The $1000 Genome Is Revolutionizing How We Study Biology and Practice Medicine

$2.7B Over Decade $1000 in ~3 Days
$1000 Human Genome “Resequencing”

Massively Parallel Sequencing  Reference Human Assembly

~30x coverage

Every base covered by approximately least 30 reads

Reference sequence

Your genome for $9,500
Illumina’s Personal Genome Service (http://www.everygenome.com)
Biology Increasingly Data Intensive

Need to leverage existing knowledge to advance your own research
Analysis Techniques Becoming More Transparent
Analysis of Genomic Data
Not Only Critical, But Fun

Serendipitous discovery of Wolbachia genomes in multiple Drosophila species
Steven L Salzberg*, Julie C Dunning Hotopp*, Arthur L Delcher*,
Mihai Pop*, Douglas R Smith‡, Michael B Eisen‡ and William C Nelson*

Wolbachia – endosymbionts found in fly embryos

Genome sequences for three new Wolbachia species reconstructed from sequence traces from 7 Drosophila species

95% (1,440,650 base pairs) for one of the new species
**Bioinformatics** - the creation and advancement of algorithms, computational and statistical techniques, and theory to solve formal and practical problems posed by or inspired from the management and analysis of biological data.

**Computational Biology** - hypothesis-driven investigation of a specific biological problem using computers, carried out with experimental and simulated data, with the primary goal of discovery and the advancement of biological knowledge.
Bioinformatics Example:

Comparative Toxicogenomics Database
Comparative Toxicogenomics Database

Data Status: September 2014

These statistics reflect our latest monthly data release.

- Chemical–gene interactions (curated): 1,014,487
  - Unique chemicals: 10,847
  - Unique genes: 35,543
  - Unique organisms: 522

- Gene–disease associations: 15,436,754
  - Curated: 29,779
    - Unique genes: 7,483
    - Unique diseases: 4,925
  - Inferred: 15,406,975
    - Unique genes: 35,381
    - Unique diseases: 3,008

- Chemical–disease associations: 1,673,033
  - Curated: 192,440
    - Unique chemicals: 8,574
    - Unique diseases: 3,058
  - Inferred: 1,480,593
    - Unique chemicals: 10,583
    - Unique diseases: 4,209

- Chemical–GO associations (enriched): 3,417,672

- Chemical–pathway associations (enriched): 265,570

- Disease–pathway associations (inferred): 55,649

- Gene–gene interactions: 293,565

- Gene–GO annotations: 1,082,492

- Gene–pathway annotations: 63,633

- GO–disease associations (inferred): 654,295

- Chemicals with curated data: 13,697

- Diseases with curated data: 6,351
  - Via OMIM curation: 3,423
  - Genes with curated data: 36,631
    - Via OMIM curation: 3,423
  - Curated references: 110,402
... that transforms into a discovery tool

**Curated Facts**

\[ \text{Attributes} \quad \text{GO, KEGG, Reactome, etc.} \]

**Inferred Discoveries**

Allan Peter Davis
Autistic Disorder: genes

1. Direct
   - 240 direct genes

2. Inferred via network
   - G
   - M
   - C
   - autism
   - inferred

3. Direct and inferred
   - G
   - M
   - C
   - autism

Allan Peter Davis
Autistic Disorder: chemicals

<table>
<thead>
<tr>
<th>Chemical</th>
<th>Disease</th>
<th>Direct Evidence</th>
<th>Inference Network</th>
<th>Inference Score</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diazinon</td>
<td>Autistic Disorder</td>
<td>27 genes: ADRB2; CHRNA4; CHRNA7; CHRN2B2; COMT; GPX1; GRIK1; GRIK2; GRIN2A; GRM8; GSTM1; HTR1D; HTR2A; HTR3A; HTR5A; IFNG; IL10; IL6; IL4; MAOB; NTRK2; PON1; SLC1A3; SLC6A4; SLC6A8; SLC6B2</td>
<td>2. inferred C-D, via network</td>
<td>51.97</td>
<td>26</td>
</tr>
</tbody>
</table>

Inference Score
computed using local network topology statistics;
ranks C-D inferences for atypical connectivity
(higher score = better)

Allan Peter Davis
BioGRID Gene and Protein Interactions at CTD

Inferred C-D, via network

Bisphenol A

220 genes

breast neoplasms

inferred
Theory of Computation
Theory of Computation

ENIAC – Unveiled on Feb. 14, 1946
Computation
Many applications including:

Statistics

Bioinformatics and Computational Biology

Computation Often Involves The Use of Models

Develop a model that describes a system
- Helps you understand the system
- Use it to predict responses of the system
Model of Electric Current Flow Across the Cell Membrane of the Giant Nerve Fiber of a Squid

A.L. Hodgkin and A.F. Huxley, 1952

\[ I = C_M \cdot \frac{dV_M}{dt} + i_{Na} + i_K + i_L \]
Figure 17. Time courses if $g_{Na}$ and $g_K$ at 5 potentials. Squid giant axon depolarized to indicated potentials at $t = 0$. (O) ionic conductances calculated from separated currents at $6.3^\circ C$ using Eq. (12) and 13. Smooth curves, time courses of $g_{Na}$ and $g_K$ calculated from Hodgkin-Huxley model. (Source: From Hodgkin 127, adapted from Hodgkin & Huxley 133.)

Genetic regulatory network controlling the development of the body plan of the sea urchin embryo

Modeling Protein Structure


**Molecular Dynamics**

Proteins constantly moving:

- **Local Motions** (0.01 to 5 Å, $10^{-15}$ to $10^{-1}$ s)
  - Atomic fluctuations
  - Sidechain motions
  - Loop motions
- **Rigid Body Motions** (1 to 10 Å, $10^{-9}$ to 1s)
  - Helix motions
  - Domain motions (hinge bending)
  - Subunit motions
- **Large-Scale Motions** (> 5 Å, $10^{-7}$ to $10^4$ s)
  - Helix coil transitions
  - Dissociation / Association
  - Folding and unfolding

**Model System Containing Protein**:

- Energy within system
  - $E = kT$
- Each atom has:
  - Position in system (x, y, z)
  - Velocity
  - Set of constraints (e.g., bonds)

![Diagram of molecular dynamics with coordinates and angles]
Influenza Neuraminidase N1 Complexed with Tamiflu (Oseltamivir) Inhibitor

https://www.youtube.com/watch?v=Vj8ri57GE_M
Algorithms

Precise method usable by a computer for the solution of a problem

Composed of a finite set of steps, each of which may require one or more operations

• Each operation must be definite

• Each operation must be effective (can be completed)

• Operations must terminate after a finite number of operations

• Have zero or more inputs

• Have one or more outputs
Types of Algorithms

Deterministic algorithms

Result of each operation clearly defined

Nondeterministic algorithms

Contains operations whose outcome are not uniquely defined

Limited to a specific set of possibilities
Algorithms are Written in a Programming Language

Languages are defined so that each legitimate sentence has a unique meaning.

A program is the expression of an algorithm in a programming language.

**C/C++**  
**Java**

**Compiler**

**Compiled Code**  
(executable in operating system)

**Processor**

**Scripting Languages**

**Perl**  
**Python**

**Program**

**Use Interpreter to execute commands**

**Processor**
Algorithms Research

How to devise algorithms
   e.g., Dynamic programming, ...

How to validate algorithms
   Proof that algorithm works

How to analyze algorithms (Computational Complexity Theory)

How to express algorithms
   e.g., Object-oriented programming, procedural programming

How to test a program
   Debugging – test for faults
   Profiling – measuring time and space it takes to run
Computational Complexity Theory

How much computing time and space (memory) an algorithm will require?

Used to make qualitative judgments about the value of one algorithm over another

\( O \) – notation

Represent computing time

\[ f(n) = O(g(n)) \]

Number of inputs or outputs
“Big O Notation” – Represents length of computing time

<table>
<thead>
<tr>
<th>Notation</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(1)$</td>
<td>constant</td>
</tr>
<tr>
<td>$O(\log^* n)$</td>
<td>iterated logarithmic</td>
</tr>
<tr>
<td>$O(\log n)$</td>
<td>logarithmic</td>
</tr>
<tr>
<td>$O((\log n)^\circ)$</td>
<td>polylogarithmic</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>sublinear</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>linear</td>
</tr>
<tr>
<td>$O(n \log n)$</td>
<td>linearithmic, loglinear, quasilinear or suprilinear</td>
</tr>
<tr>
<td>$O(n^2)$</td>
<td>quadratic</td>
</tr>
<tr>
<td>$O(n^c), c &gt; 1$</td>
<td>polynomial, sometimes called algebraic</td>
</tr>
<tr>
<td>$O(c^n)$</td>
<td>exponential, sometimes called geometric</td>
</tr>
<tr>
<td>$O(n!)$</td>
<td>factorial, sometimes called combinatorial</td>
</tr>
<tr>
<td>$O(n^n)$</td>
<td>Exponential</td>
</tr>
</tbody>
</table>

sorting

two nested loops (e.g., all versus all search)
Complexity Classes of Computational Problems

P = deterministic in polynomial time
   Size of input determines length of computation by a polynomial

NP = non-deterministic in polynomial time

NP-hard -> not solvable by an algorithm that is guaranteed to:
   • run in polynomial time and
   • always produces an optimal solution
   • e.g., Find shortest route for salesman

NP-complete -> Special case of NP-hard problems that are also NP
   • e.g., Given costs and a budget, is there a route that costs less than budget?

Traveling salesman problem
Shortest route to visit all points
n! permutations

Reason that
Heuristic Algorithms are used
Reference for ways to implement many algorithms

Example topics:
• Fitting data to a straight line
• Markov Chain Monte Carlo
• Fast Fourier Transform
Overview of 4 Lectures

- **Introduction to Computation and Programming**: Mon. Sept. 14
- **Programming (Text File Processing)**: Wed. Sept 16 & Mon. Sept 21
- **Genome Sequencing and Informatics**: Wed. Sept 23

**Homework Due on Oct. 7th (Wed) by 2pm**
Perl

http://www.cpan.org – Comprehensive Perl Archive Network
http://www.activestate.com – Active Perl (for PC, Mac OS X, Linux)
Programming Concepts

Variables

Used to store:
- character string
- integer
- real number
- Boolean value (True or False)

\$a = \text{“Go Bears”;} \\
\$b = 25; \\
\$c = 3.1415; \\
\$d = 0;

Data Structures

Store “collections” of data in an organized fashion

Common Operations

Mathematical operations
Testing for specific values (if / then loop)
Iteration (for, while loops)
Translation operations
Printing messages
Reading in files
Writing output
#!/usr/bin/perl

# Header
# Example script

# Variable declarations
$a = "Go ";
$b = "Black ";
$c = "Bears";

# Main
print $a, $b, $c;

perl go_bears.pl
Go Black Bears
Anatomy of a Program

#!/bin/perl

# Header

# Variable declarations

# Input and Output file handling

# Main
Variables

Scalar Types:
- character string
- integer
- real number
- Boolean value (True or False)

```perl
$a = "TAATAA";
print $a;

$n = 25;
$m = 100;
$sum = $n + $m;
```