reporter
2. Audiologist	199. Lumberjack
3. Mathematician	198. Enlisted Military Personnel
4. Statistician	197. Cook
5. Biomedical Engineer	196. Broadcaster
6. Data Scientist	195. Photojournalist
7. Dental Hygienist	194. Corrections Officer
8. Software Engineer	193. Taxi Driver
9. Occupational Therapist	192. Firefighter
10. Computer Systems Analyst	191. Mail Carrier

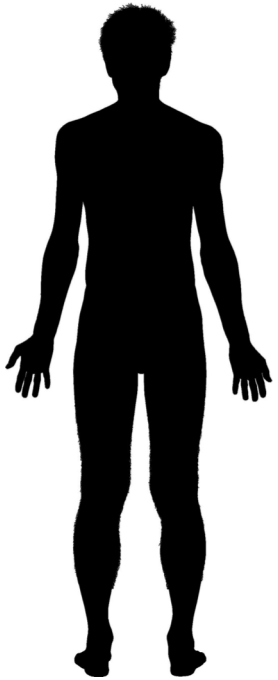
http://www.careercast.com/jobs-rated/jobs-rated-report-2015-ranking-top-200-jobs

Donald Knuth
Professor emeritus of Computer Science at Stanford University
Turing Award winner
"father of the analysis of algorithms."

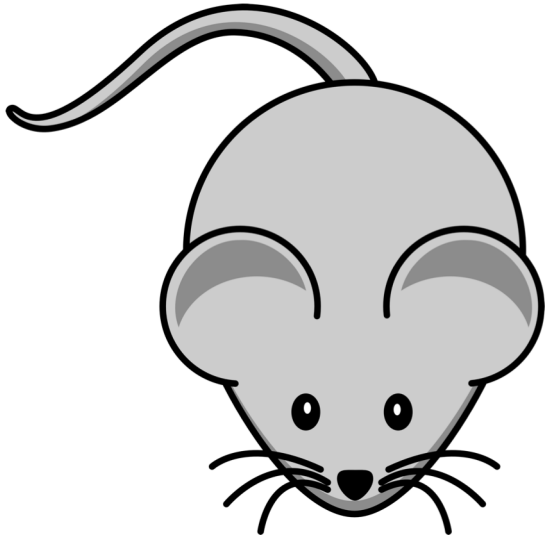
"I can't be as confident about computer science as I can about biology. Biology easily has 500 years of exciting problems to work on. It's at that level."

Course Topic #1: Sequence Alignment

Question: How do we compare two genes/genomes?



vs.



Human Genome:

…ACTCGACTGAGAGGATTTCGAGCATGA…

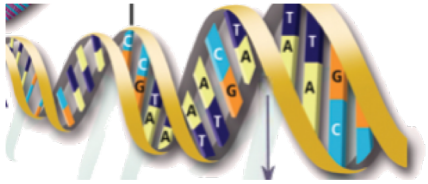
≈3.2 x 10^9 bp

Mouse Genome:

…ACTCAACTGAGATTCGAGCTTCAATGA…

≈2.8 x 10^9 bp

Course Topic #2: Genome Assembly



... CATTCAGTAG ...

... AGCCATTAG ...

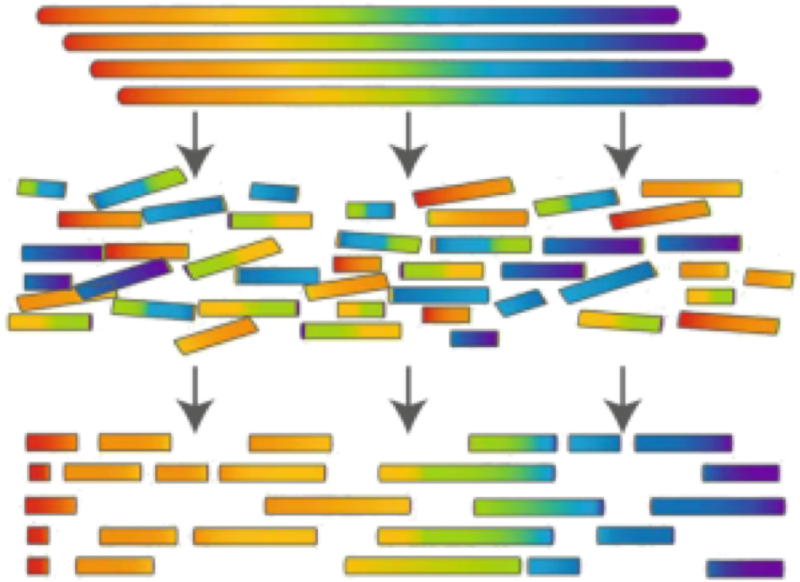
... GGTAGTTAG ...

... GGTAAACTAG ...

... TATAATTAG ...

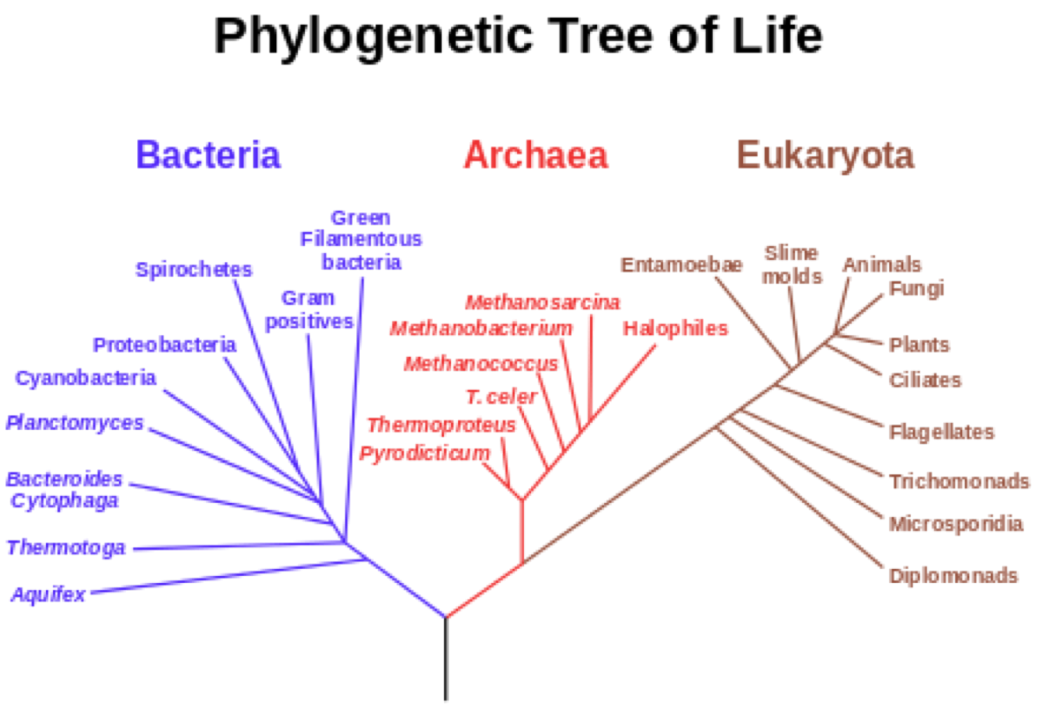
... CGTACCTAG ...

Question: How do we put all the pieces back together?



ATGTTCCGATTAGGAAACCTATCTGTAACTGTTTCATTCAGTAAAAGGAGGAAATATAA

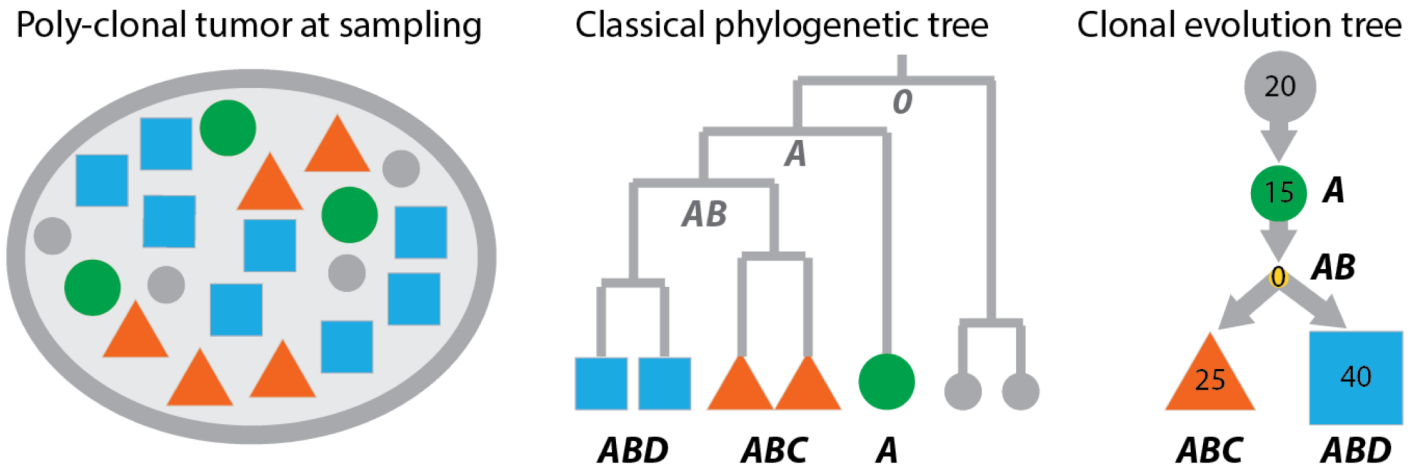
Course Topic #3: Phylogenetics



https://en.wikipedia.org/wiki/Phylogenetic_tree

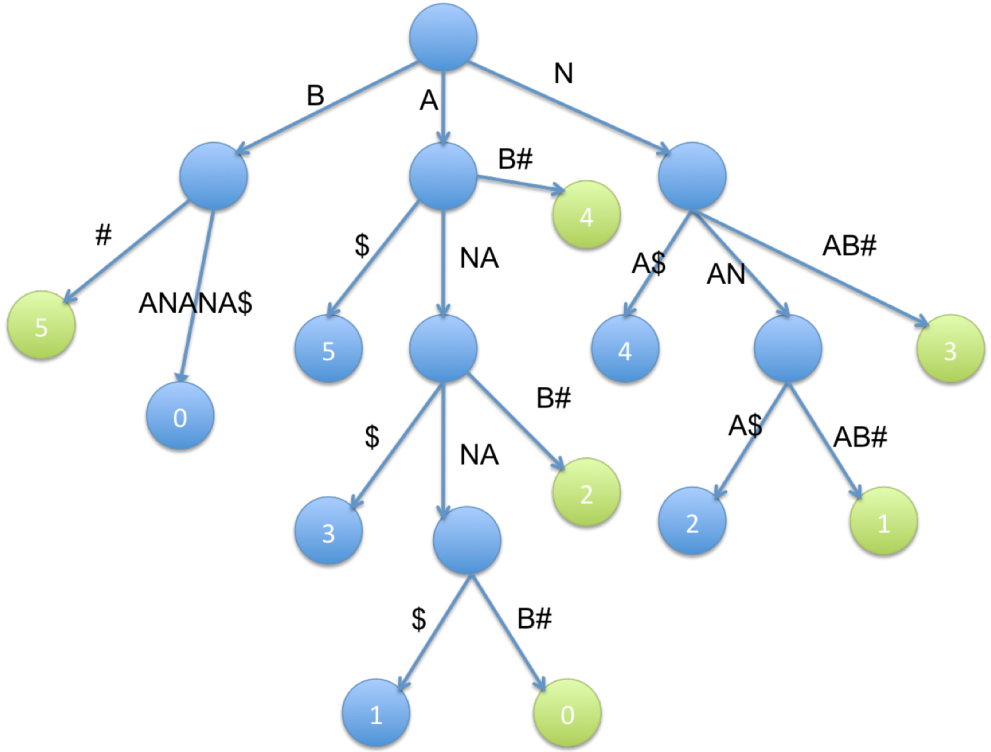
Question: Can we reconstruct the evolutionary history of different species?

Question: Can we recover how a tumor has evolved overtime?



https://scientificbsides.wordpress.com/2014/06/09/inferring-tumour-evolution-2-comparison-to-classical-phylogenetics/

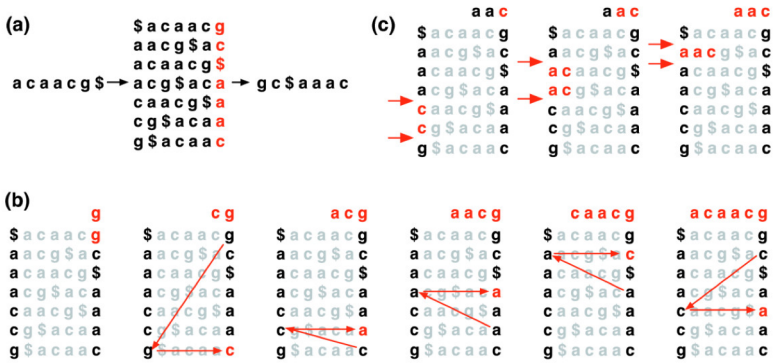
Course Topic #4: Pattern Matching



Suffix Trees

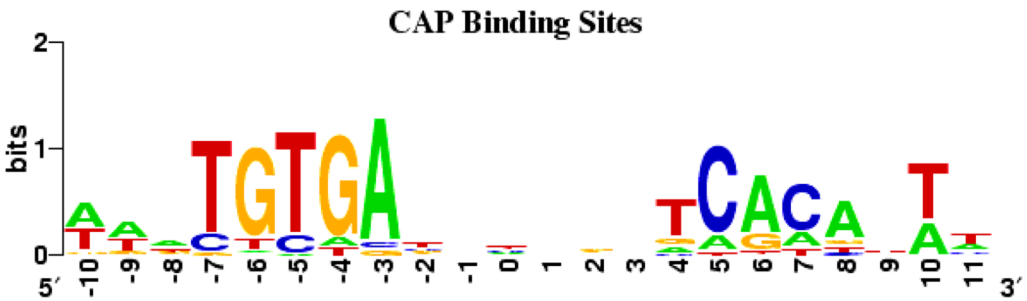
Question: How do we start to make sense of all these sequences?

Burrows Wheeler Transform

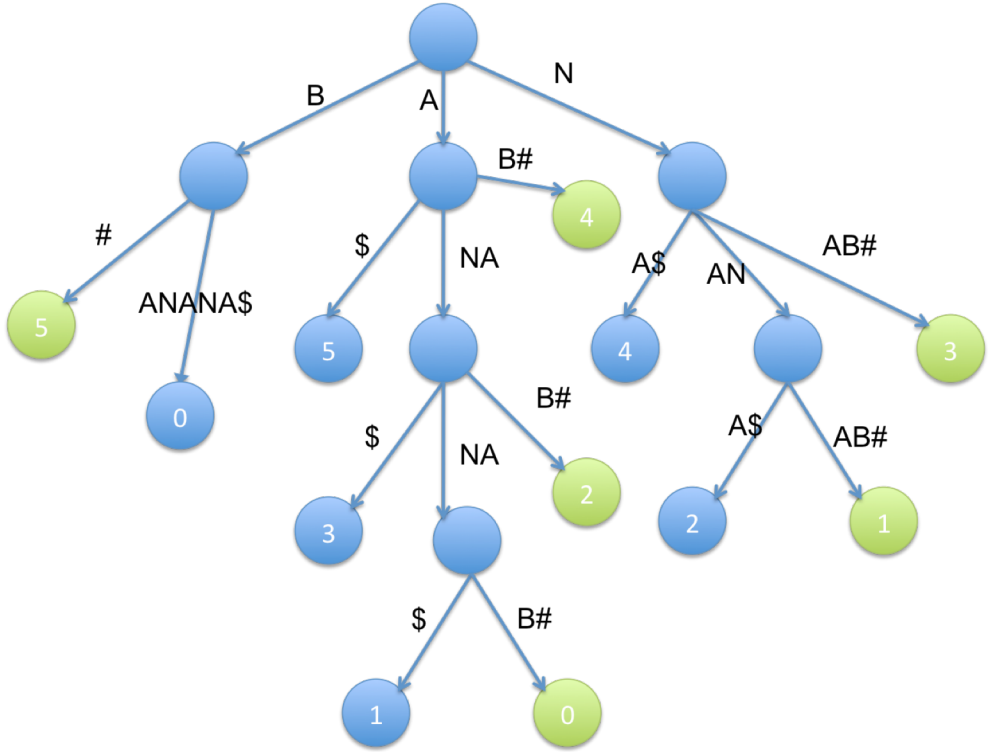


http://www.genomebiology.com/2009/10/3/R25/figure/F1?highres=y

Motif Finding



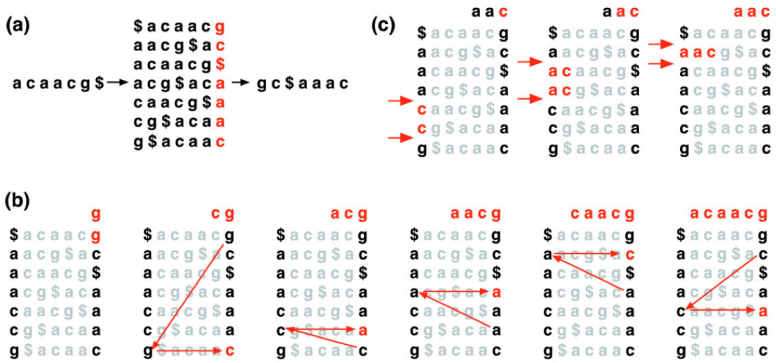
Course Topic #4: Pattern Matching



Suffix Trees

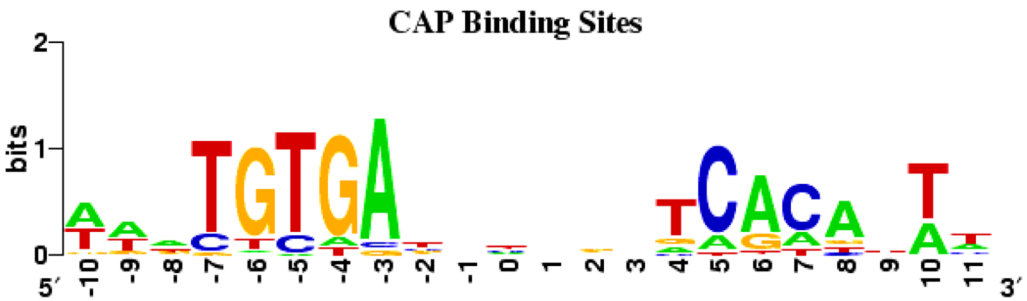
Question: How do we start to make sense of all these sequences?

Burrows Wheeler Transform

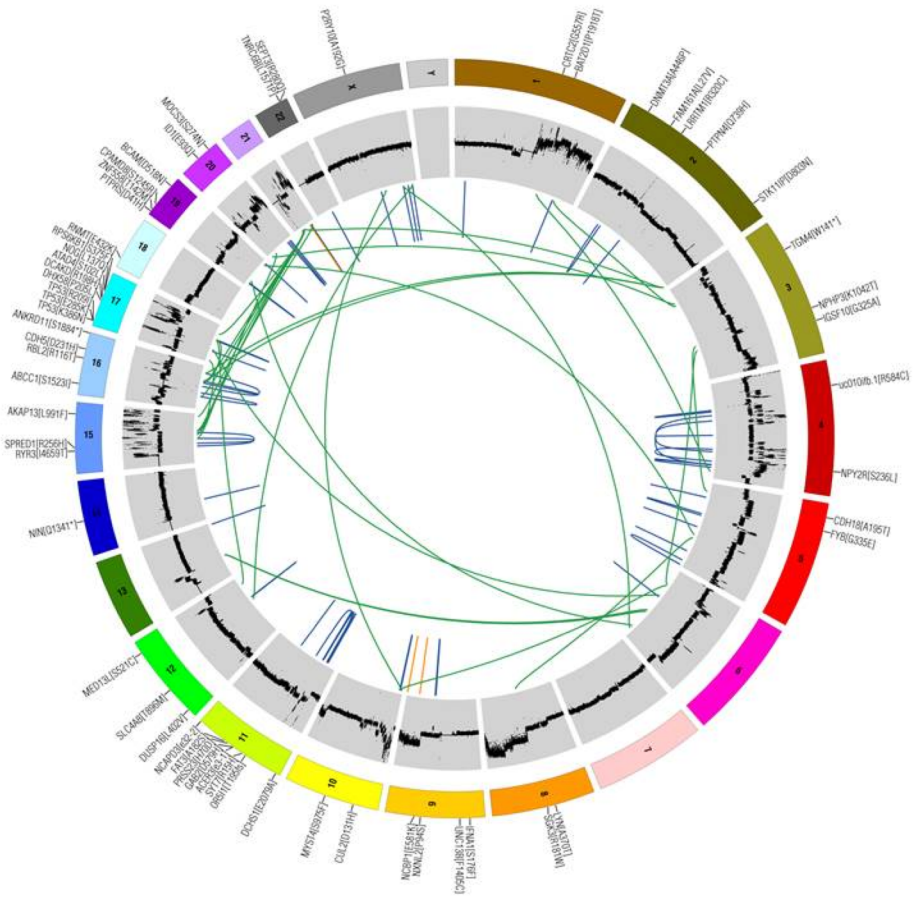


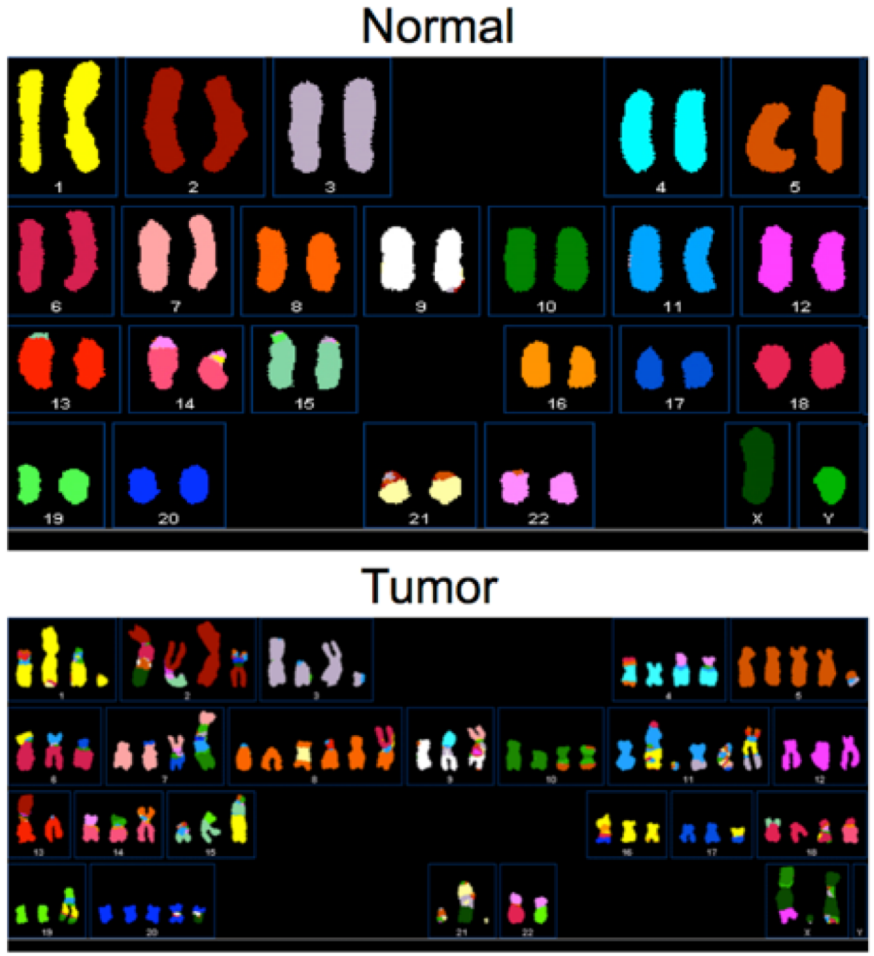
http://www.genomebiology.com/2009/10/3/R25/figure/F1?highres=y

Motif Finding



Course Topic #5: Cancer Genomics





Question: How can we analyze available data to determine what drives tumor growth and how to treat or prevent it?

Course Topics

1. Sequence alignment
   'How do we compare two genes/genomes?'

2. Genome assembly
   'How do we put all the pieces back together?'

3. Phylogenetics
   'What is the evolutionary history of different sequences?'

4. Pattern matching
   'How do we start to make sense out of all these sequences?'

5. Cancer genomics
   'How do we identify what drives tumor growth and how to treat/prevent it?'

Course Topics

1. Sequence alignment
   Dynamic programming: edit distance

2. Genome assembly
   Graphs: de Bruijn graph, Eulerian and Hamiltonian paths

3. Phylogenetics
   Trees and distances: distance matrices, neighbor joining, hierarchical clustering.
   Phylogenies: Sankoff/Fitch algorithms, perfect phylogeny and compatibility

4. Pattern matching
   Suffix trees/arrays. Burrows-Wheeler transform, Hidden Markov Models (HMMs)

5. Cancer genomics
   Cancer phylogenies: Integer linear optimization and graph algorithms

Problem != Algorithm

Problem Π with instance X and solution set $\Pi(X)$:

- Decision problem:
  - Is $\Pi(X) = \emptyset$?
- Optimization problem:
  - Find $y^* \in \Pi(X)$ s.t. $f(y^*)$ is optimum.
- Counting problem:
  - Compute $|\Pi(X)|$.
- Sampling problem:
  - Sample uniformly from $\Pi(X)$.
- Enumeration problem:
  - Enumerate all solutions in $\Pi(X)$

Algorithms:

Set of instructions for solving problem.

- Exact
- Heuristic

The Change Problem

- Suppose we have three coins:

- What is the minimum number of coins needed to make change for ***M*** cents?

The Change Problem

- Suppose we have three coins:

$\mathbf{c} = ($ 5 cent , 3 cent , 1 cent $)$

- What is the minimum number of coins needed to make change for ***M*** cents?

Change Problem: Given coins $\mathbf{c} = (c_1, \dots, c_n)$ and amount M,
find $\mathbf{d} = (d_1, \dots, d_n) \in \mathbb{N}^n$ such that:
(i) $M = \sum_{i=1}^{n} c_i d_i$ and (ii) $\sum_{i=1}^{n} d_i$ is minimum.

Idea #1: Choose largest coin possible

GreedyChange($M, c_1, \ldots, c_n$)

1. **while** $M > 0$
2. $\quad x \leftarrow$ largest coin c_i possible such that $c_i \leq M$
3. $\quad M \leftarrow M - x$

Idea #1: Choose largest coin possible

GreedyChange($M, c_1, \ldots, c_n$)

1. **while** $M > 0$
2. $\quad x \leftarrow$ largest coin c_i possible such that $c_i \leq M$
3. $\quad M \leftarrow M - x$

Is this a good algorithm?

Idea #1: Choose largest coin possible

GreedyChange($M, c_1, \ldots, c_n$)

1. **while** $M > 0$
2. $\quad x \leftarrow$ largest coin c_i possible such that $c_i \leq M$
3. $\quad M \leftarrow M - x$

Is this a good algorithm? Two properties of a good algorithm:

Correctness: gives the correct output for any input.

- Seem to work for $\mathbf{c} = (5, 3, 1)$.
- But what about $\mathbf{c} = (5, 4, 1)$ and $M = 8$?

Efficient: *running time* of the algorithm does not increase to rapidly with input size.

Idea #2: Don't be smart, apply brute force

Change Problem: Given coins $\mathbf{c} = (c_1, \dots, c_n)$ and amount M,
find $\mathbf{d} = (d_1, \dots, d_n) \in \mathbb{N}^n$ such that:
(i) $M = \sum_{i=1}^{n} c_i d_i$ and (ii) $\sum_{i=1}^{n} d_i$ is minimum.

$\mathbf{c} = ($ 5 cent , 4 cent , 1 cent $)$

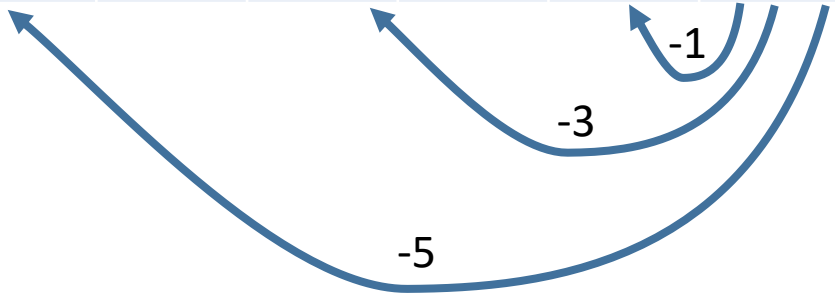
- Check all possible solutions:
  - 11 = 5 + 5 + 1
  - 11 = 5 + 4 + 1 + 1
  - 11 = 5 + 1 + 1 + 1 + 1 + 1 + 1
  - 11 = 4 + 4 + 1 + 1 + 1
  - …

Correct? yes
Efficient? no

Idea #3: Recursion

$\mathbf{c} = ($ 5 cent , 3 cent , 1 cent $)$

Value	1	2	3	4	5	6	7	8	9	10	11
Min # coins	?	?	?	?	?	?	?	?	?	?	?



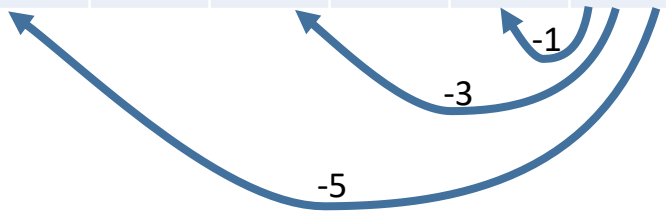
Optimal substructure:

Optimal solution is obtained from optimal solutions of subproblems

Idea #3: Recursion

$\mathbf{c} = ($ 5 , 3 , 1 $)$

Value	1	2	3	4	5	6	7	8	9	10	11
Min # coins	?	?	?	?	?	?	?	?	?	?	?



- This example can be expressed using a recurrence relation
- Let minNumCoins(M) be the minimum number of coins to make change for M cents

$$\text{minNumCoins}(M) = \min \begin{cases} \text{minNumCoins}(M-1)+1, \\ \text{minNumCoins}(M-3)+1, \\ \text{minNumCoins}(M-5)+1. \end{cases}$$

Idea #3: Recursion

Change Problem: Given coins $\mathbf{c} = (c_1, \dots, c_n)$ and amount M,
find $\mathbf{d} = (d_1, \dots, d_n) \in \mathbb{N}^n$ such that:
(i) $M = \sum_{i=1}^{n} c_i d_i$ and (ii) $\sum_{i=1}^{n} d_i$ is minimum.

$$\text{minNumCoins}(M) = \min \begin{cases} \text{minNumCoins}(M - c_1) + 1, \\ \text{minNumCoins}(M - c_2) + 1, \\ \dots \\ \text{minNumCoins}(M - c_n) + 1. \end{cases}$$

Idea #3: Recursion

Given coins $\mathbf{c} = (1, 3, 7)$ and amount $M = 77$, find $\mathbf{d} = (d_1, \ldots, d_n) \in \mathbb{N}^n$ such that: (i) $M = \sum_{i=1}^{n} c_i d_i$ and (ii) $\sum_{i=1}^{n} d_i$ is minimum.

$$\text{minNumCoins}(77) = \min \begin{cases} \text{minNumCoins}(77-1)+1, \\ \text{minNumCoins}(77-3)+1, \\ \text{minNumCoins}(77-7)+1, \end{cases}$$

$$\text{minNumCoins}(76) = \min \begin{cases} \text{minNumCoins}(76-1)+1, \\ \text{minNumCoins}(76-3)+1, \\ \text{minNumCoins}(76-7)+1, \end{cases}$$

$$\vdots$$

$$\text{minNumCoins}(7) = 1$$

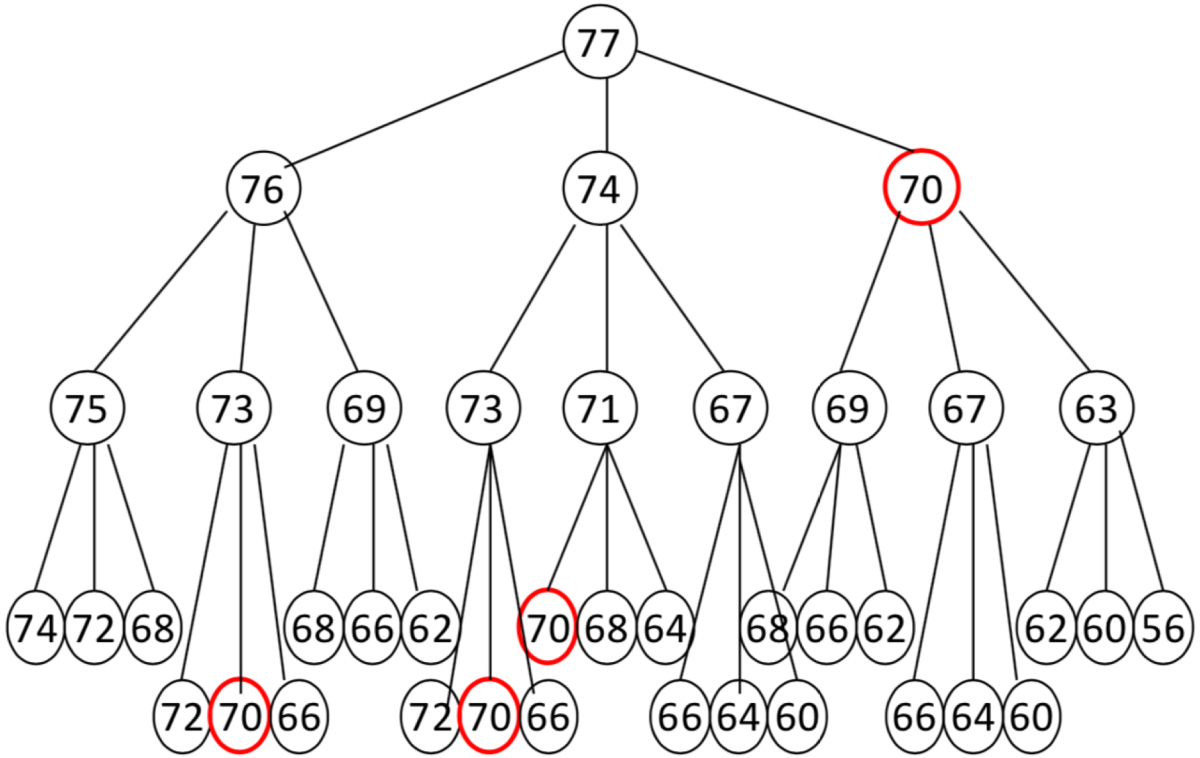
$$\text{minNumCoins}(3) = 1$$

$$\text{minNumCoins}(1) = 1$$

Idea #3: Recursion

RecursiveChange($M, c_1, \ldots, c_n$)

1. **if** $M = 0$
2. **return** 0
3. bestNumCoins ← ∞
4. **for** i ← 1 **to** n
5. **if** $M \geq c_i$
6. numCoins ← RecursiveChange($M - c_i, c_1, \ldots, c_n$)
7. **if** numCoins + 1 < bestNumCoins
8. bestNumCoins ← numCoins + 1
9. **return** bestNumCoins

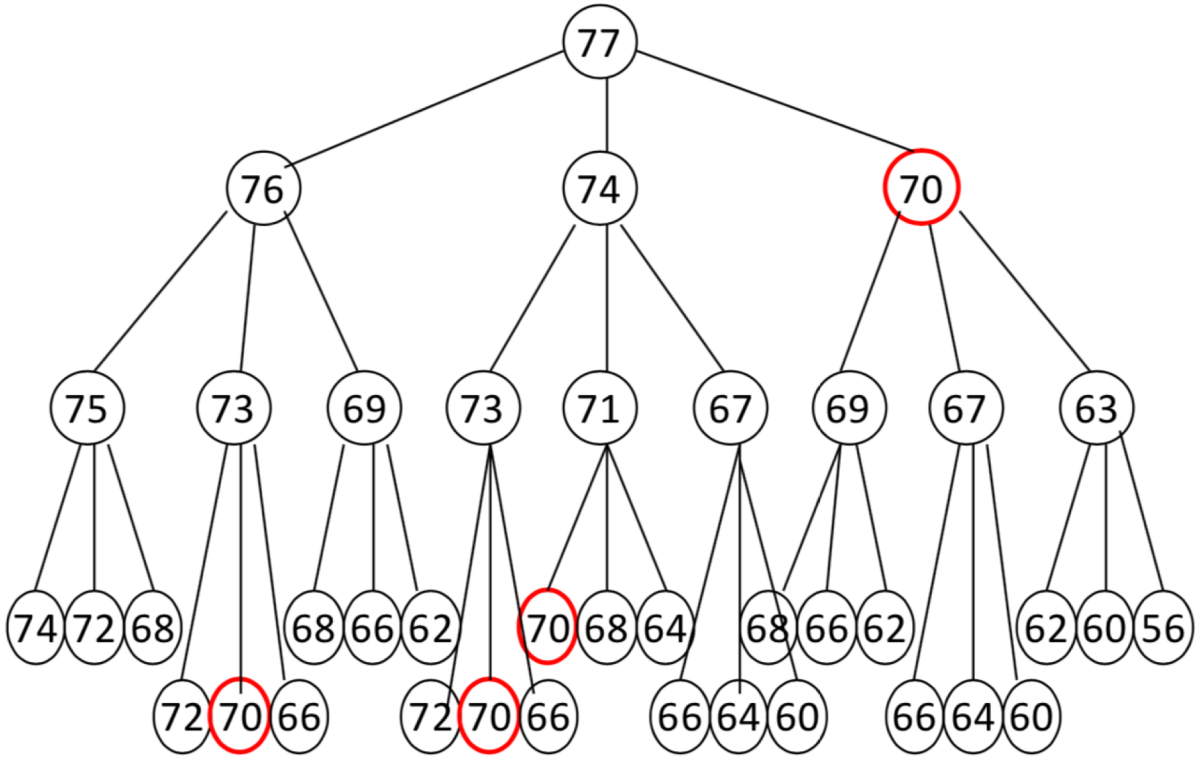


Correct but inefficient:
Same subproblem is solved many times!

Idea #3: Recursion

RecursiveChange($M, c_1, \dots, c_n$)

1. **if** $M = 0$
2. **return** 0
3. bestNumCoins ← ∞
4. **for** i ← 1 **to** n
5. **if** $M \geq c_i$
6. numCoins ← RecursiveChange($M - c_i, c_1, \dots, c_n$)
7. **if** numCoins + 1 < bestNumCoins
8. bestNumCoins ← numCoins + 1
9. **return** bestNumCoins



Correct but inefficient:
Same subproblem is solved many times!

Solutions:
- Remember previously computed values: memoization
  - Bottom up computation: dynamic programming

Idea #4: Solve recurrence with dynamic programming

Fill in table "bottom up": from smallest to largest.

$\mathbf{c} = ($ 5 , 3 , 1 $)$

Value	1	2	3	4	5	6	7	8	9	10	11
Min # coins	**1**		**1**		**1**						

$$\text{minNumCoins}(M) = \min \begin{cases} \text{minNumCoins}(M-1)+1, \\ \text{minNumCoins}(M-3)+1, \\ \text{minNumCoins}(M-5)+1. \end{cases}$$

Only one coin is needed to make change for the values 1, 3 and 5

Idea #4: Solve recurrence with dynamic programming

Fill in table "bottom up": from smallest to largest.

$\mathbf{c} = ($ 5 , 3 , 1 $)$

Value	1	2	3	4	5	6	7	8	9	10	11
Min # coins	1	2	1	2	1	2					

$$\text{minNumCoins}(M) = \min \begin{cases} \text{minNumCoins}(M-1)+1, \\ \text{minNumCoins}(M-3)+1, \\ \text{minNumCoins}(M-5)+1. \end{cases}$$

Two coins are needed to make change for the values 2, 4 and 6

Idea #4: Solve recurrence with dynamic programming

Fill in table "bottom up": from smallest to largest.

$\mathbf{c} = ($ 5 , 3 , 1 $)$

Value	1	2	3	4	5	6	7	8	9	10	11
Min # coins	**1**	**2**	**1**	**2**	**1**	**2**	**3**				

$$\text{minNumCoins}(M) = \min \begin{cases} \text{minNumCoins}(M-1)+1, \\ \text{minNumCoins}(M-3)+1, \\ \text{minNumCoins}(M-5)+1. \end{cases}$$

Three coins are needed to make change for the value 7

Idea #4: Solve recurrence with dynamic programming

Fill in table "bottom up": from smallest to largest.

$\mathbf{c} = ($ 5 , 3 , 1 $)$

Value	1	2	3	4	5	6	7	8	9	10	11
Min # coins	**1**	**2**	**1**	**2**	**1**	**2**	**3**	**2**			

$$\text{minNumCoins}(M) = \min \begin{cases} \text{minNumCoins}(M-1)+1, \\ \text{minNumCoins}(M-3)+1, \\ \text{minNumCoins}(M-5)+1. \end{cases}$$

Optimal substructure: Optimal solution obtained from optimal subsolutions

Idea #4: Solve recurrence with dynamic programming

Fill in table "bottom up": from smallest to largest.

$\mathbf{c} = ($ 5 , 3 , 1 $)$

Value	1	2	3	4	5	6	7	8	9	10	11
Min # coins	**1**	**2**	**1**	**2**	**1**	**2**	**3**	**2**	**3**	**2**	**3**

$$\text{minNumCoins}(M) = \min \begin{cases} \text{minNumCoins}(M-1)+1, \\ \text{minNumCoins}(M-3)+1, \\ \text{minNumCoins}(M-5)+1. \end{cases}$$

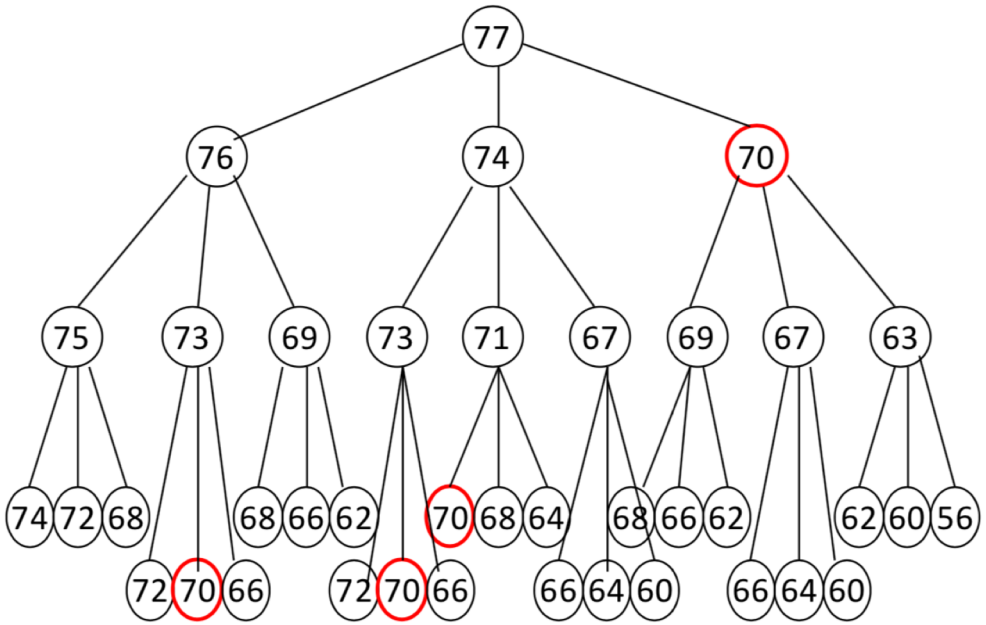
Optimal substructure: Optimal solution obtained from optimal subsolutions

Idea #4: Solve recurrence with dynamic programming

Change Problem: Given coins $\mathbf{c} = (c_1, \ldots, c_n)$ and amount M,
find $\mathbf{d} = (d_1, \ldots, d_n) \in \mathbb{N}^n$ such that:
(i) $M = \sum_{i=1}^{n} c_i d_i$ and (ii) $\sum_{i=1}^{n} d_i$ is minimum.

DPChange($M, c_1, \ldots, c_n$)

1. **for** $m \leftarrow 1$ **to** M
2. minNumCoins[c_i] $\leftarrow \infty$
3. **for** $i \leftarrow 1$ **to** n
4. minNumCoins[c_i] $\leftarrow 1$
5. **for** $m \leftarrow 1$ **to** M
6. minNumCoins[m] $\leftarrow 1 + \min_{i=1}^{n}\{\text{minNumCoins}[m - c_i]\}$
7. **return** minNumCoins[M]



Correct? yes
Efficient? yes

Different algorithm techniques

Change Problem: Given coins $\mathbf{c} = (c_1, \dots, c_n)$ and amount M, find $\mathbf{d} = (d_1, \dots, d_n) \in \mathbb{N}^n$ such that:
(i) $M = \sum_{i=1}^{n} c_i d_i$ and (ii) $\sum_{i=1}^{n} d_i$ is minimum.

Technique	Correct?	Efficient?
Greedy algorithm [GreedyChange]	no	yes
Exhaustive enumeration [ExhaustiveChange]	yes	no
Recursive algorithm [RecursiveChange]	yes	no
Dynamic programming [DPChange]	yes	yes

Summary

- DNA, RNA and proteins are sequences
  - Central dogma of molecular biology: DNA -> RNA -> protein
- Problem != algorithm
- Different algorithm techniques
  - Greedy
  - Exhaustive search/brute force
  - Recursive algorithm
  - Dynamic programming algorithm
- Reading:
  - “Biology for Computer Scientists” by Lawrence Hunter (http://www.el-kebir.net/teaching/CS466/Hunter_BIO_CS.pdf)
  - Jones and Pevzner: Chapters 2.1, 2.3, 2.4, 6.2

Sources

- CS 362 by Layla Oesper (Carleton College)
- CS 1810 by Ben Raphael (Brown/Princeton University)
- An Introduction to Bioinformatics Algorithms book (Jones and Pevzner)