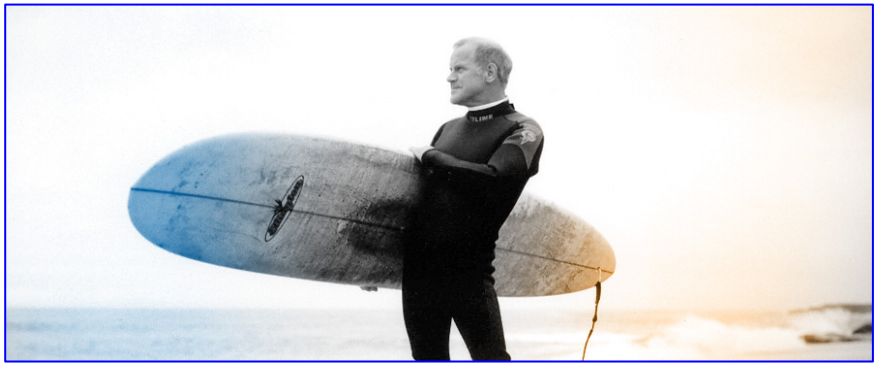
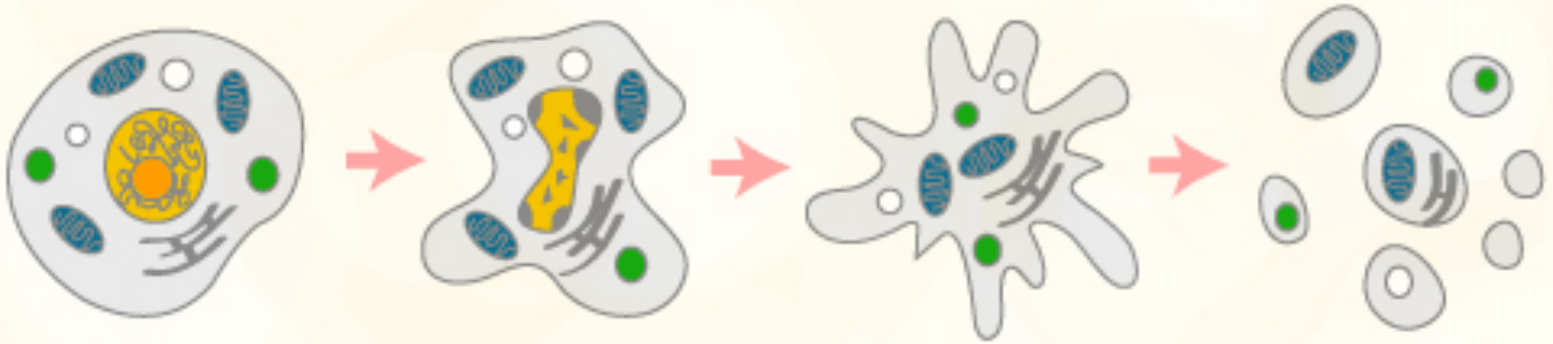


BIOL/CS 123B
Midterm 1 Review

Spring 2021

Philip Heller

Apoptosis



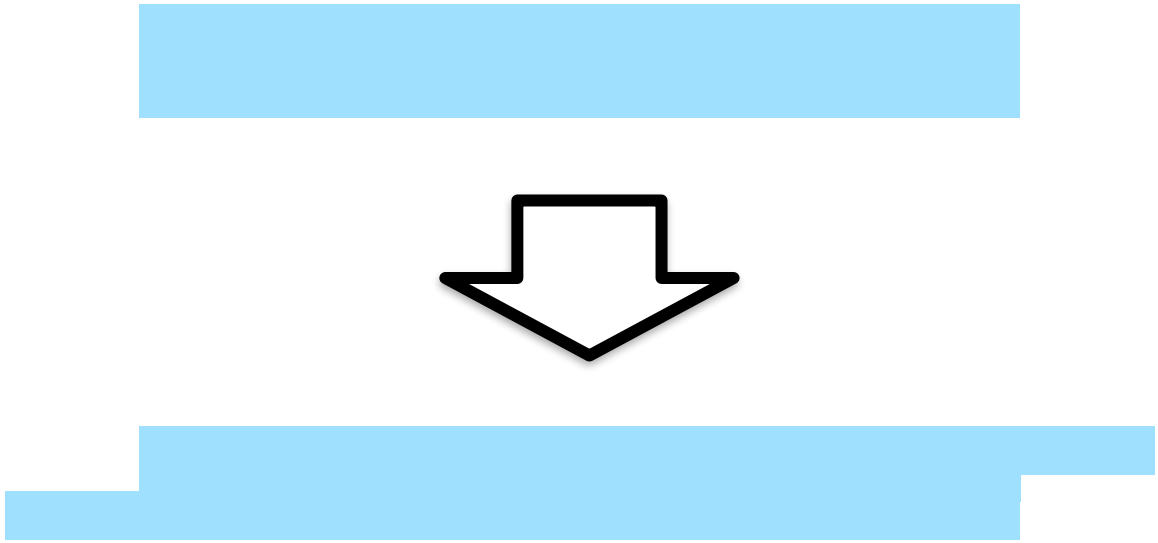
Cloning: 1972

- Technology
  - Restriction enzymes: cut DNA molecule at a specific sequence
  - DNA ligases: glue together 2 DNA fragments

- Bacteria contain *plasmids*
  - Small (~5000 bp) circular DNA fragments
  - Reproduced during cell reproduction
  - Relatively easy to sequence, even with 1960s/1970s technology
  - Some sequences 100% known

Cloning: Step 1

Add sticky ends to original gene



Cloning: Step 2

Cut a plasmid with restriction enzymes

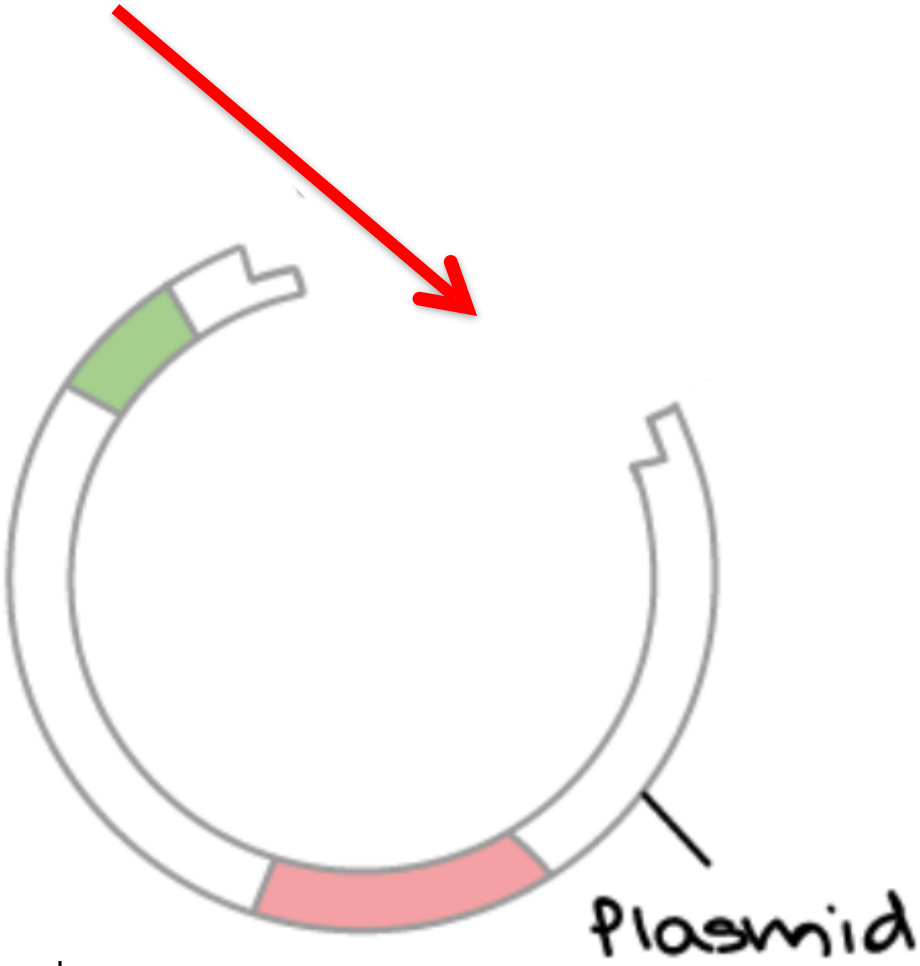
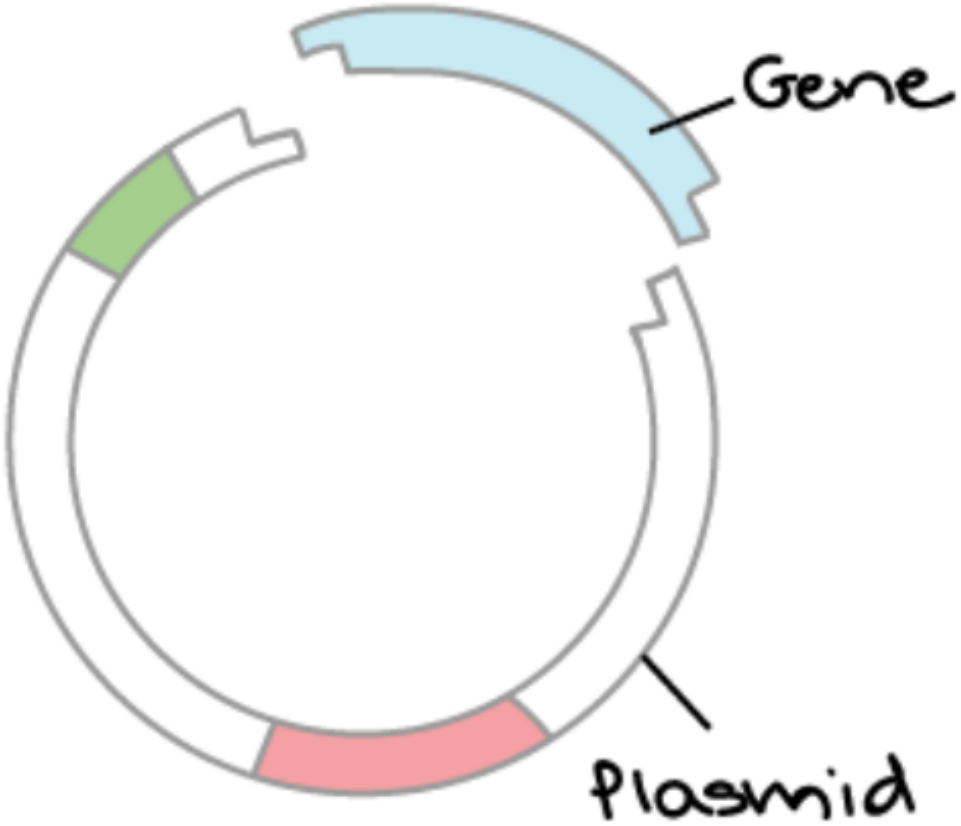


Image source: Khanacademy.org

Cloning: Step 3

Add gene with appropriate sticky ends
Add ligase

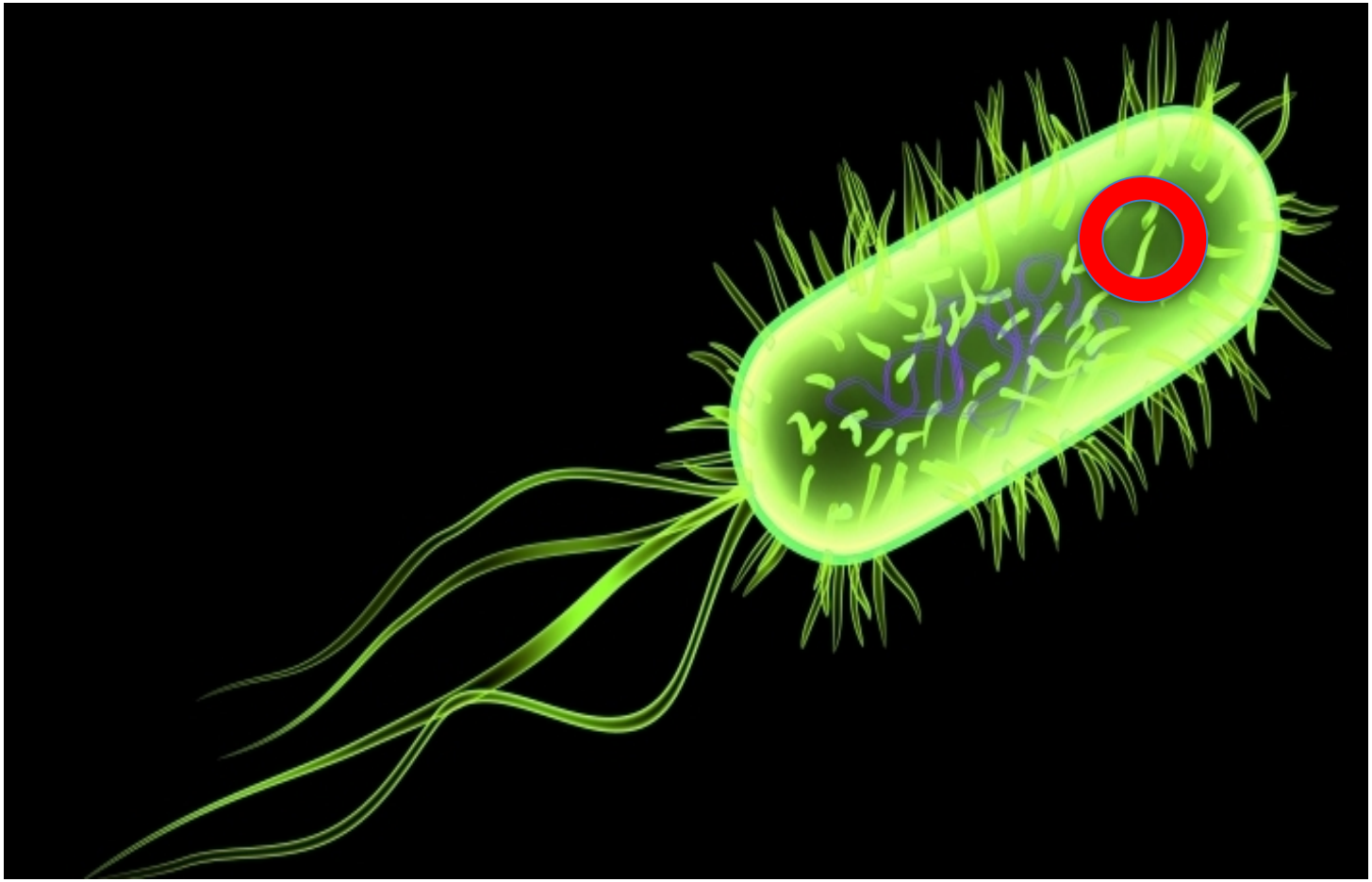


Cloning: Step 4

Plasmid and gene combine → recombinant DNA

Cloning: Step 5

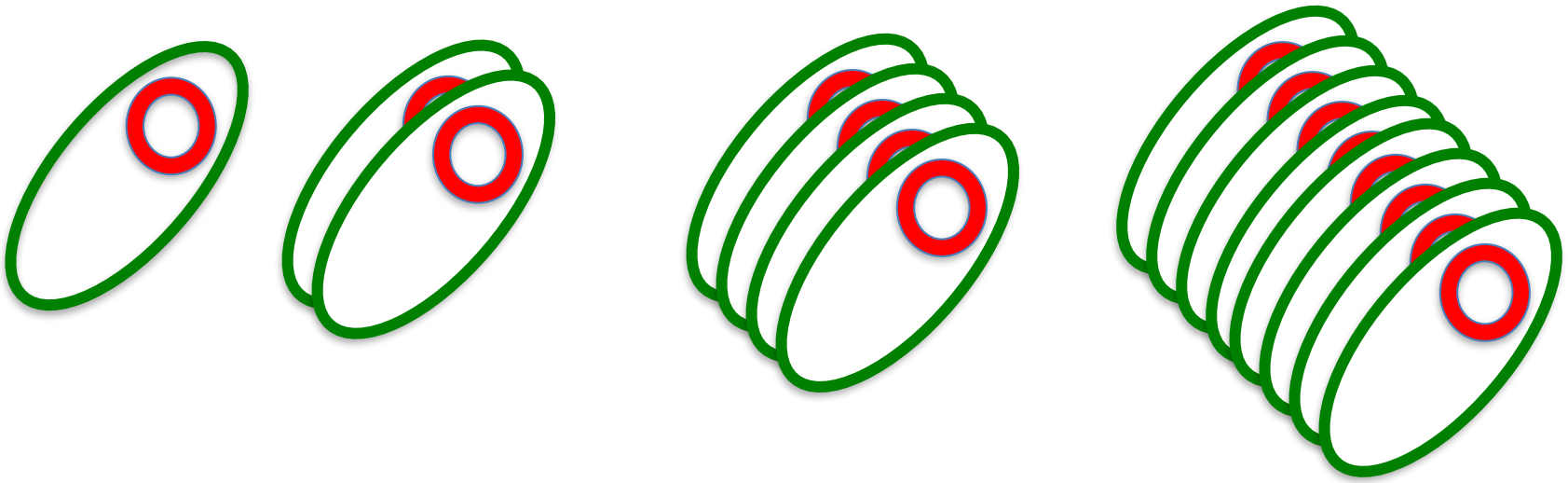
Insert engineered plasmid into a bacterium



Cloning: Step 6

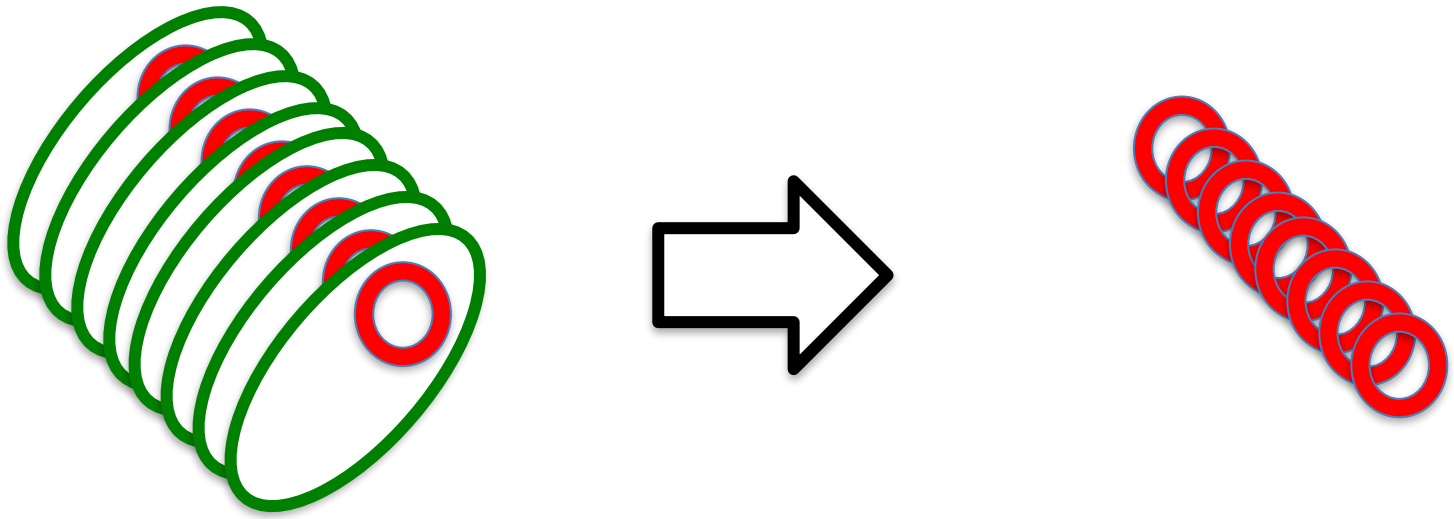
Bacterium reproduces, doubling population each generation

1 → 2 → 4 → 8 → 16 → 32 …



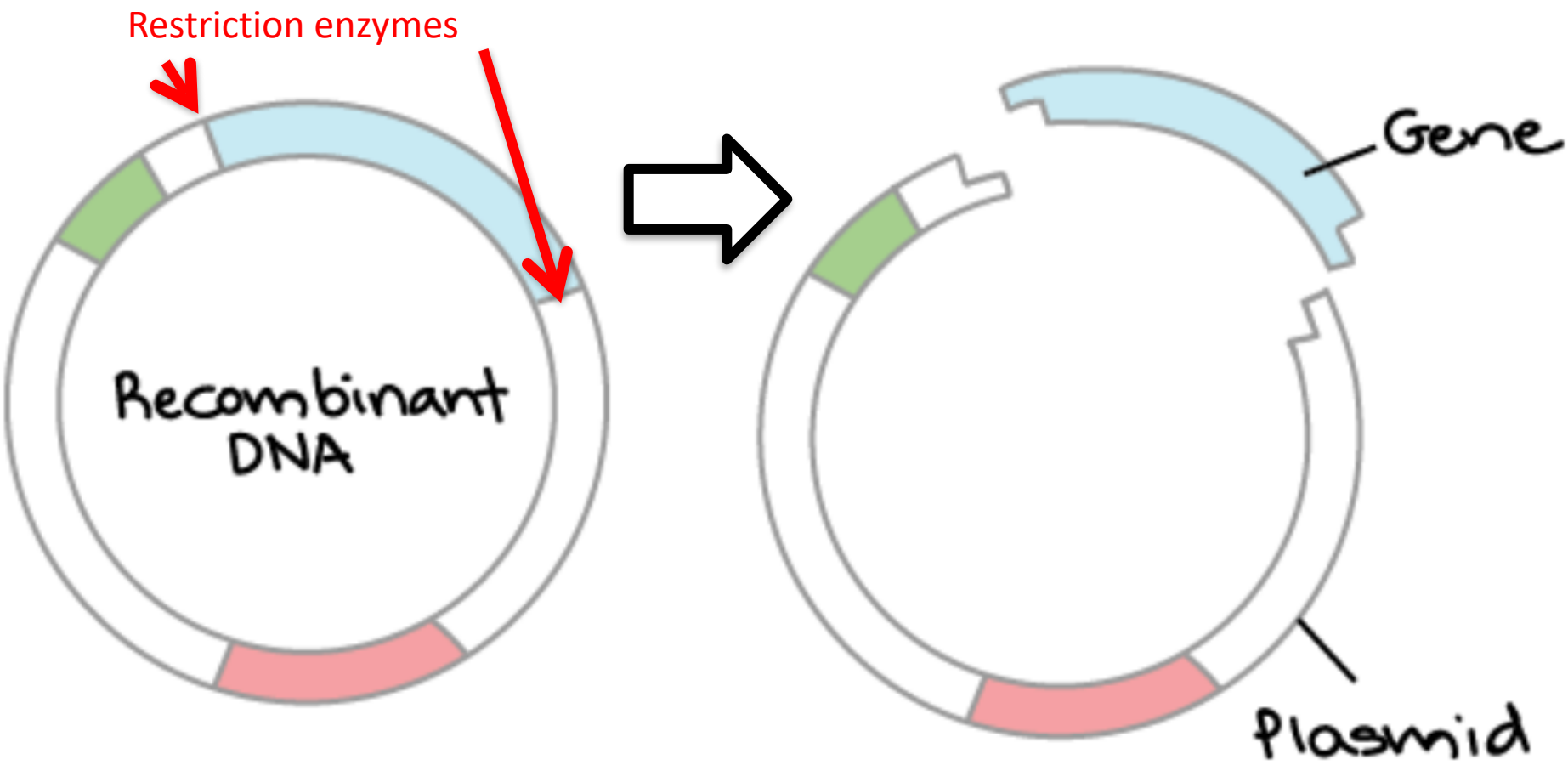
Cloning: Step 7

Extract all the replicated plasmids, discard everything else



Cloning: Step 8

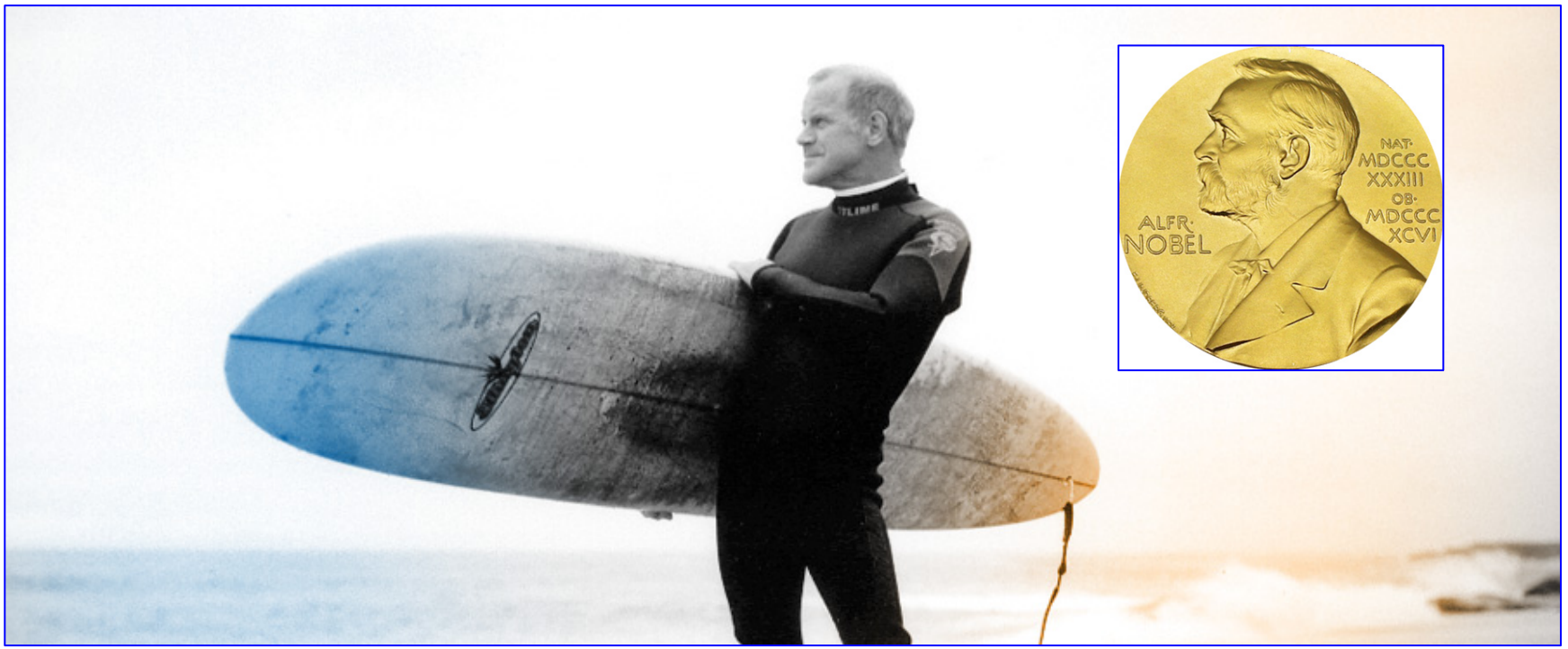
Extract target gene from each replicated plasmid, discard everything else



Drawbacks of cloning as an amplification technology

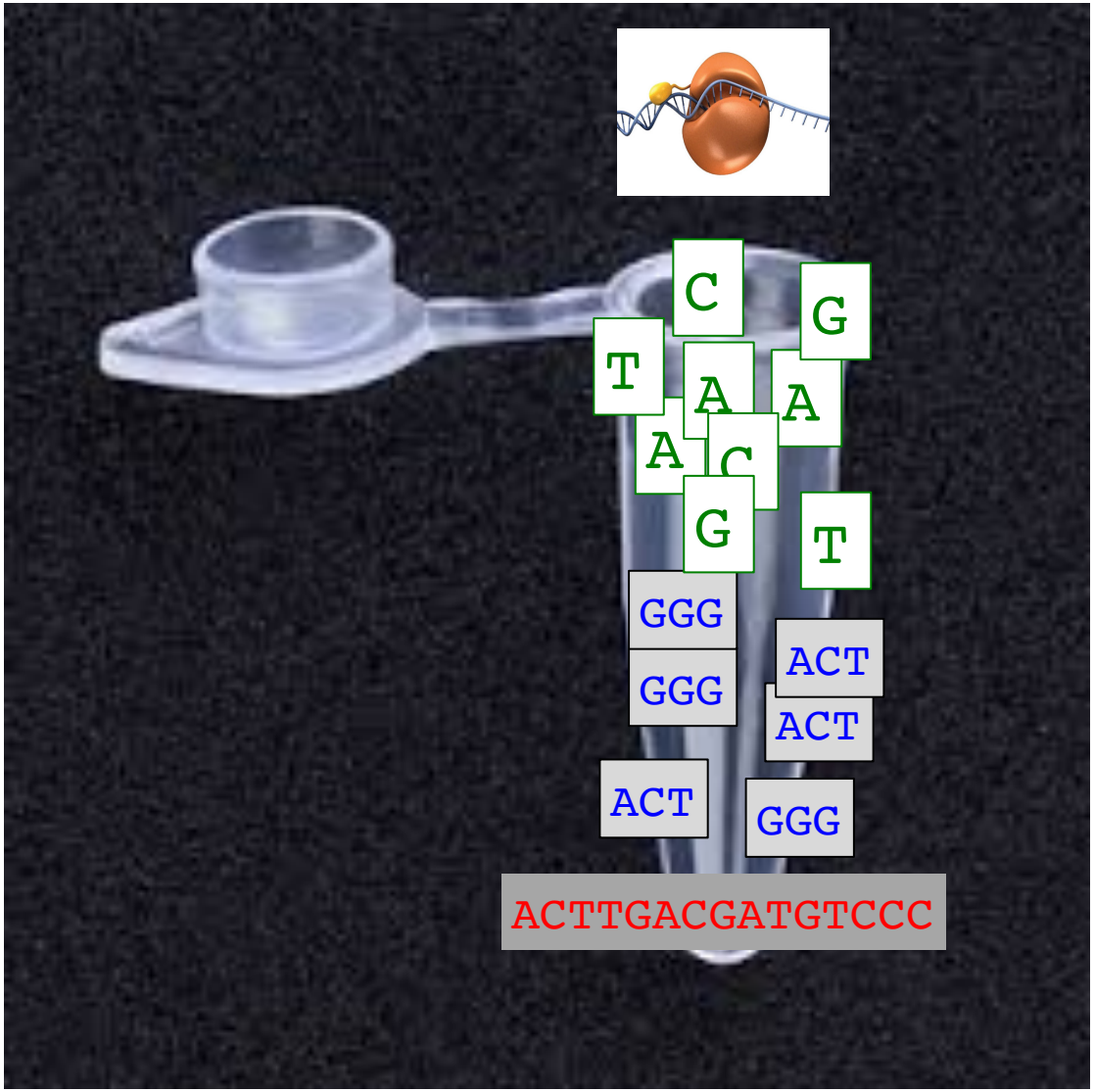
- To replicate a gene, you need plasmids and bacteria.
- The doubling time is limited by the generation time of the bacteria (best case = ~1/2 hour).
- → If only the target gene could be made to replicate directly!
  - No bacteria.
  - No plasmids.

1983: Kary Mullis develops Polymerase Chain Reaction (PCR)



No more plasmids, no more bacteria

The PCR Tube: where it all happens



Add DNA template

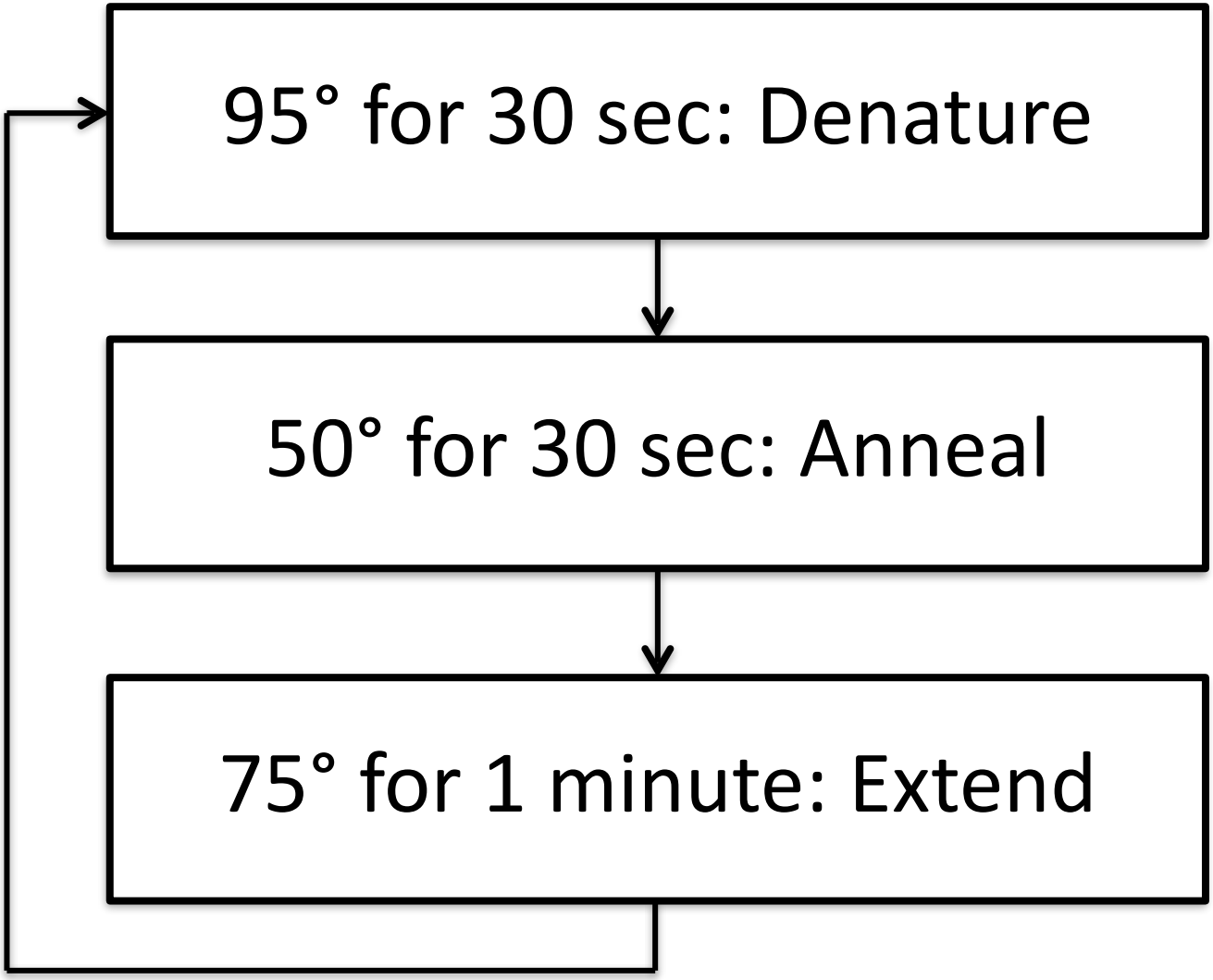
Add many copies of primers

Add lots of As, Cs, Gs & Ts

Add DNA polymerase

Now just cycle the temperature

Temps & times are rough approximates



Sanger Chain-Termination Sequencing

- Developed ~1977 by Fred Sanger
  - 1918 – 2013
  - Won Nobel Prize twice
  - Only 3 others have ever done that: Curie, Pauling, Bardeen
- ddNTP

4 kinds of ddNTP

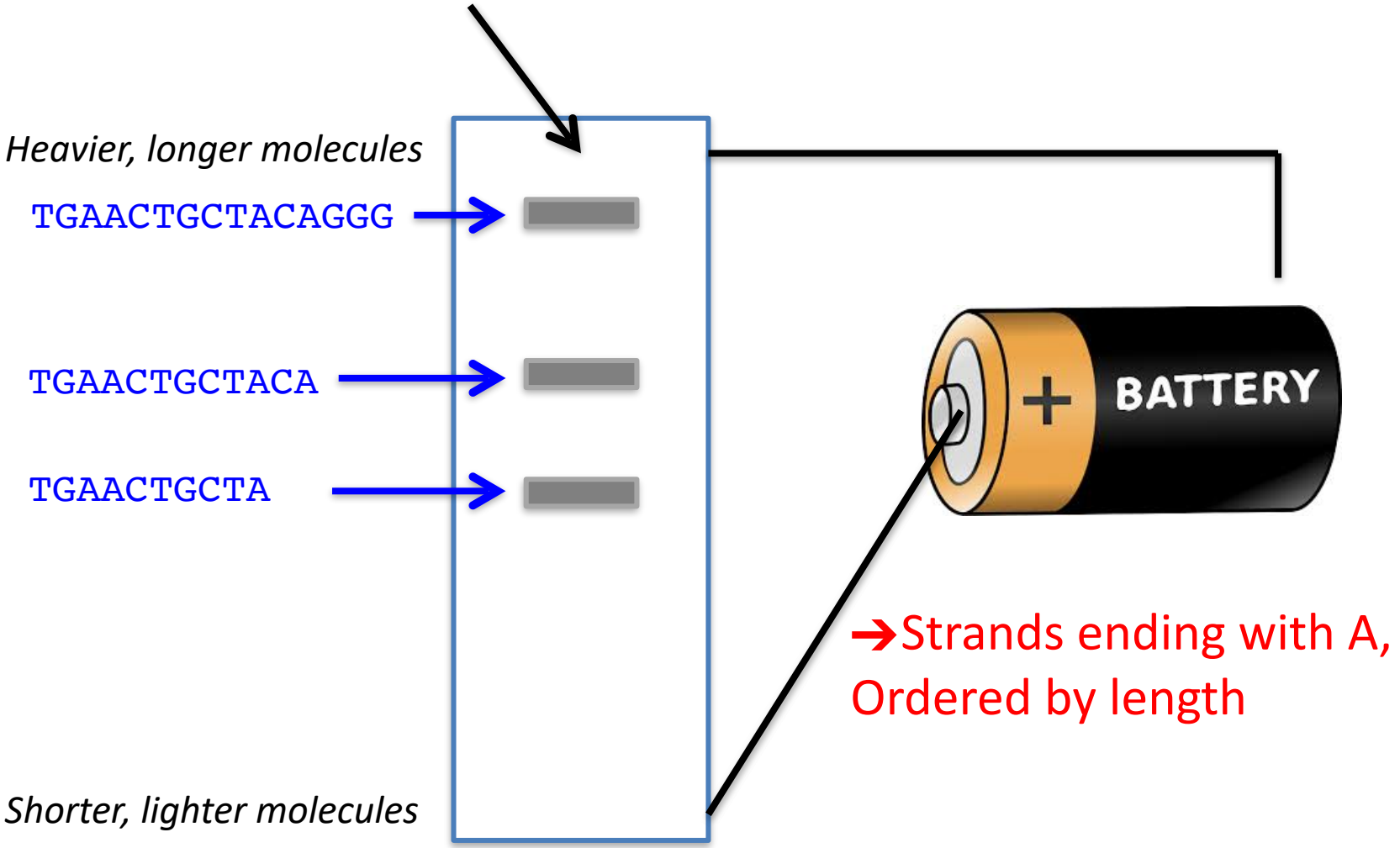
- Chain-terminating Adenine = ddATP
- Chain-terminating Cytosine = ddCTP
- Chain-terminating Guanine = ddGTP
- Chain-terminating Thymine = ddTTP

- ➔ “N” in “ddNTP” is a wildcard char
  - CS majors: think of it as “dd*TP”
- "TP" in "ddNTP" = TriPhosphate
  - We won't go into that chemistry

Sanger Sequencing

- Denature (separate) target DNA strands (like PCR)
- Primer will anneal to 3’ end of template strand (like PCR)
- Nucleotides in solution will extend from the primer (like PCR) until …
- Until the growing strand incorporates a ddATP, which terminates extension

TGAACTGCTA +
TGAACTGCTACA +
TGAACTGCTACAGGG



→Strands ending with A,
Ordered by length

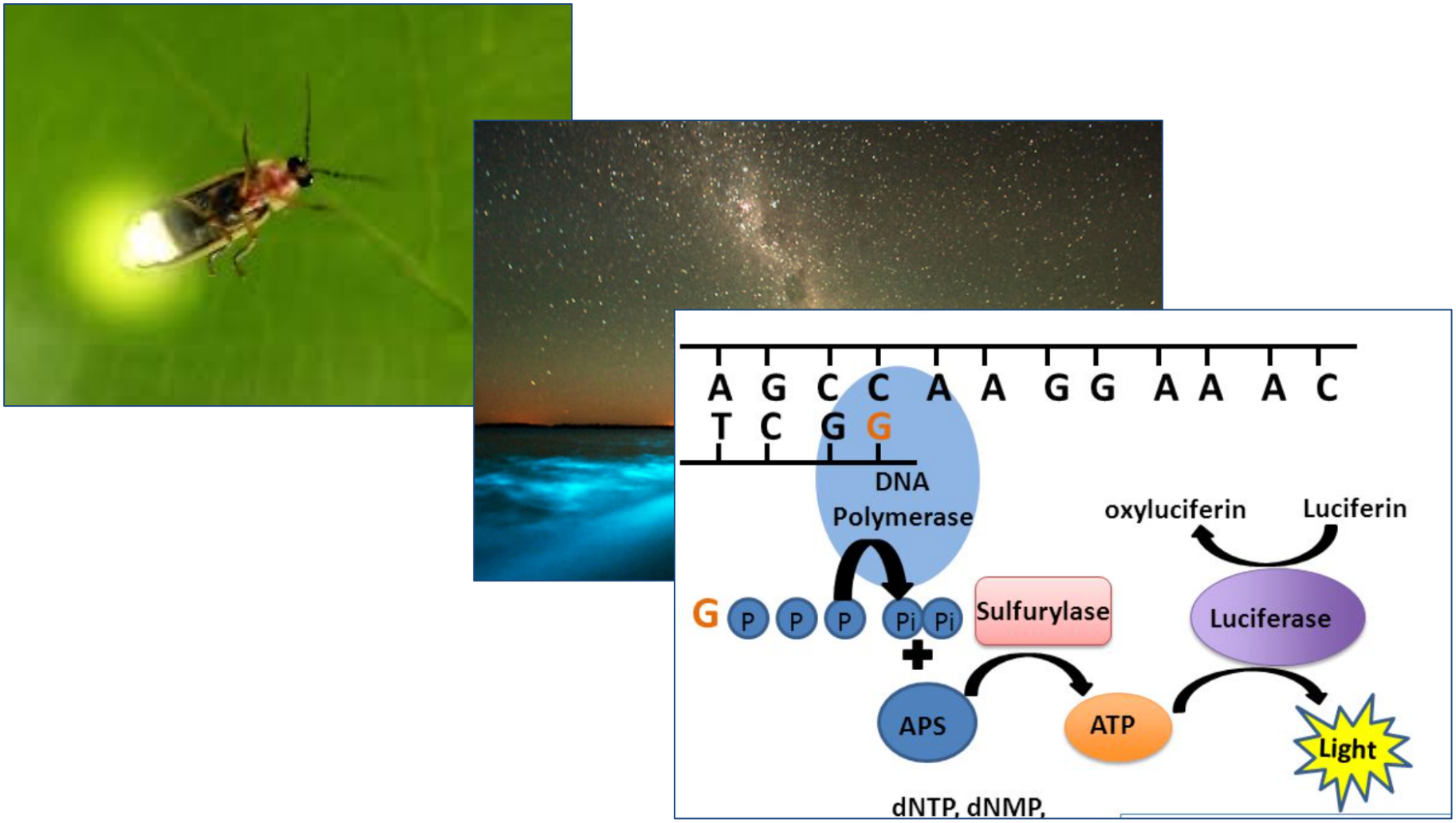
Template strand ACTTGACGATGTCCC

Coding strand TGAACTGCTACAGGG

Primer = 1st n bases of coding strand, so gel readout is remainder of coding strand

Don’t complement, reverse, or reverse complement

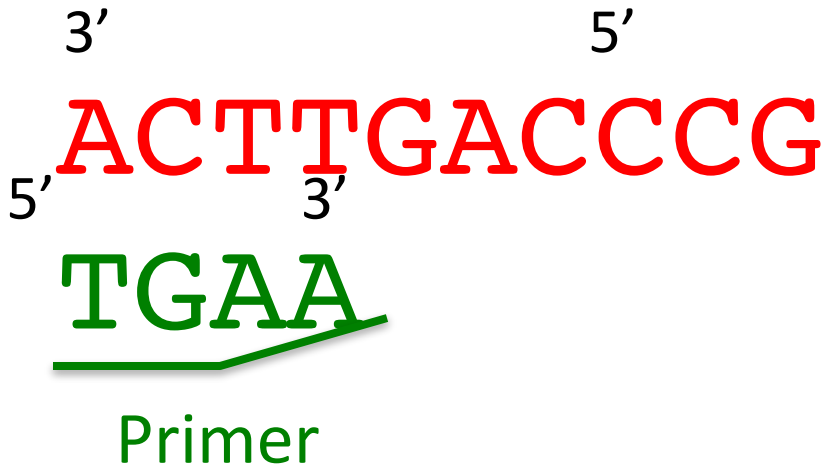
1990: Pyrosequencing



dnATP, dnCTP, dnGTP, dnTTP

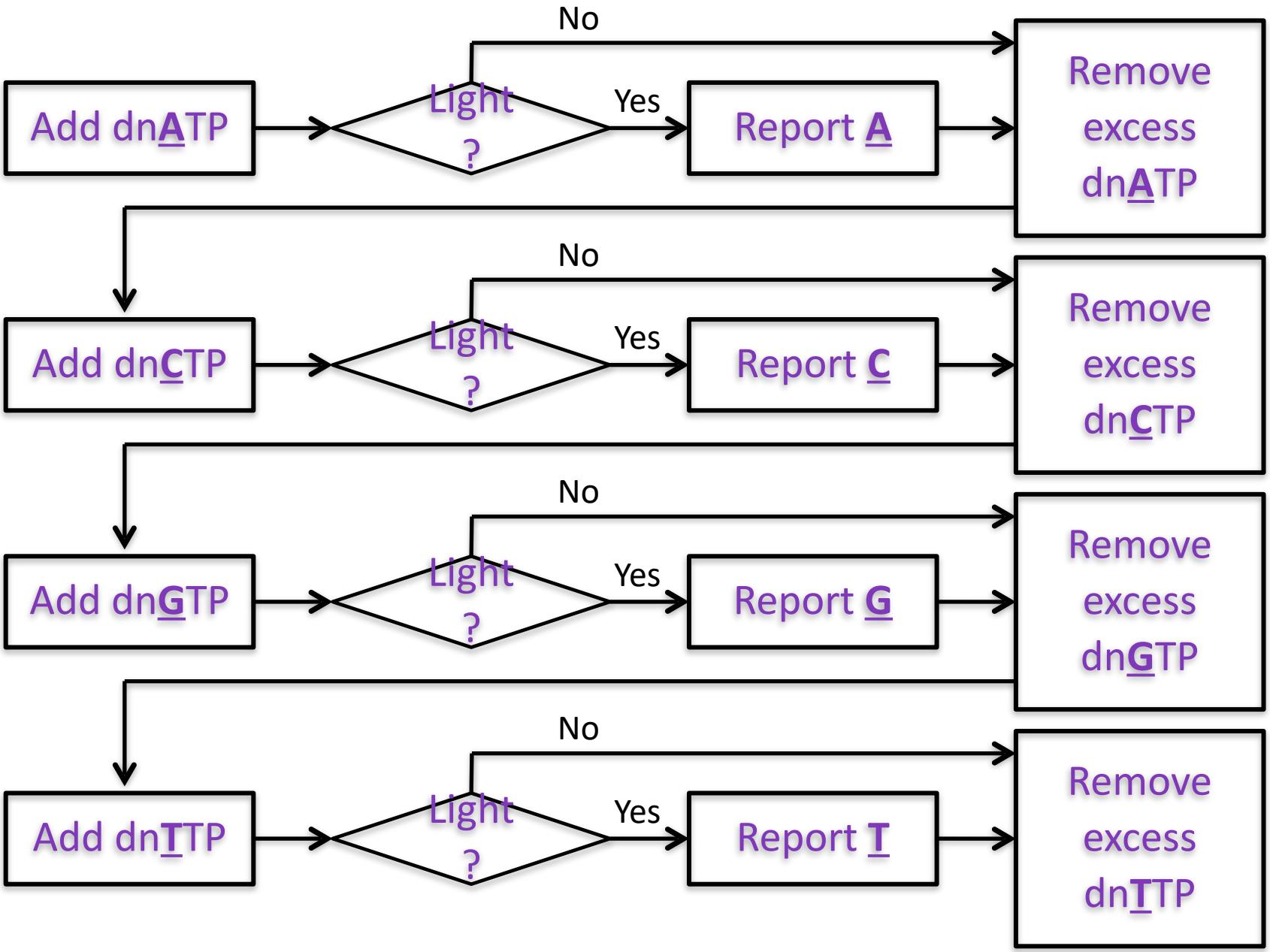
- Not chain-terminating.
- "TP" means tri<u>phosphate</u>.
- <u>Di</u>phosphate molecules are stable.
- Adding a 3rd phosphate to a diphosphate requires energy (like loading a nerf gun).
- Triphosphate molecules are less stable
  - Under the right conditions, release 1 phosphate and energy (like pulling the trigger of the nerf gun.

Pyrosequencing: Before 1st cycle



+ luciferin,
luciferase,
and
other secret sauce

Pyrosequencing Cycle



Pyrosequencing is error prone

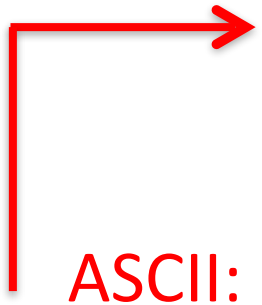
- Flashes are brief
- Photodetectors are imperfect
- Hard to distinguish XXXXX from XXXXXX
- It's cheap and it's fast, but its quality is lower than Sanger sequencing
- → Use PCR to quickly/cheaply make lots of copies, which are quickly/cheaply sequenced
- → Trust the majority

Sequencing devices estimate the quality of every base they report

- Users can ignore data whose quality is too low
- “Too low” varies with what you’re doing
- Quality is a single character
- Sequencers output files in “fastq” format
  - 4 fields per read
    - Unique identifier (“defline”), starts with @
    - Nucleotide sequence
    - +
    - Quality sequence, same length as nucleotide sequence

Fastq record: toy example

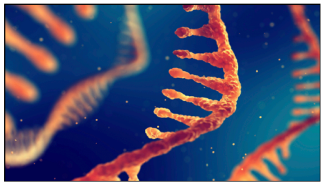
```
@This is read #1
ACGTACGTTGACTAGC
+
7887MN#+;;,87837
```



ASCII: Hard for humans to interpret

Review HW2
Answers are on Canvas

CS/BIOL 123B S21
Homework 2: 20 points
Due Friday Feb 19 11:59 PM
Upload a Word doc to Canvas



Caravaggio “The Cardsharps” ~1594

Probability: What is the chance of some "event"

- P(an odd number in roulette) = ?

- P(roll total 7 or 11 with 2 dice) = ?

- P(get a royal flush) = ?

The 2 kinds of roulette question

"Casablanca", 1942

- While you're playing: "What is the probability that the next roll will be red? (Assuming a fair wheel)"
  - P(red | fair wheel) = ?

- Walking home because you can't afford Uber fare because you lost all your money because you bet on red 500 times and *never* won: "What is the probability that the wheel is unfair?"
  - P(unfair wheel | 500 not-reds) = ?

2 very different issues:

- P(an outcome | hypothesis)
  - We can rely on the hypothesis

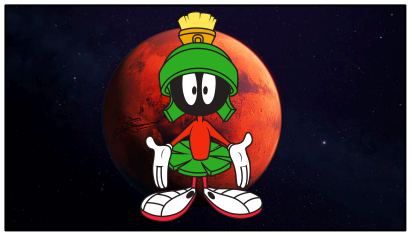
- P(hypothesis | an outcome)
  - The hypothesis is a *hypothesis* about the unknowable.
  - We call P "the likelihood" rather than "the probability".
  - We accept a hypothesis if its likelihood is high compared to likelihoods of all alternative hypotheses.
  - High = ???? Depends on the situation and your inclination.

- As scientists, we're in the hypothesis business.

What does P(hypothesis) mean? Or P(hypothesis|observations)?

- Example:
  - Hypothesis: There is life on Mars
  - P(There is life on Mars) = .25
- What it doesn't mean: If we observe 28 Marses, we expect there will be life on ~7 of them.

What does P(life on Mars) = .25 mean?



- A number but not a probability.
- A measure of my confidence, the strength of my belief.
- What can influence strength of belief?
  - Evidence ← The only scientifically valid influence
  - Trust in the source
  - Hope
  - Culture
  - Fear of the alternative
- We're getting into philosophy here.
- P(hypothesis|observation) is a different kind of thing from P(observation|hypothesis) → We use a different word for it: "Likelihood".

E-values: like p-values for blast

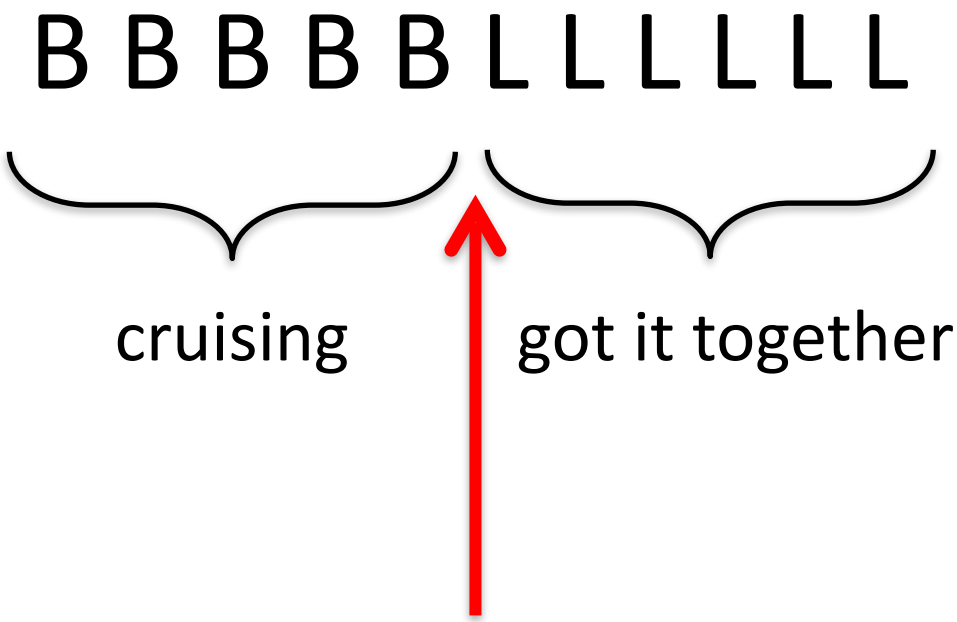
- The strict definition of the E-value of a blast hit:
  - with a query sequence of length L
  - against some database DB
  - that hits a subject with % identity x
- E-value = probability that
  - blasting a random query sequence of length L
  - against the same database DB
  - will hit a subject with % identity >= x
- An imperfect interpretation (but not bad): E-value is the probability that the similarity between the query and the subject is due to coincidence, rather than evolutionary relationship.

Competing Statistical Models

- E.g. Markov chains and (soon) HMMs
- Attempts to explain hidden aspects of reality
- Competing models can be evaluated based on relative probability of observation
- Example: *The #1 Students' Detective Agency*
  - Are you worried about how your son/daughter is doing in college?
  - Pay us, we can help!

Client #1 according to alternative models

- P(LLLLLLLLLLL | successful) ~= 9E-9
  - Is this a lot or a little?
  - ***Individual probabilities have no intuitive meaning***
  - They are meaningful when compared to other probabilities
- P(LLLLLLLLLLL | distressed) ~= 3E-12
- P(…|successful) is 3300x greater than P(…| distressed)
- A successful student is 3300x more likely than a distressed student to be at the library 12 consecutive times
- A student who spends 12 consecutive turns in the library is 3300x more likely to be successful than to be distressed



- This student might not get an A
- But as their professor, I'm no longer worried
- The Markov Chain can't represent attitude change

Hidden Markov Models

Jack Kirby

- Many similarities to Markov Chains:
  - States
  - Initial probabilities
  - Transition probabilities
- The big difference
  - States are separate from observations
  - With the student Markov Chains, states were Library/Café/Bar, because we can observe students in those places
  - With HMMs, states generate observations according to random ("stochastic") rules
- Example: A weather god
  - Thor has moods (states)
  - Weather is created according to Thor's mood/state
  - E.g. Storms are more likely if Thor is angry

Viterbi Algorithm

- Given a pattern of emissions, compute the most likely responsible state path
- Dynamic Programming
  - It will remind you of pairwise alignment
- O(# of states^2 * pattern length)
  - Whew!
  - The more obvious “brute-force” approach is to generate all possible state paths (slow but easy), compute probability of each, and choose the most probable
  - That’s O(# of states ^ pattern length)

Review of Big-Oh

- ***Not*** the official CS146 definition
- A way to compare execution time of algorithms
  - Not particular programs which implement algorithms
  - Independent of implementation, independent of computer
- "An algorithm is O(n^2)" roughly means that execution time is proportional to n^2
  - n = input size
  - time α n^2 ➔ time = k * n^2
  - k varies across implementations and computers

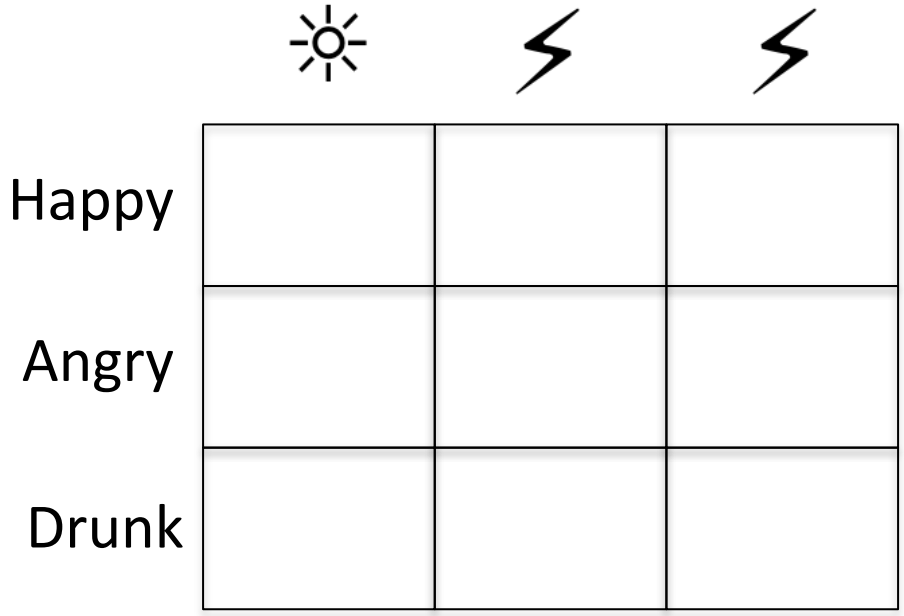
Using Big-O

- Your data might be too big to analyze in reasonable time.
- Do an experiment:
  - Use a very small data set (n << actual n).
  - Run the algorithm and measure t (execution time, hopefully reasonable).
  - Now you know t and n in the Big-O formula. Solve for k.
- Compute t for the full data set:
  - You know k and n in the Big-O formula. Solve for t.

Viterbi Algorithm

Example: Find the Viterbi (most likely) path through the Thor HMM that generates Sun/Thunder/Thunder.

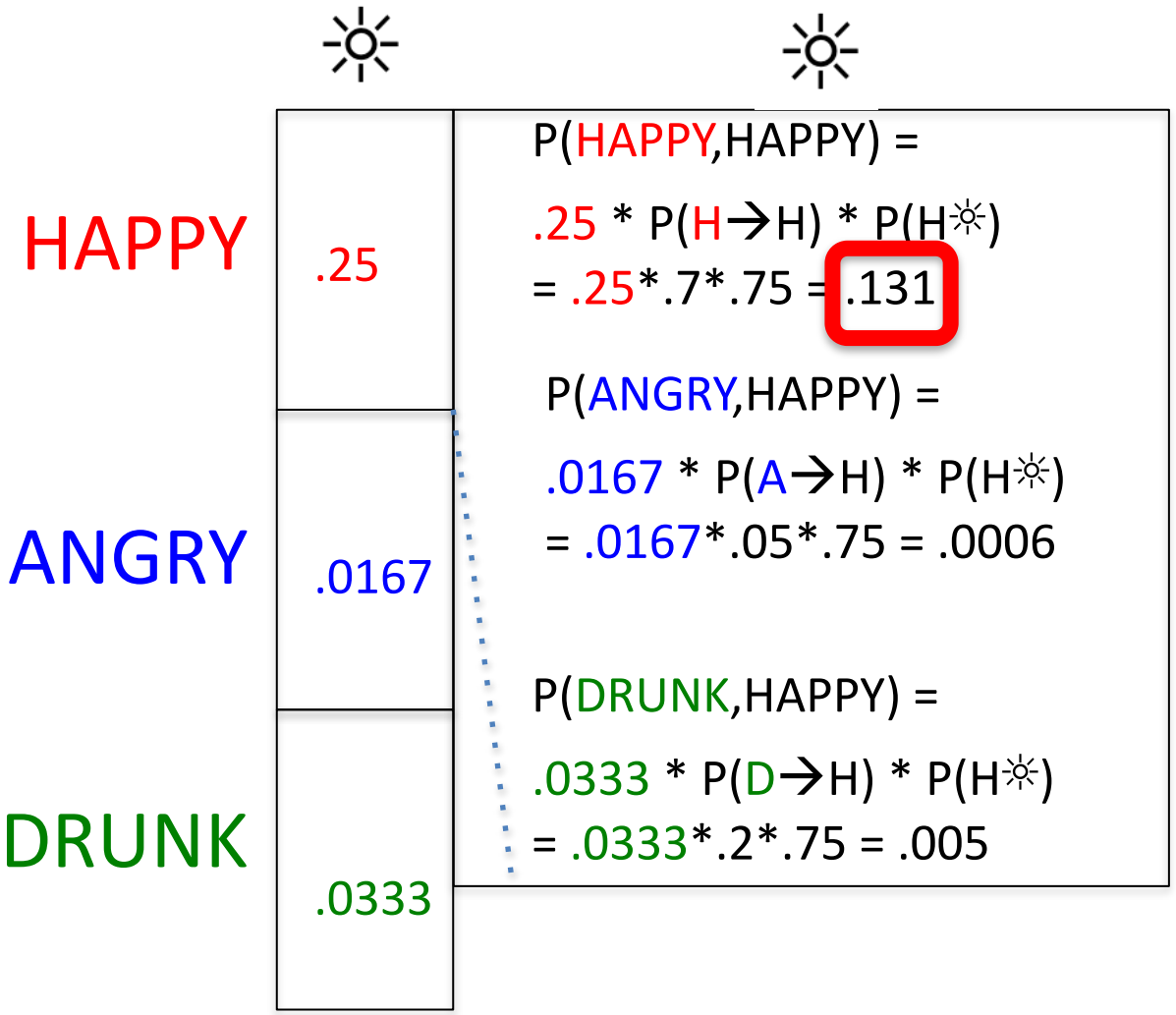
Step 1: Draw a grid. 1 row for each state of the HMM, 1 col for each member of the observation.



Viterbi Algorithm, first column:

	☼	⚡	⚡
Happy	P(start = State for row) * P(Emit weather for col from state for row)		
Angry	P(start = State for row) * P(Emit weather for col from state for row)		
Drunk	P(start = State for row) * P(Emit weather for col from state for row)		

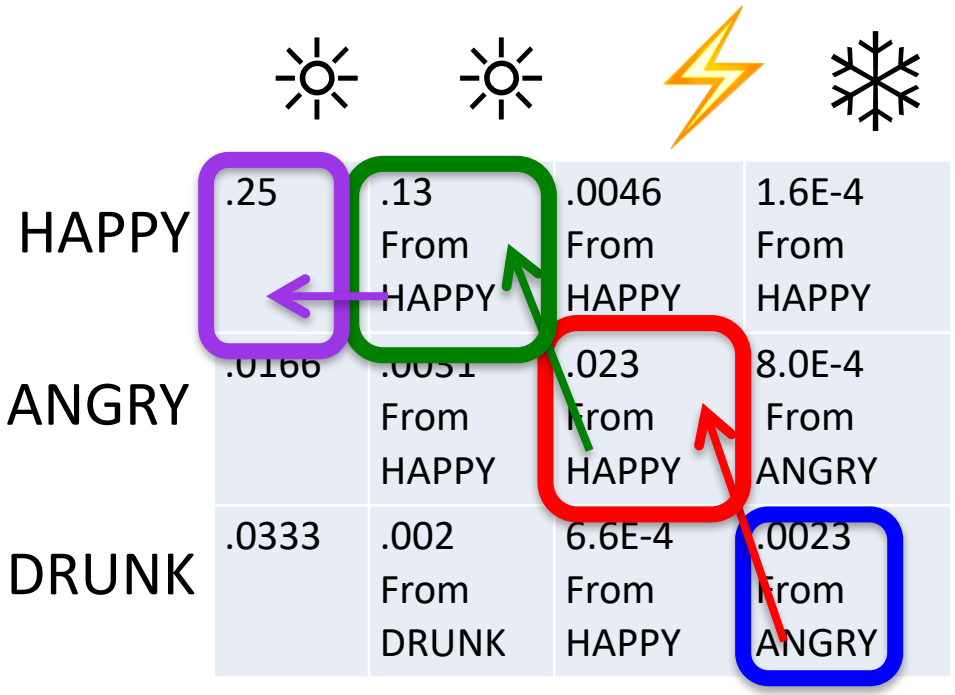
Step 3: Fill in 2nd column, HAPPY cell



Max of the 3 probs is for path = HAPPY,HAPPY

→ This is the most probable path that emits ☼☼ and ends at HAPPY. Its probability is .131

Retain .131 and remember that state from prev col was HAPPY



HAPPY	.25	.13 From HAPPY	.0046 From HAPPY	1.6E-4 From HAPPY
ANGRY	.0166	.0031 From HAPPY	.023 From HAPPY	8.0E-4 From ANGRY
DRUNK	.0333	.002 From DRUNK	6.6E-4 From HAPPY	.0023 From ANGRY

The rest of the Viterbi path is found by tracing back along the "From" states

Viterbi path = HAPPY HAPPY ANGRY DRUNK

CpG Islands in the human genome

ISLANDS	A	C	G	T
A	.180	.274	.426	.120
C	.170	.368	.274	.188
G	.161	.339	.375	.125
T	.079	.355	.384	.182

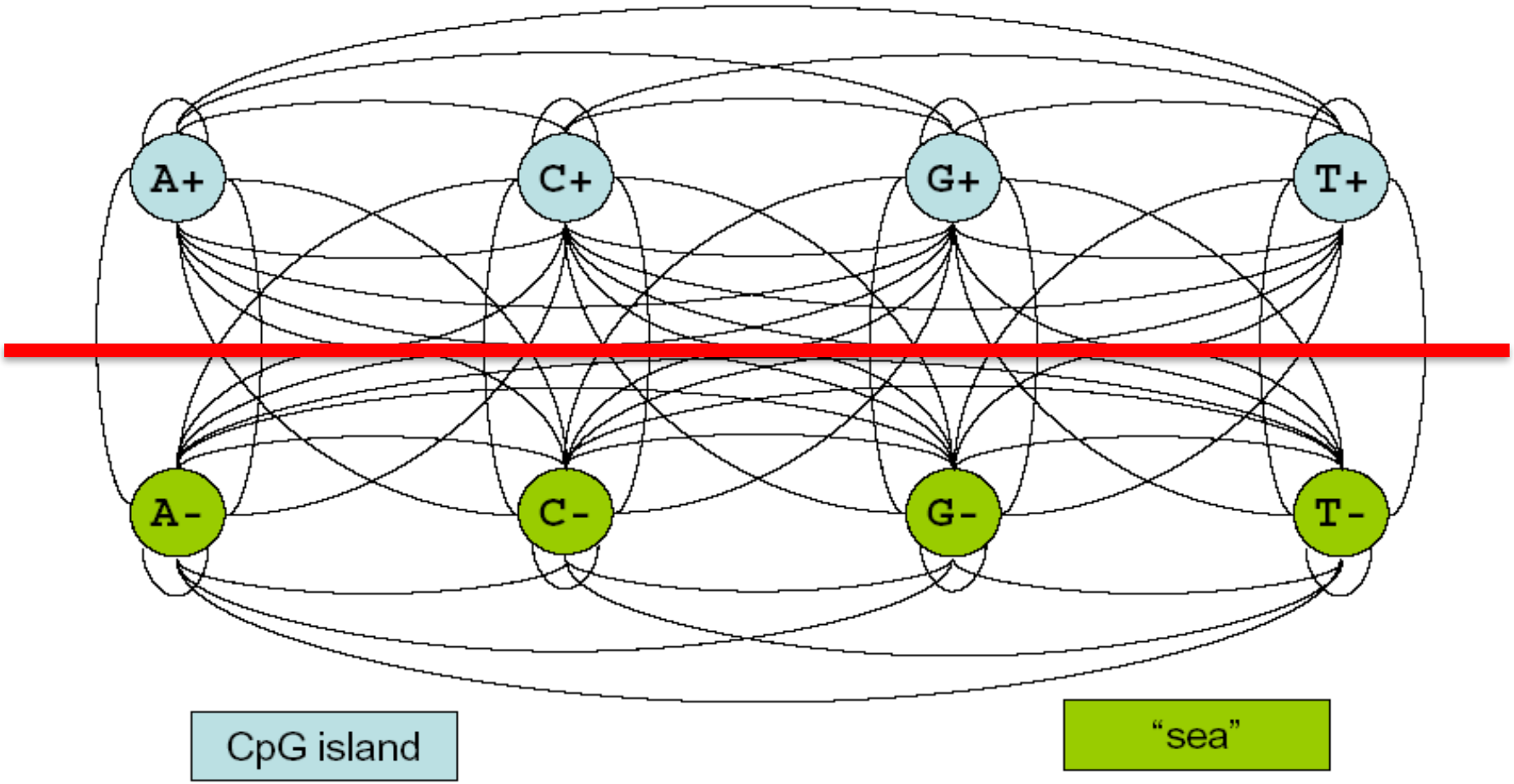
1st nucleotide

2nd nucleotide

NOT ISLANDS	A	C	G	T
A	.300	.205	.285	.210
C	.322	.298	.078	.302
G	.248	.246	.298	.208
T	.177	.239	.292	.292

The CpG HMM

- 8 states
  - A+, C+, G+, T+, A-, C-, G-, T-
  - + means an island state, - means a not-island state
- From each state, only 1 emission is possible
  - That's a bit weird for an HMM
  - Emissions are 'A', 'C', 'G', and 'T'
  - States A+ and A- can only emit 'A', etc
- State names look like emissions so don't get confused



Prob(Any "+" state → Any "–" state) = zero

This machine can never model or evaluate a transition *to* or *from* a CPG island

Fixing the transition probabilities

- Analysis shows P(Any + state → Any – state) ~= P(Any - state → Any + state) ~= 0.01% = 0.0001 = P(crossing the red line)
- Multiply all transition probabilities by 0.9999
- Now P(transition from any state) has been reduced from 1 to .9999
- Let P(A+→A-) = P(A+→C-) = P(A+ →G-) = P(A+ → T-) = .0001/4 = .000025
- Similar for all other island-to-not-island and not-island-to-island transitions

Remember The Goal

- Input: long DNA sequences, maybe entire chromosomes
- Output: software-predicted locations of all CpG islands
- How do we do that with the CpG HMM?

How to do it

- Compute the Viterbi path through the CpG HMM for your input
  - E.g. A+ C+ G+ C+ G+ A- T- G+ …
- Look at the “+” or “-” parts of the state names
  - + + + + + - - +
- Runs of “+” of length >= 200 are CpG islands

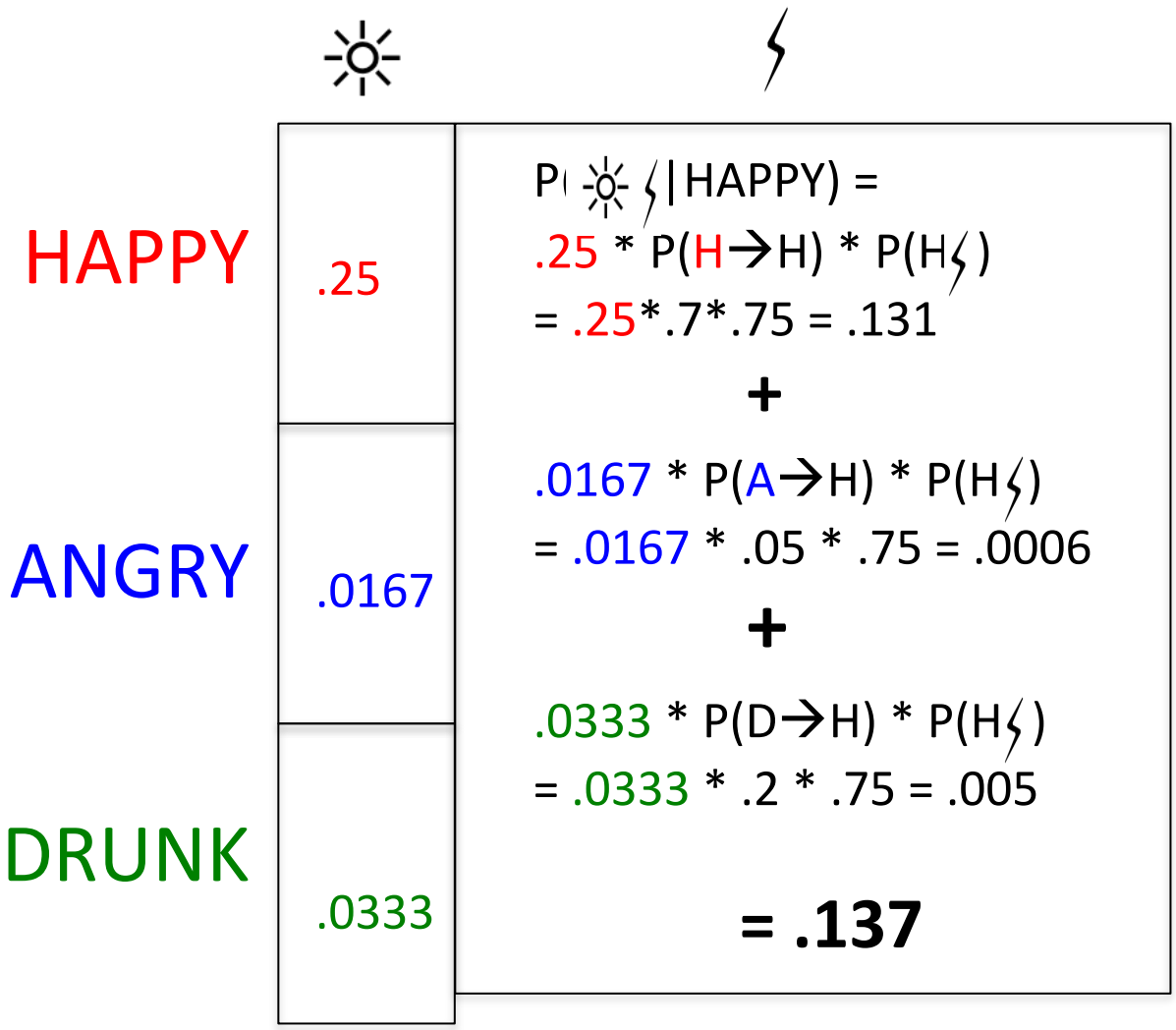
The Forward Algorithm

- Computes P(HMM emits observation) = P(obs|HMM)
  - No matter what the path might have been
- Like Viterbi, but…
- Given string of observations, Viterbi finds most likely state path, and that path's probability
- But P(obs|HMM) = sum of probabilities over all paths that can emit that observation, including the Viterbi path and many others
- P(HMM emits observation) is often what we care about for protein identification using pHMMs
- Viterbi is about max(probabilities), FA is about Σ(the same probabilities)
- Example: what is the probability that the Thor HMM emits

The Forward Algorithm starts like Viterbi :

	☼	⚡	⚡
Happy	P(start = State for row) * P(Emit weather for col from state for row)	P(Emit the 1st weather from this state)	
Angry	P(start = State for row) * P(Emit weather for col from state for row)	P(Emit the 1st weather from this state)	
Drunk	P(start = State for row) * P(Emit weather for col from state for row)	P(Emit the 1st weather from this state)	

2nd column, HAPPY cell



- Before: Viterbi chose max of the 3 terms (.131, .0006, and .005)
- With the Forward Algorithm, take the sum of all 3
- = .137 (was .131)
- That's not much difference, but it grows as the algorithm progresses

Finishing FA
No “From” notes, no traceback

	☼	⚡	⚡
HAPPY	.025	.0091	5.0E-4
ANGRY	.0166	.057	.03
DRUNK	.0333	.0037	.0017

.0005 + .03 + .0017 = .032

➔ P(☼⚡⚡) = .032

More formally:

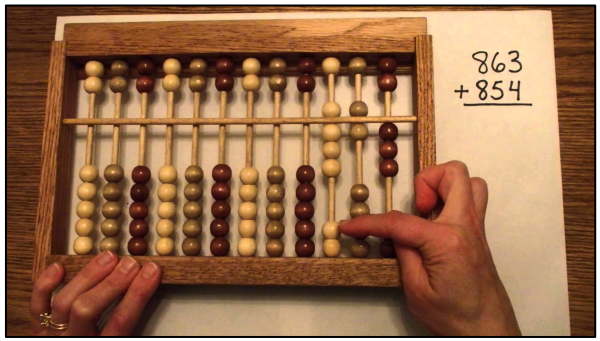
P(☼⚡⚡ | this HMM) = .032

5x10^-324: A special number

- The smallest fraction that most computers can represent
- If you multiply this by *any fraction*, the result will be rounded down to zero
- Viterbi or Forward score(any long enough sequence | any realistic HMM) will be wrongly reported as zero
- There's an easy fix for Viterbi
- ***No fix is possible for FA (or BA)***

Remember logarithms?

- log(x) = the power to which we raise 10, in order to get x
- 10 ^ log(x) = x
- log(100) = 2, log(1000) = 3, log(1/10) = -1
- log(a*b) = log(a) + log(b)
- Adding is easier than multiplying, especially on an abacus in 1614



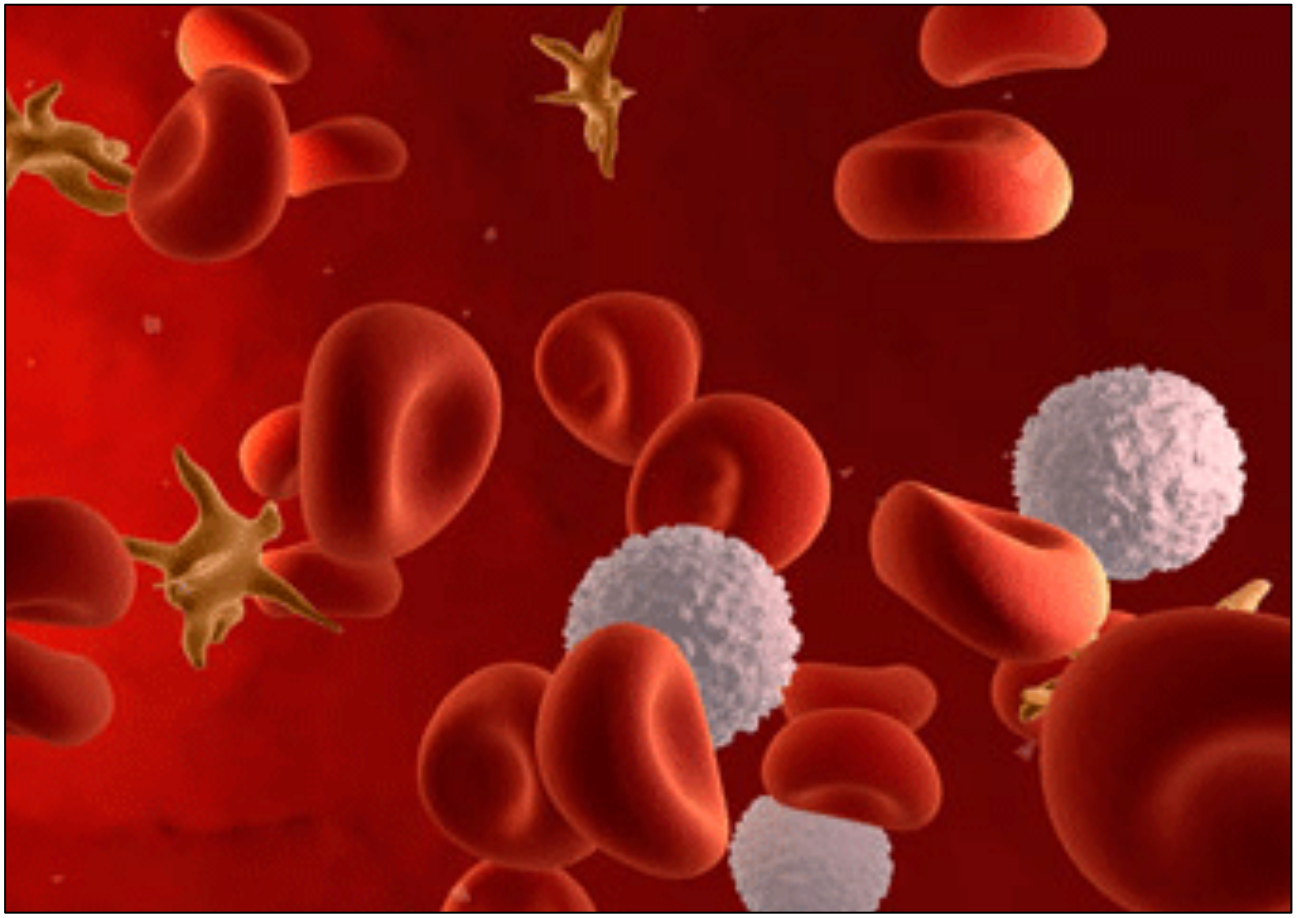
What we want, what we settle for

- What we want
  - Given a sequence and some HMMs, compare probabilities that the HMMs emitted the sequence
  - Requires Forward Algorithm
- What we settle for (when sequence is too long)
  - Given a sequence and some HMMs, compare Viterbi probabilities for the sequence from each HMM
  - Use Viterbi probability (probability of best path) as a "proxy" for sum of probabilities of all paths

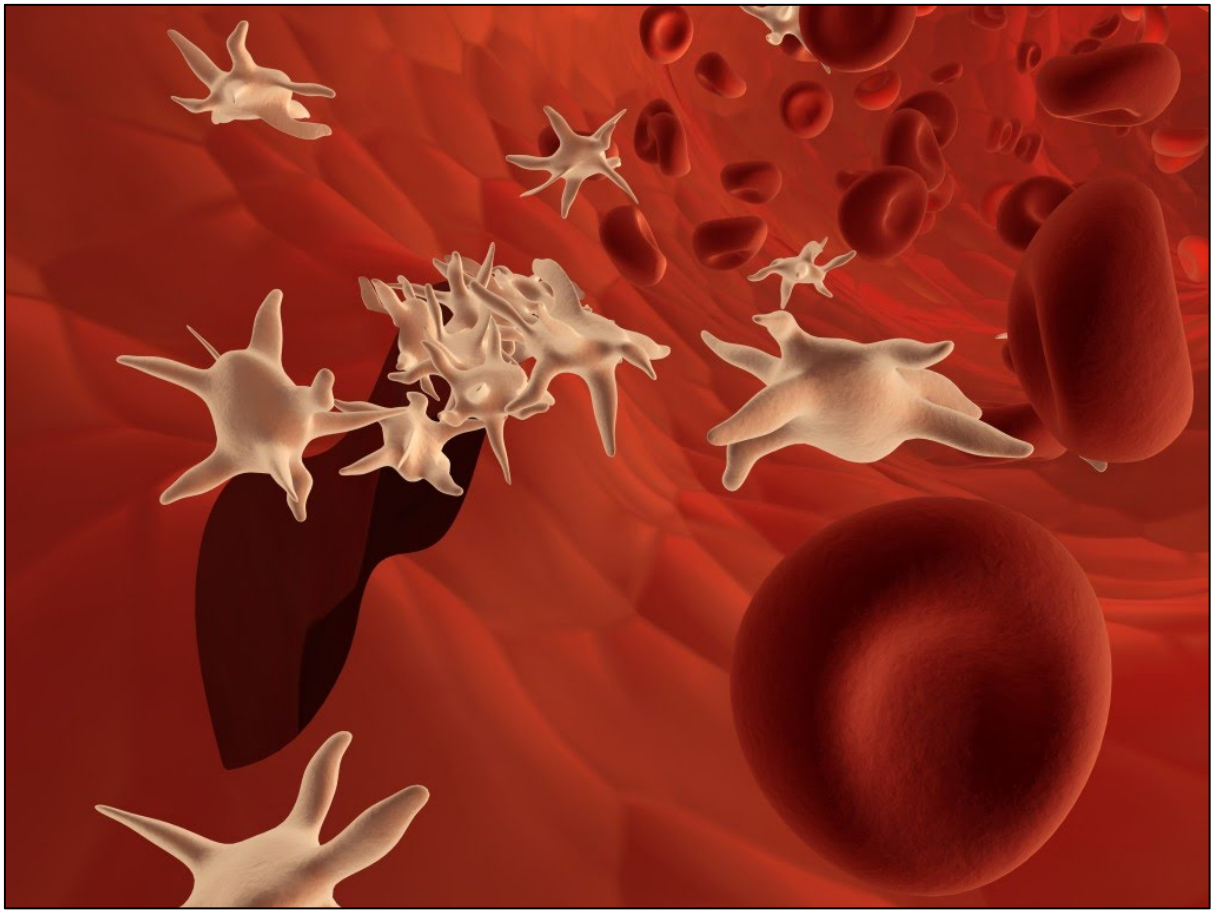
Protein HMMs

- In bioinformatics, most HMMs are about protein sequences, not nucleotide sequences.
- Model proteins using HMMs
  - "Profile" HMMs
  - Identify sequences by checking P(seq|HMM)
- Model protein families using HMMs
  - Composite HMMs

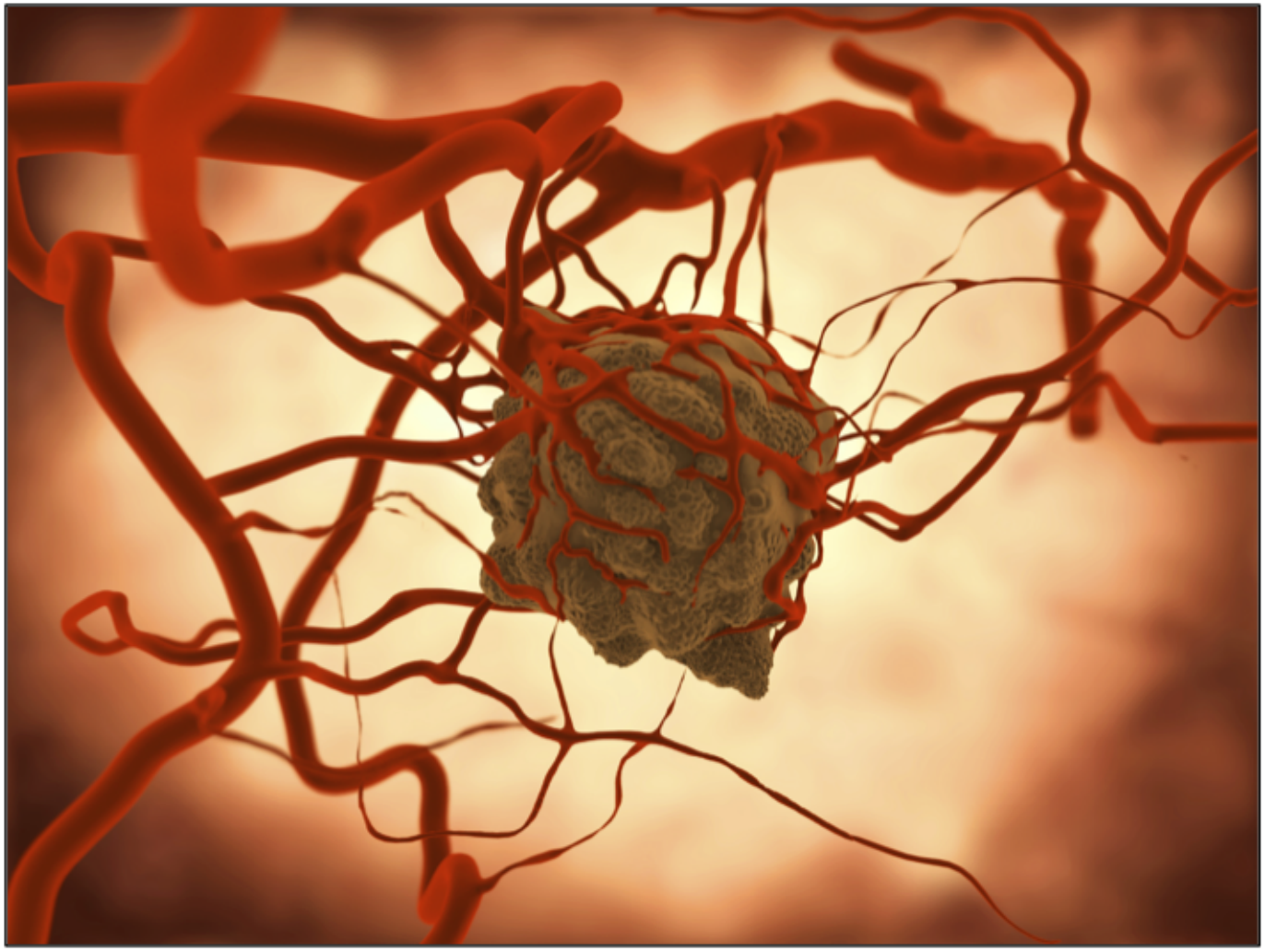
Let's start with an example: Platelet-derived growth factor (PDGF)



Platelets heal wounds by secreting a hormone that stimulates growth



Tumors subvert PDGF to recruit blood vessels and steal resources

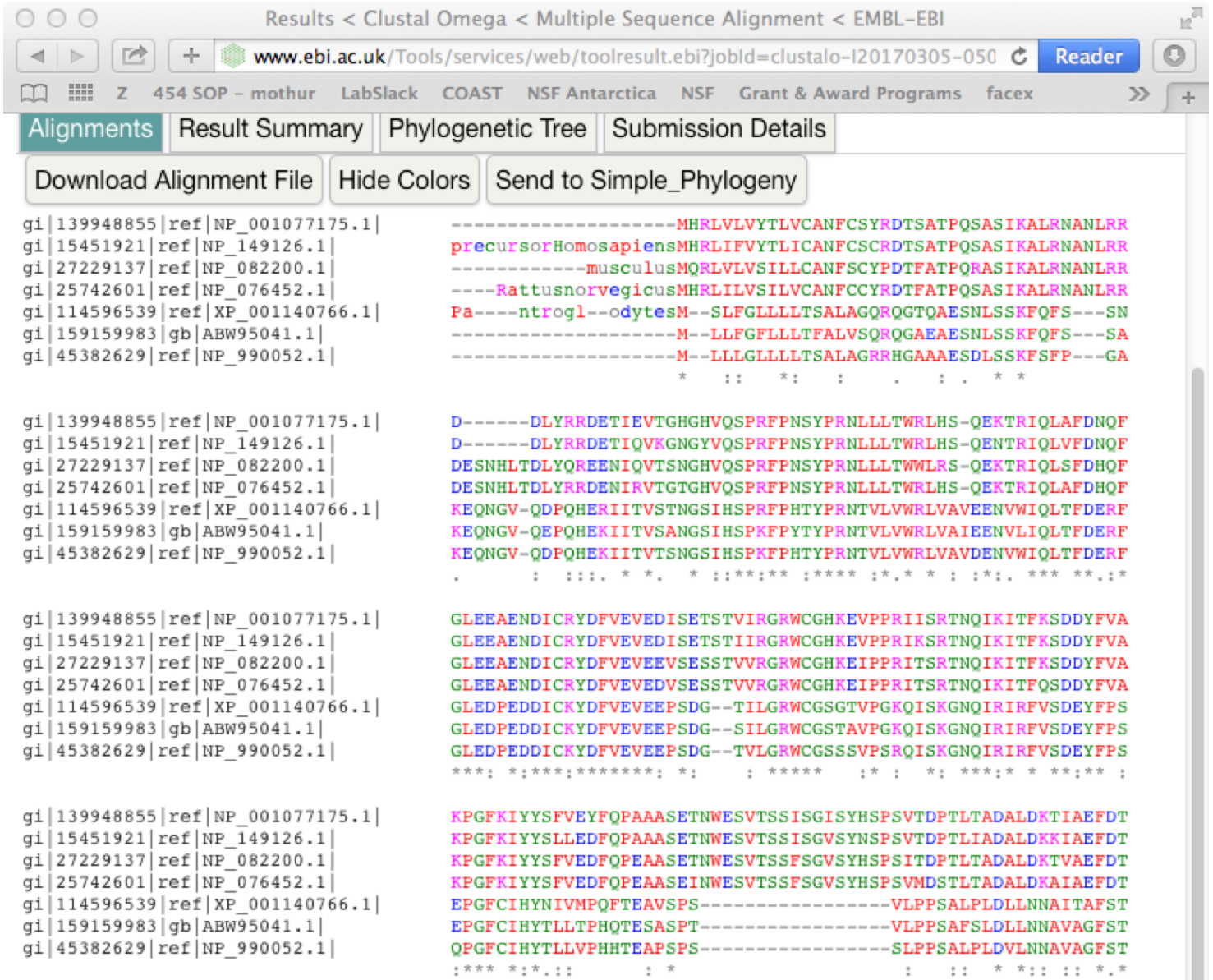


How can we identify PDGF in a tumor genome?

- Bioinformatically identify PDGF genes
- Bioinformatically translate to protein sequences
- BLAST against a reference database of known PDGF proteins?
  - Might work, might not
  - Tumor genomes mutate wildly ➔ highly variable domains might lower BLAST scores
- So build a HMM that describes PDGF conserved domains
  - If a gene evolved from PDGF but lost the ability to promote growth, it's not a threat
  - The real threat: mutated PDGF genes that can still function, because they kept their conserved domains
  - Compute FA score (if possible) else Viterbi score of candidate tumor genes
  - Investigate high-scoring sequences

Step 1: Collect a training set (search GenBank)

Step 2: Align (e.g. ClustalΩ)



Step 3: Look for conserved domains

Step 4: Build a profile Hidden Markov Model (pHMM)

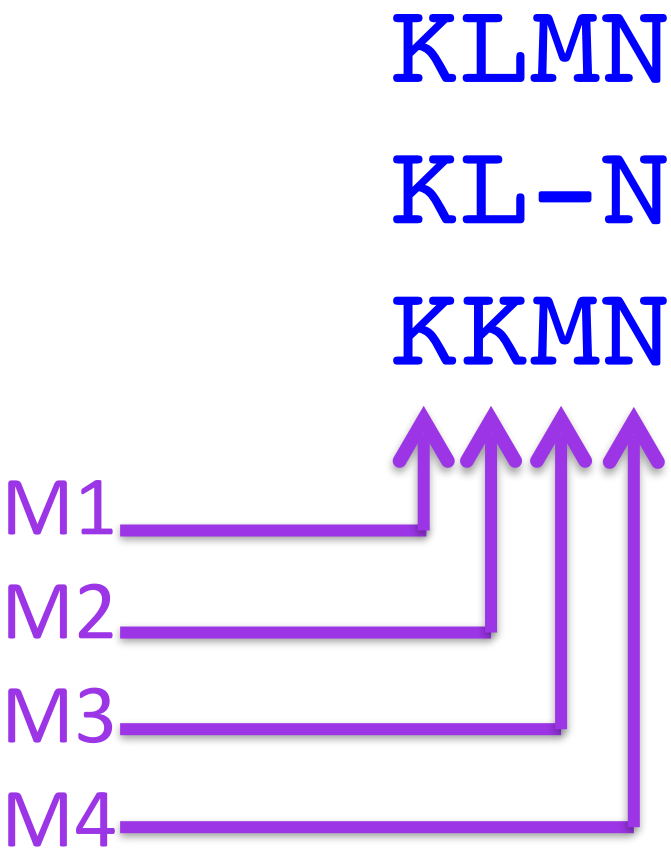
To build a protein HMM …

- Collect trusted representatives of the protein you want to model
  - "Positive training set"
- Align the positive training set

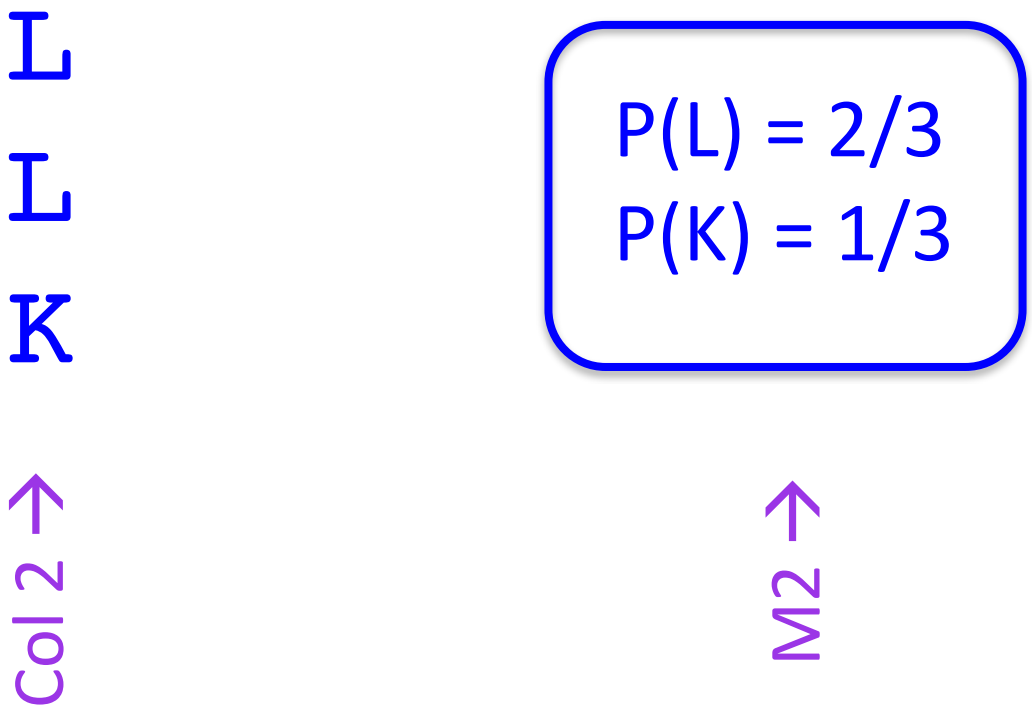
KLMN, KLN,KKMA →

```
KLMN
KL-N
KKMN
```

Each alignment column becomes a state



Frequency of char in col becomes its emission probability



Do that for every column

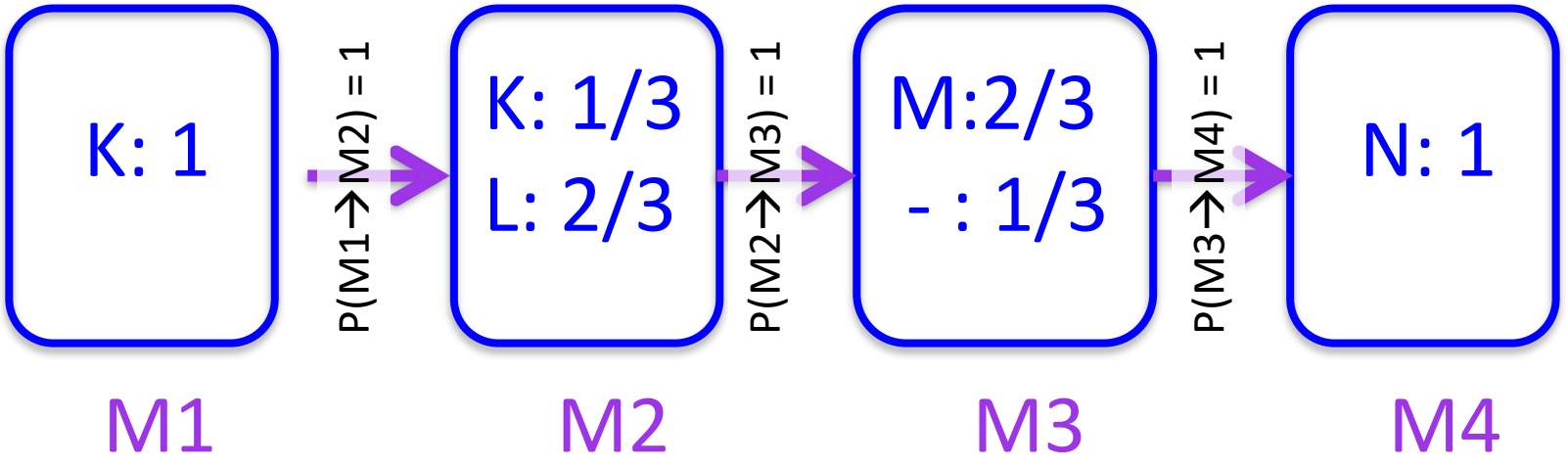
KLMN

KL-N →

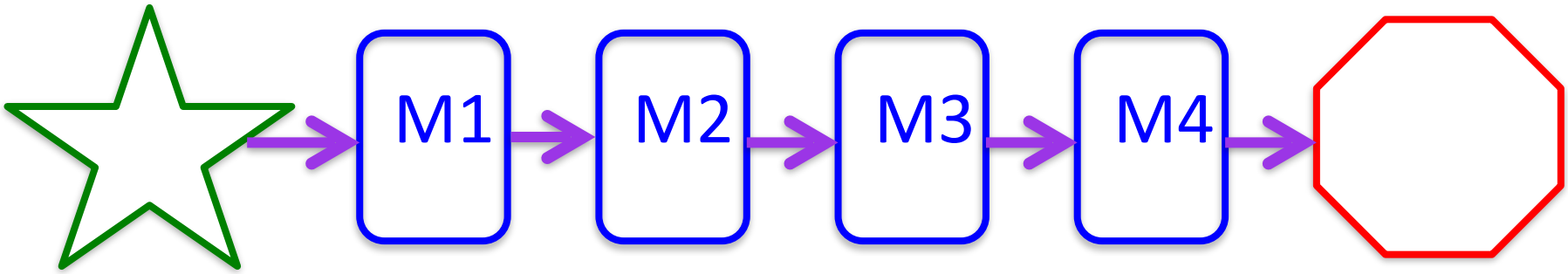
KKMN

Transitions: Only go from one column's state to the next column's state

```
KLMN
KL-N →
KKMN
```

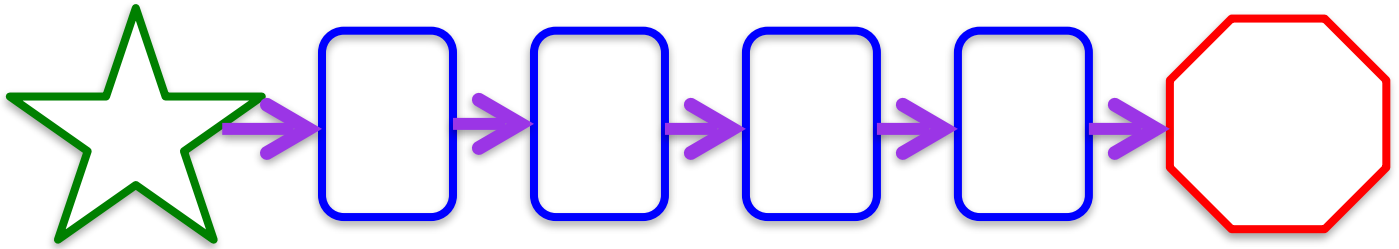


Special states: Start & Stop



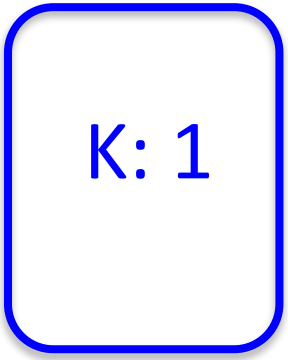
- Start:
  - All paths start here
  - No emissions
  - Transitions from here reflect initial probs
- Stop
  - All paths end here
  - No emissions

Too Simple: 3 problems



- P(Any sequence containing aas not represented in the positive training set) = 0
- Indel isn't really an emission in the same sense as the amino acids
- This HMM can only handle sequences of length=4
  - P(Any sequence of any other length) = 0

K = Lysine: Polar, hydrophilic



- Similar to Arginine (R) & Histidine (H)
- P(R|M1) = P(H|M1) = 0
- This model says that in state M1, it is *impossible* for R or H to appear
  - Anywhere on the planet
  - Or any other planet
  - Ever
- Model gives high score for KLMN
- Model gives zero score for HLMN, should give low score
  - Unless you're 100% certain that H is impossible in column 1

Pseudocounts

- A more realistic column from positive training set alignment might have counts like this:
  - A=100, K=120, L=200, M=50
  - Training set size was 470
  - P(A|this state) = 100/470 = .213
  - P(C|this state) = 0 /470 = 0
- Pretend the column also contained 1 of every unrepresented amino acid
  - C = D = E = F = G = H = N = P = Q = R = S = T = V = W = Y = 1
  - P(A|this state) = 100/486 = .206
  - P(C|this state) = 1 / 486 = .00206

Like a tax on probabilities of represented aas

Pseudoprobabilities

- Like pseudocounts, but tax probabilities rather than frequencies
- For each state, decide on a small value ε for P(emit *any* unrepresented aa|that state)
  - E.g. .01 or .001
- Reduce P(every represented amino acid|that state) by same small amount, total = ε
- Set P(every unrepresented aa|that state) to same small amount, total = ε

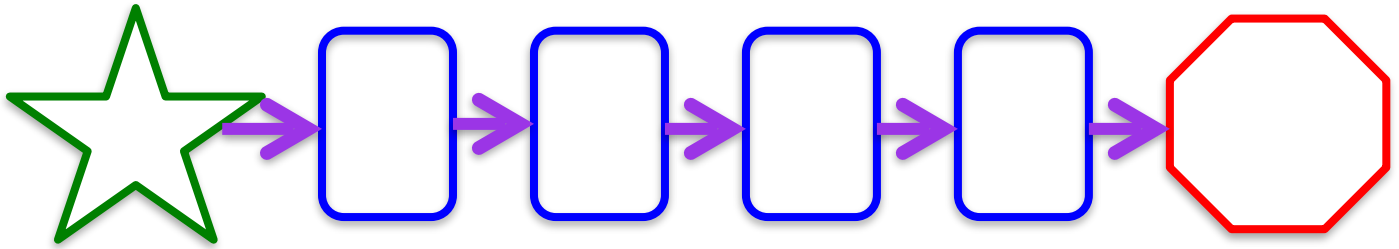
Pseudoprobabilities Example

- A=100, L=250, M=60, P=90
  - Original P(A) = 100 / (100+250+60+90) = 100/500 = .2
  - Original P(L) = 250/ (100+250+60+90) = 250/500 = .5
  - Original P(M) = 60/ (100+250+60+90) = 60/500 = .12
  - Original P(P) = 90/ (100+250+60+90) = 90/500 = .18
  - Orig P(A) + Orig P(L) + Orig P(M) + Orig P(P) = 1
- Decide that ε = .01
- 4 represented aas → reduce P(every represented aa) by ε/4 = .0025
  - P(A) = .1975, P(L) = .4975, P(M) = .1175, P(P) = .1775
- 16 unrepresented aas → Set P(every unrepresented aa) to ε/16 = .00015625
- Total emission probs from this state still = 1

With pseudocounts or pseudoprobabilities

- Any state can emit any aa, though some emission probabilities are small
- Sequences that would have zero probability now have positive probability

Too Simple: 3 problems



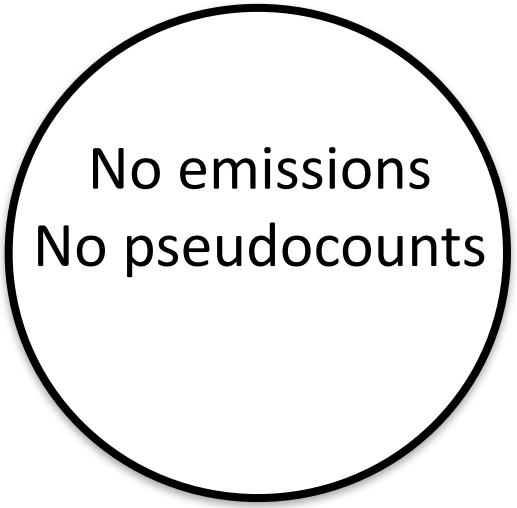
- P(Any sequence containing aas not represented in the positive training set) = 0 ✔
- Indel isn't really an emission in the same sense as the amino acids ?
- This HMM can only handle sequences of length=4
  - P(Any sequence of any other length) = 0

Gaps in the alignment

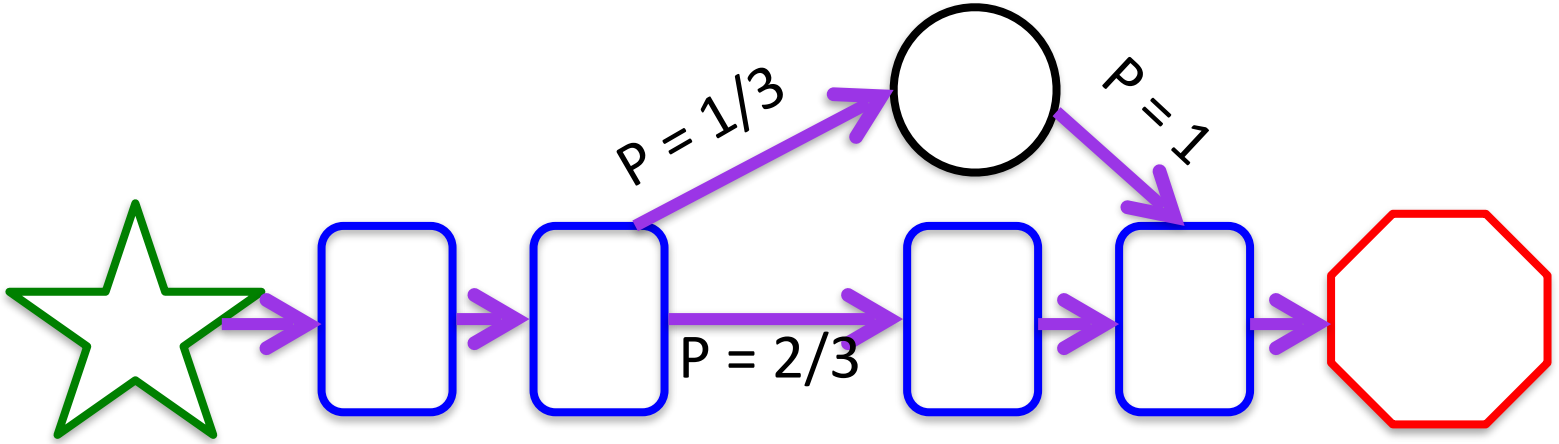
```
KLMN
KL-N
KKMN
```

- Biochemically different from amino acids.
- A gap has no physical existence.
- Don’t represent in alignment-col states (“match states”).
- Represent in special “delete states”, usually drawn as circles.

A Delete State



```
K   L     M   N
K   L     -   N
K   K     M   N
```

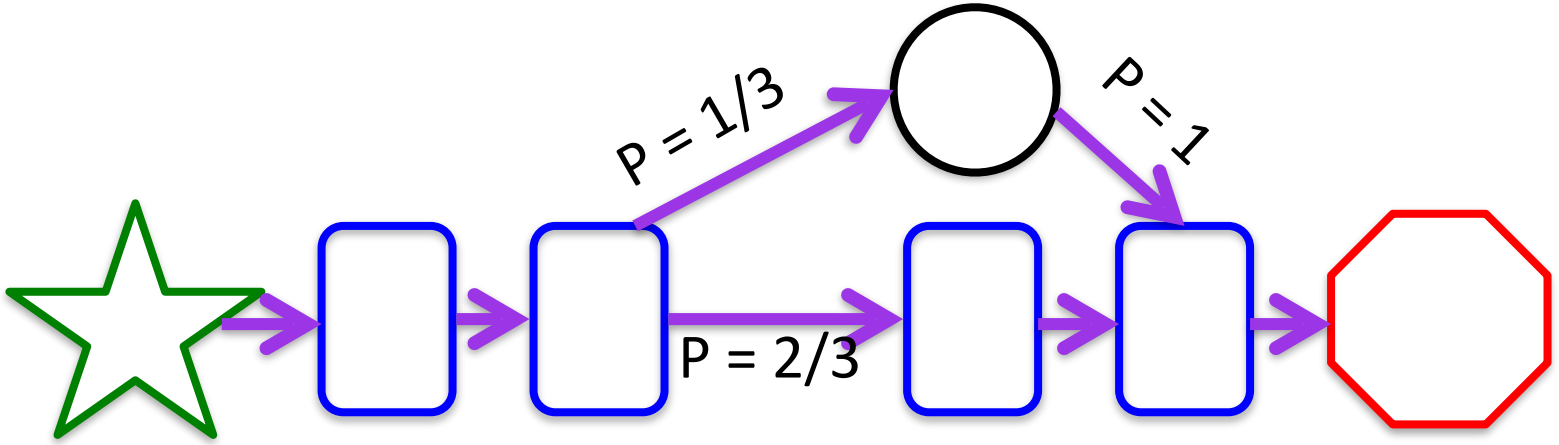


What about a more complicated alignment?

K L M N

K L – N

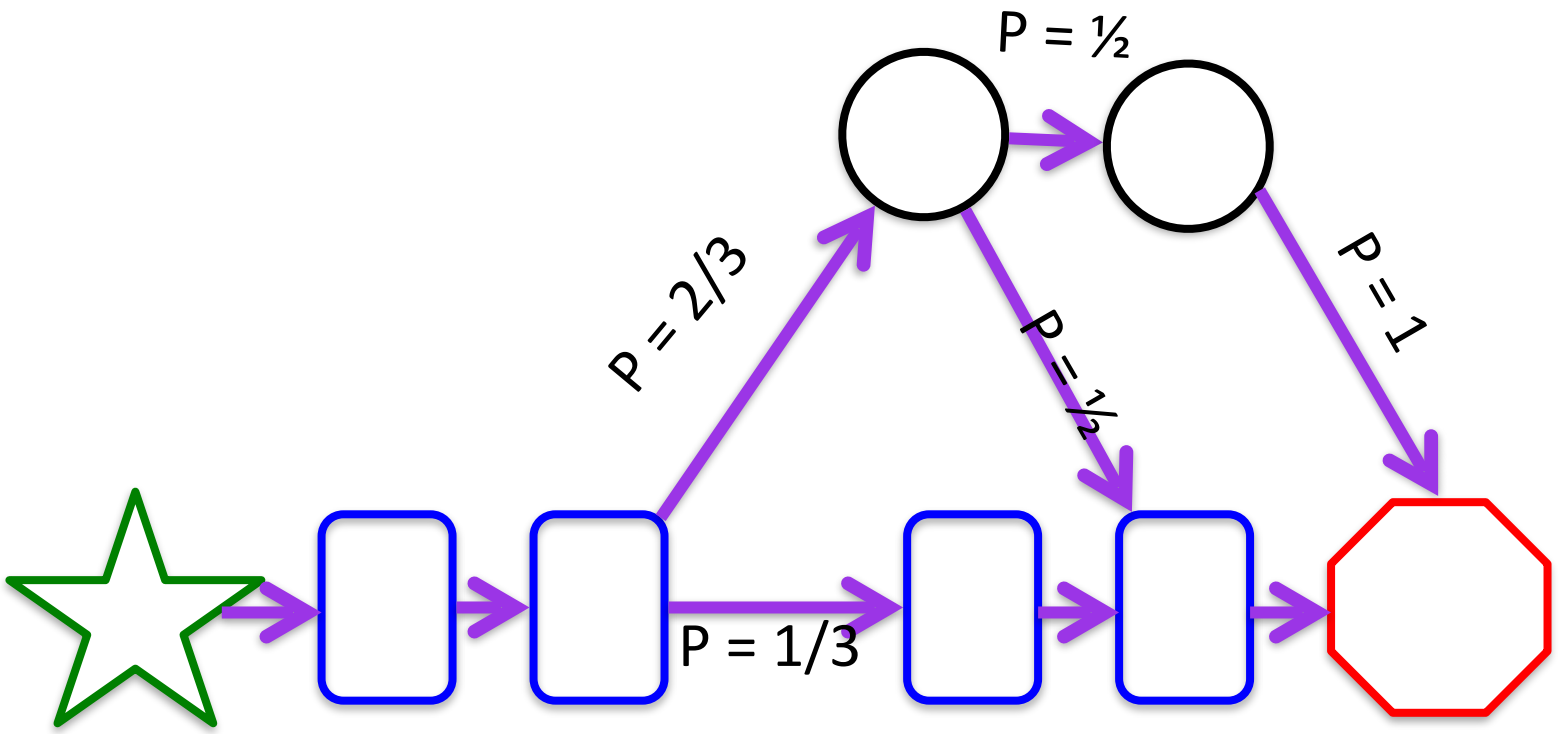
K K – – ???



After state M2:

- P(length=4) = 1/3
- P(len=3, skip M3) = 1/3
- P(len=2, skip M2&M3) = 1/3

```
K  L  M  N
K  L  –  N
K  K  –  –
```



or, more simply …

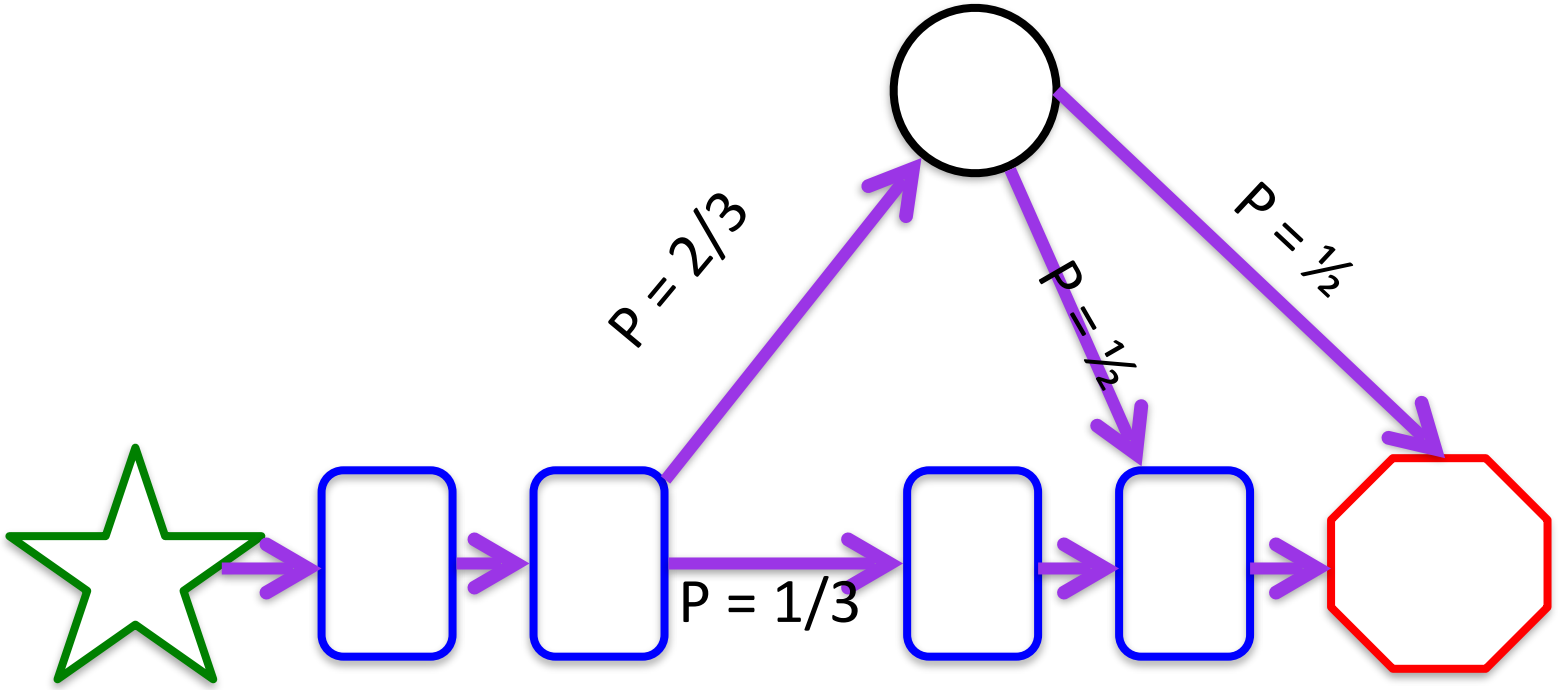
After state M2:

- P(length=4) = 1/3
- P(len=3, skip M3) = 1/3
- P(len=2, skip M2&M3) = 1/3

K L M N

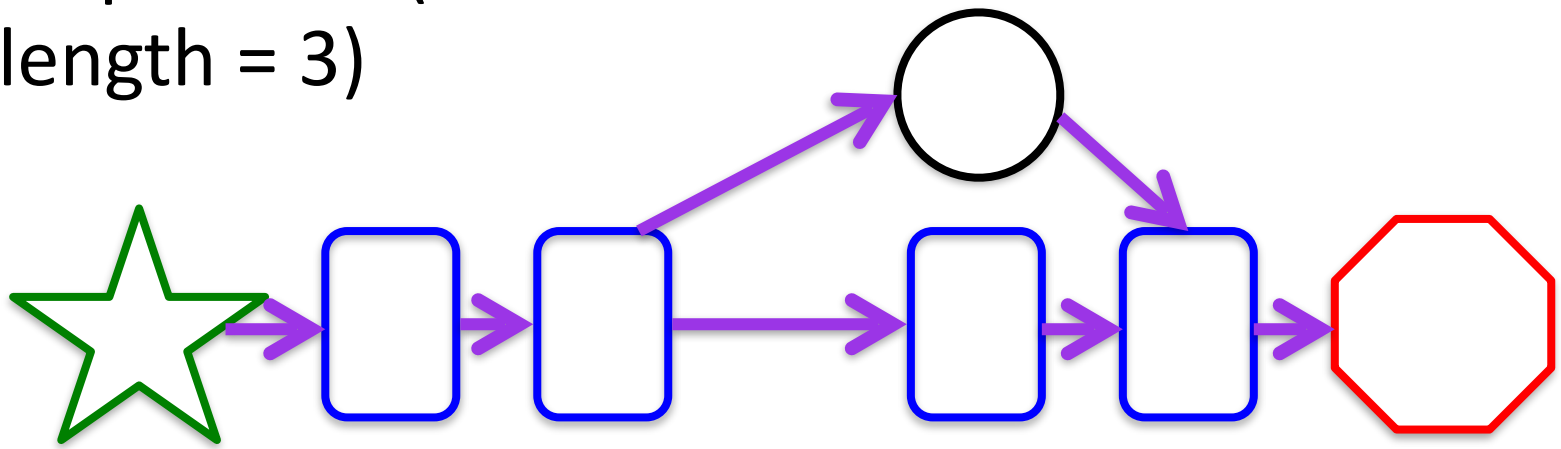
K L – N

K K – –

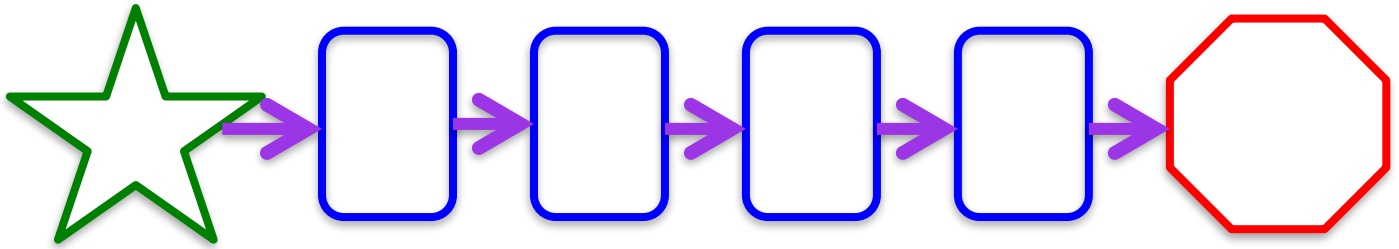


- Indel indicates a sequence that is shorter than the alignment of the training set.
- Delete states let HMM deal with shorter sequences. (Here: length = 3)

```
K  L  M  N
K  L  -  N
K  K  M  N
```



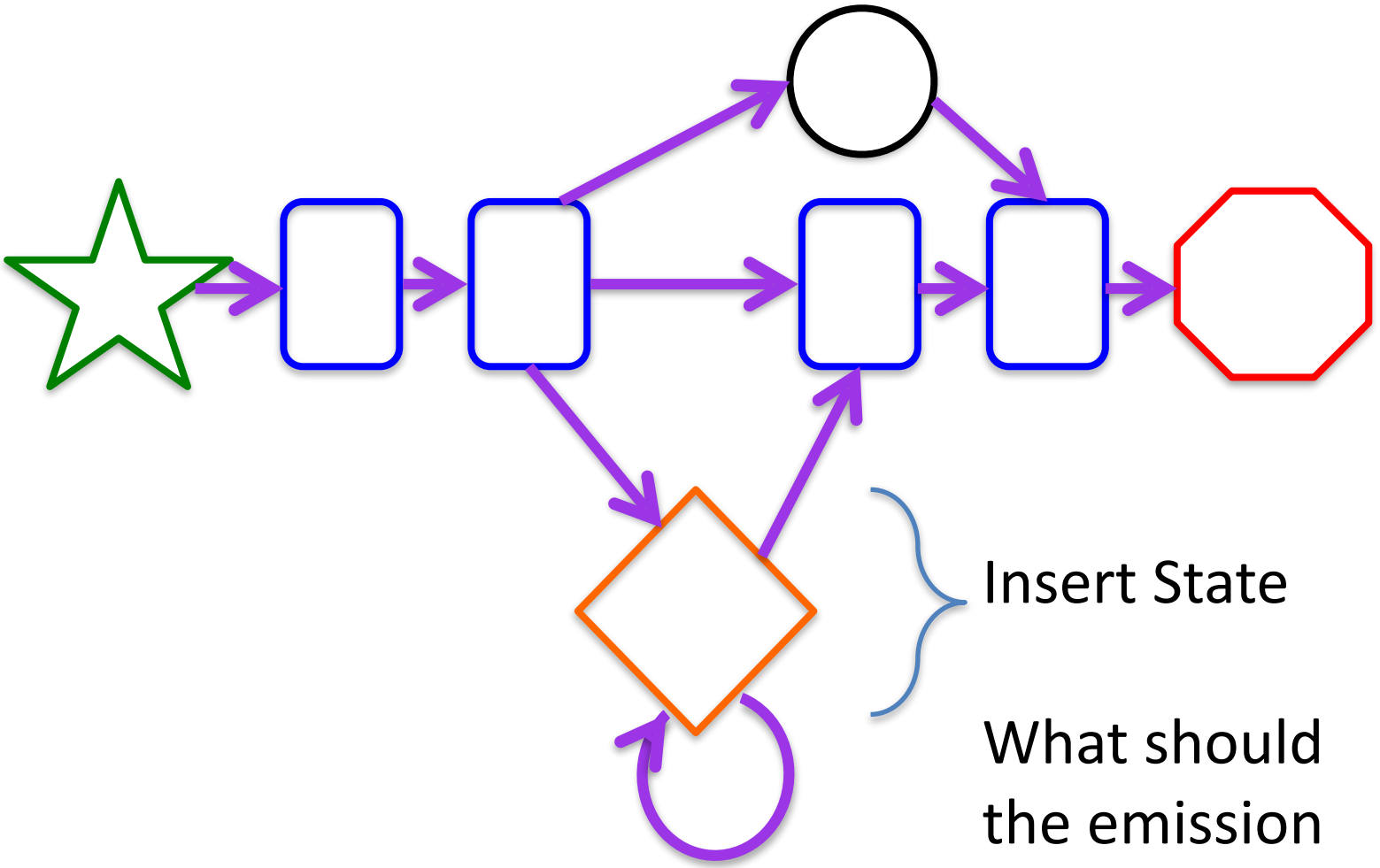
Too Simple: 3 problems



- P(Any sequence containing aas not represented in the positive training set) = 0 ✔
- Indel isn't really an emission in the same sense as the amino acids ✔
- This HMM can only handle sequences of length=4 ?
  - P(Any sequence of any other length) = 0

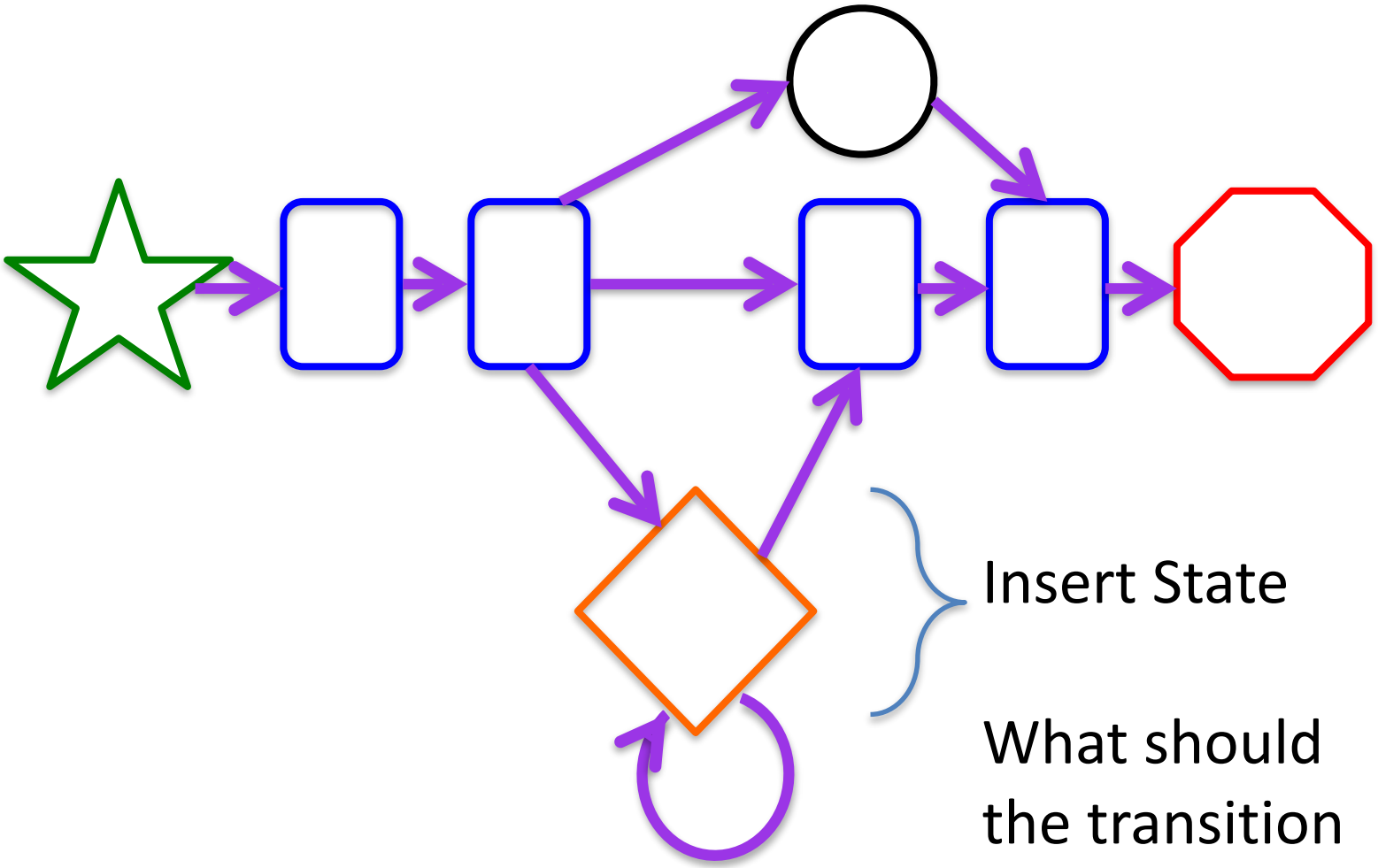
Problem: HMM can only handle sequences of same length as alignment of positive training set

- Delete states take care of shorter sequences
- Another special state takes care of longer sequences:
  - Insert States
  - Usually drawn as diamonds between/below the match states


Insert State
What should the emission probabilities be?

How do we set the emission probabilities for an insert state?

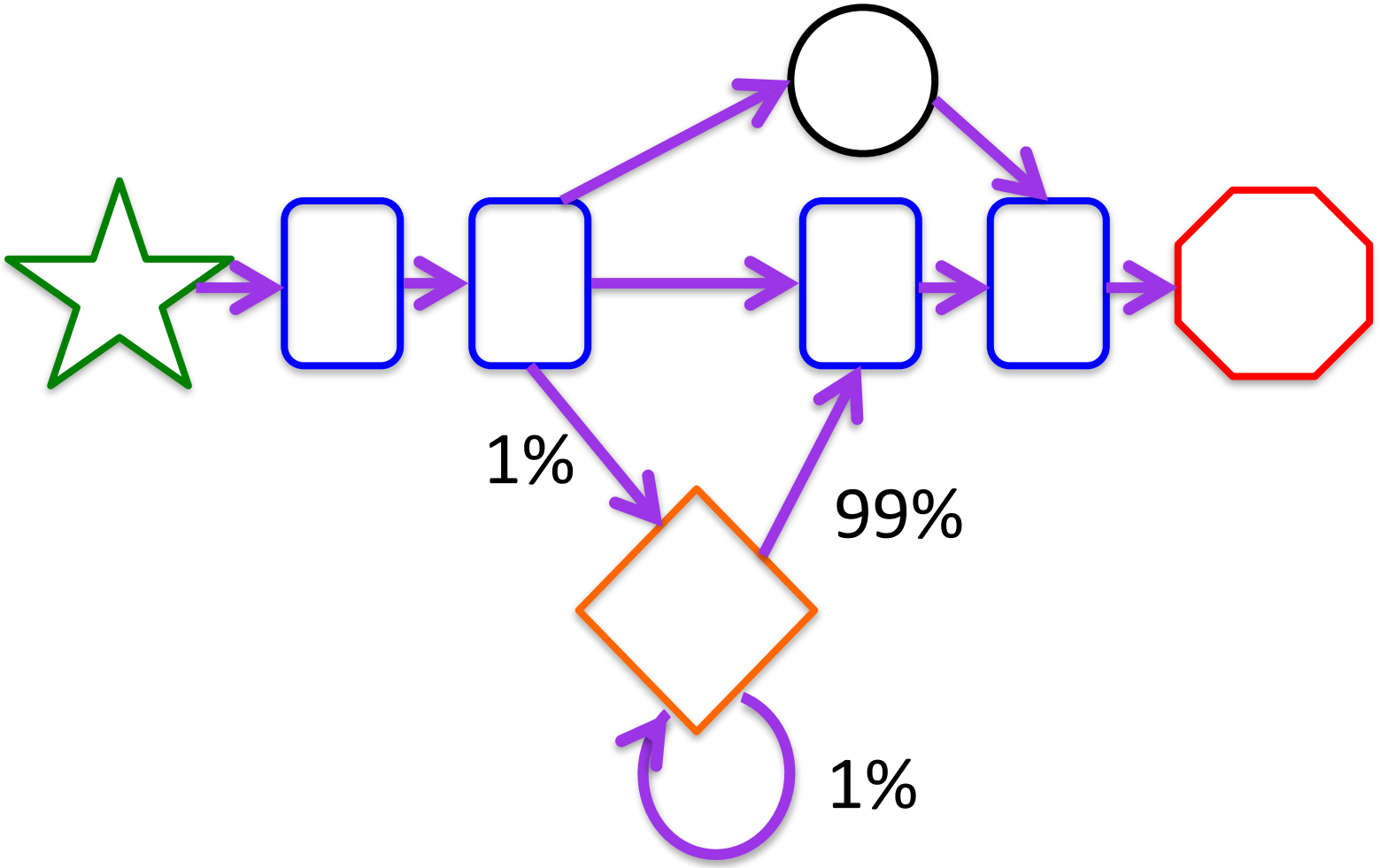
- The match states represent columns in the alignment of the positive training set.
- The insert states represent insertions that don't appear anywhere in the positive training set.
- So we don't know anything about these mysterious insertions.
  - We're just making sure that *if* they happen, the HMM can handle the extra length
- There is no reason to believe any aa is more or less likely to be emitted than any other aa.
- So set all emission probabilities to be equal (.05).
- In pHMM diagrams, such insert states are drawn as empty diamonds


Insert State
What should the transition probabilities be?

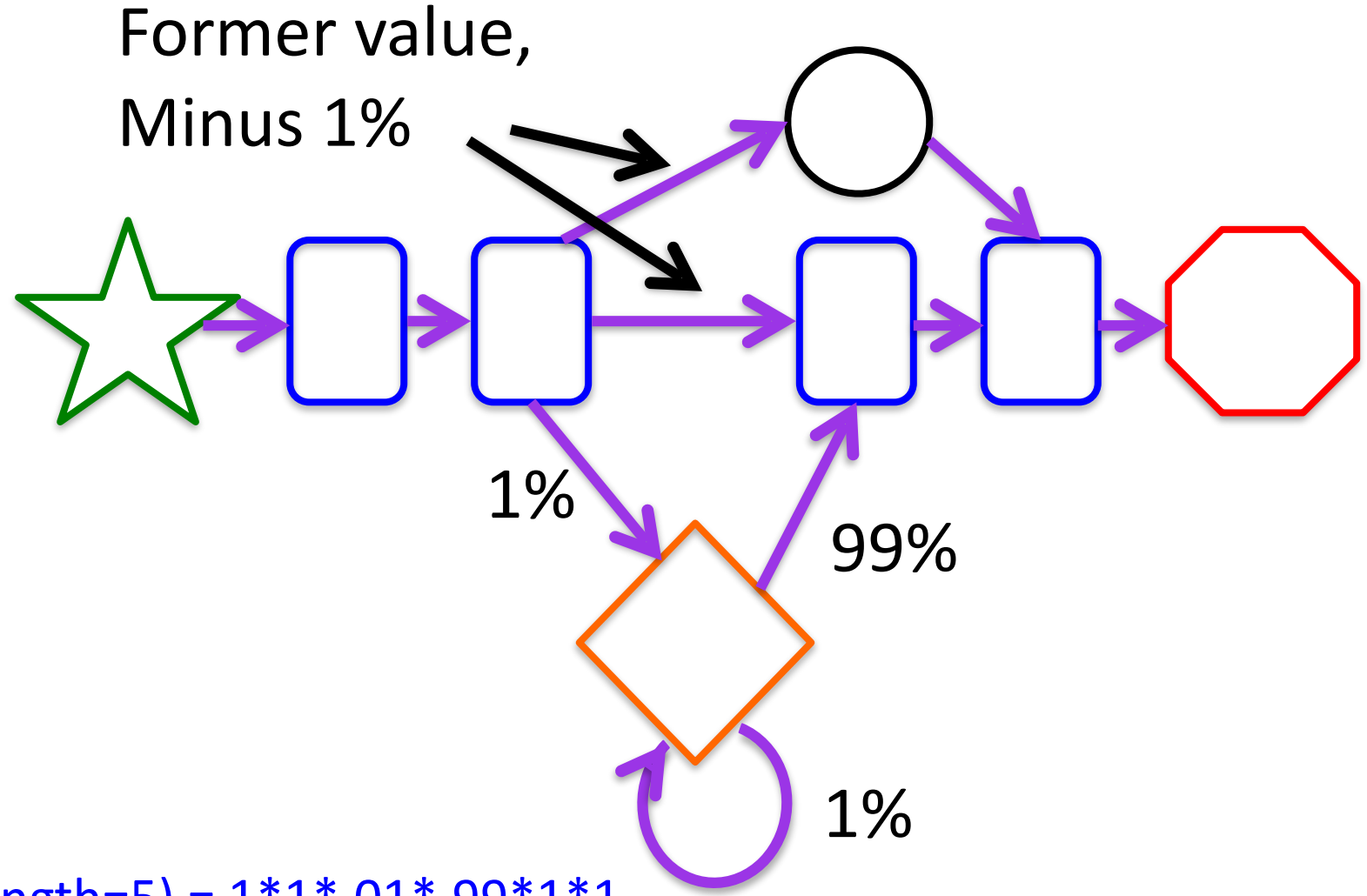
Insert state transition probabilities

- Best we can do is a good guess
- Insert states are for sequences that are longer than the training alignment
- If we had and examples, we would have included some in the training set
  - ➔ Then the alignment would be wider, and we wouldn't need to use Insert States!
- We don't know how long a mysterious "longer sequence is"
  - A good guess: single inserted aa is more likely than 2 consecutive inserted aas, which are more likely than 3, etc.

1% seems to work:



1% seems to work:

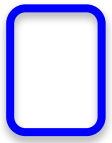


P(length=5) = 1*1*.01*.99*1*1

Profile HMM (pHMM) Algorithms

- A pHMM is just an HMM with certain kinds of states:
  - Start & Stop
  - Match (unbalanced emission probs)
  - Insert (balanced emission probs)
  - Delete (no emission probs)
  - Most transition probs = 0
  - Many paths are START → MATCH*n → STOP
  - pHMM algorithms need to be adjusted to handle these special states

Adjusting the algorithms

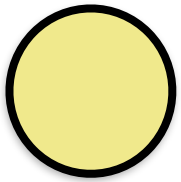


- Match states are normal

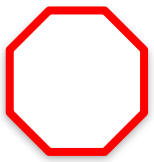
- Insert states are normal

- Start state???

- Delete states???



- Stop state???



Adjusting for start state

- Without a start state, we have a distribution of initial probabilities
  - Ex: P(start in A) = .5, P(start in B) = .4, P(start in C) = .1
- To introduce a start state
  - Initial probability distribution becomes p(start in ☆) = 1
  - Original initial probs become transition probs from
  - P(☆ → A) = .5, P(☆ → B) = .4, P(☆ → C) = .1

Adjusting for delete states

- Don’t think of indels as emissions. They aren’t *things*.
- pHMM can’t emit indels.
- Viterbi, FA, and BA algorithms need straightforward adjustments which we won’t go into.

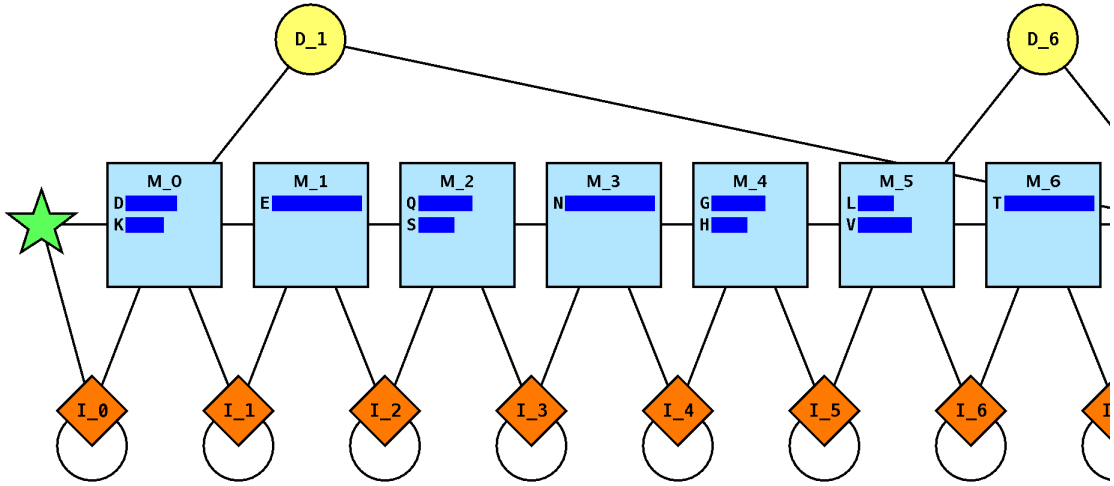
Adjusting for Stop state: look at final column

	☼	☼	⚡	❄
HAPPY	.025	.13 From HAPPY	.0046 From HAPPY	1.6E-4 From HAPPY
ANGRY	.0166	.0031 From HAPPY	.023 From HAPPY	8.0E-4 From ANGRY
DRUNK	.0333	.002 From DRUNK	6.6E-4 From HAPPY	.0023 From ANGRY

Before you compute max or sum of scores…

HAPPY	1.6E-4 From HAPPY	* P(HAPPY →)
ANGRY	8.0E-4 From ANGRY	* P(ANGRY →)
DRUNK	.0023 From ANGRY	* P(DRUNK→)

This concludes our study of Hidden Markov Models



TO DO!!!!

Check math on pseudoprobs

Add “M2” etc labels on match states.

Delete deets on adjustments to algorithms for pHMM