Multiple-source shortest path, and the Floyd-Warshall Algorithm

September 22, 2026

Brief review

Until now we've looked at Single-source shortest paths

- BFS, for unweighted graphs
- Topological search of DAG
- Bellman-Ford for graphs with negative weights (negative cycles)
- Dijkstra

Forward

In this lecture we will look at:

- Abstract Algebra
  - Monoids and Semirings
- Fast exponentiation
  - In a Monoid
- Multi-source Shortest paths
  - Floyd-Warshall algorithm
- Homework problems
  - Implement Floyd-Warshall on different semirings

End of video

Start of video 2:

Abstract Algebra

Abstract Algebra

Very vague intuitions

Structure	**Operations supported**
Field	$+$ and $-$ and $\times$ and $\div$
Ring	$+$ and $-$ and $\times$
Semiring	$+$ and $\times$
Group	$(+$ and $-)$ or $(\times$ and $\div)$
Monoid	$+$ or $\times$

The Group

A group $(G, \circ, e)$ is a set G having a well-defined associative operator, $\circ$, and a neutral element, e.

- closure: G is closed under $\circ$
- associativity: $\forall x, \forall y, \forall z,\ x \circ (y \circ z) = (x \circ y) \circ z$
- neutral element: $\forall x,\ x \circ e = e \circ x = x$
- inverse elements: $\forall x \in G$ there is an inverse element, denoted $\overline{x}$, with the property that: $x \circ \overline{x} = \overline{x} \circ x = e$.

A group may be denoted $(G, +, 0)$ for an additive group, or $(G, \times, 1)$ for a multiplicative group.

An inverse may also be denoted $-x$ or x^{-1}.

Abelian Groups

If $a \circ b = b \circ a$ for all $a, b \in G$ then we call the group commutative or more often Abelian.

Examples of Groups

Which of these are groups?

- $(\mathbb{N}, +, 0)$
- $(\mathbb{N}, \times, 1)$
- $(\mathbb{Z}, \times, 1)$
- $(\mathbb{Z}, +, 0)$
- $(\mathbb{Q} \setminus \{0\}, \times, 1)$
- $m \times n$ matrices with addition
- $n \times n$ matrices with multiplication

Vector space, example of Abelian Group

Axiom	Meaning
Associativity of addition	$\mathbf{u} + (\mathbf{v} + \mathbf{w}) = (\mathbf{u} + \mathbf{v}) + \mathbf{w}$
Commutativity of addition	$\mathbf{u} + \mathbf{v} = \mathbf{v} + \mathbf{u}$
Identity element of addition	There exists an element $\mathbf{0} \in V$, called the *zero vector*, such that $\mathbf{v} + \mathbf{0} = \mathbf{v}$ for all $\mathbf{v} \in V$.
Inverse elements of addition	For every $\mathbf{v} \in V$, there exists an element $-\mathbf{v} \in V$, called the *additive inverse* of $\mathbf{v}$, such that $\mathbf{v} + (-\mathbf{v}) = \mathbf{0}$.
Compatibility of scalar multiplication with field multiplication	$a(b\mathbf{v}) = (ab)\mathbf{v}$ [nb 2]
Identity element of scalar multiplication	$1\mathbf{v} = \mathbf{v}$, where 1 denotes the multiplicative identity in F.
Distributivity of scalar multiplication with respect to vector addition	$a(\mathbf{u} + \mathbf{v}) = a\mathbf{u} + a\mathbf{v}$
Distributivity of scalar multiplication with respect to field addition	$(a + b)\mathbf{v} = a\mathbf{v} + b\mathbf{v}$

The Monoid

A monoid $(M, \circ, e)$ is a set M having an associative operator, $\circ$, and a neutral element, e.

- closure: M is closed under $\circ$.
- associativity: $\forall x, \forall y, \forall z,\ x \circ (y \circ z) = (x \circ y) \circ z$
- neutral element: $\forall x,\ x \circ e = e \circ x = x$

Think of a monoid as a group *perhaps* without inverses.

Examples:

- Positive integers under multiplication,
- A rational language, Σ^*, under concatenation, with the empty word as neutral element.
- Unary functions under composition

The `fold` operation

The `fold` operation is available in many programming languages, functional and otherwise.
Let $(M, \circ, e)$ be a monoid, and $S = \{m_1, m_2, ..., m_n\}$, $m_i \in M$, then

$$fold(S) = \begin{cases} e & \text{if } n = 0 \\ m_1 & \text{if } n = 1 \\ m_1 \circ fold(S \setminus \{m_1\}) & \text{if } n > 1 \end{cases}$$

$\texttt{fold}(S, +, 0)$	$\sum_{i=i}^{n} m_i$	$m_1 + m_2 + m_3 + ... + m_n$
$\texttt{fold}(S, \cup, \emptyset)$	$\bigcup_{i=i}^{n} m_i$	$m_1 \cup m_2 \cup m_3 \cup ... \cup m_n$
$\texttt{fold}(S, \times, 1)$	$\prod_{i=i}^{n} m_i$	$m_1 \times m_2 \times m_3 \times ... \times m_n$
$\texttt{fold}(S, \wedge, \top)$	$\bigwedge_{i=i}^{n} m_i$	$m_1 \wedge m_2 \wedge m_3 \wedge ... \wedge m_n$
$\texttt{fold}(S, strcat, "")$		$\texttt{strcat}(m_1, m_2, m_3, ..., m_n)$
$\texttt{fold}(S, compose, id)$		$x \mapsto m_n(...m_3(m_2(m_1(x))))$

The `fold` operation in Python

Use `functools.reduce` with named function or `lambda` function.

```
In [5]: import functools

        functools.reduce(lambda a,b: a+b, [1,2,3,4,5], 0)
Out[5]: 15

In [6]: functools.reduce(lambda a,b: a*b, [1,2,3,4,5], 1)
Out[6]: 120

In [13]: def plus(a,b):
             return a + b

         functools.reduce(plus, [1,2,3,4,5], 1)
Out[13]: 16
```

The function `reduce` was removed from Python 3.

End of video

Start of video 3:

Abstract Algebra (cont)

The Ring

A ring is an object $(R, \oplus, 0, \otimes, 1)$, for which

- $(R, \oplus, 0)$ is a Abelian group,
- $(R, \otimes, 1)$ is an monoid, and
- $\otimes$ is distributive across $\oplus$.

$$z \otimes (a \oplus b) = (z \otimes a) \oplus (z \otimes b)$$
$$(a \oplus b) \otimes z = (a \otimes z) \oplus (b \otimes z)$$

In this case, 0 necessarily annihilates w.r.t. $\otimes$.
I.e., $a \otimes 0 = 0 \otimes a = 0$ for all $a \in R$.

Example: $\mathbb{M}_{n \times n}(\mathbb{Z})$: $n \times n$ matrices containing integers.

Right 0-annihilation

Theorem

If $(R, \oplus, 0, \otimes, 1)$ *is a ring, then* $a \otimes 0 = 0$ *for every* $a \in R$.

Proof.

Let $a \in R$.

$$\begin{aligned} a \otimes 0 &= (a \otimes 0) \oplus 0 = (a \otimes 0) \oplus (a \oplus -a) \\ &= ((a \otimes 0) \oplus a) \oplus -a \\ &= ((a \otimes 0) \oplus (a \otimes 1)) \oplus -a \\ &= (a \otimes (0 \oplus 1)) \oplus -a \\ &= (a \otimes 1) \oplus -a = a \oplus -a \\ &= 0 \end{aligned}$$

Right 0-annihilation

Easier to read using different notation

Proof.

Let $a \in R$.

$$\begin{aligned} a0 &= a0 + 0 = a0 + (a + -a) \\ &= (a0 + a) + -a \\ &= (a0 + a1) + -a \\ &= a(0 + 1) + -a \\ &= a1 + -a = a + -a \\ &= 0 \end{aligned}$$

Square matrices

How to show that $(\mathbb{M}_{n\times n}(\mathbb{Z}),\ +, 0_{n\times n},\ \times, I_{n\times n})$ is a ring.

1. Show $(\mathbb{M}_{n\times n}(\mathbb{Z}), +, 0_{n\times n})$ is an Abelian group.
2. Show $(\mathbb{M}_{n\times n}(\mathbb{Z}), \times, I_{n\times n})$ is a monoid.
3. Show the distributive properties.

Square matrices

How to show that $(\mathbb{M}_{n\times n}(\mathbb{Z}),\ +, 0_{n\times n},\ \times, I_{n\times n})$ is a ring.

1. Show $(\mathbb{M}_{n\times n}(\mathbb{Z}), +, 0_{n\times n})$ is an Abelian group.

   If $A, B, C \in \mathbb{M}_{n\times n}(\mathbb{Z})$ then
   - Closure: $A + B \in \mathbb{M}_{n\times n}(\mathbb{Z})$
   - Associativity: $A + (B + C) = (A + B) + C$
   - Identity: $A + 0_{n\times n} = 0_{n\times n} + A = A$
   - Inverse: $\exists A'$ such that $A + A' = A' + A = 0_{n\times n}$
   - Commutativity: $A + B = B + A$

2. Show $(\mathbb{M}_{n\times n}(\mathbb{Z}), \times, I_{n\times n})$ is a monoid.
3. Show the distributive properties.

Square matrices

How to show that $(\mathbb{M}_{n\times n}(\mathbb{Z}),\ +, 0_{n\times n},\ \times, I_{n\times n})$ is a ring.

1. Show $(\mathbb{M}_{n\times n}(\mathbb{Z}), +, 0_{n\times n})$ is an Abelian group.
2. Show $(\mathbb{M}_{n\times n}(\mathbb{Z}), \times, I_{n\times n})$ is a monoid.

   If $A, B, C \in \mathbb{M}_{n\times n}(\mathbb{Z})$ then
   - Closure: $A \times B \in \mathbb{M}_{n\times n}(\mathbb{Z})$
   - Associativity: $A \times (B \times C) = (A \times B) \times C$
   - Identity: $A \times I_{n\times n} = I_{n\times n} \times A = A$

   $A \times B \neq B \times A$, but not necessary for monoid.
   A^{-1} does not exist, but not necessary for monoid.
3. Show the distributive properties.

Square matrices

How to show that $(\mathbb{M}_{n\times n}(\mathbb{Z}),\ +, 0_{n\times n},\ \times, I_{n\times n})$ is a ring.

1. Show $(\mathbb{M}_{n\times n}(\mathbb{Z}), +, 0_{n\times n})$ is an Abelian group.
2. Show $(\mathbb{M}_{n\times n}(\mathbb{Z}), \times, I_{n\times n})$ is a monoid.
3. Show the distributive properties.

If $A, B, C \in \mathbb{M}_{n\times n}(\mathbb{Z})$ then

$$(A + B) \times C = (A \times C) + (B \times C)$$
$$C \times (A + B) = (C \times A) + (C \times B)$$

The Semiring

A semiring is an object $(S, \oplus, 0, \otimes, 1)$, for which

- $(S, \oplus, 0)$ is a commutative monoid,
- $(S, \otimes, 1)$ is a monoid,
- $\otimes$ is distributive across $\oplus$,

$$z \otimes (a \oplus b) = (z \otimes a) \oplus (z \otimes b)$$
$$(a \oplus b) \otimes z = (a \otimes z) \oplus (b \otimes z)$$

- 0 annihilates w.r.t. $\otimes$ (i.e. $\forall x \in S, x \otimes 0 = 0 \otimes x = 0$)

Think of a semiring as a ring *perhaps* without additive inverses. A ring already may be without multiplicative inverses.

The Tropical semiring

Defined as: $S = (\mathbb{Z} \cup \{\infty\}, \underbrace{\min, \infty}_{\oplus}, \underbrace{+, 0}_{\otimes})$.

Careful, $0_S = \infty$, $1_S = 0_{\mathbb{Z}}$, and $\otimes_S = +_{\mathbb{Z}}$.

The semiring operations are as follows:

$$x \oplus y = \min\{x, y\} \qquad \textit{e.g.} \quad 3 \oplus 2 = \min\{3, 2\} = 2$$
$$x \otimes y = x + y \qquad \textit{e.g.} \quad 3 \otimes 2 = 3 + 2 = 5$$

The neutral elements:

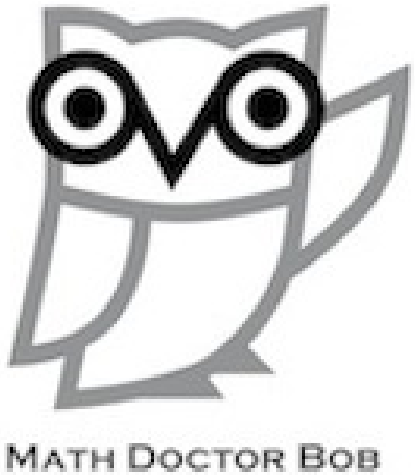
$$x \oplus 0_S = \min\{x, \infty\} = x$$
$$x \otimes 1_S = x + 0_{\mathbb{Z}} = x$$

0-annihilation:

$$x \otimes 0_S = x + \infty = \infty$$

Additional Resources

Socratica https://www.socratica.com.



Math Dr Bob https://mathdoctorbob.org.

End of video

Start of video 4:

Fast Exponentiation

Fast Exponentiation

Classic Exponentiation

$$a^b = \underbrace{a \times a \times \cdots \times a}_{b-1 \text{ multiplications}}$$

The naïve algorithm for calculating a^b uses a loop of $\Theta(b)$ multiplications.

Classic Algorithm

In [22]:
```
def simple_power(a,b):
    answer = 1
    for i in range(b):
        answer = answer * a
    return answer
```

In [23]:
```
simple_power(2,5)
```

Out[23]: 32

In [24]:
```
simple_power(1.01,10)
```

Out[24]: 1.1046221254112045

Fast Exponentiation (Intuition)

We want to a power such as 3^{32}.

We can simplify the problem:

$$3^{32} = 3^{2 \times 16} = (3^2)^{16} = 9^{16}$$

And then use the same technique, recursively, to compute: 9^{16}.

$$9^{16} = 81^8$$

Fast Exponentiation (Intuition)

But this fails if the exponent is odd: 3^{33}.

We could simplify this problem:

$$3^{33} = 3^{32+1} = 3^{32} \times 3^1 = 3^{32} \times 3.$$

And then use the previous technique.

Fast Exponentiation (Algorithm)

$$power(a, b) = \begin{cases} 1 & \text{if } b = 0 \\ power(a \times a,\ b/2) & \text{if } b \text{ is even} \\ a \times power(a,\ b-1) & \text{if } b \text{ is odd} \end{cases}$$

```
FAST-POWER(a, b)
1  if b = 0
2        return 1
3  if odd(b)
4        return a×FAST-POWER(a,b − 1)
5  else
6        return FAST-POWER(a × a, b/2)
```

In fact, we need two $\times$'s per 1 bit and one $\times$ per 0 bit within the binary representation of b. That is $\Theta(\log b)$ $\times$'s.

Generalization

Let $(M, \circ, e)$ be a monoid.

Exponentiation

For $n \in \mathbb{N}$ and $x \in M$ we define exponentiation as follows:

$$x^n = \begin{cases} e & \text{if } n = 0 \\ x^{n-1} \circ x & \text{otherwise} \end{cases}$$

We can use the fast exponentiation algorithm to calculate x^n in $\Theta(\log n)$ operations. That M is a monoid is important because of associativity. For we wish to calculate: $x^4 = \underbrace{(x \circ x)}_{x^2} \circ \underbrace{(x \circ x)}_{x^2}$,

instead of $x^4 = ((x \circ x) \circ x) \circ x$ as in the definition.

Exponentiation of matrices

Question Let A be an $n \times n$ matrix, and let $b \in \mathbb{N}$.
How many operations are necessary to compute A^b ?

$$\begin{bmatrix} 0 & 6 & 1 & -3 \\ 1 & 0 & -1 & 2 \\ -2 & 3 & 0 & 6 \\ 4 & -1 & -2 & 0 \end{bmatrix}^{17}$$

Answer We need $\Theta(\log b)$ matrix $\times$'s to calculate the power.
And if each matrix $\times$ requires n^3 scalar $\times$'s.
So A^b needs $\Theta(n^3 \log b)$ scalar $\times$'s.

Recall The Strassen algorithm uses $\Theta(n^{log_2(7)}) < \Theta(n^3)$ operations.

End of video

Start of video 5:

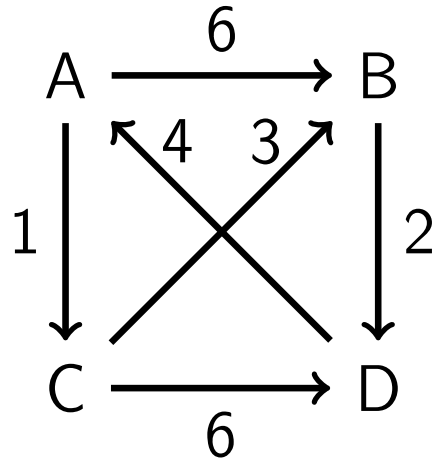
Shortest Paths

Shortest Paths

Shortest Paths

Consider a directed graph, in which the edges are weighted representing distances or travel times.

G :



$$M_G = \begin{array}{c} \begin{matrix} A & B & C & D \end{matrix} \\ \begin{bmatrix} 0 & 6 & 1 & \infty \\ \infty & 0 & \infty & 2 \\ \infty & 3 & 0 & 6 \\ 4 & \infty & \infty & 0 \end{bmatrix} \end{array} \begin{matrix} A \\ B \\ C \\ D \end{matrix}$$

Question: What is the shortest path from x to y?
I.e., for all x and y.

First approach

Let D_k denote the $n \times n$ matrix with components $D_k[x, y]$ being the length of the minimum path from $x \to y$ with at most k edges.

Whenever

$$\overbrace{\underbrace{x \to v_1 \to v_2 \to \cdots \to v_{k-1} \to v_k}_{\text{shortest path } x \to v_k} \to y}^{\text{shortest path } x \to y}$$

is a shortest path between x and y, then

$$x \to v_1 \to v_2 \to \cdots \to v_{k-1} \to v_k$$

is a shortest path between x and v_k.

First approach

Recall D_k is the matrix such that $D_k[x, y]$ is the length of the shortest path from $x \to y$ with at most k edges.

$$D_k[x, y] = \begin{cases} M[x, y] & \text{if } k = 1 \\ \min_z(D_{k-1}[x, z] + M[z, y]) & \text{otherwise} \end{cases}$$

D_{n-1} contains the shortest path lengths, because the longest possible path has $n - 1$ edges.

Algorithm (1/2)

$$D_k[x,y] = \begin{cases} M[x,y] & \text{if } k = 1 \\ \min_{z \in V}(D_{k-1}[x,z] + M[z,y]) & \text{otherwise} \end{cases}$$

- To obtain D_{n-1}, we iteratively calculate the sequence $D_2, D_3, \ldots, D_{n-1}$.
- To calculate D_k, we need only M and D_{k-1}.
- We can simply retain D_k and D_{k-1} and optimize space.
- We use two $n \times n$ arrays rather than $n-1$ arrays.
- We swap D_k for D_{k+1} on each iteration.

Algorithm (2/2)

SLOW-SHORTEST-PATH(M)

1 $n \leftarrow \text{size}(M)$
2 $D_{new} \leftarrow M$
3 for $k \leftarrow 2$ to $n-1$
4 $\quad D_{old} \leftarrow D_{new}$
5 $\quad$ for $x \leftarrow 1$ to n
6 $\quad\quad$ for $y \leftarrow 1$ to n
7 $\quad\quad\quad D_{new}[x,y] \leftarrow \min_{z=1}^{n}(D_{old}[x,z] + M[z,y])$
8 return D_{new}

This algorithm, sadly, has complexity $\Theta(n^4)$.

End of video

Start of video 6:

Matrix Multiplication in the Tropical Semiring

Algorithm (reminder)

SLOW-SHORTEST-PATH(M)

1 $n \leftarrow \text{size}(M)$
2 $D_{new} \leftarrow M$
3 for $k \leftarrow 2$ to $n-1$
4 $\quad D_{old} \leftarrow D_{new}$
5 $\quad$ for $x \leftarrow 1$ to n
6 $\quad\quad$ for $y \leftarrow 1$ to n
7 $\quad\quad\quad D_{new}[x,y] \leftarrow \min_{z=1}^{n}(D_{old}[x,z] + M[z,y])$
8 return D_{new}

Application and remark

$$M = \begin{bmatrix} 0 & 6 & 1 & \infty \\ \infty & 0 & \infty & 2 \\ \infty & 3 & 0 & 6 \\ 4 & \infty & \infty & 0 \end{bmatrix}$$

$$D_2 = \begin{bmatrix} 0 & 4 & 1 & 7 \\ 6 & 0 & \infty & 2 \\ 10 & 3 & 0 & 5 \\ 4 & 10 & 5 & 0 \end{bmatrix}$$

$$D_3 = D_2 \otimes M = \begin{bmatrix} 0 & 4 & 1 & 6 \\ 6 & 0 & 7 & 2 \\ 9 & 3 & 0 & 5 \\ 4 & 8 & 5 & 0 \end{bmatrix}$$

$\min\{6+4, 0+10, 3+5, \infty+0\} = 8$

We have:
$D_3[i,j] = \min_z(D_2[i,z] + M[z,j])$.

Compare to:
$D^3_{i,j} = \sum_z (D^2_{i,z} \times M_{z,j})$.

To compute D_3 we do a *sort of* matrix product, $D_2 \times M$, but replacing $(+, \times)$ with $(\min, +)$.

We are calculating M^k, but with different operations, *i.e.*. the tropical semiring.

Matrices

If $(S,\ \oplus, e,\ \otimes, f)$ is a semiring and $n \in \mathbb{N}$, then the object $(\mathbb{M}_{n\times n}(S),\ +, O_{n\times n},\ \times, I_{n\times n})$ is a semiring under the following assumption that $c = a \times b$, $d = a + b$, $I_{n\times n}$, and $O_{n\times n}$ are defined as follows:

$$c_{i,j} = \bigoplus_{k=1}^{n} (a_{i,k} \otimes b_{j,k}) \qquad \textit{row} \times \textit{column}$$

$$d_{i,j} = a_{i,j} \oplus b_{i,j} \qquad \textit{element} + \textit{element}$$

$$I_{i,j} = \begin{cases} f & \text{if } i = j \\ e & \text{otherwise} \end{cases} \qquad \text{Identity matrix}$$

$$O_{i,j} = e \qquad \text{Zero matrix}.$$

This means $(\mathbb{M}_{n\times n}(S), \times, I_{n\times n})$ is a monoid.
Thus, we can compute M^n using fast exponentiation.

Powers of matrices in particular semirings

$S = (\mathbb{N}, +, \times, 0, 1)$
The matrices having values $\{0, 1\}$ represent the unweighted graphs. Their powers (values in $\mathbb{N}$) count the k-step walks between the vertices.

$S = (\mathbb{Z} \cup \{\infty\}, \min, +, \infty, 0)$ (called tropical semiring)
These matrices, $\mathcal{M}_n(S)$, represent maps with distances (possibly negative). Their successive powers consider longer and longer sequences. This is our case.

$S = (\mathbb{N} \cup \{\infty\}, \max, \min, 0, \infty)$
These matrices, $\mathcal{M}_n(S)$, represent flow maps, with flows limited to a single path.

$S = (\mathbb{B}, \vee, \wedge, \bot, \top)$ ... A matrix represents an unweighted graph. The n'th power is the transitive closure.

Faster computing the shortest distance

Now we wish to compute

$$D_{n-1} = M^{n-1}$$

with elements of M coming from semiring $(\mathbb{Z} \cup \{\infty\},\ \min, \infty,\ +, 0)$.

Rather than using $\textsc{Slow-Shortest-Path}(M)$ (complexity $\Theta(n^4)$), we will use $\textsc{Fast-Power}(M, n-1)$ (complexity $\Theta(n^3 \log n)$).

Faster computing the shortest distance

```
FAST-POWER(a, b, ×)
1  if b = 0
2        return GENERATE-IDENTITY()
3  if b = 1
4        return a
5  if odd(b) then
6        return a × FAST-POWER(a, b − 1, ×)
7  else
8        return FAST-POWER(a × a, b/2, ×)
```

Faster computing the shortest distance

To compute

$$D_{n-1} = M^{n-1}, .$$

Special case:

$$A^b = A^{n-1} \ \forall b \geq n-1 \, .$$

This is true for semiring $(\mathbb{Z} \cup \{\infty\}, \min, +, \infty, 0)$, and when the graph contains no negative cycles.

Thus it is not necessary to calculate exactly A^{n-1}. Any higher power of A suffices.

FAST-POWER can quickly compute the next power of 2: FAST-POWER$(M, 2^{\lceil \log(n-1) \rceil})$.

Algorithm with complexity $\Theta(n^3 \log n)$

FAST-SHORTEST-PATH(M)
  return FAST-POWER(M, $2^{\lceil \log(n-1) \rceil}$, MATRIX-MULTIPLY)

MATRIX-MULTIPLY(M_1, M_2)
1 for $x \leftarrow 1$ to n
2 for $y \leftarrow 1$ to n
3 $P[x, y] \leftarrow \min_{z=1}^{n} M_1[x, z] + M_2[z, y]$
4 return P

Algorithm with complexity $\Theta(n^3 \log n)$

We can efficiently in-line MATRIX-MULTIPLY and FAST-POWER

- We omit the case $b = 0 \rightarrow$ *return Identity*.
- We omit the odd branch which is never taken.
- We only multiply $M_1 = M_2$.

Algorithm with complexity $\Theta(n^3 \log n)$

FAST-SHORTEST-PATH(M)

1 $n \leftarrow \text{size}(M)$
2 $D \leftarrow M$
3 for $k \leftarrow 2$ to $\lceil \log(n-1) \rceil$
4 for $x \leftarrow 1$ to n
5 for $y \leftarrow 1$ to n
6 $D_{new}[x,y] \leftarrow \min_{z=1}^{n} D[x,z] + \underbrace{D[z,y]}_{\text{was } M[z,y]}$
7 $D \leftarrow D_{new}$
8 return D

End of video

Start of video 7:

Floyd-Warshall

Second approach

Imagine the vertices of a graph, numbered from 1 to n and let $D_k[x, y]$ be the shortest distance between x and y but which only passes through vertices $1 \ldots k$.

$$D_0[x, y] = M[x, y]$$
$$D_k[x, y] = \min(\underbrace{D_{k-1}[x, k] + D_{k-1}[k, y]}_{\text{passing explicitly through } k}, \underbrace{D_{k-1}[x, y]}_{\text{avoiding } k}) \qquad \Theta(1)$$

Previously we had

$$D_{new}[x, y] \leftarrow \min_{z=1}^{n}(D_{old}[x, z] + M[z, y]) \qquad \Theta(n)$$

This suggests an algorithm with complexity $\Theta(n^3)$ in time and $\Theta(n^2)$ in space.

The Floyd-Warshall algorithm

FLOYD-WARSHALL(M)

1 $n \leftarrow \text{size}(M)$
2 $D_{new} \leftarrow M$
3 for $k \leftarrow 1$ to n
4 $\quad D_{old} \leftarrow D_{new}$
5 $\quad$ for $x \leftarrow 1$ to n
6 $\quad\quad$ for $y \leftarrow 1$ to n
7 $\quad\quad\quad D_{new}[x, y] \leftarrow \min\{D_{old}[x, k] + D_{old}[k, y], D_{old}[x, y]\}$
8 return D_{new}

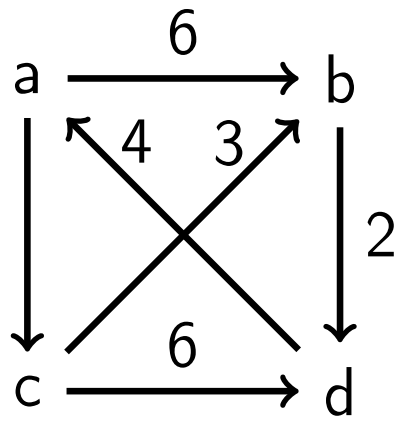
The Floyd-Warshall algorithm for Semiring

FLOYD-WARSHALL(M)

1 $n \leftarrow \text{size}(M)$
2 $D_{new} \leftarrow M$
3 for $k \leftarrow 1$ to n
4 $\quad D_{old} \leftarrow D_{new}$
5 $\quad$ for $x \leftarrow 1$ to n
6 $\quad\quad$ for $y \leftarrow 1$ to n
7 $\quad\quad\quad D_{new}[x, y] \leftarrow \left(D_{old}[x, k] \otimes D_{old}[k, y]\right) \oplus D_{old}[x, y]$
8 return D_{new}

Application

G :



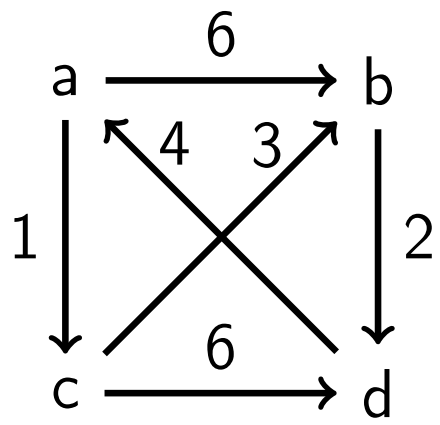
Semiring: eg. $\underbrace{(\mathbb{Z} \cup \{\infty\},\ min, \infty,\ +, 0)}_{(S,\ \oplus, 0,\ \otimes, 1)}$

Initialize $D_\emptyset$ using weights of G,
1 of semiring, *i.e.* 0 on diagonal, and
0 of semiring, *i.e.*, ∞ elsewhere. .

$$D_\emptyset = \begin{array}{c} \begin{matrix} a & b & c & d \end{matrix} \\ \begin{bmatrix} 0 & 6 & 1 & \infty \\ \infty & 0 & \infty & 2 \\ \infty & 3 & 0 & 6 \\ 4 & \infty & \infty & 0 \end{bmatrix} \end{array} \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

Application

G :



Calculate D_a via vertex a.

$$D_\emptyset = \begin{array}{c} \begin{matrix} a & b & c & d \end{matrix} \\ \begin{bmatrix} 0 & 6 & 1 & \infty \\ \infty & 0 & \infty & 2 \\ \infty & 3 & 0 & 6 \\ 4 & \infty & \infty & 0 \end{bmatrix} \end{array} \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

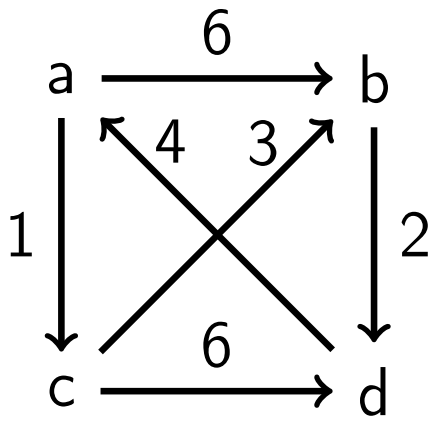
$$D_a = \begin{array}{c} \begin{matrix} a & b & c & d \end{matrix} \\ \begin{bmatrix} 0 & 6 & 1 & \infty \\ \infty & 0 & \infty & 2 \\ \infty & 3 & 0 & 6 \\ 4 & 10 & 5 & 0 \end{bmatrix} \end{array} \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

$$D_a[d,b] = \min\{D_\emptyset[d,a] + D_\emptyset[a,b], D_\emptyset[d,b]\} = \min\{4+6, \infty\} = 10$$

$$D_a[d,c] = \min\{D_\emptyset[d,a] + D_\emptyset[a,c], D_\emptyset[d,c]\} = \min\{4+1, \infty\} = 5$$

Application

G :



Calculate D_b via vertex b.

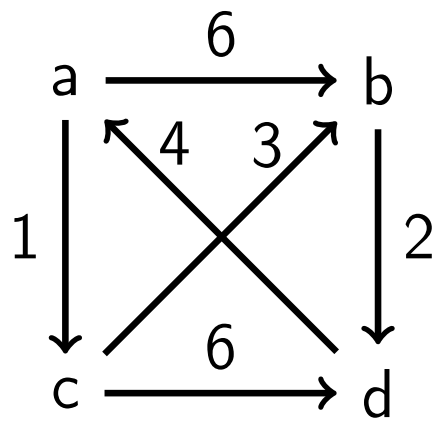
$$D_a = \begin{array}{c} \begin{matrix} a & b & c & d \end{matrix} \\ \begin{bmatrix} 0 & 6 & 1 & \infty \\ \infty & 0 & \infty & 2 \\ \infty & 3 & 0 & 6 \\ 4 & 10 & 5 & 0 \end{bmatrix} \end{array} \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

$$D_b = \begin{array}{c} \begin{matrix} a & b & c & d \end{matrix} \\ \begin{bmatrix} 0 & 6 & 1 & 8 \\ \infty & 0 & \infty & 2 \\ \infty & 3 & 0 & 5 \\ 4 & 10 & 5 & 0 \end{bmatrix} \end{array} \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

$$D_b[a,d] = \min\{D_a[a,b] + D_a[b,d], D_a[a,d]\} = \min\{6+2, \infty\} = 8$$
$$D_b[c,d] = \min\{D_a[c,b] + D_a[b,d], D_a[c,d]\} = \min\{3+2, 6\} = 5$$

Application

G :



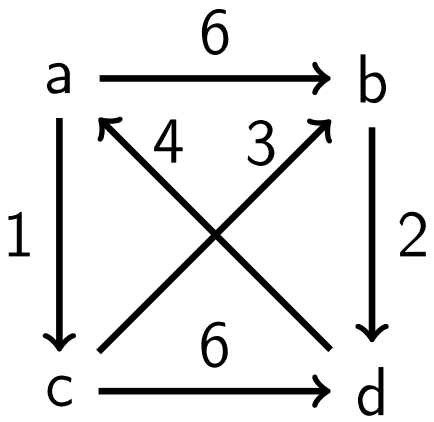
Calculate D_c via vertex c.

$$D_b = \begin{array}{c} \begin{matrix} a & b & c & d \end{matrix} \\ \begin{bmatrix} 0 & 6 & 1 & 8 \\ \infty & 0 & \infty & 2 \\ \infty & 3 & 0 & 5 \\ 4 & 10 & 5 & 0 \end{bmatrix} \end{array} \begin{matrix} a \\ b \\ c \\ d \end{matrix} \qquad D_c = \begin{array}{c} \begin{matrix} a & b & c & d \end{matrix} \\ \begin{bmatrix} 0 & 4 & 1 & 6 \\ \infty & 0 & \infty & 2 \\ \infty & 3 & 0 & 5 \\ 4 & 8 & 5 & 0 \end{bmatrix} \end{array} \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

$$D_c[a,d] = \min\{D_b[a,c] + D_b[c,d], D_b[d,b]\} = \min\{1+5, 10\} = 6$$
$$D_c[d,b] = \min\{D_b[d,c] + D_b[c,b], D_b[d,b]\} = \min\{5+3, 10\} = 8$$

Application

G :



Calculate D_d via vertex d.

$$D_c = \begin{array}{c} \begin{matrix} a & b & c & d \end{matrix} \\ \begin{bmatrix} 0 & 4 & 1 & 6 \\ \infty & 0 & \infty & 2 \\ \infty & 3 & 0 & 5 \\ 4 & 8 & 5 & 0 \end{bmatrix} \end{array} \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

$$D_d = \begin{array}{c} \begin{matrix} a & b & c & d \end{matrix} \\ \begin{bmatrix} 0 & 4 & 1 & 6 \\ 6 & 0 & 7 & 2 \\ 9 & 3 & 0 & 5 \\ 4 & 8 & 5 & 0 \end{bmatrix} \end{array} \begin{matrix} a \\ b \\ c \\ d \end{matrix}$$

$$D_d[b,a] = \min\{D_c[b,d] + D_c[d,a], D_c[b,a]\} = \min\{2+4, \infty\} = 6$$
$$D_d[b,c] = \min\{D_c[b,d] + D_c[d,c], D_c[b,c]\} = \min\{2+5, \infty\} = 7$$
$$D_d[c,a] = \min\{D_c[c,d] + D_c[d,a], D_c[c,a]\} = \min\{5+4, \infty\} = 9$$

Recovering the path?

Like in the previous algorithm, let $\Pi[x, y]$ denote predecessor of y in the shortest path from x to y.

```
FLOYD-WARSHALL(M)
1   n ← size(M)
2   D' ← M
3   for x ← 1 to n
4     for y ← 1 to n
5       Π[x, y] ← x
6   for k ← 1 to n
7     D ← D'
8     for x ← 1 to n
9       for y ← 1 to n
10        D'[x, y] ← min{D[x, k] + D[k, y],
11                       D[x, y]}
12        if D'[x, y] ≠ D[x, y]
13          Π[x, y] ← Π[k, y]
14  return D', Π
```

End of video