Memo No. _____
Date / /

Mo Tu We Th Fr Sa Su

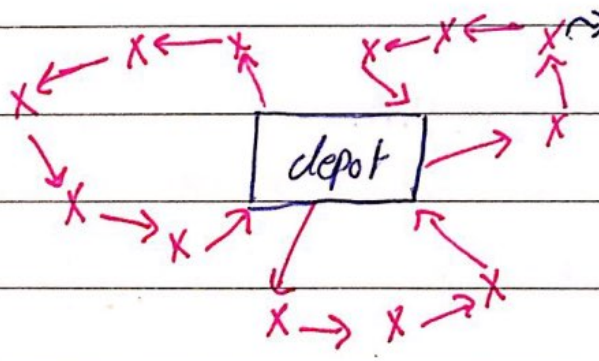
* Short haul Freight Transportation 8-

- Graph theory :- [http://people.hofstra.edu/geotrans]

Short haul : pickup/ delivery of goods in relatively small area using fleet of vehicles

- unless stated other wise, vehicles stational at single depot (مستودع)

- Routes include several pick up and/or delivery points
- Routes performed in single work shift



operation تعبير problem

Memo No.
Date / /

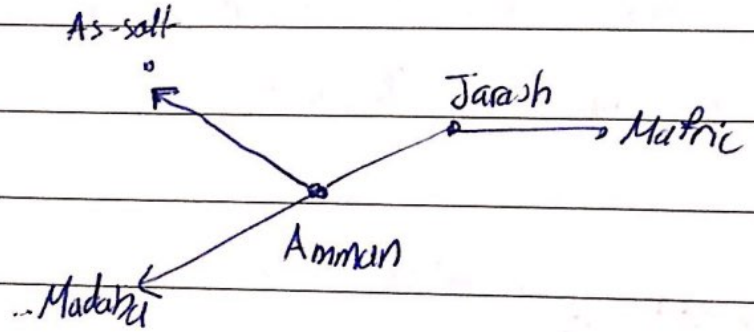
Mo Tu We Th Fr Sa Su

* date knowledge :-

- Static vehicle routing problem :- all customer requests (location, amount of goods, ..) are known in advance.

- Dynamic vehicle routing problem :- all customer requests (location, amount of goods, ..) not all is known in advance

• Rout → planned for set of requests known in advance, dynamically updated to incorporate new requests



* Graph def :- a graph $G=(V,E)$ is a set of vertices/nodes' V connected by a set of edges/arcs A/E

- vertex: terminal point, intersation point

Memo No.
Date / /

Mo Tu We Th Fr Sa Su

Represats city, customs location, depot, road intersection.

- Edge :- link between two vertices

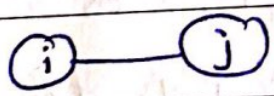
(i, j) → Connects verter i to j

- Arrow representation direction

Head node i

Tail node j

ملوش اتجاه



→ bidirectional

باتجاهين

- Edge/arcs have number of characteristics

Cost C_{ij} → cost (time, distance, money, ...) associate with i to j

ممكن يكون

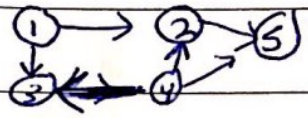
capacity V_{ij} → maximum flow on edge (i, j)

traffic, amount of goods

Mo Tu We Th Fr Sa Su

Memo No. _____
Date / /

* order of a graph :- # of vertices |V| ← nodes
* size of a graph :- # of edges |E|



$V = \{1, 2, 3, 4, 5\}$

$E = \{(1,2)(1,3)(3,3)(4,2)(2,5)(4,5)\}$

order = 5

size = 6

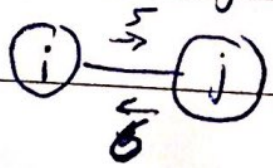
* Structural properties of Graphs :-

① undirected versus directed

↳ connection between i & j
implies connected between j & i

→ symmetrical network / directed ⇒ symmetr
flow in i to j is the same at j to i

→ asymmetric
flow $i \to j \neq$ flow $i \leftarrow j$



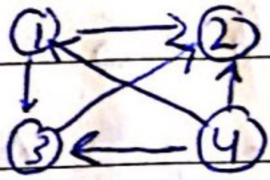
bidirectionne

Memo No. ______
Date / /

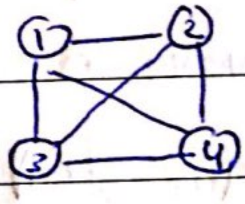
Mo Tu We Th Fr Sa Su

* **completness:** a graph is complete if an edge exists between every pair of vertices (regardless of direction)

واحد مرتبط بالكل وكلهم نفس الـ
فهو complet



complet undircted ←



→ how many arces/edges in complete undirection graph

size = $(n-1)+(n-2)+(n-3)+\dots + (n-(n-1))$ ← 1

↳ $(n-1)$ term عدد

Avg edg = $\frac{n-1+1}{2} = \frac{n}{2}$

# of edges to complet graph total = $\frac{n}{2}(n-1)$

هي نفسها size ده ✓

Memo No. _____
Date / /

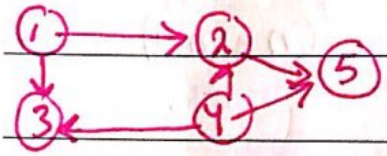
Mo Tu We Th Fr Sa Su

* paths, cycles, Tress :-

For the following definitions, assume there are n vertices in V an m edge in E

- path : apath in G is sequence of vertices $\{i_1, i_2, i_3 \dots i_n\}$ such that
  - $i_k \in V \quad \forall\, k=1,\dots,p$
  - i_k unique $\forall\, k=1,\dots,p$ (no vester is visited more than once)
  - (i_k, i_{k+1}) or $(i_{k+1}, i_k) \in E \quad \forall\, k=1,\dots,p-1$ (on time)

i_k unique: يعني انو كل وحدة مرة وحدة



path from 1 to 5 : {1, 2, 5}

بالعكس الاتجاه ← {1, 2, 4, 5}

not path from 1 to 5: {1, 3, 5} → disconnected

{1, 2, 4, 3, 1, 2, 5} → فيه تكرار

→ Path not necessarly a sequance of feasible directions

* Directed path :-

additional condition $(i_k, i_{k+1}) \in E \quad \forall\, k=1,\dots,p-1$

is a feasible sequence of decisions

{1, 2, 5} → directed path

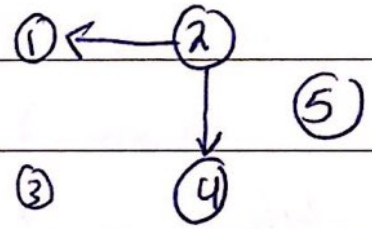
Mo Tu We Th Fr Sa Su

Memo No.
Date / /

* connectedness :-

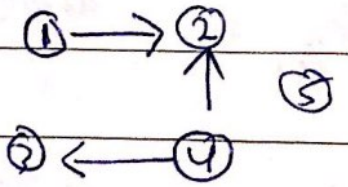
* connected : two vertices i & j are connected in G if there exists a path from i & j
↳ regardless of direction

* adjacency is a special case of connectedness
If i & j are connected by a path of ~~length~~ length 1 ⇒ i & j are adjacent



nodes 1 & 4 are connected
nodes 1 & 2 are adjacent, connected ⇒

a graph is connected if every pair of vertices in the graph are connected



✓

this graph is not connected

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

Cycle : set of edges that form a closed loop in Graph.

→ begin from a vertex follow only edges in the cycle regardless of their direction and then return to the same vertex

*A cycle C in G is a sequence $\{i_1, i_2, \ldots i_p, i_1\}$ such that :-

loop برجع لأول ال

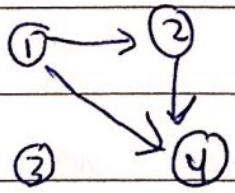
$i_k \in V \quad k = 1, \ldots, p$

i_k unique $\forall k = 1, \ldots, p$

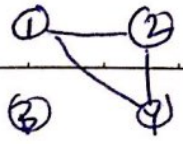
→ (i_k, i_{k+1}) or $(i_{k+1}, i_1) \in E \quad \forall k = 1, \ldots, p-1$

either (i_p, i_1) or $(i_1, i_p) \in E$

*directed network :-



it's cycle $\{1, 2, 4, 1\}$ but Not feasible



it's cycle $\{1, 2, 4, 1\}$ and feasible

Mo Tu We Th Fr Sa Su

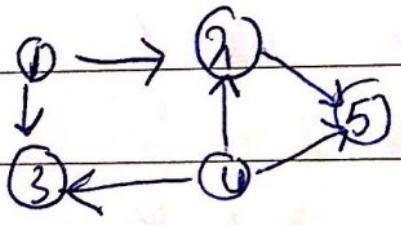
Memo No.
Date / /

* Tree :- A tree is Graph is a collection of edges such that all vertices connected an no cycle exists

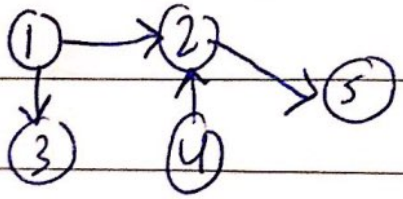
• Equivelent definition :- n vertices in the network.
→ A tree is aset of n-1 edges such that all vertices are connected

• Important proporty of atree: there is a unique (not neccessrly directed) path between each pair of vertices

directed network



it's not tree it's cycle "3 cycle"

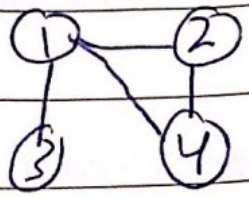


it's tree

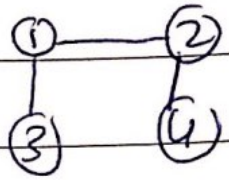
Memo No.
Date / /

Mo Tu We Th Fr Sa Su

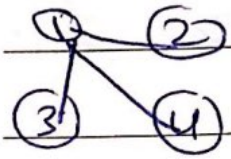
undirected



it's not tree it's cycle



it's tree, 3 edge



it's tree

- what happend if you remove an edge?
  some vertex is no longer connected, so the Graph become not connected
- what happend if you add an edge?
  creat a cycle

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

Minimum cost path models

* cost can be time, distance, $, ...

- Single-pair shortest path problem
  ⇒ find the shortest path between a single source node S, and a destination node t.

- Singe-source shortest path problem
  ⇒ to find the shortest path between a single source node "S", and all other vertices node

- single-destination shortest path problems
  ⇒ to find the shortest path between a single destination from all vertices

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

• All pairs shortest path problem

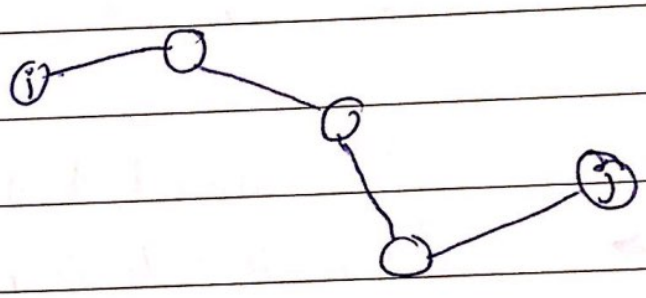
⇒ to find shortest path between all pairs of vertices

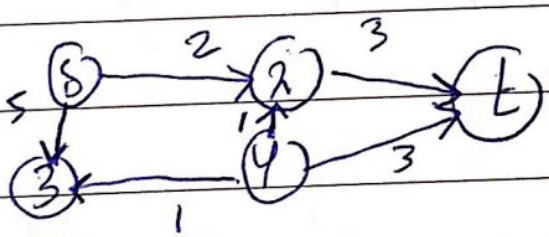
بين كل نقطتين اقصر مسافة للكل

* Formulation :- cost of C_{ij} of $(i,j) \in E$

Problem to find path "p" from s to t that minimize

$$\sum_{(k,j) \in p} C_{ij}$$





path from s to t → {s, 2, t} → cost = 5 ✓ the best

{s, 3, 4, t} → cost = 9

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

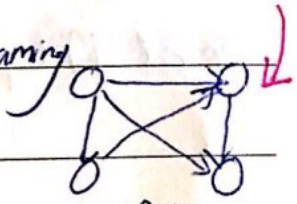
* Solving minimum cost path :-

[1] Dijkstra's algorithm : single pair → Single source
→ single destination

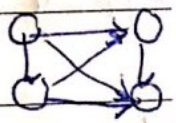
required $c_{ij} \geq 0$

- pros :- - Fast run times for dense networks $m \to n^2$
- can be truncated if you have the path of interest

- cons :- - Alternative method faster for sparse networks
- efficient implemetation requires Programing



sparse and dense



papé's modification

[2] Bellman-Ford : single-source, allows $c_{ij} \leq 0$

- pros :- - Fas for sparse networks
- simple to implement

- cons :- - cannot be truncated
- may be slow on dense networks

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

* Dijkstra algorithum :- assumtion $C_{ij} \geq 0$
it is a labeling algorithum in which at each iteration, a different vertex P is given a permunent kable V(j) that indicates the cost of the shortest path from S to P

Natation :-

- V(j): Label of vertex j → denes the cost of path from S to j
- pred(j): predecessor of node j on shortest path from S to j
- PERM: set of permently labeled nodes
- $\overline{PERM}$: not permanent

[1] * Initialization

$$V(j) = \begin{cases} 0 & j = s \\ \infty & j \neq s \end{cases}$$

$$pred(j) = \begin{cases} 0 & j = s \\ -1 & j \neq s \end{cases}$$

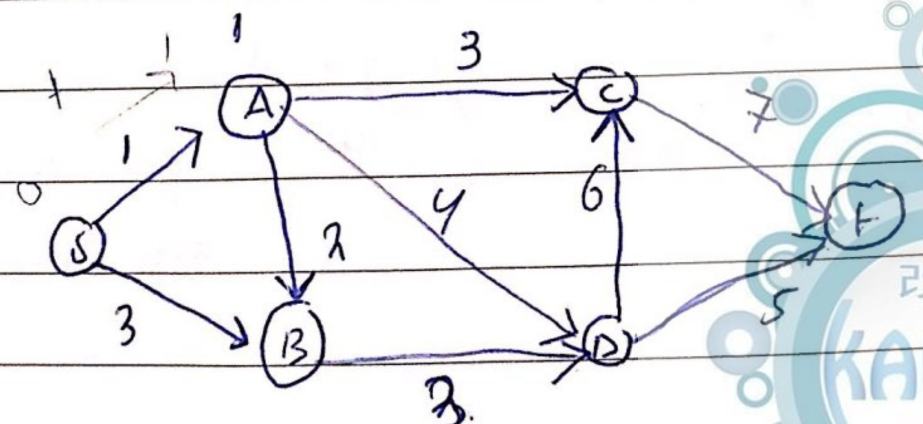
$PERM = \{ \}$, $\overline{PERM} = V$

[2] * Iterations :-

. while $\overline{PERM} \neq \emptyset$

1. let $p \in \overline{PERM}$ be anode for which $V(p)$ &
$V(p) = \min\{ V(j) , j \in \overline{PERM} \}$
2. $PERM \leftarrow PERM + \{p\}$
3. $\overline{PERM} \leftarrow \overline{PERM} \setminus \{p\}$ ← ما عملنا طرح تقريباً"
4. For all $(p,j) \in E$, $j \in \overline{PERM}$ leading from p
   if $V(p) + C_{pj} < V(j)$ then
   $V(j) = V(p) + C_{pj}$
   $pred(j) = p$

Ex :-



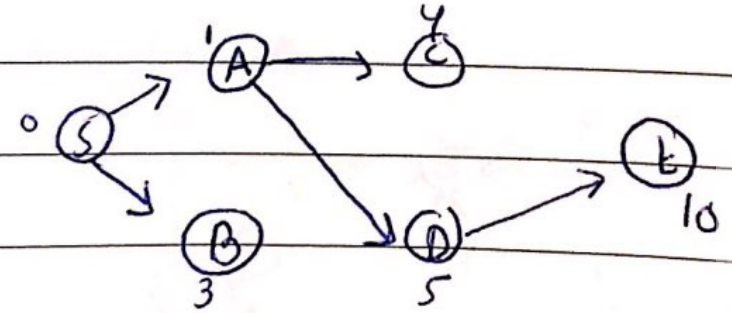
Mo Tu We Th Fr Sa Su

Memo No.
Date / /

V(j) [prev(j)]

Iteration	S	A	B	C	D	T	P	PERM	PERM
Init	0[0]	∞[-1]	∞[-1]	∞[-1]	∞[-1]	∞[-1]	∞[-1]	-	{}	S,A,B,C,D,T
1	0[0]	1[S]	3[S]					S	S	A,B,C,D,T
2				4[A]	5[A]			A	S,A	B,C,D,T
3								B	S,A,B	C,D,T
4								C	S,A,B,C	D,T
5								D	S,A,B,C,D	T
6	0[0]	1[S]	3[S]	4[A]	5[A]		10[D]		S A B C D T	-

shortest Path from S to T



Mo Tu We Th Fr Sa Su

Memo No.
Date / /

* Bellman - Ford :-

Label - correcting algoritum

Initialization

- $v(j) = \begin{cases} 0 & j = s \\ \infty & j \neq s \end{cases}$

- $prad(j) = \begin{cases} 0 & j = s \\ -1 & j \neq s \end{cases}$

- LIST = {s}

Iterations

- while LIST $\neq \emptyset$

1. Remove node p from top of LIST
2. for every (p, j) leading from p
   a. if $v(p) + c_{pj} < v(j)$ then
      i. $v(j) = v(p) + c_{pj}$
      ii. $prad(j) = p$

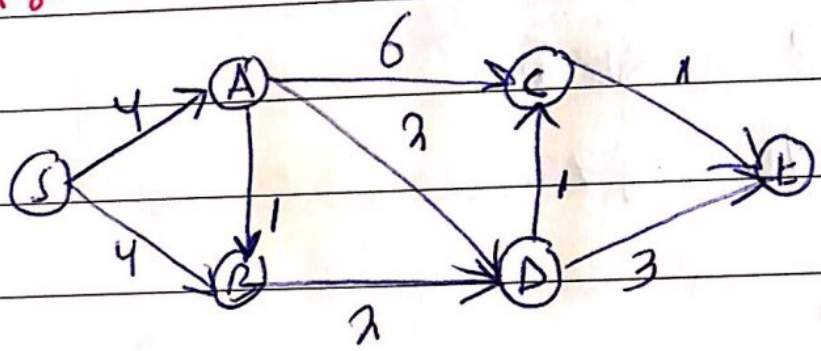
Memo No. _____
Date / /

Mo Tu We Th Fr Sa Su

iii. IF $j \notin$ LIST $\Rightarrow$ LIST $\leftarrow$ LIST + {j}

Pap's modification if $j \notin$ LIST, then if j has never been in LIST but j at end of LIST Else, put j at beginning of LIST

* EX :-



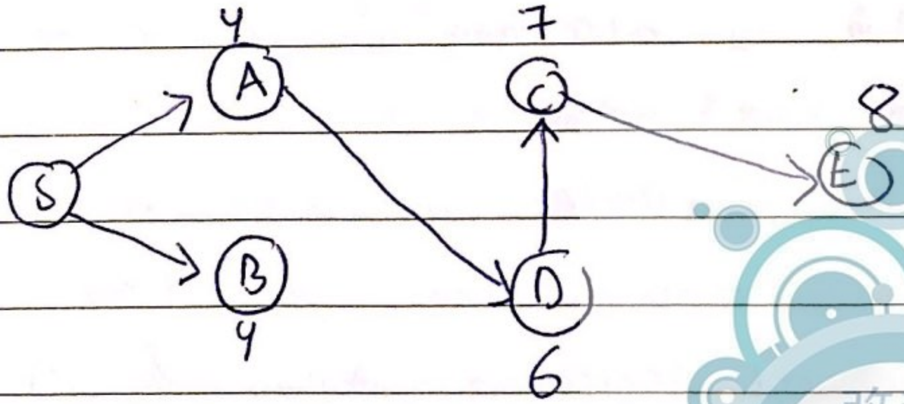
Memo No. ______
Date / /

رايحة وين

iter	S	A	B	C	D	E	P	LIST
init	0[0]	∞[-1]	∞[-1]	∞[-1]	∞[-1]	∞[-1]	—	S
1		4[S]	4[S]				S	A,B
2				10[A]	6[A]		A (top of the LIST)	B,C,D
3							B	C,D
4						11[C]	C	D,E
5				7[D]		9[D]	D	C,E
6						8[C]	C	E
7	0[0]	4[S]	4[S]	7[D]	6[A]	8[C]	E	

رجعناها كانت بال LIST ما منضيف بعضيا لاه

→ the shortes path

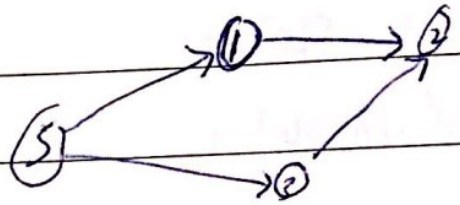


Memo No. _____
Date / /

Mo Tu We Th Fr Sa Su

* Multi-Lable algorithum for shortest paths :-

- when blancing multiple objechives, distance & time
- vertices given labels with two components (dk, tk)
  - dk : first cost component e.g. total distance required to reach this vertex along some path from S
  - tk : second cost component e.g : total time



- Vertices may have multiple labels resulting from diffrat Path
- one path to vertex many have longes distance but shortes time
- Instead of a vertex having a predecessor label, each cost component label (dk, tk) has an associated predecessor Laber → pred (k)

Mo Tu We Th Fr Sa Su

Memo No. _____
Date / /

⊙ only maintain non-dominated labels at each vertex

● A lable (d_q, t_q) is dominated by (d_k, t_k) if both $d_q > d_k$ & $t_q > t_k$

⊙ Maintain a LIST of labels instead of vertices :-

✱ Initialization :-

• origin vertex S given the laber $(0,0)^S_1$ where
indicates the vertex where the lable exists
indicates lable number k=1

⇒ The predeccsson of this label is 0 pred(k) = pred(1) = 0

◉ No lables given to other vertices

• LIST = $\{(0,0)^S_1\}$

⊙ r = 2 "keep track of number of next label generate

✱ Iterations :-

⊙ while LIST ≠ {}

➔ 1. Remove lable k from beginning of LIST

Suppose Lable k has components such that it exists at vertex p

Memo No. _____
Date / /

Mo Tu We Th Fr Sa Su

2. For every edge (p,j) leading from vertex p:
   a. create anew label rat vertex j where
      $(d_r, t_r) = (d_k + d_{pj}, t_k + t_{pj})$
   b. Define pred(r) = k
   c. Add Laber r to end of LIST
   d. compate Label r to each existing Lable (d_q, t_q) curently at vertex j
      i. If label r is dominated by labelq, discard the new label r and remove it from LIST
         $d_r \geq d_q$ and $t_r \geq t_q$
      ii. Else, if label q is dominated by r, discard q
      iii. Else, retain both
   e. r = r+1

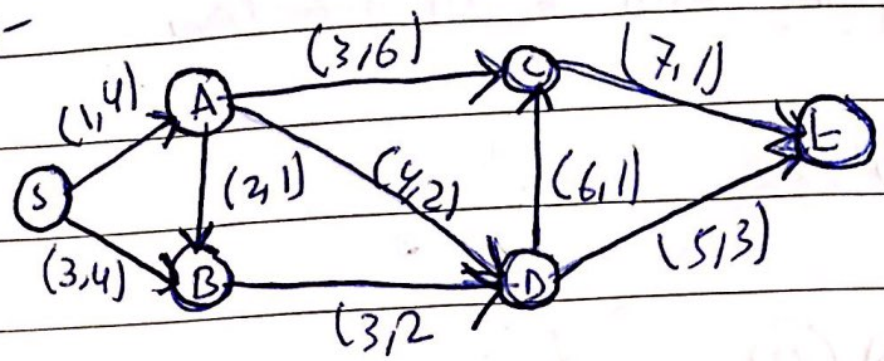
→ stopping rule: when no labels in LIST
→ multi paths

Memo No. _____

Date / /

Mo Tu We Th Fr Sa Su

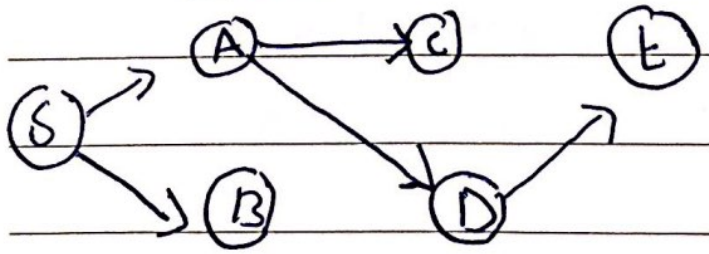
EX :-



S	A	B	C	D	L	LIST	r
$(0,0)^S_1$;0	$(1,4)^A_2$;1	$(3,4)^B_3$;1	$(4,10)^C_4$;2	$(5,6)^D_5$; 2	~~$(11,11)^t_8$;4~~	~~$(0,0)^S_1$~~	~~2~~
	~~$(3,5)^B_6$;2~~	$(11,7)^C_9$;5	~~$(6,6)^D_7$;3~~	$(10,9)^t_{10}$;5	$(18,8)^t_{11}$;9	~~$(1,4)^A_2$~~	~~3~~
						~~$(3,4)^B_3$~~	~~4~~
						~~$(4,10)^C_4$~~	~~5~~
						~~$(5,6)^D_5$~~	6
						~~$(3,5)^B_6$~~	7
						~~$(6,6)^D_7$~~	8
						~~$(11,11)^E_8$~~	9
						$(11,7)^C_9$	10
						$(10,9)^t_{10}$	11
						$(18,8)^E_{11}$	

First cost comparcat: distance

shortest distance path to all nods from S



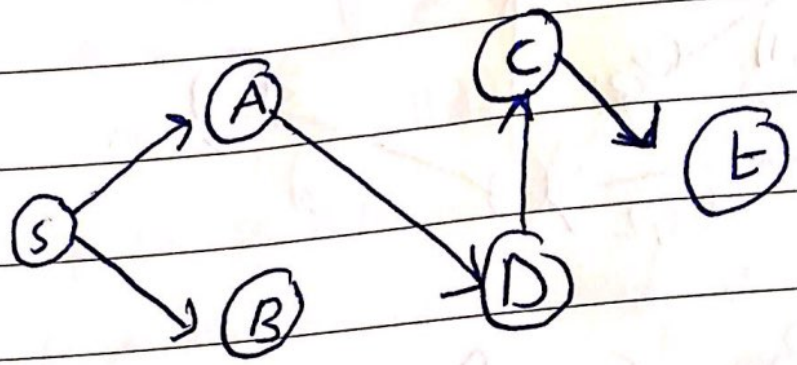
Memo No. _____

Date / /

Mo Tu We Th Fr Sa Su

Secound cost: compound : time

Shortest [time] path to all nodes from s



Mo Tu We Th Fr Sa Su

Memo No.
Date / /

* Minimum Spanning Tree :-

- Definition of a tree:

a tree is a collection of n-1 edges such that all vertices in G are connected, no cycles exist

- A minimum spanning Tree (MST) T^* in $G(V, E)$ is a tree such that the total cost of all arcs in the tree is minimize

$$T^* = \arg \min_{T \in G} \left[\sum_{i,j \in T} c_{ij} \right]$$

* Question :- is MST the same as shortest path tree?

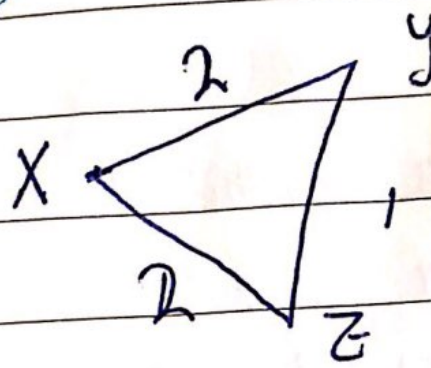
No, the sp tree minimizes length shortest path(s) from some origin node to all other nodes, while the MST minimize the total cost of all arcs in the tree

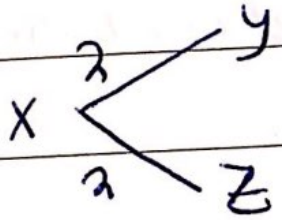
Memo No.

Date / /

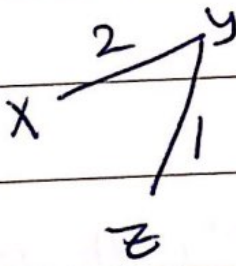
Mo Tu We Th Fr Sa Su

Ex:-





$\Sigma C_{ij} = 4$ ← SPT



$\Sigma C_i = 3$ ← MST

Mo Tu We Th Fr Sa Su

Memo No. _____
Date / /

* Two algorithms :- MST
- kruskal's algorthum
- pim's algorthum

→ kruskal's algorthum

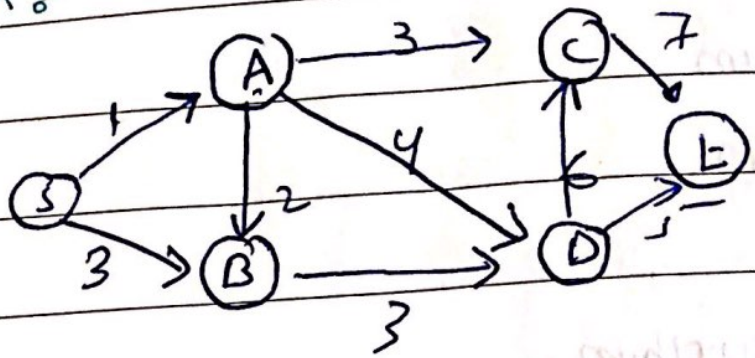
- Initialization :-
Tree = { }

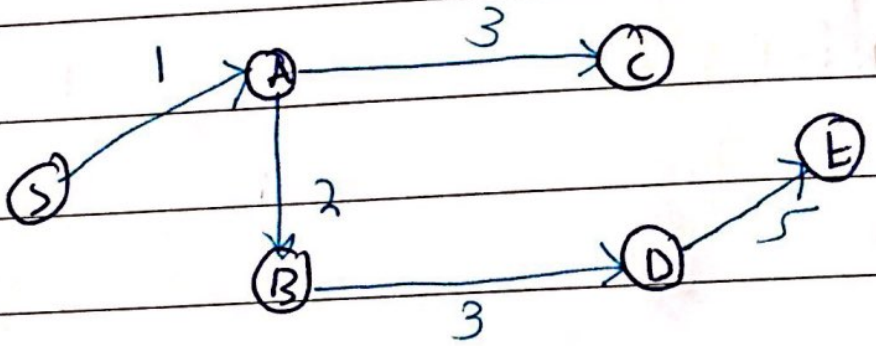
- Iterations :-
sort arcs in order of non-direcing C_{ij} in b.LIST
while |TREE| < n-1
Remove (i,j) from top of LIST
* add (i,j) to Tree if doing so does not creat a cycle
at completion
$T^* = TREE$

Memo No.
Date / /

Mo Tu We Th Fr Sa Su

Ex:-



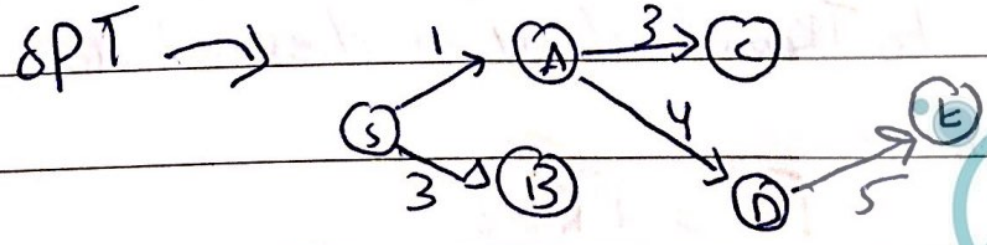


arcs = n-1
= 6-1 = 5

يوقف اول ما أوصل ٥

① حاتهم اني ما تعمل cycle

② بروح على اقل اقيم بختار طريق موجه



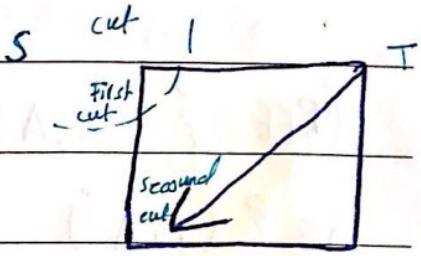
cost = 16

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

* Prim's Algorithm :-

Cut in a network : suppose vertices in G partitioned into two independent subsets S & T
The cut set (S,T) is given by all arc's (i,j) such that : $i \in S$ & $j \in T$ or
$j \in S$ & $i \in T$



* Algorithm :-

- Initialization →

. TREE = { }

. S = {1} , T = V \ S (S is the vertices set)

- Iteration →

while |TREE| < n-1

* Find arc (i,j) ∈ (S,T) with minimum cost c_{ij}

. add (i,j) to TREE

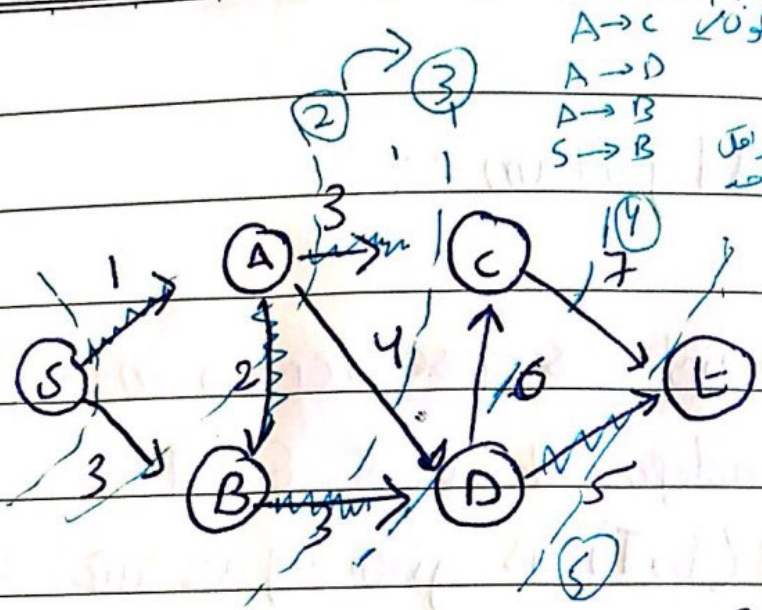
. Remove j from T add to S

Stop شرط توقف

Memo No.
Date / /

Mo Tu We Th Fr Sa Su

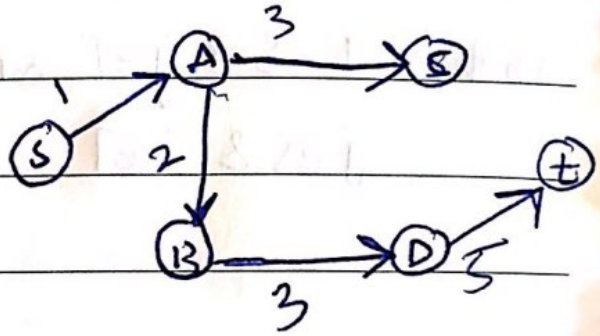
EX:-



① TREE = { (S,A) }

S = { 1, A }

T = { B, C, D, t }



② TREE = { (S,A), (A,B) }

S = { 1, A, B }

T = { C, D, t }

③ TREE = { (S,A), (A,B) (A,C) }

S = { 1, A, B, C }

T = { D, t }

Mo Tu We Th Fr Sa Su

Memo No. _____

Date / /

(4) TREE = { (S,A), (A,B) (A,C) (B,D) }

δ = { I, A, B, C, D }

T { F }

(5) TREE { (S,A), (A,B), (A,C) (B,D) (D,E) }

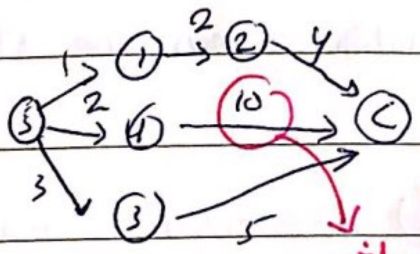
δ = { I, A, B, C, D, E }

T = { }

* Miniumum spaning Tree Applications :-

Standard minuimum cost path (SP) ⇒ are called mini-sum path; because among all paths P conneting s to t
→ the minimum path P_s^* has minimum total cost:

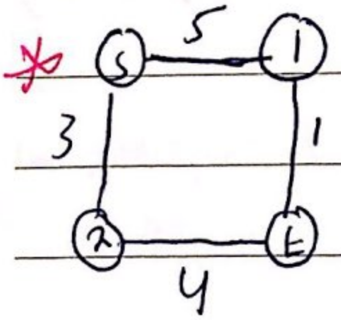
$$P_s^* = \arg\min_{P \in G} \sum_{(i,j) \in P} c_{ij}$$



هاد ملغي
لانه اكبر رقم فانا بهتم
بال min cost للarrow

→ mini-max paths (P_m^*) encounters the Smallest maximum arc cost

$$P_M^* = \arg\min_{P \in G} \left(\max_{(i,j) \in P} c_{ij} \right)$$



Path from s t	$\sum c_{ij}$	max c_{ij}
P_s^* s – 1 – t	6 ← mini.sum	5
P_M^* s – 2 – t	7	4 ← mini.max

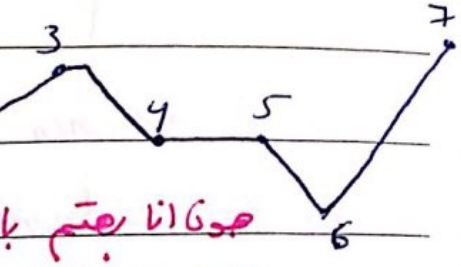
Memo No.
Date / /

Mo Tu We Th Fr Sa Su

* useful applications of mini-max paths :-

(1) Elevation change

very steep road segments ↑ fuel & ↑ maintenance



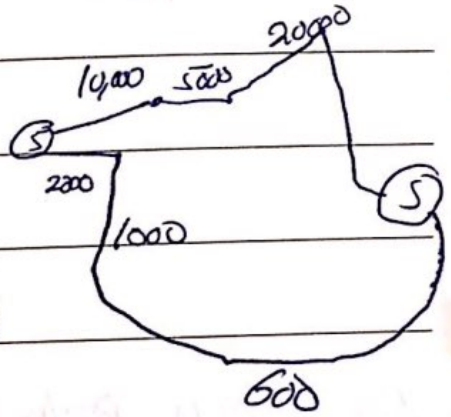
هون انا بهتم بالارتفاعات مثلاً
لانه بهتم بالبنزين أكثر صيانة

⇒ A void road segment with a maximum elevation change

(2) Hazardous materials

if material spilled population hazard ⇒ A void densely population

⇒ minimu. max population exposed



هون انا بهتم بالمسافة
وبالتكلفة انا بهتم
انه يكون في اقل عدد سكان
لانه هي مواد خطيرة

* The minimum path problem is solved by solving sp problem

* The minu. max problem is solved by solving minimun spring tree problem

Mo Tu We Th Fr Sa Su

Memo No. _____
Date / /

* Vehicle Routing problems :-

vehicle departs from anode 8 (link depot) visiting a number of locations (1, 2, ... P) and returning to node 8. The objective is to chose the sequence of nodes 1, 2, ... P which minimizes the cost of circuit

Q: How is this different from pathing problems we have been studying? It considers returing to the same starting point (cycle)

شخص عم بلف ورجع بالنهاية لنفس نقطة البداية

When routing only a single vehicle, → we refere to this problem as Traveling salesman problem TSP

When a number of whicles rout → VRP genera case

* Triangle-inequality :- The first step in solving routing problems is to develop a complete undirected network $G = (V, E)$ satisfying the triangle-inequality

Memo No.
Date / /

Mo Tu We Th Fr Sa Su

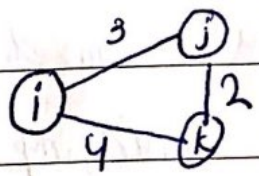
, where V is limited to the nodes that will be visited (includes soura. S, & 1, 2, ..., P)

- A complete network $G = (V, E)$ satisfies the Δ-inquality if and only if "iff" $\forall\ i, j, k \in V$, the following relationship holds $\rightarrow$ $C_{ik} \leq C_{ij} + C_{jk}$

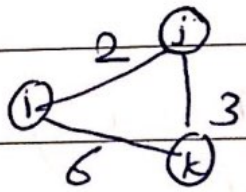
$4 \overset{?}{\leq} 3 + 2$ ✓

$2 \leq 4 + 3$

$3 \leq 2 + 4$



$6 \not\leq 2 + 3$ No ulp ←



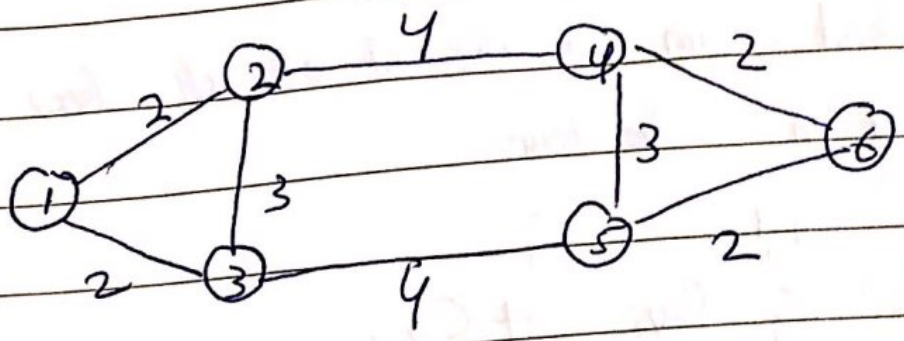
* Generating a complete undirected network satisfying the Δ- inequality :-

Assumption a path exists between all nodes in V' in G' and cost from i to j does not differ from the cost from j to i

goal ←

Memo No. _____
Date / /

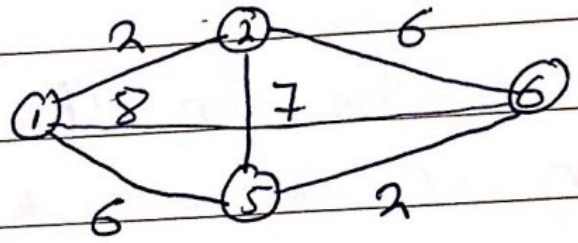
Mo Tu We Th Fr Sa Su



$G' = \{V', E'\}$

حاول يكون بدون اباه

$V' = \{1,2,3,4,5,6\}$, $V = \{1,2,5,6\}$



لازم يكون Complete

*TSP tour on a complete undirected (symmetric) network $G(V,E)$:- let n be the number of nodes in V A TSP tour T is a sequence of all nodes in V, $T = \{i_1, i_2, \dots i_n, i_1\}$ where $i_k \in V$ and each i_k unique $\forall k \neq 1, \dots n$

$T = \{1, 2, 6, 5, 1\}$

Mo Tu We Th Fr Sa Su Date / /

Tour cost: Sum of the cost of all edges traversed in the tour

$$C(T) = \sum_{i=1}^{n-1} C_{v_i v_{i+1}} + C_{v_n v_1}$$

SP

optimal TSP tour: $TSP = \arg\min_{T \in G} C(T)$

* An inefficient algorithm for TSP :-
  - Enumerate every tour in the network
  - choose the one the lowest cost

$(n-1) * (n-2) * (n-3) \ldots = (n-1)!$

$S \xrightarrow{10}$ 10 nodes * Tour $= 10! = 3,628,800$ ← ثلاثة ملايين وستمائة وثمانية وعشرون ألف وثمانمائة

→ Algorithm :- it guerantees to converge to a solution

→ Heuristic :- solution feasible حل ممكن

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

* Heuristic algorithms :-

- provides a feasible solution that may or may not be optimal
- Good heuristic
  - provide solutions close to optiopal
  - Efficient in run time

⇒ Desiruble property of heuristic

- Ability to compare quality of solutions produced by the heuristic to the optimal solution
- worst-case quality of solutions produce can be guaranteed
  - (A) These 'guartees" (Maximum deviations from optimality) developed are produced using worst-case analysis.
  - (B) If the cost of optimal solution is C^*, then heuristic finds a solution with cost no greater than αC^* ((minimization problem))

Mo Tu We Th Fr Sa Su

Memo No. _____
Date / /

* Heuristics for TSP :-

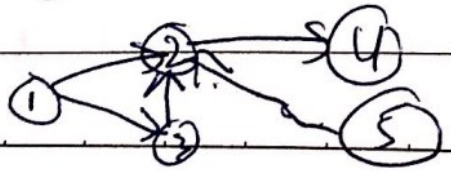
1. Christofides heuristic
2. Constructive heurstic → nearest neighbor
   → nearest insertion
   → farthest insertion
   → cheapest insertion
3. Improvement heuristics : 2- opt
   k -opt

→ Christofides heursistic :-

- Assumption complete network satisfying the Δ- inequality

→ few definitions

- Node degree, def the degree of a node $i \in V$ with respect to an edge set E (→ how many edge enter / leave i) is the number of edge in E having i as an edge point



node (2) → 4

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

* perfect node matching def:
Give a set of nodes S such that the number of nodes in S is given, a perfect node matching W is a set of |S|/2 node pairings such that each node in S is in exactly one pairing

→ in a complete network, each node pairing represent an arc (i,j)

→ Christofides heursistic

- Iterations:-

- Given an undirected complete network G = (V, E) satisfying the Δ-inequality, find a minimum spaoing tree (MST*) on G
- let S be the nodes of G with odd degree with respect MST*
* Find the minimum cost perfect node matching W* of the nose in S

بس برد كتير

Mo Tu We Th Fr Sa Su

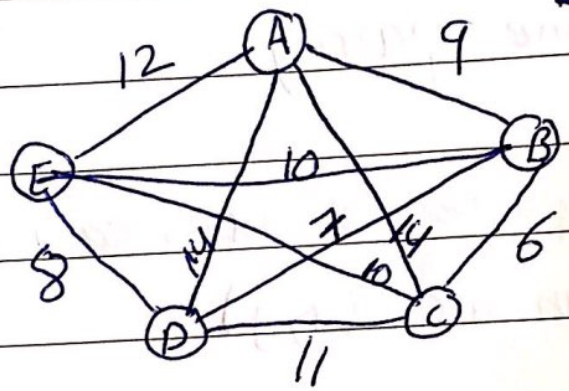
Memo No. _____

Date / /

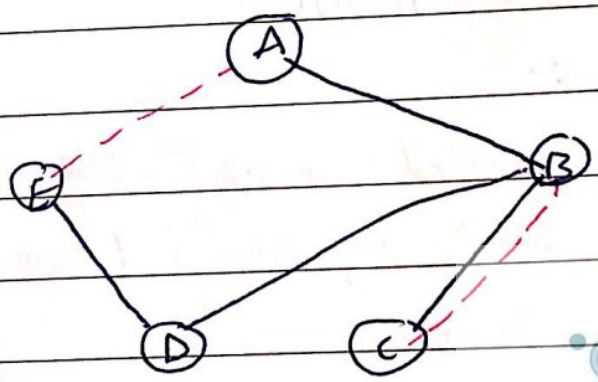
* find $R = W^* \cup MST^*$
* Generate atour T^c of the nodes using each of the arcs in the set $MST^* \cup W^*$

EX:-

complet ليه كده fix Graph



(1) MST



(2) * nodes with odd degree with respect to MST

A, B, C, E

Memo No.
Date

Mo Tu We Th Fr Sa Su

③

AB 9
CE 10
19

AC 14
BE 10
24

AE 12
BC 6
18

W*

④ W* U MST*

←

⑤ A - E - D - B - C - B - A → يتكرر مرتين

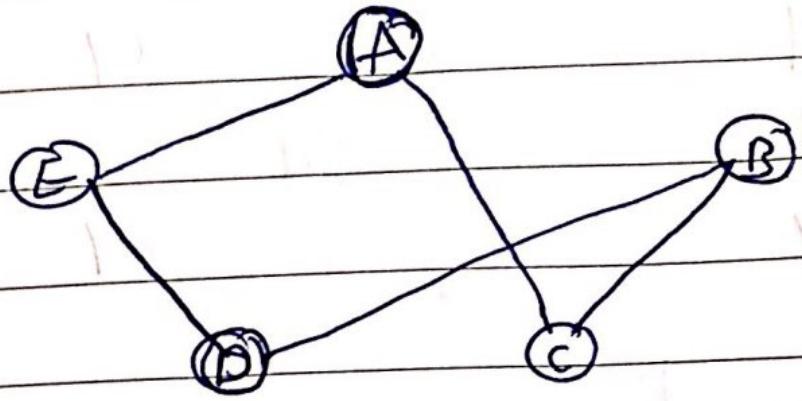
لازم ارجع كل نقطة البداية

ET = {A, E, D, B, C, A}

Final أو Tour →

Mo Tu We Th Fr Sa Su

Memo No. _____
Date / /



$C(T) \leq 47$

→ let T^c be a tour produced by a christofides.

Theorem, worst-case performance bound for christofides

$$Cost(T^c) \leq \frac{3}{2} C(TSP^*)$$

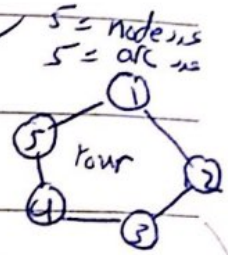
Mo Tu We Th Fr Sa Su Sajeda Naser Fouri

Memo No.

Date 28/3/2019

1 * A Nearest neighbor heuristic :-

Assumption :- complete undirected network satisfying Δ-inequality



→ Initialization : 2-options

- let $p = \{i, j, i\}$ where $C_{ij} = \min_{(i,j) \in E} C_{ij}$

OR

- let i be arbitrary start node, find $j \in V \setminus \{i\}$ such that C_{ij} minimized → let $p = \{i, j\}$

→ Interations :-

- while $|p| < n$

  - let s be the begining node of the path and l to be the end

  - choose the node $k \in V \setminus p$ to enter the path :

    - let k be such that C_{lk} or C_{ks} is minimized

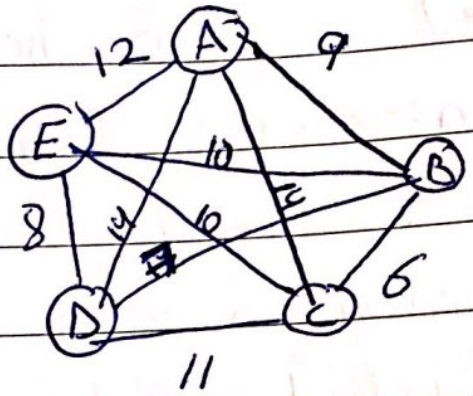
  - The new path is either $p + \{k\}$ or $\{k\} + p$

Mo Tu We Th Fr Sa Su

Memo No.

Date / /

EX8-



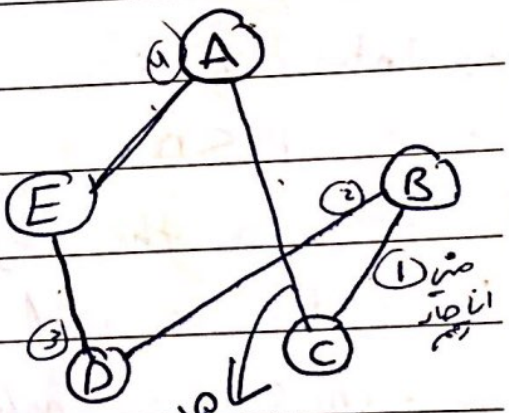
→ Initialization:-

① min c_{ij} ⇒ B-C P_s {B-C}

→ Iter 1

B: closest is D 7 ✓

C: closest is E 10



② P_s {D-B-C}

→ Iter 2

D: closest is E 8 ✓

C: closest is E 10

③ P_s {E-D-B-C}

Memo No.
Date / /

Mo Tu We Th Fr Sa Su

ما رحنا ع اللي من الوحيد اصلاً P

→ iter 3

E : closest is A 12 ✓

C : closest is A 14

(1) P = { A - ~~E~~ - D - B - C }

∴ T = { A - E - D - B - C - A }

بنوقف لما عدد ال nodes يساوي عدد ال arc

C = 47

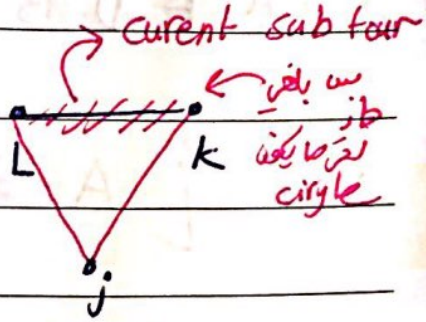
Mo Tu We Th Fr Sa Su

Memo No.
Date / /

② Nearest Insetion Heuristic :-

Insertions cost, def :- inserting node j between b/w nodes L & k where L & k are a pair of successive nodes in a tour has a cost = $c_{lj} + c_{jk} - c_{lk}$

⇒ Assumption: undirected network $G = (V, E)$ satisfying Δ-inequality



⇒ Initilization :-

Find arc (i,j) with min cost in E ⇒ T = {i,j}

⇒ Iterations :-

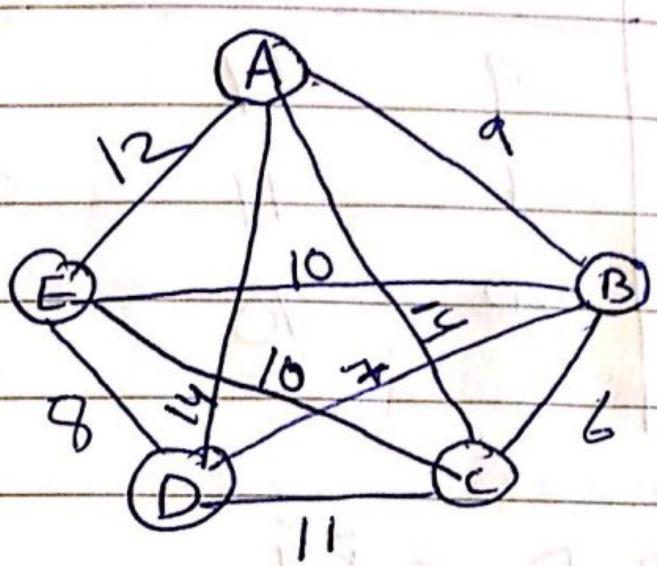
- while $|T| < n+1$
  - Find the node not in tour closest to a node in the tour
    - ⇒ let $c_j(T)$ be min c_{ij} for all nodes $j \notin T$, $i \in T$
  - Find insertion location in tour with min insertion cost
    - Find arc (L,k) in tour with min $c_{lj} + c_{jk} - c_{lk}$

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

- Insert J between i & k

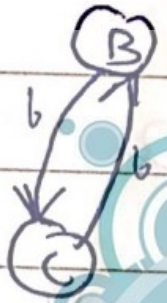
EX :-



(1) intial

Init. T = {B-C-B}

C(T) = 12



Mo Tu We Th Fr Sa Su

Memo No. _____

Date / /

② Iter 1

$T = \{B - C - B\}$

وين اضيف D بعدد هون

i∈T / j∉T	B	C
A	9	14
D	[7]	11
E	10	10

⇒ Cost of insertion

$C_{BD} + C_{DC} - C_{BC}$

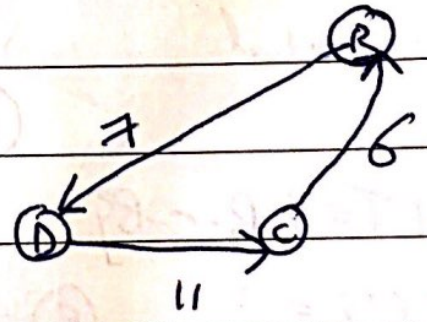
$= 7 + 11 - 6 = 12$

هون نقص الاخير
مارح تعرف وين

$T = \{B - D - C - B\}$

بهطريقة وف بيي قبل

$C(T) = 12 + 12 = 24$



③ Iter 2

i∈T / j∉T	B	C	D
A	9	14	14
E	10	10	[8]

$T = n+1$

بوقف

Memo No.

Date / /

Mo Tu We Th Fr Sa Su

هسا $T = \{B - D - C - B\}$

E ممكن ادخلها ثلاث خيارات

بعتنا وصلها //

Insertion:

$(B,D) = C_{BE} + C_{ED} - C_{BD} = 10 + 8 - 7 = 11$

$(D,C) = C_{DE} + C_{EC} - C_{DC} = 8 + 10 - 11 = \boxed{7}$

$(C,B) = C_{CE} + C_{EB} - C_{CB} = 10 + 10 - 6 = 14$

$T = \{B - D - E - C - B\}$

$c(T) = 24 + 7 = 31$

(4) iter 3 — ما في غير A الها

بحسب insertion.

→ $T = \{B - D - E - C - B\}$

Mo Tu We Th Fr Sa Su

Memo No. _____

Date / /

Insertion

$(B,D) = C_{BA} + C_{AD} - C_{BD} = 16$

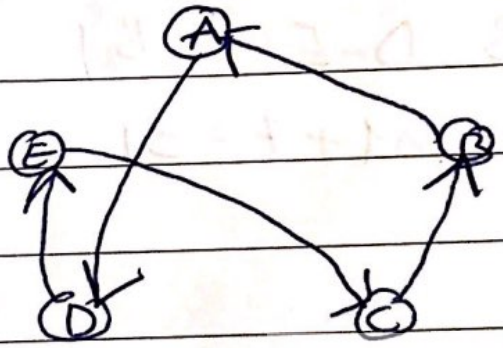
$(D,E) = C_{DA} + C_{AE} - C_{DE} = 18$ نختار أي واحد

$(E,C) = C_{EA} + C_{AC} - C_{EC} = 16$ كلن

$(C,B) = C_{CA} + C_{AB} - C_{CB} = 17$

$T = \{B-A-D-E-C-B\}$

$C(T) = 16 + 31 = 47$



Mo Tu We Th Fr Sa Su

Memo No.
Date / /

3 Farthest Insertion Heuristic

Assumption: undirect

⇒ Initialization:

Find arc (i, j) with Maximum cost in E

T = {i, j, i} "Max Cij"

⇒ Iterations:

Identical to nearest insertion, except when choosing the node to enter the tour in each iteration, use the following criteria

- For every node j not in the tour, Find minimum distance edge (j, i) connecting node j to a node in the tour

that is:

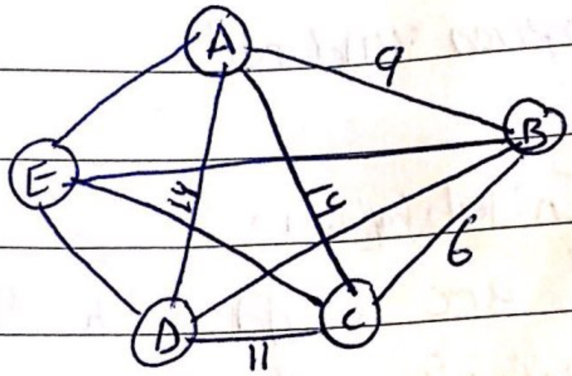
let Cj(T) be min Cij for j ∉ T, i ∈ T

Pick node j for which Cj(T) is Maximized

→ The node to insert is the node with maximum Cj(T)

Memo No. ______
Date / /

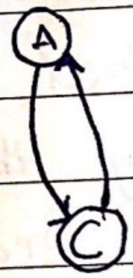
Ex :-



. initial :-

T = { A - C - A } بنأخذ اطول واحد

C(T) = 28



- Iter. 1 :-

j∉T \ i∈T	A	C
B	9	6
D	14	11
E	12	10

بأخذ أقل مسافة بعدين أكبر وحدة

بين 9 و6 مين اقرب لـ T الجديد 6 ...

ما بختار 11 الأكبر

find min in each row then max

insertion

$(A, C) \rightarrow C_{AD} + C_{DC} - C_{AC} = 11$

T = { A - D - C - A }

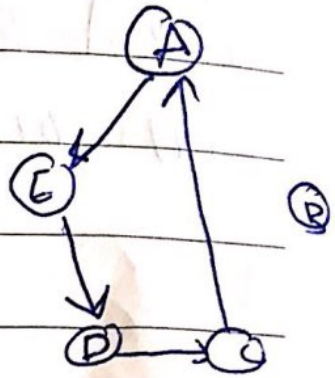
$C(T) = 11 + 28 = 39$

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

Iter 2:

i / j	A		
B	9	6	7
E	12	10	8



insertion

$(A,D) \rightarrow C_{AE} + C_{ED} - C_{AD} = 12 + 8 - 14 = 6$

$(D,C) \rightarrow C_{DE} + C_{EC} - C_{DC} = 10 + 8 - 11 = 7$

$(C,A) \rightarrow C_{CE} + C_{EA} - C_{CA} = 8 + 12 - 14 = 8$

$T = \{A, E, D, C, A\}$

$C(T) = 39 + 6 = 45$

iter 3:- B ka h/o

insertion

$(A-E) \rightarrow C_{AB} + C_{EB} - C_{AE} =$

$(E,D) \rightarrow C_{EB} + C_{BD} - C_{ED}$ s

$(D-C) \rightarrow C_{DB} + C_{BC} - C_{DC}$ s

$(C-A) \rightarrow C_{CB} + C_{BA} - C_{CA}$ s

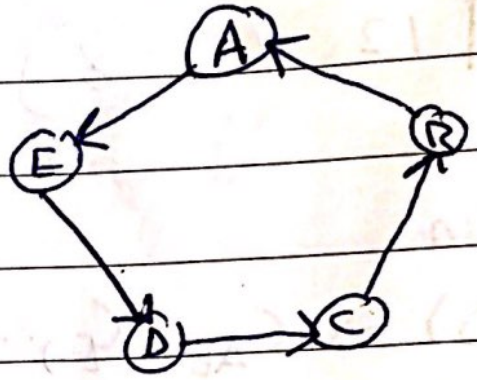
Mo Tu We Th Fr Sa Su

Memo No. _____

Date / /

Ts = { A - E - D - C - B - A }

C(T) = 45 + 154 = 46



(4)

✗ cheapest insation Heuristic :-

Assumphous: undirated complete graph, satistaying D-ineq

intioal:- Find Min arc (i,j) with min cost $C_{ij} \Rightarrow T = \{i,j,i\}$

Itations:-

While $|T| < n+1$

- Find j not in tour $\Rightarrow$ Find insertion cost for all possible insertion locations
- Pick min insartion cost

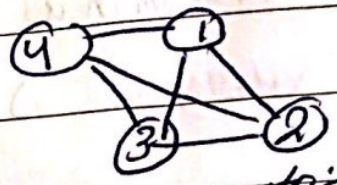
insertion location / Ket	(B-C)	(C-D)	(D-B)
A	$C_{BA}+C_{AC}-C_{BC}$	$C_{CA}+C_{AD}-C_{CD}$	$C_{DA}+C_{AB}-C_{DB}$
E	$C_{BE}+C_{EC}-C_{BC}$	$C_{CE}+C$	

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

* Mathmatical formulation of TSP :-

The symmetric TSP is for mulatuli



cij cost ← symmetric & complet
cji cost

note that $i \leq j$ $(i,j) \rightarrow$

$E = \{(1,2)(1,3)(1,4)$
$(1,5)(2,3),(2,4),$
$(2,5),(3,4)(3,5)$
$(4,5)\}$

variable :-

binary

① $X_{ij} = \begin{cases} 1 & \text{if } (i,j) \text{ is include in toure} \\ 0 & \text{o.w.} \end{cases}$

② c_{ij}

① $\min \sum_i \sum_j c_{ij} X_{ij} = \min \sum_{(i,j) \in E} c_{ij} X_{ij}$

② s.t $\sum_{i \in V : (i,j) \in E} X_{ij} + \sum_{i \in V : (j,i) \in E} X_{ji} = 2 \quad \forall j \in V$ — ②

Memo No. ________
Date / /

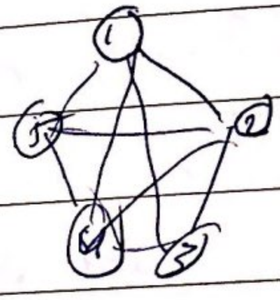
Mo Tu We Th Fr Sa Su

(3) $\sum_{(i,j)\in E:\, i\in S,\, j\notin S} x_{ij} + \sum_{(j,i)\in E:\, i\in S,\, j\notin S} x_{ji} \geq 2 \quad \forall S \subset V \;\&\; 2 \leq |S| \leq \frac{|V|}{2}$

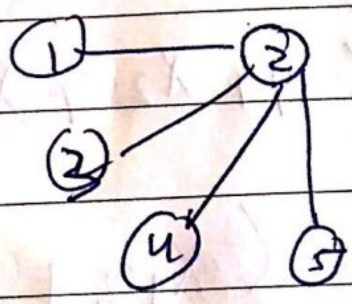
$x_{ij} \in \{0, 1\}$

- objective (1): minimize total cost of included arcs
- constraint set (2): exactly 2 edges included to each vertex

لازم اذا كل وصار 1 رح تبطل cycle
واذا صار 3 صارت اكثر من one cycle



for j = 2



For j = 2

$(x_{12}) + (x_{23} + x_{24} + x_{25}) = 2$

$\sum x_{ij}$ $\qquad \sum x_{ji} \rightarrow i > j$

$i < j$

فقط دون تبين 2 بس

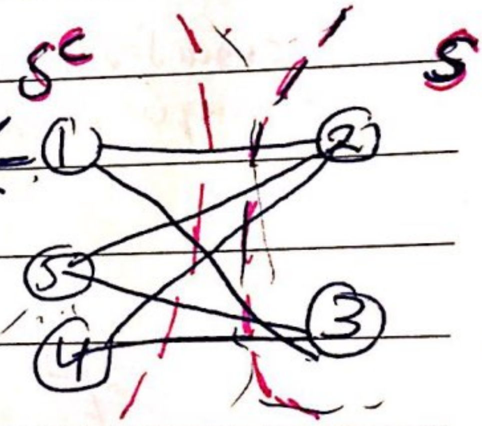
صار 2 ... 2 او

complet تكون graph لازم تكون ال
symetric
indirected
~~sets~~ traing Δ-inequalty

لو كانت $j = 1$ يكون $\Sum X_{ji}$ في
→ $X_{ij} = 0$ هاد معناه انه اي

- Constrain set (3): Connectivity constraints
  → we should have one constraint for each possible subset of size 2 or 3

For $S = \{2,3\}$ $V \setminus S = \{1,4,5\}$



$$(X_{24} + X_{25} + X_{34} + X_{35}) + (X_{12} + X_{13}) \geq 2$$

لازم يطلع اثنين ما يصير sub-Tour

Ex: $S = \{1,2,3\}$, $S^c = \{4,5\}$

$$(X_{14} + X_{15} + X_{24} + X_{25} + X_{34} + X_{35}) + (0) \geq 2$$

$\lceil 2.4 \rceil \to 2$ $\lceil 2.9 \rceil \to 3$

$\lceil 2.5 \rceil \to 3$ Memo No. $\lceil 2.3 \rceil \to 3$

$\lfloor 2.5 \rfloor \to 2$ Date / /

Mo Tu We Th Fr Sa Su

• constraint set (3). continued :-

it is only necessary to formulate a constraint of type (3) for subsets of size $2 \le |s| \le \lceil \frac{|V|}{2} \rceil$

↳ why not subsets of size 1?

constraint (2) takes care of it

→ for subsets of size $> \lceil \frac{|V|}{2} \rceil$

the subsets are complements to some subsets of size $< \lceil \frac{|V|}{2} \rceil$

↳ Why not subset of size $|V|$? it is what we looking for

(4) constraint set (4) ; Introduce here :-

↳ is an alternative (N.) for constraint set (3)

↳ would replace constraint set (3)

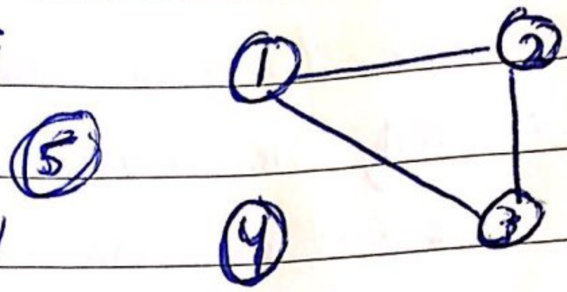
$$\sum_{(i,j) \in E : i \in S, j \in S} x_{ij} \le |s| - 1 \qquad \forall s \subset V \quad 2 \le s \le \lceil \frac{|V|}{2} \rceil$$

Mo | Tu | We | Th | Fr | Sa | Su

Memo No. ______
Date / /

EX :- For $S = \{1, 2, 3\}$

arc



$x_{12} + x_{13} + x_{23} \leq |S| - 1$

$3 - 1$

$x_{12} + x_{13} + x_{23} \leq 2$

⇒ sub tour elimination

هاذ ضمن
Constraint Set(4)

✳ Vehicle Routing problem "VRP"

TSP: all customers request are several by a single vehiche "tour"

→ In VRP, m vehicles are based at a depot, and the problem is to find the set of m least-cost tours, based at the depot serving all (or subset of) customer request

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

* Common operational Constraints in "VRP"
  - Number of vehicles m
    - ↳ can be Fixed or Variable
  - capacitated vehicles
    - ↳ capacity Q # of, weight of, volume of that can fit in a tracks
  - work shift duration.
    - ↳ capacity constraind
  - Time windows of customers

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

* The classic VRP:-

suppose whomogeneous fleat of vehicles based in a deput each with capacity Q

They have to serve

* each of n customers demands. customer demand q_i, $i = 1, \ldots, n$ * demand cannot be split ← يعني طلب واحد لازم يوصل بنفس سيارة ما ينقسم

* Travel cost between locations i & j is C_{ij}

The VRP is to find a set of vehicles tours, each begining and ending at the depot such that each customers demand is served, no vehicle violates capacity constraint, and total cost is minimized

↳ "total travel cost"

* which problem that we have studied is a special case of the VRP?

~~TSP ⇒ m = 1 , Q ≥ ∞~~

Mo Tu We Th Fr Sa Su

Memo No.
Date

* Another problem with important applications in transportion & logistics is a special case of VRP

↓

Bin Packing problem

⇒ Bin packing problem : assumes $c_{ij} = 0$ for all (i, j) + TSP = VRP
capacity constraint remains

↓

BPP + TSP = VRP

$c_{ij} \geq 0$ $q_i \leq 0$

BPP:

- let m be a number of items to be loaded
- & n be the number of bins
- let p_i is $1, \ldots, m$ be the weight of item i

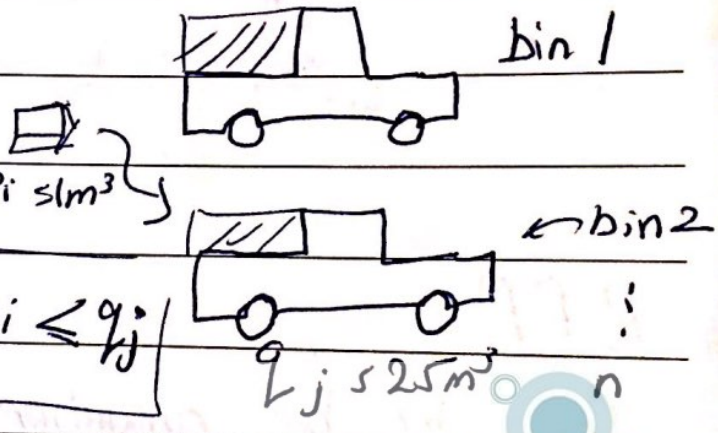
Mo Tu We Th Fr Sa Su

Memo No.
Date / /

let q_j $j = 1, \dots n$ be the capacity of bin j

Item cannot be split between bins

* The problem is to assign each item to a bin such that
  - total weight of items in bin $j \leq q_j$



$$\sum_{i=1}^{m} P_i \leq q_j$$

  - smallest # of bins used

Memo No.
Date / /

* mathmatical formulation of BPP :-

- let $Y_j = \begin{cases} 1 & \text{if bin } j \text{ is used} \\ 0 & \text{o.w} \end{cases}$ اذا بنستخدمها

- let $X_{ij} = \begin{cases} 1 & \text{if item is assigned to bin } j \\ 0 & \text{o.w} \end{cases}$

- $\min \sum_{j=1}^{n} Y_j$ --- [1]

- s.t $\sum_{j=1}^{n} X_{ij} = 1 \quad \forall i = 1, \dots, m$ --- [2]

هاداني ال item لازم يكون assigned
ل bin واحد $\overset{i=1}{\longleftarrow} X_{11} + X_{12} + \dots = 1$ يعني وحدة منهم بس ادتك 1

- $\sum_{i=1}^{m} P_i X_{ij} \leq q_j \quad \forall j = 1, \dots, n$ --- [3]

$P_1 * X_{11} + P_2 * X_{21}$ [3]
$+ P_3 X_{31} \leq q_1$

هاي ضرب لانه برا لحاره ١

Memo No. ______
Date / /

(4)

- $X_{ij} \leq Y_j \quad \forall i = 1, \dots, m$

$j = 1, \dots, n$

$X_{ij}, Y_j \Rightarrow \{0, 1\}$

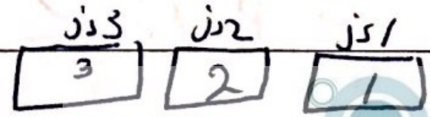
الـ هاي انا ما بقدر اعمل item assign
إذا نفس الـ bin مش موجود او مش فاتح

* objective(1) :- minimize # of bins

- Constraint set (2) : Each item is assigned to exactly one bin ①

هنا ممكن يكون الـ item مقسم لأكثر
من مكان "bin" يعني مثلاً four meter
ممكن اخلي bin ينقسم نصفه و bin الثاني ربعه
ويكون هنا $0 < X_{ij} < 1$

- constraint set (3) : bin capacity is not violated ②
- constraint set (4) : we can't assign an item to a bin unless bin is opened



هاي بس اخذها فيها مثل معناها $y_1 = 1$, $y_2 = 0$, $y_3 = 0$

* lower bounds for BPO :-

- Assume $q = Q$ for all $j = 1, \dots, n$
- lower bound on the # of bins required

$LB = \left\lceil \frac{\sum p_i}{Q} \right\rceil$ ← هاد اقوى حد من الـ bin نحتاجها بـ اقل ~~[crossed out]~~ حالة بشكل عام

$Q = 50$ $\quad LB = \frac{200}{50} = 4$ ← هاد الحكي مش صحيح ليش لانه انا بفرض الحمولة تكون صافية ١٠٠٪ بحتاج ممكن يكون للحل bin ناقصة ٤ كغم وانا الـ 4 وزنهم كان 4 كغم صح ما بقدر احطها ...

$\sum p_i = 200$

if $q_j \neq Q$ → largest capacity r (→ max cap.)

(q_j → اذا كان bin مختلفة)

$LB = \left\lceil \frac{\sum p_i}{r} \right\rceil$

* Heuristic solution approaches:

- online problem — مش معروف الـ advice / date بيجي — الحل بالحل
  - → next Fit (NF) , 2-opt.
  - → First Fit (FF) , 1.7-optimal

Memo No.
Date / /

→ Best Fit (BF) ; 1.7 - optimal

• offline problem ال date تيجي بالترتيب لعمر كلها شغالة

→ next Fit Decresing (NFD)
→ First Fit Decresing (FFD); 1.5 - optimal
→ Best Fit Decresing (BFD) ; 1.5 optimal

⇒ offline : all item sizes are known in advance
⇒ on line : In put date is revealed over time
→ Data is sorted in adecresing order
4, 9, 5, 7, 10, 3, 2, 11
11, 10, 9, 7, 5, 4, 3, 2
* In each of the heuristics let L be the list of item weights L = { P1, P2, ..., Pm }
→ All example we will assume the
L = { 16, 10, 14, 12, 4, 8, 3, 12, 7, 14, 8, 6, 3 }
9 = 9 = 20 (في كل bin)
20

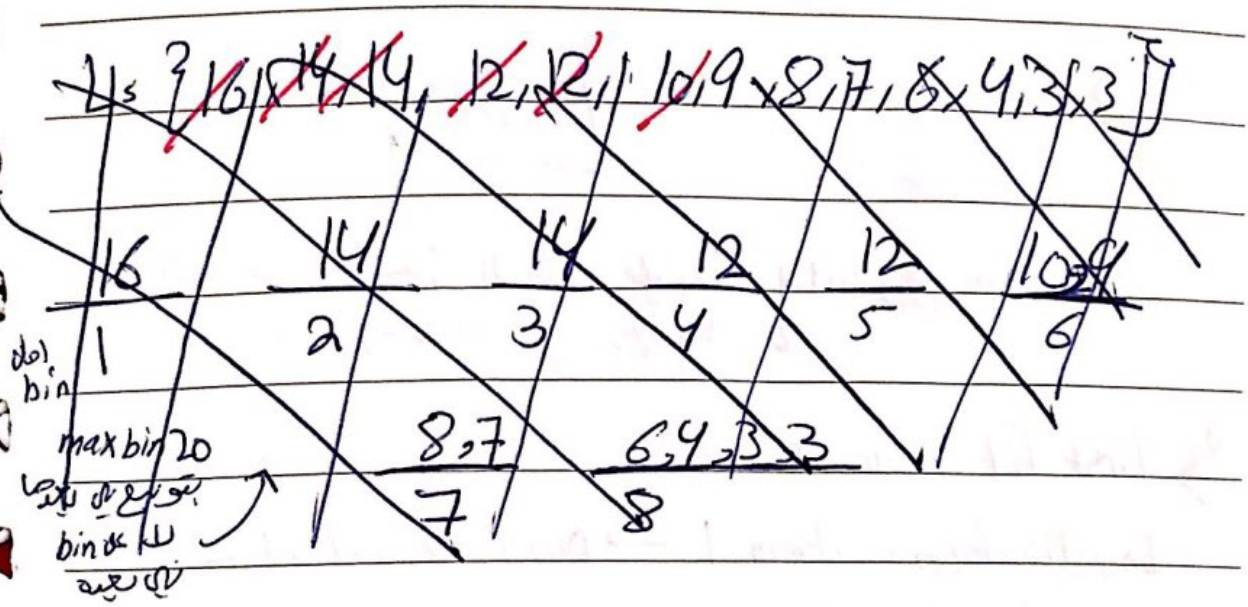
Mo Tu We Th Fr Sa Su

Memo No.
Date

$LB = \lceil \frac{\sum P_i}{20} \rceil = \lceil \frac{118}{20} \rceil \leq 6$ bin ← الحل الأمثل

① next Fit (NF) "online problem"

Advantage :- allows immediate dispatch of bin j once bin j+1 is opend



* Initialization : place item 1 in bin 1 and remove from L, let j=1 . i=2

- Iterations:

• If item i fits in bin j, place i in j, other wise open anew bin j+1 and place i in j+1

Mo Tu We Th Fr Sa Su | Memo No. ____ Date / /

- Remove i from L → let i = i+1
- While items remains in L, Repeat

Solution trades on very صاحبها مابرتبه
بيغير هو line

16	10	14	12,4	8,3	12,7	14
1	2	3	4	5	6	7

9,6,3 / 8

8 bins

هسا هون انا بمشي ال order يلي بعده اذا ما بوسع بطلع
بن ممكن يصير يحدد انه بيمشي اربع ساعات بس

② → First Fit heurstic

Initilization: item 1 → bin 1 → j = 1, i = 2

Iteration: place item i in the lowest numbered bin for which it fits, IF iteam i does not fit in any open bin, open bin j+1 & place i

- Remove i from L → i = i+1
- while item remain L Repeat

Mo Tu We Th Fr Sa Su

Memo No.
Date

solution :-

16,4 / 1 | 10,8 / 2 | 14,3,3 / 3

12,7 / 4 | 12,6 / 5 | 14 / 6 | 9 / 7

7 bins

(3) → Best Fit heuristic

Initialization : item 1 → bin 1 → Remove 1 from L ⇒ j=1 i=2

Iterations: Find the bin j whose content (total weight of items) are maximum but not greater than Q - w_i ← i fits in the bin.

If item i does not fit in any open bin open anew bin ↷ j+1 ← i

. Remove i from L . i = i+1

. while item in L, repeat

Mo Tu We Th Fr Sa Su

Memo No.
Date / /

L = {16, 10, 14, 12, 4, 8, 3, 3, 12, 7, 14, 9, 6, 3}

Solution:

بحط ال 4 كمان — max wight

16, 4 — 1
10, 9 — 2
14, 3, 3 — 3
12, 8 — 4

12, 7 — 5
14, 6 — 6

6 bin

هون انا بشوف مين تاخد ال wight
اكبر مثال بوسع انا او إله انا
لأ من شان أكبر wight ...

شروط ممكن تغير
الحل ممكن تضغط

→ if bin j is full تطلع
→ if specific bine has been passed 1/80 Q = 20
16 لا يوصل 16
ممكن يطلع

* practice:
NFD → 8 bin
FFD → 6 bin
BFD → 6 bins
} كل offline

لازم تحكم بين بنرتيهم تنازلي

L = {16, 14, 14, 12, 12, 10, 9, 8, 7, 6, 4, 3, 3}
هاي مرتبة

Mo Tu We Th Fr Sa Su

Date / Page Fourti

* Recall in the VRP :-

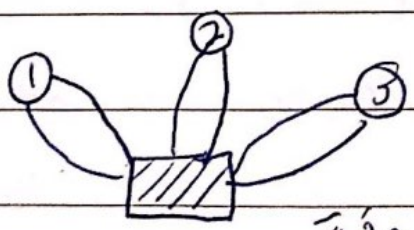
- Fleet of homogeneous vehicles, each with capacity Q, based at depot
- n customer with demand q_i, $i = 1, \dots, n$ $\boxed{\Sigma q \leq Q}$
- Find least cost set of vehicle tours

→ Heuristics for VRP

1. Clark-wright savings heuristics
2. Route-first cluster-second heurstic (RFCS)
3. cluster-first Router-secound heurstic (CFRS)

* Clark-wright savings heuristics :-

Begins with a seperate tour for each customer
Then combined tours, if feasble with respeact to demand and if saving result



Mo Tu We Th Fr Sa Su

Memo No. / /
Date

let T_i and T_j be two tours, and let end(i) denotes the end of tour i and begin(j) dentes the beging of tour j

savings :- $S_{ij} = C_{end(i),0} + C_{0,begin(j)} - C_{end(i),begin(j)}$

1. For each node $i \in N$, construct a tour T_i beginning at the depot, traveling to node i and returing to the depot S set at tours.
2. calculate the savings S_{ij} for each pair of nodes i, j -> $S_{ij} = C_{oj} + C_{jo} - C_{ij}$
3. Sort the savings in non-increasing (Decreasing) order
4. walk down the saving combination list. and Find the next feasible pair (i, j) such that :

A. i and j are in different tours: (T_k and i is the last node on its tours) T_k and j is the first node on its tour T_l

B. $S_{ij} \geq 0$

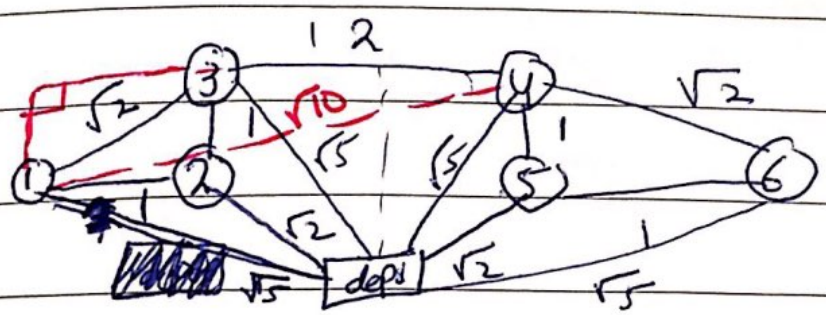
Mo Tu We Th Fr Sa Su

Memo No.
Date / /

c. $\sum_{k \in T_j} q_k \leq Q$ demand كل الموجودين في الرحلة T

EX:

$C_{14} = \sqrt{3^2+1^2}$



Vechicle capacity is 5, demand at customer 1,2,3 is 3
demad at custmer 4,5,6 is 2

$$LB = \left\lceil \frac{\sum q}{Q} \right\rceil = \frac{15}{5} = 3$$

$T_1 = \{0-1-0\}$ $T_2 = \{0-2-0\}$ $T_3 = \{0-3-0\}$
$T_4 = \{0-4-0\}$ $T_5 = \{0-5-0\}$ $T_6 = \{0-6-0\}$

$\delta_{12} = \sqrt{5} + \sqrt{2} - 1 = 3.65$
$\delta_{13} = \sqrt{5} + \sqrt{5} - \sqrt{2} = 3.06$
$\delta_{14} = \sqrt{5} + \sqrt{5} - \sqrt{10} = 1.31$

$\delta_{15} = \sqrt{5} + \sqrt{2} - 3 = 0.65$

$\delta_{16} = \sqrt{5} + \sqrt{5} - 4 =$

$\delta_{46} = 2 \quad = 3.06$

Q بحيث يختار أكبر δ ويتأكد من

$\delta_{13} = 3+3 > 5 \quad X$

$\delta_{46} = 2+2 \leq 5 \quad \checkmark$

$\Rightarrow \delta = T_1, T_2, T_3, T_{46}, T_5$

ويحذف $\delta = T_1, T_2, T_{35}, T_{46}$