Computer Science for Solving Problems

“Directions from California Academy of Sciences to Ferry Building?”

• Recurring problem

• Should have a “formula” or general scheme

• Need formal model!

model |ˈmädl|

[…]  
• a simplified description, especially a mathematical one, of a system or process, to assist calculations and predictions  

[...]  

New Oxford American Dictionary
A Model for the Directions Problem
A Model for the Directions Problem
A Model for the Directions Problem
A Model for the Directions Problem
Directed Acyclic Graphs

Graph $G = (V, E)$ where $E \subseteq V \times V$

edge-label function $c: E \rightarrow \{1, 2, \ldots\}$
Directed Acyclic Graphs

directed graph $G = (V, E)$ where $E \subseteq V \times V$

edge-label function $c : E \rightarrow \{1, 2, \ldots\}$
Linearizing DAGs

Can move vertices so that edges from left to right!
Linearizing DAGs

Can move vertices so that edges from left to right!
Subproblem Structure

\[ d(F) = \min\{d(D) + 4, d(E) + 2\} \]
A Dynamic Program for Shortest Paths

∀ \( v \in V \setminus \{A\} \): \( d(v) = \min_{(u, v) \in E} \{d(u) + c(u, v)\} \)
A Dynamic Program for Shortest Paths

\[ \forall v \in V \setminus \{A\} : d(v) = \min_{(u,v) \in E} \{d(u) + c(u, v)\} \]
A Dynamic Program for Shortest Paths

∀v ∈ V \ {A} : d(v) = \min_{(u,v)\in E} \{d(u) + c(u, v)\}
A Dynamic Program for Shortest Paths

∀v ∈ V \ {A} : d(v) = \min_{(u,v) \in E} \{d(u) + c(u, v)\}
A Dynamic Program for Shortest Paths

∀v ∈ V \ {A} : d(v) = \min_{(u,v) \in E} \{d(u) + c(u, v)\}
A Dynamic Program for Shortest Paths

∀\(v \in V \setminus \{A\} : d(v) = \min_{(u, v) \in E} \{d(u) + c(u, v)\}\)
∀v ∈ V \ {A} : d(v) = \min_{(u, v) ∈ E} \{d(u) + c(u, v)\}
Subproblem DAG

- Vertex $\approx$ (optimization) problem
- Predecessor vertex $\approx$ subproblem
  - “Acyclic” is crucial
- Subproblems may overlap
  - Optimal solution for one vertex induces optimal solution for at least one predecessor
- “Bottom-up”: Progressively larger problems
Fibonacci Numbers

\[ F_n = F_{n-1} + F_{n-2} \]

\[ F_1 = 1 \text{ and } F_0 = 0 \]
Fibonacci Numbers

\[ F_n = F_{n-1} + F_{n-2} \]

\[ F_1 = 1 \text{ and } F_0 = 0 \]

Example: Genealogical tree of male bee
“Top-Down” Recursion

\[ F_n = F_{n-1} + F_{n-2} \]

\[ F_1 = 1 \text{ and } F_0 = 0 \]

This Java code is excruciatingly slow! Why?
“Bottom-up” Dynamic Program

- Subproblem DAG is implicit
- Operation on subproblem results is just addition
“Bottom-up” Dynamic Program

• Subproblem DAG is implicit

• Operation on subproblem results is just addition
“Bottom-up” Dynamic Program

- Subproblem DAG is implicit
- Operation on subproblem results is just addition
“Bottom-up” Dynamic Program

- Subproblem DAG is implicit
- Operation on subproblem results is just addition
“Bottom-up” Dynamic Program

- Subproblem DAG is implicit
- Operation on subproblem results is just addition
“Bottom-up” Dynamic Program

- Subproblem DAG is implicit
- Operation on subproblem results is just addition
Dynamic Programming

• Term coined by Richard Bellman in the 1950s
• Programming $\approx$ planning over time
• Secretary of Defense hostile to mathematical research

[…] it’s impossible to use the word \textit{dynamic} in a pejorative sense. […] It was something not even a Congressman could object to. […]

**Edit Distance**

- Measure for dissimilarity of two character strings
- Intuitive: minimum number of elementary edit operations (insert, delete, replace)
- Can represent as alignment

```
  the  the e  the  the  the
the e t  e a t e a
t e a  t e a
t e a
t e a
t e a
t e a
```

- Edit distance between “the” and “tea” = 2
Formal Problem Definition

• Input: Sequences $x [1..n]$ and $y [1..m]$

\[
\begin{array}{cccc}
  x_1 & x_2 & \ldots & x_n \\
  \hline
  y_1 & y_2 & \ldots & y_m \\
\end{array}
\]

• Output: length $d$ of a minimum-length alignment
  (note: $0 \leq n + m \leq d$)
Where is the Subproblem DAG?

Only three alignments of \( x[1\ldots n] \) and \( y[1\ldots m] \)

\[
\begin{array}{|c|c|}
\hline
x[1\ldots n - 1] & x[n] \\
\hline
y[1\ldots m - 1] & y[m] \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|}
\hline
x[1\ldots n - 1] & x[n] \\
\hline
y[1\ldots m] & - \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|}
\hline
x[1\ldots n] & - \\
\hline
y[1\ldots m - 1] & y[m] \\
\hline
\end{array}
\]

The diagram shows the DAG structure with nodes labeled from \( m \) to \( n \), with edges diffing \( x[n] \) and \( y[m] \) and other relevant nodes.
Where is the Subproblem DAG?

Only three alignments of $x[1\ldots n]$ and $y[1\ldots m]$
Recall: Optimal Substructure

• Let $u$ be predecessor (subproblem) of $v$

• $d(v) = d(u) + c(u, v)$

$\iff u$ on shortest path from $A$ to $v$
Edit Distance Has Optimal Substructure

An optimal alignment has optimal sub-alignments

\[
d(3,3) = d(2,2) + \text{diff}(x[3], y[3])
\]
A Dynamic Program for Edit Distance

\[
\begin{align*}
\begin{array}{|c|c|}
\hline
x[1...n-1] & x[n] \\
\hline
y[1...m-1] & y[m] \\
\hline
\end{array}
\begin{array}{|c|c|}
\hline
x[1...n-1] & x[n] \\
\hline
y[1...m] & - \\
\hline
\end{array}
\begin{array}{|c|c|}
\hline
x[1...n] & - \\
\hline
y[1...m-1] & y[m] \\
\hline
\end{array}
\end{align*}
\]

\[
d(i, 0) = i \text{ and } d(0, j) = j
\]

\[
d(n, m) = \min \left\{ d(n-1, m) + 1, \\
d(n, m-1) + 1, \\
d(n-1, m-1) + \text{diff}(x[n], y[m]) \right\}
\]
Example

<table>
<thead>
<tr>
<th></th>
<th>t</th>
<th>h</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

- t
- e
- a
Example

<table>
<thead>
<tr>
<th></th>
<th>t</th>
<th>h</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>t</td>
<td>t</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>t</td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>e</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>t</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
Example

<table>
<thead>
<tr>
<th></th>
<th>t</th>
<th>h</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

```
<table>
<thead>
<tr>
<th></th>
<th>t</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>t</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
```
Example
Example

<table>
<thead>
<tr>
<th>a</th>
<th></th>
<th></th>
<th>t</th>
<th>h</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>t</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

<table>
<thead>
<tr>
<th></th>
<th>t</th>
<th>h</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>t</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>e</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagram showing the movement from 0 to 3.
Example

<table>
<thead>
<tr>
<th></th>
<th>t</th>
<th>h</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>t</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>e</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>a</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

<table>
<thead>
<tr>
<th></th>
<th>t</th>
<th>h</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>t</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>e</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>a</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The table represents the occurrence of each letter in the word "the".
Example

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>h</td>
<td>e</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>t</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>e</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>a</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
Example

<table>
<thead>
<tr>
<th></th>
<th>t</th>
<th>h</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>t</td>
<td></td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>e</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>a</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Example

<table>
<thead>
<tr>
<th></th>
<th>t</th>
<th>h</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>t</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>e</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>a</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Extensions

• Equal cost for insertions, deletions, substitutions not necessary (or even appropriate)

• Example: DNA contains “junk” (so-called introns)

  • Insertions are expected in alignment
Smith-Waterman (1981)

\[ s(n, m) = \max \begin{cases} 
0 \\
\max_{1 \leq i \leq n}\{s(n - i, m) - W_i\} \\
\max_{1 \leq i \leq m}\{s(n, m - i) - W_i\} \\
s(n - 1, m - 1) + \text{diff}(x[n], y[m])
\end{cases} \]

- Measure of similarity instead of dissimilarity
- \( \text{diff}(x, x) > 0 \)
- Local alignment: Focus on regions with positive score
Smith-Waterman (1981)

\[ s(n, m) = \max \begin{cases} 
0 & \\
\max_{1 \leq i \leq n} \{ s(n - i, m) - W_i \} & \\
\max_{1 \leq i \leq m} \{ s(n, m - i) - W_i \} & \\
s(n - 1, m - 1) + \text{diff}(x[n], y[m]) & 
\end{cases} \]
Where is this used?

- Genome analysis for clinical use
- Treatments
- Drugs
- Clinical trials

## Pan-Cancer Somatic Panel Results

<table>
<thead>
<tr>
<th>INDICATED THERAPIES</th>
<th>CONTRAINDICATED THERAPIES</th>
<th>ACTIONABLE VARIANTS DETECTED</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0</td>
<td>22</td>
</tr>
</tbody>
</table>

**Pathologist Comments**

Signature: [Signature Field]

Date: [Date Field]
My Computer-Science Career Path

- Passion for mathematics
- Software Engineering as Hobby
- Participation in maths and programming Contests
My Computer-Science Career Path

Passion for mathematics

Software Engineering as Hobby

Participation in maths and programming Contests

Undergrad in computer science and maths
My Computer-Science Career Path

- Passion for mathematics
- Software Engineering as Hobby
- Participation in maths and programming Contests
- Undergrad in computer science and maths
- Graduate studies in theoretical CS
My Computer-Science Career Path

- Passion for mathematics
- Undergrad in computer science and maths
- Graduate studies in theoretical CS
- Software Engineering as Hobby
- Participation in maths and programming Contests
- Postdoc
My Computer-Science Career Path

- Passion for mathematics
- Software Engineering as Hobby
- Participation in maths and programming Contests
- Undergrad in computer science and maths
- Graduate studies in theoretical CS
- Postdoc
- Faculty
My Computer-Science Career Path

- Passion for mathematics
- Software Engineering as Hobby
- Participation in maths and programming Contests
- Undergrad in computer science and maths
- Graduate studies in theoretical CS
- Postdoc

Future role: Faculty
My Computer-Science Career Path

Passion for mathematics

Software Engineering as Hobby

Participation in maths and programming Contests

Undergrad in computer science and maths

Graduate studies in theoretical CS

Postdoc

First industry job: Algorithms for "Big Data"

Faculty
My Computer-Science Career Path

Passion for mathematics

Software Engineering as Hobby

Participation in maths and programming Contests

Undergrad in computer science and maths

Graduate studies in theoretical CS

Postdoc

First industry job: Algorithms for "Big Data"

Startup: Architect computational infrastructure for cancer tests

Faculty?
module AlignmentAndVariantCalling {
  in reference: FASTAFile
  in fragments: FASTQFile
  out annotations: List<AnnotatedVariant> = ga.annotations

  al = Alignment(reference = reference, fragments = fragments)
  vc = VariantCalling(genome = al.genome)
  ga = GenomicAnnotation(variants = vc.variants)
}
Algorithms for “Big Data”

Regression
Least Squares

Clustering
$k$-means
Algorithms for “Big Data”

Regression
Least Squares

Clustering
$k$-means
Algorithms for "Big Data"

- Regression
  - Least Squares

- Clustering
  - $k$-means
Selfish Routing

$t(x) = 1$

$t(x) = x$

Rational behavior but not optimal!
Take-Home Points

• Solve problems by identifying smaller subproblems

• Computer Science is way more than just coding
Take-Home Points

• Solve problems by identifying smaller subproblems

• Computer Science is way more than just coding

• We’re hiring! 😊