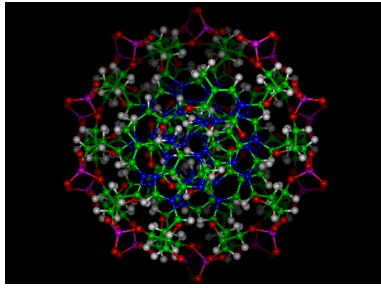


Class 26: Computing Genomes, Genomes Computing



David Evans
<http://www.cs.virginia.edu/evans>
cs302: Theory of Computation
University of Virginia Computer Science

Final Exam Sneak Preview

- Handout available now
- Honor policy: you **may** discuss these problems with others and use any resources you want until the Final
- No notes or other resources may be used during the final
- Intent is to give you an idea what to expect on the final and a chance to start thinking about some problems
 - Don't attempt to memorize answers: need to understand things since the actual questions may be different

Lecture 26: Computing Genomes and Vice Versa

2

Menu

- Computing Genomes (PS6, Problem 6)
 - Crash course in biology
- **Busy Beaver result!**
- Computing with Genomes
- Conclusion

Lecture 26: Computing Genomes and Vice Versa

3

Genome Assembly Problem

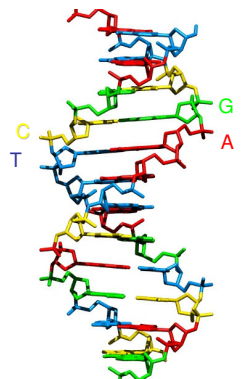
In order to assemble a genome, it is necessary to combine snippets from many reads into a single sequence. The input is a set of n genome snippets, each of which is a string of up to k symbols. The output is the smallest single string that contains all of the input snippets as substrings.

Lecture 26: Computing Genomes and Vice Versa

4

DNA

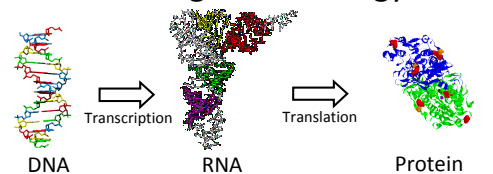
- Sequence of nucleotides: adenine (A), guanine (G), cytosine (C), and thymine (T)
- Two strands, A must attach to T and G must attach to C



Lecture 26: Computing Genomes and Vice Versa

5

Central Dogma of Biology




- RNA makes copies of DNA segments
- RNA describes sequences of amino acids
- Chains of amino acids make proteins
- Proteins make us

Lecture 26: Computing Genomes and Vice Versa

6

Human Genome




- 3 Billion Base Pairs
 - Each nucleotide is 2 bits (4 possibilities)
 - 3 B pairs * 1 byte/4 pairs = 750 MB
- Every sequence of 3 base pairs one of 20 amino acids (or stop codon)
 - 21 possible codons, but $4^3 = 64$ possible
 - So, really only $750\text{MB} * (21/64) \sim 250$ MB
- Much of it (> 95%) is may be junk (doesn't encode proteins, but some might be important)

1 CD ~ 650 MB


Lecture 26: Computing Genomes and Vice Versa 7 Computer Science
at the University of Virginia

Human Genome Race



Francis Collins
(Director of
public National
Center for Human
Genome
Research)
(Picture from UVa
Graduation 2001)

VS.






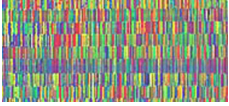
Craig Venter
(President of
Celera Genomics)

- UVa CLAS 1970
- San Mateo College
- Yale PhD
- Court-martialed
- Tenured Professor at U. Michigan
- Denied tenure at SUNY Buffalo

Lecture 26: Computing Genomes and Vice Versa 8 Computer Science
at the University of Virginia

Reading the Genome

Whitehead Institute, MIT

Lecture 26: Computing Genomes and Vice Versa 9 Computer Science
at the University of Virginia

Gene Reading Machines

- One read: about 700 base pairs
- But...don't know where they are on the chromosome

Read 3 TACCCGTGATCCA

Read 2 TCCAGAATAA

Read 1 ACCAGAATACC

Actual Genome AGGCATACCAGAATACCCGTGATCCAGAATAAGC

Lecture 26: Computing Genomes and Vice Versa 10 Computer Science
at the University of Virginia

Genome Assembly Problem

Read 1 ACCAGAATACC

Read 2 TCCAGAATAA

Read 3 TACCCGTGATCCA

Input: Genome fragments (but without knowing where they are from)

Output: The full genome

Lecture 26: Computing Genomes and Vice Versa 11 Computer Science
at the University of Virginia

Decision Problem

$$GA = \{ \langle \{ x_1, x_2, \dots, x_n \}, m \rangle \mid \text{where each } x_i \text{ is a string and there is a string } X \text{ of length } m \text{ that includes all of the } x_i \text{ strings as substrings} \}$$

If we had a decider for GA , can we find the length of the shortest common superstring?

Yes. Try all possible m values from $1, 2, \dots, \sum |x_i|$.

Lecture 26: Computing Genomes and Vice Versa 12 Computer Science
at the University of Virginia

Is GA In Class NP?

$GA = \{ \langle \{ x_1, x_2, \dots, x_n \}, m \rangle \mid \text{where each } x_i \text{ is a string and there is a string } X \text{ of length } m \text{ that includes all of the } x_i \text{ strings as substrings} \}$

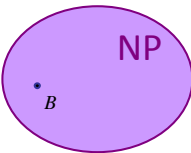
Yes. The string X is a polynomial-verifiable certificate.
 - Check it includes each substring
 - Check its length is $\leq m$

Is GA NP-Complete?

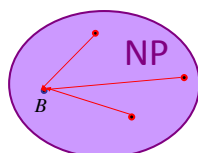
$GA = \{ \langle \{ x_1, x_2, \dots, x_n \}, m \rangle \mid \text{where each } x_i \text{ is a string and there is a string } X \text{ of length } m \text{ that includes all of the } x_i \text{ strings as substrings} \}$

NP-Complete

A language B is in NP-complete if:



1. $B \in \mathbf{NP}$



2. There is a polynomial-time reduction from every problem $A \in \mathbf{NP}$ to B .



To Prove NP-Hardness

- Pick some known NP-Complete problem X .
- Show that a polynomial-time solver for Y could be used to build a polynomial-time solver for X .
- This proves that there is no polynomial-time solver for Y (unless $P = NP$).

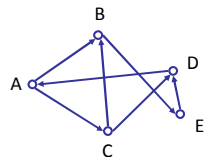
Possible Choices...

$(a \vee b \vee c) \wedge (\neg a \vee b \vee \neg c) \dots$

3SAT

$\langle \{3, 5, 12, 13, 17\}, 30 \rangle$

SUBSET-SUM



HAMPATH

By definition, all **must** work. Every NP-Complete problem can be reduced to every NP-Complete problem.

In practice, some will work much more easily than others. Try to pick a problem "close" to the target problem.

Busy Beaver Challenge Ruixin Yang

Consider Each Node

Incoming Edges
(x, a)

Outgoing Edges
(a, y)

In a Hamiltonian path, for each node (except start and end), **exactly** one incoming edge, and one outgoing edge must be used.

Lecture 26: Computing Genomes and Vice Versa 25 Computer Science
at the University of Virginia

Consider Each Node

Incoming Edges
(x, a)

Outgoing Edges
(a, y)

We need GA substrings to represent each edge, but such that only 1 can be actually followed. But, the GA superstring needs to include all substrings.

Idea: make it so the untaken edges can loop back!

Lecture 26: Computing Genomes and Vice Versa 26 Computer Science
at the University of Virginia

Simple Nodes

If there is only one edge (a, b) out of a given node, that edge must be used in the path. Add the substring: **ab**

Lecture 26: Computing Genomes and Vice Versa 27 Computer Science
at the University of Virginia

Generating the Substrings

- For each edge (a, y_i) add two substrings:
 - ay_ia This is the "back" edge.
 - y_iay_{i+1} This connects the possible destinations.

aba **aca**
bac **cab**

Possible (length 4) alignments:

aba **aca**
bac **cab**

If there is a Hamiltonian path, one of these must be used.

Lecture 26: Computing Genomes and Vice Versa 28 Computer Science
at the University of Virginia

Start and End

$\langle G, A, E \rangle$

The start node must come first:
 Add string: **@A** (where @ is unused elsewhere)
 Add string: **E\$** (where \$ is unused elsewhere)

Lecture 26: Computing Genomes and Vice Versa 29 Computer Science
at the University of Virginia

What is m?

$\langle G, A, D \rangle$

{ @A, ABA, BAC, ACA, CAB, BE, ED, DA, CBC, CDC, BCD, DBC, D\$ }

@A
ACA
CAB
ABA
BAC
CBC
BCD
CDC
DCB
BE
ED
D\$

$m = \text{Start and end} = 2$
 + one-edge nodes * 1
 + multi-edge nodes * 2 for each edge
 + 2 for align

Lecture 26: Computing Genomes and Vice Versa 30 Computer Science
at the University of Virginia

Human Genome

- 3 Billion base pairs
- 600-700 bases per read
- ~8X coverage required
- So, $n \approx 37$ Million sequence fragments
- Celera used 27.2 Million reads (but could get more than 700 bases per read)

How can we solve an NP-Complete problem for such large n ?

Approaches

- Human Genome Project (Collins)
 - Start by producing a genome map (using biology, chemistry, etc) to have a framework for knowing where the fragments should go
- Celera Solution (Venter)
 - Approximate: we can't guarantee finding the shortest possible, but we can develop clever algorithms that get close most of the time

Result: Draw



President Clinton announces Human Genome Sequence essentially complete (with Venter and Collins), June 26, 2000

But, Human Genome Project mostly adopted Venter's approach.

Genomes Computing

Solving *HAMPATH* with DNA

- Make up a two random 4-nucleotide sequences for each node:

A:	$A_1 =$ ACTT	$A_2 =$ gcag
B:	$B_1 =$ TCGG	$B_2 =$ actg
C:	$C_1 =$ GGCT	$C_2 =$ atgt
D:	$D_1 =$ GATC	$D_2 =$ tcca
- If there is a link between two cities ($X \rightarrow Y$), create a nucleotide sequence: $X_2 Y_1$

A \rightarrow B	gcagTCGG
B \rightarrow A	actgACTT

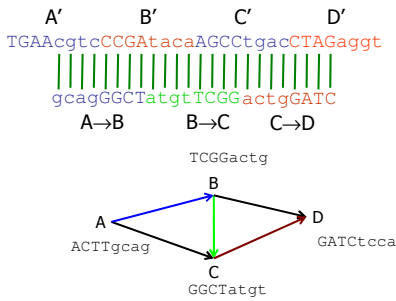
Based on Fred Haggood's notes on Adelman's talk

Encoding The Problem

- Each city nucleotide sequence binds with its complement ($A \leftrightarrow T, G \leftrightarrow C$):

A:	$A_1 =$ ACTT	$A_2 =$ gcag
A':	TGAA	cgtc
B:	TCGGactg	
B':	AGCCtgac	
C:	GGCTatgt	$C' =$ CCGAtaca
D:	GATCtcca	$D' =$ CTAGaggt
- Mix up all the link and complement DNA strands – they will bind to show a path!

Path Binding



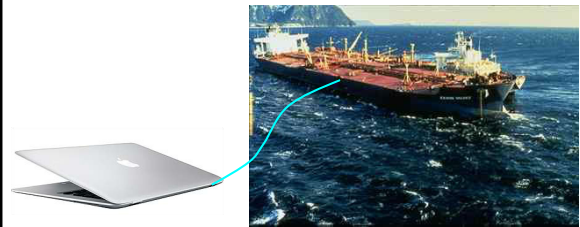
Getting the Solution

- Shake up all the DNA to get it to bind
- Extract DNA strands starting with A and ending with D
 - Can do this with chemical binding on start/end tags: remove all strands that do not start with A, and then remove all strands that do not end with D
- Weigh remaining strands to find ones with the right weight (7 * 8 nucleotides)
- Select one of these and read its sequence

Is Church-Turning Wrong?

- Time to solve problem with DNA computer doesn't scale with input size
 - Can shake up any amount of DNA in the same amount of time!
- Can DNA computers solve undecidable problems? No (at least not like this). Can simulate everything with TM.
- Is TM model robust enough for P to be the same for DNA computer?
 - No: DNA computer can solve NP-Hard problems in constant time! Volume of DNA needed grows exponentially with input size.

DNA-Enhanced PC



To solve HAMPATH for 45 vertices, you need ~20M gallons

Where to Go From Here

- Talks today and tomorrow (both in new Library)
 - 4:00pm today: Curtis Wong
 - 3:00 tomorrow: Bill Wulf
- Security and Theory Lunch Groups
 - See handout for links
- Courses:
 - CS660: Graduate Theory of Computation
 - CS432, MATH450, PHIL233, Cryptography

Thanks!