Syllabus

UNIT I INTRODUCTION: Introduction to Algorithms – Fundamentals of Algorithmic Problem Solving – Fundamentals of the Analysis of Algorithmic Efficiency – Analysis Framework – Asymptotic Notations and Basic Efficiency Classes – Mathematical Analysis of Recursive Algorithms – Mathematical Analysis of Non-recursive Algorithms

UNIT II DIVIDE-AND-CONQUER: Divide and Conquer Methodology – Binary Search – Merge Sort – Quick Sort – Heap Sort – Multiplication of Large Integers – Strassen's Matrix Multiplication

UNIT III DYNAMIC PROGRAMMING: Dynamic Programming – Change-making Problem – Computing a Binomial Coefficient – All-pairs Shortest-paths Problem – Warshall's and Floyd's Algorithms – 0/1 Knapsack Problem

UNIT IV GREEDY TECHNIQUE: Greedy Technique – Minimum Spanning Tree – Prim's Algorithm – Kruskal's Algorithm – Single-source Shortest-paths Problem – Dijkstra's Algorithm – Huffman Coding – Fractional Knapsack Problem

UNIT V BACKTRACKING AND BRANCH-AND-BOUND: Backtracking – N-Queens Problem – Hamiltonian Circuit Problem – Subset Sum Problem – Branch-and- Bound – Travelling Salesman Problem

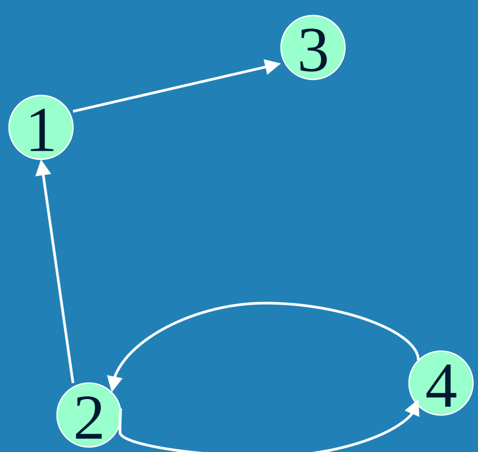
UNIT VI LIMITATIONS OF ALGORITHM POWER: P and NP Problems – NP-Complete Problems – Decision Trees – Information Retrieval – Pattern Matching – Data Science Algorithms

UNIT III DYNAMIC PROGRAMMING:

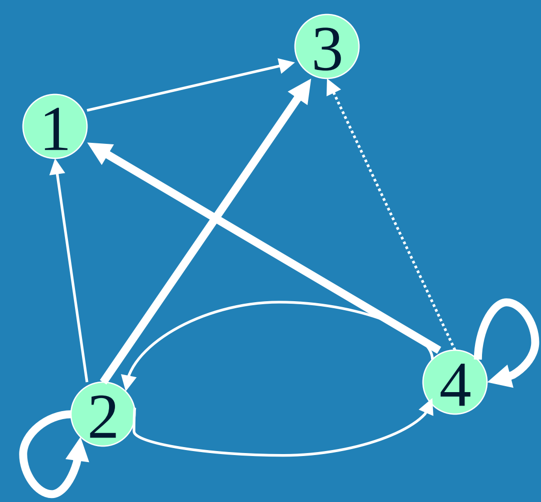
Dynamic Programming – Change-making Problem – Computing a Binomial Coefficient **– All-pairs Shortest-paths Problem – Warshall's and Floyd's Algorithms** – 0/1 Knapsack Problem

Warshall's Algorithm: Transitive Closure

- Computes the transitive closure of a relation
- Alternatively: existence of all nontrivial paths in a digraph
- Example of transitive closure:



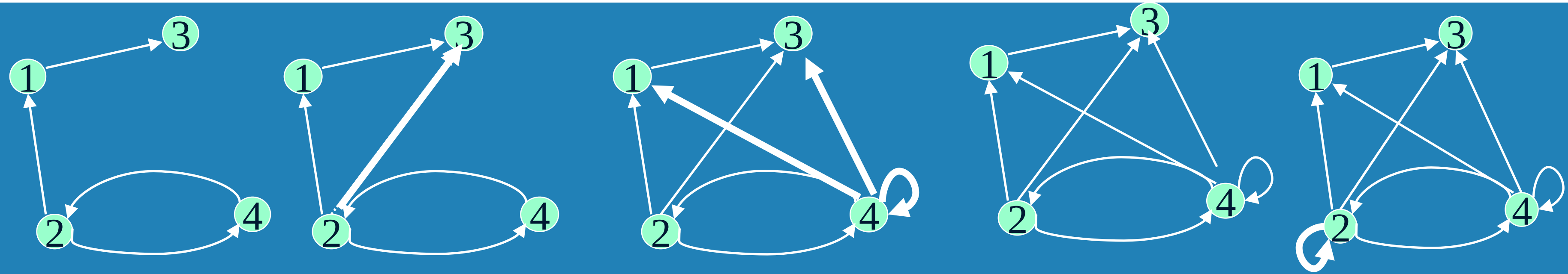
0	0	1	0
1	0	0	1
0	0	0	0
0	1	0	0



0	0	1	0
1	**1**	**1**	1
0	0	0	0
1	1	**1**	**1**

Warshall's Algorithm

Constructs transitive closure T as the last matrix in the sequence of n-by-n matrices $R^{(0)}, \dots, R^{(k)}, \dots, R^{(n)}$ where
$R^{(k)}[i,j] = 1$ iff there is nontrivial path from i to j with only the first k vertices allowed as intermediate
Note that $R^{(0)} = A$ (adjacency matrix), $R^{(n)} = T$ (transitive closure)



$R^{(0)}$

0	0	1	0
1	0	0	1
0	0	0	0
0	1	0	0

$R^{(1)}$

0	0	1	0
1	0	**1**	1
0	0	0	0
0	1	0	0

$R^{(2)}$

0	0	1	0
1	0	1	1
0	0	0	0
1	1	**1**	**1**

$R^{(3)}$

0	0	1	0
1	0	1	1
0	0	0	0
1	1	1	1

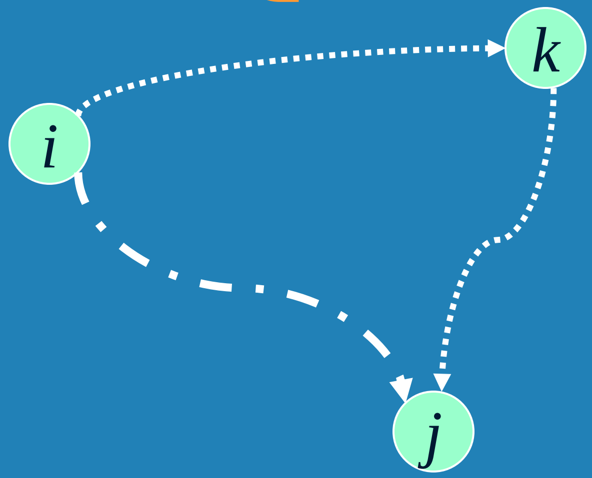
$R^{(4)}$

0	0	1	0
1	**1**	1	1
0	0	0	0
1	1	1	1

Warshall's Algorithm (recurrence)

On the *k*-th iteration, the algorithm determines for every pair of vertices *i, j* if a path exists from *i* and *j* with just vertices 1,…,*k* allowed as intermediate

$$R^{(k)}[i,j] = \begin{cases} R^{(k-1)}[i,j] & \text{(path using just } 1,\ldots,k\text{-1)} \\ \text{or} & \\ R^{(k-1)}[i,k] \text{ and } R^{(k-1)}[k,j] & \text{(path from } i \text{ to } k \text{ and from } k \text{ to } j \text{ using just } 1,\ldots,k\text{-1)} \end{cases}$$



Initial condition?

Warshall's Algorithm (matrix generation)

Recurrence relating elements $R^{(k)}$ to elements of $R^{(k-1)}$ is:

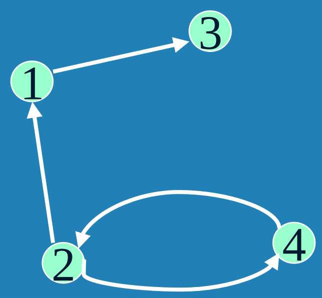
$$R^{(k)}[i,j] = R^{(k-1)}[i,j] \text{ or } (R^{(k-1)}[i,k] \text{ and } R^{(k-1)}[k,j])$$

It implies the following rules for generating $R^{(k)}$ from $R^{(k-1)}$:

Rule 1 If an element in row i and column j is 1 in $R^{(k-1)}$, it remains 1 in $R^{(k)}$

Rule 2 If an element in row i and column j is 0 in $R^{(k-1)}$, it has to be changed to 1 in $R^{(k)}$ if and only if the element in its row i and column k and the element in its column j and row k are both 1's in $R^{(k-1)}$

Warshall's Algorithm (example)



$$R^{(0)} = \begin{matrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{matrix}$$

$$R^{(1)} = \begin{matrix} 0 & 0 & 1 & 0 \\ 1 & 0 & \mathbf{1} & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{matrix}$$

$$R^{(2)} = \begin{matrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ \mathbf{1} & 1 & \mathbf{1} & \mathbf{1} \end{matrix}$$

$$R^{(3)} = \begin{matrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{matrix}$$

$$R^{(4)} = \begin{matrix} 0 & 0 & 1 & 0 \\ 1 & \mathbf{1} & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{matrix}$$

Warshall's Algorithm (pseudocode and analysis)

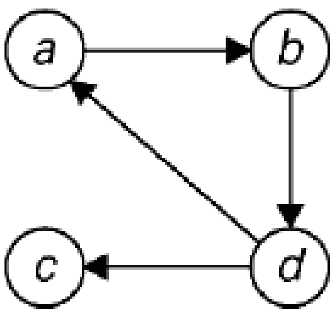
ALGORITHM *Warshall*$(A[1..n, 1..n])$

//Implements Warshall's algorithm for computing the transitive closure
//Input: The adjacency matrix A of a digraph with n vertices
//Output: The transitive closure of the digraph
$R^{(0)} \leftarrow A$
for $k \leftarrow 1$ **to** n **do**
  for $i \leftarrow 1$ **to** n **do**
    for $j \leftarrow 1$ **to** n **do**
      $R^{(k)}[i, j] \leftarrow R^{(k-1)}[i, j]$ **or** $(R^{(k-1)}[i, k]$ **and** $R^{(k-1)}[k, j])$
return $R^{(n)}$

Time efficiency: Θ(n^3)

Space efficiency: Matrices can be written over their predecessors (with some care), so it's Θ(n^2).

Warshall's Algorithm: Transitive Closure



(a)

$$A = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

(b)

$$T = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

(c)

(a) Digraph. (b) Its adjacency matrix. (c) Its transitive closure.

Warshall's Algorithm (matrix generation)

Recurrence relating elements $R^{(k)}$ to elements of $R^{(k-1)}$ is:

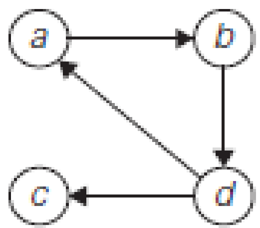
$$R^{(k)}[i,j] = R^{(k-1)}[i,j] \text{ or } (R^{(k-1)}[i,k] \text{ and } R^{(k-1)}[k,j])$$

It implies the following rules for generating $R^{(k)}$ from $R^{(k-1)}$:

Rule 1 If an element in row *i* and column *j* is 1 in $R^{(k-1)}$, it remains 1 in $R^{(k)}$

Rule 2 If an element in row *i* and column *j* is 0 in $R^{(k-1)}$, it has to be changed to 1 in $R^{(k)}$ if and only if the element in its row *i* and column *k* and the element in its column *j* and row *k* are both 1's in $R^{(k-1)}$

Thank You



$R^{(0)}$ =

	a	b	c	d
a	0	1	0	0
b	0	0	0	1
c	0	0	0	0
d	1	0	1	0

1's reflect the existence of paths with no intermediate vertices ($R^{(0)}$ is just the adjacency matrix); boxed row and column are used for getting $R^{(1)}$.

$R^{(1)}$ =

	a	b	c	d
a	0	1	0	0
b	0	0	0	1
c	0	0	0	0
d	1	**1**	1	0

1's reflect the existence of paths with intermediate vertices numbered not higher than 1, i.e., just vertex *a* (note a new path from *d* to *b*); boxed row and column are used for getting $R^{(2)}$.

$R^{(2)}$ =

	a	b	c	d
a	0	1	0	**1**
b	0	0	0	1
c	0	0	0	0
d	1	1	1	**1**

1's reflect the existence of paths with intermediate vertices numbered not higher than 2, i.e., *a* and *b* (note two new paths); boxed row and column are used for getting $R^{(3)}$.

$R^{(3)}$ =

	a	b	c	d
a	0	1	0	1
b	0	0	0	1
c	0	0	0	0
d	1	1	1	1

1's reflect the existence of paths with intermediate vertices numbered not higher than 3, i.e., *a*, *b*, and *c* (no new paths); boxed row and column are used for getting $R^{(4)}$.

$R^{(4)}$ =

	a	b	c	d
a	**1**	1	**1**	1
b	**1**	**1**	**1**	1
c	0	0	0	0
d	1	1	1	1

1's reflect the existence of paths with intermediate vertices numbered not higher than 4, i.e., *a*, *b*, *c*, and *d* (note five new paths).