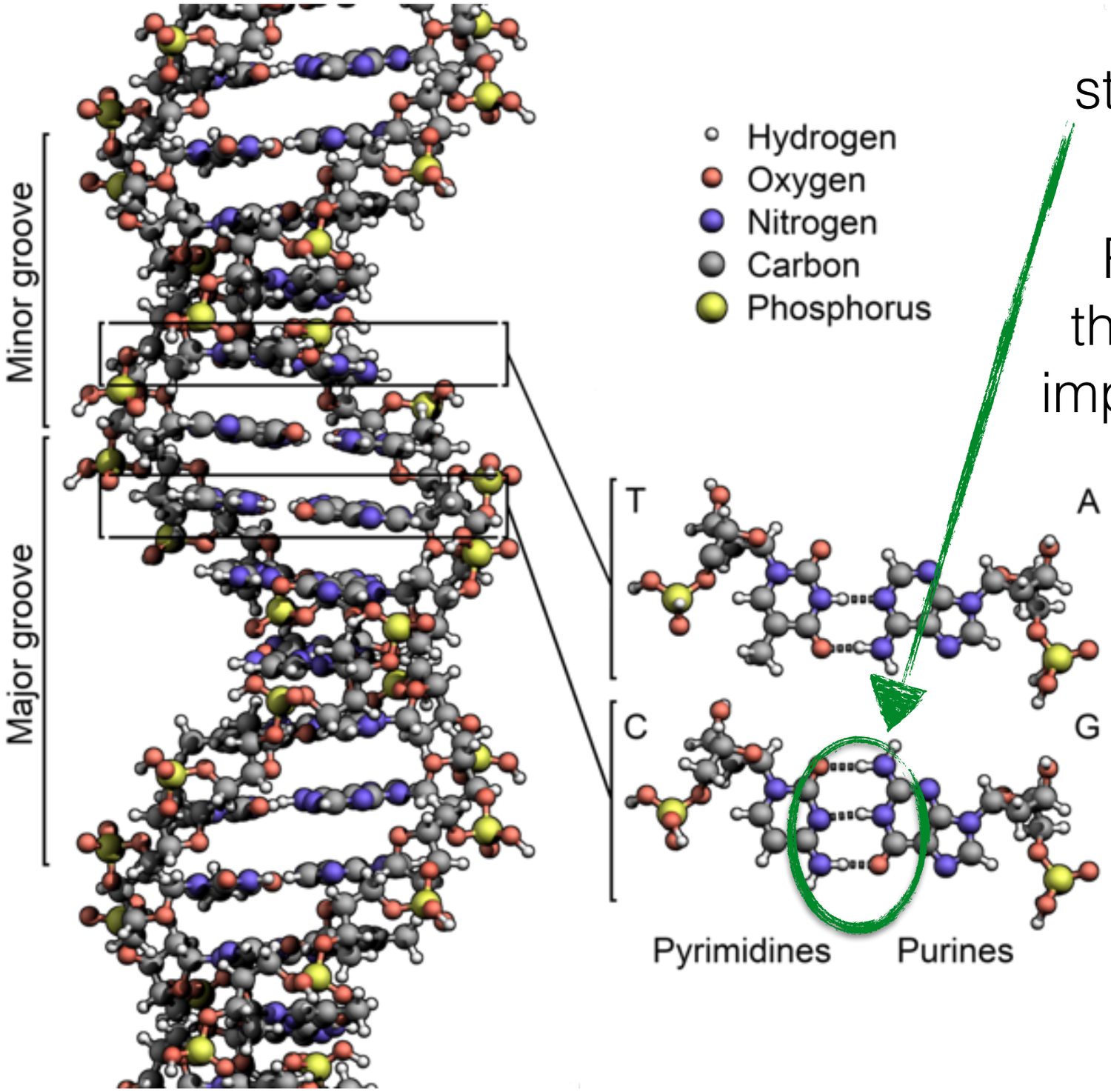
DNA (the genome)



G-C pairing generally stronger than A-T pairing

Ratio of G+C bases — the “GC content” — is an important sequence feature

DNA (the genome)

gene — will go on to become a protein

“non-coding DNA” — may or may not produce transcripts (e.g. functional non-coding RNA)

In humans, most DNA is “non-coding” ~98%

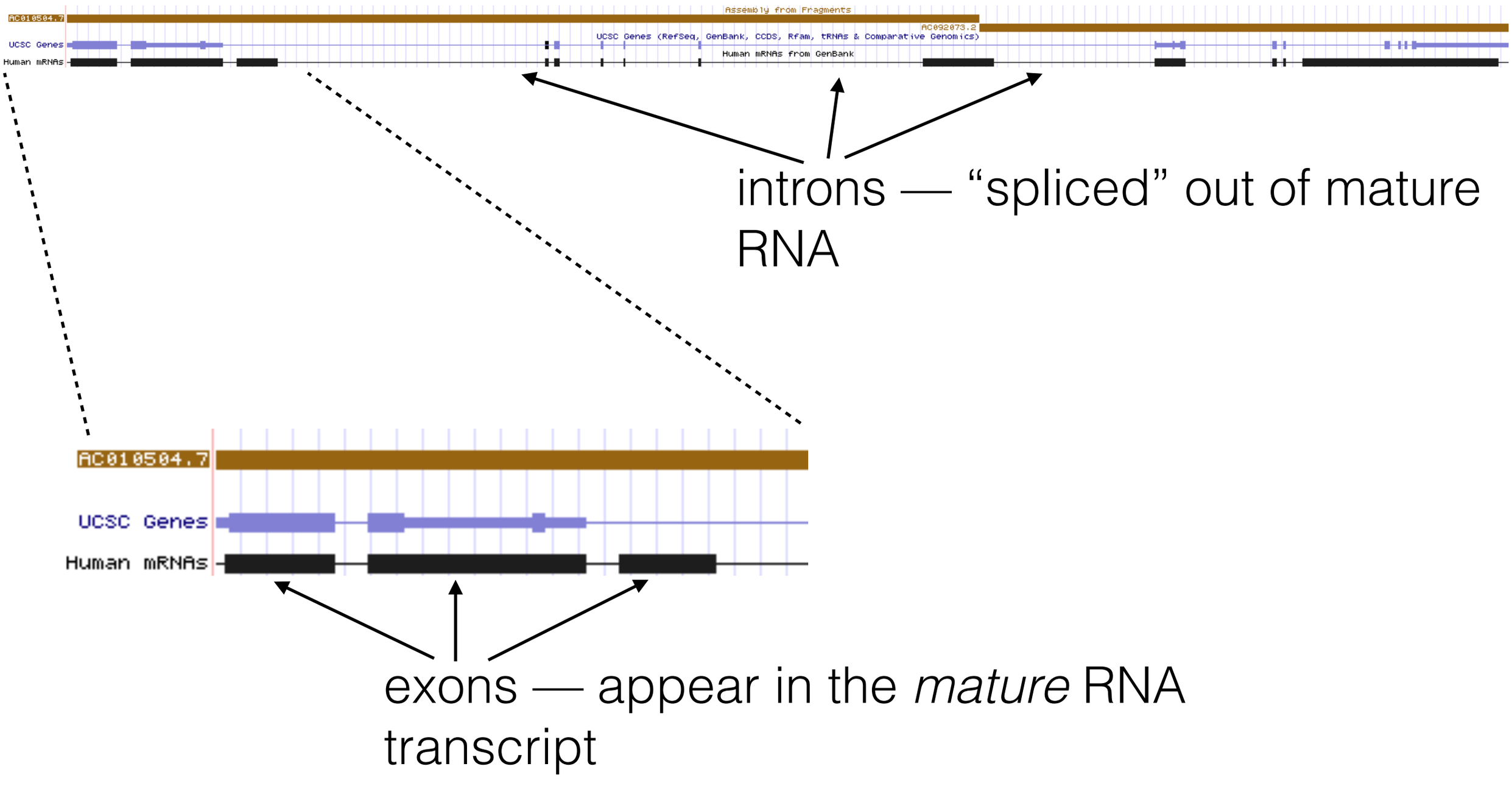
In typical bacterial genome, only small fraction — ~2% — of DNA is “non-coding”

Sometimes referred to as “junk” DNA — much is not, in any way, “junk”

DNA (the genome)

In **prokaryotes**, genes are typically contiguous DNA segment

In **eukaryotes**, genes can have complex structure

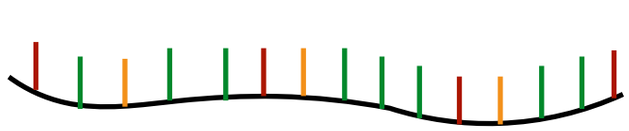


“Flow” of information in the cell

DNA

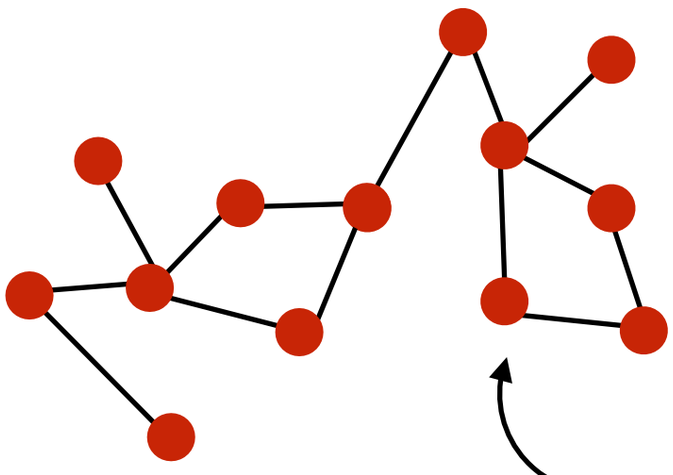
RNA Polymerase
(transcription)

RNA



Ribosomes
(translation)

Protein



Form networks &
pathways; perform a
vast set of cellular
functions

DNA → RNA : Transcription

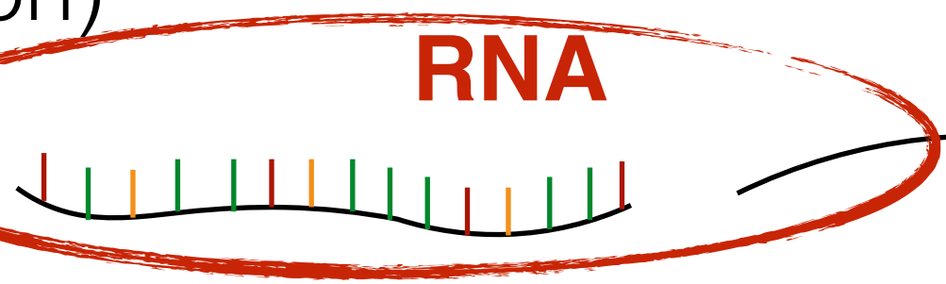
https://www.dnalc.org/resources/3d/12-transcription-basic.html

“Flow” of information in the cell

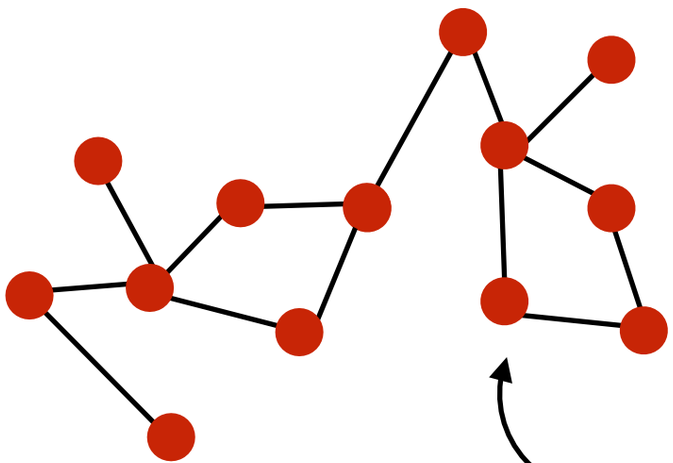
DNA

RNA Polymerase
(transcription)

RNA



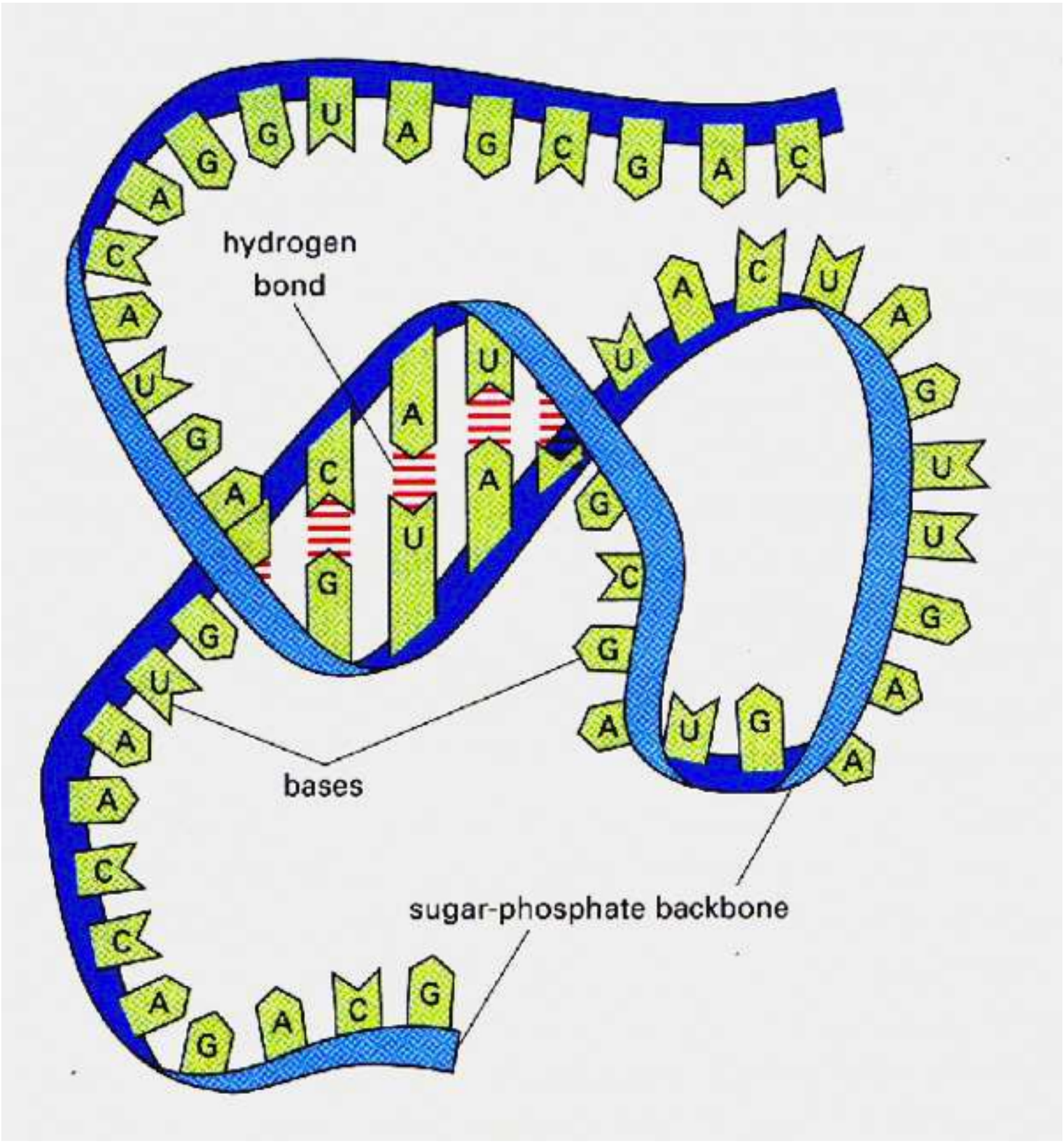
Ribosomes
(translation)



Form networks & pathways; perform a vast set of cellular functions

Protein

RNA



Less regular structure than DNA

Generally a single-stranded molecule

Secondary & tertiary structure can affect function

Act as transcripts for protein, but also perform important functions themselves

Same “alphabet” as DNA, except thymine replaced by uracil

http://tigger.uic.edu/classes/phys/phys461/phys450/ANJUM04/RNA_sstrand.jpg

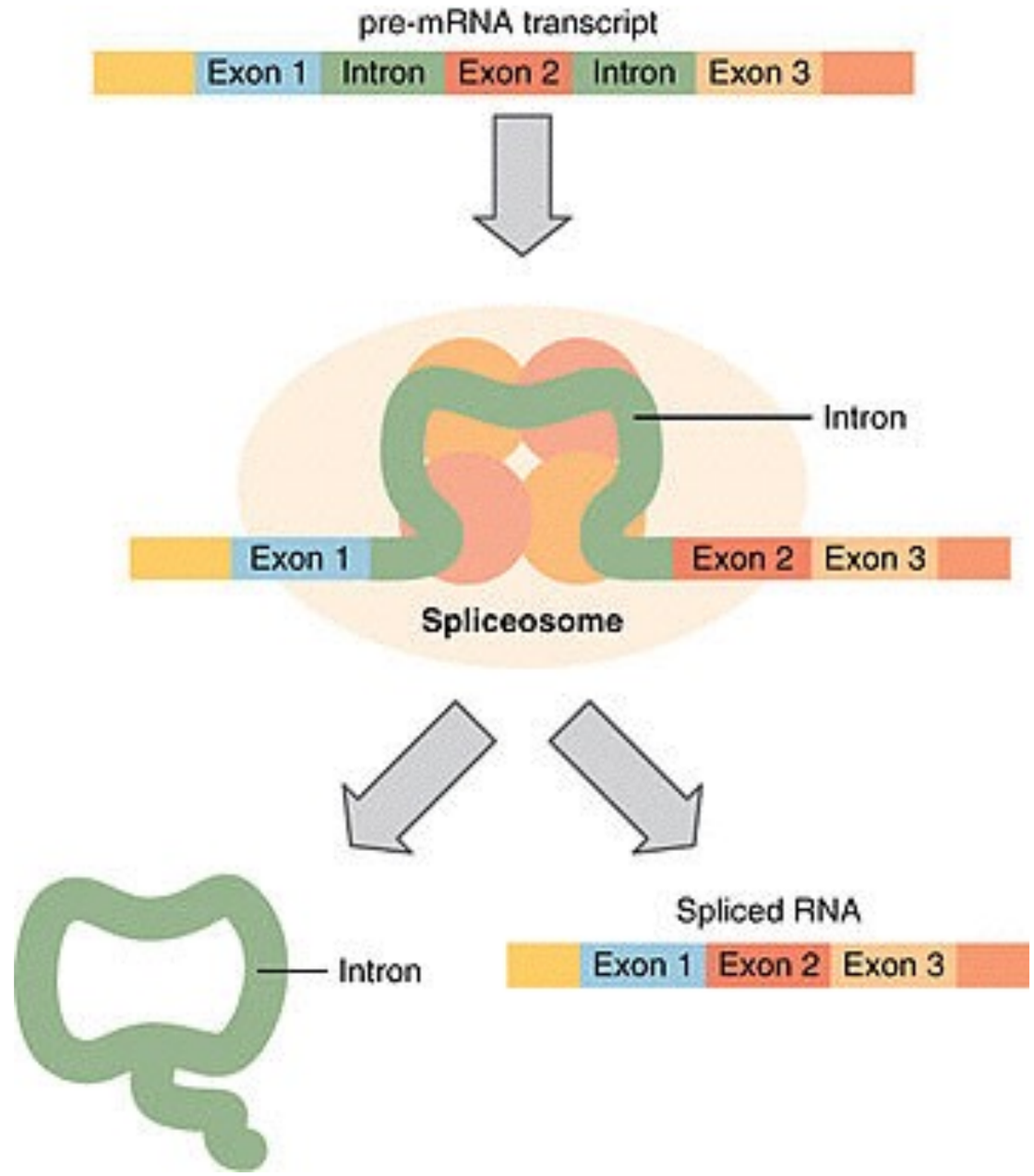
RNA Splicing

DNA transcribed into pre-mRNA

Some “processing occurs”
capping & **polyadenylation**

Introns removed from pre-mRNA

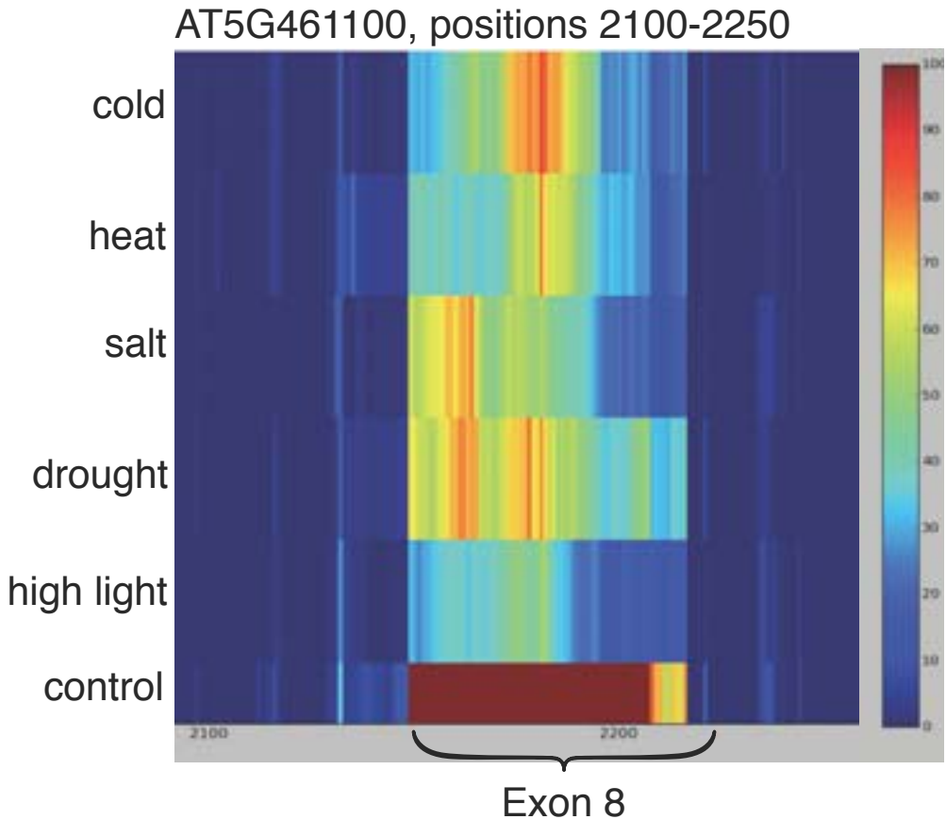
Introns removed resulting in
mature mRNA

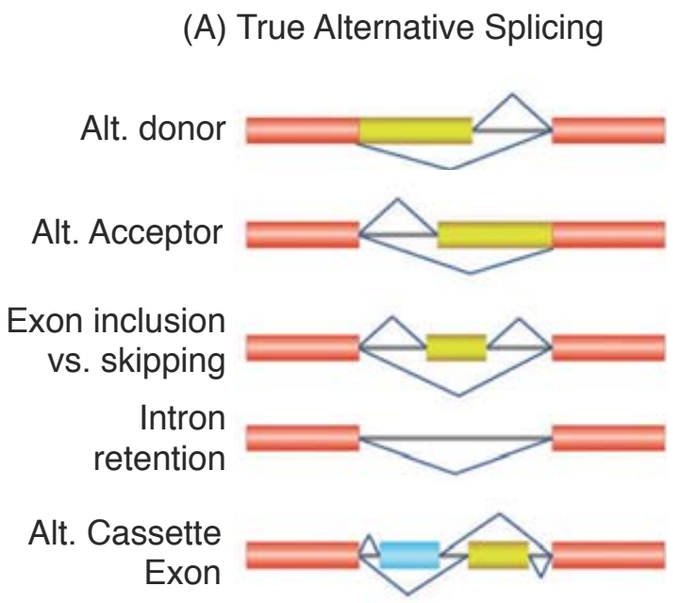


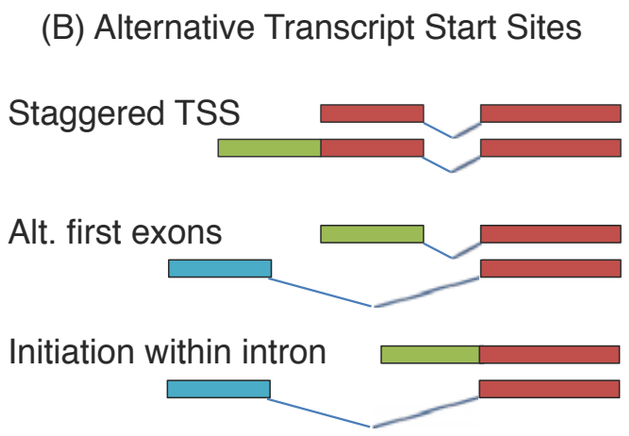
en.wikipedia.org

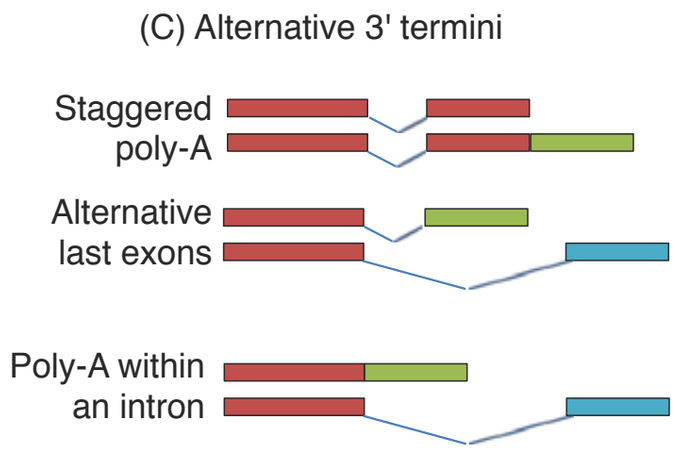
Alternative Splicing & Isoform Expression

- Expression of genes can be measured via RNA-seq (sequencing transcripts)
- Sequencing gives you short (35-300bp length reads)









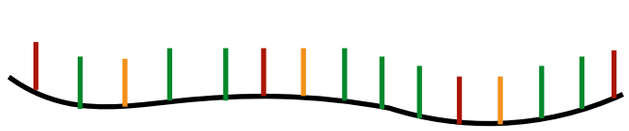
slide courtesy of Carl Kingsford

“Flow” of information in the cell

DNA

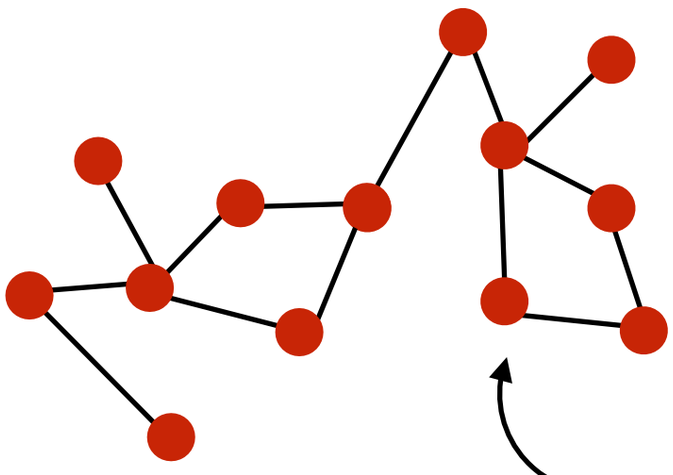
RNA Polymerase
(transcription)

RNA



Ribosomes
(translation)

Protein



Form networks &
pathways; perform a
vast set of cellular
functions

mRNA→ Protein: Translation

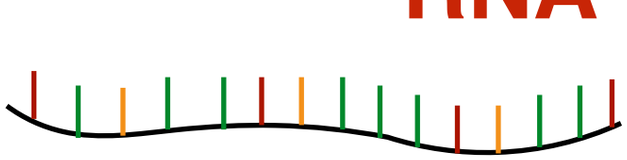
https://www.dnalc.org/resources/3d/15-translation-basic.html

“Flow” of information in the cell

DNA

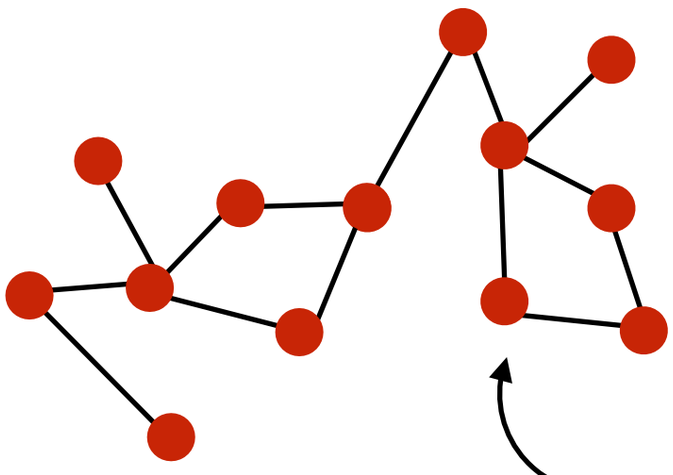
RNA Polymerase
(transcription)

RNA



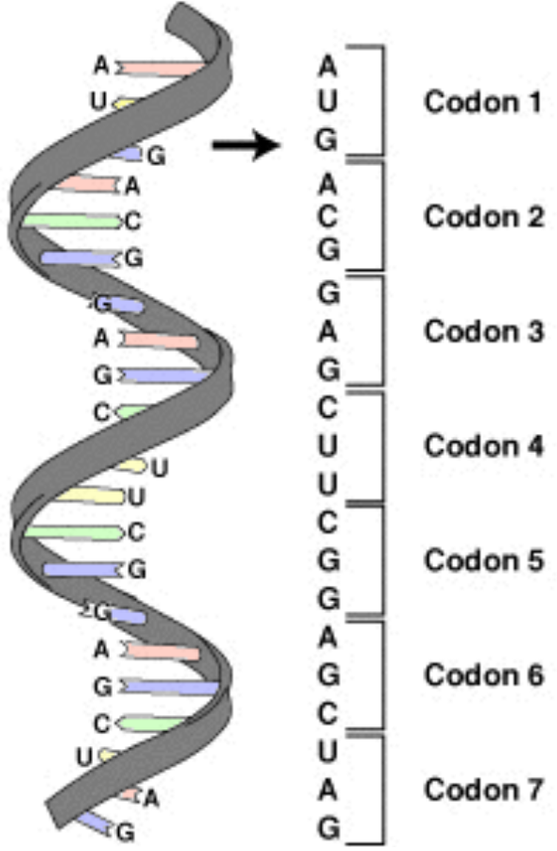
Ribosomes
(translation)

Protein



Form networks &
pathways; perform a
vast set of cellular
functions

Protein



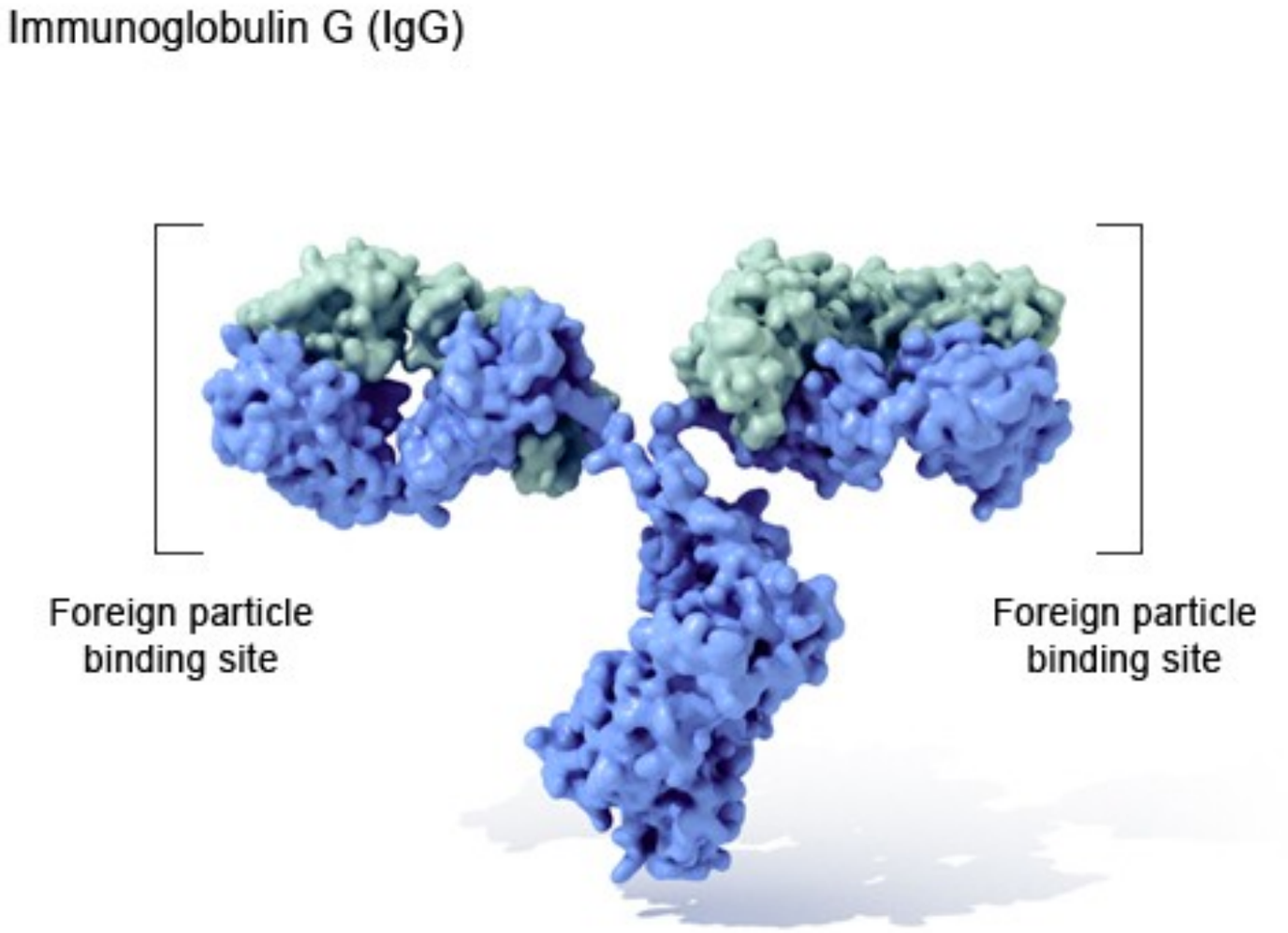
Triplets of mRNA bases (codons) correspond to specific amino acids

This mapping is known as the “genetic code” — an *almost* law of molecular Biology

Inverse table (compressed using IUPAC notation)

Amino acid	Codons	Compressed	Amino acid	Codons	Compressed
Ala/A	GCU, GCC, GCA, GCG	GCN	**Leu/L**	UUA, UUG, CUU, CUC, CUA, CUG	YUR, CUN
Arg/R	CGU, CGC, CGA, CGG, AGA, AGG	CGN, MGR	**Lys/K**	AAA, AAG	AAR
Asn/N	AAU, AAC	AAY	**Met/M**	AUG	
Asp/D	GAU, GAC	GAY	**Phe/F**	UUU, UUC	UUY
Cys/C	UGU, UGC	UGY	**Pro/P**	CCU, CCC, CCA, CCG	CCN
Gln/Q	CAA, CAG	CAR	**Ser/S**	UCU, UCC, UCA, UCG, AGU, AGC	UCN, AGY
Glu/E	GAA, GAG	GAR	**Thr/T**	ACU, ACC, ACA, ACG	ACN
Gly/G	GGU, GGC, GGA, GGG	GGN	**Trp/W**	UGG	
His/H	CAU, CAC	CAY	**Tyr/Y**	UAU, UAC	UAY
Ile/I	AUU, AUC, AUA	AUH	**Val/V**	GUU, GUC, GUA, GUG	GUN
START	AUG		**STOP**	UAA, UGA, UAG	UAR, URA

Protein



Perform vast majority of intra & extra cellular functions

Can range from a few amino acids to *very* large and complex molecules

Can bind with other proteins to form protein complexes

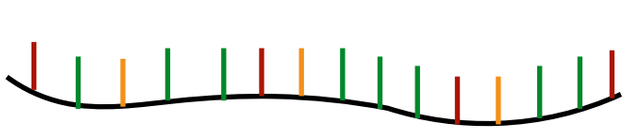
The shape or *conformation* of a protein is intimately tied to its function. Protein shape, therefore, is strongly conserved through evolution — even moreso than sequence. A protein can undergo sequence mutations, but fold into the same or a similar shape and still perform the same function.

“Flow” of information in the cell

DNA

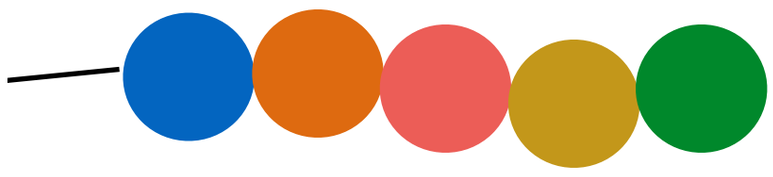
RNA Polymerase
(transcription)

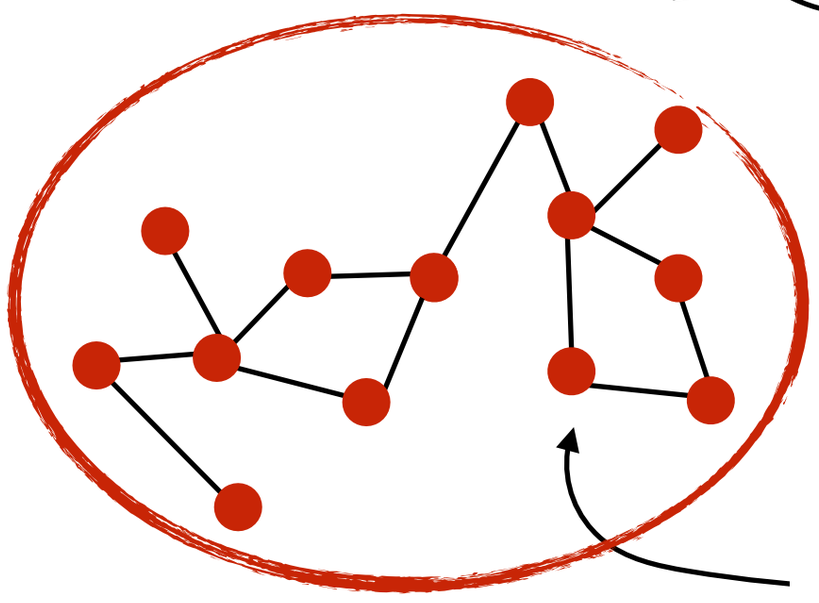
RNA



Ribosomes
(translation)

Protein





Form networks &
pathways; perform a
vast set of cellular
functions

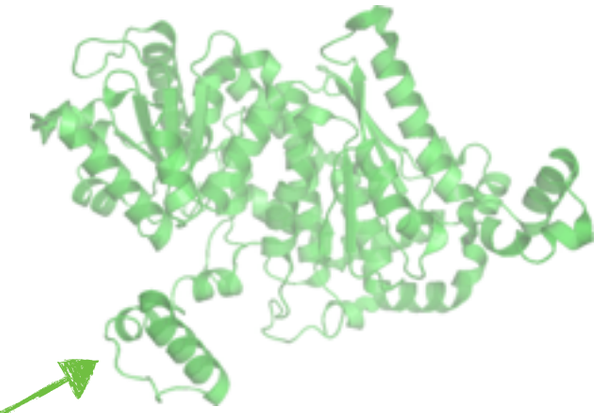
Glycolysis Pathway

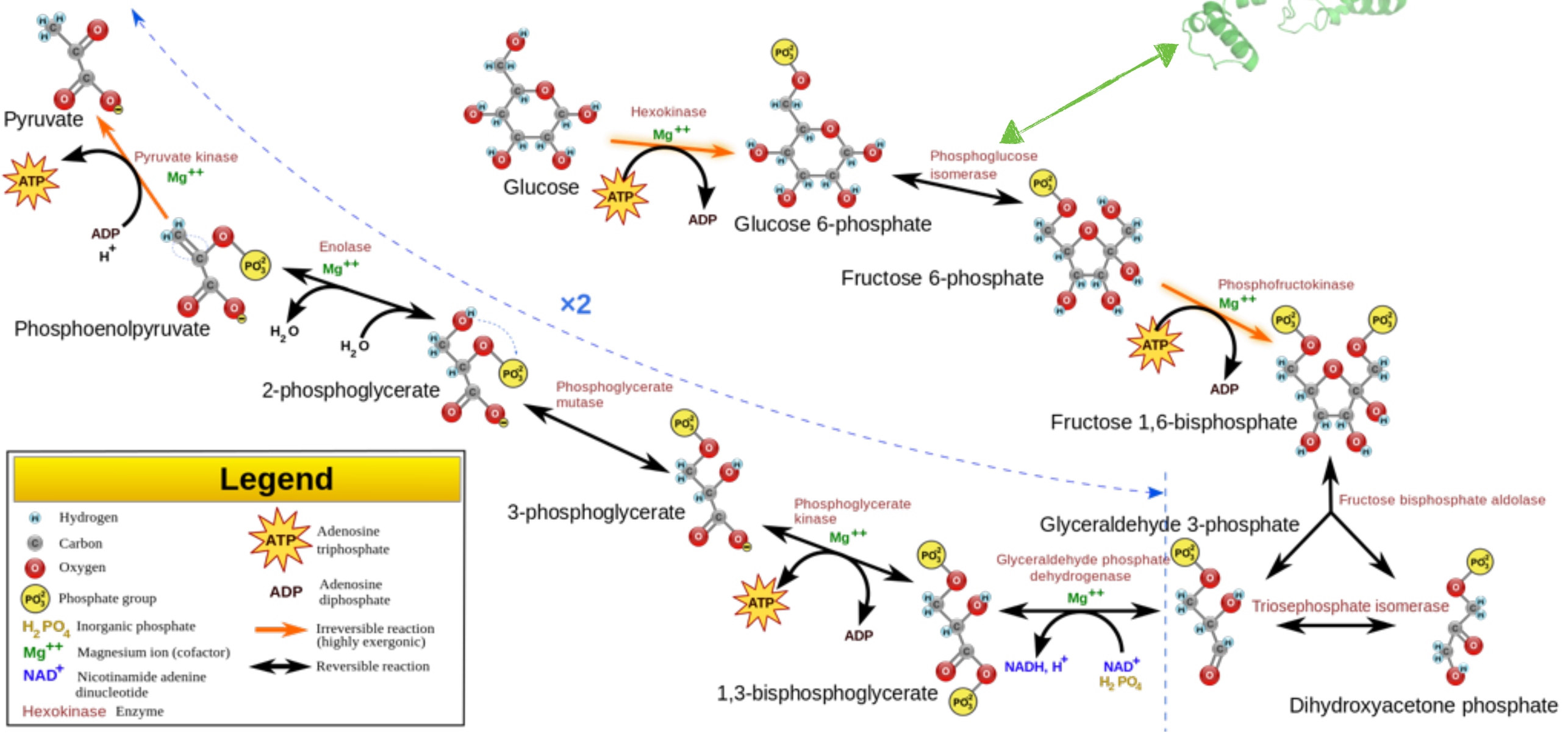
Converts glucose → pyruvate

Generates ATP (“energy currency” of the cell)

this is an **example**, no need to memorize this Bio.

phosphoglucose isomerase





Some Interesting Facts

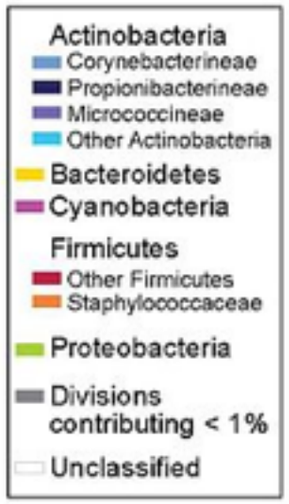
Organism	Genome size	# of genes
ϕX174 (*E. coli* virus)	~5kb	11
E. coli K-12	~4.6Mb	~4,300
Fruit Fly	~122Mb	~17,000
Human	~3.3Gb	~21,000
Mouse	~2.8Gb	~23,000
P. abies (a spruce tree)	~19.6Gb	~28,000

No strong link between genome size & phenotypic complexity

Plants can have **huge** genomes (adapt to environment while stationary!)

http://users.rcn.com/jkimball.ma.ultranet/BiologyPages/G/GenomeSizes.html

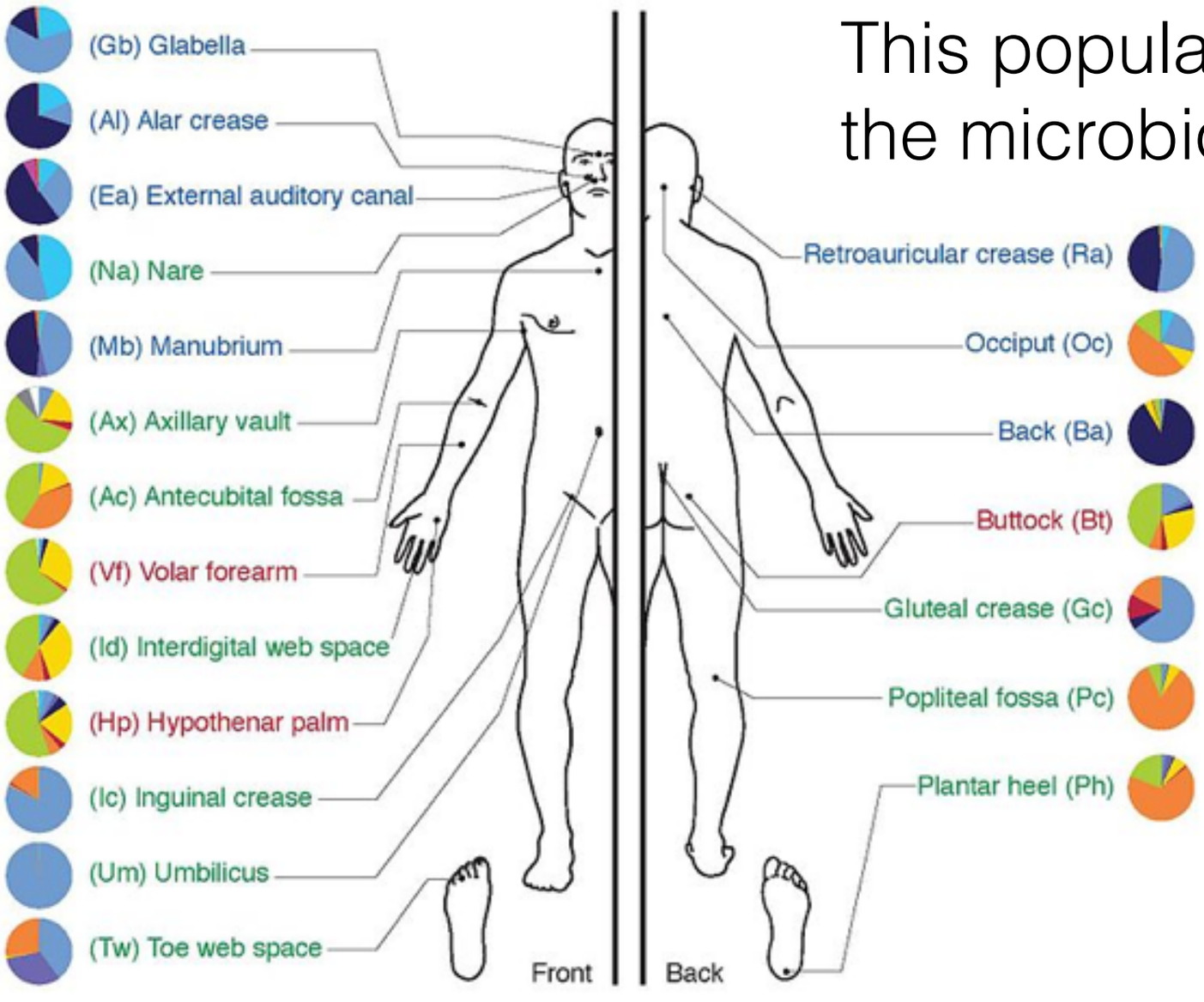
Some Interesting Facts



You are mostly bacteria, fungi & arches

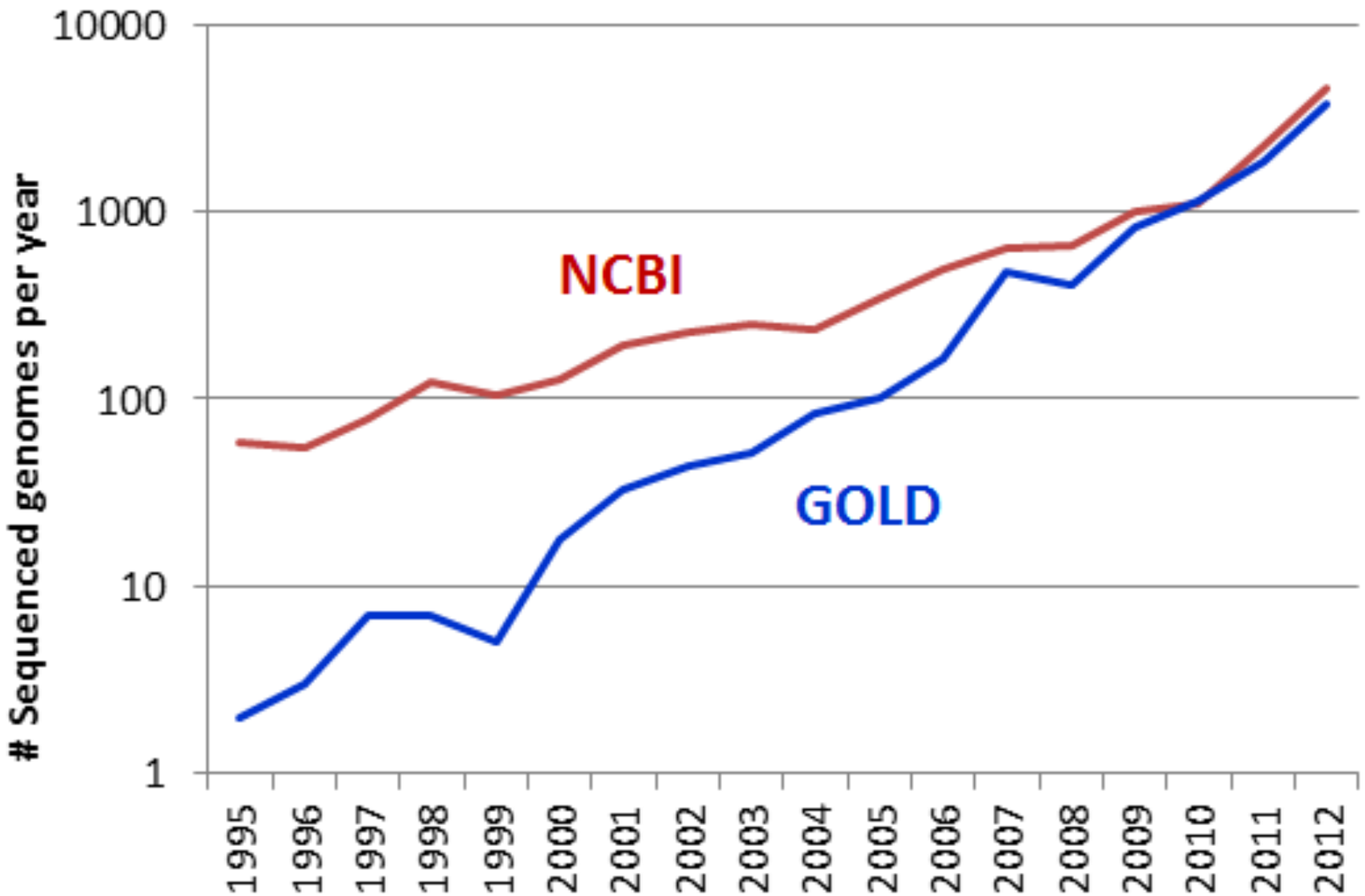
Non-human cells outnumber human cells ~10:1 in the human body

This population of organisms is called the microbiome



en.wikipedia.org : public domain

Some Interesting Facts



http://figshare.com/articles/Sequenced_Genomes_by_Year/715898

. . . Out of 8.7 ± 1.3 Mil*

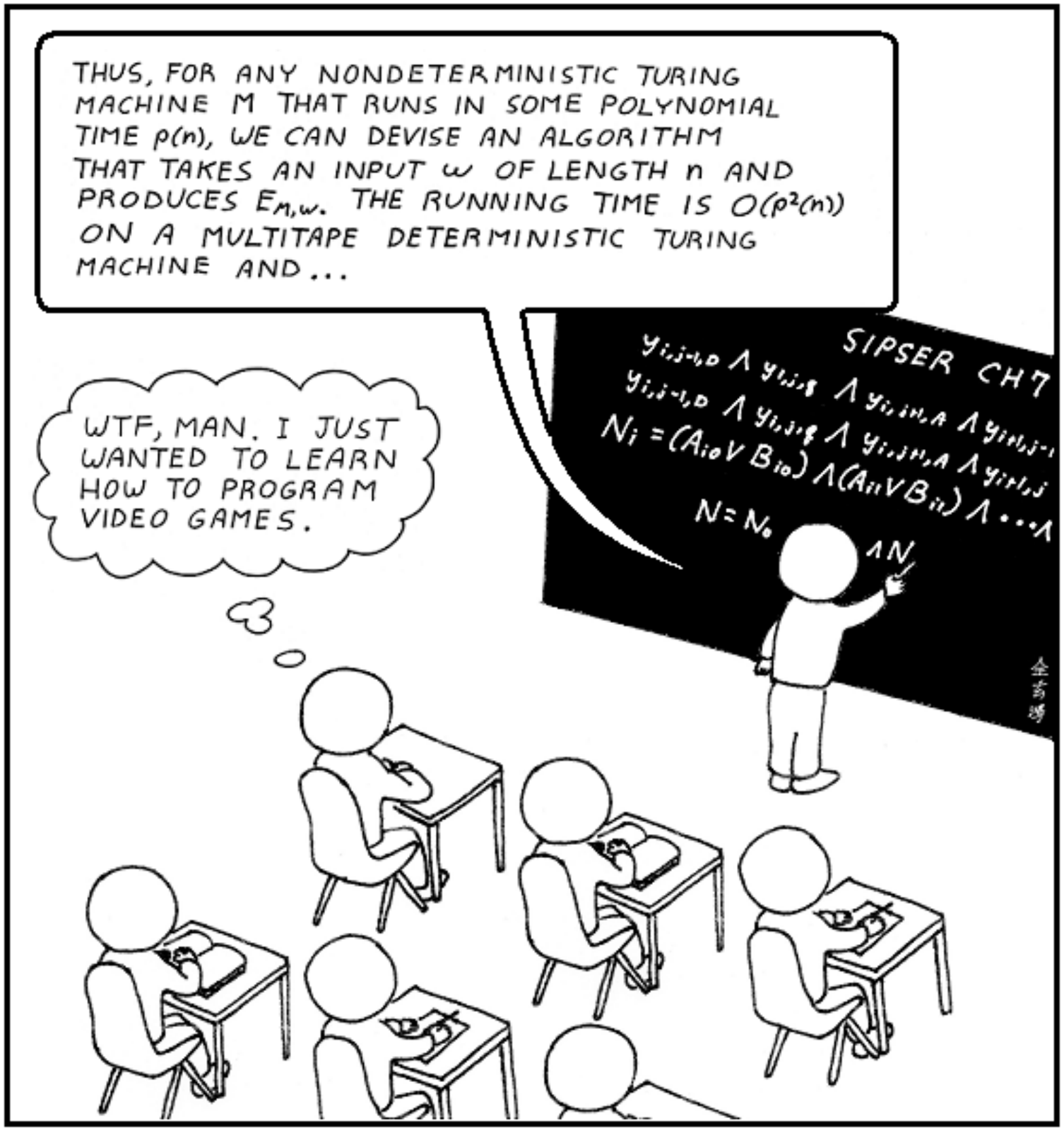
Vast majority of species unsequenced & *can not be cultivated in a lab* (motivation for metagenomics)

*Mora, Camilo, et al. "How many species are there on Earth and in the ocean?." PLoS biology 9.8 (2011): e1001127.

CSE 549:
Computational Biology

Computer Science for ~~Biologists~~ Biology

What is Computer Science?



http://people.cs.pitt.edu/~kirk/cs2110/computer_science_major.PNG

What is Computer Science?

Not actually simple to define constructively

Still debate whether certain areas constitute CS

Computer science is the scientific and practical approach to computation and its applications. It is the systematic study of the feasibility, structure, expression, and mechanization of the methodical procedures (or algorithms) that underlie the acquisition, representation, processing, storage, communication of, and access to information* …

What isn't Computer Science?

Don't install operating systems (may develop them)

Don't set up the office network (may study / design network protocols)

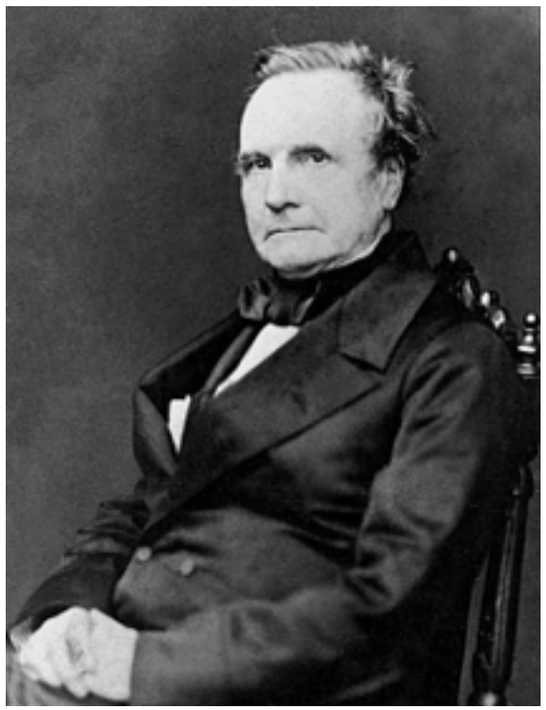
Not about Hacking together a program or learning a web-framework — programming ≠ CS (may study formal languages and develop new programming languages)

*http://www.cs.bu.edu/AboutCS/WhatIsCS.pdf

What is Computer Science?

Started as a branch of Mathematics — early computing machines

Charles Babbage (1791-1871)



Difference engine → Analytical engine*

Ada Lovelace (1791-1871)

Commonly considered the first “programmer”; developed an algorithm for the analytical engine to compute the Bernoulli numbers

*Analytical engine (would have been the first Turing-complete, general purpose computer) was never completed

en.wikipedia.org

What is Computer Science?

What is “computable”? Early 20th century & birth of “modern” CS

What is Computer Science?

What is “computable”? Early 20th century & birth of “modern” CS

Kurt Gödel

What is Computer Science?

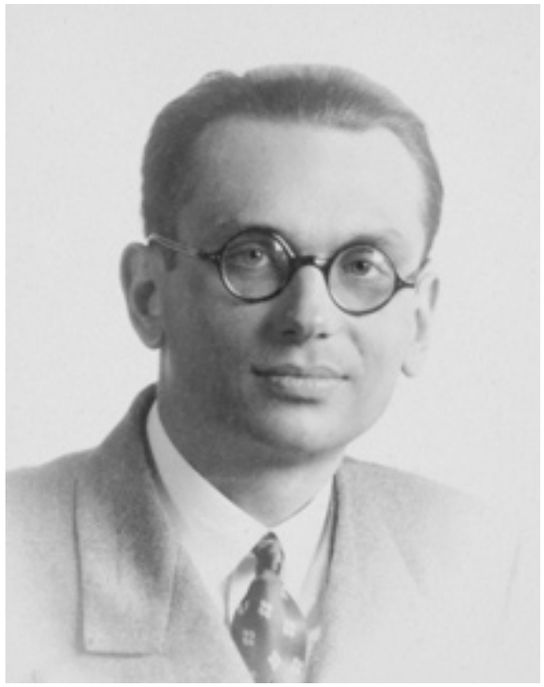
What is “computable”? Early 20th century & birth of “modern” CS

Kurt Gödel

Alan Turing

What is Computer Science?

What is “computable”? Early 20th century & birth of “modern” CS



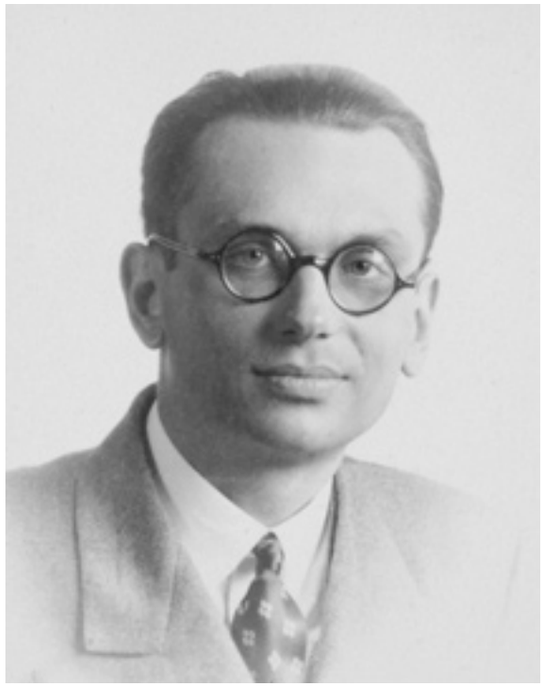
Kurt Gödel

Alan Turing

Alonzo Church

What is Computer Science?

What is “computable”? Early 20th century & birth of “modern” CS



Kurt
Gödel

Alan
Turing

Alonzo
Church

John
von Neumann

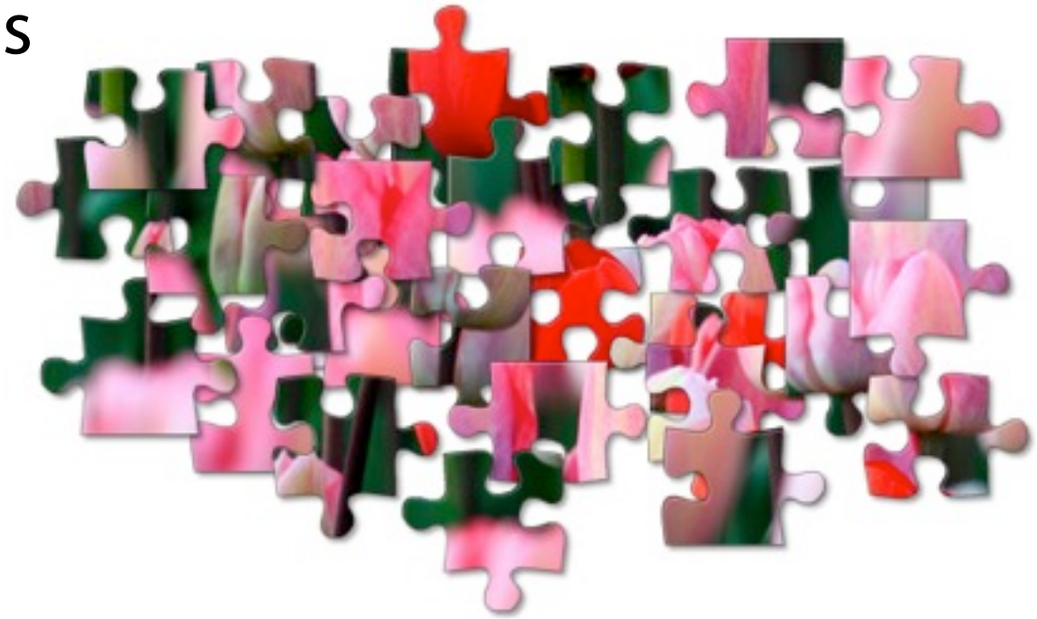
What is Computer Science?

Concerned with the development of provably **correct** and **efficient** computational procedures (algorithms & data structures) to answer **well-specified** problems.

To answer a computational question, we first need a well-formulated problem.

Assembly

Reads



+

Reference genome

How to assemble puzzle without the benefit of knowing what the finished product looks like?

Input DNA

Next 5 slides courtesy of Ben Langmead

Assembly

Whole-genome "shotgun" sequencing starts by copying and fragmenting the DNA

("Shotgun" refers to the random fragmentation of the whole genome; like it was fired from a shotgun)

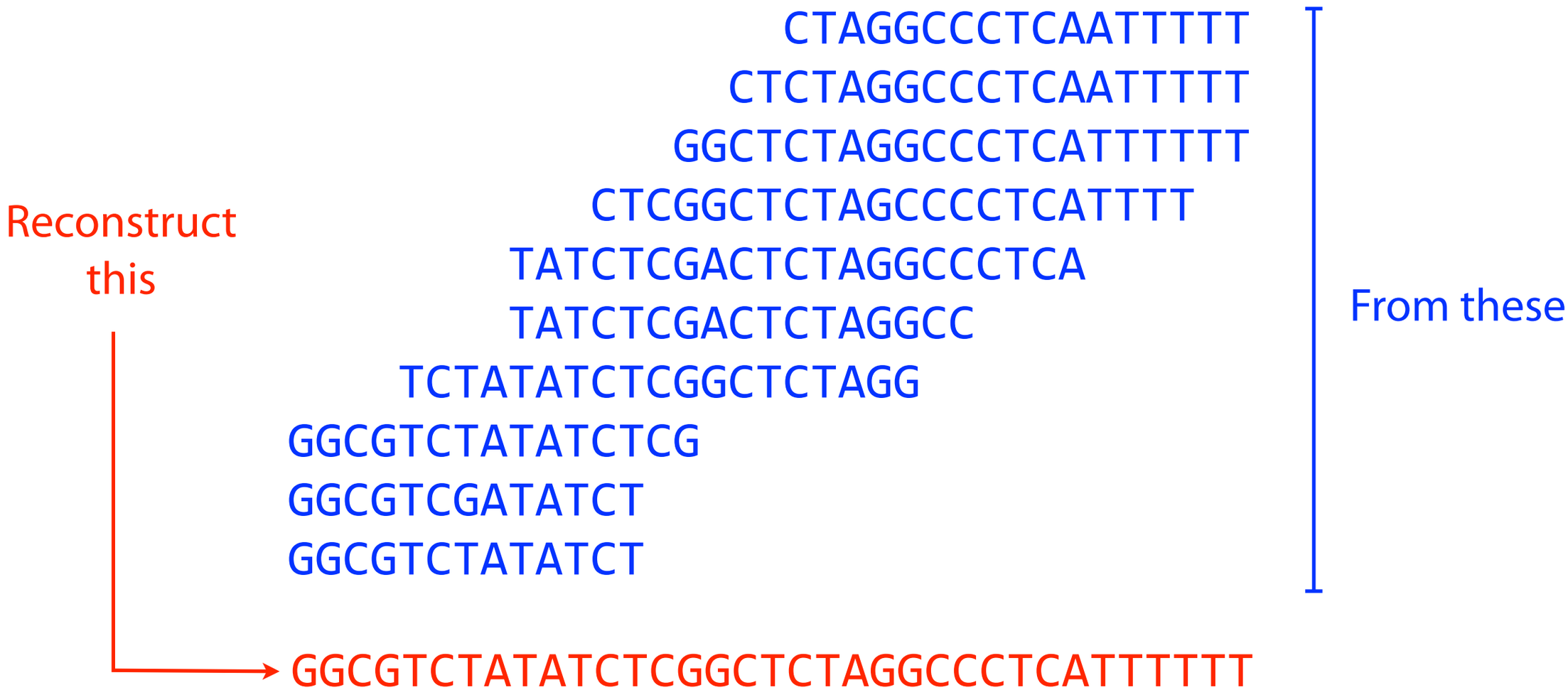
Input: GGCGTCTATATCTCGGCTCTAGGCCCTCATTTTTT

Copy: GGCGTCTATATCTCGGCTCTAGGCCCTCATTTTTT
GGCGTCTATATCTCGGCTCTAGGCCCTCATTTTTT
GGCGTCTATATCTCGGCTCTAGGCCCTCATTTTTT
GGCGTCTATATCTCGGCTCTAGGCCCTCATTTTTT

Fragment: GGCGTCTA TATCTCGG CTCTAGGCCCTC ATTTTTT
GGC GTCTATAT CTCGGCTCTAGGCCCTCA TTTTTT
GGCGTC TATATCT CGGCTCTAGGCCCT CATTTTTT
GGCGTCTAT ATCTCGGCTCTAG GCCCTCA TTTTTT

Assembly

Assume sequencing produces such a large # fragments that almost all genome positions are *covered* by many fragments...



Assembly

...but we don't know what came from where

CTAGGCCCTCAATTTTT
GGCGTCTATATCT
CTCTAGGCCCTCAATTTTT
TCTATATCTCGGCTCTAGG
GGCTCTAGGCCCTCATTTTTT
CTCGGCTCTAGCCCCTCATTTT
TATCTCGACTCTAGGCCCTCA
GGCGTCGATATCT
TATCTCGACTCTAGGCC
GGCGTCTATATCTCG

From these

Reconstruct this

GGCGTCTATATCTCGGCTCTAGGCCCTCATTTTTT

What is Computer Science?

Concerned with the development of provably **correct** and **efficient** computational procedures (algorithms & data structures) to answer **well-specified** problems.

To answer a computational question, we first need a well-formulated problem.

Given: a collection, R, of sequencing reads (strings)

Find: The genome (string), G, that generated them

What is Computer Science?

Concerned with the development of provably **correct** and **efficient** computational procedures (algorithms & data structures) to answer **well-specified** problems.

To answer a computational question, we first need a well-formulated problem.

Given: a collection, R, of sequencing reads (strings)

Find: The genome (string), G, that generated them

Not well-specified.
What makes one genome more likely than another?
What constraints do we place on the space of solutions?

What is Computer Science?

Concerned with the development of provably **correct** and **efficient** computational procedures (algorithms & data structures) to answer **well-specified** problems.

To answer a computational question, we first need a well-formulated problem.

Given: a collection, R, of sequencing reads (strings) ✔

Find: The shortest genome (string), G, that contains all of them

Shortest Common Superstring

Given: a collection, $S = \{s_1, s_2, \ldots, s_k\}$, of sequencing reads (strings)

Find*: The shortest possible genome (string), G, such that $s_1, s_2, \ldots, s_k$ are all substrings of G

How, might we go about solving this problem?

*for reasons we'll explore later, this isn't actually a great formulation for genome assembly.

Shortest common superstring

Given a collection of strings *S*, find *SCS*(*S*): the shortest string that contains all strings in *S* as substrings

Without requirement of "shortest," it's easy: just concatenate them

Example: *S*: BAA AAB BBA ABA ABB BBB AAA BAB

Concatenation: BAAAABBBAABAABBBBBBAAABAB

24

SCS(S): AAABBBABAA

10

AAA
AAB
ABB
BBB
BBA
BAB
ABA
BAA

Slide courtesy of Ben Langmead

Shortest common superstring

Can we solve it?

Imagine a modified overlap graph where each edge has cost = - (length of overlap)

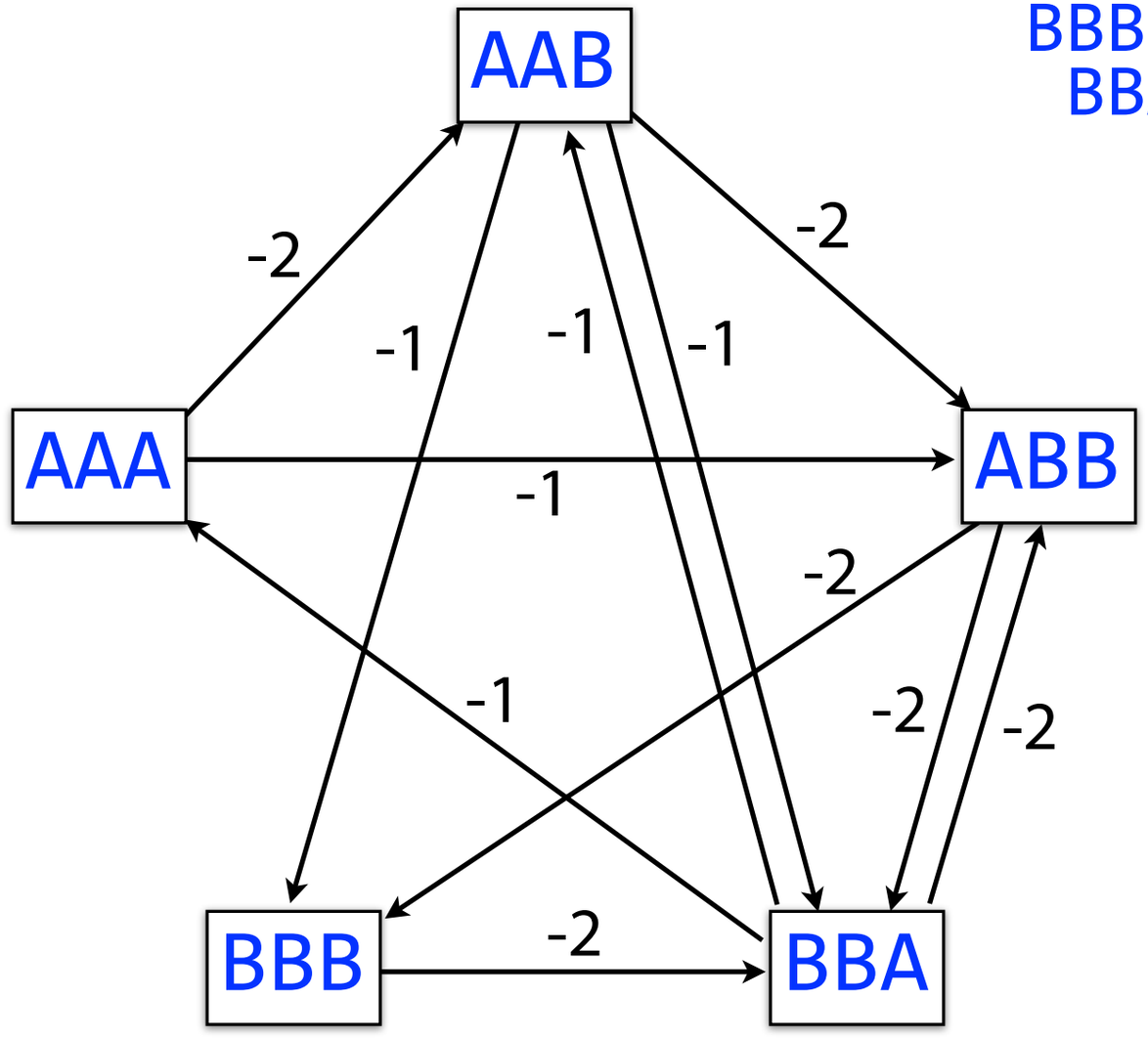
SCS corresponds to a path that visits every node once, minimizing total cost along path

That's the *Traveling Salesman Problem* (*TSP*), which is NP-hard!

S: AAA AAB ABB BBB BBA

SCS(S): AAABBBA

AAA
AAB
ABB
BBB
BBA



Slide courtesy of Ben Langmead

Shortest common superstring

Say we disregard edge weights and just look for a path that visits all the nodes exactly once

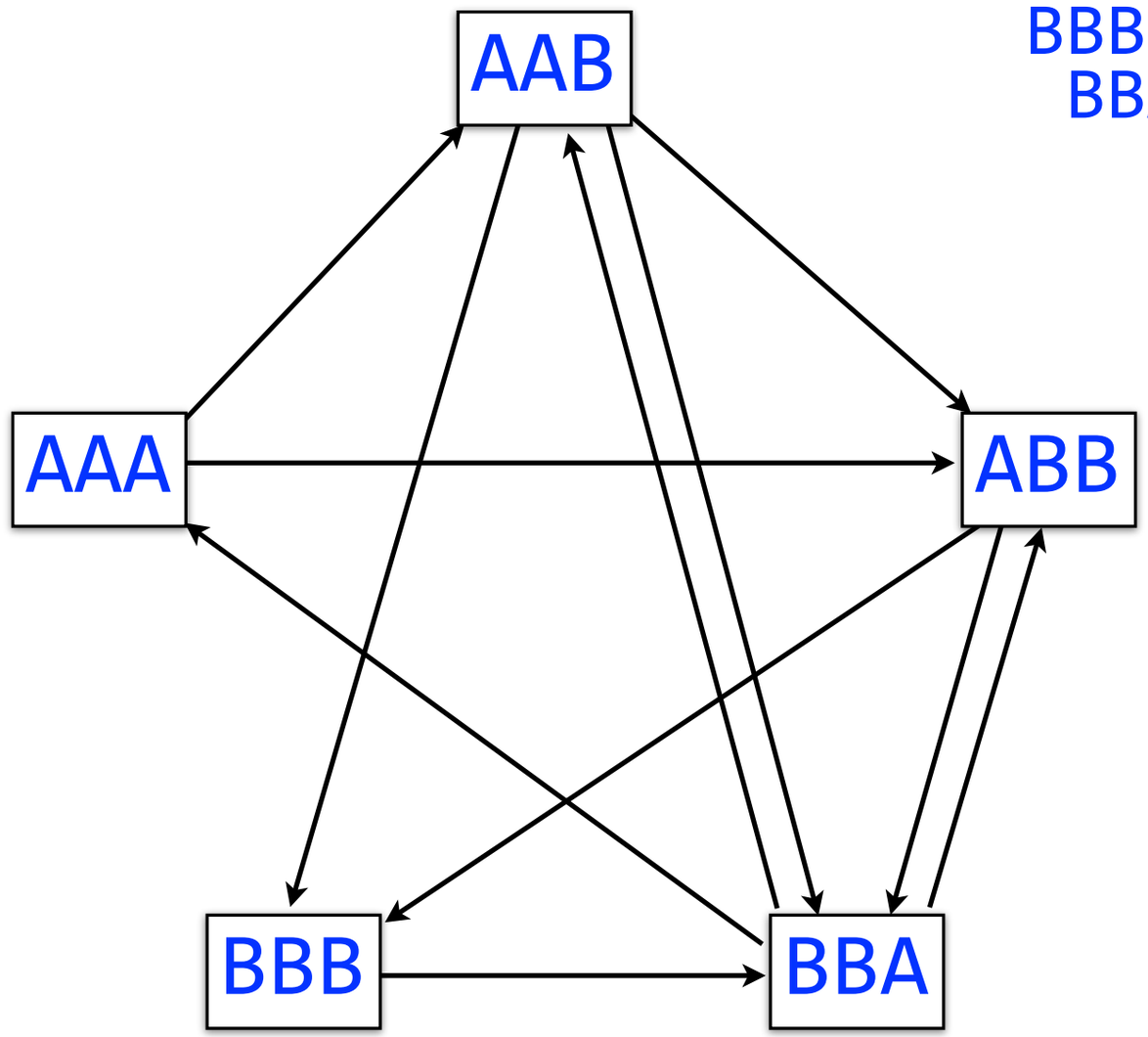
That's the *Hamiltonian Path* problem: NP-complete

Indeed, it's well established that SCS is NP-hard

S: AAA AAB ABB BBB BBA

SCS(S): AAABBBA

AAA
AAB
ABB
BBB
BBA



Slide courtesy of Ben Langmead

Shortest common superstring & friends

Traveling Salesman, Hamiltonian Path, and Shortest Common Superstring are all NP-hard

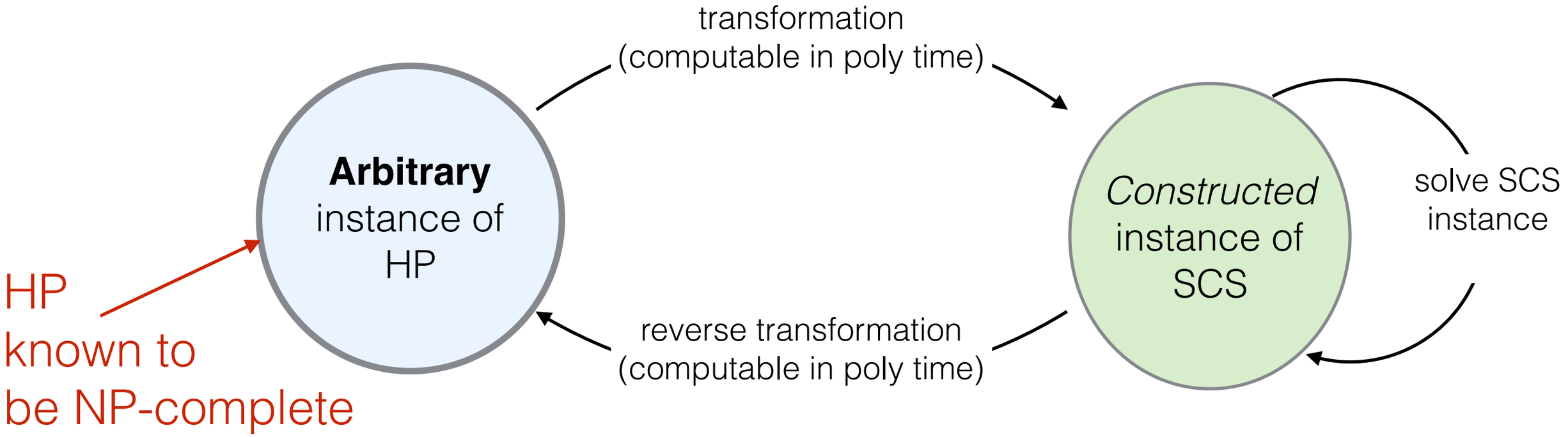
For refreshers on Traveling Salesman, Hamiltonian Path, NP-hardness and NP-completeness, see Chapters 34 and 35 of "Introduction to Algorithms" by Cormen, Leiserson, Rivest and Stein, or Chapters 8 and 9 of "Algorithms" by Dasgupta, Papadimitriou and Vazirani (free online: http://www.cs.berkeley.edu/~vazirani/algorithms)

Slide courtesy of Ben Langmead

Important note: The fact that we modeled SCS as NP-hard problems (TSP and HP) **does not** prove that (the decision version of) SCS is NP-complete. To do that, we must **reduce** a known NP-complete problem to **SCS**.

Given an instance I of a known hard problem, generate an instance I' of SCS such that if we can solve I' in polynomial time, then we can solve I in polynomial time. This *implies* that SCS is *at least* as hard as the hard problem.

This can be done e.g. with HAMILTONIAN PATH



Shortest Common Superstring

The fact that SCS is **NP-complete** means that it is unlikely that there exists *any* algorithm that can solve a general instance of this problem in time polynomial in n — the number of strings.

If we give up on finding the *shortest* possible superstring G, how does the situation change?

Shortest Common Superstring

There's a "greedy" *heuristic* that turns out to be an *approximation algorithm* (provides a solution within a constant factor of the the optimum)

At each step, chose the pair of strings with the maximum overlap, merge them, and return the merged string to the collection.

Greedy conjecture factor of 2-OPT *is* the worst case — proof for factor 3.5

Different approx. (not all greedy)

ratio	authors	year
	approximating SCS	
3	Blum, Jiang, Li, Tromp and Yannakakis [4]	1991
$2\frac{8}{9}$	Teng, Yao [23]	1993
$2\frac{5}{6}$	Czumaj, Gasieniec, Piotrow, Rytter [8]	1994
$2\frac{50}{63}$	Kosaraju, Park, Stein [15]	1994
$2\frac{3}{4}$	Armen, Stein [1]	1994
$2\frac{50}{69}$	Armen, Stein [2]	1995
$2\frac{2}{3}$	Armen, Stein [3]	1996
$2\frac{25}{42}$	Breslauer, Jiang, Jiang [5]	1997
$2\frac{1}{2}$	Sweedyk [21]	1999
$2\frac{1}{2}$	Kaplan, Lewenstein, Shafrir, Sviridenko [12]	2005
$2\frac{1}{2}$	Paluch, Elbassioni, van Zuylen [18]	2012
$2\frac{11}{23}$	Mucha [16]	2013

Golovnev, Kulikov, & Mihajlin. "Approximating Shortest Superstring Problem Using de Bruijn Graphs." Combinatorial Pattern Matching. Springer Berlin Heidelberg, 2013.