

Combinatorial Problems from Genomics

Greg Butler

Department of Computer Science

Concordia University, Montreal

www.cs.concordia.ca/~gregb

gregb@cs.concordia.ca

Abstract

Genomics seeks to unravel the role and regulation of genes in the genome for an organism, and to understand how these are conserved or evolved over time. Key combinatorial problems within genomics are the alignment of sequences, the construction of phylogenetic trees, and the interpretation of networks which represent metabolic pathways, regulatory networks, and interaction networks. Most of these problems are approximate rather than exact problems in that they must be able to treat partial matches, noisy data, and missing data in particular. This talk will introduce the problems.

Outline

Overview of Genomics and its Computational Problems

Classical Problems: Sequence Alignment, Phylogeny Trees, Patterns

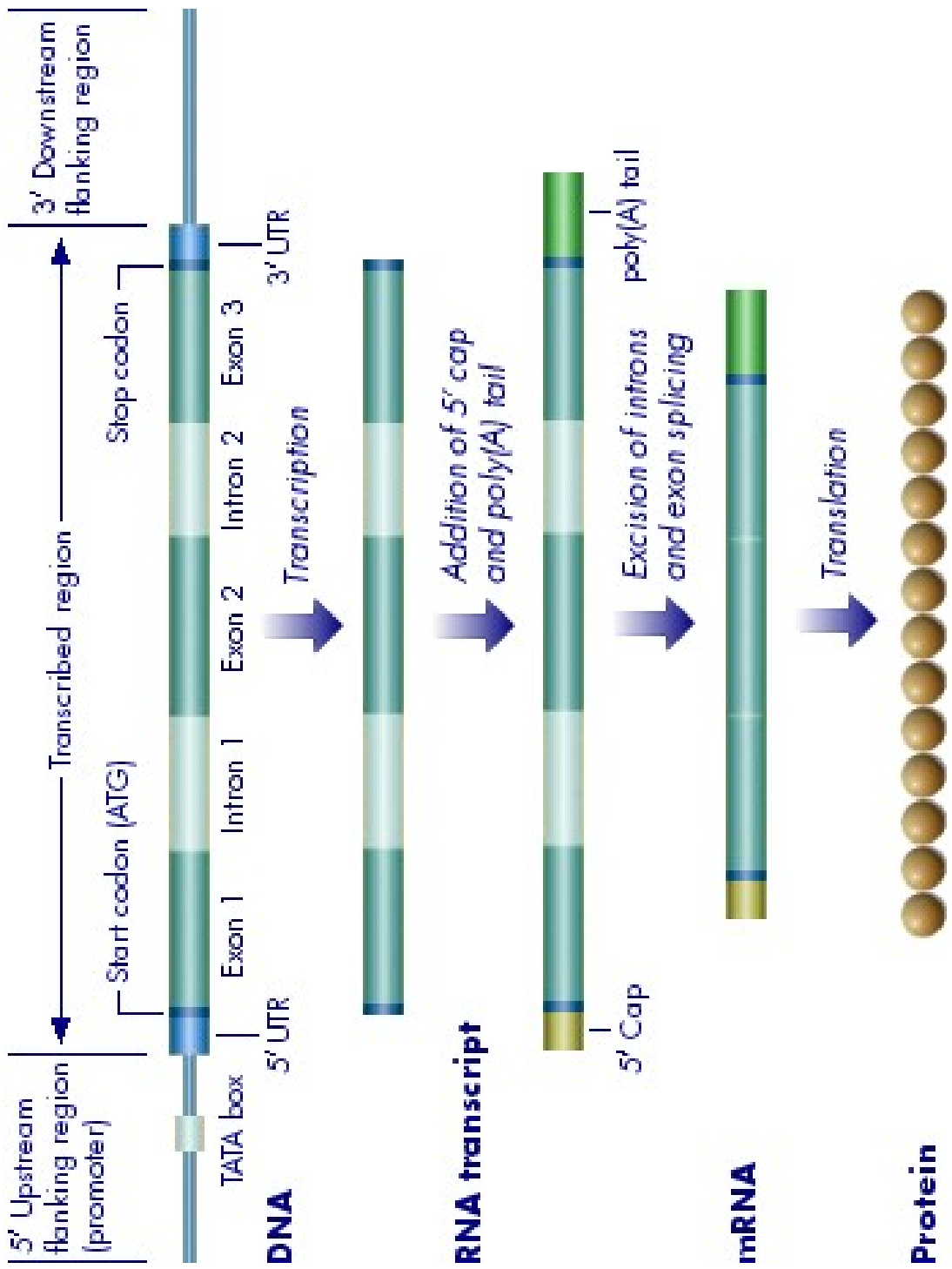
Modern Sequence Algorithms: Genomes and Phylogenomics

Network Reconstruction: Filling holes in annotations

Interaction Networks: The CytoScape example

Conclusion

Biology — Transcription and Translation



Biology — Transcription Regulation

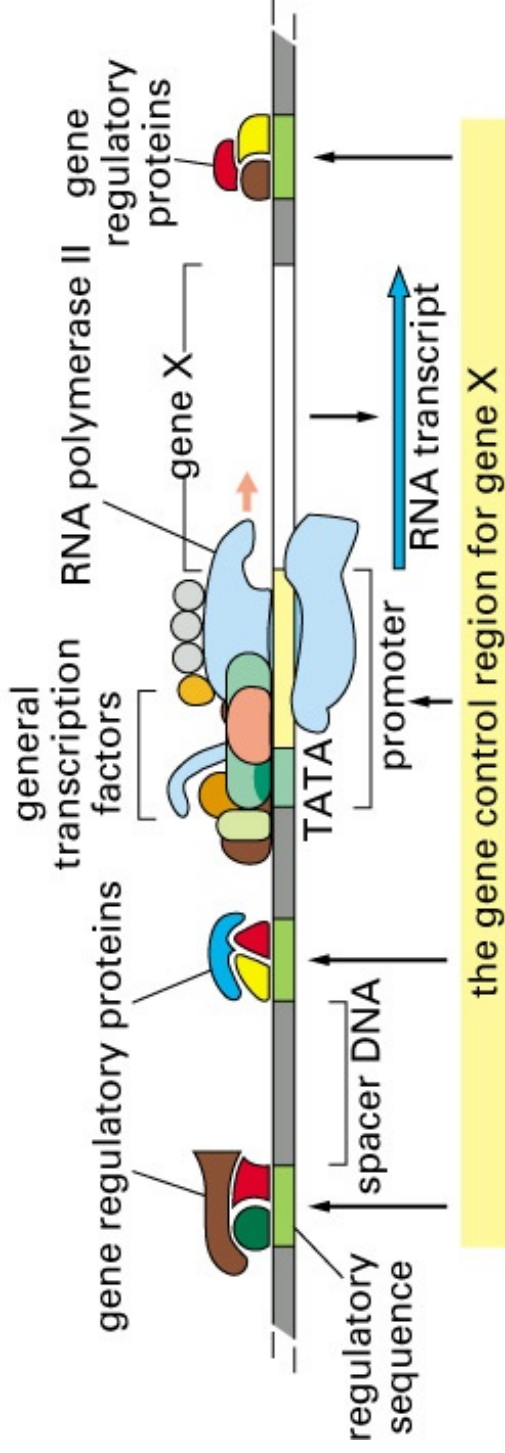
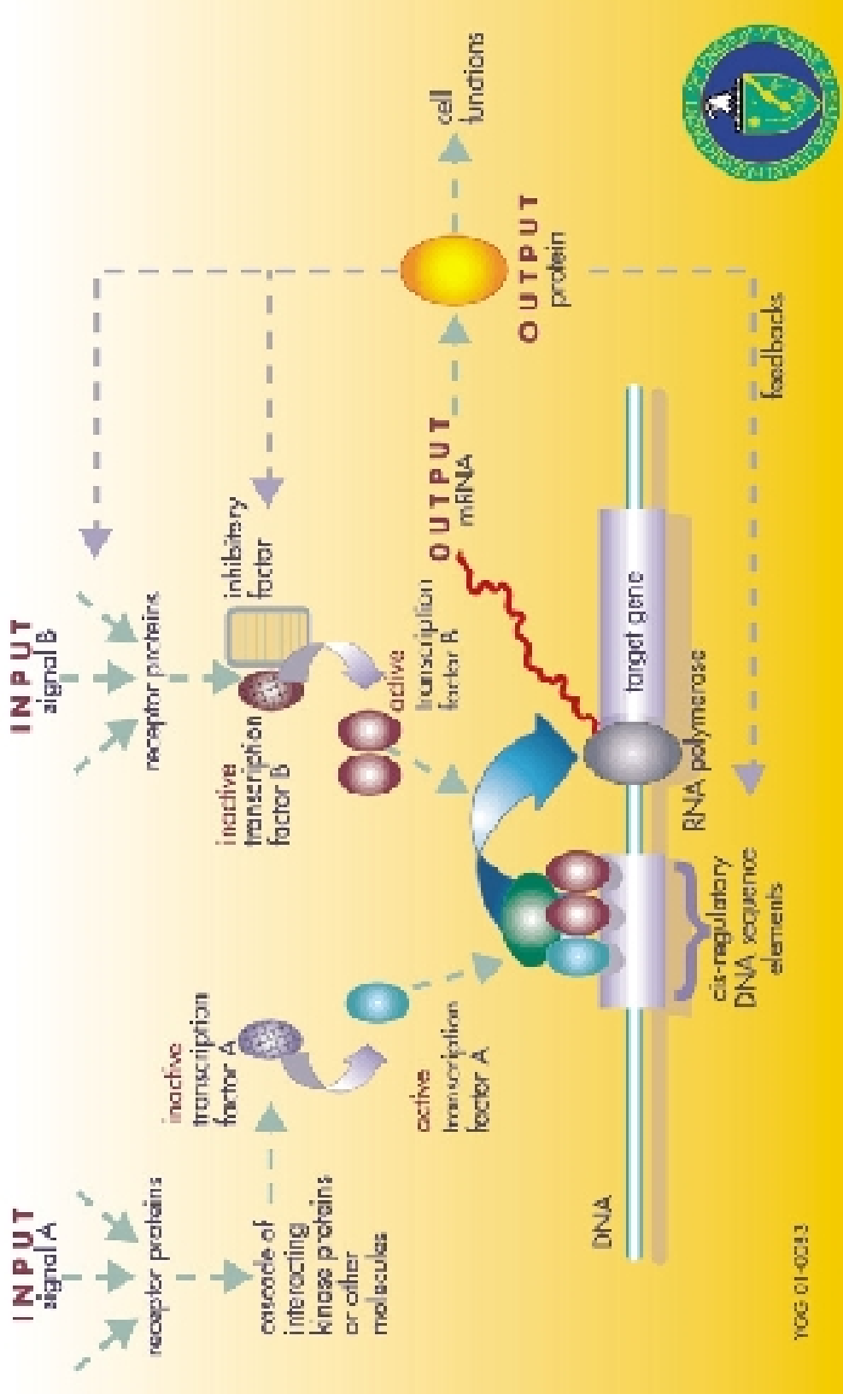


Figure 7-41. Molecular Biology of the Cell, 4th Edition.

Biology — Regulatory Networks

A GENE REGULATORY NETWORK



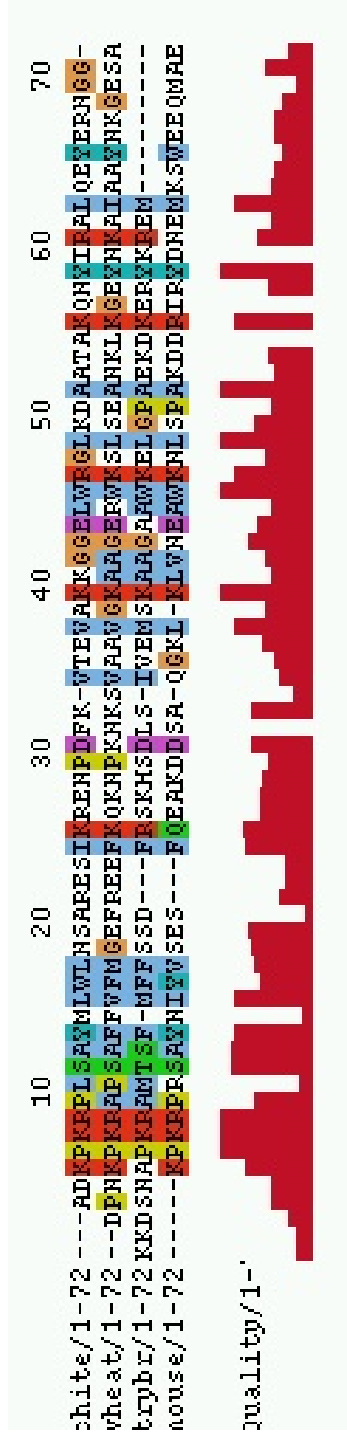
YOG 01-0033



Biology implies ...

... use Multiple Sequence Alignment (MSA)

MSA Problem: Given a set of protein sequences, and an *objective function*, determine the *optimal* alignment of the sequences.



Why?

Amino acid sequence

determines protein structure

determines enzyme function

Genomics

high-throughput, collecting or using all genes

Genomics project: determine full chromosome sequence, predict genes

EST Project: mine mRNA, assemble into "unigenes"

Microarray Project: genome-on-a-chip, study gene activity

Expression Project: splice gene into host, produce enzyme

Enzyme Assay and Characterization Project: chemistry and biochemistry

Proteomics, ...

What is Bioinformatics?

To store, organize & analyze biological data

Models — mathematical, statistical, physical

Algorithms — string, tree, graph, patterns

Database technology

Workflow and computational grids

Intelligent reasoning — expert systems, agents

Web access, visualization, usability

Rapid system development and evolution

Bioinformatics — Housekeeping Tasks

Materials Tracking

- identify (bar code labels) and record materials
- know location of all relevant physical materials
- know mappings caused by physical transfer of materials

Data Collection

- keep all data secure and accessible
- know quality and provenance of all data
- support analysis and interpretation of data

Quality Control

- know the quality of data
- provide reports to monitor quality of lab processes
- assist in diagnosis of problems with lab processes

Bioinformatics — Analysis Tasks

Sequence Analysis

determine high-quality sequence segment
... base call quality, remove contaminants, trim
assemble ESTs into unigenes

Sequence Annotation

similarity against known nucleic and protein sequences
... especially against targeted enzyme families
search for protein motifs and domains
is it secreted enzyme? other localization info?
classify to Gene Ontology category

Microarray Data Analysis

normalize: QC determines bad spots and dynamic range
set threshold for significant expression levels
determine highly expressed genes

Combinatorial Problems in Bioinformatics

Classical Problems: Sequence Alignment, Phylogeny Trees, Patterns
approximate string matching
clustering for trees
MSA and conservation for patterns

Modern Sequence Algorithms: Genomes and Phylogenomics

Myers' shotgun

BLAT

genome-EST sim4

genome-genome

MSA, phylogenomics

Network Reconstruction: Filling holes in annotations

need set of template paths
representations of networks

Interaction Networks: The CytoScope example
protein-protein interaction

Combinatorial Problems in Bioinformatics

Classical Problems

exact string matching

approximate string matching

Pairwise Alignment of Sequences

Problem: Given two sequences Q , S , find the optimal alignment.

Problem: Given a query sequence Q ,
and a sequence database D ,
find all sequences $S \in D$ similar to Q .

Why?

- infer biological relationship from similarity structure, function, ...
- filter candidate sequences for multiple alignment
- filter out “new” sequences Q for further study
- use similarity as one piece of information in larger analysis

Caveat: No black/white inference can be drawn from similarity!

Issues

- efficiency
- sensitivity
- scores, probabilities, and statistical significance

Nucleotide Sequence and Protein Sequence

>gi|29150082|emb|BX295539.1|NCB1D14 *Neurospora crassa* genomic DNA, BAC contig B1D14
GAATTCITTAAGCATTCTACAAACGGTTACTATATATAATTAATAACTAAAGCAATACTAATATATAAAA
TTCTAAATCGCITTTAAAGAATAAACTCCGGAATAAAGAGCGCTACCCAATATATTACACITTAGAAAGTA
GCTAGTAAAAATAATTTACGAAACTAGGGTTTAATTACAACCTCTATATAGTATTTAAAAATAATATAAA
GACTTCITTA AAAAATAACACAACTACTAAACAACGTAATTAACCTATAGTAAATATACCCCTAACCC
TATTACTATAAATTTAATAACTAATATATATTCTAAACCTATAATTTAATACTATATTTCGAATATAGTAT
TACTAAAACTCTCGTTTTTATAGTAGGTAGTACTACTACTAATGGTCTAACGAGTAATAACGATATTAC
AAATTAATTTATAGTAAAAAGAGGGGTTTTCTAAATAITTTTAATATTATTATAATAATAATAACAATTTCT
AATACTTCTAATACITTCCTAAACATTTCTAAACATTTCTAGTATTTTAGTATTTTGAATATATAAATTC
TAATATA CATAAAATCCATCTCTCCCTTCATACTCTACTACCCCAATAACAATACCCGTCCTCTA
TACTTAACTAACTTAAATATAATAITCGCTATACIATAAGATAAATAGCACAATGGTTATCTTAAATGA
TCTTAGATATAITGTTTTCTAAAAATCTAACTACTATCTAGAGGGTCACAGCATCGGATTTTATACCTCGA
TAATATACGGGCAGTTGGATGCCCTATCGGGCATACTACCGGTCTAAACAATCGTCAAACTACCATCCG
ACTACCATCCTTCCCTCACAATAACCCCTCCATAATAATAATCAATTTCAATTTCTATAAAATACATAAA
GGTAGATAGTCTTTTAGTATAACCGTCAAAAATAGTAATTTATAACTAATAAGCGCAGACTTACAATTTATAGAG

>gi|28416421|gb|AA042609.1| extracellular multicopper oxidase [*Phanerochaete chrysosporium*]
MLFALLALGAAIGVGAAPSSGPPVVQVPSMDNFVLGSGVAGPPQVRHYEFVVEEMLGAPDGVKPKMLVW
NGLFPPTIEVNQYDRLVVRVINRIQNATTHWHGIPQNGTAYDGTAGITECGIPPGQSLTYDFTFGDF
SGTTWHSYDTQYTDGVTGALIVHPTFPDPPGFPPTWDEELVIELIDVYHTFSTILNAQYLSPSGPIGGS
AGDEPV PDS GALNGLGQYHGLGNYFNFTLQPNKTYRLRLIHTGSLADIRFSVDNHPPLTVIEADGTLTEPT
VVAGLTLAVAQRYSVLLTTNQINGSYWMRTLQQIDMFTYKLPQGQNPDRNGIISYGTGTGPTLPPAKEDPG
ATLGGMQLNDLDPYTLPPAVPEAVPDNTKFYQVSFEFDNLP SGASRAYMNGTSWDPLPKTNILLEIQRAY
NAGRAFAPEGASVQFGNQFLITEDKIEVLDLVLENDNGDHPFHLHGYSYPMIGVDSYDPPNNVTLNTV
NPLSRDTVEIPANGWVRLRFITYNPGAWTLHCHISWHMSAGLLMQFVTLTPSVAATLPIPQVIANMCSAA

Gapped Alignment of Protein Sequences by Blast

gi|1346405|sp|P17489|LACL_EMENI Laccase I precursor (Benzenediol:oxygen oxidoreductase) (Urishiol oxidase) (Conidial laccase)
Length = 609

Score = 85.9 bits (211), Expect = 3e-16

Identities = 47/123 (38%), Positives = 67/123 (54%), Gaps = 7/123 (5%)

Query: 48 H Y E F V V E E M ----- L G A P D G V T - K P M L V V N G L F P G P T I E V N Q Y D R L V V R V I N R I Q N A I T T I 101

H F V E + G + P + G T + M + N G + P G P + + + D + V V I N + T I +

Sbjct: 19 H A R F V R E T L E L T W E Y G S P N G G T P R E M V F T N G E Y P G P D L I F D E D D D V E V L V I N N L P E N T I V 78

Query: 102 H W H G I P Q N G T A Y D G T A G I T E C G I P P G Q S L T Y D F T F G D F S G T T W H S H Y D T Q Y T D G V T G A 161

H W H G + T D G G + T + I P G + T Y F + G T W + H S H Y D G G A

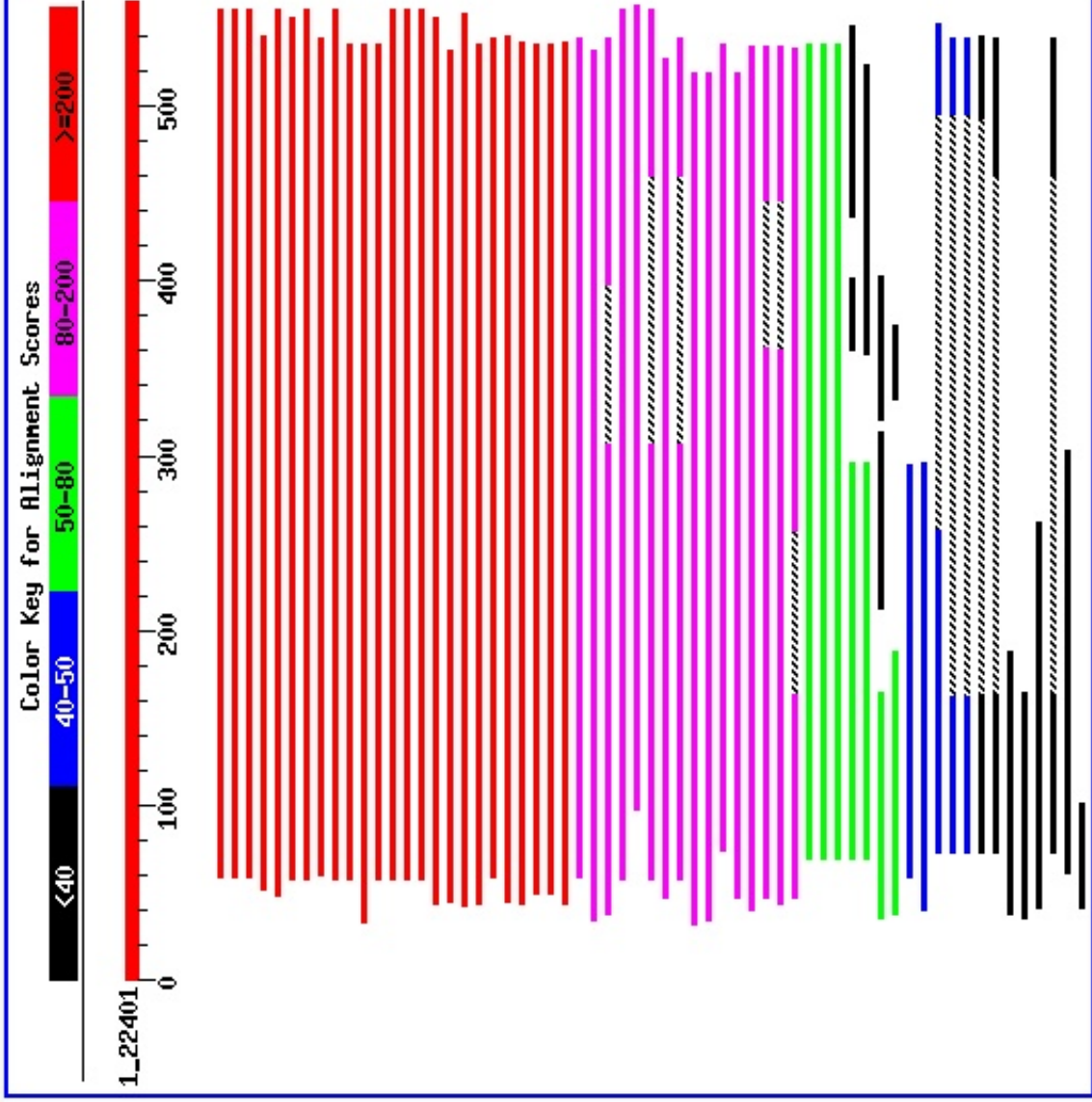
Sbjct: 79 H W H G L E M R E T P E A D G V P G L I Q T P I E P G A T F T Y R F R A Y P - A G T F W Y H S H Y K G L M Q D G Q V G A 137

Query: 162 L I V 164

+ +

Sbjct: 138 M Y I 140

Graphical Summary of Blast Results



Textual Summary of Blast Results

Sequences producing significant alignments:		Score (bits)	E Value
gi 1351670 sp O09920 YAK8_SCHPO	Putative multicopper oxidas...	249	2e-65 G
gi 18281739 sp Q99056 LAC5_TRAVI	Laccase 5 precursor (Benze...	245	3e-64
gi 2833232 sp Q12717 LAC5_TRAVE	Laccase 5 precursor (Benzen...	240	6e-63
gi 2829670 sp P78722 LAC2_PODAN	Laccase II precursor (Benze...	236	2e-61
gi 47117883 sp Q01679 LAC1_PHLRA	Laccase precursor (Benzene...	234	6e-61
gi 2833228 sp Q12542 LAC2_AGABI	Laccase II precursor (Benze...	232	2e-60
gi 2828219 sp P38993 FET3_YEAST	Iron transport multicopper ...	231	4e-60 G
gi 1730082 sp Q02497 LAC1_TRAHI	Laccase precursor (Benzened...	231	4e-60
gi 2842752 sp Q99044 LAC1_TRAVI	Laccase 1 precursor (Benzen...	230	9e-60
gi 2833234 sp Q12719 LAC4_TRAVE	Laccase 4 precursor (Benzen...	229	1e-59
gi 2827764 sp P24792 ASO_CUCMA	L-ascorbate oxidase precurs...	228	3e-59
gi 2842755 sp Q99055 LAC4_TRAVI	Laccase 4 precursor (Benzen...	228	3e-59
gi 2842753 sp Q99046 LAC2_TRAVI	Laccase 2 precursor (Benzen...	226	1e-58
gi 2833233 sp Q12718 LAC2_TRAVE	Laccase 2 precursor (Benzen...	225	2e-58
gi 34922426 sp O59896 LAC1_PYCCI	Laccase precursor (Benzene...	225	2e-58
gi 2833227 sp Q12541 LAC1_AGABI	Laccase I precursor (Benzen...	224	5e-58

Pairwise Alignment — Concepts

Nucleotide and amino-acid sequences

Alignment

- Global vs Local Alignment
- Gapped vs Non-Gapped Alignment

Alignment Scores

- edit operations: substitution, insert, delete, (indel)
 - scoring matrix
 - gap weights
 - optimal alignment
- similarity is measured by alignment score
- statistical models

Deterministic vs Heuristic Algorithms

Pairwise Alignment — Brief History

Needleman-Wunsch (1970)

- global alignment with gaps
- dynamic programming
- cubic algorithm

Smith-Waterman (1981)

- local alignment with gaps
- dynamic programming
- $O(qN)$

FASTA (Lipman & Pearson, 1985; 1988) “Fast-All”

- global/local alignment with gaps
- heuristic
- sensitive

BLAST 1.4 (Altschul, Gish, Miller, Myers, Lipman, 1990)

“Basic Local Alignment Search Tool”

- local alignment without gaps
- heuristic
- faster than FASTA but less sensitive

BLAST 2.0 (Altschul, Madden, Schaffer, Zhang, Miller, Lipman, 1997)

BLAST in Detail

BLAST is an exhaustive search of a sequence database

Uses heuristics; turns approximate match problem into exact match

Process Overview

1. Read in query, pre-process query
2. Scan each DB sequence for exact hits
extend hits, note alignments
3. Post-processing:
 - rank alignments by score
 - do statistical analysis

Combinatorial Problems in Bioinformatics

Modern Sequence Algorithms

Myers' shotgun assembly

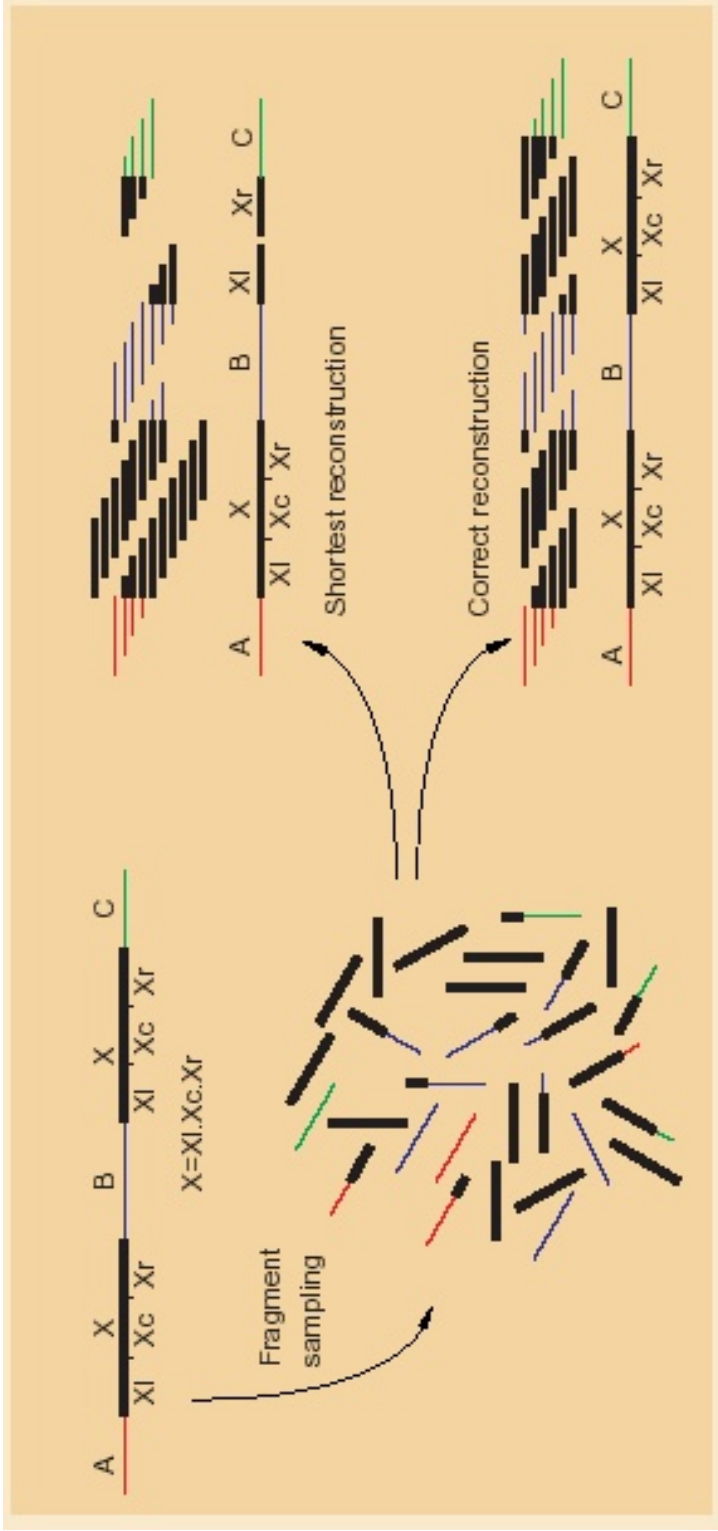
BLAT

genome-EST: sim4

genome-genome

MSA and phylogenomics

Myer's Shotgun Assembly



The fragment-assembly problem

Given the reads obtained from a shotgun protocol, the computational problem, called *fragment assembly*, is to infer the source sequence given the collection of reads. For the purposes of illustration, we might parameterize a typical problem occurring in practice today as follows (see the "Definitions" box for help with the terminology). For a source strand of length $G = 100$ Kbp, we would then typically sequence $R = 1,500$ reads of average length $\bar{L} = 500$. Thus, we would collect altogether $N = R\bar{L} = 750$ Kbps of data, so that we have sequenced on average every base pair in the source $\bar{c} = N/G = 7.5$ times. The quantity \bar{c} is the average sequencing coverage, and practitioners say that the source has been sequenced to $7.5\times$ coverage. In practice, an investigator will decide on a given level of coverage and then sequence inserts until a total of $N = G\bar{c}$ base pairs of data have been collected. Software for fragment

Definitions

G Length of target sequence
 \bar{L} Average length of sequence read
 R Number of sequencing reads in shotgun data set
 N $R\bar{L}$, total number of base pairs sequenced
 \bar{l} Average length of a clone insert
 \bar{c} N/G , average sequence coverage
 \bar{m} $R\bar{l}/2G$, average clone or map coverage

assembly must account for the following essential characteristics of the data:

- **Incomplete coverage:** Not every source base pair is sequenced exactly \bar{c} times due to both the stochastic nature of the sampling and cloning bias I've mentioned. Some portions of the source might be covered by more than \bar{c} reads, and others might not be covered at all. In general, there can be several such *gaps* or maximal contiguous regions where the source sequence has not been sampled. Gaps necessarily dictate a fragmented, incomplete solution to the problem.

- **Sequencing errors:** The gel-electrophoretic experiment yielding a read, like most physical experiments, is prone to error, especially near the end of a read where the signal strength and separation of consecutive prefix fragments become small. In a very stringently controlled production environment, the error rate is less than 1% in the first 500 or so bases. Thereafter, the error rate increases rapidly, reach-

ing more than 15% from 650 to 900 bases into the read, after which the resulting sequence is effectively unusable. (However, I have seen data sets where an error rate of 5% occurs in the "sweet" part of the read, consisting of the first 500 bases.)

- **Unknown orientation:** DNA is a double-stranded helix. Which of the source sequence's two strands is actually read depends on the arbitrary way the given insert orient itself in the vector. Thus we do not know whether to use a read or its Watson-Crick complement in the reconstruction. The Watson-Crick complement $(a_1 a_2 \dots a_n)^c$ of a sequence $a_1 a_2 \dots a_n$ is $wc(a_n) \dots wc(a_2) wc(a_1)$ where $wc(A) = T, wc(T) = A, wc(C) = G,$ and $wc(G) = C$.

I will now develop a mathematical formulation of the fragment-assembly problem. For input, we have a collection of reads

$$\mathcal{F} = \{f_i\}_{i=1}^R$$

that are sequences over the four-letter alphabet $\Sigma = \{A, C, G, T\}$. An *ε -layout* is a string S over Σ and a collection of R pairs of integers, $(s_i, e_i)_{i \in [1, R]}$, such that

- if $s_j < e_j$ then f_j can be aligned to the substring $S[s_j, e_j]$ with less than

ϵ $|f_i|$ differences, and

- if $s_i > e_i$ then f_i can be aligned to the substring $S[a_i, s_i]^k$ with less than ϵ $|f_i|$ differences, then
- $\cup_j [\min(s_i, e_i), \max(s_i, e_i)] = [1, |S|]$.

The string S represents the reconstruction of the source strand, and the integer pairs indicate the substrings of S that gave rise to each read. The order of s_i and e_i encode the *orientation* of the fragment read in the layout—that is, whether f_i was sampled from S or its complement strand. The parameter $\epsilon \in [0, 1]$ models the maximum error rate of the sequencing process.

The set of ϵ -layouts models the set of all possible solutions to the fragment-assembly problem. Of course, there are many such solutions, so the computational problem is to find one that is in some sense best. Traditionally, the fragment-assembly problem has been phrased as one of finding a shortest common superstring (SCS) of the fragment reads within error rate ϵ ; that is, find an ϵ -layout for which S is as short as possible. Unfortunately, as Figure A illustrates, this appeal to parsimony often produces over-compressed results when the source sequence contains repeated subsegments. This tendency has prompted the proposal of maximum-likelihood criteria based on the distribution of fragment start points in the layout.¹

While such a criteria provides a better objective function, algorithm designs for computing it have proven elusive.

A common computational architecture for fragment assembly, advocated by several authors,²⁻⁴ divides the problem into three phases: *overlap*, *layout*

and *consensus*. The overlap phase compares every fragment read against every other read (in both orientations) to determine if they overlap. Given the presence of sequencing errors, an overlap is necessarily approximate in that not all characters in the overlapping region coincide. This problem is a variation on traditional sequence comparison where the degree of difference permitted is bounded by ϵ . The best deterministic designs for finding all ϵ -overlaps lets us solve problems on the order of $M = 1$ to 5 Mbp in a matter of minutes on a typical workstation.⁵ For contexts requiring even greater speed, most investigators resort to heuristics that detect overlapping reads by finding exact common substrings of some length k using a hashing scheme. Typically, they choose k to provide the best compromise between sensitivity and

speed for a given M and ϵ . Conceptually, we can think of the result of the overlap phase as producing an *overlap graph* in which every vertex models a read and every edge an ϵ -overlap between two reads.

The layout phase determines the pairs (s_i, e_i) that position every fragment in the assembly. In graph theoretic terms, we accomplish this by selecting a spanning forest of the overlap graph; such a subset positions every fragment with respect to every other, transitively, through the overlaps on the path between them. Finding a spanning forest that optimizes a criterion such as shortest or most likely is known to be NP-hard.⁶ Investigators have proposed greedy algorithms that come within a given factor of optimal,^{7,8} simulated annealing⁹ and genetic algorithms,¹⁰ relaxation methods based on generat-

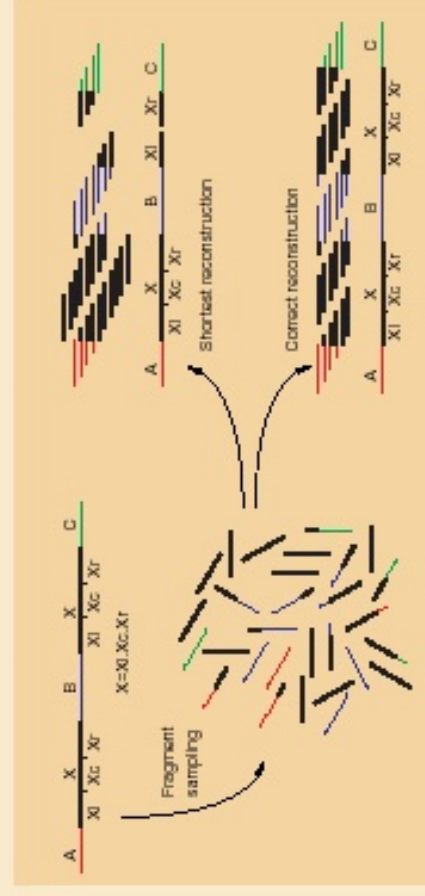


Figure A. The shortest answer isn't always the correct one. A DNA source at the upper left consists of unique stretches A, B, and C separated by a repeated sequence X. Below it, the source has been sampled perfectly uniformly across the target, as evidenced by the correct reconstruction of the pieces shown at lower right. But note the result in the upper right of a program that produces the minimum-length reconstruction. The interior portion X_c of the relevant sequence, which is covered only by reads completely interior to X, is over-represented.

MSA — Partial Order Alignment

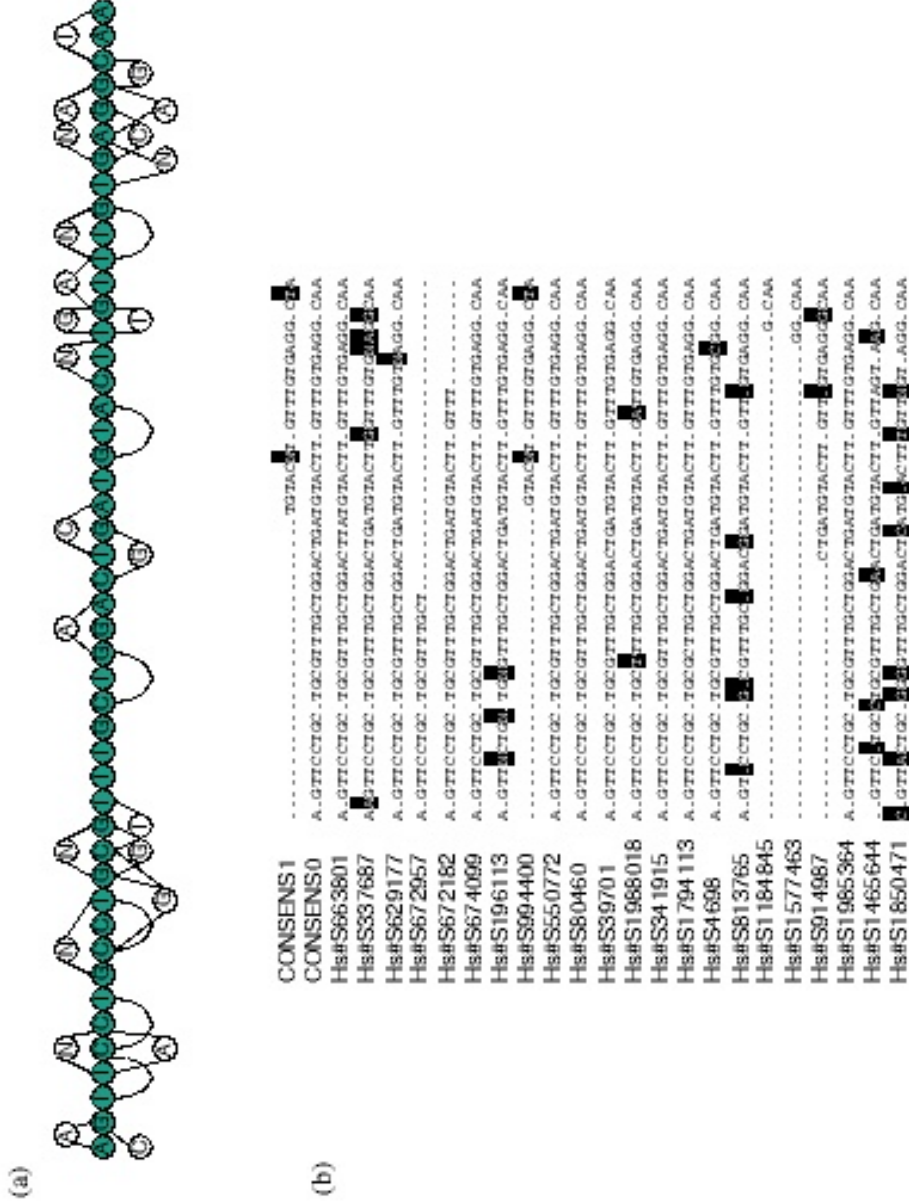


Fig. 2. Partial order alignment representation of UniGene EST sequences. Part of an alignment generated by POA of EST sequences from UniGene cluster Hs. 100194: (a) PO-MSA representation; (b) traditional RC-MSA representation of the same data. Only part of the very large alignment is shown.

Combinatorial Problems in Bioinformatics

Network Reconstruction: Filling holes in annotations

need set of template paths

representations of networks

Peter Karp, SRI, MetaCyc and PathoLogic Tools

Network Reconstruction — PathoLogic

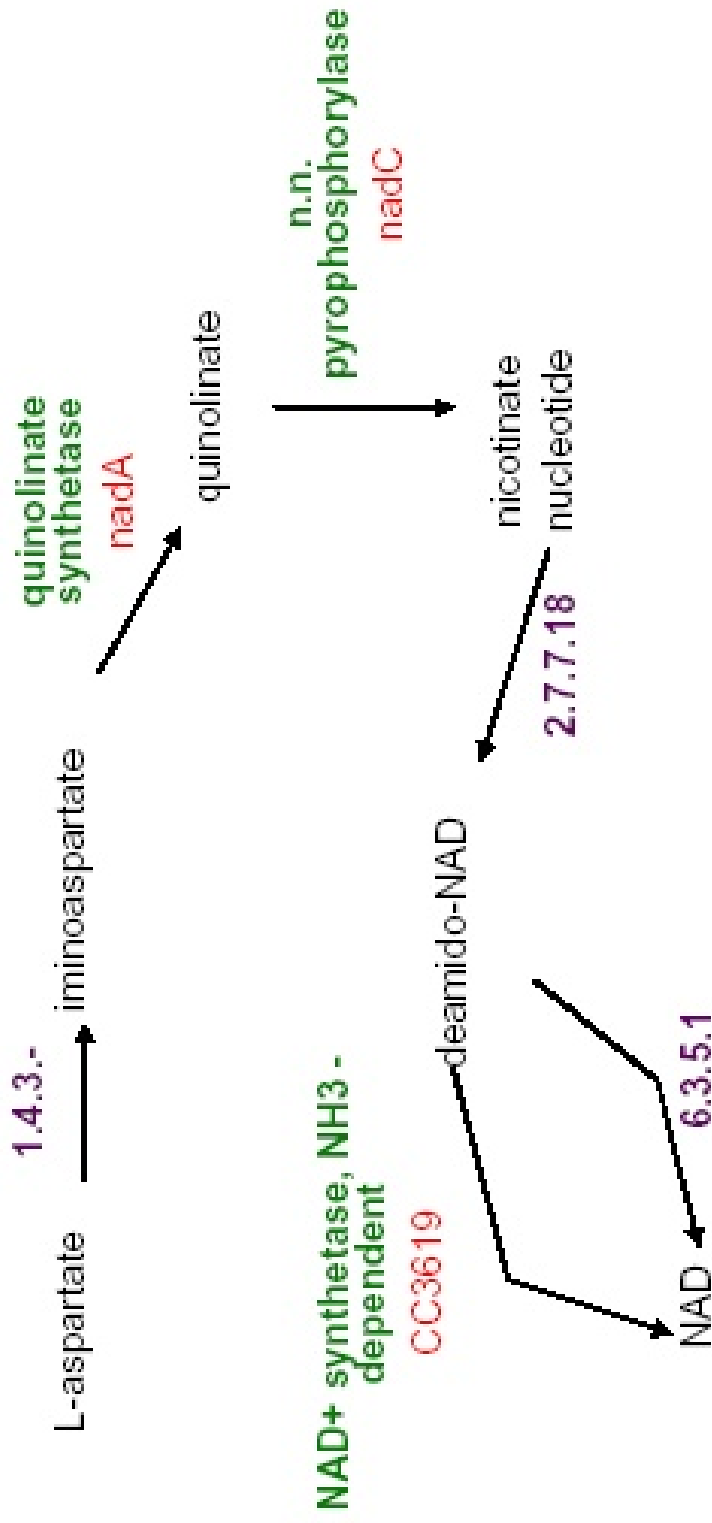


Figure 1

Example pathway created by PathoLogic for the *Caulobacter crescentus* PGDB, CauloCyc. The enzymes for the quinolinate synthetase, nicotinate-nucleotide pyrophosphorylase, and NAD(+) synthetase reactions are known. The enzymes for the 1.4.3.-, nicotinate-nucleotide adenyltransferase, and NAD(+) synthase (glutamine-hydrolyzing) reactions are missing.

Network Reconstruction — PathoLogic

Table 1: Node names and descriptions for Bayes classifier.

node	description
has-function	true if the protein has the function required to fill the pathway hole, false if it does not
Shotgun-score	the number of query sequences whose BLAST output included the candidate sequence
best-E-value	negative log of the E-value for the best alignment of the candidate with a query sequence
average-rank	the average rank of the candidate sequence in the BLAST output lists (e.g., if a candidate is the best hit in each search, the average rank for the candidate is 1)
average-fraction-aligned	the average of each alignment length normalized by the length of the query sequence
pathway-directon	true if the hit is in the same directon as another gene in the same pathway; a directon is a contiguous series of genes transcribed in the same direction
adjacent-rxns	true if the hit is adjacent to one of the genes coding the enzyme for an adjacent reaction in the pathway

Combinatorial Problems in Bioinformatics

Interaction Networks: The CytoScape example

Cytoscape

Tool to construct and analyse interaction graphs with attributes and features.

Example: Halobacterium 929 proteins with 5022 inferred from

- (1) Domain-fusion interactions
- (2) Phylogenetic interactions
- (3) Inferred protein-protein interactions from yeast two-hybrid data

Simulated annealing to search for “significant” subgraphs

Cytoscape

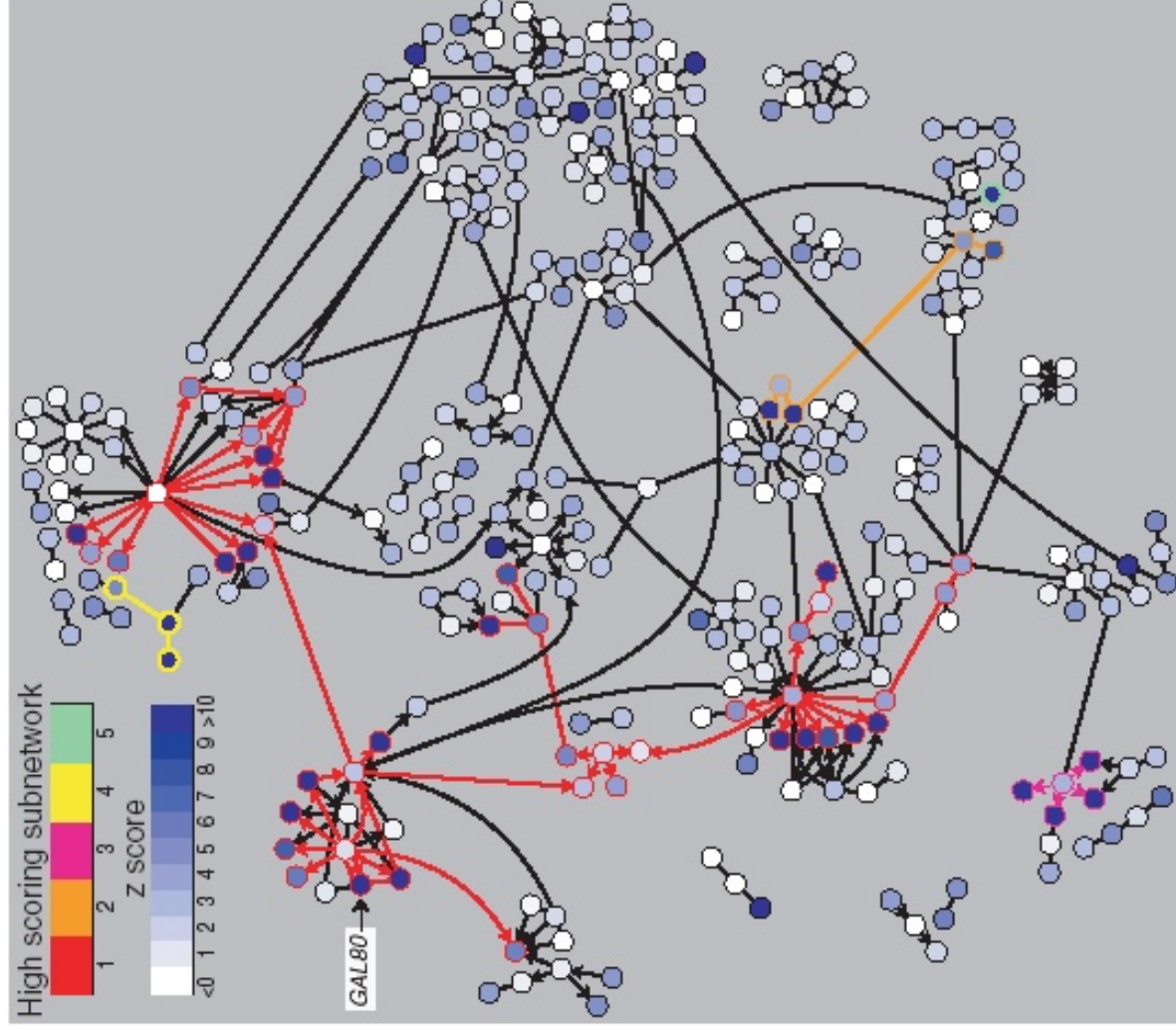


Fig. 1. Performance on a small molecular interaction network. Nodes represent genes, an edge directed from one node to another signifies that the protein encoded by the first gene can influence the transcription of the second by DNA binding (protein→DNA), and an undirected edge between nodes signifies that the corresponding proteins can physically interact. Z-scores (blue scale) indicate the likelihood of differential expression of each gene in a *GAL80* knockout experiment. Z-scores were used to search for active subnetworks using our simulated annealing method; the five top-scoring subnets are shown. For further information on this network, including gene labels, see

Conclusion

There is still interesting combinatorics being done for genomics!

Thank you!

Questions?

References

- Dan Gusfield, **Algorithms on Strings, Trees, and Sequences**, Cambridge University Press, 1997.
- Pavel Pevzner, **Computational Molecular Biology**, MIT press, 2000.
- Eugene Myers, *Whole-Genome DNA Sequencing*, IEEE Computational Engineering and Science 3, 1 (1999), 33–43.
- W. James Kent. *BLAT: The BLAST-like Alignment Tool*, Genome Research 12(4) (April 2002) 656–664.
- Liliana Florea, George Hartzell, Zheng Zhang, Gerald M. Rubin, and Webb Miller, *A Computer Program for Aligning a cDNA Sequence with a Genomic DNA Sequence*, Genome Research 8, 9 (1998) 967–974.
- Hohl M, Kurtz S, Ohlebusch E. *Efficient multiple genome alignment*, Bioinformatics 18 Suppl 1 (2002) S312–20.
- Christopher Lee, Catherine Grasso, and Mark F. Sharlow, *Multiple sequence alignment using partial order graphs*, Bioinformatics Vol. 18 no. 3 (2002) 452–464.

References

- Christopher Lee, Catherine Grasso, and Mark F. Sharlow, *Multiple sequence alignment using partial order graphs*, Bioinformatics Vol. 18 no. 3 (2002) 452–464.
- Kimmen Sjölander. *Phylogenomic inference of protein molecular function: advances and challenges*. Bioinformatics 2004 (20) 170–179.
- Michelle L Green and Peter D Karp, *A Bayesian method for identifying missing enzymes in predicted metabolic pathway databases*, BMC Bioinformatics 5 (2004) 76.
- Ideker, T., Ozier, O., Schwikowski, B., and Siegel, A.F. *Discovering regulatory and signalling circuits in molecular interaction networks*. Bioinformatics 18 (2002) S233–S240.