Detecting Functional Regions in DNA with Sequence Comparison Methods: Program Development and Evaluation

A Ph.D. Thesis in Computer Science and Engineering by

Liliana Florea
Penn State University

Overview

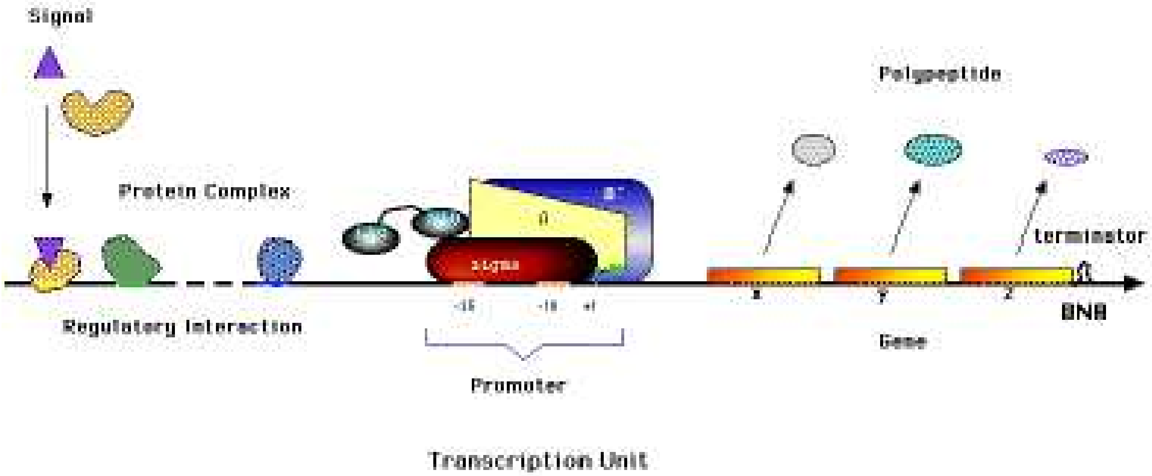
- Biological motivation
- Computational tools (alignments)
- Applications
  - A program for aligning expressed DNA and genomic sequences (*Sim*4)
  - Evaluation methods for programs detecting conserved regions (motifs) in multiple alignments
  - A tool for visualizing annotated multiple alignments of bacterial genomes (*Menteric*)
- Conclusions

Biological motivation

Pathway to gene expression

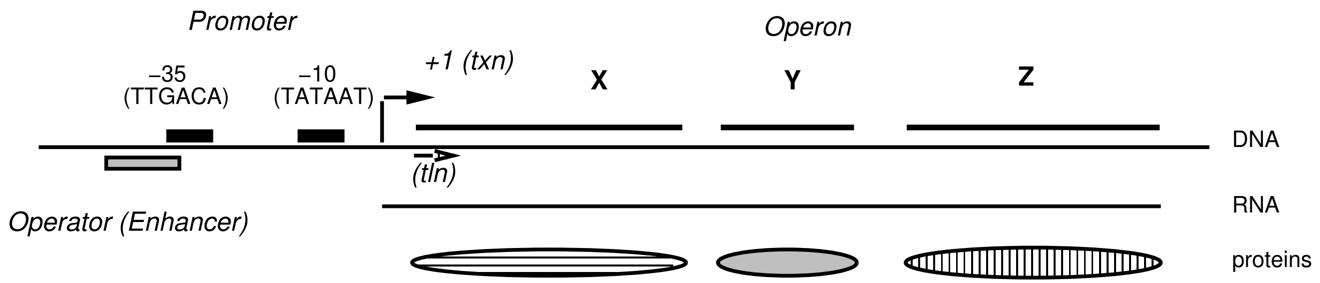
$$\text{DNA (gene)} \overset{transcription}{\Longrightarrow} \text{RNA} \overset{translation}{\Longrightarrow} \text{protein (polypeptide)}$$

Expression of all genes is regulated by *cis*- and *trans*-acting elements.



Prokaryotic transcription

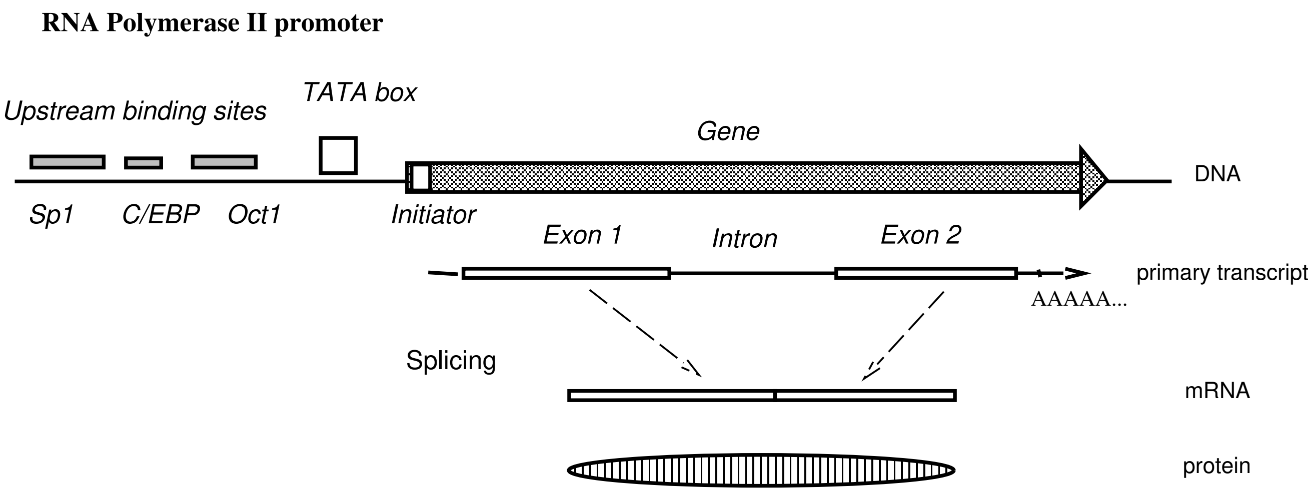
- synthesis by RNA polymerase
- *cis*-acting elements: promoter, enhancers, operators
- *trans*-acting elements: regulatory proteins (activators, repressors)
- genes in operons are transcribed coordinately



Regulatory proteins recognize short (usually) conserved sequences (motifs) in DNA.

Eukaryotic transcription

- synthesis by an RNA polymerase
- more complex regulatory apparatus (promoter, transcription factors)
- genes are interrupted (exons are more conserved, introns vary)
- presence of repetitive elements



Computational tools

DNA sequences: ACAGGACT..... $\in \{A,C,T,G\}^*$

Alignment: a mapping between the letters of two (or more) sequences, with possible spacers to equalize the sequence lengths.

Pairwise alignment:

```
  10 CC-CCTCTGGAGGACCCCCGATGACGCTGCCGATACCGTCC  ...
3356 ..G..G........-..T.............A.........  ...
```

Multiple alignment:

```
4067:  AAACCTGGGAGGATGCGGCTGAGTTTTT-TTGCTCGG  ...
  21:  .........-A..C..T..C..A...C.GC....G..  ...
1021:  ..........A..C..T..C..A...C.GC....G..  ...
```

Central paradigm

Selection lowers the rate of mutation in functional DNA, and hence such regions tend to be resistant to evolutionary drift. Hence, functional regions are better conserved than random DNA.

Applications in computational biology

Pairwise alignments

- used to locate regions of homology between two species:
  - protein-coding regions
  - regulatory sites in non-coding regions
- used to study insertion, deletion and gene re-arrangement events, and to infer evolutionary relationships between the genomes

Multiple alignments

- are more reliable indicators of homology
- conserved blocks are strong candidates for regulatory sites

Pathway to genome analysis and annotation using comparative methods

- design algorithms to recognize and align regions of good conservation, customized for the target problem (*blast*) (*Sim4* – Florea et al., 1998)
- evaluate the performance of programs against a set of benchmarks (Florea et al., 2000a)
- provide tools to allow access to related data in new, integrative ways (databases, visualization tools) (*Menteric*; Florea et al., 2000b)

Sim4 - A program for aligning expressed DNA and genomic sequences

Stages and implementation techniques:

I. Detect basic homology blocks ('exon cores')

- *'blast'*-like homology search to construct a list of gap-free matches
- sparse dynamic programming to connect the matches into larger blocks

II. Approximate the introns: extend/trim the 'exon cores' to eliminate gaps or overlaps in the cDNA sequence

- *greedy* sequence comparison algorithms to extend the exons
- *'blast'*-like homology search tuned for higher sensitivity to find new 'exon cores'

III. Refine the introns: detect the splice junctions

- dynamic programming to compute a combined measure of the accuracy of the alignment and the intensity of splice signals at the ends of the intron

IV. Generate the alignment

- greedy alignment algorithms

(Please refer to the slide genview.ps.)

I: Detection of basic homology blocks ('exon cores')

A. Search for gap-free matches with a '*blast*'-like homology search (*blast*, Altschul et al. 1990)

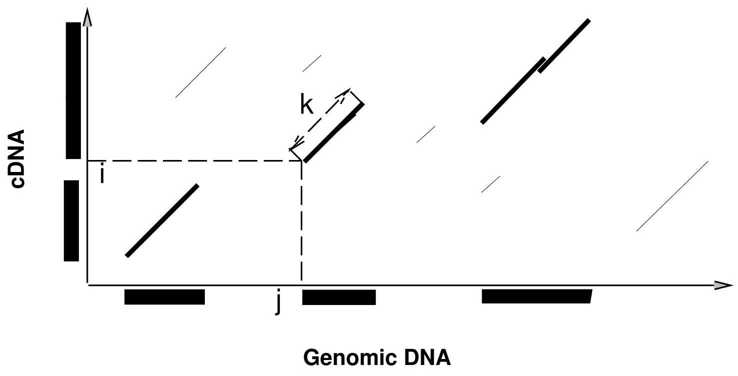
- Detect all exact matches of words of size $W = 12$ (12-mers) between the sequences
  - encode each W-mer as a $2W$-bit binary word, e.g.:

    $$A \rightarrow 00,\ C \rightarrow 01,\ G \rightarrow 10,\ T \rightarrow 11$$
    $$ATGAC\ldots \longrightarrow 0011100001\ldots$$
  - use open hashing to store all W-mers in the shorter sequence in a hash table indexed by the first 15 bits of a code word (2^{15} entries)
  - scan the longer sequence for exact word matches

- Extend the exact matches to obtain gap-free <u>M</u>aximal <u>S</u>coring <u>P</u>airs (MSPs). Algorithmic constraints:
  - substitution scores (1 for a match, -5 for a mismatch)
  - threshold for terminating word extensions ($X = 12$)
  - score threshold for MSPs ($K = 16$)

B. Link selected MSPs to generate exon cores

- Sort the MSP list by the position in the cDNA sequence
- Select a maximal scoring chain using sparse dynamic programming
- Merge ‘close’ consecutive MSPs into exon cores, and separate ‘distant’ consecutive MSPs by introns



II. Approximate the introns by extending/trimming the exon cores

Two types of situations: consecutive exons may overlap or may enclose unmatched cDNA fragments.

To align the free cDNA fragments:

1. first extend the enclosing exons using greedy sequence comparison algorithms, up to a cost C (e.g., .2*length of the cDNA fragment)
2. if this fails, find new exon cores with the '*blast*'-like search tuned for higher sensitivity (smaller W and K), and extend them with greedy methods

Different combination strategies are used when the unmatched cDNA is at the ends or is internal to the sequence.

(Please refer to the slide extensions_composite.ps.)

Extending exon cores with greedy sequence comparison algorithms (Miller and Myers 1985, Ukkonen 1985, Wu et al. 1990, Chao et al. 1997)

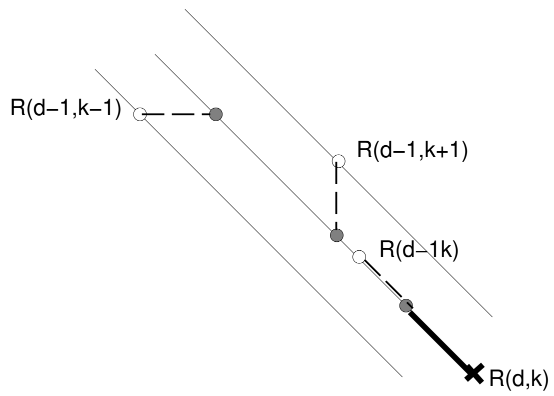
This approach is faster than dynamic programming when comparing highly similar sequences, because it works with a measure based on the differences (*cost*), rather than the similarities, between two sequences.

Let $A = a_1 \ldots a_M$, $B = b_1 \ldots b_N$ be the two sequences being compared. Let $D(i, j)$ be the cost of aligning the prefixes $a_1 \ldots a_i$ and $b_1 \ldots b_j$.

- Fill in the values in the alignment grid by increasing costs $D = 0, \ldots D(M, N)$, starting from the source $(0, 0)$, until the sink (M, N) is reached (or some other exit condition is verified).

		A	G	C	T	A	C	A
	0	1	2	.	.	.	.	.
A	1	0	1	2	.	.	.	.
A	2	1	1	2	.	.	.	.
G	.	2	1	2	.	.	.	.
C	.	.	2	1	2	.	.	.
A	.	.	.	2	2	2	.	.
A	.	.	.	.	.	2	.	.

- For a diagonal k, compute the farthest row that can be reached from the source with cost d, $R(d,k)$.



$$R(d,k) = slide(k, \max\{\overbrace{R(d-1,k-1)}^{right}, \overbrace{R(d-1,k)+1}^{diagonal}, \overbrace{R(d-1,k+1)+1}^{down}\})$$

- At cost d, only diagonals between $-d$ and d are inspected. Eventually, only points in the band between diagonals $-D(M,N)$ and $D(M,N)$ are analyzed.
- Time complexity: $\mathcal{O}(MD(M,N))$; space requirements: $\mathcal{O}(M+N)$

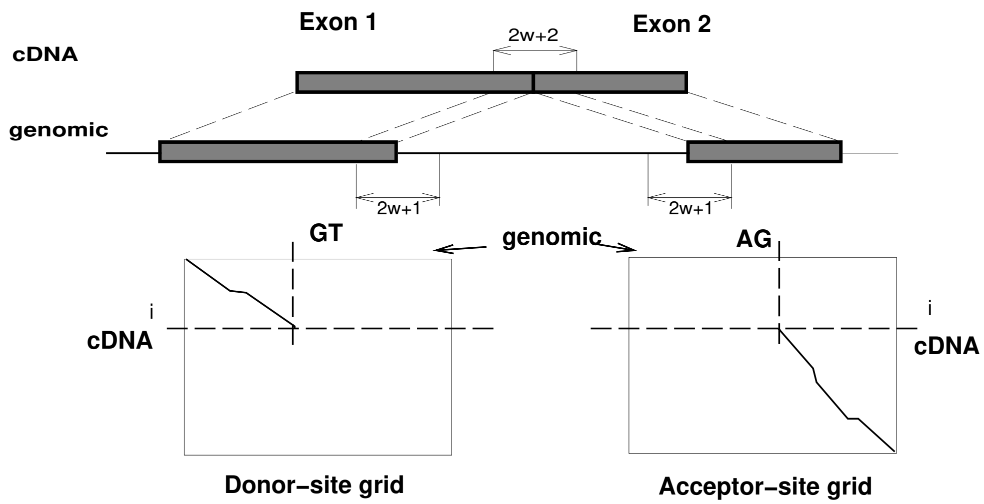
III. Refine the introns to best approximate the splice junctions

For each predicted intron, align a 14 bp cDNA window surrounding the cDNA splice position with each of the 13 bp regions centered at the beginning and at the end of the intron. Select a splice configuration that maximizes a global *splice score*.

$$Score_x(i,j) = \overbrace{\frac{X(i,j)}{X(i,j)+D(i,j)}}^{alignment\ accuracy} + \overbrace{Splice_x(j)}^{intensity\ of\ splice\ signals}$$

where: $D_x(i,j)$ = cost of an optimal alignment ending at (i,j) in the grid x

$X_x(i,j)$ = number of matches in an optimal alignment ending at (i,j)



IV. Generate the global alignment

- align each exon using a linear space implementation of a greedy alignment algorithm (Chao et al. 1997; Myers 1986)
- connect individual alignments by gaps ('introns')

Performance evaluation

- Speed: it can be further optimized to allow more efficient EST database searches
- Accuracy:

rate	1	2	3
1%	0.15	0.27	88.9
3%	0.20	0.19	81.3
5%	0.26	0.48	75.2

rate	1	2	3
0.1%	0.030	0.030	98.3
1%	0.057	0.058	94.8

a. With simulated ESTs b. With simulated full-length mRNAs.

(1) % of nucleotide false positives (FP)

(2) % of nucleotide false negatives (FN)

(3) % of splice junctions that were determined exactly

(Data published in *Genome Research* 8(9), 1998, refer to the original version.)

Evaluation methods for programs detecting conserved regions in multiple alignments

Programs: *infocon*, *phylogen*, *agree*, *kkno*, *kunk* [Nick]

Evaluation: How closely do the predictions match the known functional landmarks?

Calibration: What are the best parameter values for the programs?

Elements of the method

- A collection of reference data (known functional sites in a regulatory region)
  - sources: literature, databases (GenBank, RegulonDB, TRRD)
  - difficulties: inconsistencies among different sources, and incompleteness due to lack of experimental investigation
  - reference regions: HS2, HS3, and the HBB promoter at the β-globin locus; the bacterial *araC*-*araBAD* intergenic region

- Quality indicators (*cost* functions)
  - *False negative* = functional position in the reference sequence (e.g., human, *E. coli*) not reported by the program
  - *False positive* = position in the reference sequence that was reported by the program, but is not included in the reference set
  - aggregate measures – e.g., FN+FP, CC, SMC, ACP

(Please refer to the table tab.prn.)

Menteric

A Web-Based Visualization Tool for Bacterial Genome Alignments

Description

The Menteric *web server* computes and displays on-the-fly *nucleotide-level* multiple alignments of sequences from several related bacterial genomes in a 1 Kb region surrounding a user-specified address in the *E. coli* sequence.

The alignment is rendered as a PostScript or PDF document in which annotations of *ORFs (genes)*, *promoters* and *protein binding sites* are color coded.

(Please refer to the slide ara_slide.ps.)

Graphical features

Color underlays

- Are used to mark known functional regions (green = promoters, red = protein binding sites, light chocolate = ORFs)
- Annotation data were obtained from two sources:
  - the GenBank version M54 of the annotated *E. coli* sequence (Blattner et al., 1997)
  - the web-based RegulonDB database (Huerta et al., 1998; Salgado et al., 2000);

Annotation labels

- Used to mark locations of interest in the *E. coli* sequence (e.g., transcription start sites, names of binding proteins)

Boxes around conserved regions

- The user can select from five different techniques for detecting the conserved regions. [Nick]

  A ‘conserved’ region is one of high information content (*infocon*), good letter agreement (*agree*), low phylogenetic distance (*phylogen*) or in which each row of the alignment differs in at most one position from either an a priori specified (*kkno*) or an unknown ‘center’ sequence (*kunk*).

- The parameter values for each method were calibrated against the reference functional sites in the bacterial *ara*BAD-*ara*C intergenic region.

- In the example shown, the conserved regions were determined based on the information content measure.

Hyperlinks to various related web resources

- Labels in the PDF document contain transparent pdfmark links to other web resources:

  - gene labels – links to the gene section of the WIT database;
  - regulatory site labels – links to the GenBank *E. coli* annotation file, or to the query page of RegulonDB;

- or are used as a mechanism for retriving contig sequence data:

  - 'trailer' labels – transmit parameter values (organism, contig name, start and end positions of the aligned fragment) to a query interface for downloading contig data

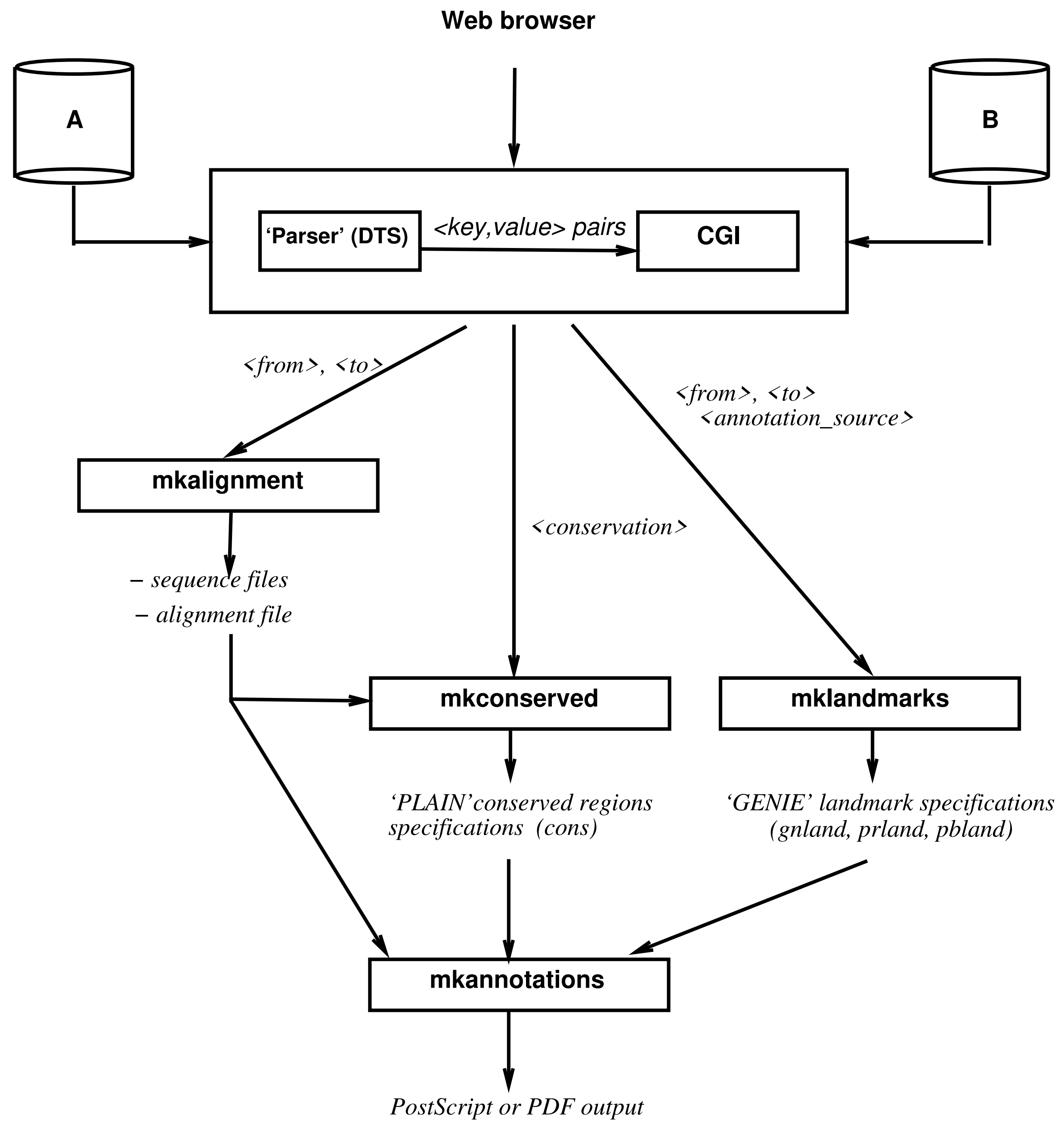
Structural features

- It interfaces with the web browser through a CGI script.
- It has a built-in mechanism that allows extension by addition of new data, and portability to other model genetic systems.
- It contains four internal modules, one for each type of element to compute and display (mkalignment, mkconserved, mklandmarks), and a final annotation module (mkannotations).
- It formats the output as either a PDF or a PostScript file.
- It accesses local repositories of annotation, alignment and sequence data.

Programming language support

C, Perl, Kornshell

High–Level View of Menteric's Architecture



A – sequence data files, files containing genome–wide pairwise alignments

B – data files containing GenBank and RegulonDB annotations

Module 1: *mkalignment*

Given an *E. coli* address, mkalignment extracts sequences from each of the other bacteria that align with the *E. coli* sequence in a 1 Kb region surrounding that position.

A set of genome-wide pairwise alignment files, pre-computed using a sequence alignment program (e.g., *blastz* [Zheng], *blast*, *MUMmer*), are used to determine the matching fragments.

A multiple alignment of the segments extracted is generated with a locally developed program (*align*5, [Zheng]).

Local data repositories

- sequence (contig) data files
- pairwise alignments of *E. coli* with each of the other bacteria
- 'seqindex' files containing contig positions in the sequence files
- index files of alignment positions in the large alignment files, containing:
  - *start* and *end* positions of the alignment in the *E. coli* sequence
  - alignment *score*
  - locations of the header and alignment records in the alignment file
  - name of aligning *contig* in the other bacterial sequence

Features:

- fast indexed access to the alignment and sequence data
- selection of the (unique) 'homolog' based on simple score examination

Module 2: ***mkconserved***

Employs one of five methods for detecting conserved regions in multiple alignments (*infocon, phylogen, agree, kkno, kunk*; [Nick]), according to the conservation criteria selected by the user.

Features:

- dynamic adjustment of the phylogenetic tree (for *phylogen*)
- clustering of overlapping conserved regions (for *kunk*, *kkno*)

Module 3: ***mklandmarks***

Extracts known functional sites in a specified region from local files containing annotation data extracted from GenBank and RegulonDB. Creates three files containing annotation data for genes (*gnland*), promoters (*prland*) and protein binding sites (*pbland*).

Module 4: ***mkannotations***

Creates the graphical representation of the elements (nucleotide-level alignment, conserved elements and known functional landmarks) as a PostScript or PDF file.

The main software component (*maps*) has a built-in parser which recognizes a feature specification language (FSL) [Nick]. The FSL allows one to describe the graphical components of the model, such as:

- the class of symbols used for annotations (boxes, underlines, underlays),
- options for displaying and formatting the text, boxes, underlines, underlays,
- source files containing the annotation data, and
- source containing the multiple alignment.

Example of FSL specification:

```
load file "ali" alias "eco",
"spa" link "/cgi-bin/xcontig/query?bug=SPA&cg=>B_SPA.Contig6518&idx=6518&b=796&e=2200",
"sty" link "/cgi-bin/xcontig/query?bug=STY&cg=>Contig308&idx=308&b=15630&e=17036";

landmarks from "gnland" underlayed color lightchocolate;
landmarks from "prland" underlayed color lightgreen;
landmarks from "pbland" underlayed color lightred;
annotations from "cons" box mark style solid;

no description;
page orientation portrait;
page width 100;
page spacing 1;
textfont 7;

top 1 tickmark every 10;
text all banner font "Courier" 8  dotted pivot 1 trailer font "Bold" 8 ended;

output postscript;
```

A *meta*-framework for specifying a class of alignment visualization tools

Characteristics

- *simple* and *secure*, but *rich* enough to capture all aspects of a large category of genetic systems
- *parsimonious* – i.e., allows code sharing by parameterizing the modules
- *extensible* – e.g., by introducing an upper level of abstraction

Components

- a collection of Data Assembly Tools (DAT)
  - generate the organism-specific infrastructure (e.g., index files, some annotation data)
- a Dynamic Tool Selector (DTS)
  - interfaces with the web browser and the CGI
  - selects an organism specific tool-description file to instantiate the system

Examples of configuration file for a collection of two sub-systems (A), and organism-specific tool description file (B)

(A)

```
#:GENSYS:
BEGIN
     model = 'bacteria'
     start = 1
     stop = 4639221
     annotations = 'GenBank|RegulonDB'
     prefix = ""
END
```

```
BEGIN
    model = 'hoxd'
    start = 1
    stop = 175667
    annotations = 'entrez'
    prefix = "hoxd."
END
```

(B)

```
export DER_DATA; DER_DATA=/usr/local/align/data/enteric/derived
export BAS_DATA; BAS_DATA=/usr/local/align/data/enteric/basic

export PRINC_SEQ;       PRINC_SEQ=ecoli
export PRINC_SPECIES;   PRINC_SPECIES="E.coli"
export PRINC_ALIAS;     PRINC_ALIAS="eco"
export ALIAS_FILE;      ALIAS_FILE="sequence.alias"
export PHYL_TREE;       PHYL_TREE="menteric.tree"
export AN_SRC_FILE;     AN_SRC_FILE="annot.sources"
export KVALUE;          KVALUE=270
```

(Please refer to the slide status.ps.)

Thanks:

Webb Miller   Ross Hardison   Piotr Berman

Cathy   Scott   Nick   Zheng