Biocomputational Puzzles

Dennis Shasha
Courant Institute, New York Univ
shasha@cs.nyu.edu

Collaborations with labs of
(bio, Duke): Philip Benfey
(bio, NYU): Gloria Coruzzi, Ken Birnbaum,
Laurence Lejay, Peter Palenchar,
Rodrigo Gutierrez
(media lab/infoviz): Crispy, Brad Paley
(Graph matching): Alfredo Ferro, Rosalba Giugno, Alfredo Pulvirenti, Diego Reforgiato
Overview

• Simple Primer in Genomics
• Activist Data Mining
• Visualization tool for multi-experiment data
• A gallery of challenging problems.
• Lessons from successful collaborations.
Genomics – a primer

• Before genomics: isolate and manipulate genes one at a time.
• However, interactions are important (mouse has about the same number of genes as humans)
• Genomics: quantitative view of an entire species. Proteomics: all proteins of a species. Omics – anything described on one or more entire species.
Tool: sequencing

- Sequencing (find all the DNA of a species).
- About 15 million species. We have sequences of about 200.
- What does it buy you? Figure out which genes do what based on “homology” (sequence similarity).
The microarray revolution: GLOBAL gene expression

although noisy in many and complicated ways (sample preparation, dyes, slide spotting, hybridization, image processing…), they allow us to record the simultaneous activity of a very large number of genes, and thus tackle a whole host of questions on gene function, regulation and interactions:

- Which genes are involved in the (complex) reactions to certain stimuli/offenses?
- What are the (complex) genetic signatures of certain diseases, and can they inform disease taxonomies?
- What are the (complex) consequences of knock-outs?
- Gene networks.
Overview

- Simple Primer in Genomics
- **Activist Data Mining**
- Visualization tool for multi-experiment data
- A gallery of challenging problems.
- Lessons from successful collaborations.
New topic: How to do Data Mining?

- Classical approach:
  - Wait for data to appear
  - Find patterns in it.
  - Hope they are actionable.
- Works well when data is pertinent, e.g. Amazon’s other books recommendation, extrapolation of trends.
Activist Data Mining

• Propose initial experiments to explore subspace of some predefined search space
• Evaluate the results
• Propose new experiments, evaluate, propose, evaluate, propose ....
• Iterative and adaptive
Which is Better for Natural Science?

- Classical is obviously right when you have no control over data generation.
- When you do, active data mining (active learning) may work much better.
- Arises naturally when you have a tight collaboration.
Activist Data Mining Philosophy

Passive Approach: Natural scientists do experiments. Computer Scientists help to glean something from it.

Activist Approach: Computer scientists help
(1) Design experiments
(2) Analyze results
(3) Design new experiments based on results
**Activist Data Mining Philosophy (Reminder)**

**Passive Approach:** Natural scientists do experiments. Computer Scientists help to glean something from it.

**Activist Approach:** Computer scientists help
(1) Design experiments
(2) Analyze results
(3) Design new experiments based on results
Our particular methodology: Adaptive Combinatorial Design
Our innovation: applying combinatorial design in an iterative way.
What is combinatorial design? Disciplined sampling. Suppose you are a thief...

**Combinatorial Safe:** 10 switches with 3 settings each. Over 59,000 ($3^{10}$) possible configurations. However there is a certain pair of switches (you don’t know which pair) and a certain pair of values of those switches that will open the safe.

**Illustration:**

<table>
<thead>
<tr>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>S7</th>
<th>S8</th>
<th>S9</th>
<th>S10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>C</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>A</strong></td>
</tr>
</tbody>
</table>

**Challenge:**
Open the safe in as few switch configurations as possible. How many? How to do?
Scientific Goal

We want to describe the factors (e.g. light, carbon, nitrogen....) that determine whether plants will produce critical amino acids and how those factors interact.

Long-term goal: Virtual plant (and later ... frankenfoods)
Design Space

Inputs:  
* Light  
* Starvation to Various Nutrients  
* Carbon  
* Inorganic N (NO3/NH4)  
* Organic N (Glu)  
* Organic N (Gln)

If inputs are take binary values (first approximation)  
6 binary (+/-) inputs = $2^6$ or 64 input combinations (or treatments)

Use 2-factor combinatorial design to reduce number of treatment combinations required to cover the experimental space, assuming that important interactions will have to do with two factors.
“Combinatorial design” finds six conditions to explore every pairwise interaction. Want to discover important factors.

<table>
<thead>
<tr>
<th>EXPT 1 NO PIVOT</th>
<th>LANE</th>
<th>ILLUMIN</th>
<th>STARVE</th>
<th>CARBON</th>
<th>NO3NH4</th>
<th>GLU</th>
<th>GLN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DARK</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>DARK</td>
<td>L</td>
<td>H</td>
<td>0</td>
<td>0</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>LIGHT</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td>H</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>LIGHT</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>LIGHT</td>
<td>L</td>
<td>H</td>
<td>0</td>
<td>H</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>DARK</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Notice: for each pair of input factors and combination of values from those factors, some experiment has that combination, e.g. Light No carbon; Starve No Glu.

After doing this experiment, certain factors suggest themselves as worth further study: Illumination, Carbon (both have significant repressive correlations).
Adapting to Results

Find most important inputs in order to see their effects in more detail.

That is, we focus our search space on those inputs that are likely to exert the most influence over outputs of interest.
Key to activist data mining is adapting to results of experiments already done. Many ways to do this, e.g. Tong, S. & Koller, D. Active Learning for Structure in Bayesian Networks. Seventeenth International Joint Conference on Artificial Intelligence, 863-869 (2001).

Advocates pool-based active learning. Pool of unlabeled instances (don’t know output value). An active learner chooses which instance to query next in hope it will reduce set of possible answers.


Our problem differs: looking for key factors.
Three questions of particular interest

1. Is any single factor so important that its presence determines the outcome regardless of the other contexts? (e.g. Light in context X is repressive compared with Dark in context Y for all X, Y)
2. Is a factor important enough that it has an effect for any particular context? (e.g. for all X, Light in context X is repressive compared with Dark in X)
3. Is a factor consistently important when compared with a fixed background? (e.g. for all X, is Light in context X repressive compared with background?)
**Pivot Design 1: Start with no pivot design**

Create dark and light pairs by just setting Illumin to light and dark respectively.

<table>
<thead>
<tr>
<th>LANE</th>
<th>ILLUMIN</th>
<th>STARVE</th>
<th>CARBON</th>
<th>NO3NH4</th>
<th>GLU</th>
<th>GLN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DARK</td>
<td>Y</td>
<td>0</td>
<td>L</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>2</td>
<td>DARK</td>
<td>N</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td>H</td>
</tr>
<tr>
<td>3</td>
<td>LIGHT</td>
<td>N</td>
<td>0</td>
<td>0</td>
<td>H</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>LIGHT</td>
<td>Y</td>
<td>L</td>
<td>L</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>LIGHT</td>
<td>Y</td>
<td>L</td>
<td>0</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>6</td>
<td>DARK</td>
<td>N</td>
<td>0</td>
<td>L</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
## Pivot Design 2: Dark Design

Exactly the same as no pivot tests but with DARK everywhere. Requires only three more experiments than in no pivot case.

<table>
<thead>
<tr>
<th>EXPT 1 NO PIVOT</th>
<th>LANE</th>
<th>ILLUMIN</th>
<th>STARVE</th>
<th>CARBON</th>
<th>NO3NH4</th>
<th>GLU</th>
<th>GLN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DARK</td>
<td>Y</td>
<td>0</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>DARK</td>
<td>N</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td></td>
<td>H</td>
</tr>
<tr>
<td>3</td>
<td>DARK</td>
<td>N</td>
<td>0</td>
<td>0</td>
<td>H</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>DARK</td>
<td>Y</td>
<td>L</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>DARK</td>
<td>Y</td>
<td>L</td>
<td>0</td>
<td>H</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>DARK</td>
<td>N</td>
<td>0</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
### Pivot Design 3: Light Design

<table>
<thead>
<tr>
<th>EXPT 1 NO PIVOT</th>
<th>LANE</th>
<th>ILLUMIN</th>
<th>STARVE</th>
<th>CARBON</th>
<th>NO3NH4</th>
<th>GLU</th>
<th>GLN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LIGHT</td>
<td>Y</td>
<td>0</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LIGHT</td>
<td>N</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td></td>
<td>H</td>
</tr>
<tr>
<td>3</td>
<td>LIGHT</td>
<td>N</td>
<td>0</td>
<td>0</td>
<td>H</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>LIGHT</td>
<td>Y</td>
<td>L</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>LIGHT</td>
<td>Y</td>
<td>L</td>
<td>0</td>
<td>H</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>LIGHT</td>
<td>N</td>
<td>0</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Exactly the same as DARK tests but with Light everywhere. Again, three more experiments than in no pivot case.

**Important:** First experiment for light = First Experiment for Dark except for Illumination itself. Differs only in pivot. Minimal pair.
A set of well-spaced minimal pairs, differing only in the pivot. Suggests answers for first two questions:

- Is any single factor so important that its presence determines the outcome regardless of the other contexts? (e.g. Light in context X is repressive compared with Dark in context Y for all X, Y).
  
  **Pivot design shows know for this biological system.**

- Is a factor important enough that it has an effect for any particular context? (e.g. for all X, Light in context X is repressive compared with Dark in X)
  
  **Pivot design suggests yes for this biological system.**
“Half-pivot” Light against a fixed background

<table>
<thead>
<tr>
<th>EXPT 1 NO PIVOT</th>
<th>LANE</th>
<th>ILLUMIN</th>
<th>STARVE</th>
<th>CARBON</th>
<th>NO3NH4</th>
<th>GLU</th>
<th>GLN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LIGHT</td>
<td>Y</td>
<td>0</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>2</td>
<td>LIGHT</td>
<td>N</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>H</td>
</tr>
<tr>
<td>3</td>
<td>LIGHT</td>
<td>N</td>
<td>0</td>
<td>0</td>
<td>H</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>LIGHT</td>
<td>Y</td>
<td>L</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>LIGHT</td>
<td>Y</td>
<td>L</td>
<td>0</td>
<td>H</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>6</td>
<td>LIGHT</td>
<td>N</td>
<td>0</td>
<td>L</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>DARK</td>
<td>N</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Exactly the same as LIGHT tests but with one added background.

Allows us to create a circuit (binary in this case because inputs are binary.)
Adaptive Experimental Design along “Borders”

Because combinatorial design explores only a (well spread) subset of possibilities, the apparent effects of factors may depend on other factors that haven’t been explored.

After constructing boolean circuits, software suggests “experiments to clarify border” between inductive and non-inductive, e.g., Starvation_Y, Carbon_N, NH4NO3_N, GLU_Y, GLN_Y
Combinatorial Design vs. Random Sampling

**Practical Question:** Adaptive Combinatorial Design is a sampling method. How well does it work compared to random sampling?

**Simulation experiment:** Create simulated data with a single important attribute and microarray-quality noise (factor of 2 to 5 change in biological replicates).

**Empirical Conclusions:** Random and Adaptive Combinatorial Design did equally well at identifying the important attribute (T-test), however Random falsely identified other attributes as important about 4 times more often than Adaptive Combinatorial Design. (see cdtables.doc)

**Ref:** Lejay et al. Systems Bio vol. 1.2 Dec. 2004
Steps of Methodology

No Pivot: Small set of well-spaced experiments to find most important influences on a target. Also, a good method in genomics applications to find clusters because of good spacing. Small? 10 inputs with 4 values gives a no pivot of about 30 experiments.

Pivot: Can find out whether an input is likely to have an effect regardless of context (for all X, for all Y) or for every context (for all contexts X)

Half-pivot: For comparison with a fixed background

Border Adaptation: Study differences between repressinve case and non-repressive one to discover fine structure.
Applicable to Many other Situations

Tuning an Algorithm Repeatedly and Online: Can’t explore whole parameter search space each time so use combinatorial design to sample the search space and then use border adaptation to fine-tune the result.

Regression testing: Given many input parameters to software, can’t test them all. This is a disciplined approach. No pivot idea only.
See Whether You’re Awake

Task: You have just designed an SQL database and you want to test the features. You have implemented joins, selects, groupbys, having clauses, uncorrelated subqueries, and correlated subqueries.

Specify:
- NumJoins 0 1 2 3
- NumSelects 0 1 2
- NumGroupby 0 1
- NumHaving 0 1
- NumUncorSub 0 1 2
- NumCorSub 0 1

Want test set that covers every possible pairwise combination. How many? Only 12. vs. 288 for
Inspiration of this approach

Combinatorial design: Inspired by work in software testing by David Cohen, Siddhartha Dalal, Michael Fredman and Gardner Patton at Bellcore/Telcordia. Their problem: how to test a good set of inputs to a program to discover whether there are any bugs. Not program coverage, but input coverage. Not all input combinations, but all combinations of every pair of input variables ("no pivot" design).

Hypothesis: every input combination should give same output: no error. If true for designed subset, then program is ok.
How This Could Help You

Use this approach: Pose an experimental setting of interest to you. (Names of input variables, possible values).
Describe a “no pivot” design for your setting. Based on that result, describe a pivot design to isolate the exact effect of a specific input.
Get a good sense of whether the pivot is decisive by itself or has a consistent strong influence.

Theoretical Guarantee: For k-factor design, if there is a set of k values that dominates the result, you will find it.
Safecracking Solution
(X = Don’t care)

- Number S1 S2 S3 S4 S5 S6 S7 S8 S9 S10
- 1: A A A A A A A A A A
- 2: A B B B B B B B B B
- 3: A C C C C C C C C C
- 4: B A B C A B C A B C
- 5: B B C A B C A B C A
- 6: B C A B C A B C A B
- 7: C A C B A C B A C B
- 8: C B A C B A C B A C
- 9: C C B A C B A C B A
- 10: X A A A B B B C C C
- 11: X A A A C C C B B B
- 12: X B B B A A A C C C
- 13: X B B B C C C A A A
- 14: X C C C A A A B B B
- 15: X C C C B B B A A A
Further Reading: combinatorial design widely used in biology

Universal DNA tag systems: a combinatorial design scheme
Recomb 2000
Amir Ben-Dor, Richard Karp, Benno Schwikowski and Zohar Yakhini.

Experimental design for gene expression microarrays,
Normal: N microarrays will be used to test N conditions against a common reference.
Authors propose to use the colors to compare N conditions against one another in a looping fashion: 1 with 2, 2 with 3, … n with 1. Result: deconvolves certain effects (e.g. binding affinity of reference dye.
Overview

• Simple Primer in Genomics
• Activist Data Mining
• Visualization tool for multi-experiment data
• A gallery of challenging problems.
• Lessons from successful collaborations.
Sungear Multifactor Visualization

Joint work with Rodrigo Gutiérrez, Manny Katari, Brad Paley, Chris Poultney, and Gloria Coruzzi
Typical Genomic Questions

• Multiple experiments (multiple time points, multiple conditions), many Go categories, or other features of genes: want to know when certain Go categories are highly represented.

• Many species, want to know which genes have presence in many species and perhaps which GO categories
Accepted Way to Compare Results: Venn Diagram
Venn Diagram Doesn’t Work Beyond 3, e.g. Intersect(D,B)
Computational Desires

• Simple, responsive interface
• Visualize lots of experiments (more than 3)
• Many ways to query
• Many different data representations
Sungear Design

- Generalizes Venn diagrams
- Visual outline is a polygon having anchors on borders and gears in the interior.
- Each gears points to associated anchors.
- Linked views to hierarchies, lists, and graphs, so can simultaneously update data depending on user queries (selection events).
**Seed-only N-only**

"Vessels"
Genes associated with Anchors (size proportional to number of genes)

**Common N-reg and Seed-Dev**

Gene list from Treatment

**Sungear Tool**

"Anchors"
Gene list from Treatment

**Sungear Tool**

TxnFactors

Nitrogen-Reg

Carbon/Light-Reg
Z-score = # S deviation from mean average GO term precalculated
Sungear Principle

- “Sungear is stupid”
- Doesn’t care what kind of data it is representing, though there is built-in support for genes (because of links to GO and to cytoscape).
- Basic Sungear representation could be used to describe anything from yachting gear to demographics.
Gene networks of NL-responsive genes
Demos

• Growth stages showing when genes are transcribed (N-reg AtGenExpSeedDev)
• Blast comparison of Arabidopsis against most fully sequenced organisms.
• Nitrogen, carbon, light, organ showing regulation -- relative expression (cnlo)
• Interspecies comparisons that might show which kinds of genes are missing in gymnosperms, for example (Vicogenta)
Sungear Sweet Spot

• Collections of data about some common entity (genes, people, goods, whatever) whose interaction you want to visualize.
• Biologists like the visual intuition this gives them: size matters, position matters.
Overview

• Simple Primer in Genomics
• Activist Data Mining
• Visualization tool for multi-experiment data
• A gallery of challenging problems.
• Lessons from successful collaborations.
Overview of Problems

• Provenance: remember where conclusions come from.
• A dream: dynamic, auto-generated docs
• Data integration: integrate data from different labs.
• Faster execution of primitives: alignment, folding
• Data Mining/Machine Learning: networks
Provenance
where does the data come from?

• Fact: some scientists/labs/equipment are better than others.

• “I wouldn’t trust SNP (human genetic variation) data from some labs ever” anon

• If your data mining model is based on bad data, it will produce garbage even if your algorithm is good.
Provenance Issues

- Record where data came from.
- Record where conclusions came from (truth maintenance).
- If data or your belief about data changes, then do something.
- Link with statistical quality control.
Curation Triggers  
(Simple truth maintenance)

- Databases are updated by people.  
- Often information is copied from one database to another, sometimes selectively.  
- Reasons are in the curator’s mind.  
- Suppose you record every change made.  
- Can you infer intent of the changes?  
- Could you infer “triggers” that change one database given changes in another?
Dynamic Auto-generated Docs
Current state of the art

- Experimenters test a dynamic system, e.g. a plant, a worm, a bacterium.
- Write up result in an article (usually with a novelistic hook).
- End result: dynamic system mapped to a (biased) static one.
Effort in the Opposite Direction

- Modeling language based on Petri nets called Statecharts.
- Used to model telecom systems, jet fighter control systems etc.
- New application: the early life of C elegans.
Problems of this Approach

• Painstaking effort to “dynamize” papers.
• Produce a model with limited appeal to experimenters who say in effect “Ho-hum, these are old papers. Look at the cool things we’re doing now. Anyway we don’t believe in models.” (cf. Danny Hillis article on why Biologists don’t believe models)
The Opportunity

- Movies track the dynamics of worm development naturally.
- They offer a uniform source of data (no problem of different authors, no author bias based on fashion).
- They are up-to-date.
- They map directly to the style of the modeling language.
Automation Possibility

• With image processing and proper recognition of states, it is possible to get a sequence of distinct events from each movie.

• These could be mapped perhaps automatically to the modeling language (research but possible if we put our minds to it)
Benefits

• A truly dynamic model, easy to update and refine.

• Possible ability to understand synergistic effects of different genes. For example, if knocking out gene g1 and gene g2 have effects on the same process, then good behavior of that process depends on the AND of those. Time dependency can be included.
Benefits II

• Mechanism for prediction of new experiments. For example, look at other genes whose expression is changed and test them based on strength of some effect. Might also test combinations of genes based on synergistic/antagonistic effects of the individual genes.
New scientific opportunity

• Completely state of the art “Dynamic Temporal Database” (like a database but gives a dynamic model, one that evolves in time). Can query based on causes (this input or that one) or based on effect (how can I get this result?)
Challenges

• Why do documents have to be static?
• Why do documents have to be authored?
• Can one develop a dynamic document framework such that incoming event sequences can update details of the framework automatically?
• “Reading” the document can be like viewing a film (normal behavior) or modifying the inputs and then watching.
Data Integration

- Different labs produce data having different attribute names, different semantics and various statistical thresholds.
- Putting this all together is now done by hand (i.e. perl hacking).
- A really good attack on this problem would be extremely useful.
Data Integration Issues

• Keep raw data: Gene expression is a floating point number. What do you consider “induced”?
• Keep meta-data: Chips for measurement are different. Growth conditions.
• Some Researchers: Zoe Lacroix, Louiqa Raschid; Phokion Kolaitis and Wang-Chiew Tan; Peter Buneman; Phil Bernstein; Alon Halevy
Faster Execution of Primitives

- Remember: key resource is people time, so biologists’ eyes glaze over logarithmic time complexity factors that save them 10 seconds.
- For some apps, e.g. phylogeny reconstruction and molecular modeling, computation is the bottleneck.
- Goal: to handle more species or larger molecules.
Fast Execution: issues

- Many of the problems require lots of simulation, so be ready for numerical problems, e.g. multi-body problems done right.

- Other problems are NP-complete (e.g. multiple alignment), so find out biologists’ tolerance for sub-optimal results.

- Ref: AntiClustAl: Multiple sequence alignment by antipole clustering, by DiPietro et al. 2005, Data Mining in Bioinformatics
Data Mining/Machine Learning Example

Network inference: Many edges connect genes (metabolic, protein-protein, co-regulation). Can one infer new edges from existing ones?

Can one predict outcome of an experiment/mutation?
Multinetwork – differently labeled edges between nodes

Gene A – Gene B
Gene D – Gene C
Transcription factor (TF)-binding: very important

A single TF binds to a single cis element (motif)

Source: U.S. Department of Energy Genomics (http://doegenomestolife.org)
Properties of Edges

- Edges have numerical value, e.g. strength of correlation.
- Some edges are directed. Others are not.
- Seek rules of the form: if two genes $g_1$ and $g_2$ are coregulated, $g_1$ is a transcription factor and there is a protein-protein interaction between them, then $g_1$ regulates $g_2$. (just an example)
Posed as a Machine Learning Problem

• Given metabolic edges (directed), protein-protein interaction (undirected), list of transcription factors, coregulations, expression data for transcription factor knockouts, (1) infer new edges; (2) figure out the effect of knocking out a transcription factor.

• DB Researchers: Jagadish, Frank Olken, Mona Singh
Metabolic network for Arabidopsis – problem is messy
Feedback loop through experiments

Expression and/or growth data

TIME

Treatment and/or Developmental

Organs

Cells

Aim1

Aim2

Metabolic & Developmental networks

Machine learning algorithm

Gene Networks

Aim3

Inferred regulator

Adaptive adjustment of evidence weights and scores.

Expression and/or growth data (mutant vs. wild type)

Gene Networks

Leaf, root, seed & root cell - specific gene networks

Identify mutants

Aim4

Metabolic regulator

Developmental regulator
Tree and Graph Searching

- Joint work with Cantone, Ferro, Giugno, Pulvirenti, Reforgiato

- Introduction:
  - Application examples
  - Framework for tree and graph matching techniques

- Algorithms:
  - Tree Searching
  - Graph Searching
  - Graph clustering
Usefulness

• Trees and graphs *represent data* in many domains in linguistics, vision, chemistry, web. (Even sociology.)

• Tree and graphs searching algorithms are used to *retrieve information* from the data.
TreeBASE Search Engine

Netscape: TreeBASE Search


Either
A. Enter your query tree here:

```
[Onychophora, (Crustacea, Hexapoda, ?), (Thelyphondida, Schizomida)]
```

Set maxdist: [ ]

Submit

*Please click here for an example query tree.*

Or:
B. Select the tree file on your local machine:

Set maxdist: [ ]

Browse...

View Query Tree: [Text]

Search Results:

Tree: Tree185 [Text]
Tree: Tree186 [Text]

Annotation:

Click the icon to view the graphic representation of a tree. (The matching nodes are shown in bold font and with a green dot.)

Click the [Text] to view the text representation of a tree.

Onychophora [2]
Crustacea [11]
Hexapoda [11]
Mysxipoda [11]
Pychnogomida [11]
Xiphosura [11]
Araecae [5]
Ambipyghi [21]
Thelyphondida [11]
Schizomida [11]
Pulgipredi [11]
Tree Searching

• Given a small tree $t$ is it present in a bigger tree $T$?
Present but not identical

• "Happy families are all alike; every unhappy family is unhappy in its own way" *Anna Karenina* by Leo Tolstoy

• Preserving sibling order or not
• Preserving ancestor order or not
• Distinguishing between parent and ancestor
• Allowing mismatches or not
TreeSearch Query Language

- Query language is simply a tree decorated with single length don’t cares (?) and variable length don’t cares (*).

```
>= 0, on each side
A
  *  ? =1
B  C  D
```

Exact Match

- Query matches exactly if contained regardless of sibling order or other nodes
Inexact Match

- Inexact match if missing or differing node labels. Higher differences cost more.
Can query with graphs as well as trees.

- **Node**
  - a/

- **Edge**
  - a/b/

- **Path**
  - a/b/c/f/

- **Branches**
  - a/(h/c/)b/
GraphClust: Basic Problem

GOAL: Given a database of labeled graphs, cluster them

Difficulties:
1) exact graph matching problems are NP-complete, hence graphclust using graph matching is extremely time consuming

2) evaluation of a distance between two graphs is still an open question
Tree/Graph Work

• Can find small trees in bigger trees.

• Small graphs in big graphs. This is now a cytoscape add-on.

• Can cluster graphs.
Conclusions

• We are well positioned to deal with time course data, network data, and pathway data.

• This may lead to good new things to do.
Overview

• Simple Primer in Genomics
• Activist Data Mining
• Visualization tool for multi-experiment data
• A gallery of challenging problems.
• Lessons from successful collaborations.
Lesson 1: caring about data

- Biologists look at data. Computer scientists (even database people) don’t.
- Data is noisy.
- Good news: qualitative results are enough. E.g., Ibuprofen fights inflammation.
- Ideal: experimental results + algorithms → testable likely hypotheses.
Lesson 2: Hard to get assumptions right

• Computer scientists solve puzzles.
• Solve problems with as little regard for semantics as possible (sorting, databases, statistical packages).
• Paradigm: design an algorithm, scientists implement it and celebrate the creator.
• Works for physics, but seldom for biology; biologists trust experiments.
Lessons from a Fruitful Collaboration III

• Remember that people time is important.
• Computer scientists should give fast turnaround (two days or less) on simple tasks.
• “Interesting” tasks are not far behind.
• Reserve those for your graduate students.
Lessons from a Fruitful Collaboration IV

• Meet every week.
• Don’t be afraid to be ignorant.
• Computer scientists should get involved in experimental design (be activist).
Closing words


You have a lot to offer, but the problems don’t come neatly packaged.

When you do discover a problem, keep your solution simple – fewer assumptions.

Remember to be lucky: the experiments have to work out for biologists to believe your method.