Gene Prediction: Similarity-Based Approaches

Outline

1. Introduction

2. Exon Chaining Problem

3. Spliced Alignment

4. Gene Prediction Tools

Section 1: Introduction

Similarity-Based Approach to Gene Prediction

- Some genomes may be well-studied, with many genes having been experimentally verified.
- Closely-related organisms may have similar genes.
- In order to determine the functions of unknown genes, researchers may compare them to known genes in a closely-related species.
- This is the idea behind the similarity-based approach to gene prediction.

Similarity-Based Approach to Gene Prediction

- There is one issue with comparison: genes are often "split."
  - The **exons**, or coding sections of a gene, are separated from **introns**, or the non-coding sections.
  - Example:

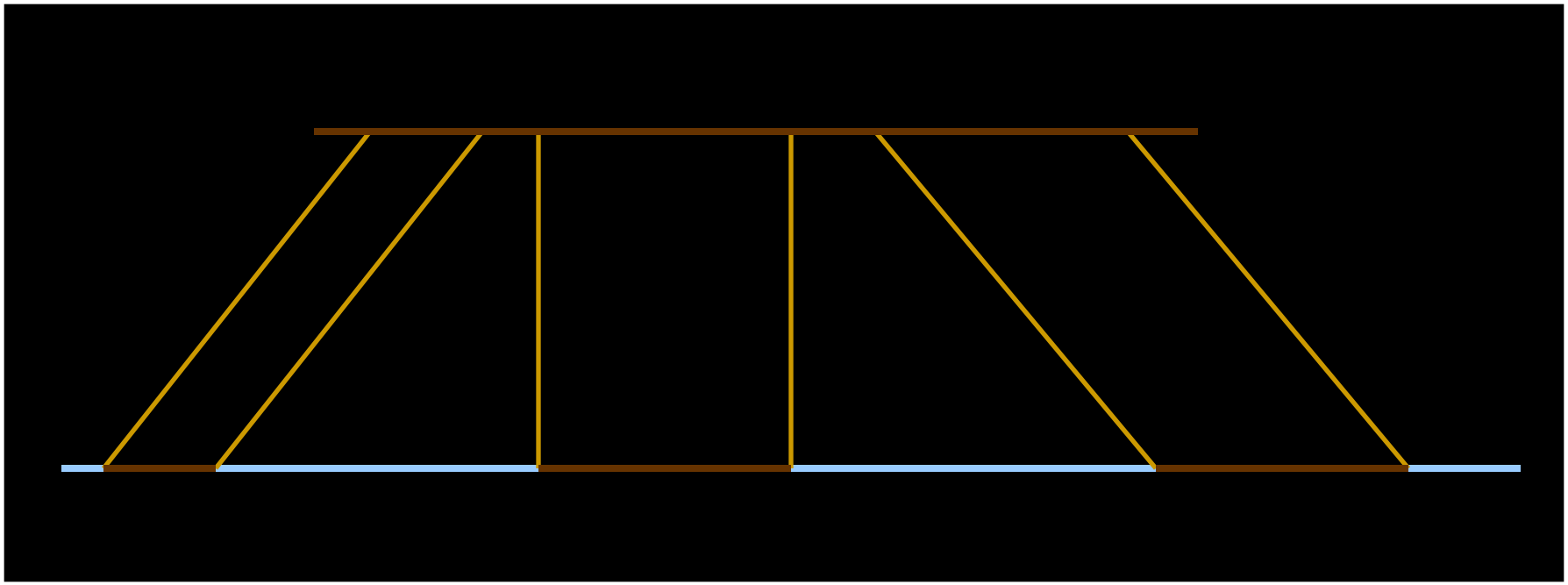
...AGGGTCTCATTGTAGACAGTGGTACTGATCAACGCAGGACTT...

Coding Non-coding Coding Non-coding

- Our Problem**:** Given a known gene and an unannotated genome sequence, find a set of substrings in the genomic sequence whose concatenation best matches the known gene.
  - This concatenation will be our candidate gene.

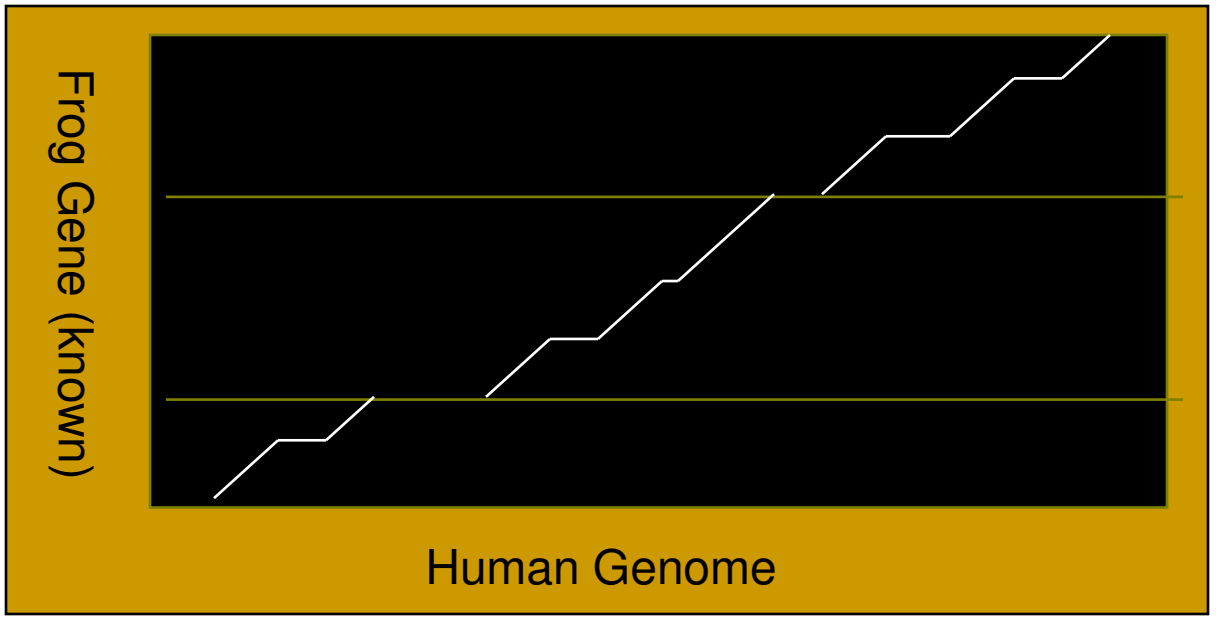
Comparing Genes in Two Genomes

- Small islands of similarity corresponding to similarities between exons



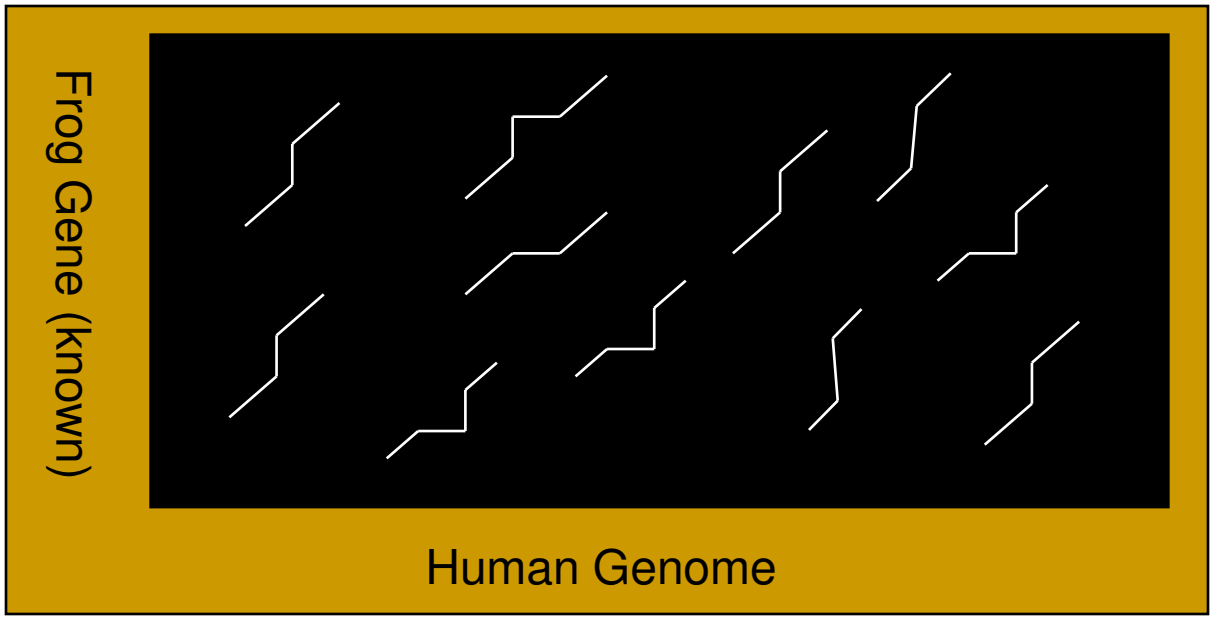
Using Similarities to Find Exon Structure

- A (known) frog gene is aligned to different locations in the human genome.
- Find the "best" path to reveal the exon structure of the corresponding human gene.



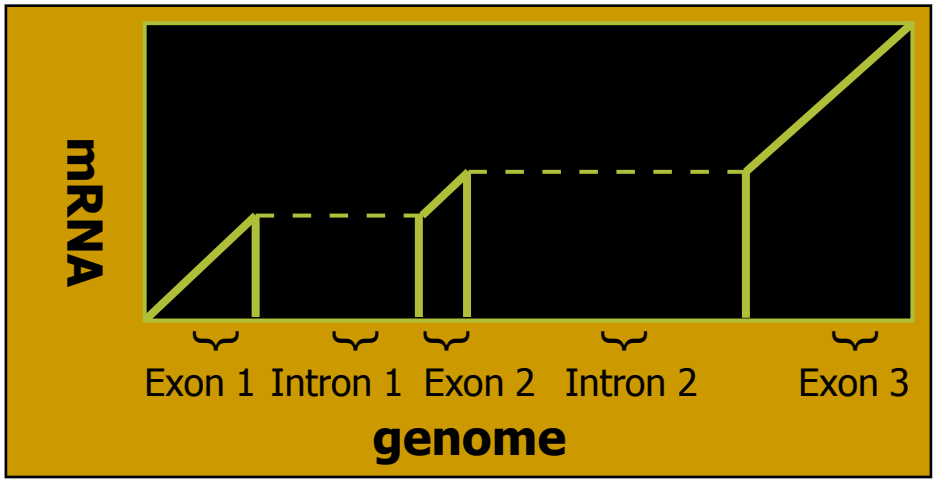
Finding Local Alignments

- A (known) frog gene is aligned to different locations in the human genome.
- Find the "best" path to reveal the exon structure of the corresponding human gene.
- Use local alignments to find all islands of similarity.

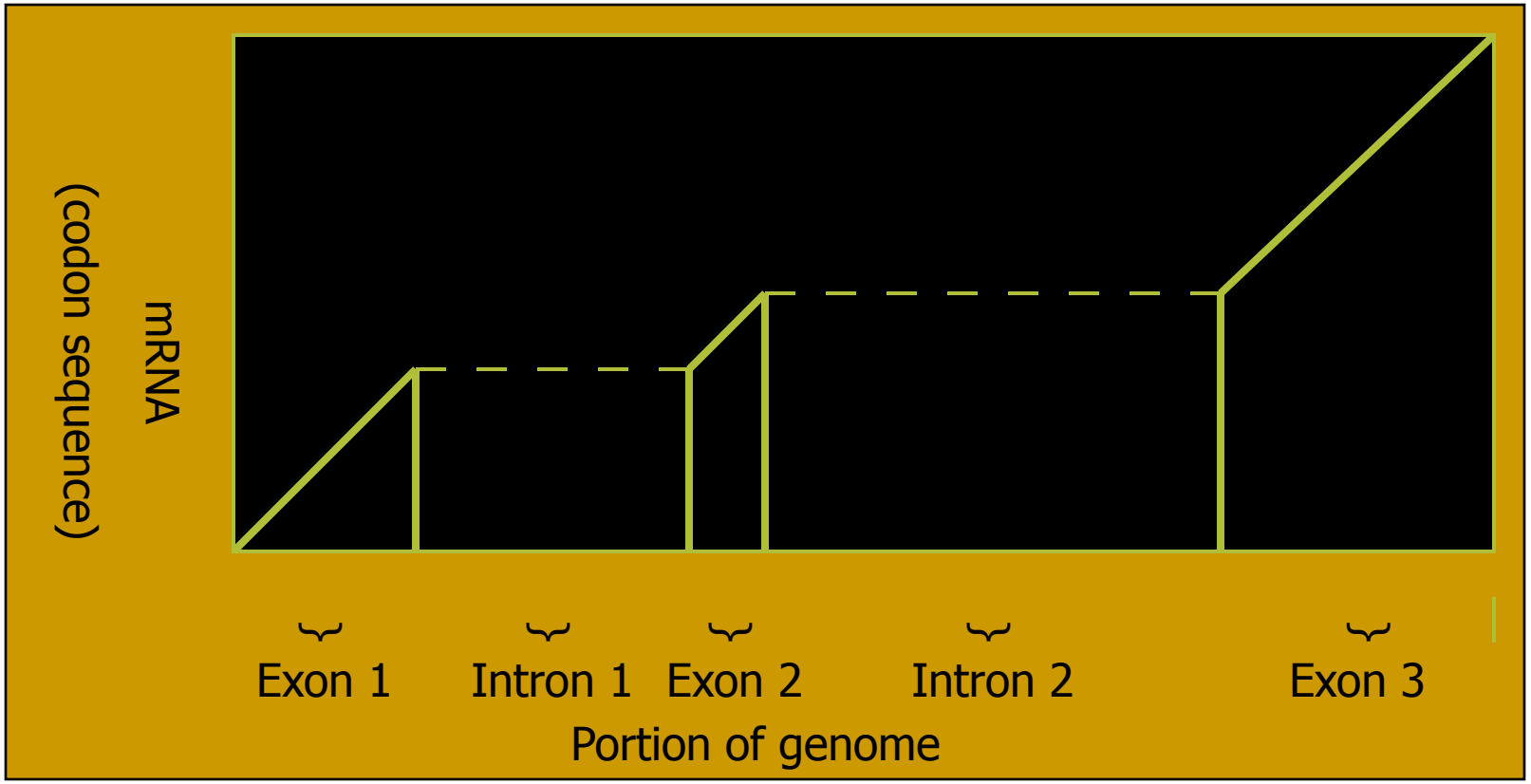


Reverse Translation

- **Reverse Translation Problem**: Given a known protein, find a gene which codes for it.
  - Inexact: amino acids map to > 1 codon
  - This problem is essentially reduced to an alignment problem.
  - **Example**: Comparing Genomic DNA Against mRNA.



Comparing Genomic DNA Against mRNA



Reverse Translation

- The reverse translation problem can be modeled as traveling in Manhattan grid with "free" horizontal jumps.
  - Each horizontal jump models insertion of an intron.
  - Complexity of Manhattan grid is $O(n^3)$.
  - Issue: Would match nucleotides pointwise and use horizontal jumps at every opportunity.

Section 2: Exon Chaining Problem

Chaining Local Alignments

- Aim: Find **candidate exons**, or substrings that match a given gene sequence.

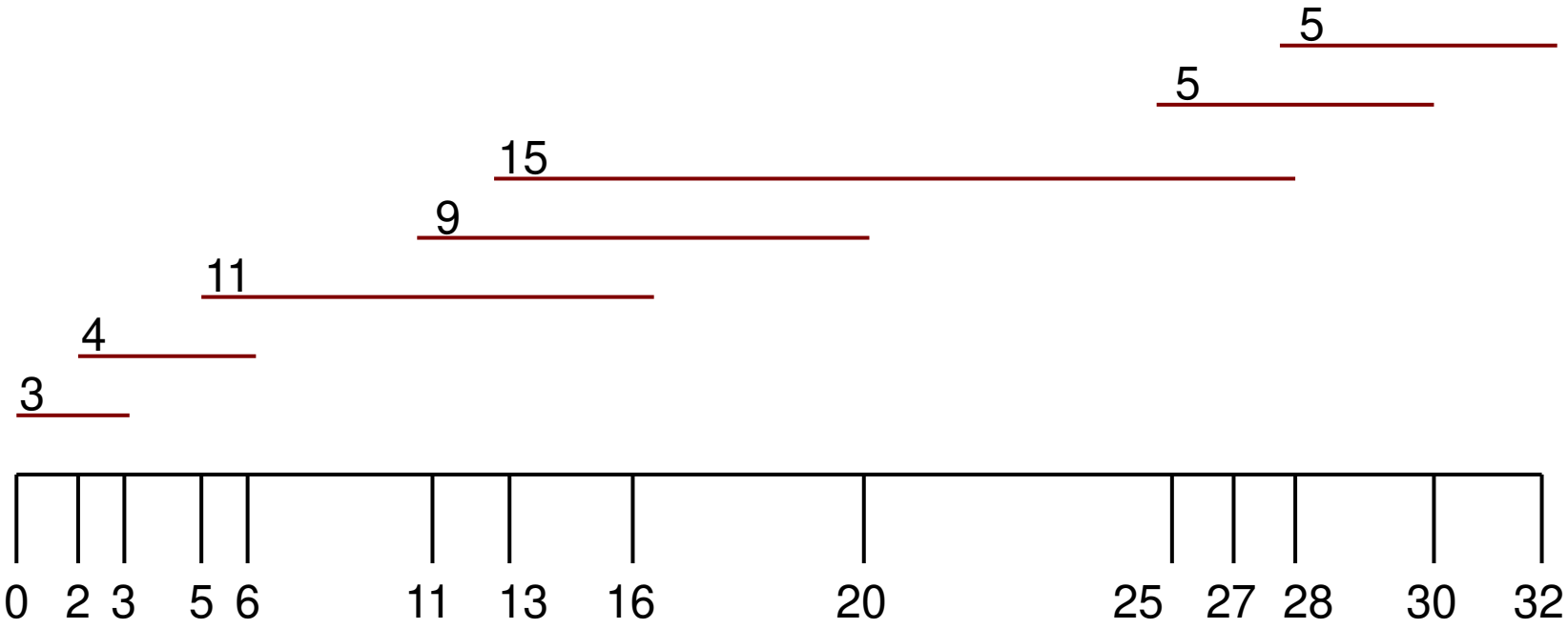
- Define a candidate exon as (l, r, w):
  - l = starting position of exon
  - r = ending position of exon
  - w = weight of exon, defined as score of local alignment or some other weighting score

- Idea: Look for a chain of substrings with maximal score.
  - **Chain**: a set of non-overlapping nonadjacent intervals.

Exon Chaining Problem: Illustration

- Locate the beginning and end of each interval ($2n$ points).

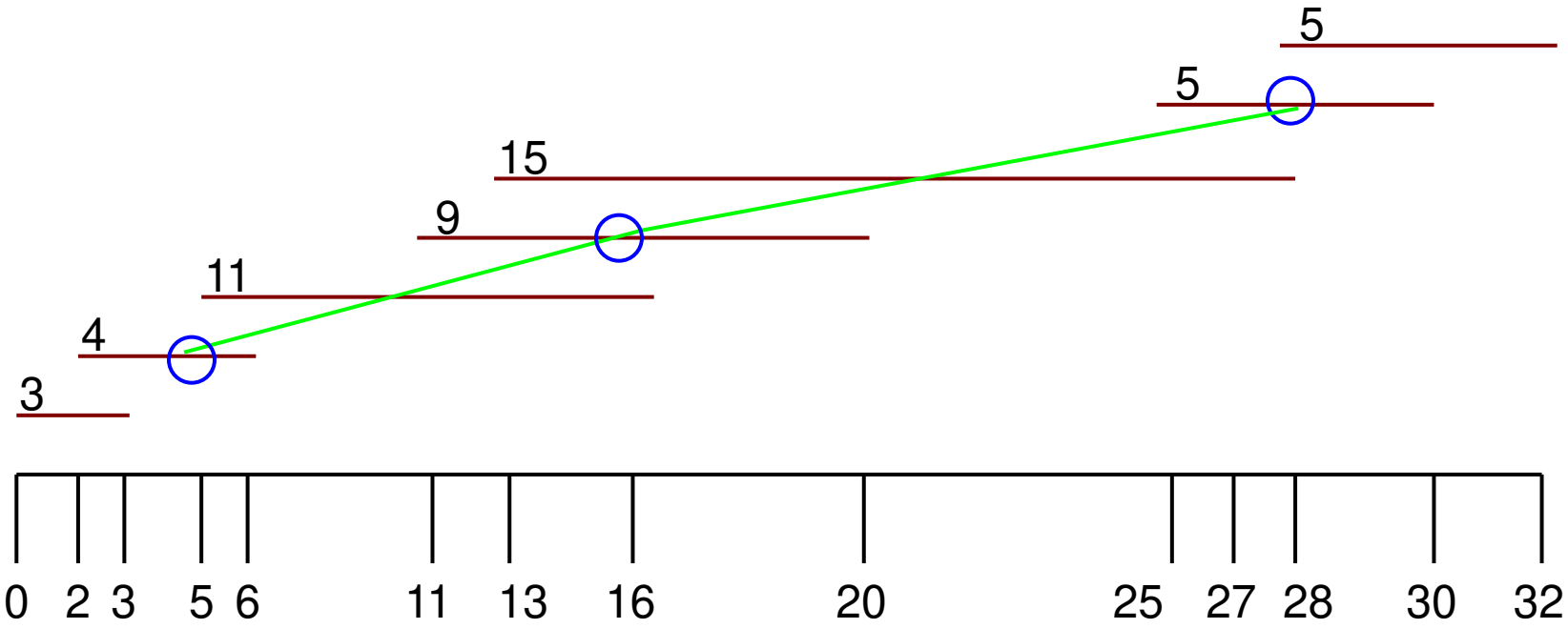
- Find the "best" concatenation of some of the intervals.



Exon Chaining Problem: Illustration

- Locate the beginning and end of each interval ($2n$ points).

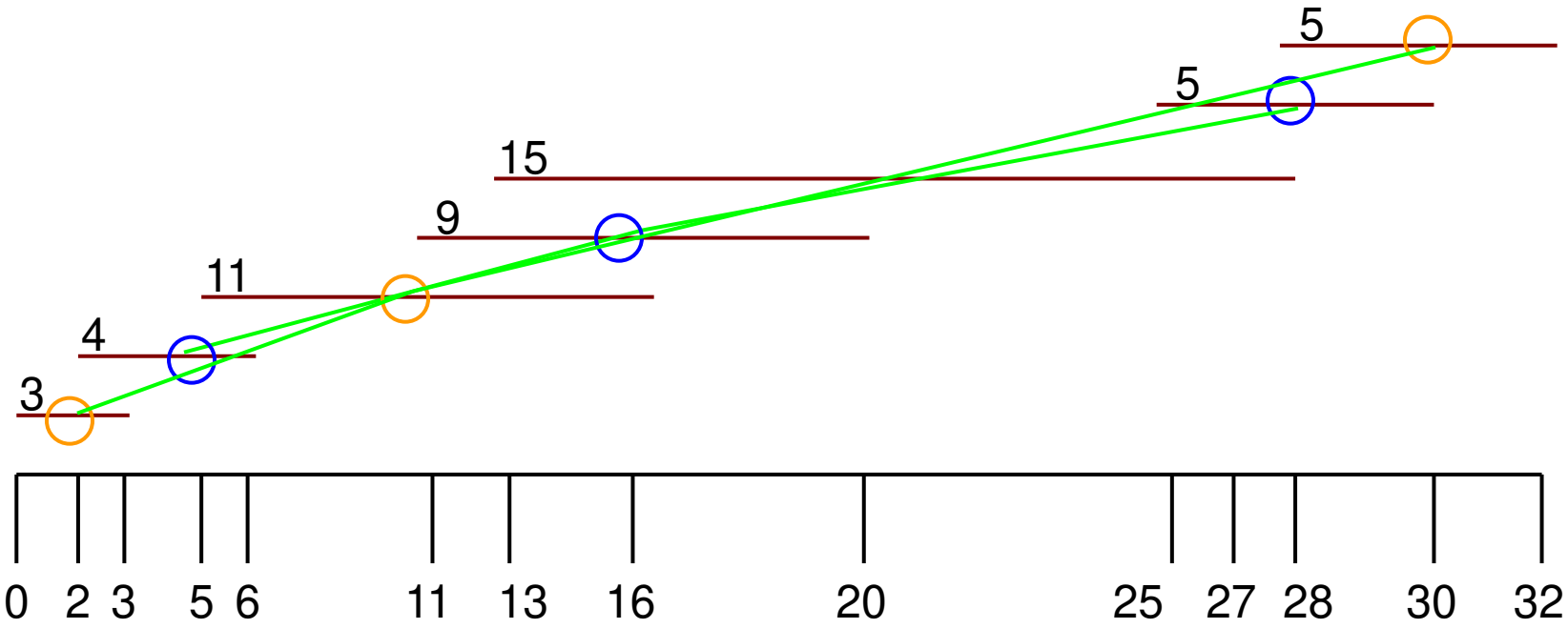
- Find the "best" concatenation of some of the intervals.



Exon Chaining Problem: Illustration

- Locate the beginning and end of each interval ($2n$ points).

- Find the "best" concatenation of some of the intervals.



Exon Chaining Problem: Formulation

- Goal: Given a set of putative exons, find a maximum set of non-overlapping putative exons (chain).

- Input: A set of weighted intervals (putative exons).

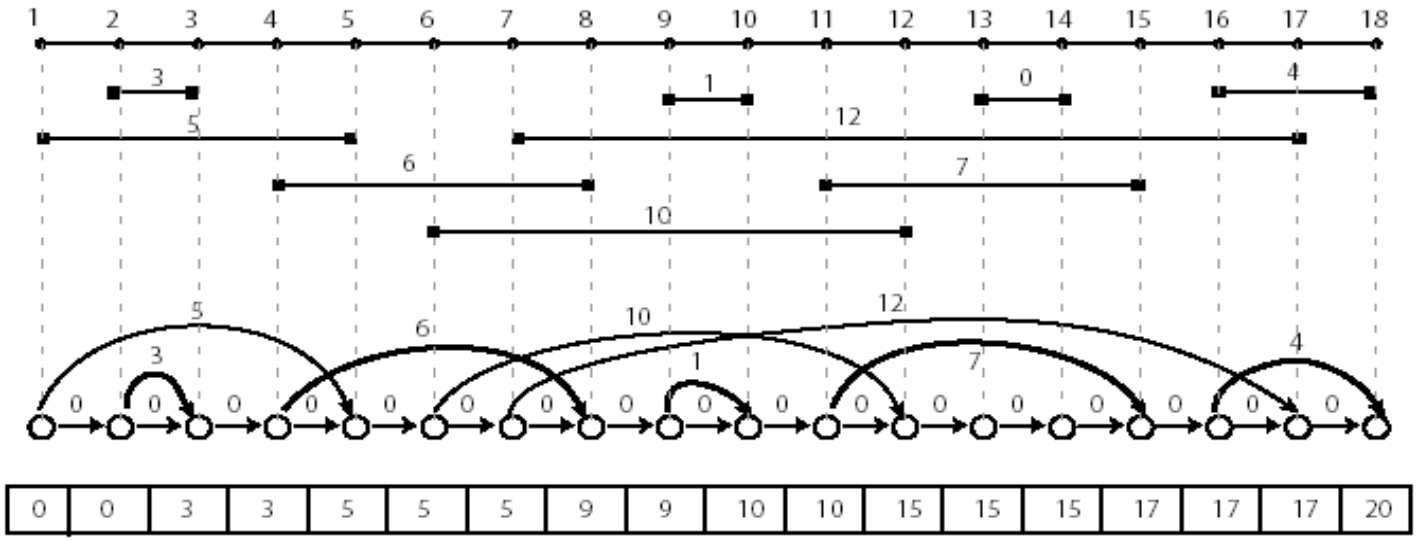
- Output: A maximum chain of intervals from this set.

Exon Chaining Problem: Formulation

- Goal: Given a set of putative exons, find a maximum set of non-overlapping putative exons (chain).
- Input: A set of weighted intervals (putative exons).
- Output: A maximum chain of intervals from this set.
- **Question**: Would a greedy algorithm solve this problem?

Exon Chaining Problem: Graph Representation

- This problem can be solved with dynamic programming in O(*n*) time.
  - Idea: Connect adjacent endpoints with weight zero edges, connect ends of weight *w* exon with weight *w* edge.



Exon Chaining Problem: Pseudocode

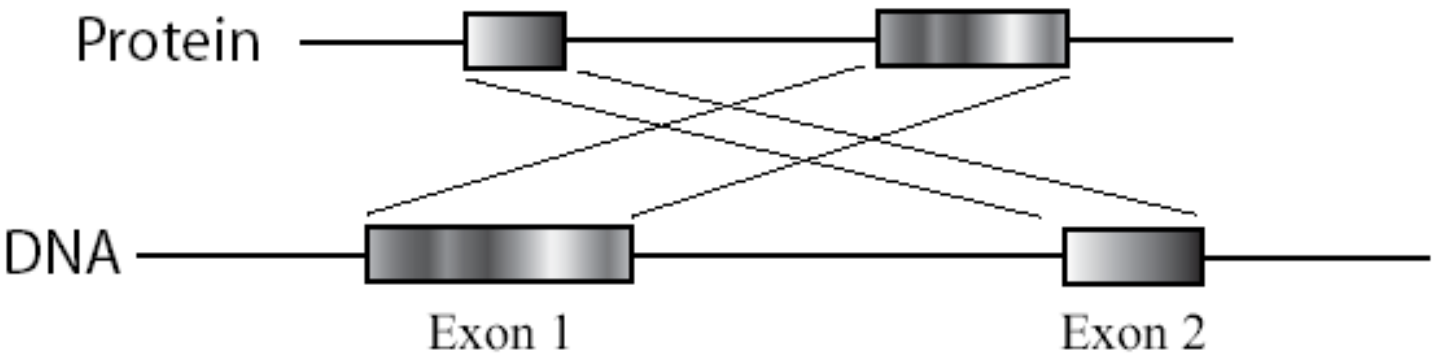
ExonChaining (G, n) //Graph, number of intervals

1. **for** $i \leftarrow 1$ to $2n$
2. $s_i \leftarrow 0$
3. **for** $i \leftarrow 1$ to $2n$
4. **if** vertex v_i in G corresponds to right end of the interval I
5. $j \leftarrow$ index of vertex for left end of the interval I
6. $w \leftarrow$ weight of the interval I
7. $s_j \leftarrow max\{s_j + w, s_{i-1}\}$
8. **else**
9. $s_i \leftarrow s_{i-1}$
10. **return** s_{2n}

Exon Chaining: Deficiencies

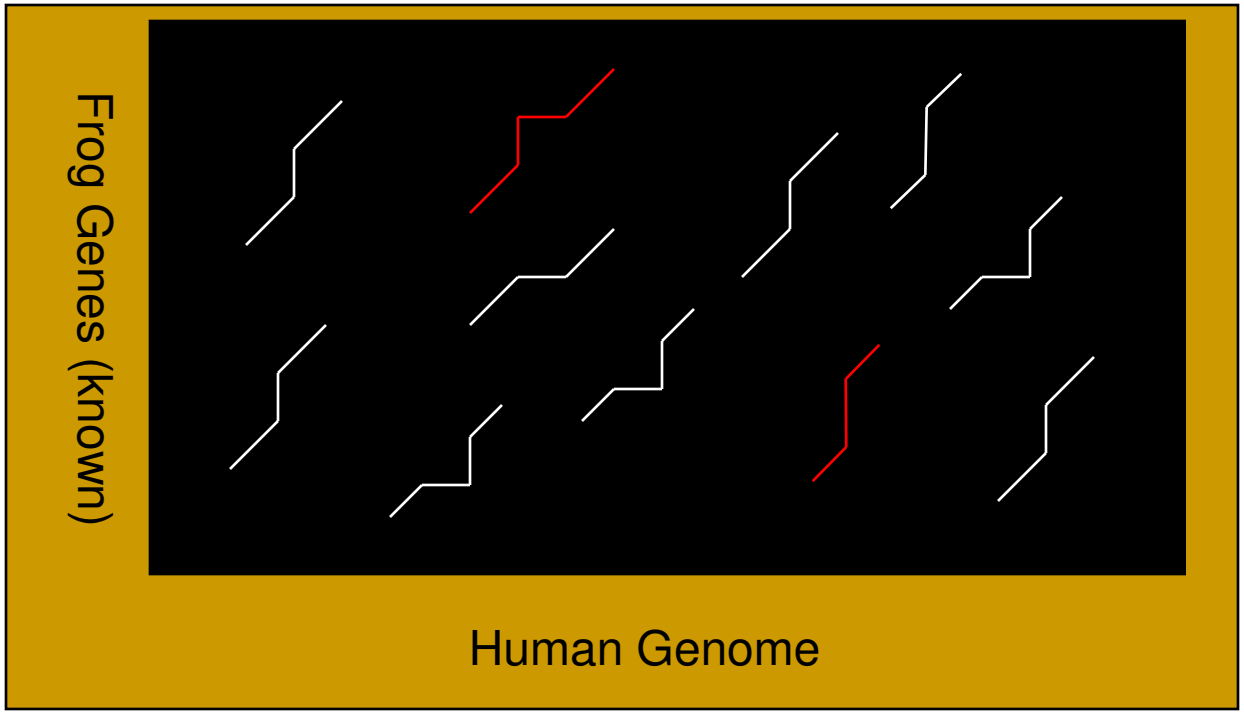
1. Poor definition of the putative exon endpoints.

2. Optimal chain of intervals may not correspond to a valid alignment.
   - Example: First interval may correspond to a suffix, whereas second interval may correspond to a prefix.



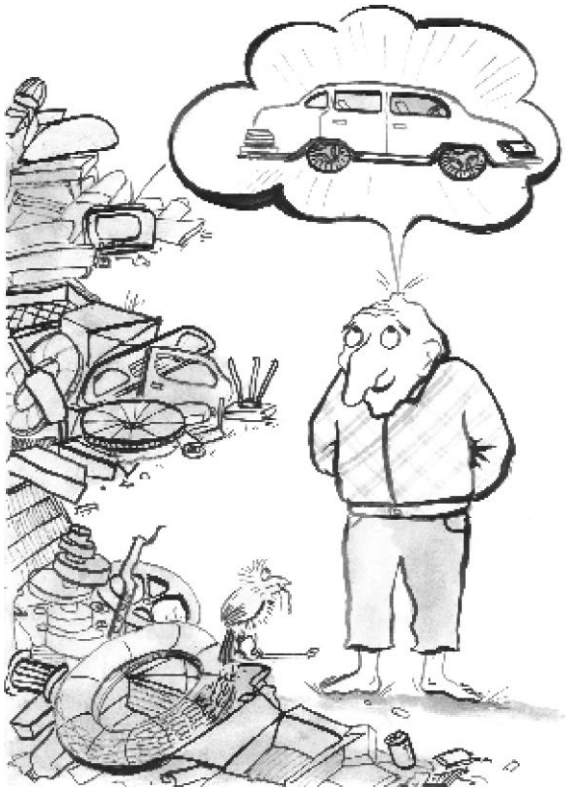
Infeasible Chains: Illustration

- Red local similarities form two non-overlapping intervals but do not form a valid global alignment.



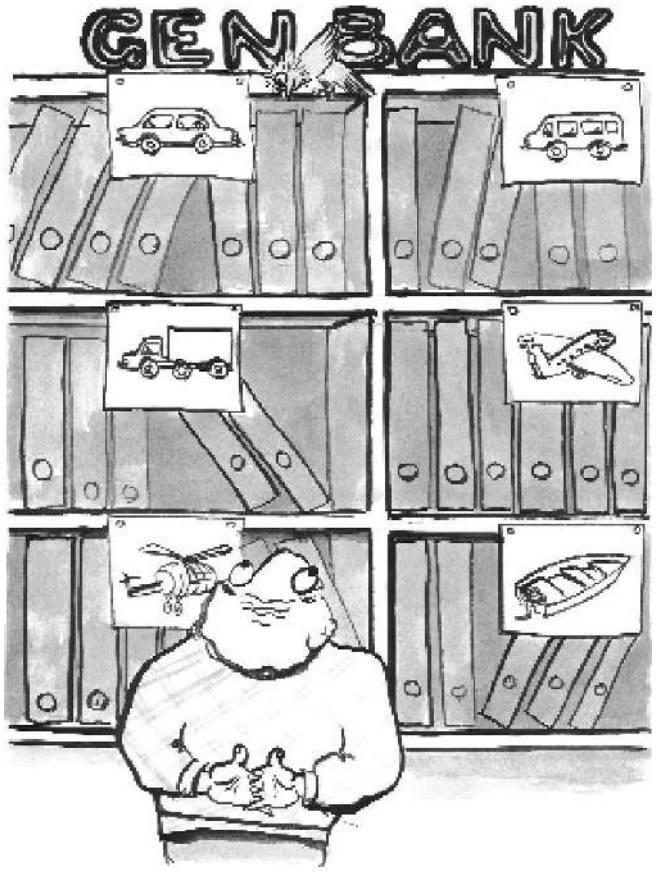
Gene Prediction Analogy

- The cell carries DNA as a blueprint for producing proteins, like a manufacturer carries a blueprint for producing a car.



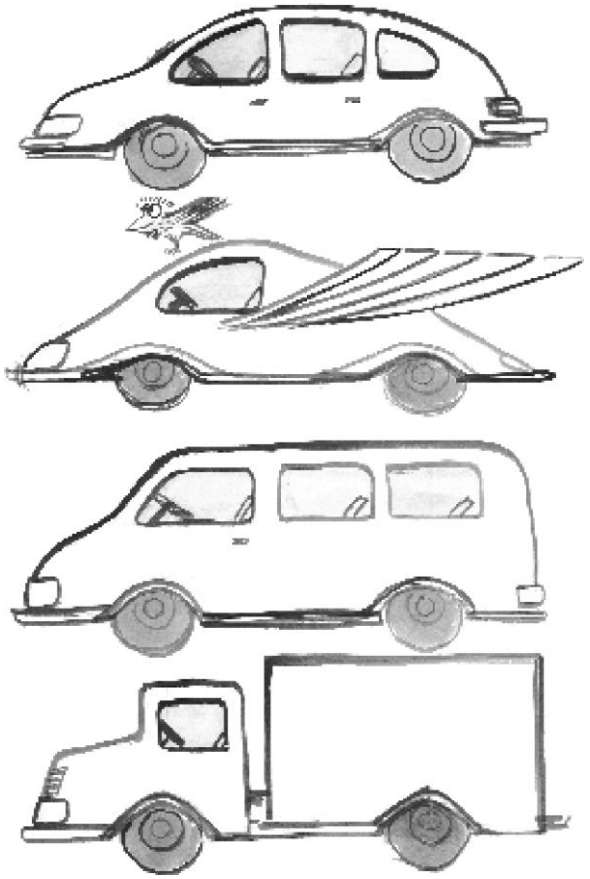
Gene Prediction Analogy: Using Blueprint

- Each protein has its own distinct blueprint for construction.



Gene Prediction Analogy: Assembling Exons

Gene Prediction Analogy: Still Assembling…



Section 3: Spliced Alignment

Spliced Alignment

- Mikhail Gelfand and colleagues proposed a **spliced alignment** approach of using a protein from one genome to reconstruct the exon-intron structure of a (related) gene in another genome.

- Spliced alignment begins by selecting either all putative exons between potential acceptor and donor sites or by finding all substrings similar to the target protein (as in the Exon Chaining Problem).

- This set is further filtered in a such a way that attempts to retain all true exons, with some false ones.

Spliced Alignment Problem: Formulation

- Goal: Find a chain of blocks in a genomic sequence that best fits a target sequence.

- Input: Genomic sequences *G*, target sequence *T*, and a set of candidate exons *B*.

- Output: A chain of exons *C* such that the global alignment score between *C** and *T* is maximum among all chains of blocks from *B*.
  - Here *C** is the concatenation of all exons from chain *C*.

Example: Lewis Carroll's "Jabberwocky"

- Genomic Sequence: "It was a brilliant thrilling morning and the slimy, hellish, lithe doves gyrated and gamboled nimbly in the waves."

- Target Sequence: "Twas brillig, and the slithy toves did gyre and gimble in the wabe."

- Alignment: Next Slide…

Example: Lewis Carroll's "Jabberwocky"

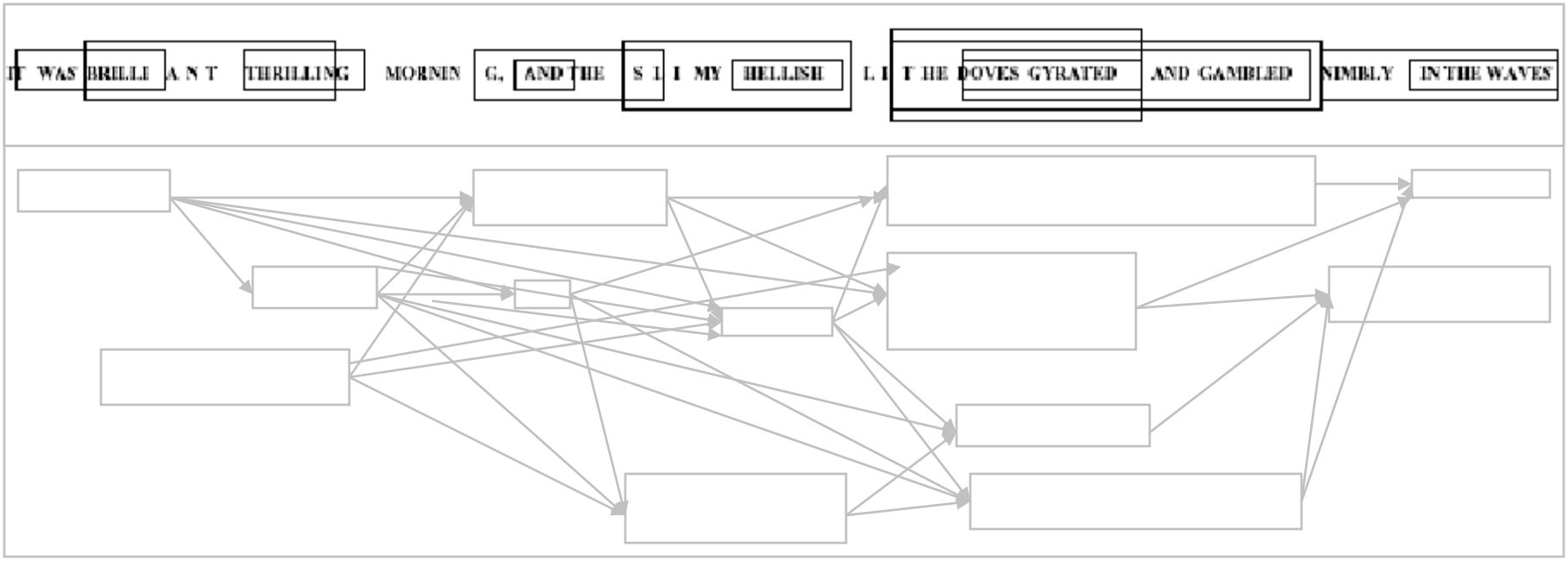
'T WAS BR I LLI G, AND TH E SLI TH Y TOVES DI D GYRE AND GI MBLE IN TH E WABE

T WAS BR I LLI G, AND TH E SL TH E DOVES GYRATED AND GAMBLED IN TH E WAVE

T WAS BR I LLI G, AND TH E SL TH E DOVES GYRATED NI MBLY IN TH E WAVE

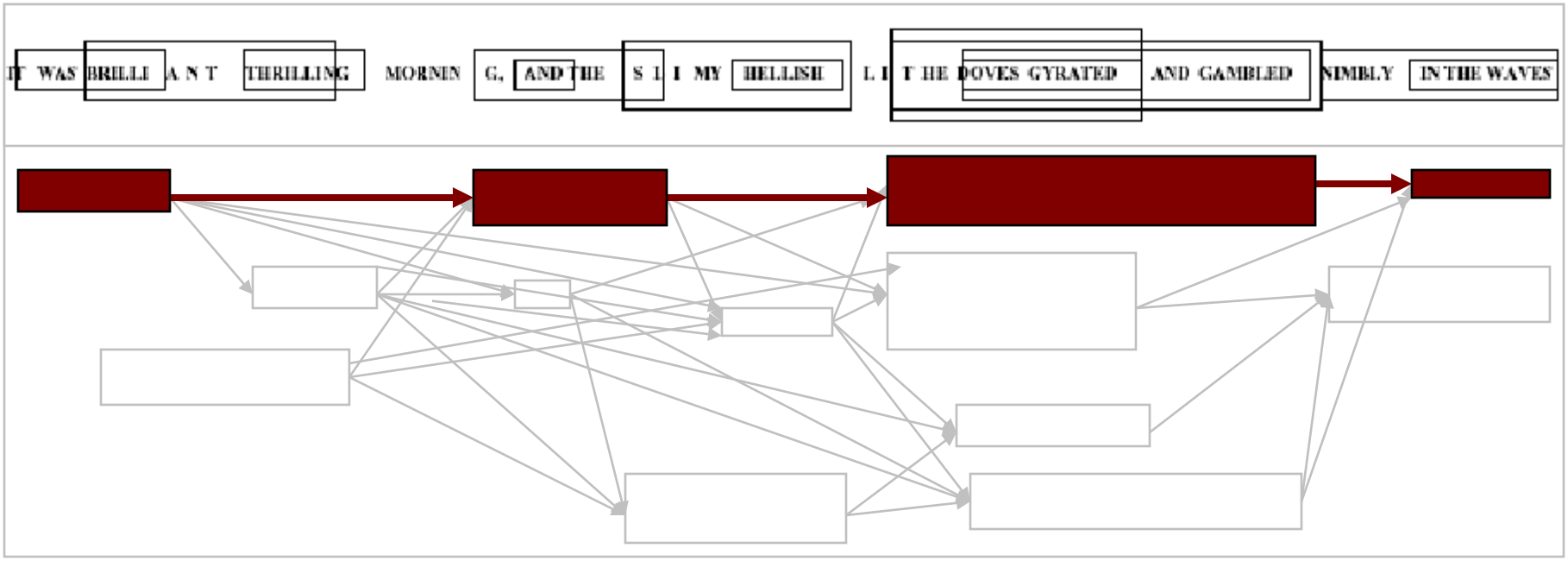
T HR I LLI NG AND H EL LI SH DOVES GYRATED AND GAMBLED IN TH E WAVE

T HR I LLI NG AND H EL LI SH DOVES GYRATED NI MBLY IN TH E WAVE

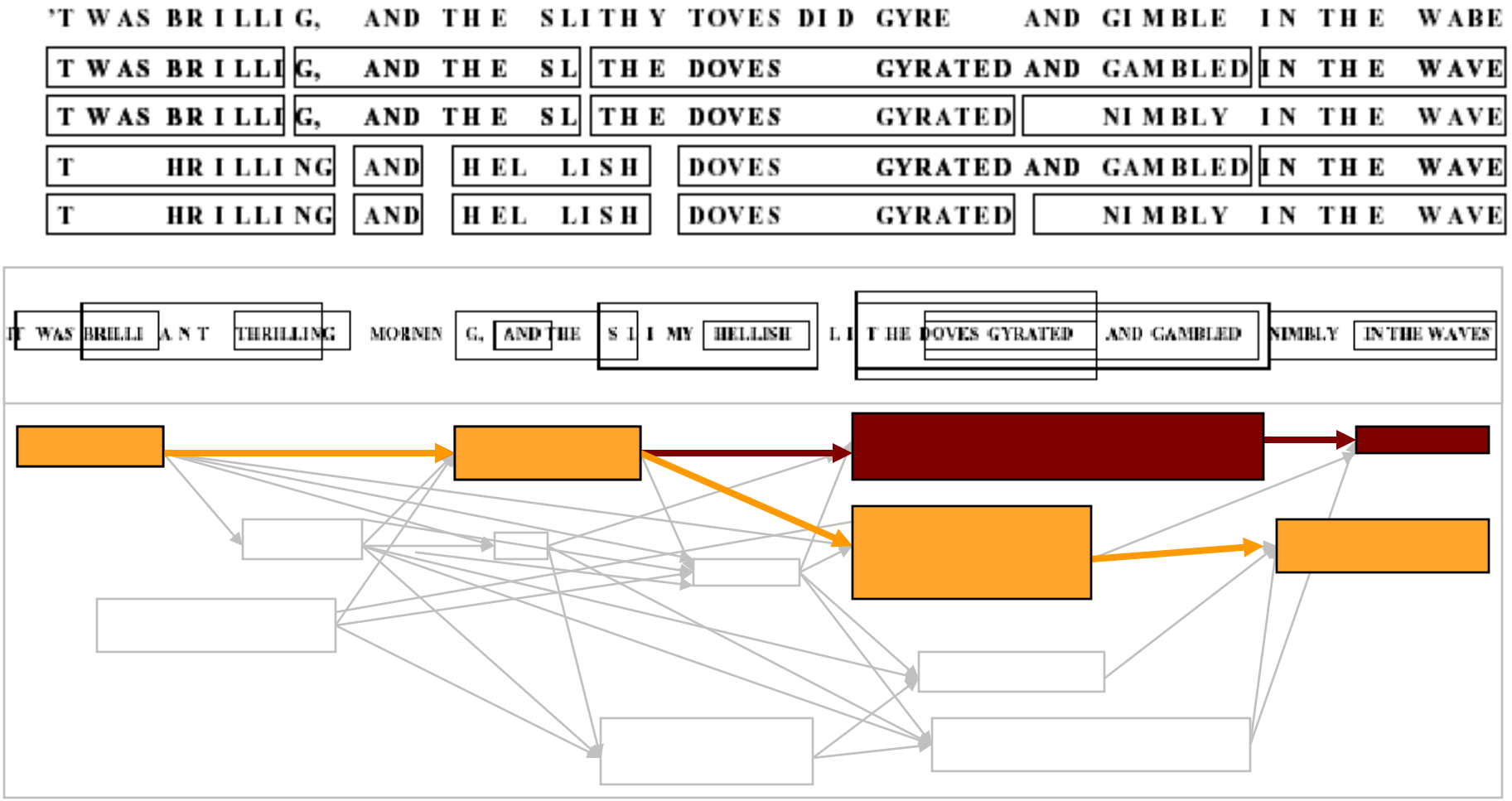


Example: Lewis Carroll's "Jabberwocky"

'T	W	A	S		B	R	I	L	L	I	G,		A	N	D		T	H	E		S	L	I	T	H	Y		T	O	V	E	S		D	I	D		G	Y	R	E			A	N	D		G	I	M	B	L	E		I	N		T	H	E		W	A	B	E
T	W	A	S		B	R	I	L	L	I	G,		A	N	D		T	H	E		S	L		T	H	E		D	O	V	E	S		G	Y	R	A	T	E	D		A	N	D		G	A	M	B	L	E	D	I	N		T	H	E		W	A	V	E
T	W	A	S		B	R	I	L	L	I	G,		A	N	D		T	H	E		S	L		T	H	E		D	O	V	E	S		G	Y	R	A	T	E	D						N	I	M	B	L	Y	I	N		T	H	E		W	A	V	E
T					H	R	I	L	L	I	N	G	A	N	D		H	E	L		L	I	S	H		D	O	V	E	S		G	Y	R	A	T	E	D		A	N	D		G	A	M	B	L	E	D	I	N		T	H	E		W	A	V	E
T					H	R	I	L	L	I	N	G	A	N	D		H	E	L		L	I	S	H		D	O	V	E	S		G	Y	R	A	T	E	D						N	I	M	B	L	Y	I	N		T	H	E		W	A	V	E



Example: Lewis Carroll's "Jabberwocky"



Example: Lewis Carroll's "Jabberwocky"

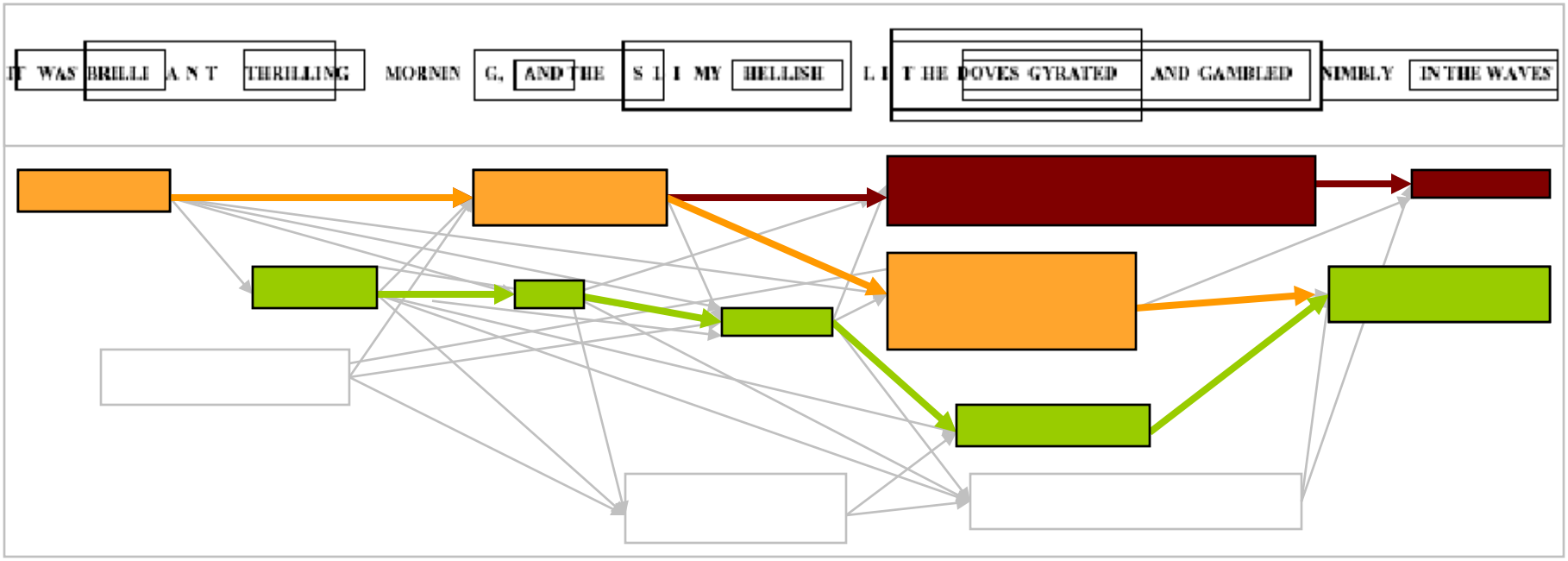
'T WAS BR I LLI G, AND TH E SLI TH Y TOVES DI D GYRE AND GI MBLE IN TH E WABE

T WAS BR I LLI G, AND TH E SL TH E DOVES GYRATED AND GAMBLED IN TH E WAVE

T WAS BR I LLI G, AND TH E SL TH E DOVES GYRATED NI MBLY IN TH E WAVE

T HR I LLI NG AND H EL LI SH DOVES GYRATED AND GAMBLED IN TH E WAVE

T HR I LLI NG AND H EL LI SH DOVES GYRATED NI MBLY IN TH E WAVE



Example: Lewis Carroll's "Jabberwocky"

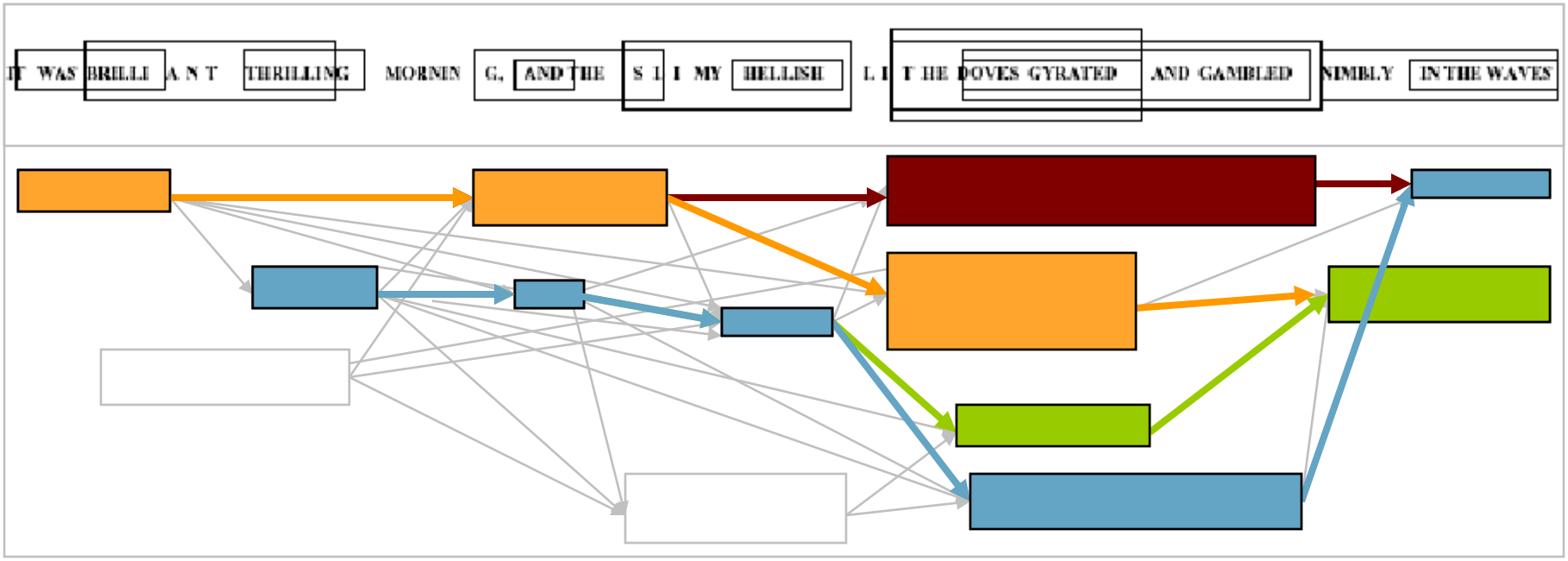
'T WAS BR I LLI G, AND TH E SLI TH Y TOVES DI D GYRE AND GI MBLE IN TH E WABE

T WAS BR I LLI G, AND TH E SL TH E DOVES GYRATED AND GAMBLED IN TH E WAVE

T WAS BR I LLI G, AND TH E SL TH E DOVES GYRATED NI MBLY IN TH E WAVE

T HR I LLI NG AND H EL LI SH DOVES GYRATED AND GAMBLED IN TH E WAVE

T HR I LLI NG AND H EL LI SH DOVES GYRATED NI MBLY IN TH E WAVE



Spliced Alignment: Idea

- Compute the best alignment between *i*-prefix of genomic sequence *G* and *j*-prefix of target *T*: *S(i,j)*
- But what is *"i*-prefix*"* of *G?*
- There may be a few *i*-prefixes of *G,* depending on which block *B* we are in.

Spliced Alignment: Idea

- Compute the best alignment between *i*-prefix of genomic sequence *G* and *j*-prefix of target *T: S(i,j)*
- But what is *"i*-prefix*"* of *G?*
- There may be a few *i*-prefixes of *G*, depending on which block *B* we are in.
- Compute the best alignment between *i*-prefix of genomic sequence *G* and *j*-prefix of target *T under the assumption* that the alignment uses the block *B* at position *i*: S(*i, j, B*).

Spliced Alignment Recurrence

- If i is *not* the starting vertex of block B:

$$S(i,j,B) = \max \begin{cases} S(i-1,j,B) - p \\ S(i,j-1,B) - p \\ S(i-1,j-1,B) + \delta(g_i, t_j) \end{cases}$$

- If i *is* the starting vertex of block B:

$$S(i,j,B) = \max_{B' \text{ preceding } B} \begin{cases} S(i,j-1,B) - p \\ S(end(B'),j,B') - p \\ S(end(B'),j-1,B') + \delta(g_i, t_j) \end{cases}$$

where p is some indel penalty and δ is a scoring matrix.

Spliced Alignment Recurrence

- After computing the three-dimensional table $S(i, j, B)$, the score of the optimal spliced alignment is:

$$\max_{B} \; S\big(end\,(B),\; length\;(T),\; B\big)$$

Spliced Alignment: Complications

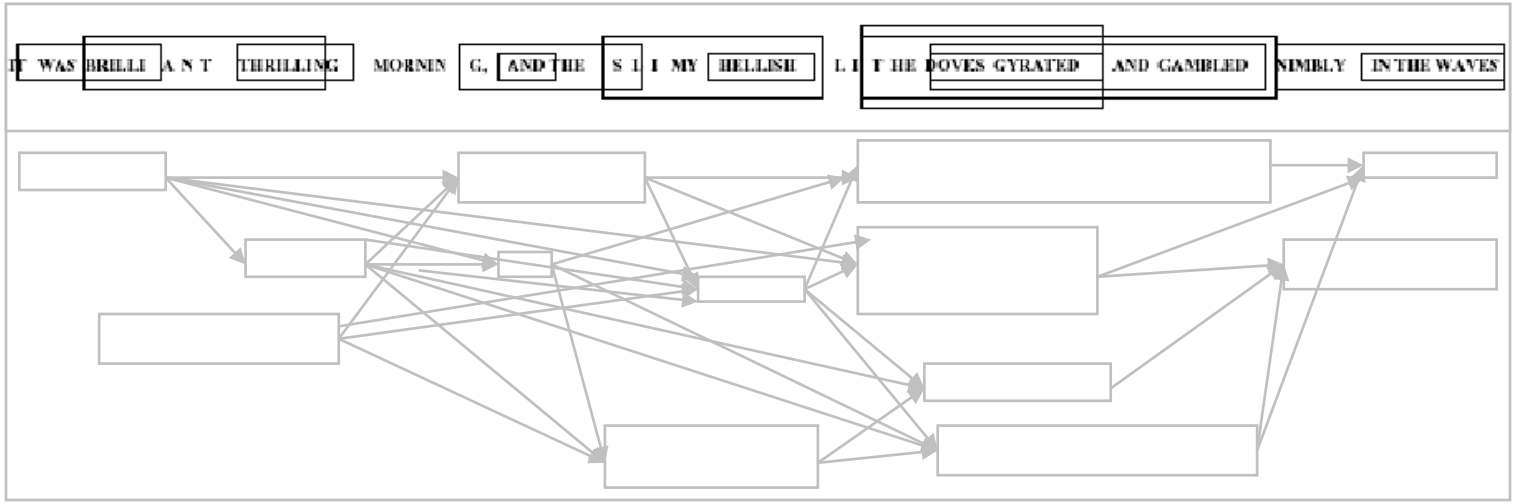
- Considering multiple *i*-prefixes leads to slow down.
  - Running time:

$$O\left(mn^2|B|\right)$$

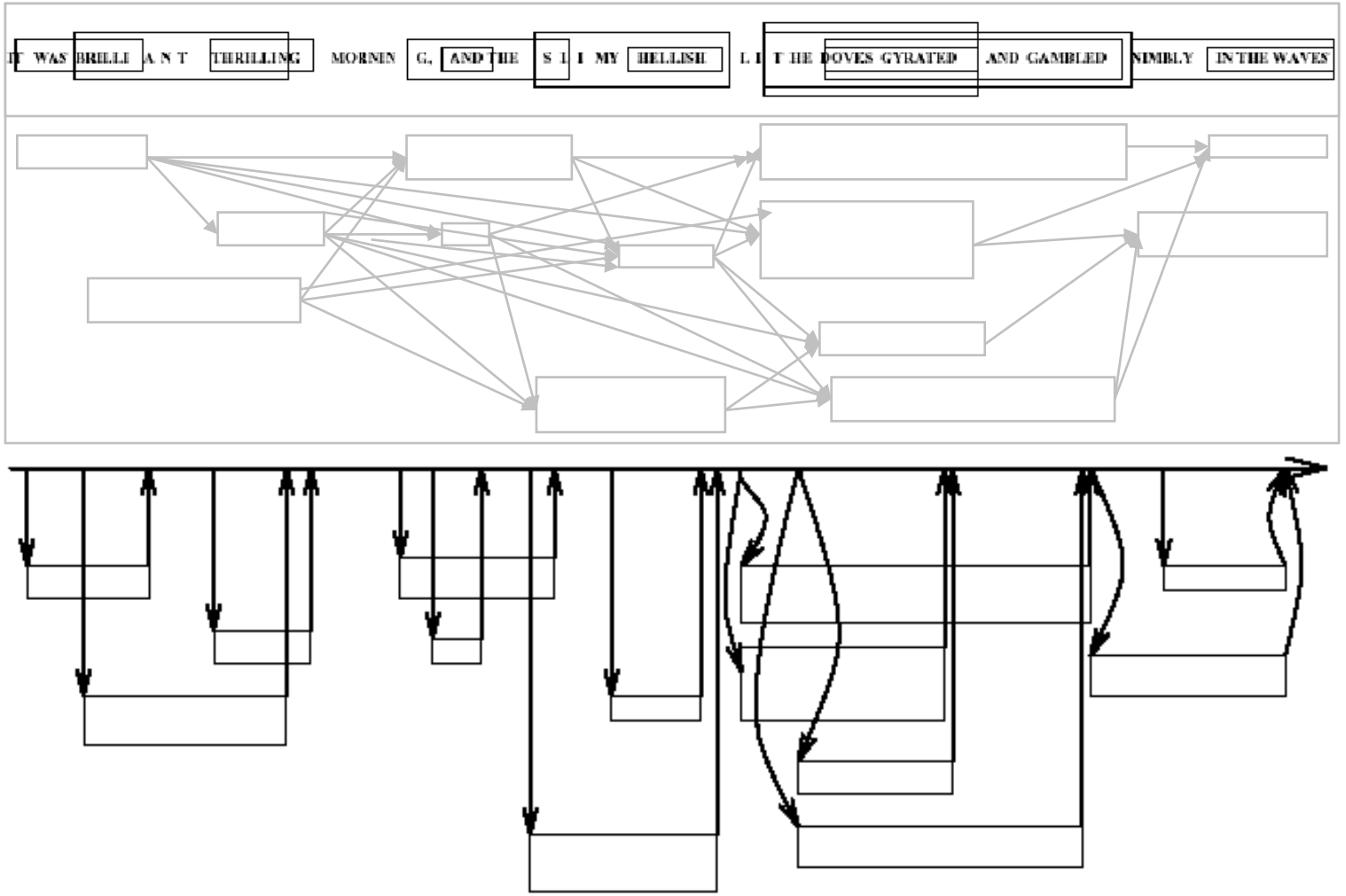
where m is the target length, n is the genomic sequence length and $|B|$ is the number of blocks.

- A **mosaic effect**: short exons are easily combined to fit any target protein.

Spliced Alignment: Speedup

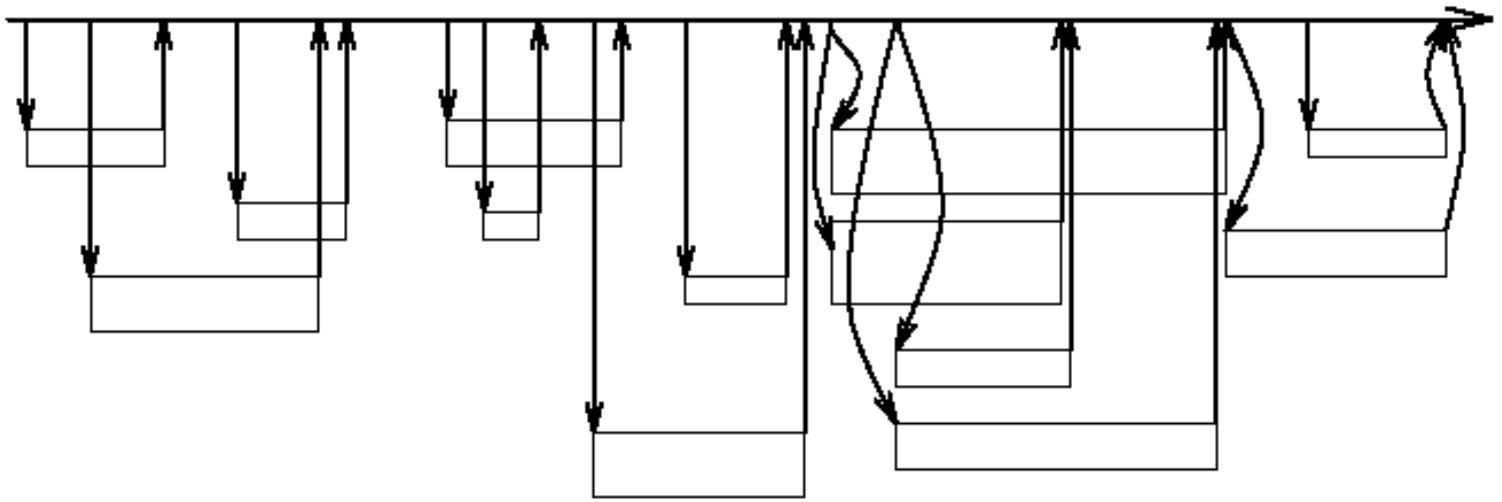


Spliced Alignment: Speedup



Spliced Alignment: Speedup

$$P(i,j) = \max_{B \text{ preceding } i} S(end(B), j, B)$$



Exon Chaining vs Spliced Alignment

- In Spliced Alignment, every path spells out the string obtained by concatenation of labels of its edges.
  - The weight of the path is defined as optimal alignment score between concatenated labels (blocks) and target sequence.
  - Defines weight of entire path in graph, but not the weights for individual edges.
- Exon Chaining assumes the positions and weights of exons are pre-defined.

Section 4: Gene Prediction Tools

Gene Prediction: Aligning Genome vs. Genome

- Goal: Align entire human and mouse genomes.
- Predict genes in both sequences simultaneously as chains of aligned blocks (exons).
- This approach does not assume any annotation of either human or mouse genes.

Gene Prediction Tools

1. GENSCAN/Genome Scan
2. TwinScan
3. Glimmer
4. GenMark

The GENSCAN Algorithm

- Algorithm is based on probabilistic model of gene structure.
- GENSCAN uses a "training set," then the algorithm returns the exon structure using maximum likelihood approach standard to many HMM algorithms (*Viterbi* algorithm).
- Biological input: Codon bias in coding regions, gene structure (start and stop codons, typical exon and intron length, presence of promoters, presence of genes on both strands, etc).
- Benefit: Covers cases where input sequence contains no gene, partial gene, complete gene, or multiple genes.

GENSCAN Limitations

- Does not use similarity search to predict genes.
- Does not address alternative splicing.
- Could combine two exons from consecutive genes together.

GenomeScan

- Incorporates similarity information into GENSCAN: predicts gene structure which corresponds to maximum probability conditional on similarity information.

- Algorithm is a combination of two sources of information:
  1. Probabilistic models of exons-introns.
  2. Sequence similarity information.

TwinScan

- Aligns two sequences and marks each base as gap (-), mismatch (:), or match (|).
  - Results in a new alphabet of 12 letters

    Σ = {A-, A:, A |, C-, C:, C |, G-, G:, G |, T-, T:, T|}.

- Then run Viterbi algorithm using emissions $e_k(b)$ where $b \in$ {A-, A:, A|, …, T|}.

http://www.stanford.edu/class/cs262/Spring2003/Notes/ln10.pdf

TwinScan

- The emission probabilities are estimated from human/mouse gene pairs.

- Example: $e_I(x|) < e_E(x|)$ since matches are favored in exons, and $e_I(x\text{-}) > e_E(x\text{-})$ since gaps (as well as mismatches) are favored in introns.

- Benefit: Compensates for dominant occurrence of poly-A region in introns.

Glimmer

- Glimmer: Stands for
  Gene **L**ocator and **I**nterpolated **M**arkov **M**odel**ER**

- Finds genes in bacterial DNA

- Uses interpolated Markov Models

Glimmer

- Made of 2 programs:
  1. **BuildIMM:**
     - Takes sequences as input and outputs the Interpolated Markov Models (IMMs).
  2. **Glimmer**:
     - Takes IMMs and outputs all candidate genes.
     - Automatically resolves overlapping genes by choosing one, hence limited.
     - Marks “suspected to truly overlap” genes for closer inspection by user.

GenMark

- Based on *non-stationary* Markov chain models.

- Results displayed graphically with coding vs. noncoding probability dependent on position in nucleotide sequence.