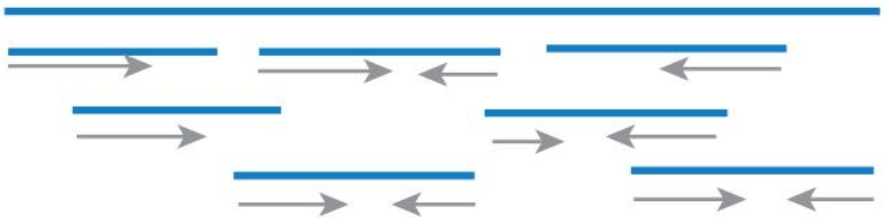
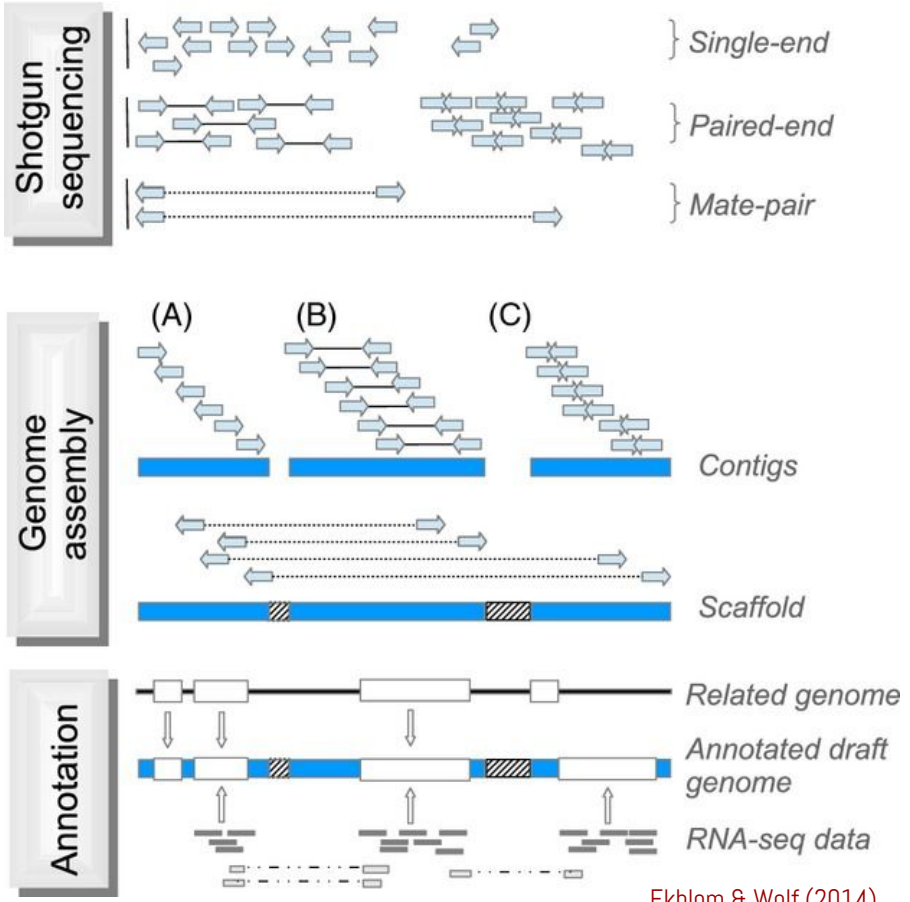
Lecture 3: Genome assembly & annotation

- Genome sequencing
  - de Bruijn graphs
- Genome annotation
  - Hidden Markov Models

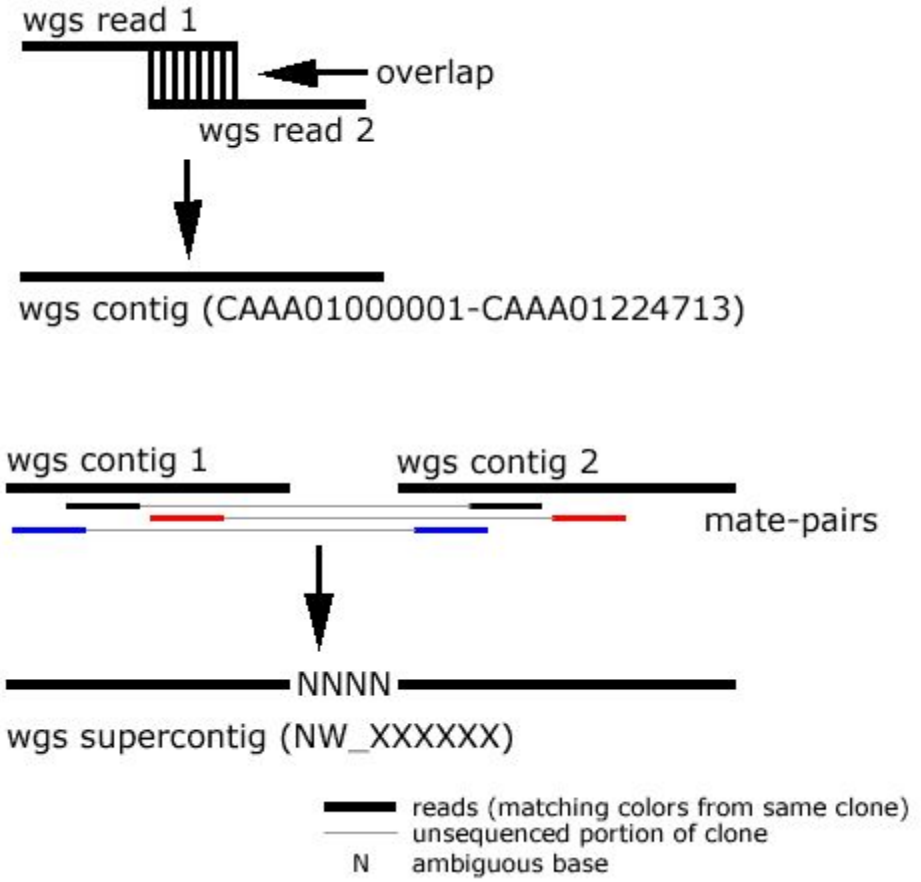
Genome assembly & annotation – Overview



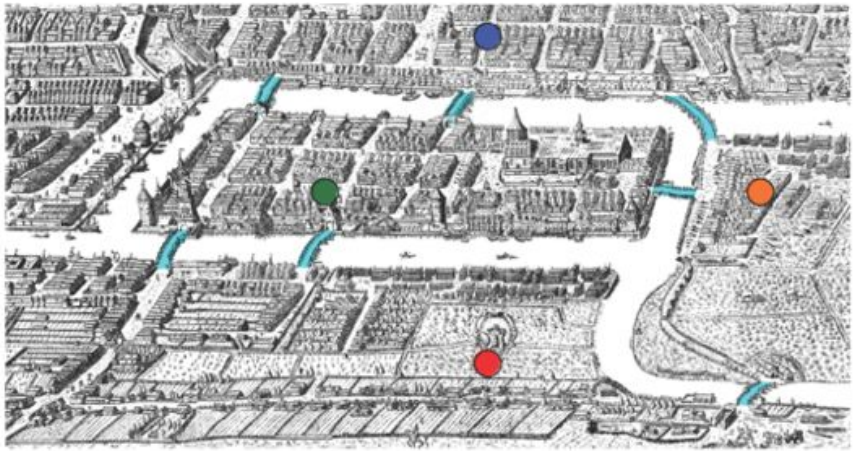


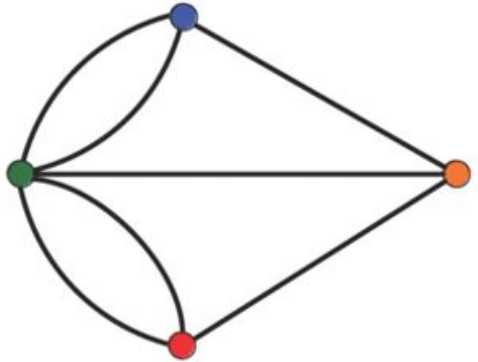
Ekblom & Wolf (2014)

Genome sequencing



Introduction to graph theory





Bridges of Königsberg

Can every part of the city be visited by walking across each of the seven bridges exactly once and returning to one's starting location?

Graph: (Nodes, Edges, Weights)

(Eulerian) Path exists if the graph contains zero or two vertices that have an odd degree.

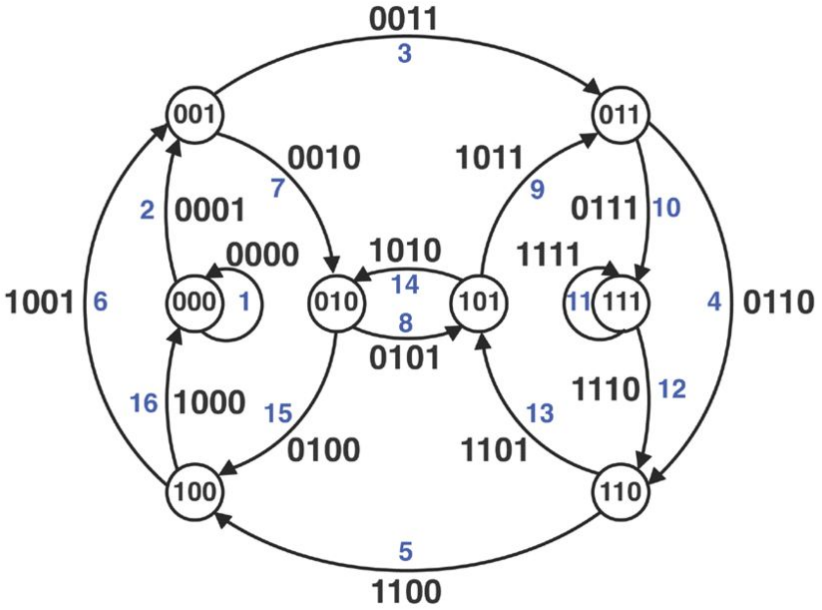
de Bruijn graphs: the 'superstring problem'

Find a shortest circular 'superstring' that contains all possible 'substrings' of length k (k-mers) over a given alphabet.

There exist n^k k-mers in an alphabet containing n symbols:
- alphabet: 0 & 1
- all 3-mers: 000, 001, 010, 011, 100, 101, 110, 111.
- The circular superstring **0001110100** contains all 3-mers & each 3-mer exactly once.

How can we construct a superstring for all k-mers in the case of an arbitrary value of k and an arbitrary alphabet?

Construct a **directed graph**: all prefixes & suffixes as nodes; all k-mers as edges.

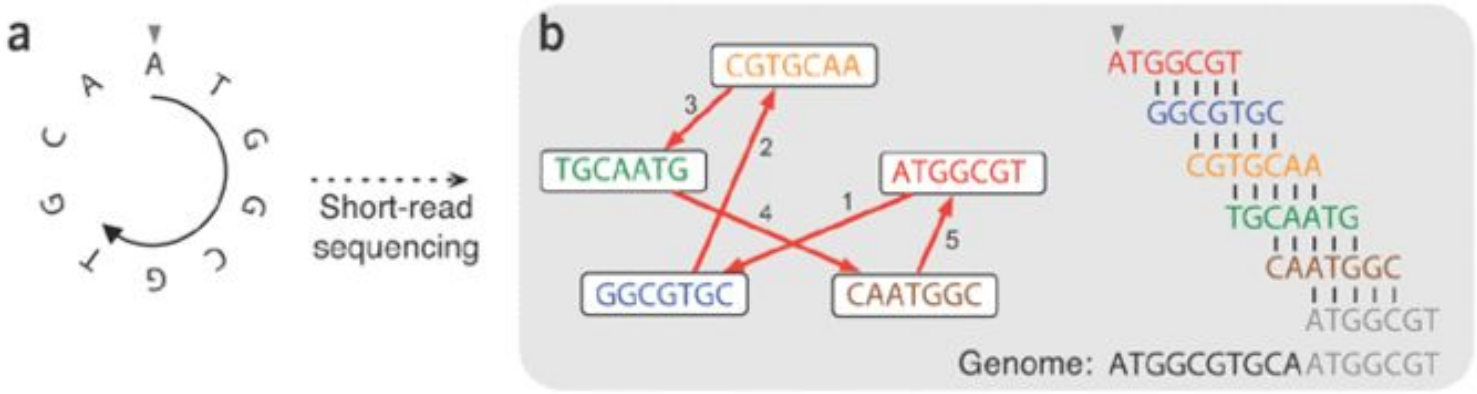


k = 4 | Two-character alphabet: digits 0 & 1

Does this graph have an Eulerian cycle? [Balanced?]

Following the blue numbered edges in order from 1 to 16 traces the cyclic superstring 0000110010111101.

de Bruijn graphs for sequence assembly

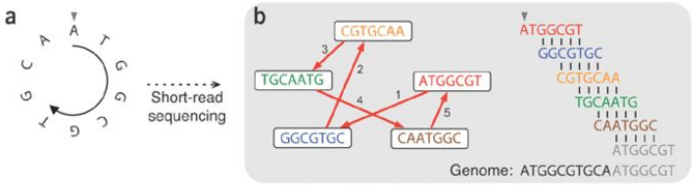


Nodes: Reads

Edges: Alignments between reads (≥ 5 bases)

Genome: Walk along a Hamiltonian cycle (visit each vertex exactly once) to combine alignments between successive reads and reconstruct the full circular genome.

de Bruijn graphs for sequence assembly



Four hidden assumptions that **do not hold** for real sequencing:

1. We can generate all k-mers present in the genome.
2. All k-mers are error free.
3. Each k-mer appears at most once in the genome.
4. The genome consists of a single circular chromosome.

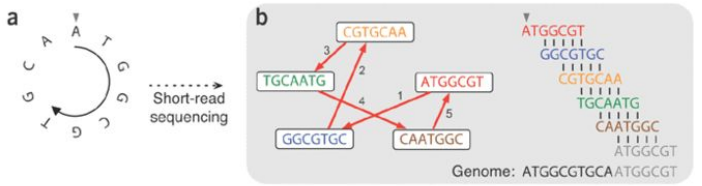
E.g., a technology that generates 100-nucleotide long reads:

- may miss some 100-mers present in the genome (even if the read coverage is high)
- the 100-mers that it does generate typically have errors.

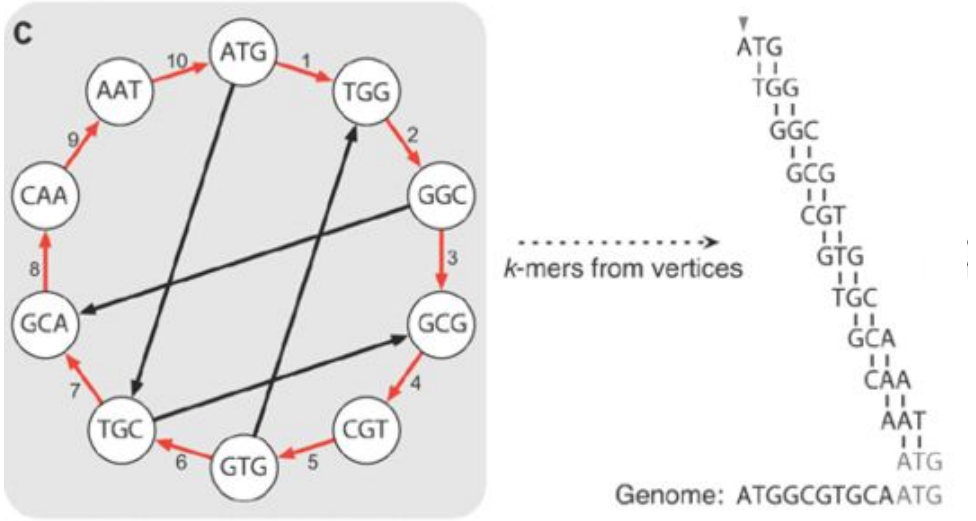
de Bruijn graphs for sequence assembly

Break reads into shorter k-mers!

Resulting k-mers often represent nearly all k-mers from the genome for sufficiently small k.



Nodes: k-mers
Edges: Overlap between k-mers
Genome: Walk the Hamiltonian cycle



Large genomes result in too many reads:

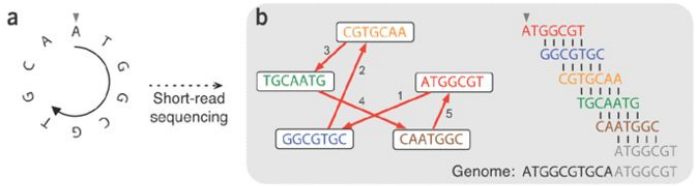
- 10^6 reads $\longrightarrow$ 10^{12} pairwise alignments.
- 10^9 reads $\longrightarrow$ 10^{18} alignments.

There is no known efficient algorithm for finding a Hamiltonian cycle in a large graph with millions (let alone billions) of nodes.

Compeau et al. (2011)

de Bruijn graphs for sequence assembly

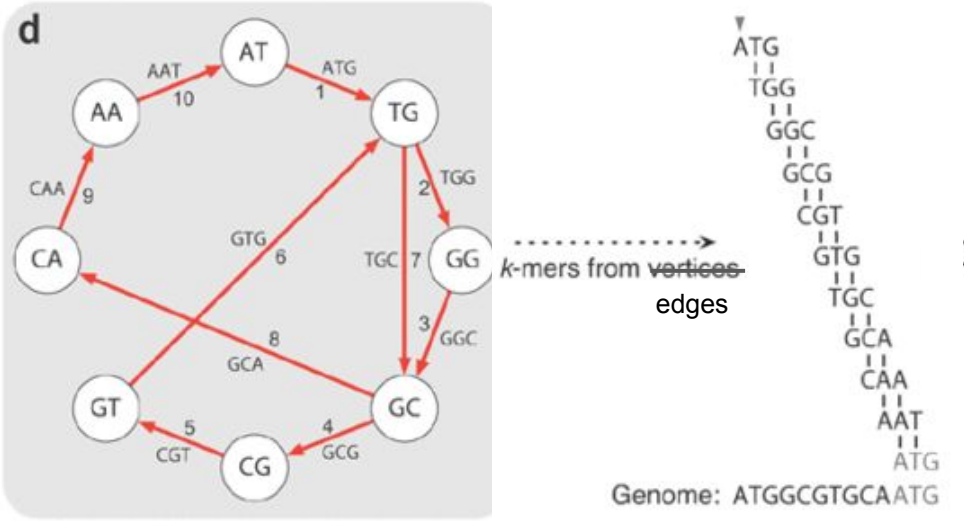
Finding a path that visits all *edges* of a graph exactly once is much easier!



Nodes: (k-1)-mers [in- & out-degrees?]
Edges: k-mers
Genome: Walking the Eulerian cycle

Send out an ant, find a cycle; Not edges traversed, send out another ant, … until all of the graph's edges have been explored. Combine all cycles to form an Eulerian cycle!

Computationally tractable; time roughly proportional to the number of edges.

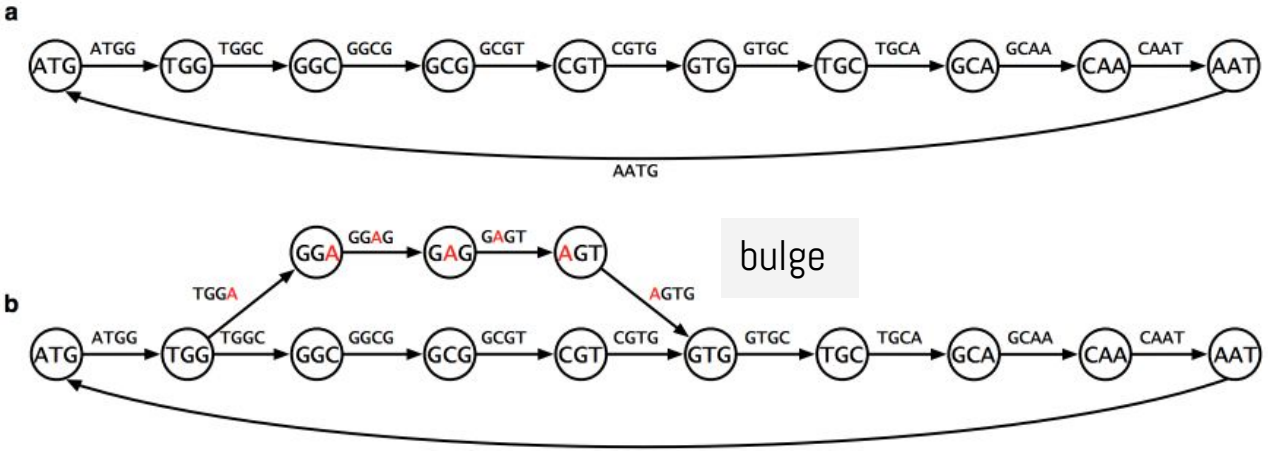


Compeau et al. (2011)

de Bruijn graphs for sequence assembly – not so easy w/ real data

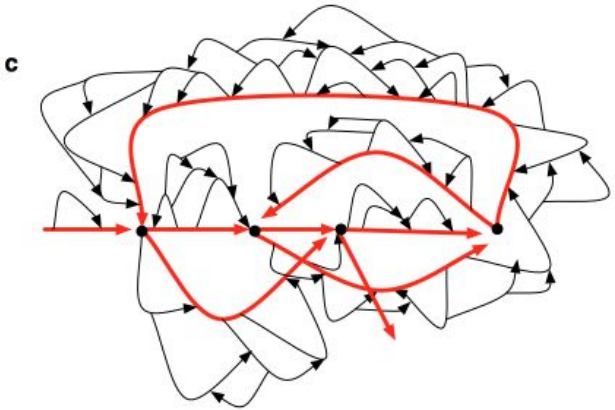
- Not all k-mers in the genome
  - Read breaking
- Errors in reads
  - Error correcting reads
  - Removing bulges in de Bruijn graphs
- DNA repeats
  - Incorporating k-mer multiplicity
- Multiple and linear chromosomes
  - Cycles to paths
- Unsequenced regions
  - Scaffolds

de Bruijn graphs from reads with sequencing errors



TGG*A*GTG (incorrect)

TGG*C*GTG (correct)



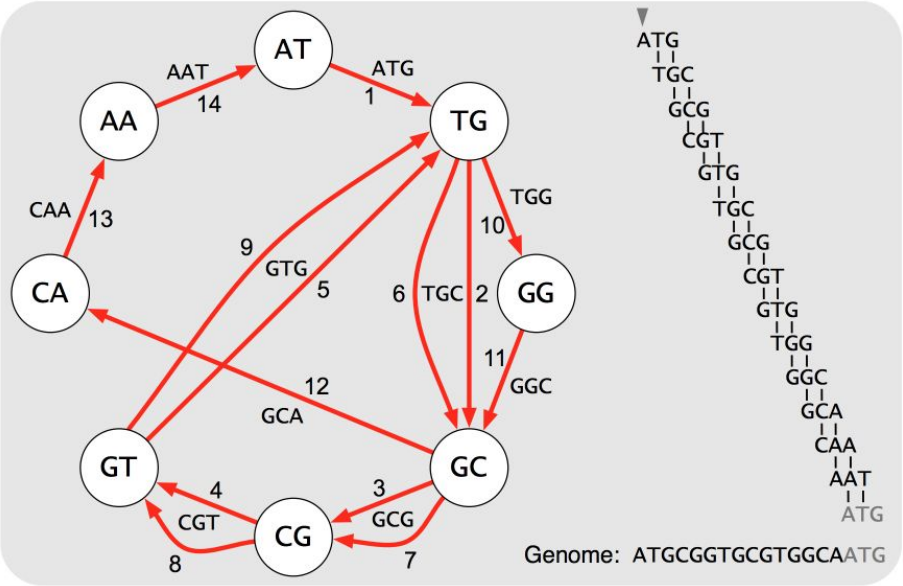
The process of bulge removal should leave only the red edges remaining, yielding an Eulerian path in the resulting graph.

Compeau et al. (2011)

de Bruijn graphs for sequence assembly – not so easy w/ real data

- Not all k-mers in the genome
  - Read breaking
- Errors in reads
  - Error correcting reads
  - Removing bulges in de Bruijn graphs
- DNA repeats [E.g., **ATGCATGC** ⟶ four 3-mers: **ATG**, **TGC**, **GCA** & **CAT**]
  - Incorporating k-mer multiplicity
- Multiple and linear chromosomes
  - Cycles to paths
- Unsequenced regions
  - Scaffolds

de Bruijn graphs for dealing with repeats



Genome: **ATGCGTGCGTGGCA**

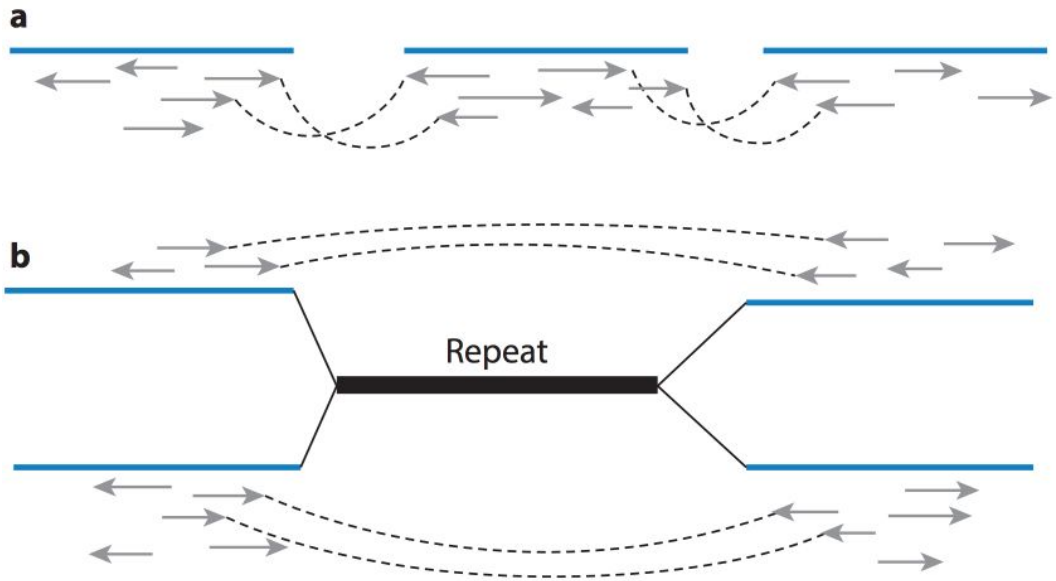
Incorporate k-mer multiplicity:

- Four 3-mers **TGC**, **GCG**, **CGT**, and **GTG**: multiplicity 2
- Six 3-mers **ATG**, **TGG**, **GGC**, **GCA**, **CAA**, and **AAT**: multiplicity 1

Compeau et al. (2011)

de Bruijn graphs for sequence assembly – not so easy w/ real data

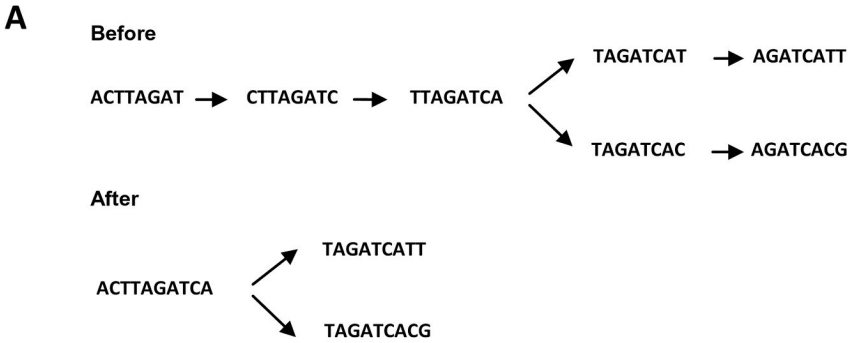
- Not all k-mers in the genome
  - Read breaking
- Errors in reads
  - Error correcting reads
  - Removing bulges in de Bruijn graphs
- DNA repeats
  - Incorporating k-mer multiplicity
- Multiple and linear chromosomes
  - Cycles to paths
- Unsequenced regions
  - Scaffolds

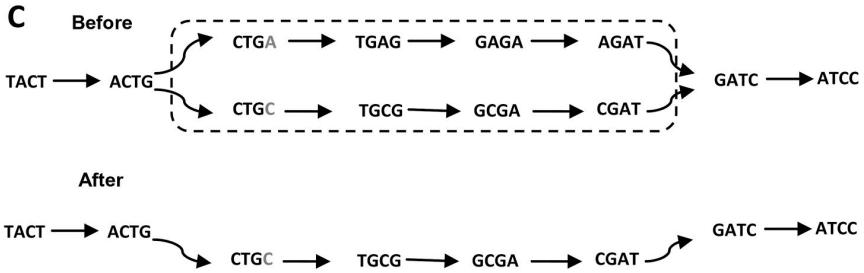


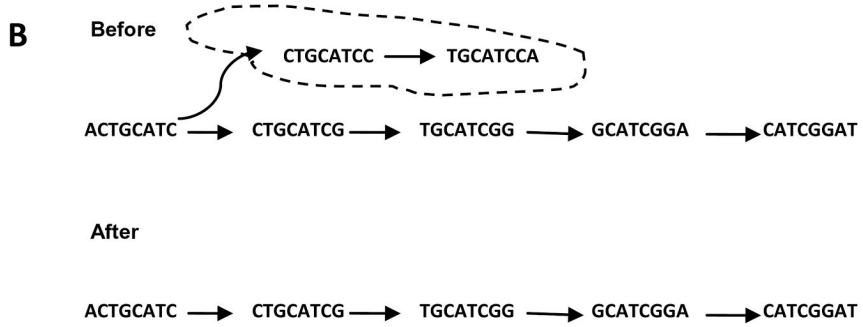
Algorithms for genome assembly

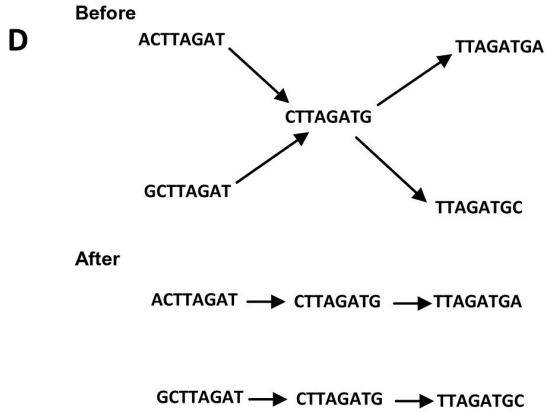
- Error detection and correction based on sequence composition of the reads.
- Graph construction to represent reads/k-mers and their shared sequence.
- Graph adjustments:
    - Reduction of simple non-intersecting paths to single nodes.
    - Removal of error-induced paths (recognized as spurs or bubbles).
    - Collapse of polymorphism-induced complexity (bubbles).
    - Simplification of tangles (using information outside the graph: individual, paired-end, or mate-pair reads to constraints on path distance & outcome.
- Conversion of reduced paths to contigs and scaffolds.
- Reduction of alignments to a consensus sequence.

Simplifying de Bruijn graphs



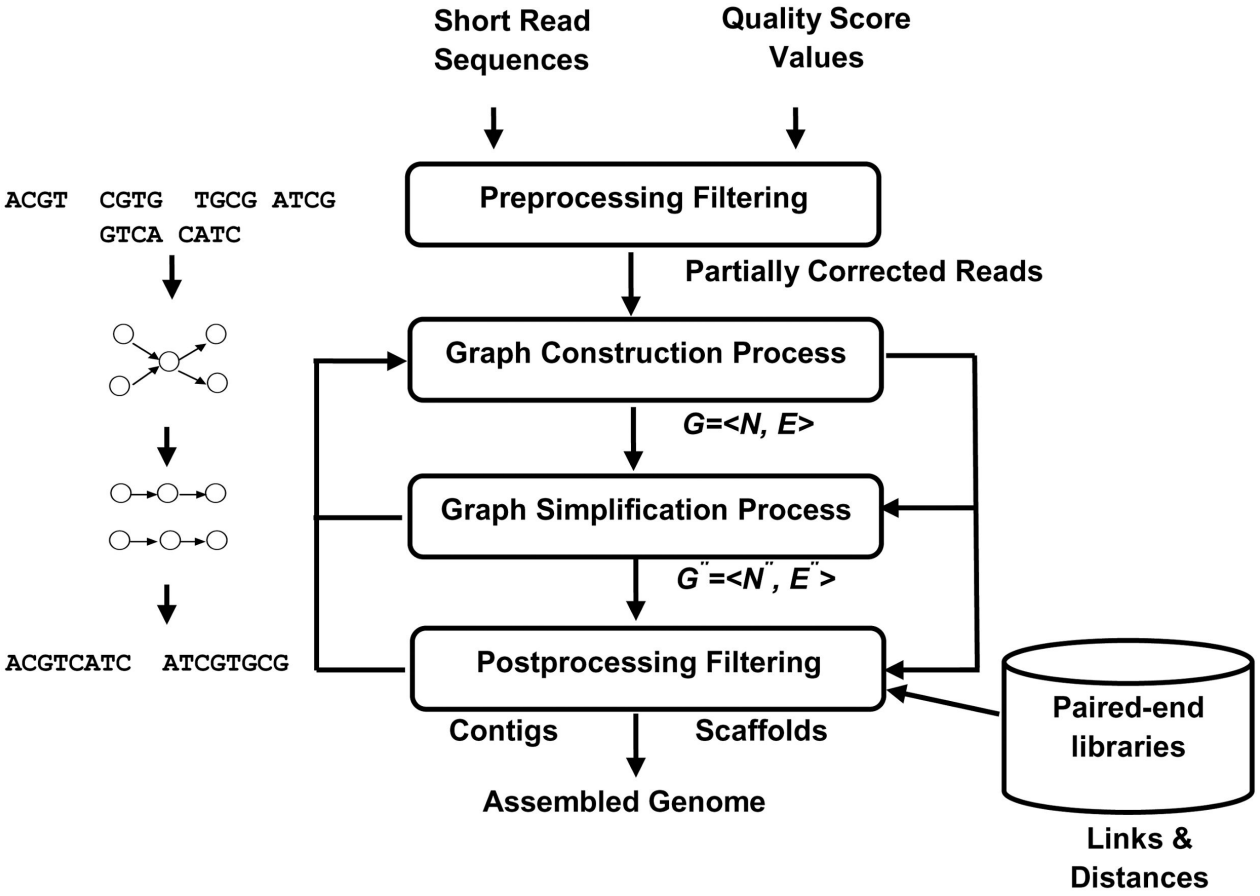






El-Metwally et al. (2013)

Algorithms for genome assembly



El-Metwally et al. (2013)

Algorithms for genome assembly

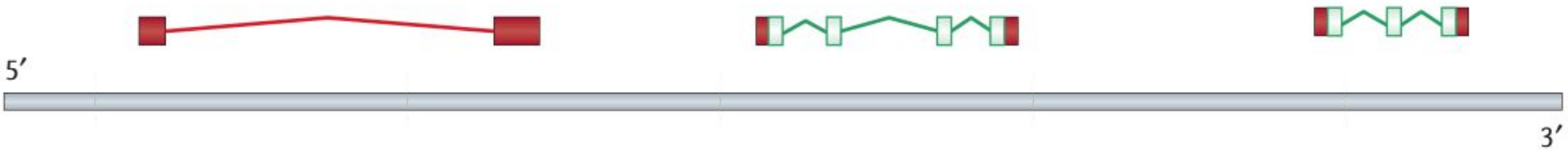
- New sequencing technologies → a different best computational strategy.
- Factors that influence the choice of algorithms:
  - Quantity of data (read length and coverage)
  - Quality of data (including error rates)
  - Genome structure (e.g., GC content and the number and size of repeated regions).
- de Bruijn graphs are best suited for current short-read sequencing technologies
  - Produce very large numbers of reads
  - Can represent genomes with repeats [Overlap methods need to mask repeats > read length]
- Long-read technology growing at a rapid pace.

Some modern genome assemblers

Assemblers	Technology	Availability	Notes
Genome assemblers			
ALLPATHS-LG	Illumina, Pacific Biosciences	ftp://ftp.broadinstitute.org/pub/crd/ALLPATHS/Release-LG	Requires a specific sequencing recipe (BOX 3)
SOAPdenovo	Illumina	http://soap.genomics.org.cn/soapdenovo.html	Also used for transcriptome and metagenome assembly
Velvet	Illumina, SOLiD, 454, Sanger	http://www.ebi.ac.uk/~zerbino/velvet	May have substantial memory requirements for large genomes
ABySS	Illumina, SOLiD, 454, Sanger	http://www.bcgsc.ca/platform/bioinfo/software/abyss	Also used for transcriptome assembly

Nagarajan & Pop (2013)

Genome annotation – Gene finding

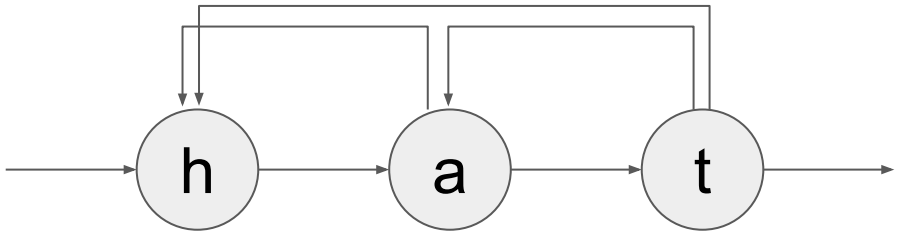


Problem: Given the entire genome, label the nucleotides as exons, introns, or intergenic sequence.

Requirements:

- Combine splice-site consensus, codon bias, exon/ intron length preferences and open reading frame analysis into one scoring system.
- Provide results that can be interpreted probabilistically. (How confident are we that the best scoring answer is correct?)
- Should be extensible, capable of modeling additional genomic features like translational initiation consensus, alternative splicing, and a polyadenylation signal.

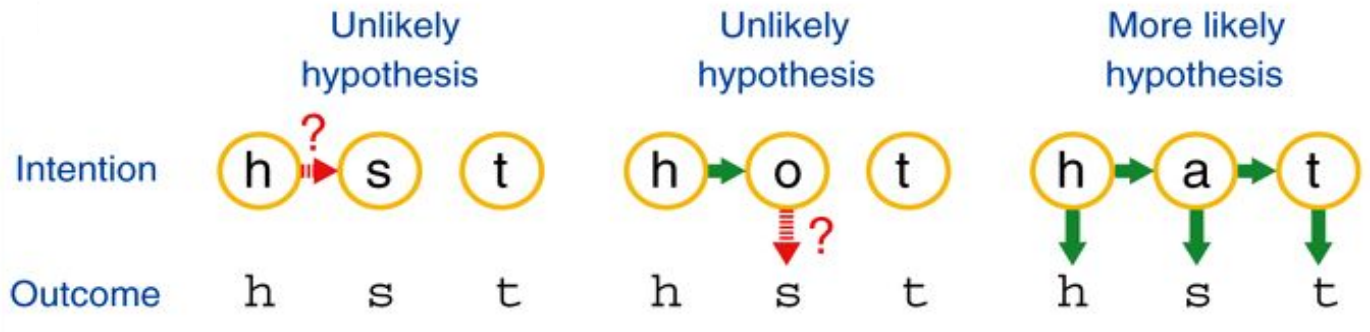
Markov models



Current state depends only on previous state and transition probability.

- $\Pr(\text{‘at’}) = \Pr(\text{‘a’}) \cdot \Pr(\text{‘t’} \mid \text{‘a’})$
- $\Pr(x_1 \ldots x_n) = \Pr(x_1) \prod \Pr(x_i \mid x_{i-1})$

Hidden Markov Models (HMMs)

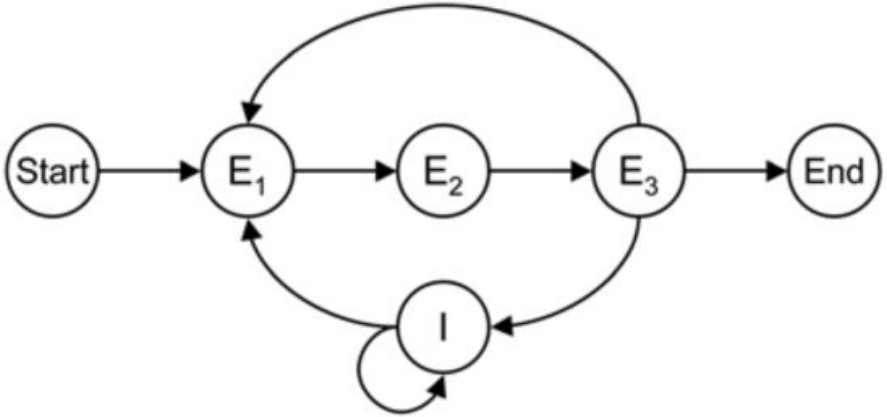


Transition probabilities: model letter sequences in correctly spelled words

Emission probabilities: model the probability of each possible typographical error.

Brent (2007)

A simple HMM for modeling eukaryotic genes



E_1 : 1st position of codon
E_2 : 2nd position of codon
E_3 : 3rd position of codon
exon
I : intron

x	A	T	G	C	G	A	C	T	G	C	A	T	A	G	C	A	C	T	T
y	E_1	E_2	E_3	E_1	E_2	E_3	E_1	E_2	E_3	I	I	I	I	E_1	E_2	E_3	E_1	E_2	E_3

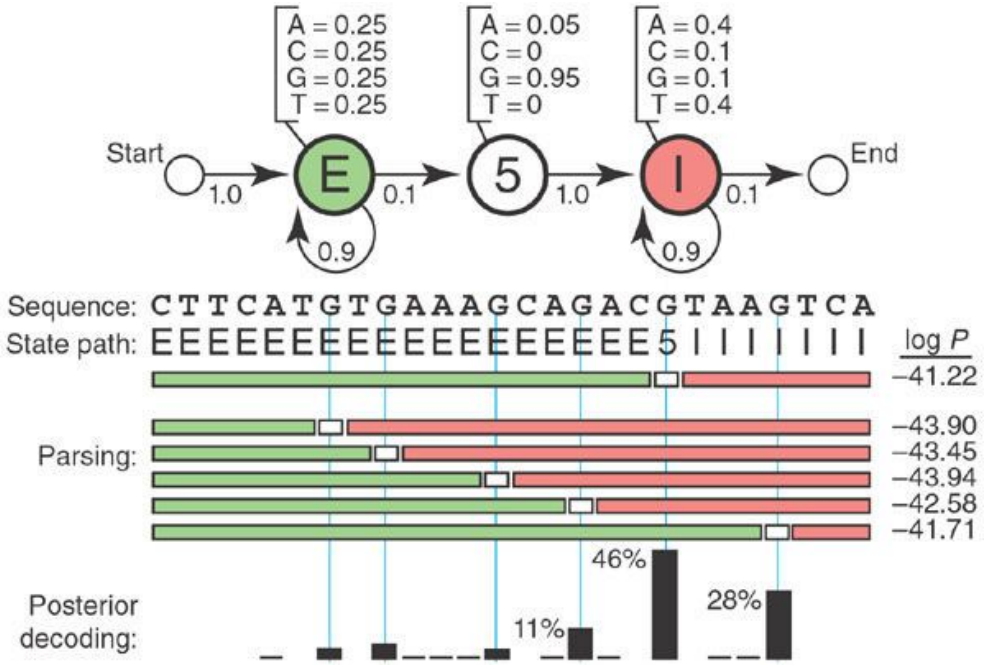
observed symbols
hidden states

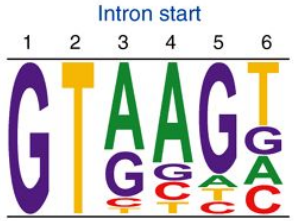
exon | intron | exon

Yoon (2009)

A simple HMM for modeling eukaryotic genes

A toy HMM for 5′ splice site recognition





Sequence logo representing the weight matrix for the first six bases of an intron.

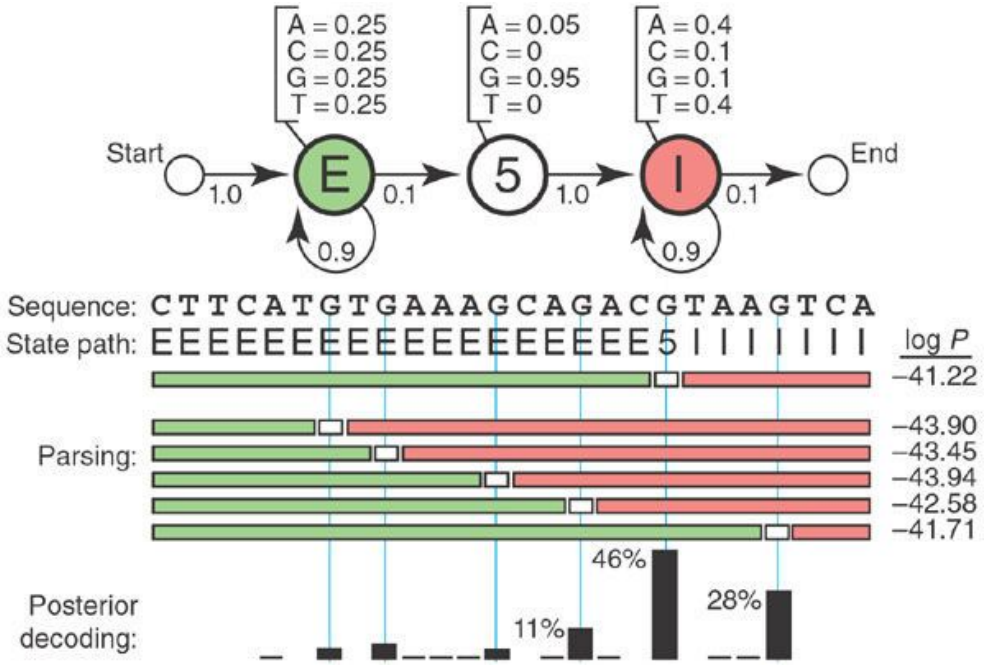
Designing the HMM:

1. Alphabet with K symbols.
2. No. of states in the model M.
3. Emission probabilities $e_i(x)$ for each state i, that sum to one over the K symbols x, $\sum_x e_i(x) = 1$.
4. Transition probabilities for each state i going to any other state j (including itself) that sum to one over the M states j, $\sum_j t_i(j) = 1$.

Eddy (2004)

A simple HMM for modeling eukaryotic genes

A toy HMM for 5′ splice site recognition



S: observed sequence

π: state path, a Markov chain that's hidden.

θ: parameters of the model

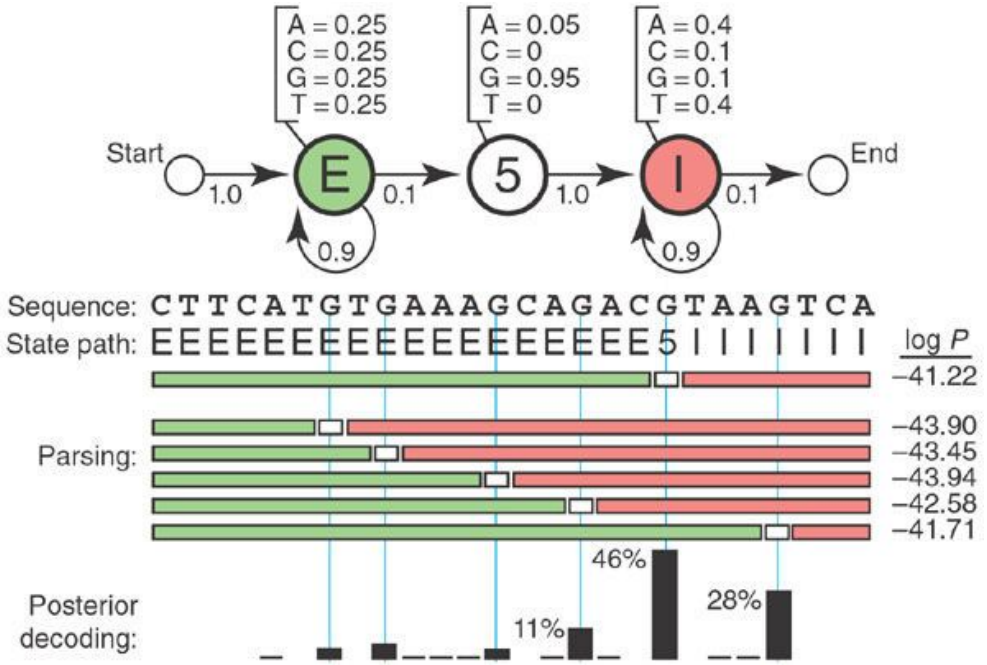
Pr(S, π | HMM, θ): product of all emission & transition probabilities. Here, 26 emissions & 27 transitions.

- How many possible paths?
- Which path to pick? The Viterbi algorithm is guaranteed to find the most probable path given seq & HMM.
- How confident are we that the 5th G is the right choice?

Eddy (2004)

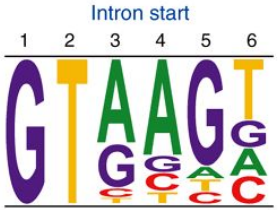
A simple HMM for modeling eukaryotic genes

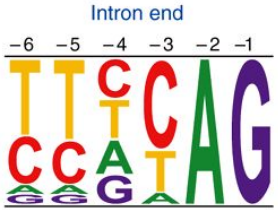
A toy HMM for 5′ splice site recognition



Adding more details:

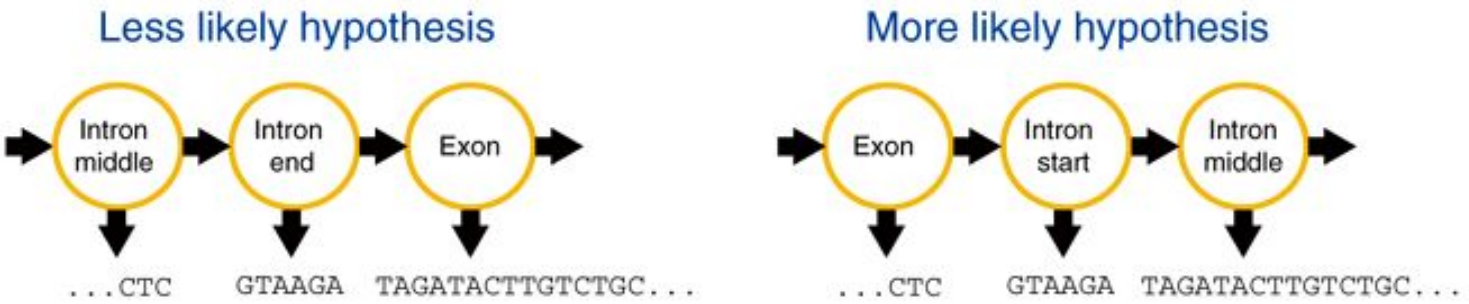
- Six-nucleotide consensus GTRAGT at the 5′ splice site.
- Similarly for the 3′ splice site.
- Add a 3′ exon state.





Eddy (2004)

Hidden Markov Models



Generalized hidden Markov models (GHMMs):

- States: variable-length segments of the DNA sequence sharing some common function in transcription, RNA processing or translation.

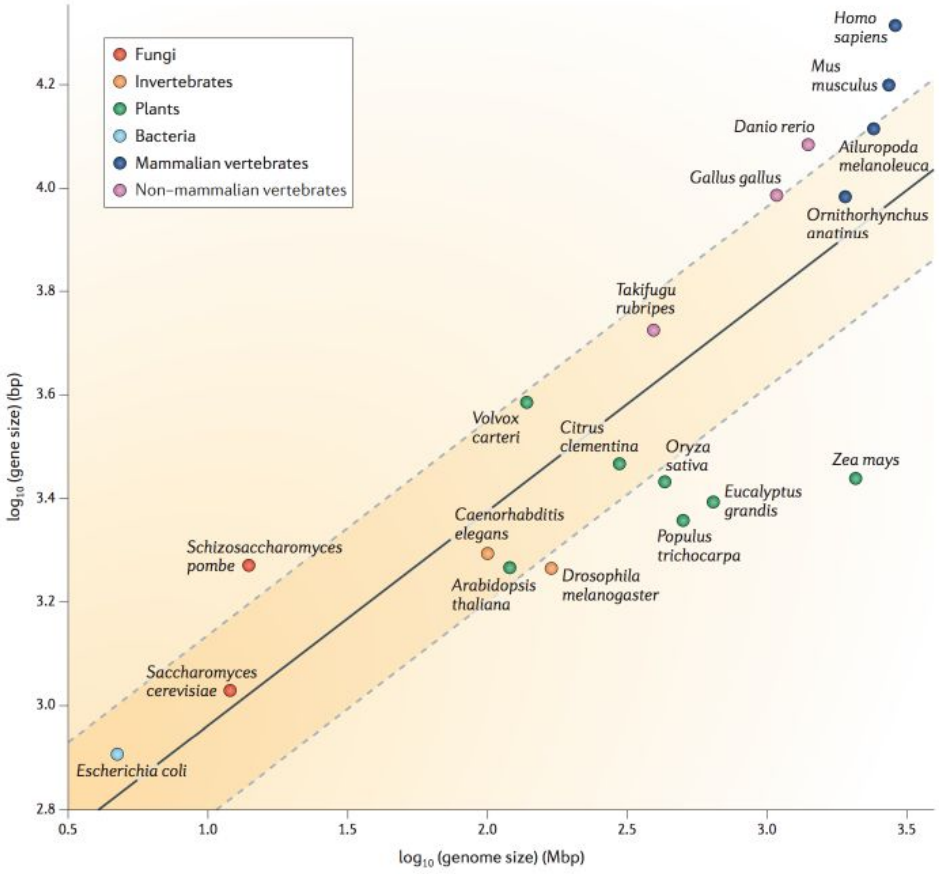
Brent (2007)

Some modern ab-initio gene prediction tools

Software	Description
Ab initio *and evidence-drivable gene predictors*	
Augustus	Accepts expressed sequence tag (EST)-based and protein-based evidence hints. Highly accurate
mGene	Support vector machine (SVM)-based discriminative gene predictor. Directly predicts 5′ and 3′ untranslated regions (UTRs) and poly(A) sites
SNAP	Accepts EST and protein-based evidence hints. Easily trained
FGENESH	Training files are constructed by SoftBerry and supplied to users
Geneid	First published in 1992 and revised in 2000. Accepts external hints from EST and protein-based evidence
Genemark	A self-training gene finder
Twinscan	Extension of the popular Genscan algorithm that can use homology between two genomes to guide gene prediction
GAZE	Highly configurable gene predictor
GenomeScan	Extension of the popular Genscan algorithm that can use BLASTX searches to guide gene prediction
Conrad	Discriminative gene predictor that uses conditional random fields (CRFs)
Contrast	Discriminative gene predictor that uses both SVMs and CRFs
CRAIG	Discriminative gene predictor that uses CRFs
Gnomon	Hidden Markov model (HMM) tool based on Genscan that uses EST and protein alignments to guide gene prediction
GeneSeqer	A tool for identifying potential exon–intron structure in precursor mRNAs (pre-mRNAs) by splice site prediction and spliced alignment

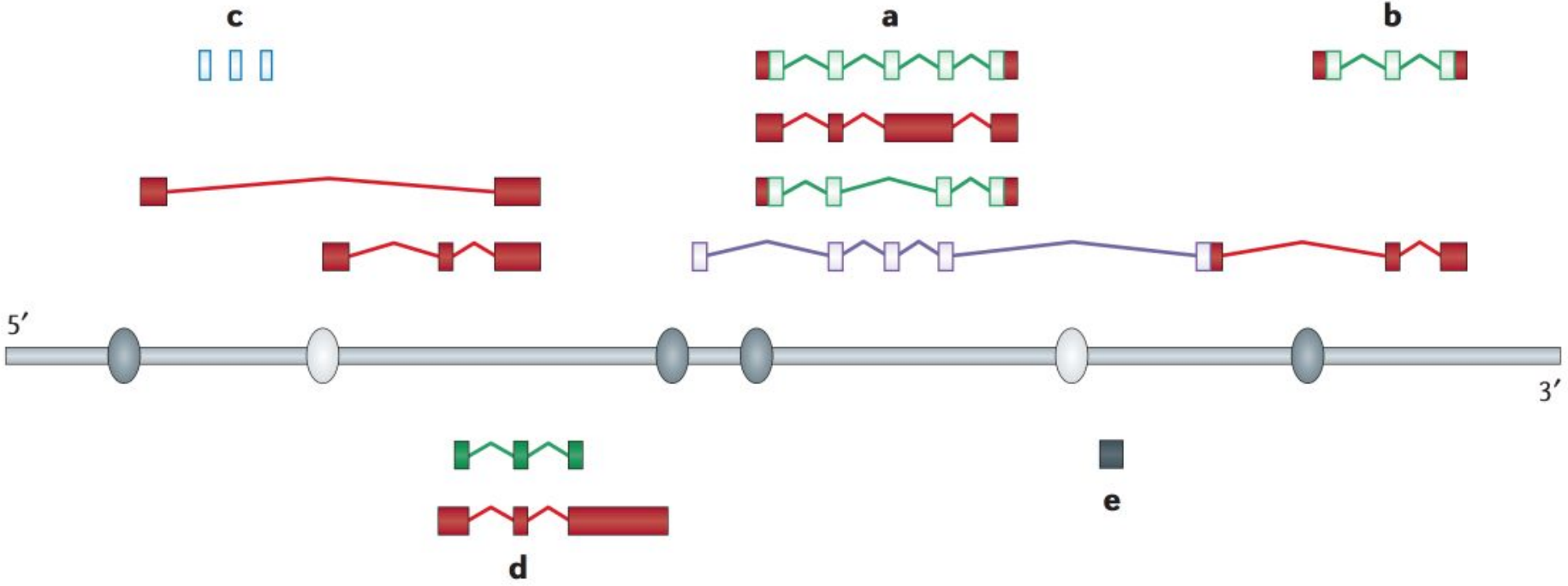
Yandell & Ence (2012)

Genomic complexity



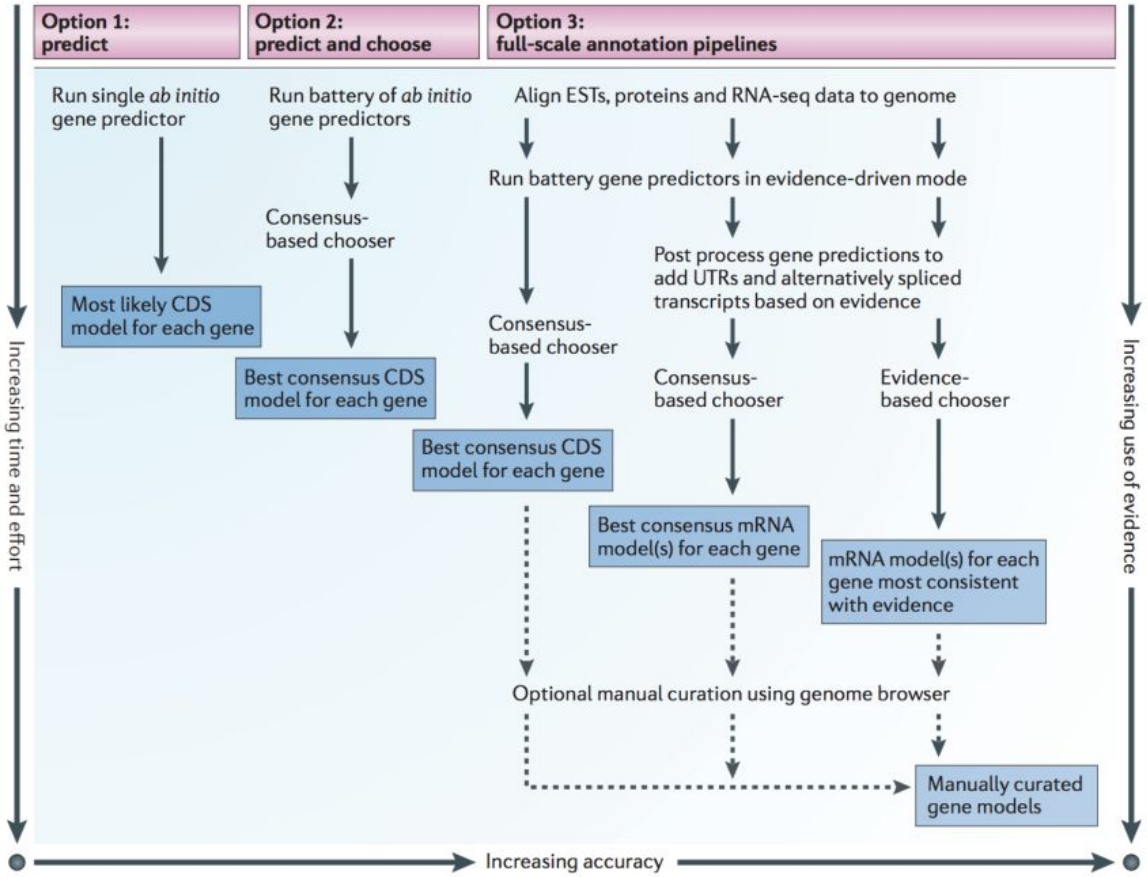
Yandell & Ence (2012)

Genomic complexity



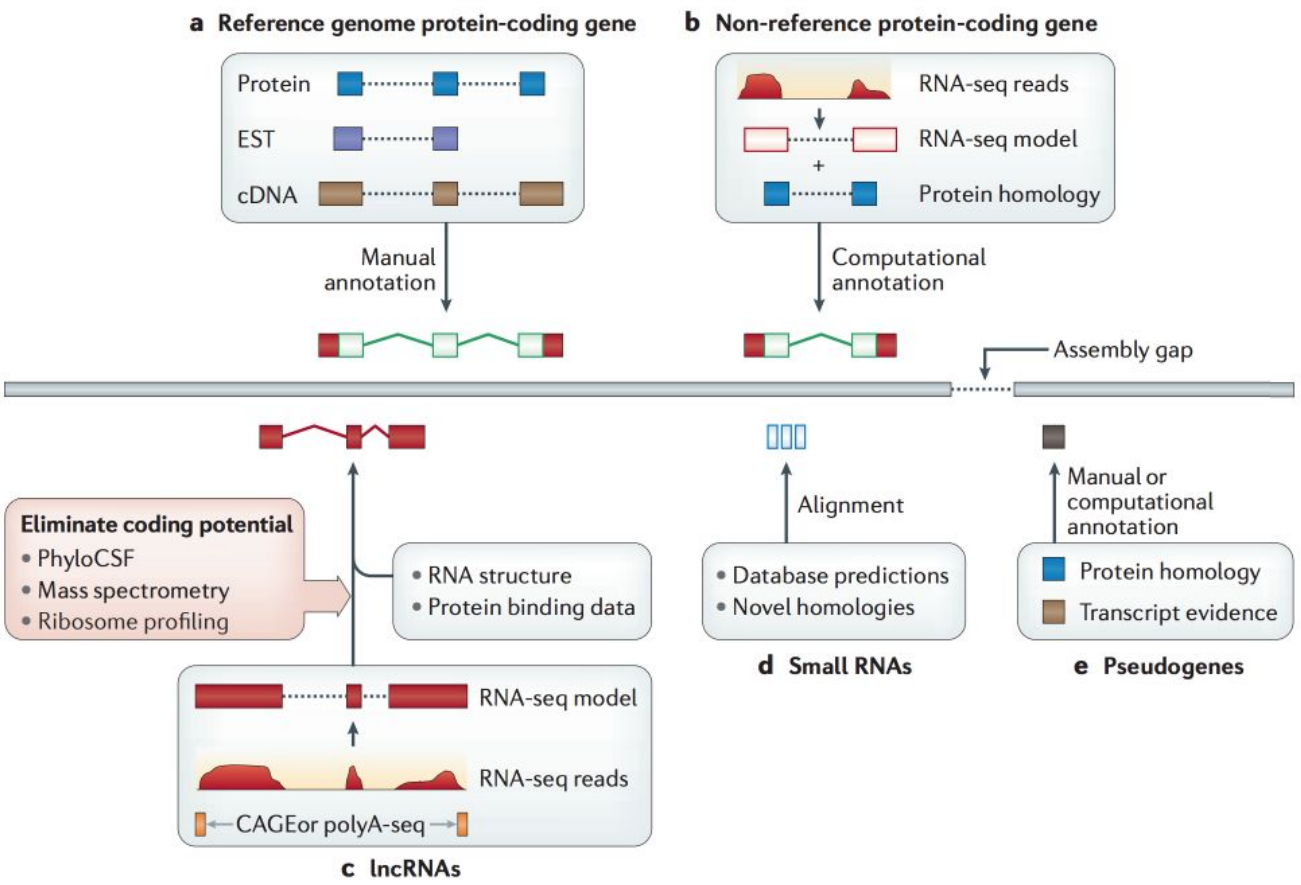
Mudge & Harrow (2016)

Approaches to genome annotation



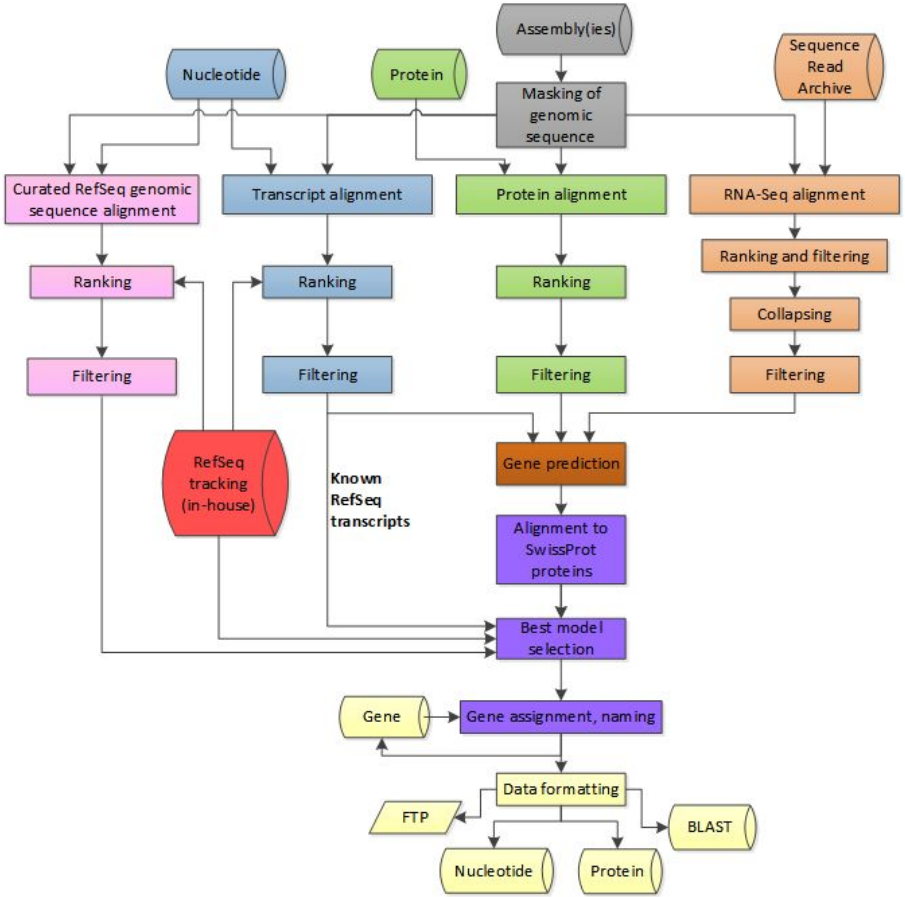
Yandell & Ence (2012)

Annotation workflows for different gene types



Mudge & Harrow (2016)

NCBI eukaryotic genome annotation pipeline



Genomic sequence preparation (grey)

Alignments of transcripts (blue)

Alignment of proteins (green)

Alignment of short reads (orange)

Alignment of curated genomic sequences (pink)

Gene prediction based on all avail. Alignments (brown)

Internal tracking database of RefSeq seqs (red)

Selection of best models & protein naming (purple)

Formatting of ann sets for public resources (yellow)