Dynamic Programming & Sequence Alignment

Florian Schoppmann

Computer Science for Solving Problems

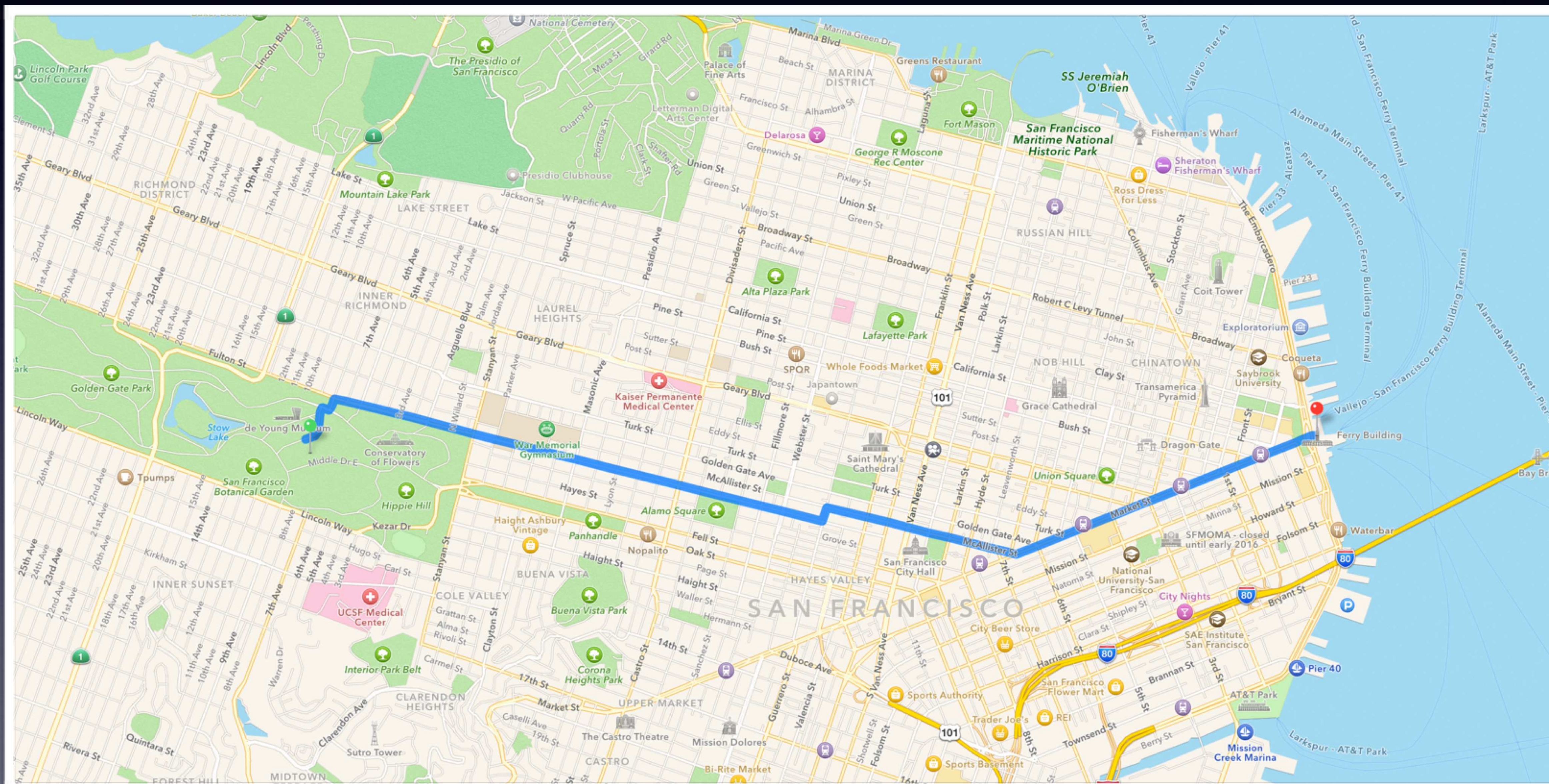
"Directions from California Academy of Sciences to Ferry Building?"

- Recurring problem
- Should have a "formula" or general scheme
- Need formal model!

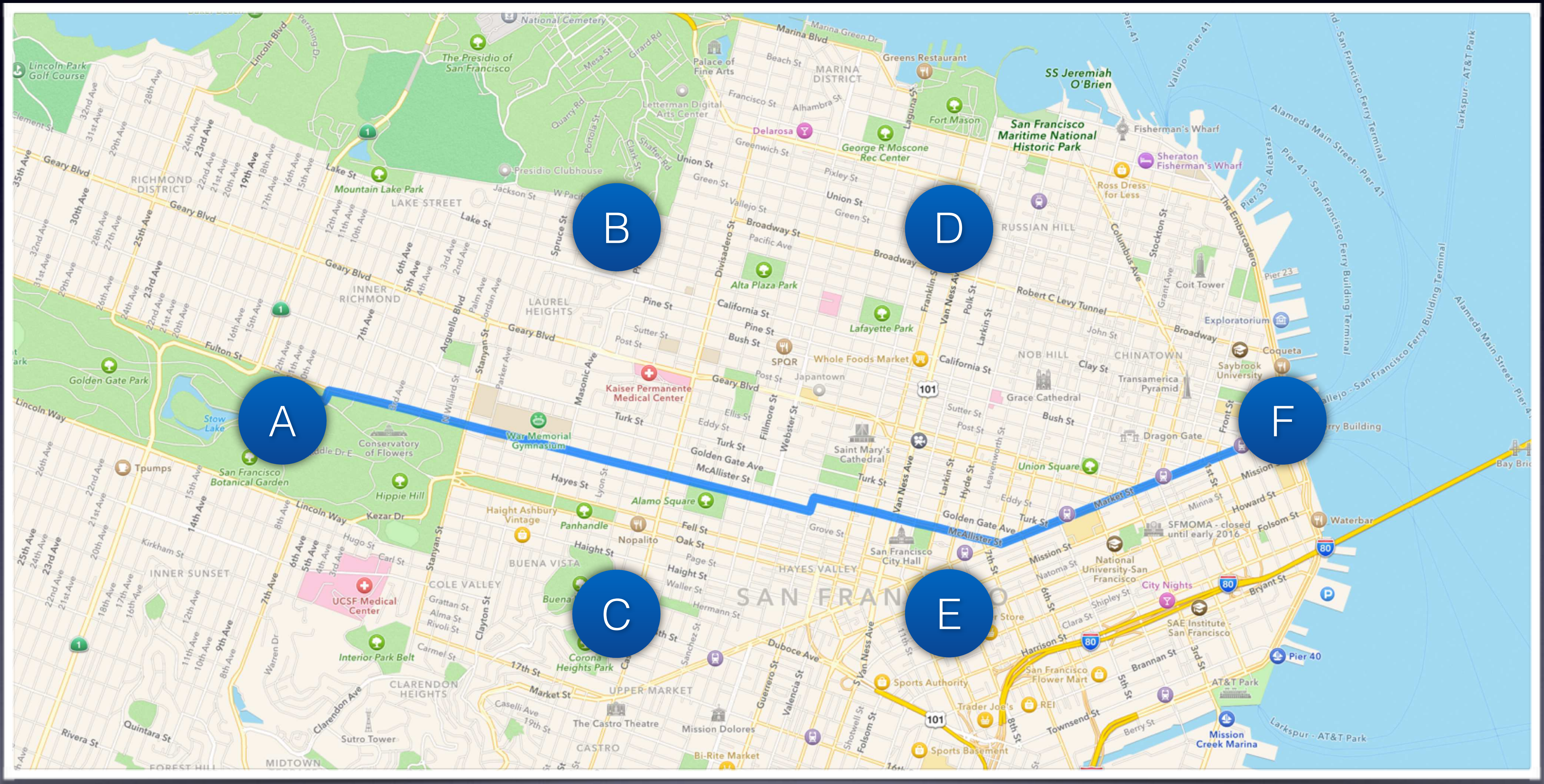
model |ˈmädl|
[…]
• a simplified description, especially a mathematical one, of a system or process, to assist calculations and predictions
[…]

New Oxford American Dictionary

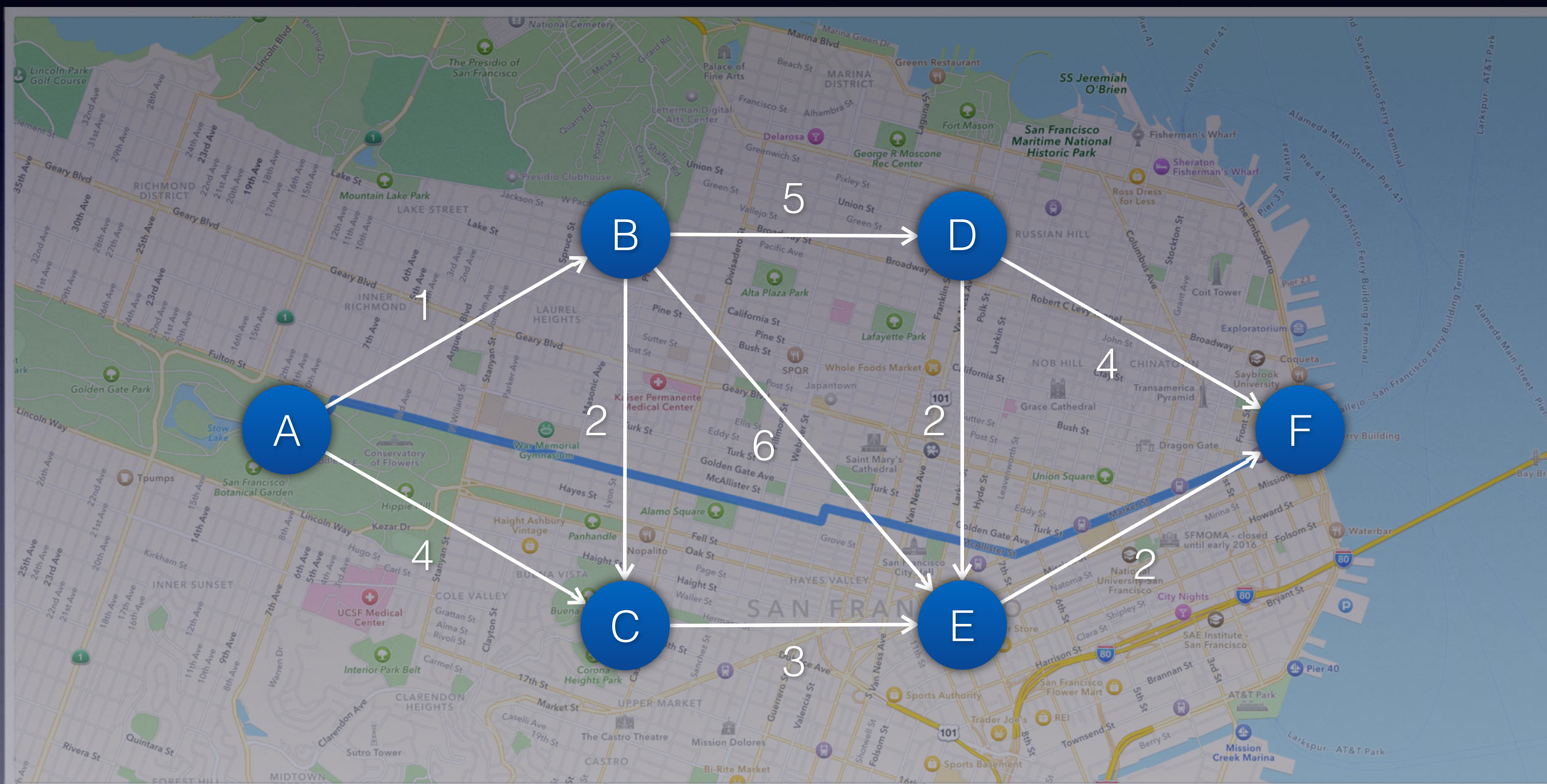
A Model for the Directions Problem



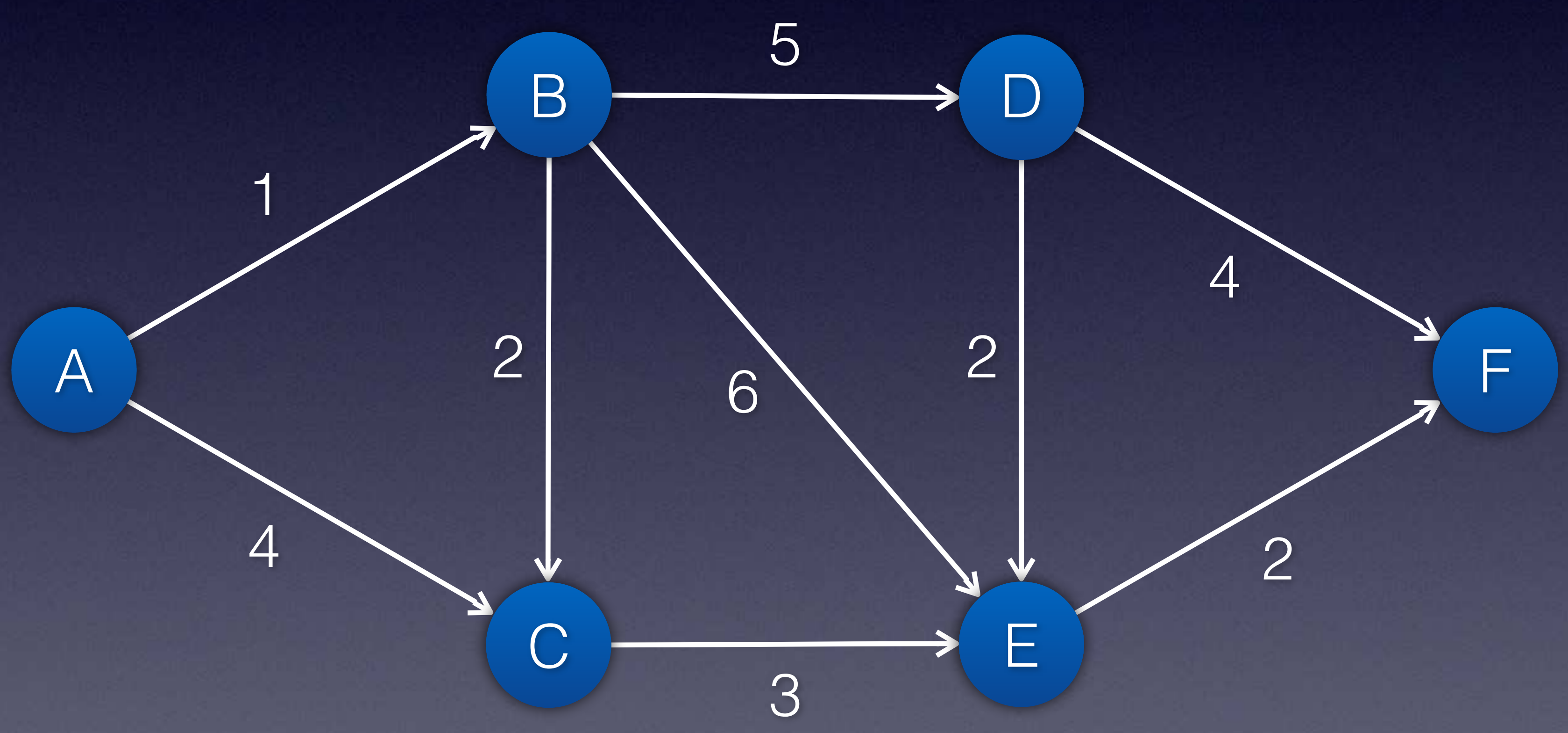
A Model for the Directions Problem



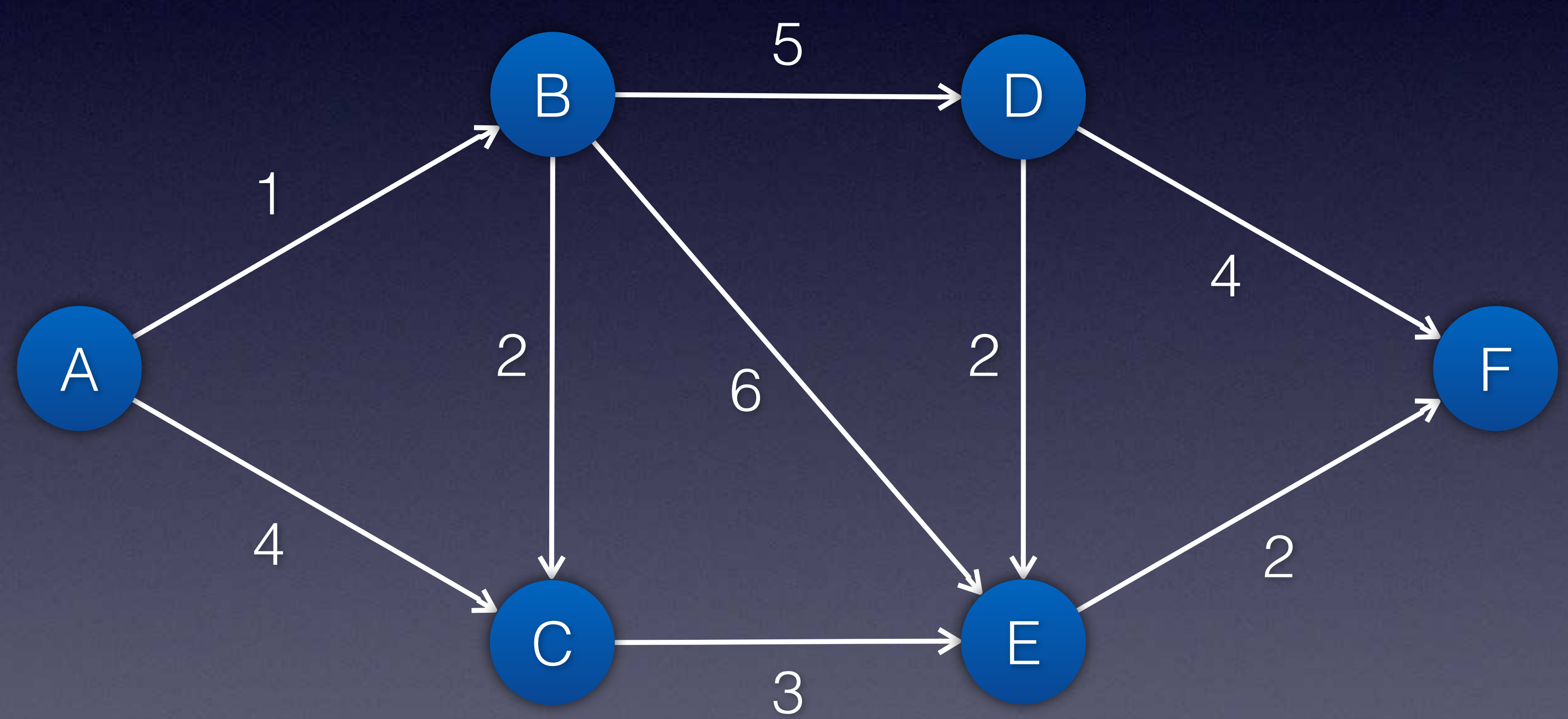
A Model for the Directions Problem



A Model for the Directions Problem



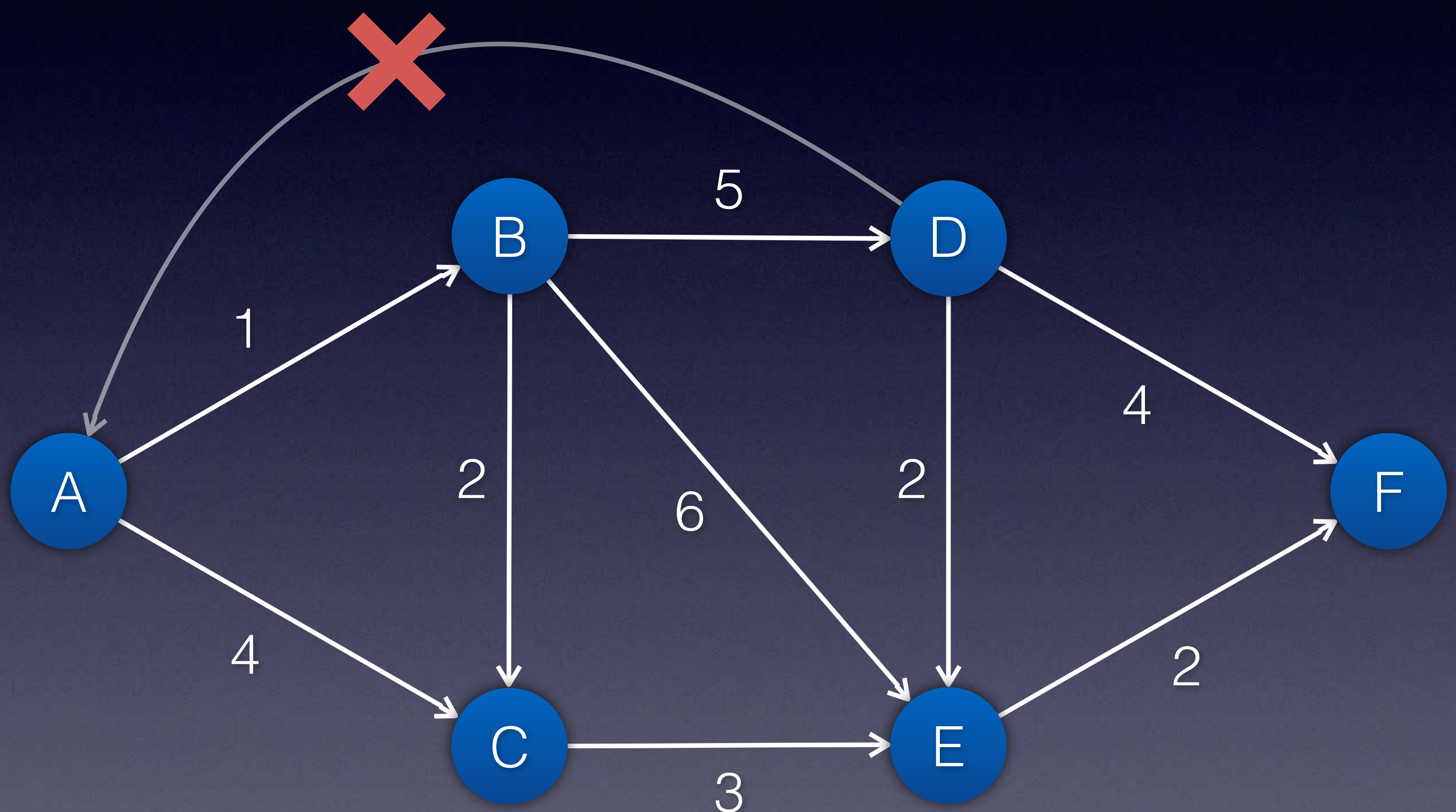
Directed Acyclic Graphs



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

edge-label function $c: E \to \{1, 2, \ldots\}$

Directed Acyclic Graphs

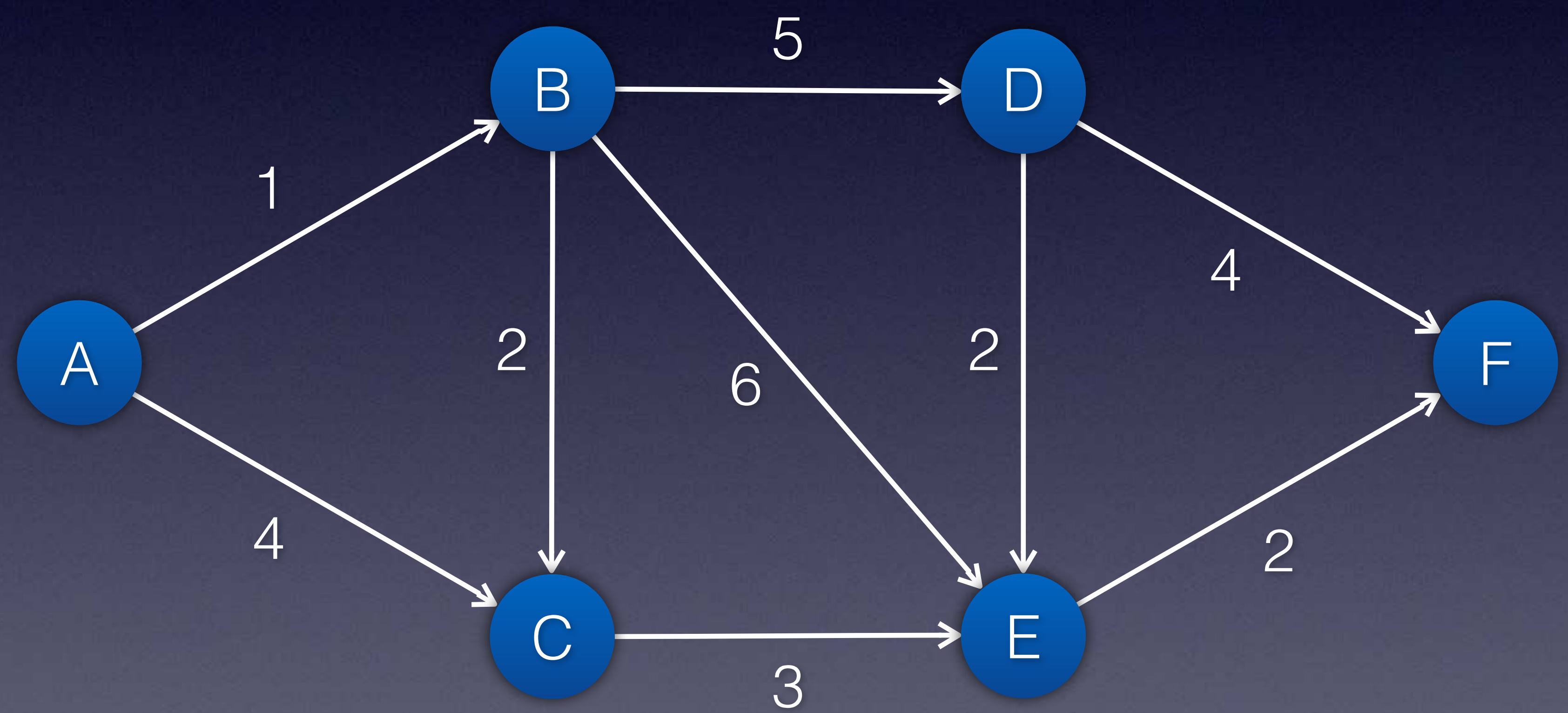


graph $G = (V, E)$ where $E \subseteq V \times V$

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

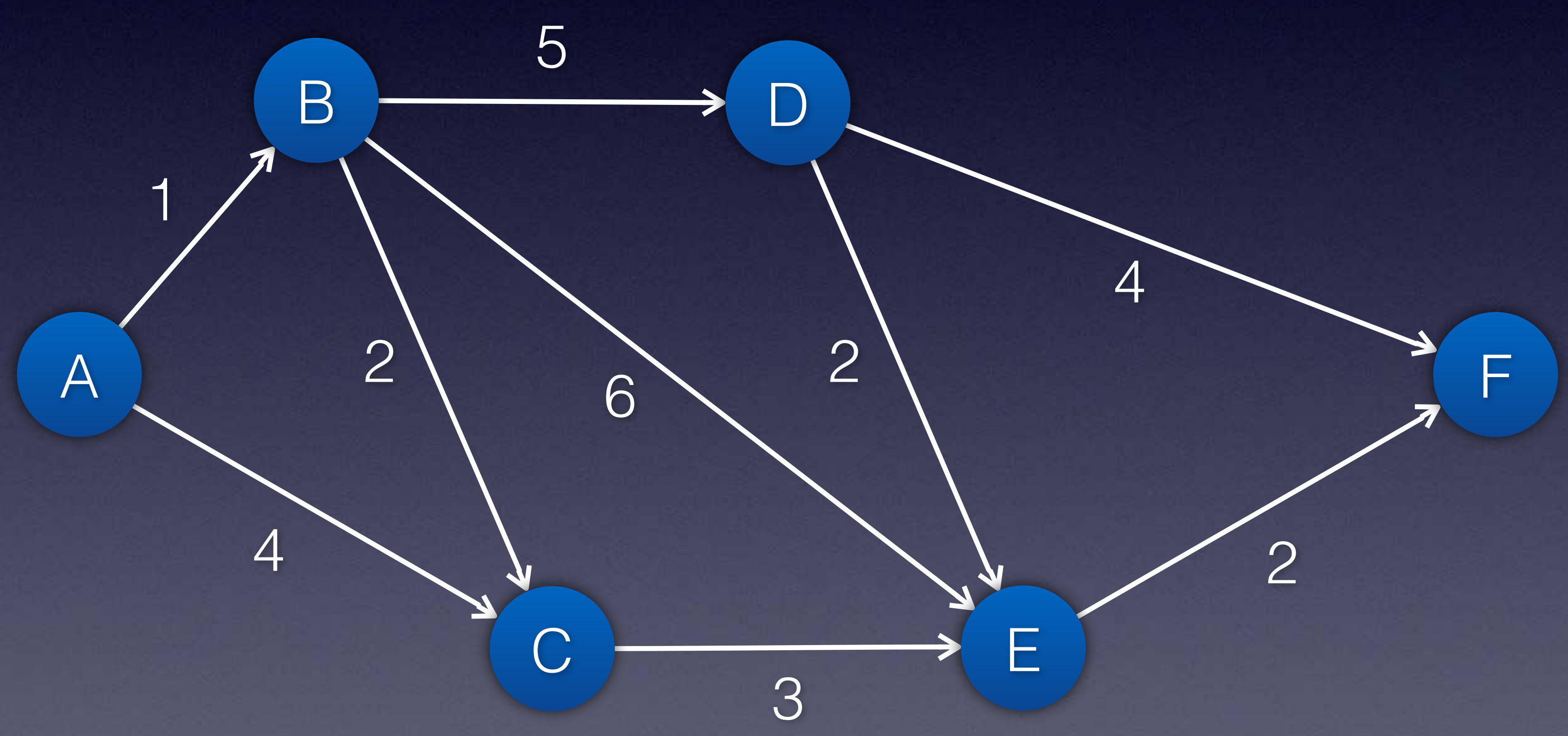
Linearizing DAGs

Can move vertices so that edges from left to right!

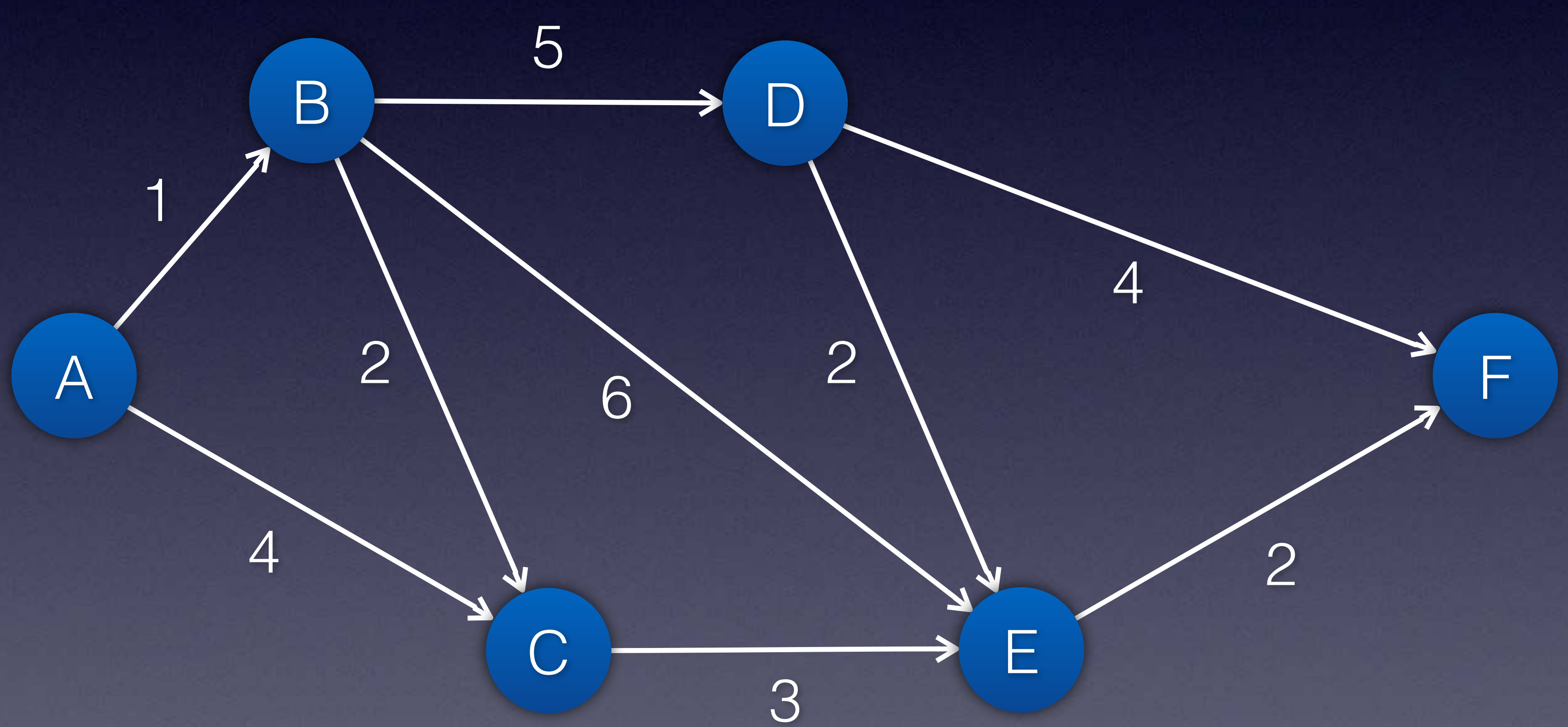


Linearizing DAGs

Can move vertices so that edges from left to right!

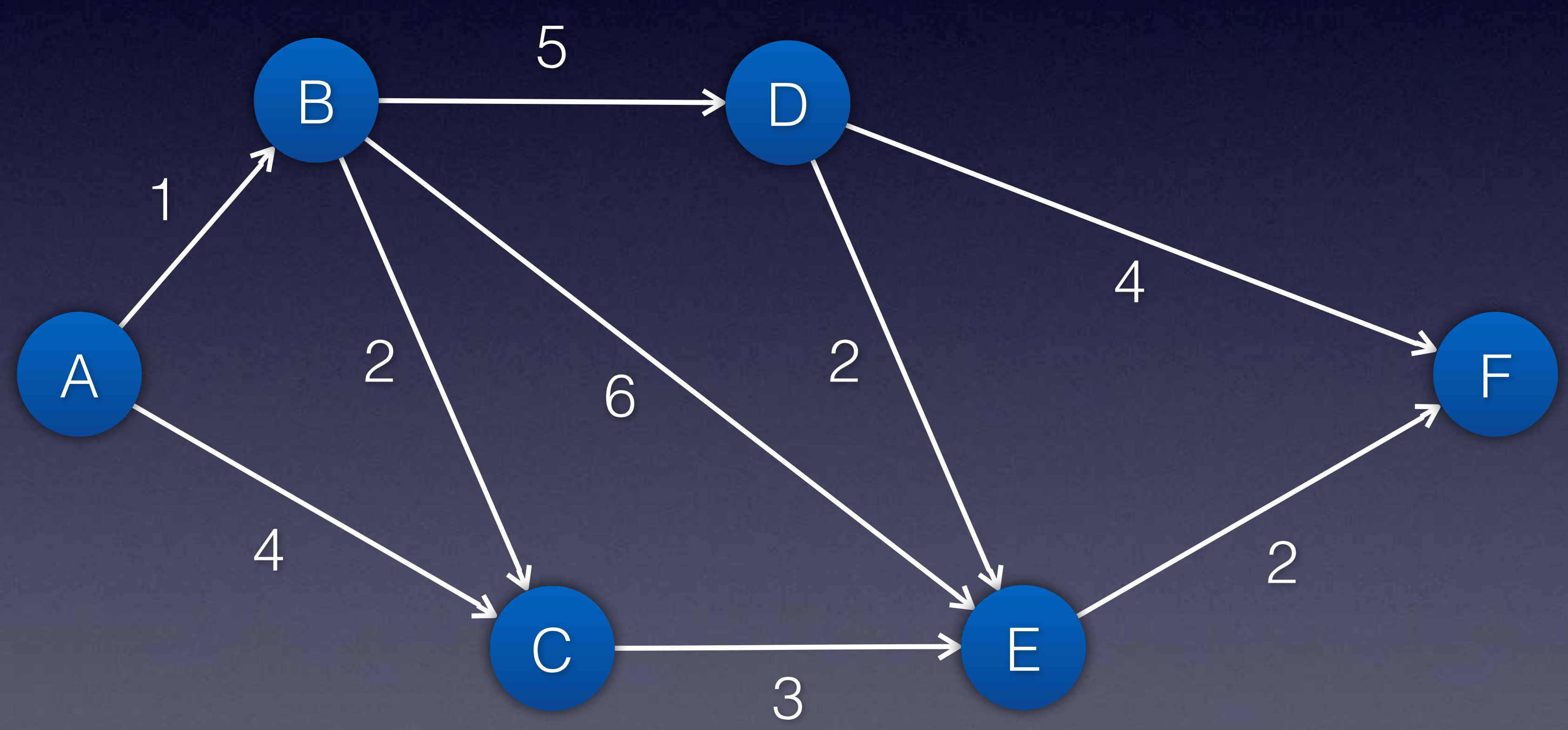


Subproblem Structure



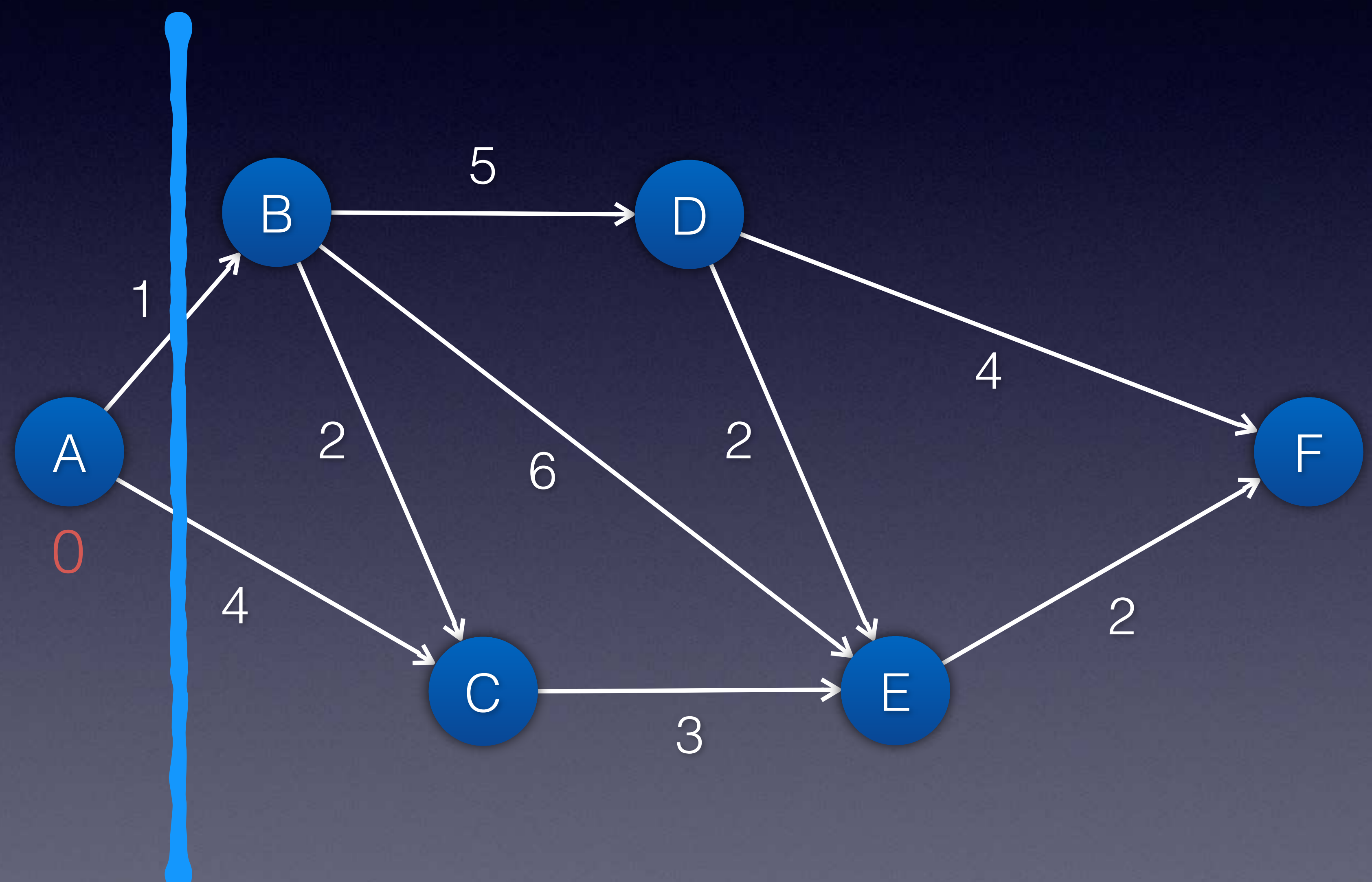
$$d(F) = \min\{d(D) + 4, d(E) + 2\}$$

A Dynamic Program for Shortest Paths



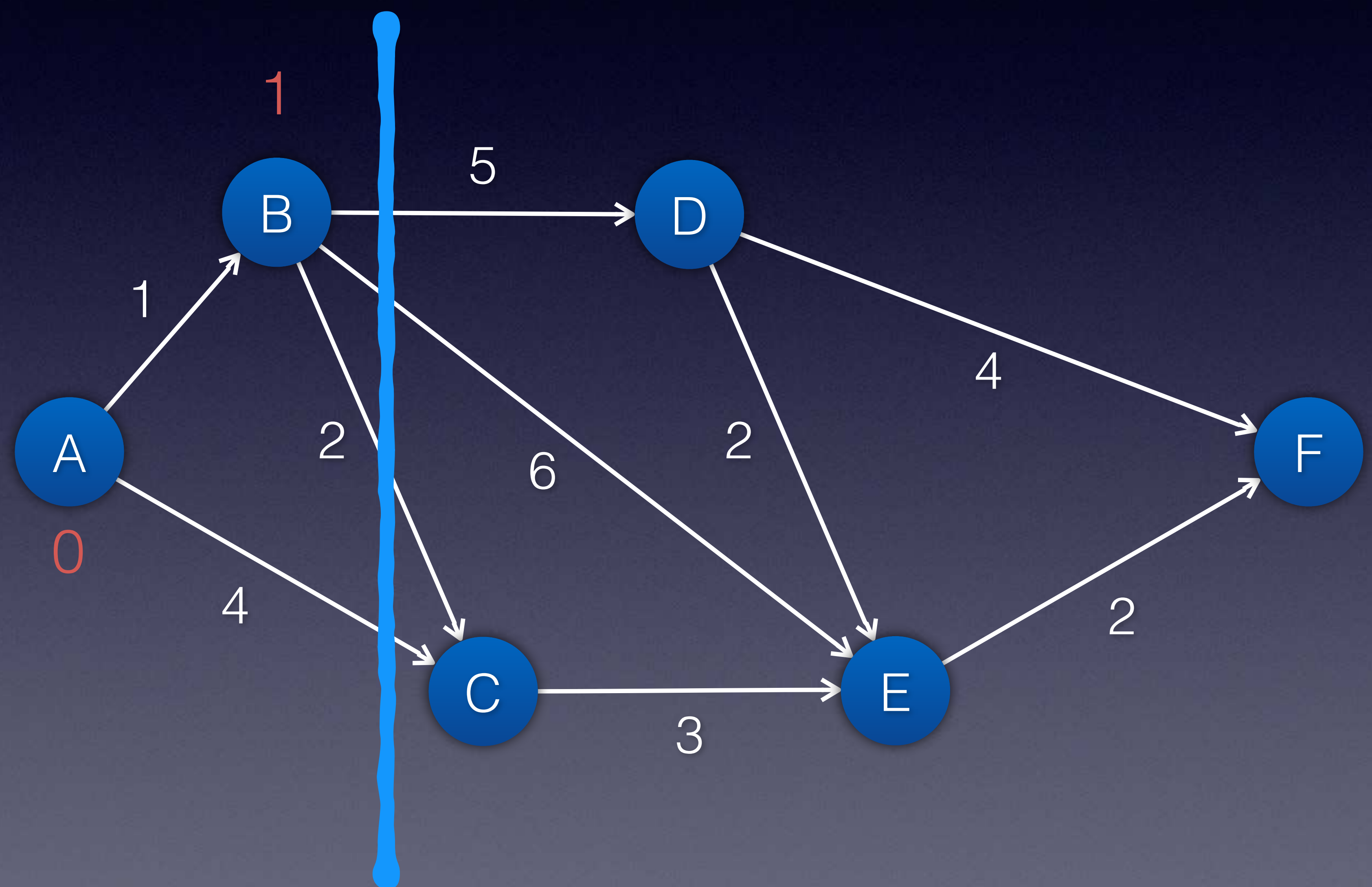
$$\forall v \in V \setminus \{A\} : d(v) = \min_{(u,v)\in E} \{d(u) + c(u,v)\}$$

A Dynamic Program for Shortest Paths



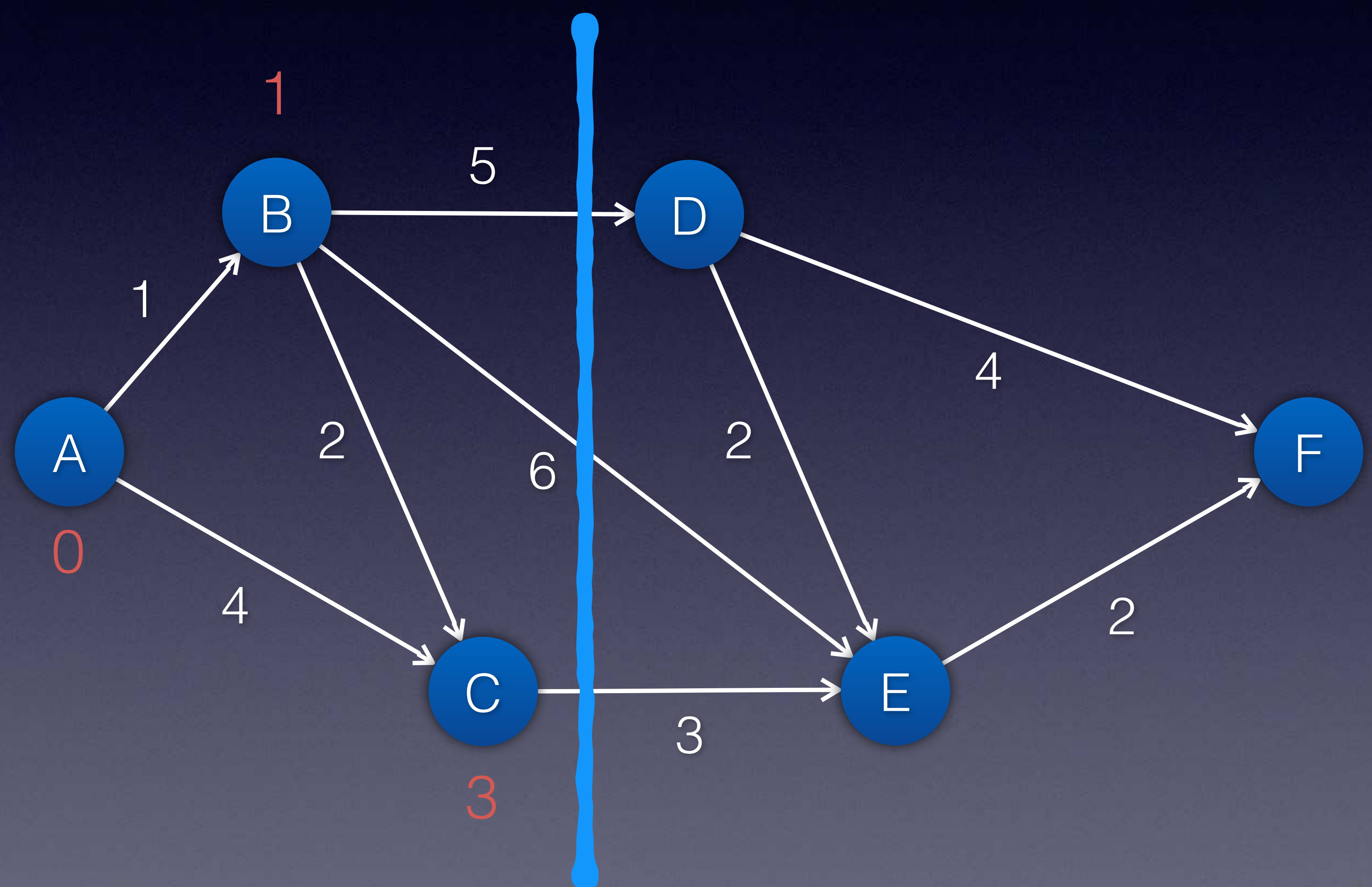
$$\forall v \in V \setminus \{A\} : d(v) = \min_{(u,v) \in E} \{d(u) + c(u,v)\}$$

A Dynamic Program for Shortest Paths



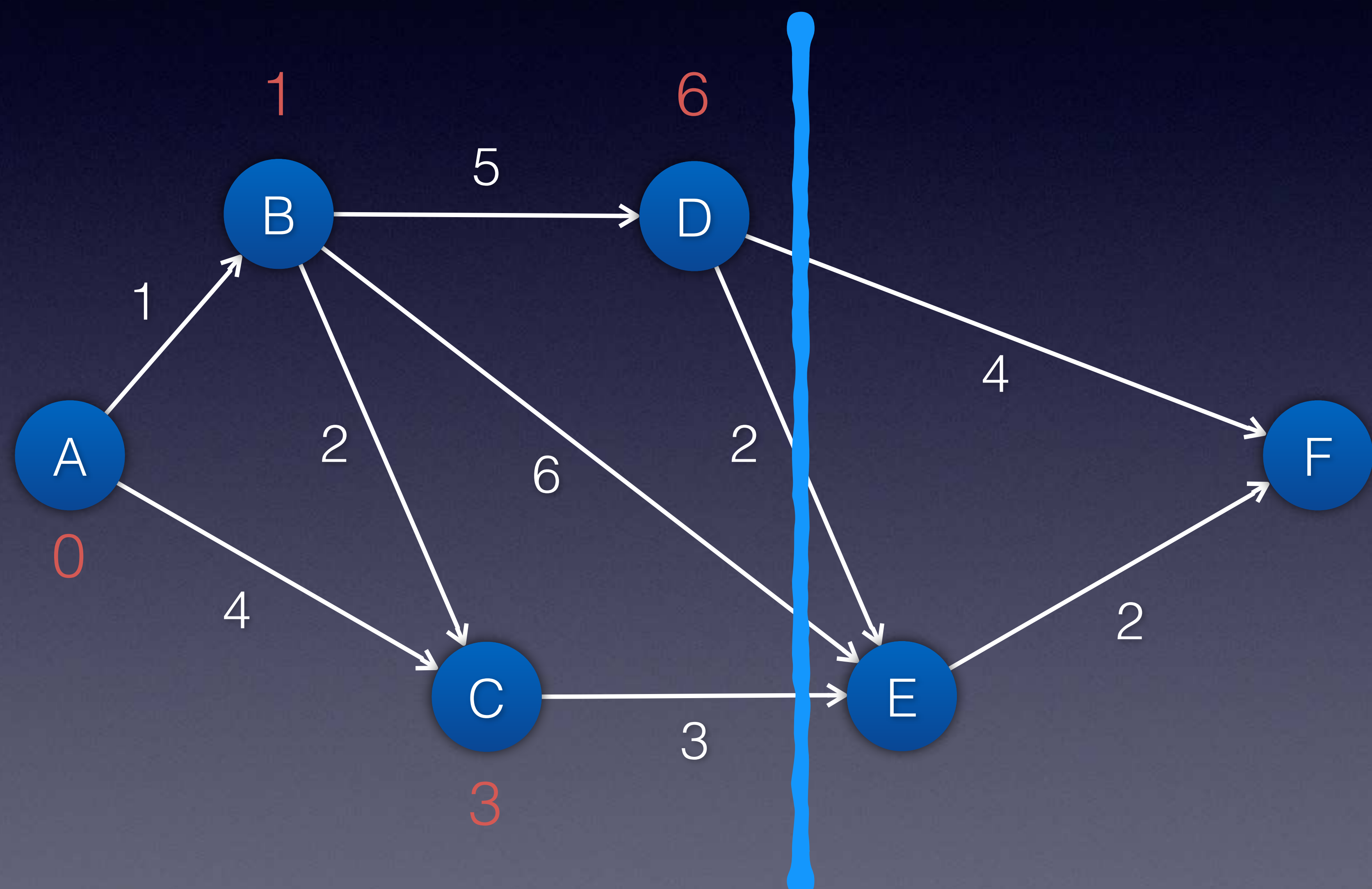
$$\forall v \in V \setminus \{A\} : d(v) = \min_{(u,v)\in E} \{d(u) + c(u,v)\}$$

A Dynamic Program for Shortest Paths



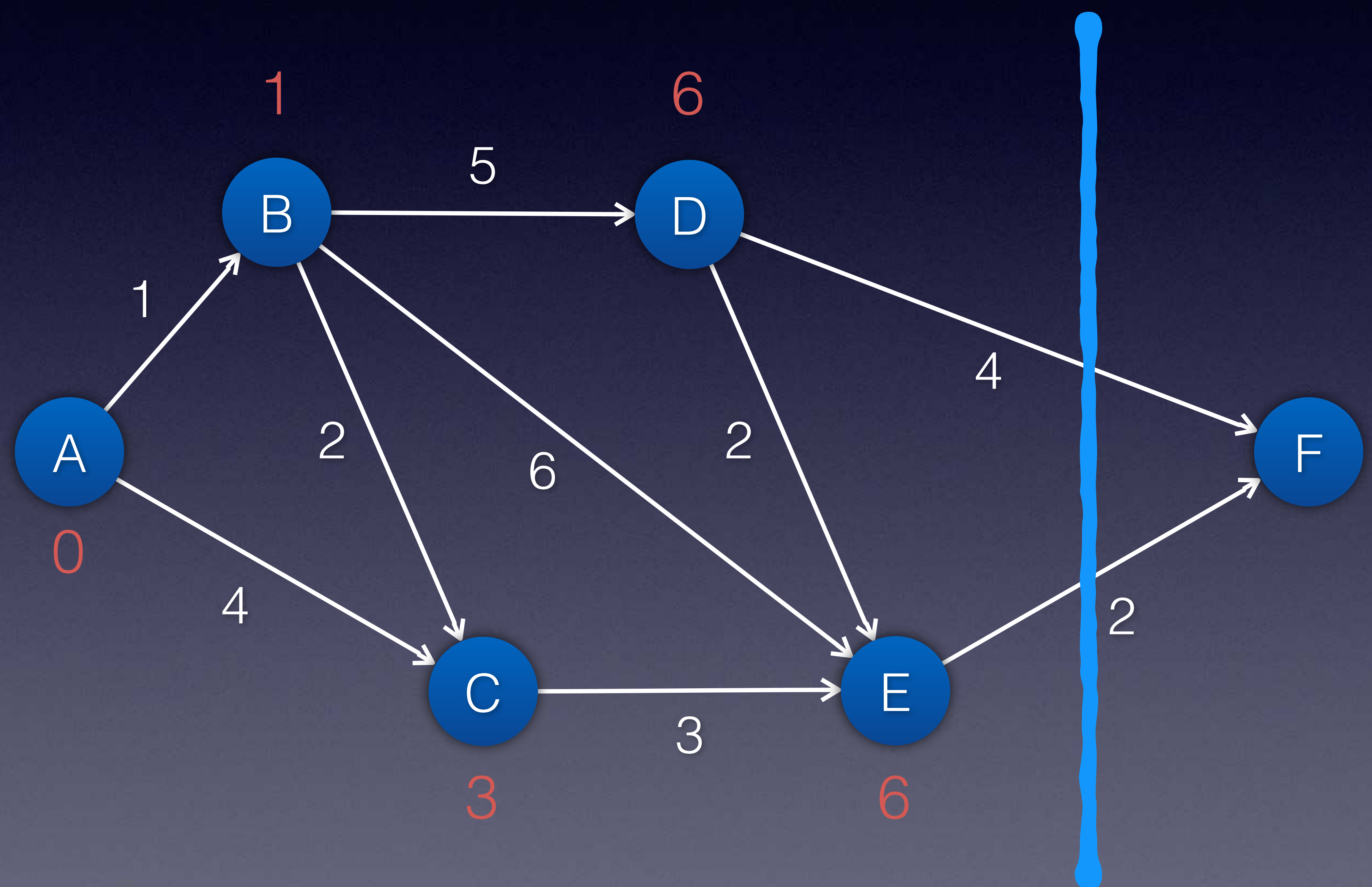
$$\forall v \in V \setminus \{A\} : d(v) = \min_{(u,v)\in E} \{d(u) + c(u,v)\}$$

A Dynamic Program for Shortest Paths



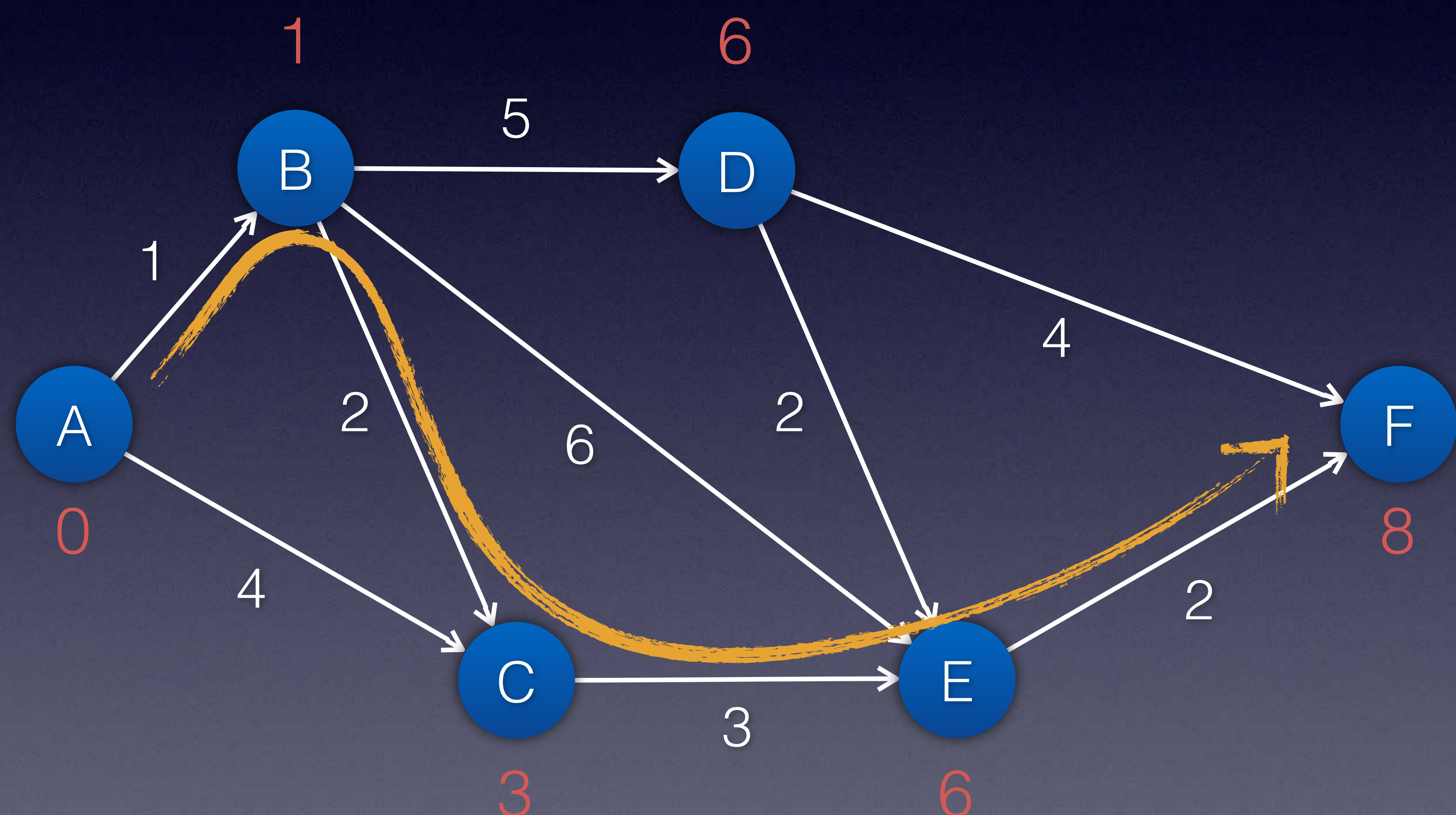
$$\forall v \in V \setminus \{A\} : d(v) = \min_{(u,v) \in E} \{d(u) + c(u,v)\}$$

A Dynamic Program for Shortest Paths



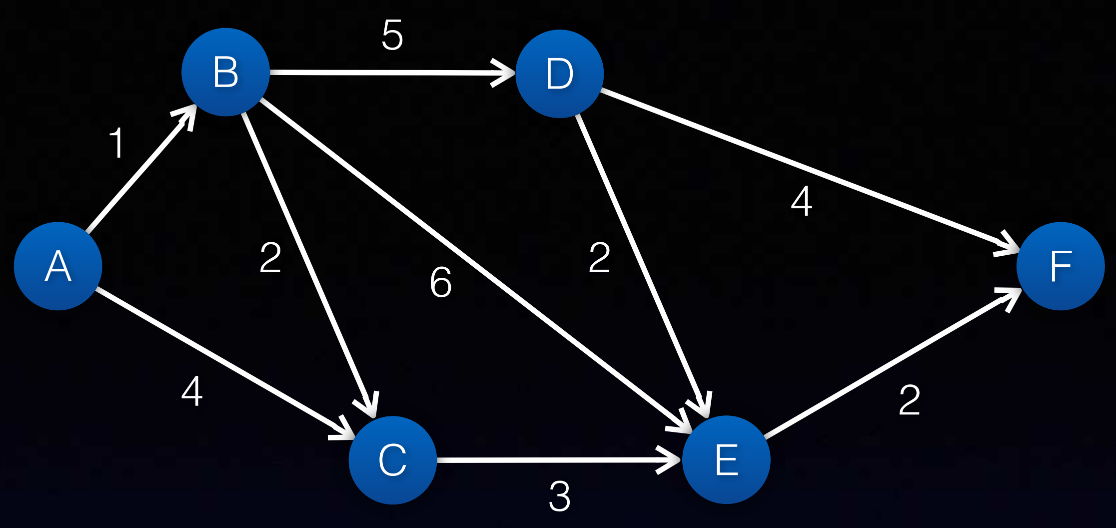
$$\forall v \in V \setminus \{A\} : d(v) = \min_{(u,v) \in E} \{d(u) + c(u, v)\}$$

A Dynamic Program for Shortest Paths



$$\forall v \in V \setminus \{A\} : d(v) = \min_{(u,v)\in E} \{d(u) + c(u,v)\}$$

Subproblem DAG



- Vertex ≈ (optimization) problem
- Predecessor vertex ≈ 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

$$F_n = F_{n-1} + F_{n-2}$$

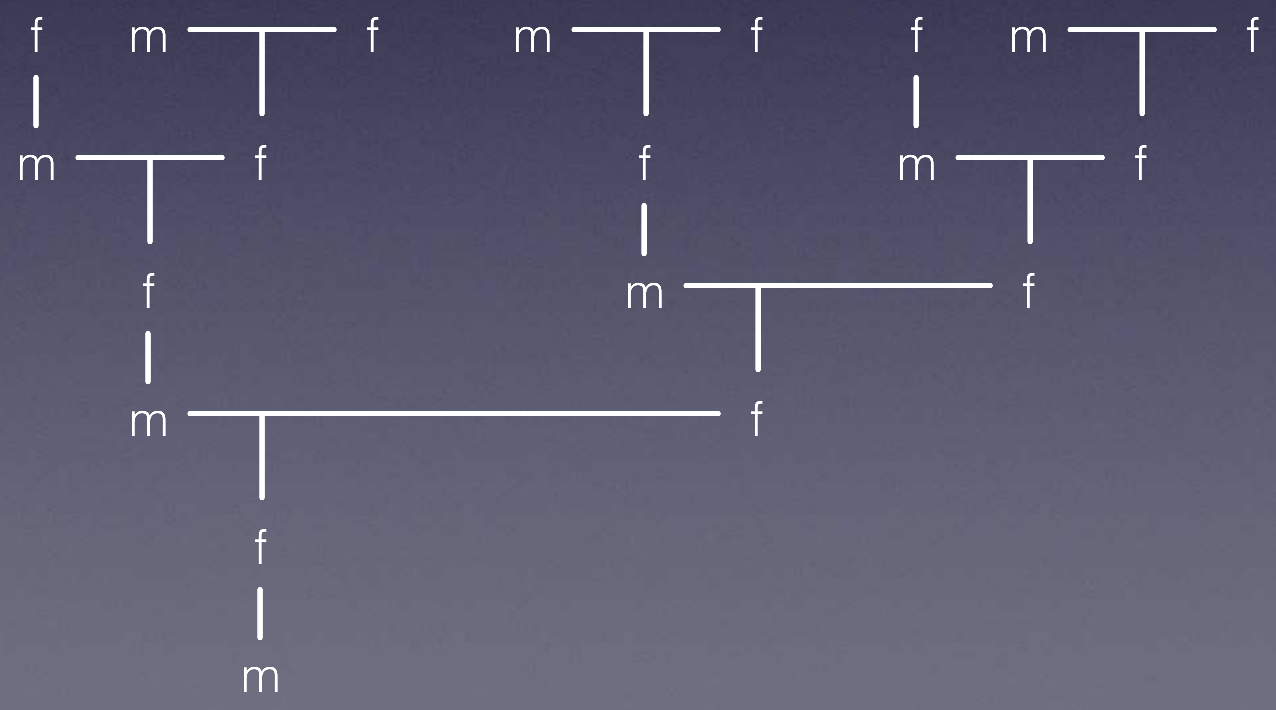
$$F_1 = 1 \text{ and } F_0 = 0$$

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

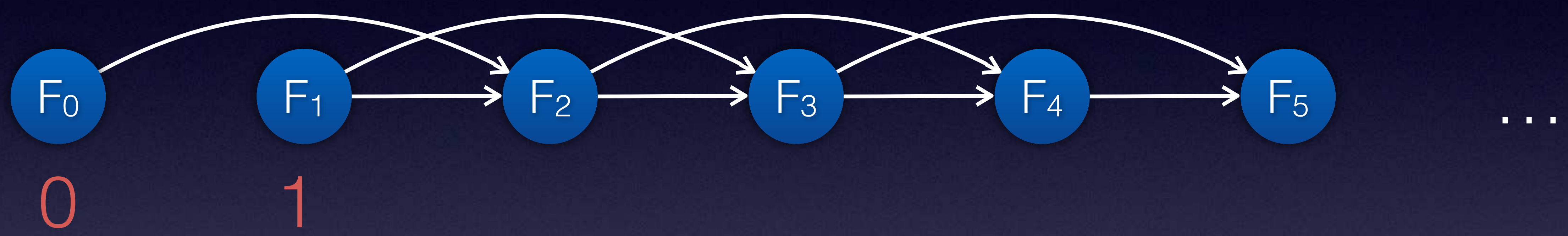
$$F_n = F_{n-1} + F_{n-2}$$

$$F_1 = 1 \text{ and } F_0 = 0$$

This Java code is excruciatingly slow! Why?

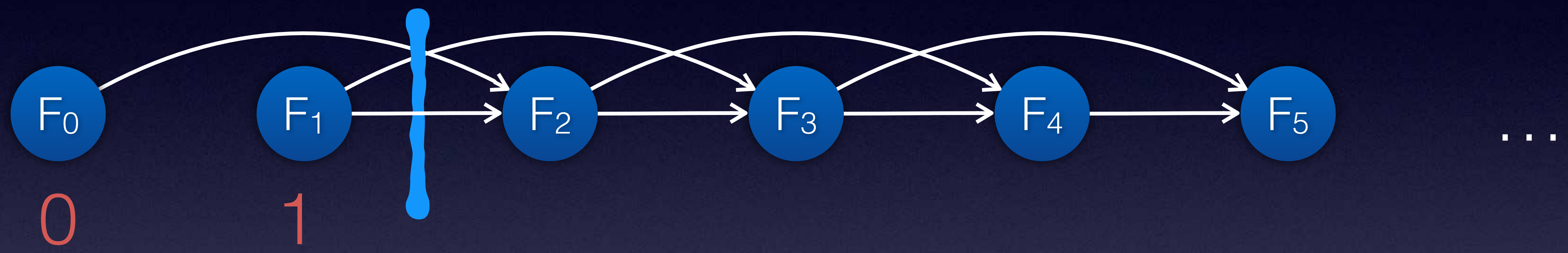
```
long fib(int n) {
    if (n == 0) {
        return 0;
    } else if (n == 1) {
        return 1;
    } else {
        return fib(n - 1) + fib(n - 2);
    }
}
```

“Bottom-up” Dynamic Program



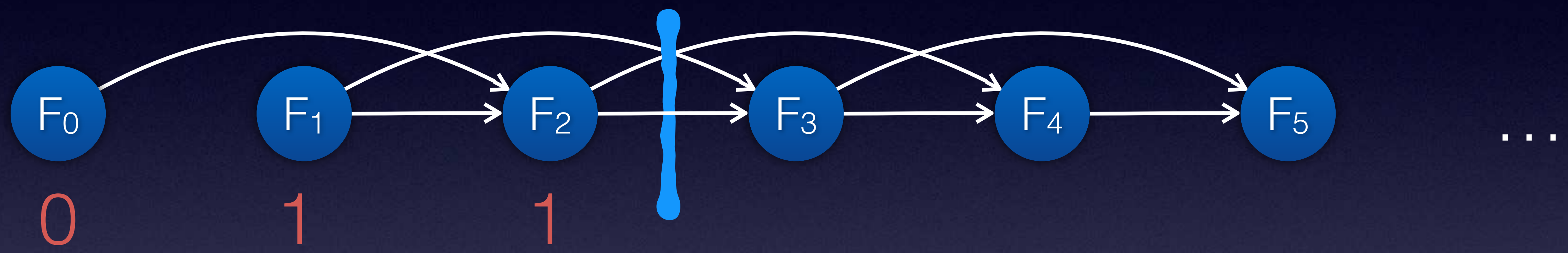
- Subproblem DAG is implicit
- Operation on subproblem results is just addition

"Bottom-up" Dynamic Program



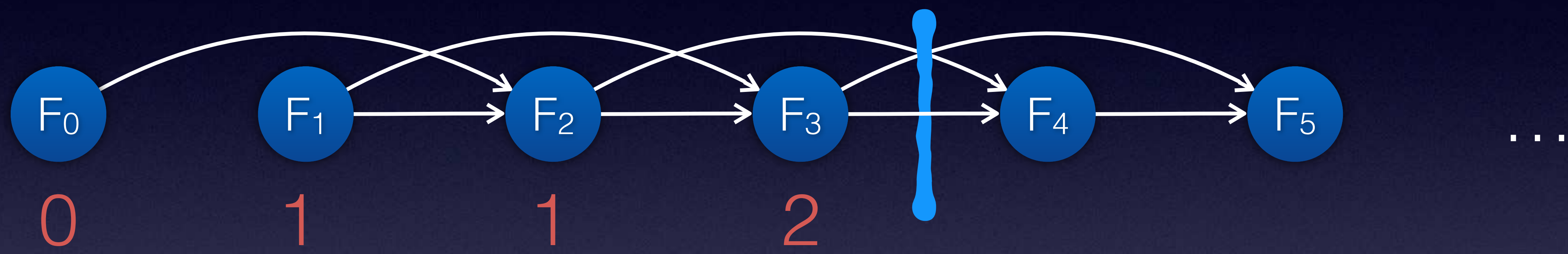
- Subproblem DAG is implicit
- Operation on subproblem results is just addition

"Bottom-up" Dynamic Program



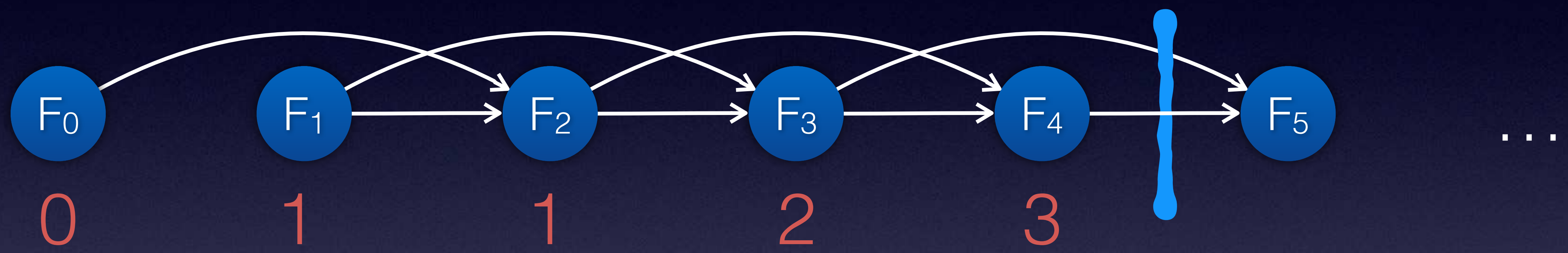
- Subproblem DAG is implicit
- Operation on subproblem results is just addition

"Bottom-up" Dynamic Program



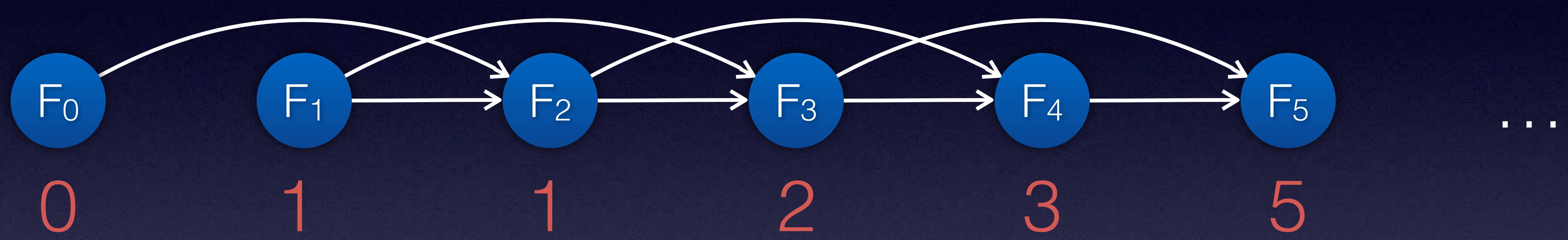
- Subproblem DAG is implicit
- Operation on subproblem results is just addition

“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 ≈ planning over time
- Secretary of Defense hostile to mathematical research

> […] it's impossible to use the word dynamic in a pejorative sense. […] It was something not even a Congressman could object to. […]
>
> *Eye of the Hurricane, An Autobiography* (1984)

Edit Distance

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

t	h	e
t	e	a

t	h	e	–
t	–	e	a

t	h	e	–	–	–
–	–	–	t	e	a

- Edit distance between “the” and “tea” = 2

Formal Problem Definition

- Input: Sequences $x\,[1..n]$ and $y\,[1..m]$

x_1	x_2	…	x_n

y_1	y_2	…	y_m

- Output: length d of a minimum-length alignment (note: $0 \leq n + m \leq d$)

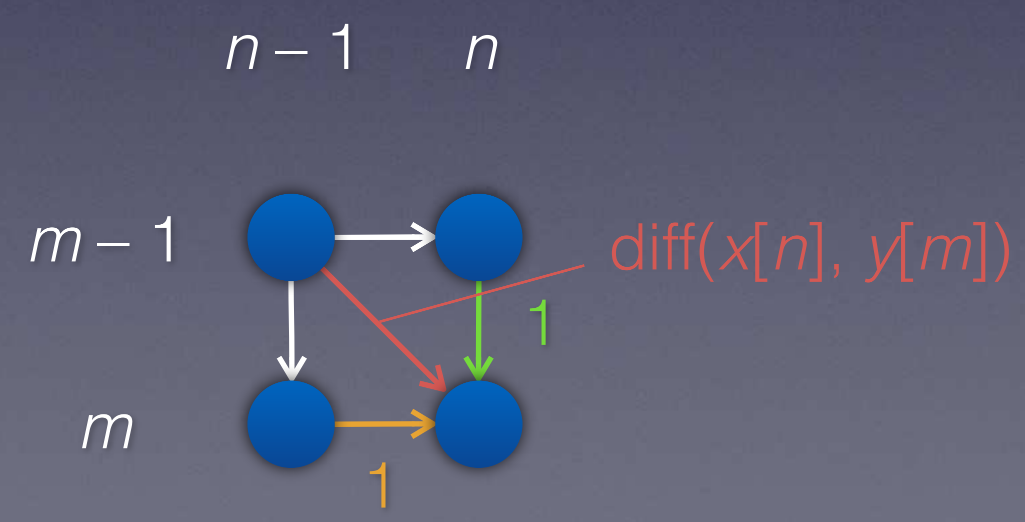
Where is the Subproblem DAG?

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

$x[1\ldots n-1]$	$x[n]$
$y[1\ldots m-1]$	$y[m]$

$x[1\ldots n-1]$	$x[n]$
$y[1\ldots m]$	–

$x[1\ldots n]$	–
$y[1\ldots m-1]$	$y[m]$



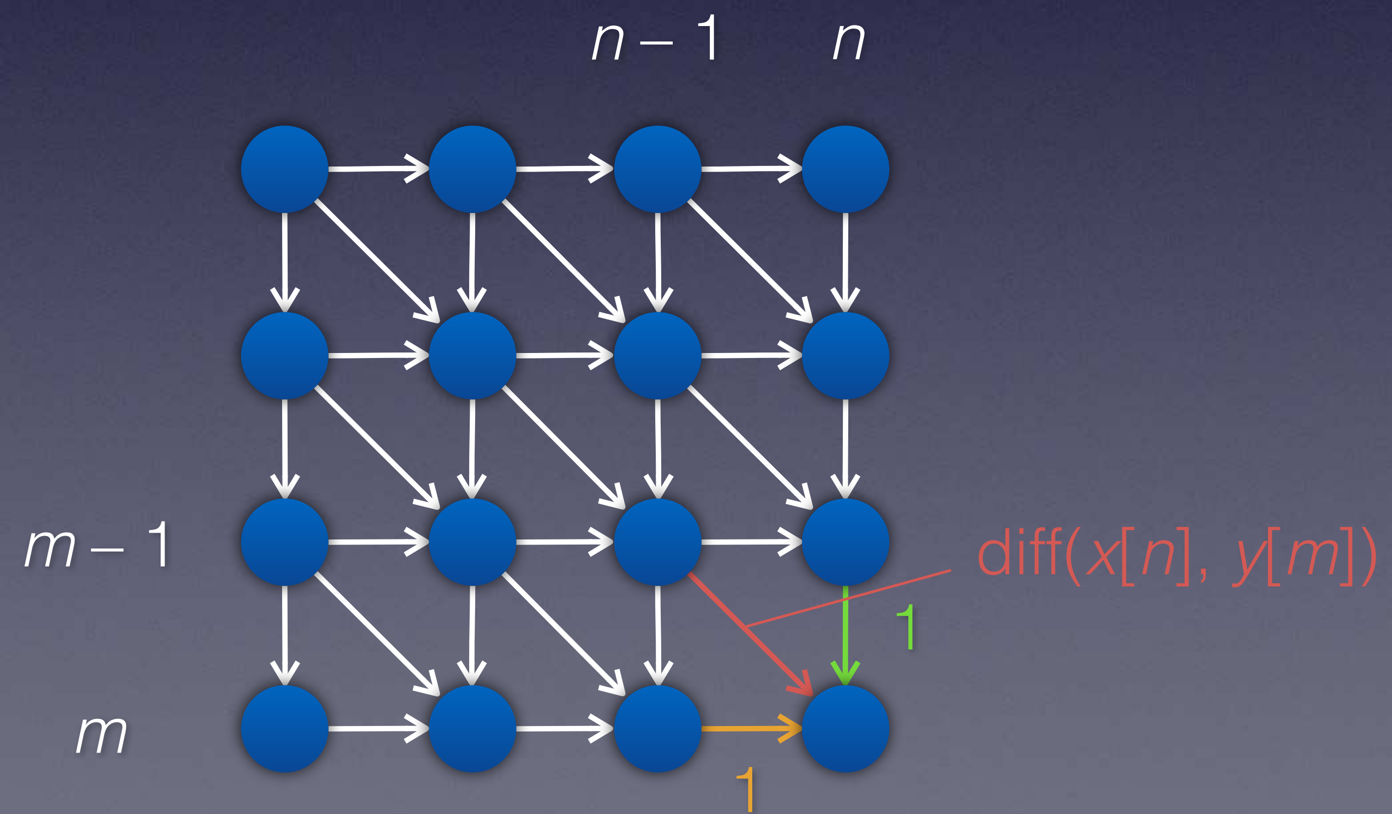
Where is the Subproblem DAG?

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

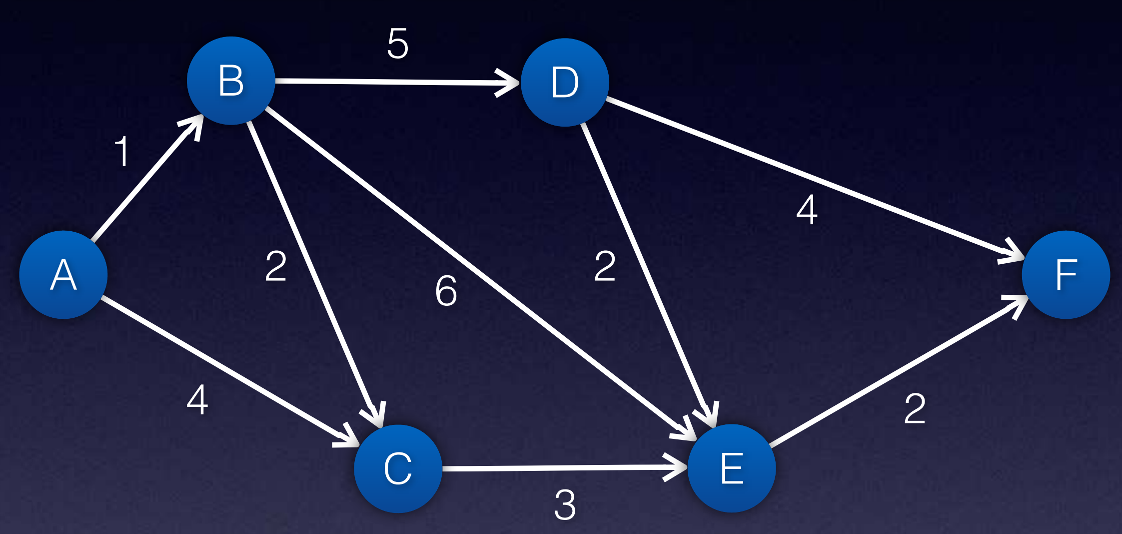
$x[1\ldots n-1]$	$x[n]$
$y[1\ldots m-1]$	$y[m]$

$x[1\ldots n-1]$	$x[n]$
$y[1\ldots m]$	–

$x[1\ldots n]$	–
$y[1\ldots m-1]$	$y[m]$



Recall: Optimal Substructure



- Let u be predecessor (subproblem) of v
- $d(v) = d(u) + c(u, v)$
  $\Leftrightarrow$ u on shortest path from A to v

Edit Distance Has Optimal Substructure

An optimal alignment has optimal sub-alignments

t	h	e
t	e	a

t	h
t	e

e
a

$$d(3,3) \quad = \quad d(2,2) + \text{diff}(x[3],\ y[3])$$

2   1   1

A Dynamic Program for Edit Distance

$x[1 \ldots n-1]$	$x[n]$
$y[1 \ldots m-1]$	$y[m]$

$x[1 \ldots n-1]$	$x[n]$
$y[1 \ldots m]$	–

$x[1 \ldots n]$	–
$y[1 \ldots m-1]$	$y[m]$

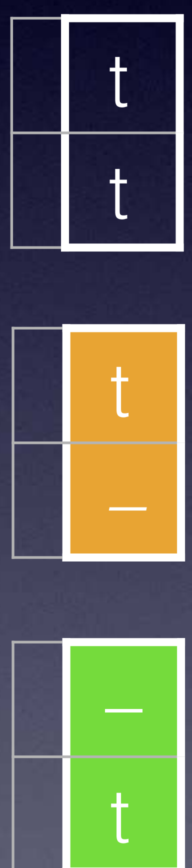
$$d(i, 0) = i \quad \text{and} \quad d(0, j) = j$$

$$d(n, m) = \min \left\{ \begin{array}{l} d(n-1, m) + 1, \\ d(n, m-1) + 1, \\ d(n-1, m-1) + \text{diff}(x[n], y[m]) \end{array} \right\}$$

Example

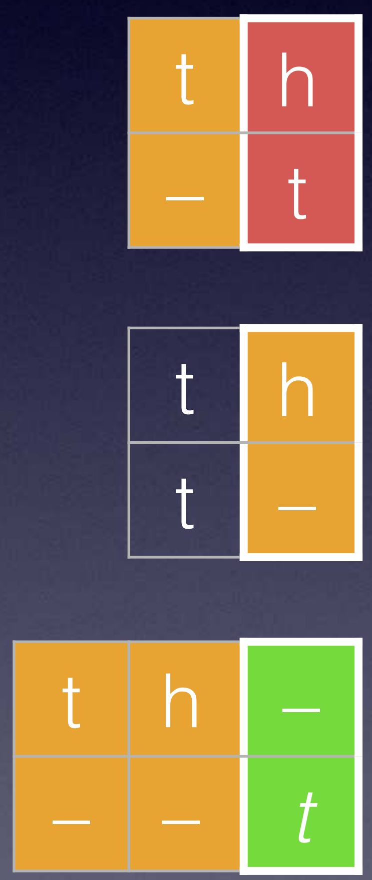
		t	h	e
	0	1	2	3
t	1			
e	2			
a	3			

Example



		t	h	e
	0	1	2	3
t	1			
e	2			
a	3			

Example



		t	h	e
	0	1	2	3
t	1	0		
e	2			
a	3			

Example

		t	h	e
	0	1	2	3
t	1	0	1	
e	2			
a	3			

Example

		t	h	e
	0	1	2	3
t	1	0	1	2
e	2			
a	3			

Example

		t	h	e
	0	1	2	3
t	1	0	1	2
e	2	1		
a	3			

Example

		t	h	e
	0	1	2	3
t	1	0	1	2
e	2	1	1	
a	3			

Example

		t	h	e
	0	1	2	3
t	1	0	1	2
e	2	1	1	1
a	3			

Example

		t	h	e
	0	1	2	3
t	1	0	1	2
e	2	1	1	1
a	3	2		

Example

		t	h	e
	0	1	2	3
t	1	0	1	2
e	2	1	1	1
a	3	2	2	

Example

		t	h	e
	0	1	2	3
t	1	0	1	2
e	2	1	1	1
a	3	2	2	2

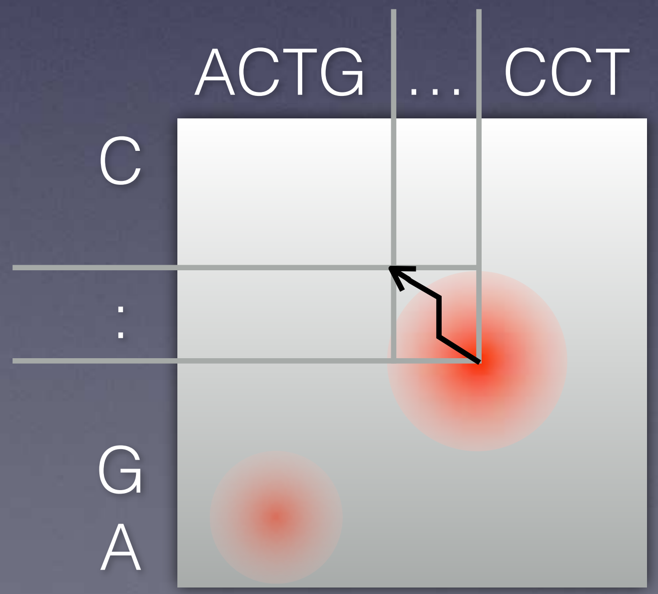
Extensions

- 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)

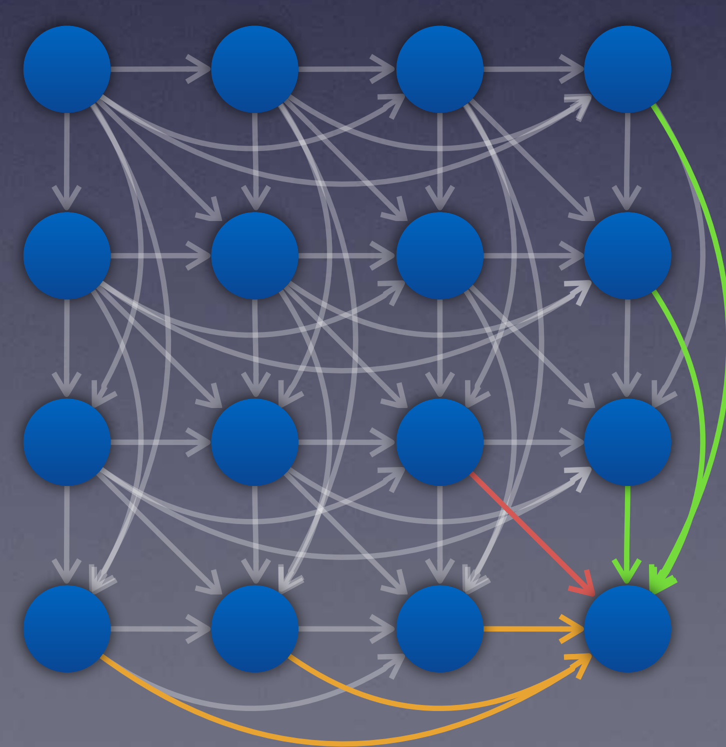
$$s(n,m) = \max \left\{ \begin{array}{l} 0 \\ \max_{1 \le i \le n} \{ s(n-i, m) - W_i \} \\ \max_{1 \le i \le m} \{ s(n, m-i) - W_i \} \\ s(n-1, m-1) + \mathrm{diff}(x[n], y[m]) \end{array} \right\}$$

- Measure of similarity instead of dissimilarity
  - diff(x, x) > 0
- Local alignment: Focus on regions with positive score



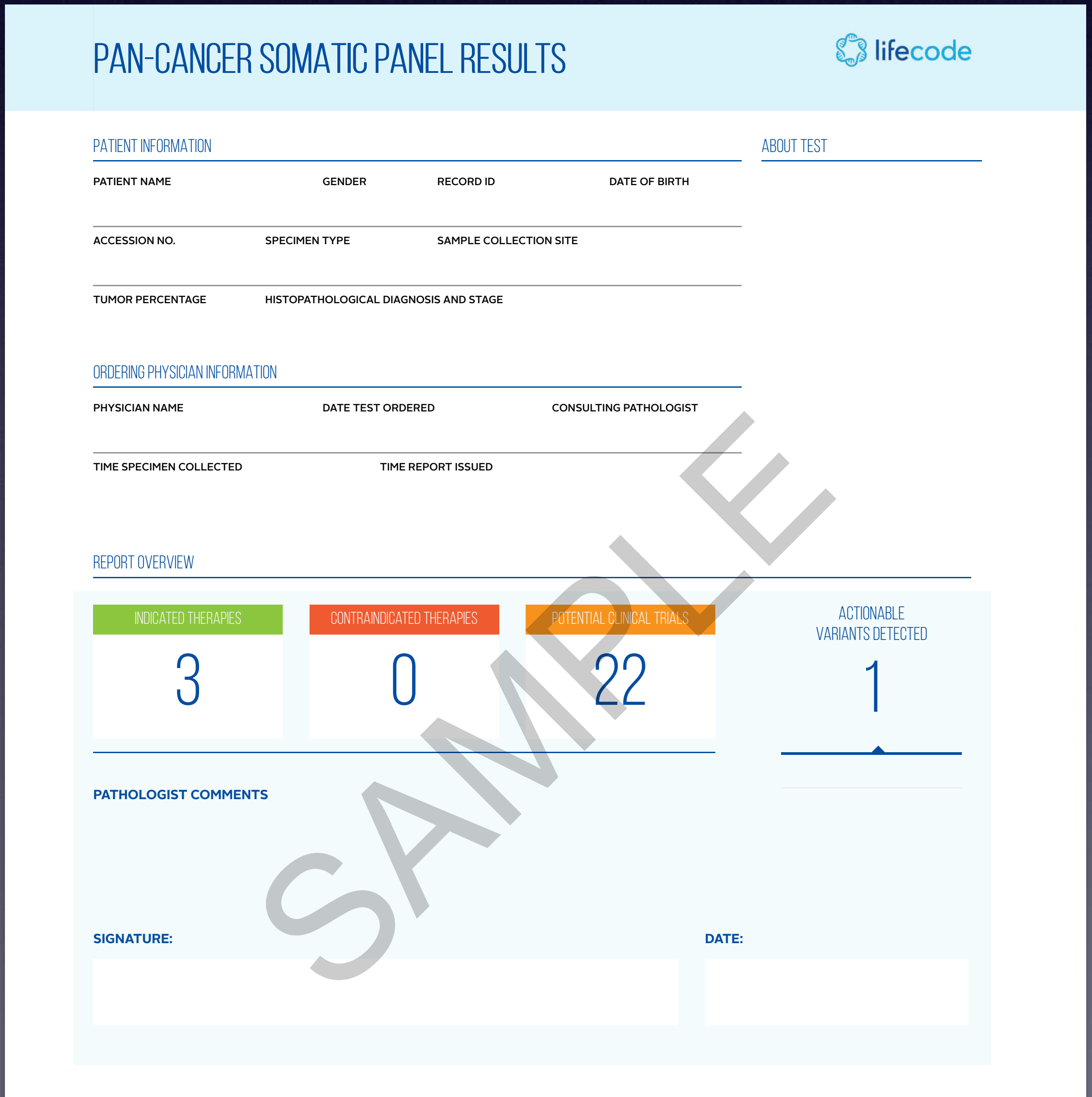
Smith-Waterman (1981)

$$s(n,m) = \max \left\{ \begin{array}{l} 0 \\ \max_{1 \le i \le n} \{ s(n-i, m) - W_i \} \\ \max_{1 \le i \le m} \{ s(n, m-i) - W_i \} \\ s(n-1, m-1) + \mathrm{diff}(x[n], y[m]) \end{array} \right\}$$



Where is this used?

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


PAN-CANCER SOMATIC PANEL RESULTS
lifecode

PATIENT INFORMATION

PATIENT NAME | GENDER | RECORD ID | DATE OF BIRTH

ACCESSION NO. | SPECIMEN TYPE | SAMPLE COLLECTION SITE

TUMOR PERCENTAGE | HISTOPATHOLOGICAL DIAGNOSIS AND STAGE

ABOUT TEST

ORDERING PHYSICIAN INFORMATION

PHYSICIAN NAME | DATE TEST ORDERED | CONSULTING PATHOLOGIST

TIME SPECIMEN COLLECTED | TIME REPORT ISSUED

REPORT OVERVIEW

INDICATED THERAPIES	CONTRAINDICATED THERAPIES	POTENTIAL CLINICAL TRIALS	ACTIONABLE VARIANTS DETECTED
3	0	22	1

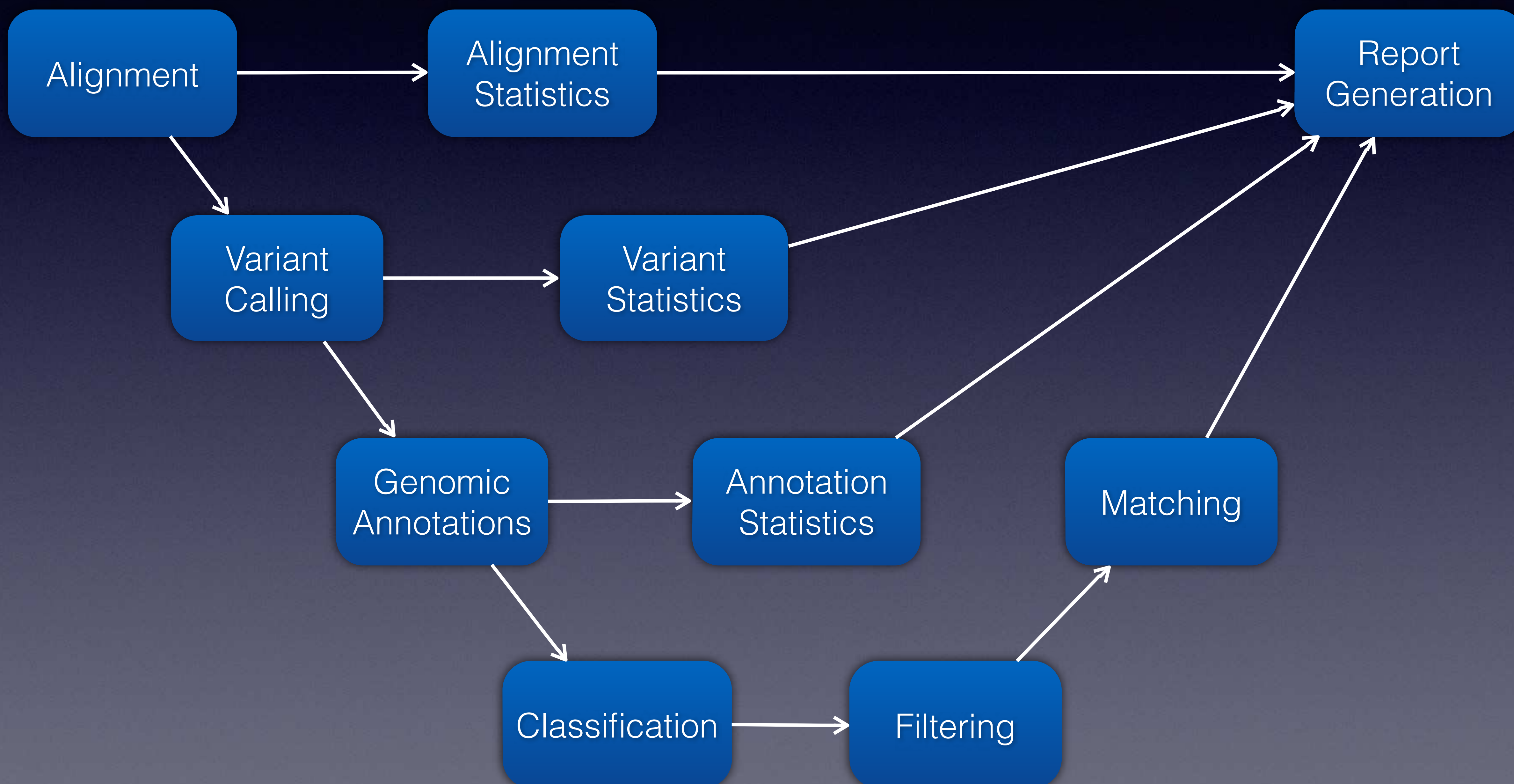
PATHOLOGIST COMMENTS

SIGNATURE:

DATE:

SAMPLE

lifecode


Alignment
Alignment Statistics
Report Generation
Variant Calling
Variant Statistics
Genomic Annotations
Annotation Statistics
Matching
Classification
Filtering

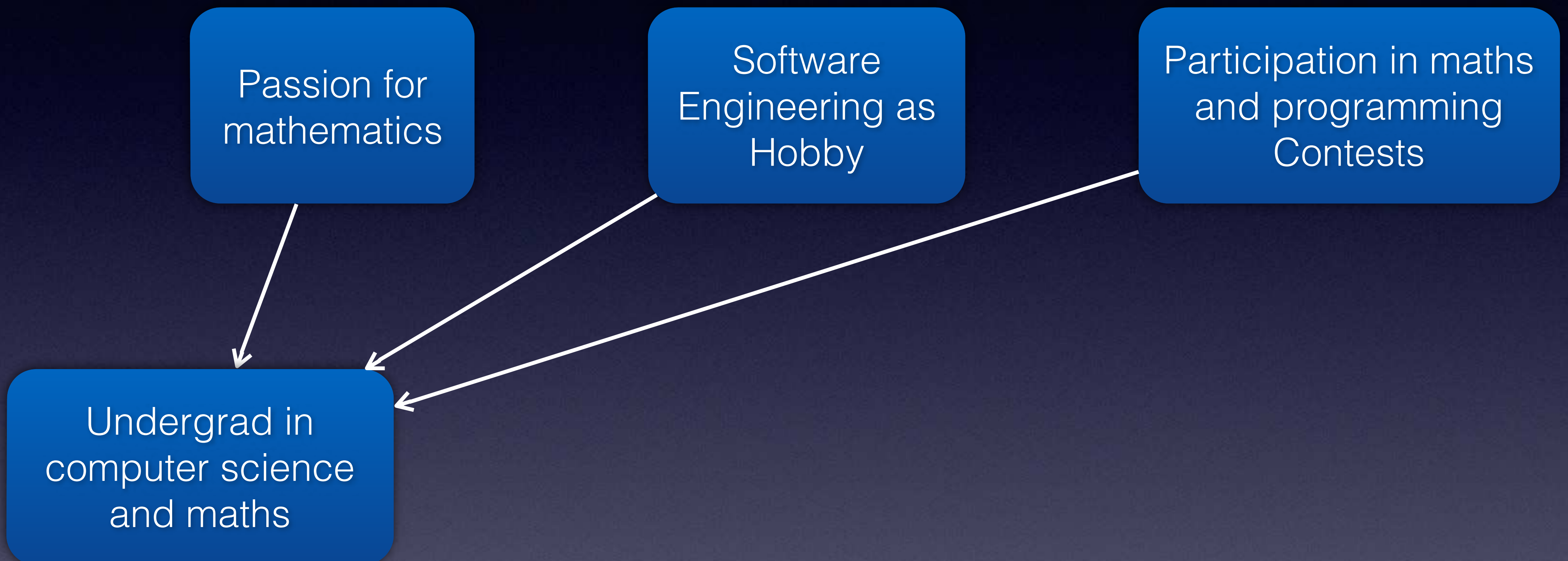
My Computer-Science Career Path

Passion for mathematics

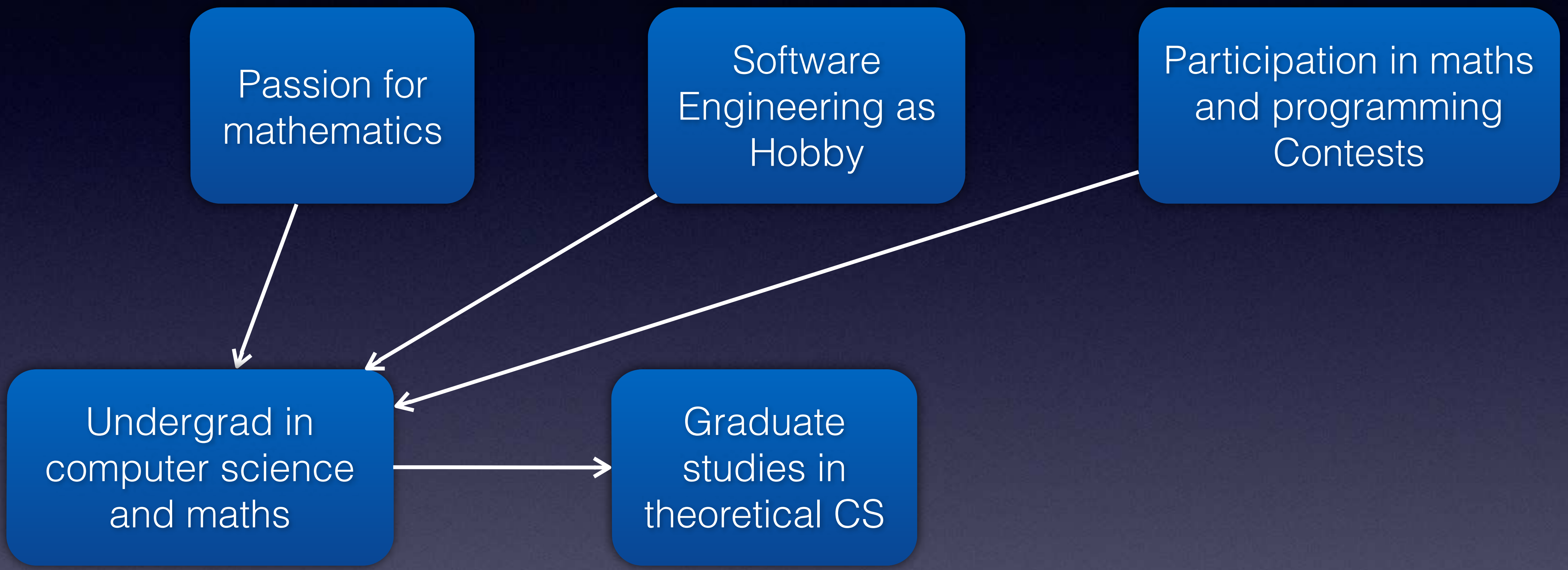
Software Engineering as Hobby

Participation in maths and programming Contests

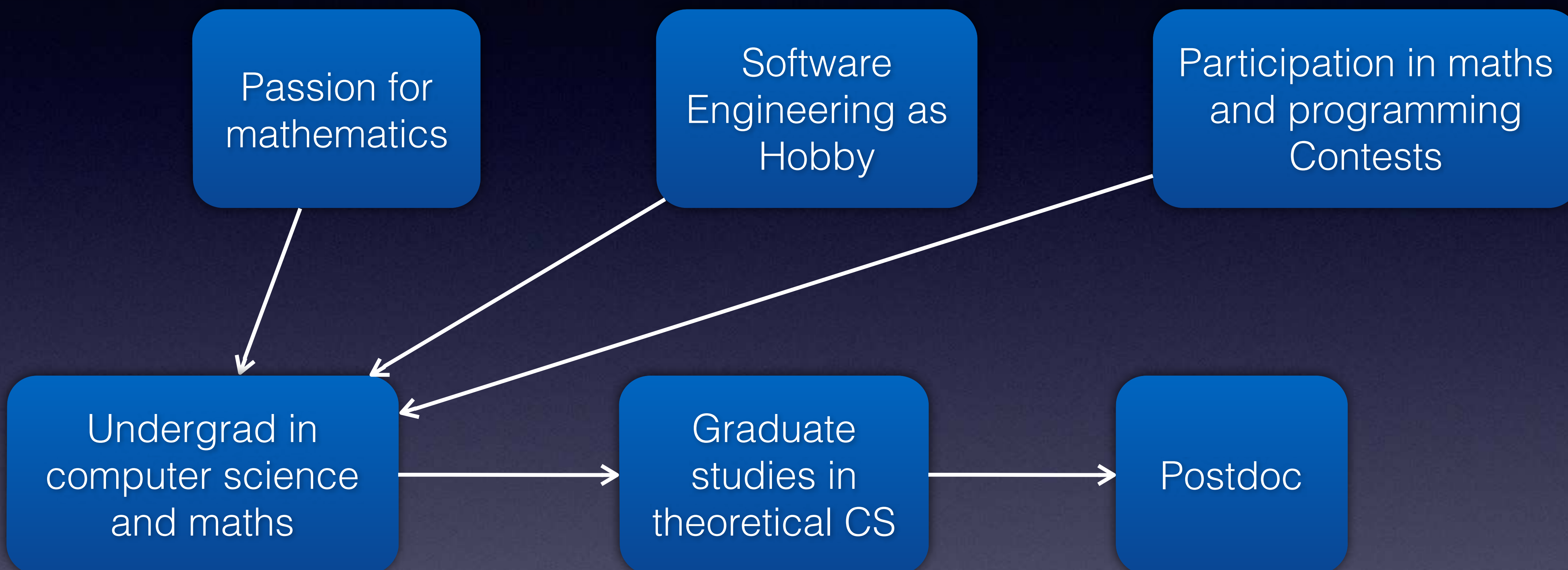
My Computer-Science Career Path



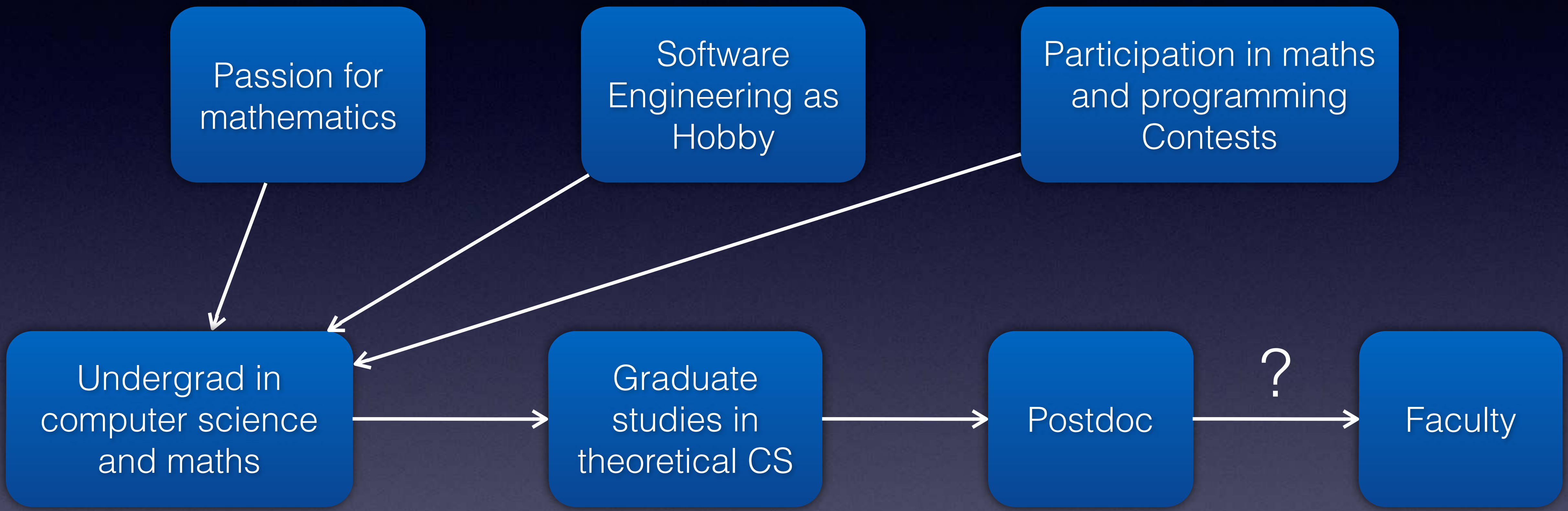
My Computer-Science Career Path



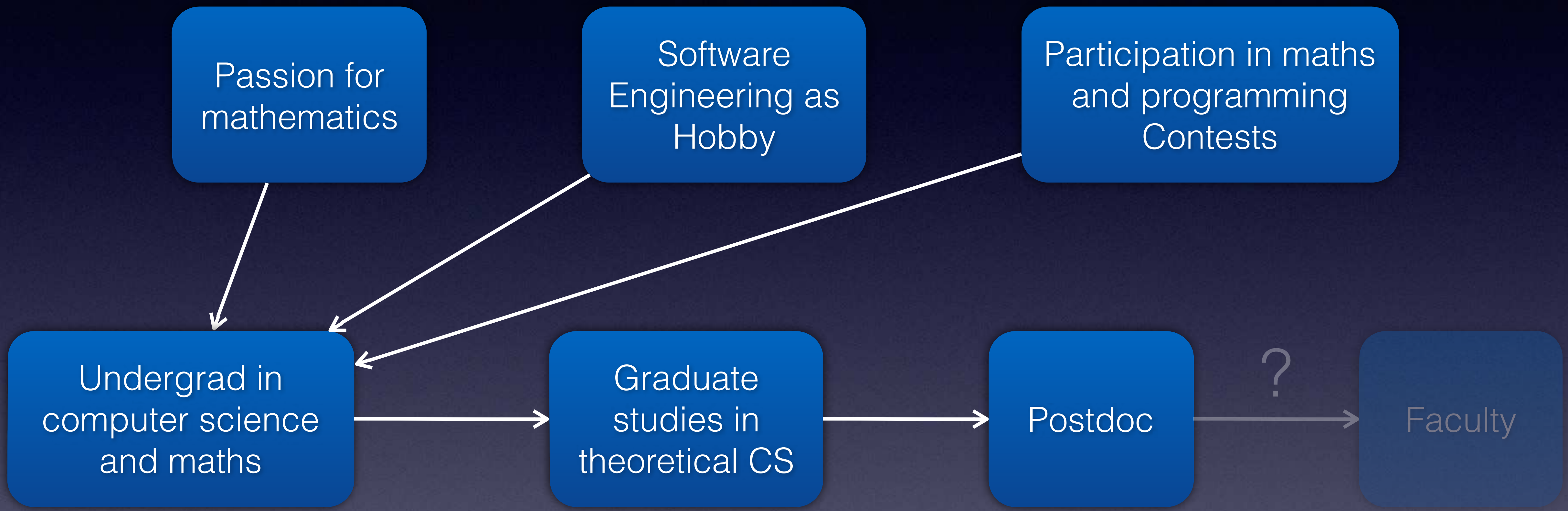
My Computer-Science Career Path



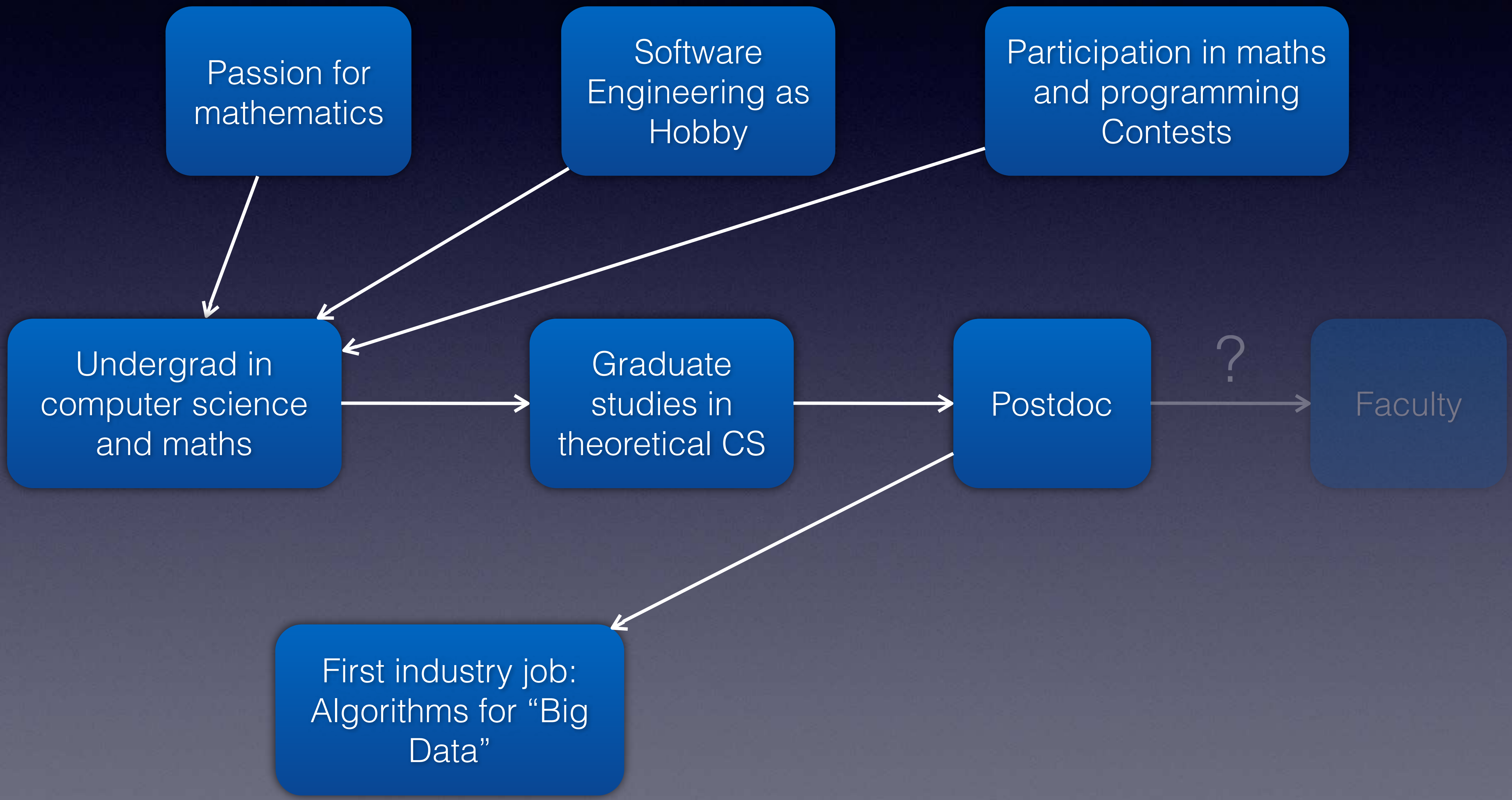
My Computer-Science Career Path



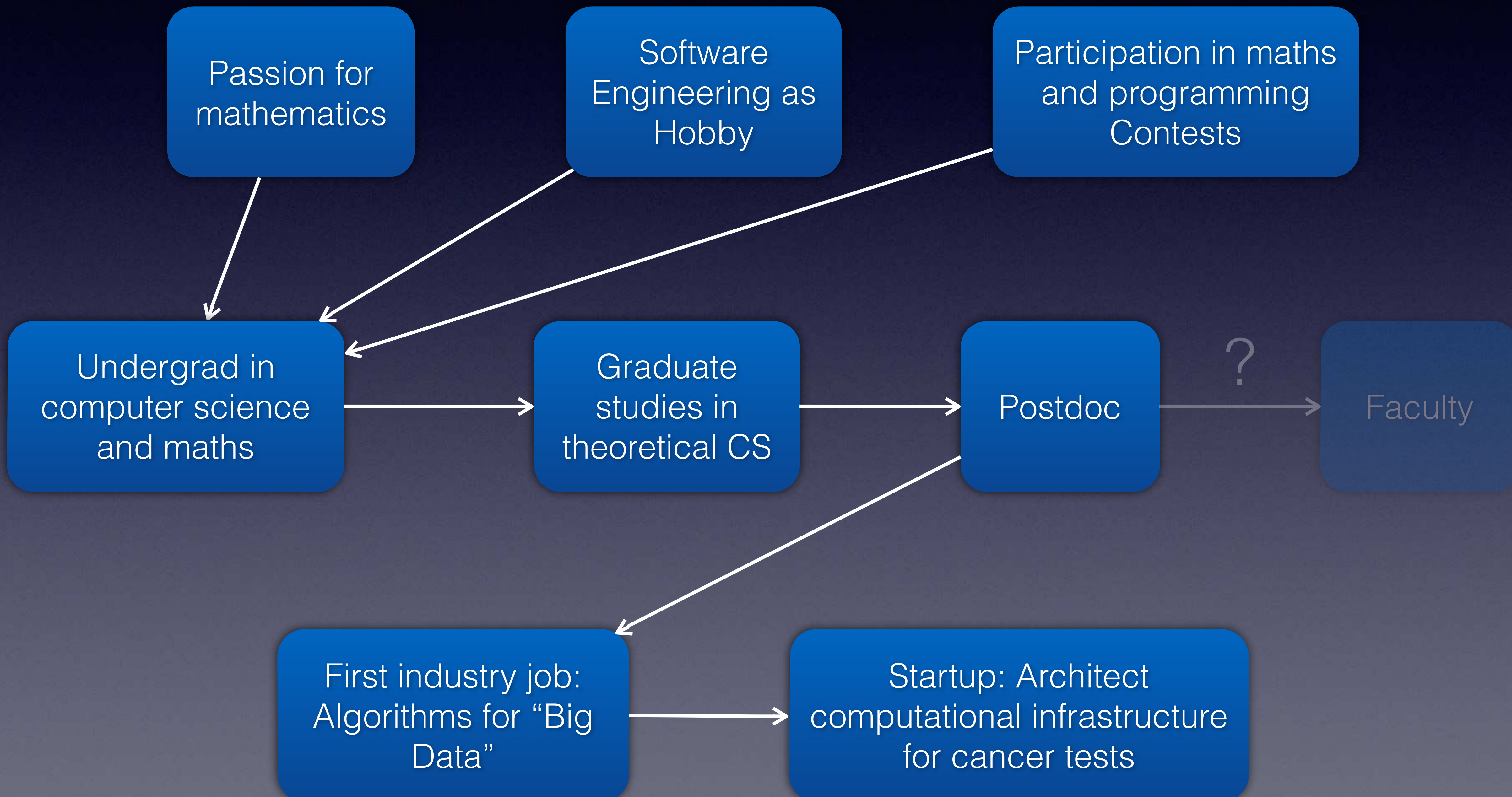
My Computer-Science Career Path



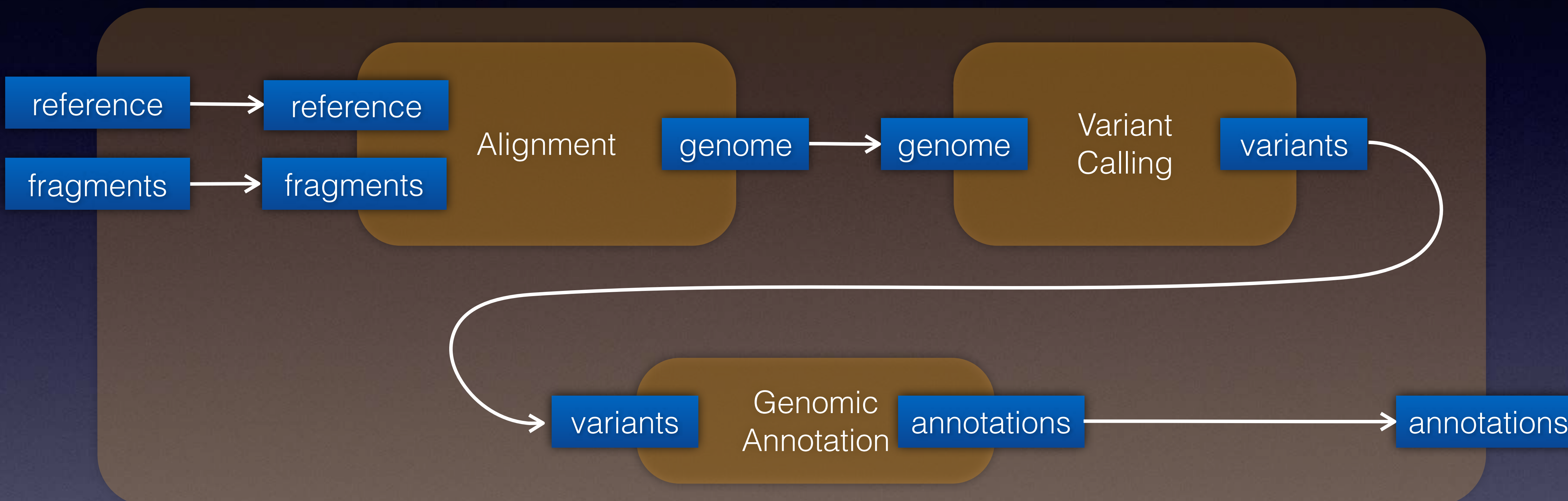
My Computer-Science Career Path



My Computer-Science Career Path



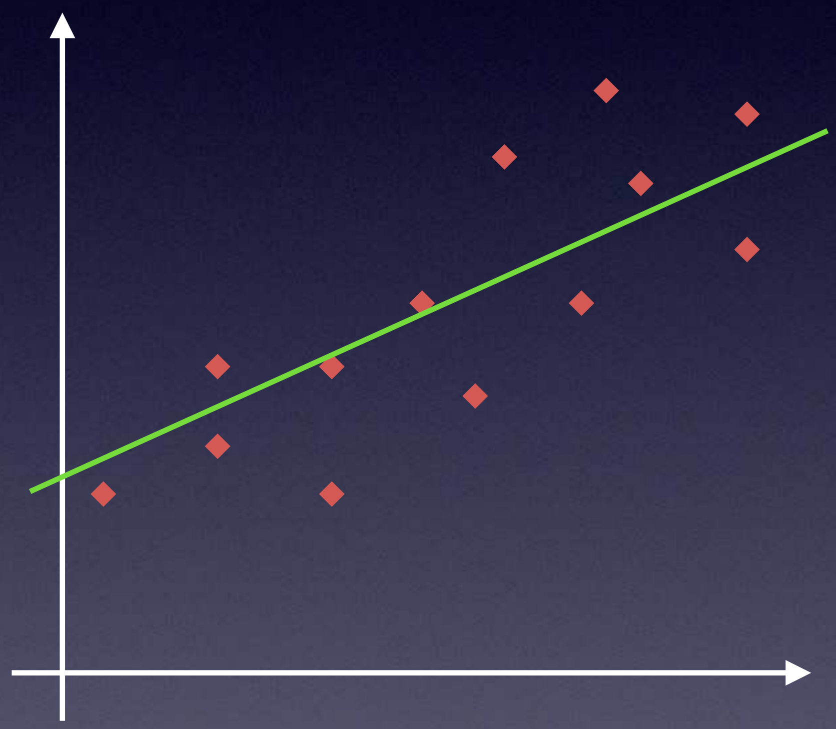
Computational Infrastructure: Dataflow Programming



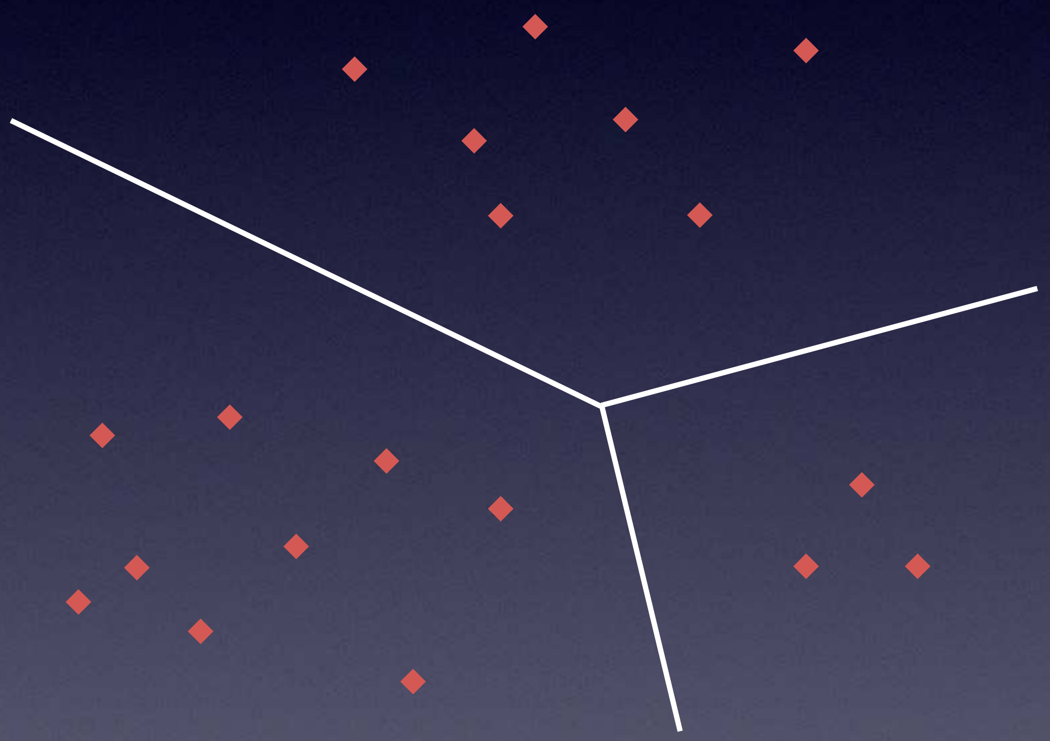

```
module AlignmentAndVariantCalling {
   in reference: FASTAFile
   in fragments: FASTQFile
   out annotations: List<AnnotatedVariant> = ga.annotations

   al = Alignment(reference = reference, fragments = fragments)
   vc = VariantCalling(genome = al.genome)
   ga = GenomicAnnotation(variants = vc.variants)
}
```

Algorithms for “Big Data”

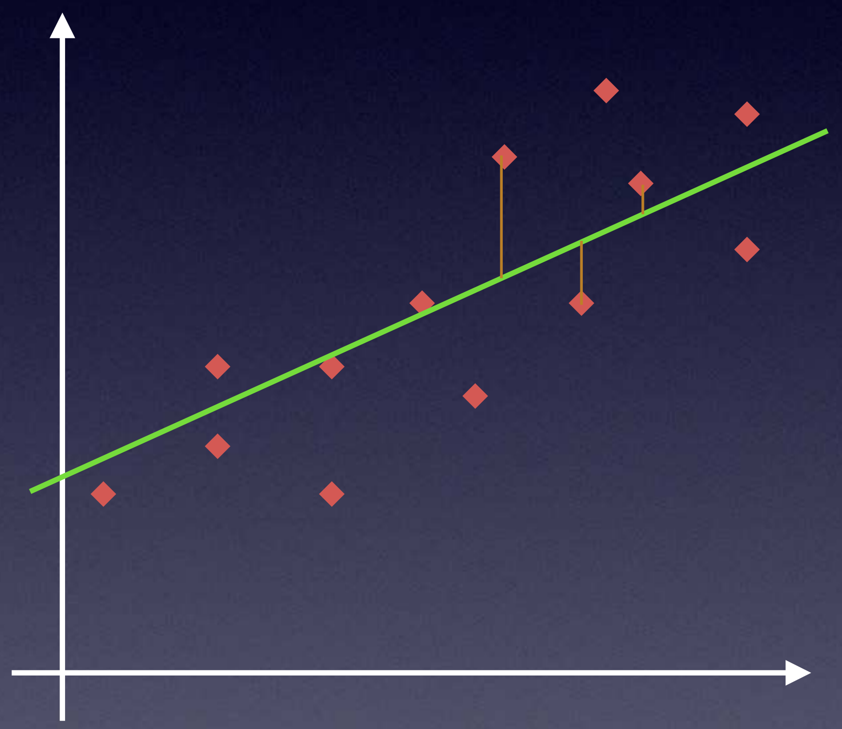


Regression
Least Squares

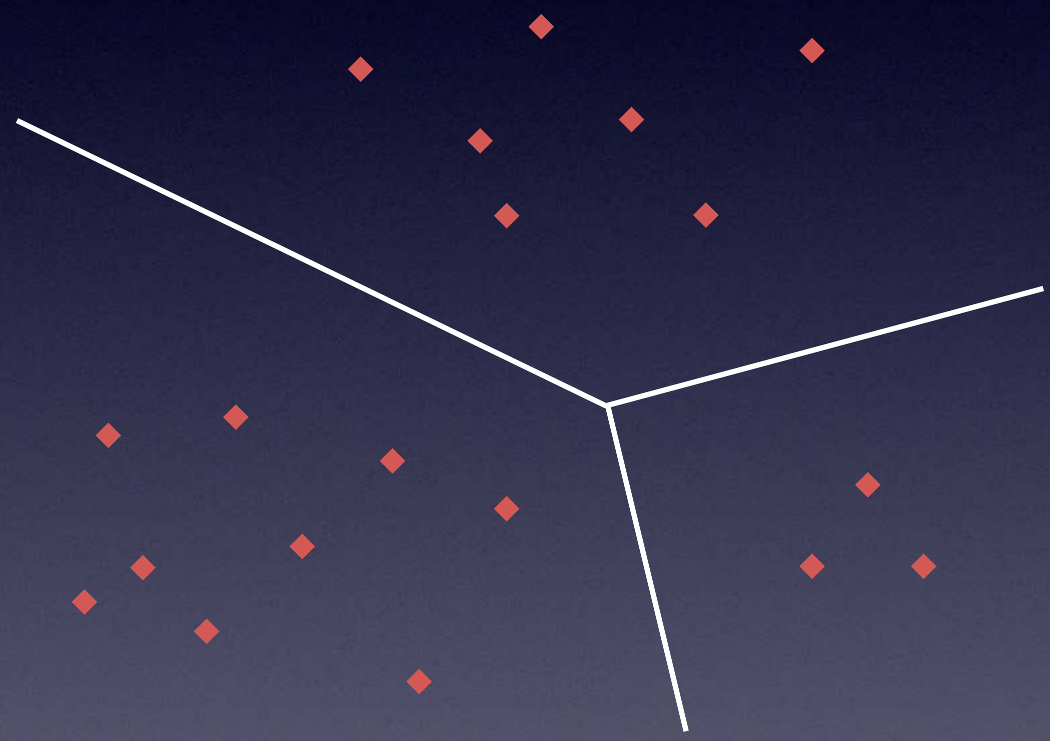


Clustering
k-means

Algorithms for “Big Data”



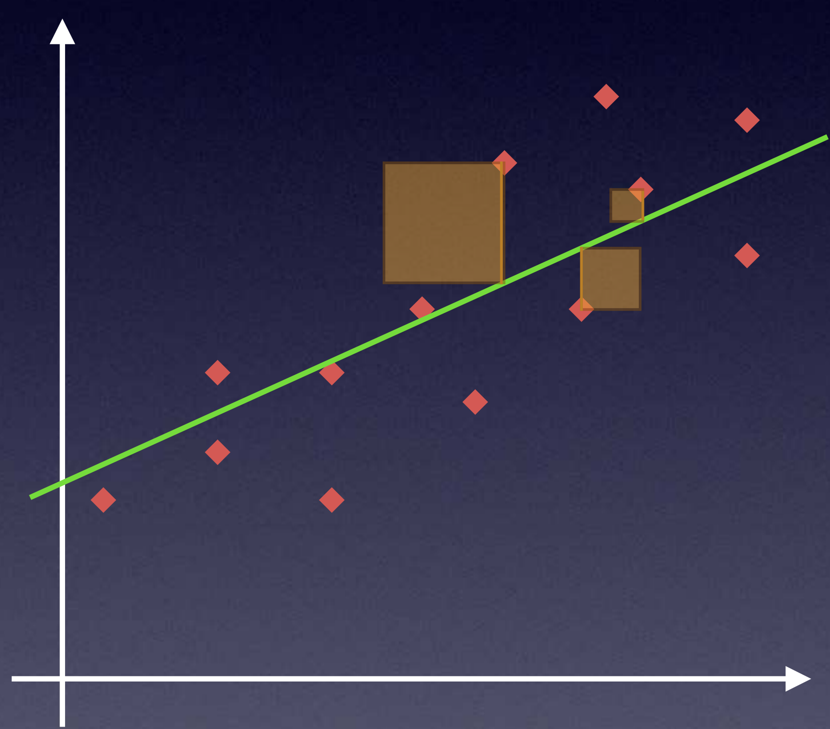
Regression
Least Squares



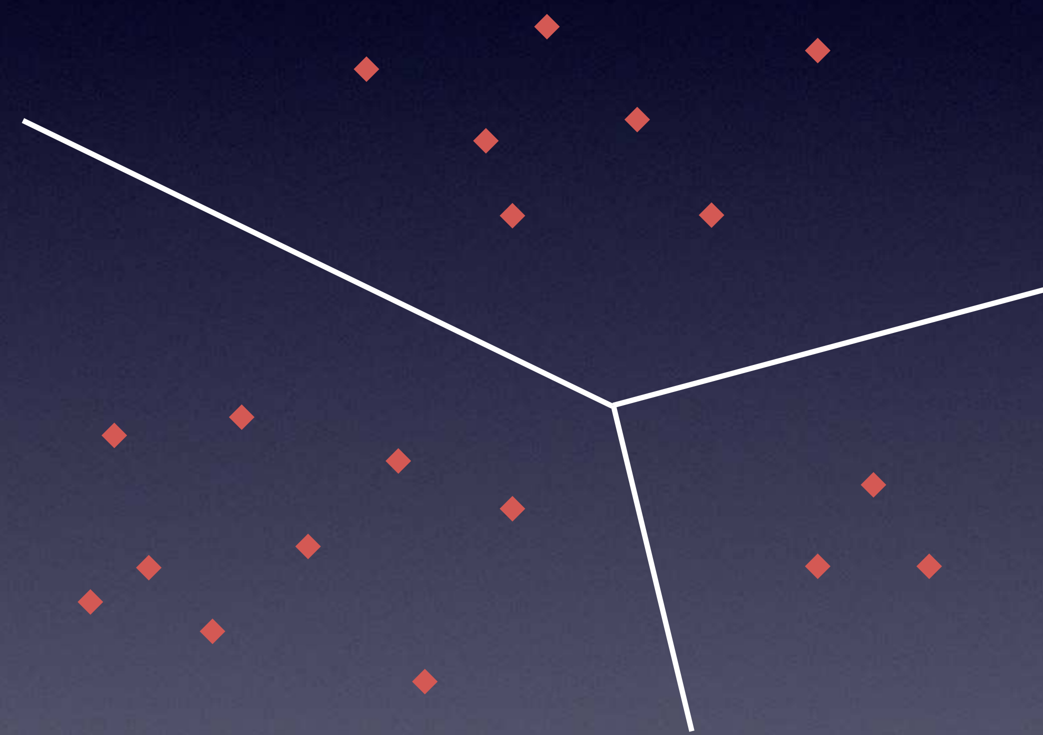
Clustering
k-means

EMC² Pivotal MADlib

Algorithms for “Big Data”



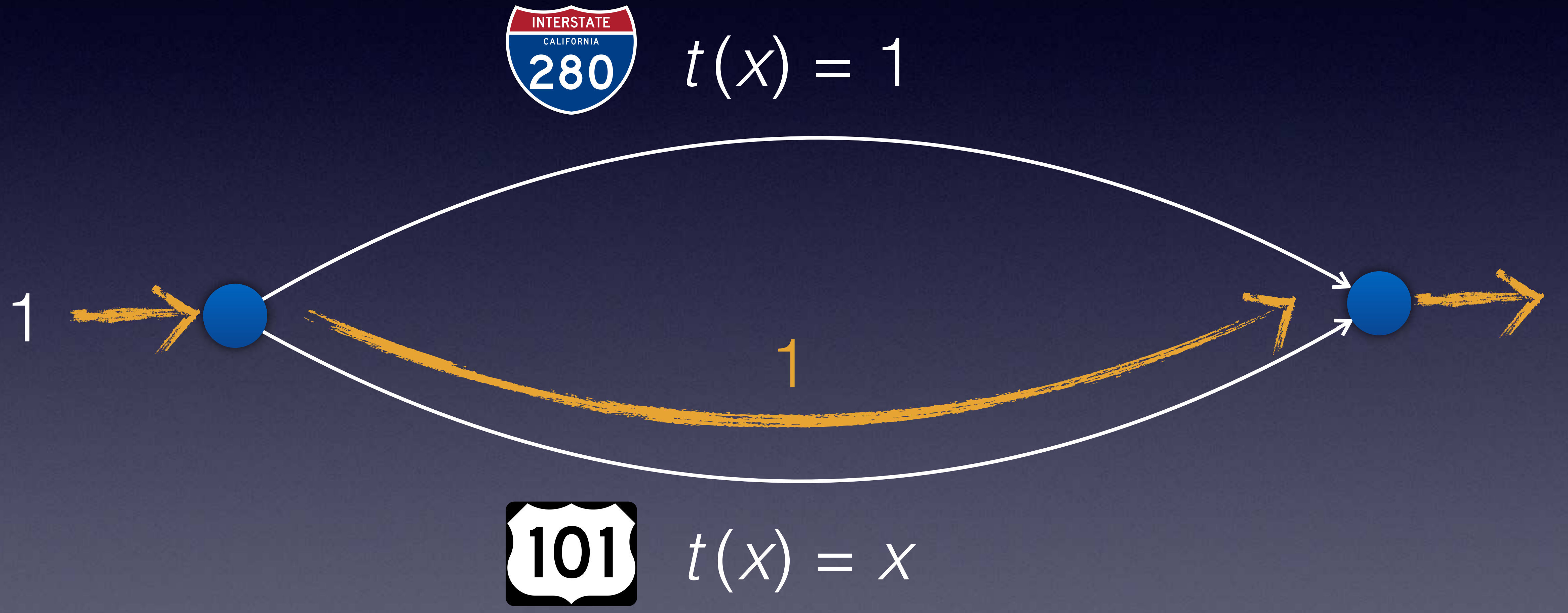
Regression
Least Squares



Clustering
k-means

EMC² Pivotal MADlib

Selfish Routing



Rational behavior but not optimal!

UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft
Stanford

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! ☺