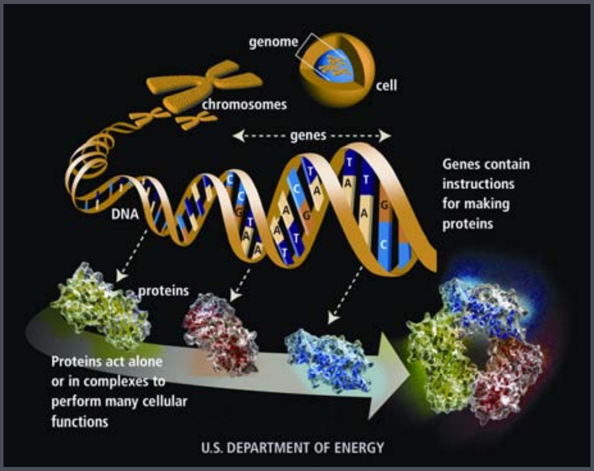
Towards Declarative and Efficient Querying on Protein Structures

Jignesh M. Patel
University of Michigan

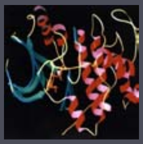
Biology 101


genome
cell
chromosomes
genes
DNA
Genes contain instructions for making proteins
proteins
Proteins act alone or in complexes to perform many cellular functions
U.S. DEPARTMENT OF ENERGY

Data Types

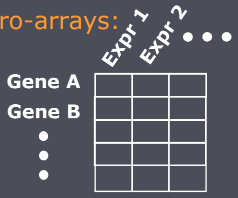
Sequences: AGCGGTA….

Structure:



Interaction Maps:

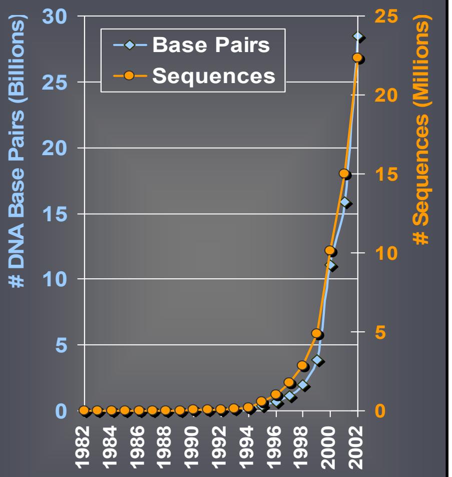
Micro-arrays:



When there is something to do in the cell, it is a protein that *does* it.

Role of DBMS in Bioinformatics

- Large Data Sets
  - Growing exponentially!
- Data Types
  - Sequences/arrays/3-D/text
- Complex Queries
  - Ad-hoc tools today
  - Integrated querying done using procedural methods
- Scalability/Parallelism
  - Home-grown techniques often used today
- Need: Declarative and efficient querying



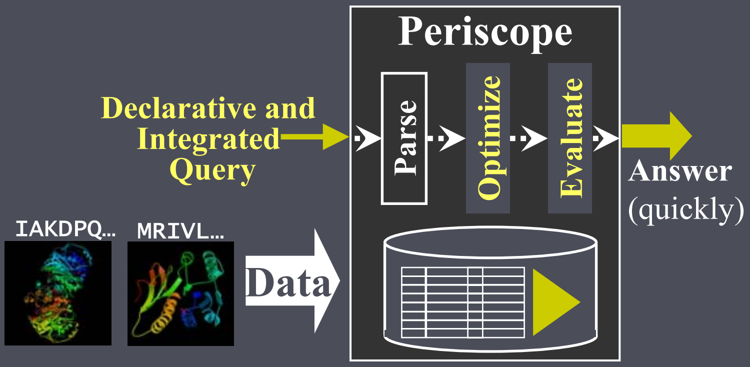
Source: GenBank

Protein Querying

- Protein functionality determined by
  - Sequence composition
  - Geometric structure
- Need to query across <u>all</u> structures
- Current methods
  - Non-declarative tools, often not very efficient on large data sets
  - Support querying on only one structure

Periscope

Goal: Design, implement, and evaluate a database management system for <u>declarative</u> and <u>efficient</u> querying on <u>all</u> protein structures



Roadmap

- Background and Introduction
- Primary Structure Sequence Matching
- Querying on Secondary Structure
- PiQA: Integrated Query Algebra
- Summary

Sequence Matching

- Find similar sequences
  - Given a protein sequence, find homologous matches in the database
  - Similarity based on "local similarity"
  - A local-alignment algorithm
  - Operations: Replace, Delete, Insert
  - Score using a substitution matrix

Database: THE TRAIN DRIVER'S CABIN

Query: DRAIN

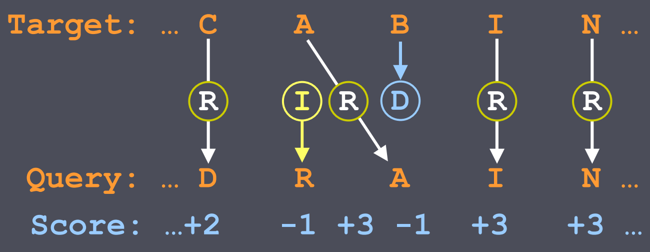
Sequence Matching

- Find similar sequences
  - Given a protein sequence, find homologous matches in the database
  - Similarity based on "local similarity"
  - A local-alignment algorithm
  - Operations: Replace, Delete, Insert
  - Score using a substitution matrix

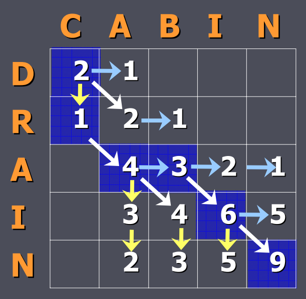
Target

Query

	-	A	B	C	D
-	-1	-1	-1	-1	-1
A	-1	3	0	0	0
B	-1	0	3	0	0
C	-1	0	0	3	0
D	-1	0	0	2	3



Smith-Waterman



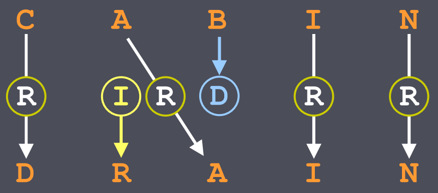
S-W Matrix: G

Target

Query	-	A	B	C	D
-	-1	-1	-1	-1	-1
A	-1	3	0	0	0
B	-1	0	3	0	0
C	-1	0	0	3	0
D	-1	0	0	2	3

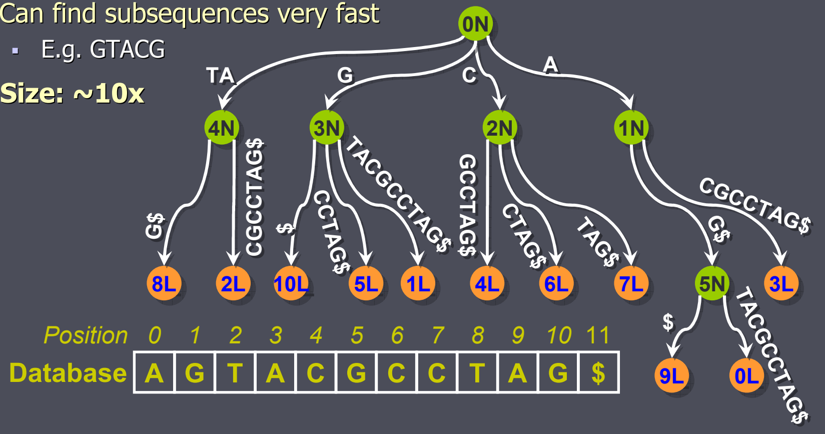
Substitution Matrix

$$G_{i,j} = \max \begin{cases} 0 \\ G_{i-1,\,j-1} + S(q_i \rightarrow t_j) \\ G_{i-1,\,j} + S(q_i \rightarrow -) \\ G_{i\,,\,j-1} + S(- \rightarrow t_j) \end{cases}$$



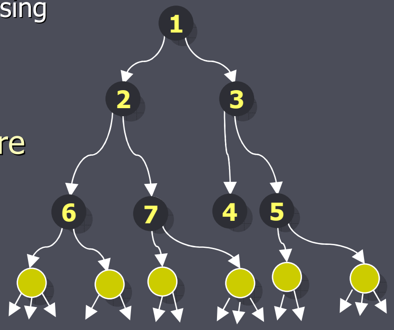
Suffix Trees

- Compact Patricia trie
  - Every suffix has a path (to leaf)
  - Every subsequence is a prefix of a path
- Can find subsequences very fast
  - E.g. GTACG
- **Size: ~10x**



OASIS

- Search driven by the suffix tree
  - Fill up the S-W columns by traversing down the suffix tree
- Best-first: expand node in the tree with highest expected score
  - Expected Score = current score + best possible score for unconsumed portion of query
  - Guarantees **<u>online</u>** behavior!
- Exploits redundancy in the database

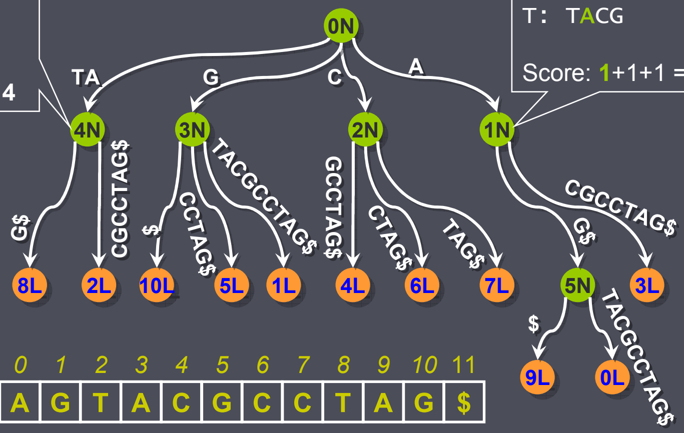


OASIS

- Query: TACG
- Unit Edit Distance Matrix: Same Symbol Substitution = 1, else -1

```
Q: .TA...
T:  TACG

Score: 2+1+1 = 4
```

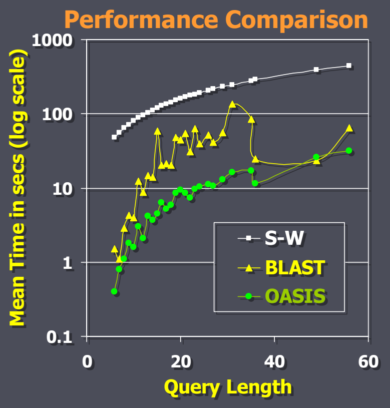
```
Q: .A...
T: TACG

Score: 1+1+1 = 3
```

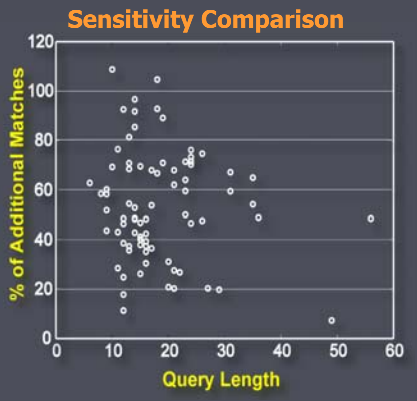


Experimental Setup: Comparison

- Data set: swissprot
  - 110K proteins, # symbols: 40M, data size: 40MB
  - Index Size: 500MB (~12.5 bytes per symbol)
- Workload: 100 queries from ProClass motif database
  - Short queries – lengths 6 - 56, avg len = 16
- PAM30 scoring matrix
- BLAST parameters (short query settings, 5-15 residues)
  - E=20,000, word size = 2
- OASIS: minScore = $\left\lceil \frac{\ln(Kmn) - \ln(E)}{\lambda} \right\rceil$
- Platform: 1.7GHz Xeon, Linux, 256MB buffer pool, 2K page size, clock replacement policy

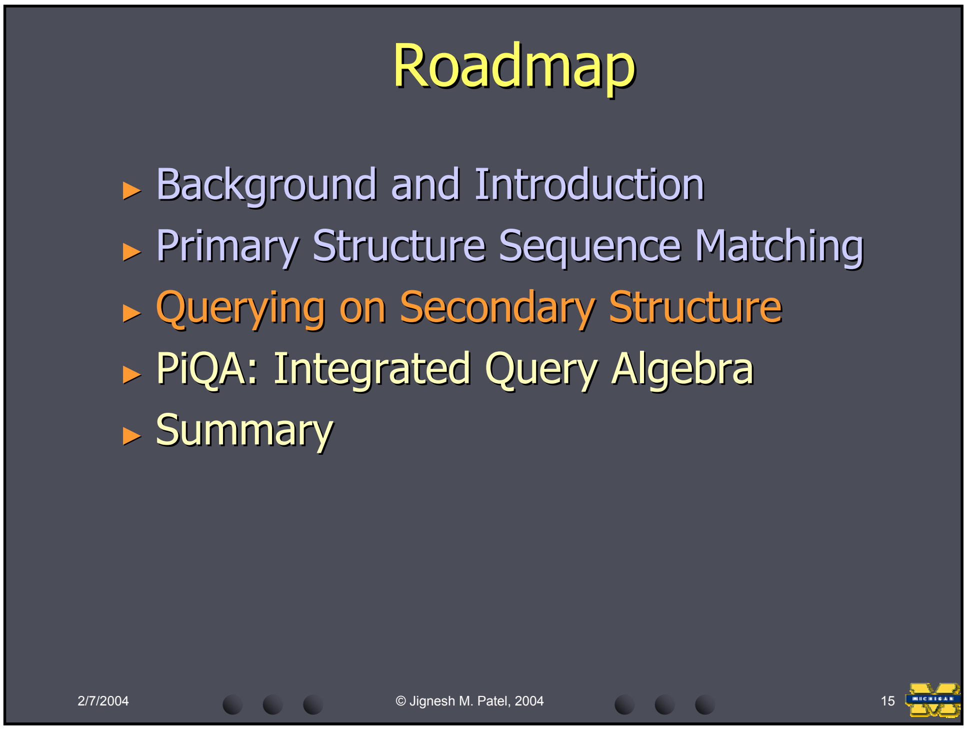
Experimental Results





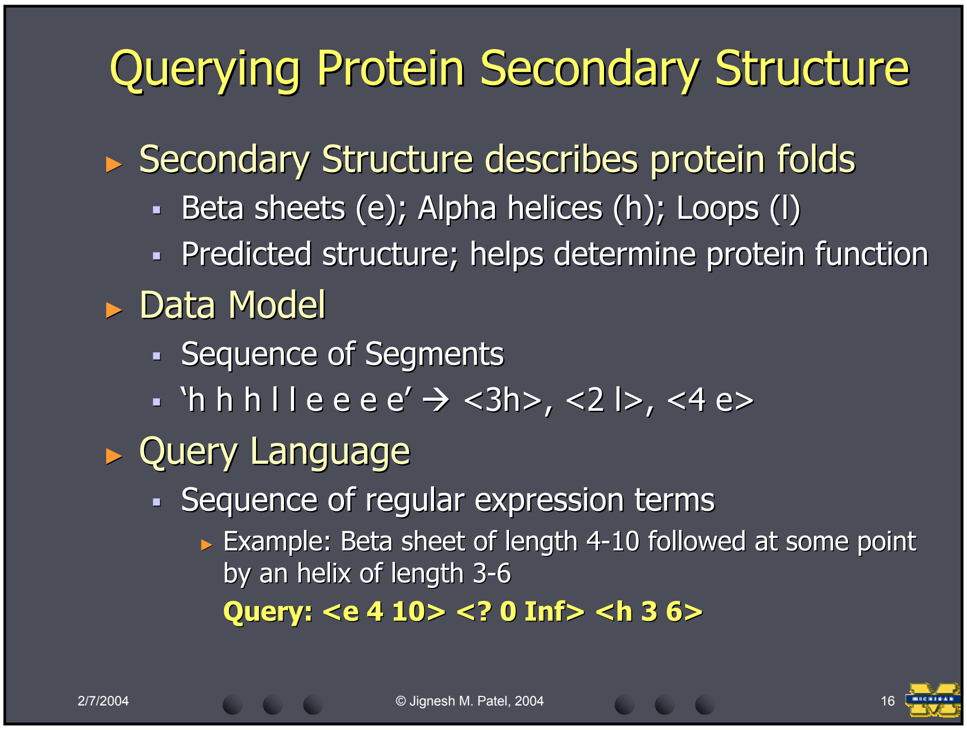
- OASIS 1-2 orders of magnitude faster than S-W, and usually also faster than BLAST
- OASIS retrieved ~ 60% more matches than BLAST

Lots more in our VLDB'03 paper



Roadmap

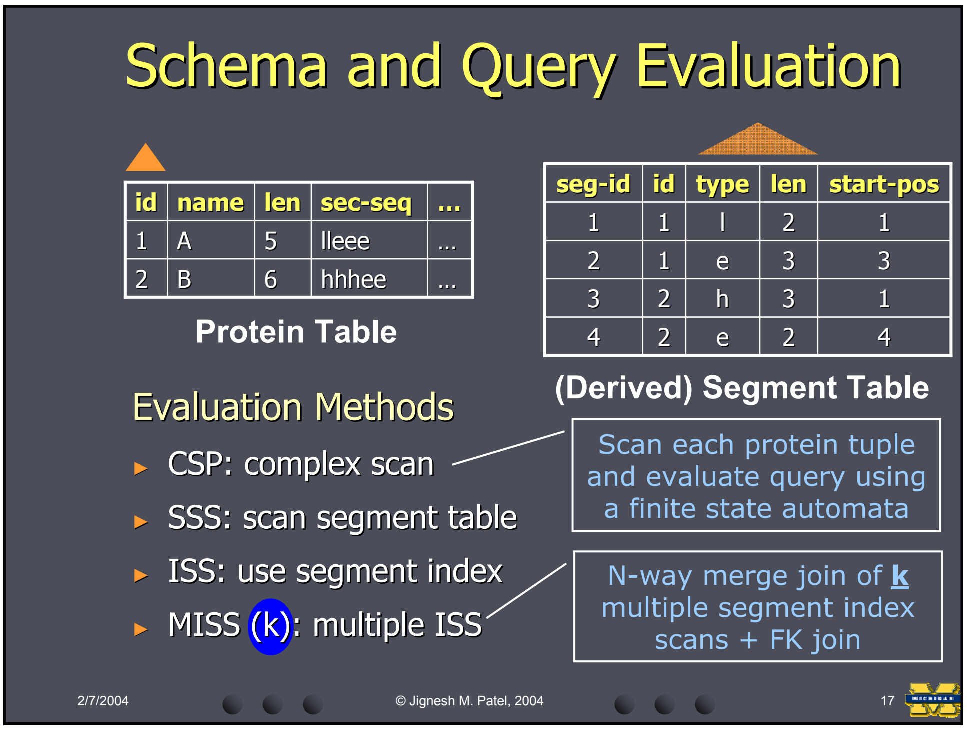
▸ Background and Introduction
▸ Primary Structure Sequence Matching
▸ Querying on Secondary Structure
▸ PiQA: Integrated Query Algebra
▸ Summary



Querying Protein Secondary Structure

- Secondary Structure describes protein folds
  - Beta sheets (e); Alpha helices (h); Loops (l)
  - Predicted structure; helps determine protein function
- Data Model
  - Sequence of Segments
  - 'h h h l l e e e e' → <3h>, <2 l>, <4 e>
- Query Language
  - Sequence of regular expression terms
    - Example: Beta sheet of length 4-10 followed at some point by an helix of length 3-6

      Query: <e 4 10> <? 0 Inf> <h 3 6>



Schema and Query Evaluation

id	name	len	sec-seq	…
1	A	5	lleee	…
2	B	6	hhhee	…

Protein Table

seg-id	id	type	len	start-pos
1	1	l	2	1
2	1	e	3	3
3	2	h	3	1
4	2	e	2	4

(Derived) Segment Table

Evaluation Methods

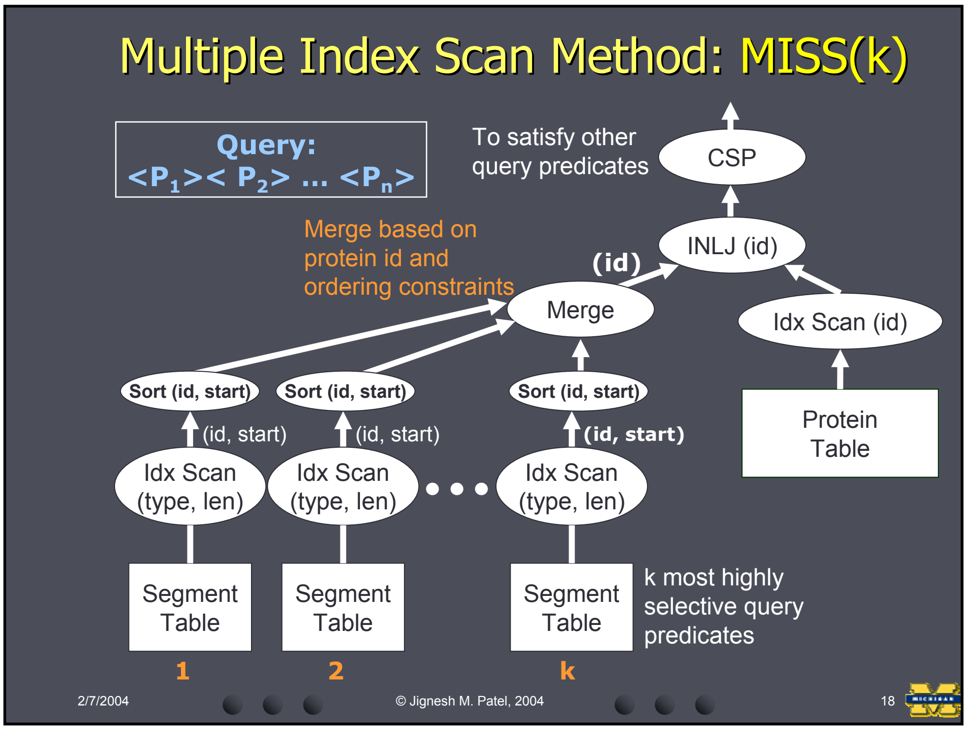
- CSP: complex scan
- SSS: scan segment table
- ISS: use segment index
- MISS (k): multiple ISS

Scan each protein tuple and evaluate query using a finite state automata

N-way merge join of **k** multiple segment index scans + FK join

2/7/2004 © Jignesh M. Patel, 2004 17

Multiple Index Scan Method: MISS(k)



2/7/2004 © Jignesh M. Patel, 2004 18

Query Optimizer

- Chooses best method for given query
- Optimization:
  - Requires estimates of query predicate selectivities and result cardinality
  - Cost model: CPU and IO
- Estimation: Two types of histograms
  - Basic: segment predicate selectivity
  - Complex: query selectivity

Basic Histogram

- Estimates selectivity of individual query predicates
- 2-D array
  - Dimensions : Length, Fold type
  - Value: number of each <type,len> segment
- Example
  - Pred <e 4 4>, Est: 52
  - Pred <e 2 4>, Est: 35+45+52

Len	H	E	L
1	20	10	18
2	23	35	25
3	36	45	33
4	44	52	35
…			
50	4	2	0
-	1	0	0

Complex Histogram

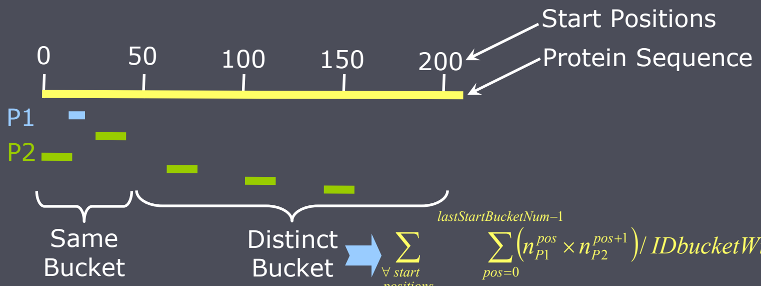
- Estimates **result** cardinality
- Four-dimensional structure
  - Protein id (equi-width buckets)
  - Start position (equi-width buckets)
  - Length (1 – 50)
  - Type ('e', 'h', 'l')

  } Same as in the basic histogram
- Example: Position [x] [y] [z] ['e']
  - holds the number of <e z z> segments whose starting position is in the range of the yth bucket and whose protein id lies within the xth bucket range

Complex Histogram

4 Dimensions
- Protein id
- Start pos
- Length
- Type

- Query: {<P1> <P2>}
  - Compute selectivity of query result



$$\sum_{\forall\ start\ positions} \sum_{pos=0}^{lastStartBucketNum-1} \left(n_{P1}^{pos} \times n_{P2}^{pos+1}\right) / IDbucketWidth$$

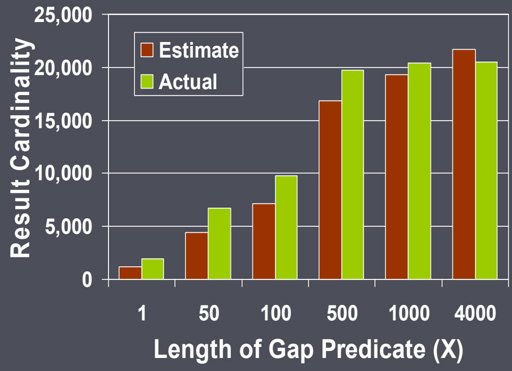
$$\sum_{\forall\ start\ positions} \int_{sp=0}^{startBucketWidth} \left(n_{P1}^{sp} \times n_{P2}^{sp+1}\right) / IDbucketWidth$$

Experimental Evaluation

- Techniques implemented in Periscope
  - Configuration: Using the SHORE storage manager, 64MB buffer pool size, 16K page size
- Also implemented in a commercial ORDBMS
  - Used type extensibility to create array-like data types for sequences & user-defined functions for FSM
- Data Set: PIR
  - 250K protein tuples, 250MB
  - Segment Table: 10M tuples, 355MB
- Platform: 1.7GHz Xeon, Linux, 40GB SCSI disk

Complex Histogram Accuracy

- Query: {<l 15 15> <? 0 X> <h 24 24>}
- Experiment: Vary Gap Predicate



Histogram Details

Data Set: PIR

100x100x50x3

Size: 5.8MB

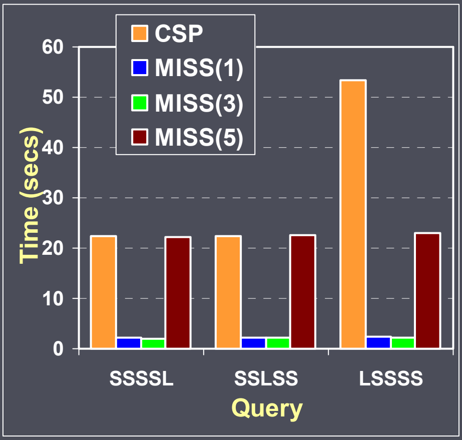
Build time: 22 secs

Estimation time: 20 ms

Histogram accurate within 80% of the actual result

Query Performance

- Query: 9 Predicates
  P_1 G_1 P_2 G_2 P_3 G_3 P_4 G_4 P_5
  P=predicate, G=Gap pred.
- Expt: Vary Selectivity of P's
  Alternatives: S (0.3%), L (7%)



- Choice of algorithm is critical
- CSP sensitive to position of **L** pred.
- Choice of k in MISS critical
  Index Probe
    + reduce # protein tuples fetched
    - probe cost, sorting and merging
  Influenced by query and predicate selectivity

More details in our VLDB'02 paper

Roadmap

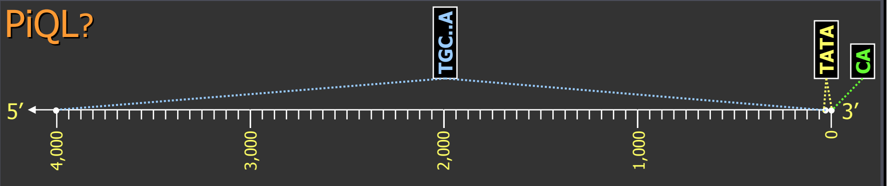
- Background and Introduction
- Primary Structure Sequence Matching
- Querying on Secondary Structure
- PiQA: Integrated Query Algebra
- Summary

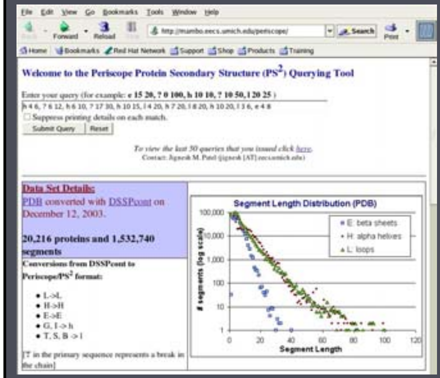
PiQA

Details in our SSDBM'03 paper

- An algebra in PNF for queries on primary & secondary structures
- Examples:
  - Match the primary sequence "AAANBPPPPSDF", but ignore mismatch in the segment "NBPPP" if it is on a loop
    (P.p * "AAA") || ((P.p * "NBPPP") U (P.s * <L 5 5>)) || (P.p * "PSDF")
  - BLAST: Match n-grams + match extension + transitive closure
  - Find all occurrences of 'TGCTGACTCAGCA' within 4000 bps upstream of a 'CA' with 'TATA' 25-30 bps upstream of the 'CA'
    $M * \text{'TGCTGACTCAGCA'} ||_{3957} M * \text{'TATA'} ||_{26} M * \text{'CA'}$

PiQL?




File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://mambo.eecs.umich.edu/periscope/
Search Print
Home Bookmarks Red Hat Network Support Shop Products Training
Submit Query Reset
Segment Length Distribution (PDB)
E: beta sheets
H: alpha helixes
L: loops
# segments (log scale)
Segment Length
100,000
10,000
1,000
100
10
1
0
20
40
60
80
100
120

Welcome to the Periscope Protein Secondary Structure (PS^2) Querying Tool

Enter your query (for example: **e 15 20, ? 0 100, h 10 10, ? 10 50, l 20 25**)

h 4 6, ? 6 12, h 6 10, ? 17 30, h 10 15, l 4 20, h 7 20, l 8 20, h 10 20, l 3 6, e 4 8

☐ Suppress printing details on each match.

To view the last 50 queries that you issued click here.
Contact: Jignesh M. Patel (jignesh [AT] eecs.umich.edu)

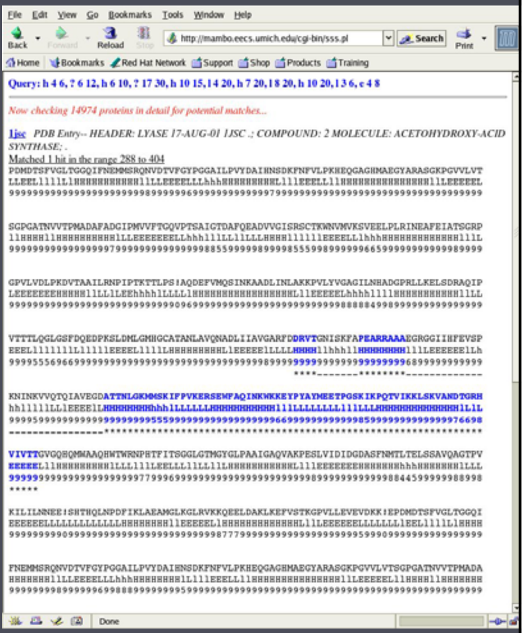
Data Set Details:
PDB converted with DSSPcont on December 12, 2003.

20,216 proteins and 1,532,740 segments

Conversions from DSSPcont to Periscope/PS^2 format:

- L->L
- H->H
- E->E
- G, I -> h
- T, S, B -> l

[? in the primary sequence represents a break in the chain]


File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://mambo.eecs.umich.edu/cgi-bin/sss.pl
Search Print
Home Bookmarks Red Hat Network Support Shop Products Training
Done

Query: h 4 6, ? 6 12, h 6 10, ? 17 30, h 10 15, l 4 20, h 7 20, l 8 20, h 10 20, l 3 6, e 4 8

Now checking 14974 proteins in detail for potential matches...

1jsc *PDB Entry-- HEADER: LYASE 17-AUG-01 1JSC .; COMPOUND: 2 MOLECULE: ACETOHYDROXY-ACID SYNTHASE; .*

Matched 1 hit in the range 288 to 404

```
PDMDTSFVGLTGGQIFNEMMSRQNVDTVFGYPGGAILPVYDAIHNSDKFNFVLPKHEQGAGHMAEGYARASGKPGVVLVT
LLEELlllllLlHHHHHHHHHHHHllLLEEEELLLhhhHHHHHHHHHLlllEEELLllHHHHHHHHHHHHHHHHHllLEEEEEL
99999999999999999999999989999996999999999999997999999999999999999999999999999999


SGPGATNVVTPMADAFADGIPMVVFTGQVPTSAIGTDAFQEADVVGISRSCTKWNVMVKSVEELPLRINEAFEIATSGRP
llHHHHllHHHHHHHHHHHlLLEEEEEEELLhhhlllLLllLLLHHHHllllllEEEELLlhhhHHHHHHHHHHHHHHllllL
99999999999999997999999999999999988559999989999855599899999966599999999999989999


GPVLVDLPKDVTAAILRNPIPTKTTLPS!AQDEFVMQSINKAADLINLAKKPVLYVGAGILNHADGPRLLKELSDRAQIP
LEEEEEEEHHHHHlllLLlLEEhhhhlLLLLlHHHHHHHHHHHHHHHHHHHLllEEEEEELhhhhllllHHHHHHHHHHHHllLL
99999999999999999999999999990969999999999999999999999999999988884998999999999999


VTTTLQGLGSFDQEDPKSLDMLGMHGCATANLAVQNADLIIAVGARFDDRVTGNISKFAPEARRAAAEGRGGIIHFEVSP
EEELllllllLLllllllEEEELllllLHHHHHHHHHLlEEEEElLLLLHHHHllhhhllHHHHHHHHlllLEEEEEEElLh
99995556966999999999999999999999999999999999899999999999999999999689999999999999
                                                ****_______*******_____________

KNINKVVQTQIAVEGDATTNLGKMMSKIFPVKERSEWFAQINKWKKEYPYAYMEETPGSKIKPQTVIKKLSKVANDTGRH
hhlllllLLlEEEElLHHHHHHHHhhhlLLLLLLHHHHHHHHHHHHlllLLLLLLLLlllLLLHHHHHHHHHHHHHHHHlLlL
99995999999999999999999555999999999999999999669999999999998599999999999999976698
----------------****************************************************************

VIVTTGVGQHQMWAAQHWTWRNPHTFITSGGLGTMGYGLPAAIGAQVAKPESLVIDIDGDASFNMTLTELSSAVQAGTPV
EEEEELllHHHHHHHHHHlLLLlllLEELLLlLLLllLHHHHHHHHHHHHLlllEEEEEEEHHHHHHHhhhHHHHHHHHlLLL
99999999999999999999977999699999999999999999999999989999999999999884459999998898
*****

KILILNNEE!SHTHQLNPDFIKLAEAMGLKGLRVKKQEELDAKLKEFVSTKGPVLLEVEVDKK!EPDMDTSFVGLTGGQI
EEEEEELLLLLLLLLLLLLLLHHHHHHHHHllEEEEELlHHHHHHHHHHHHHHLllLEEEEEELLLLLLLlEELlllllLlHHHH
99999999099999999999999999999999999999877799999999999999999999995999099999999999999


FNEMMSRQNVDTVFGYPGGAILPVYDAIHNSDKFNFVLPKHEQGAGHMAEGYARASGKPGVVLVTSGPGATNVVTPMADA
HHHHHHHllLLEEEEELLLhhhHHHHHHHHHlLlllEEELLllHHHHHHHHHHHHHHHHllLEEEEELllHHHHllHHHHHHH
99999999899999969988899999999959999999999999999999999999999999999999999999999999
```

Roadmap

- Background and Introduction
- Primary Structure Sequence Matching
- Querying on Secondary Structure
- PiQA: Integrated Query Algebra
- Summary

Summary

- Bioinformatics applications (urgently) need <u>declarative</u> and <u>efficient</u> query processing tools
  - Current procedural methods reminiscent of pre-relational days
- Database researchers have a lot to contribute!
- Periscope is our effort in this direction
  - PiQA: algebraic framework for querying on primary and secondary structures
  - Primary Structure: OASIS
  - Declarative querying on secondary structure: Periscope/PS^2
- Current Status
  - OASIS and Periscope/PS^2 currently (beta-)deployed at UM
  - Under development: "PiQA powered PiQL queries on Periscope"
- http://www.eecs.umich.edu/periscope