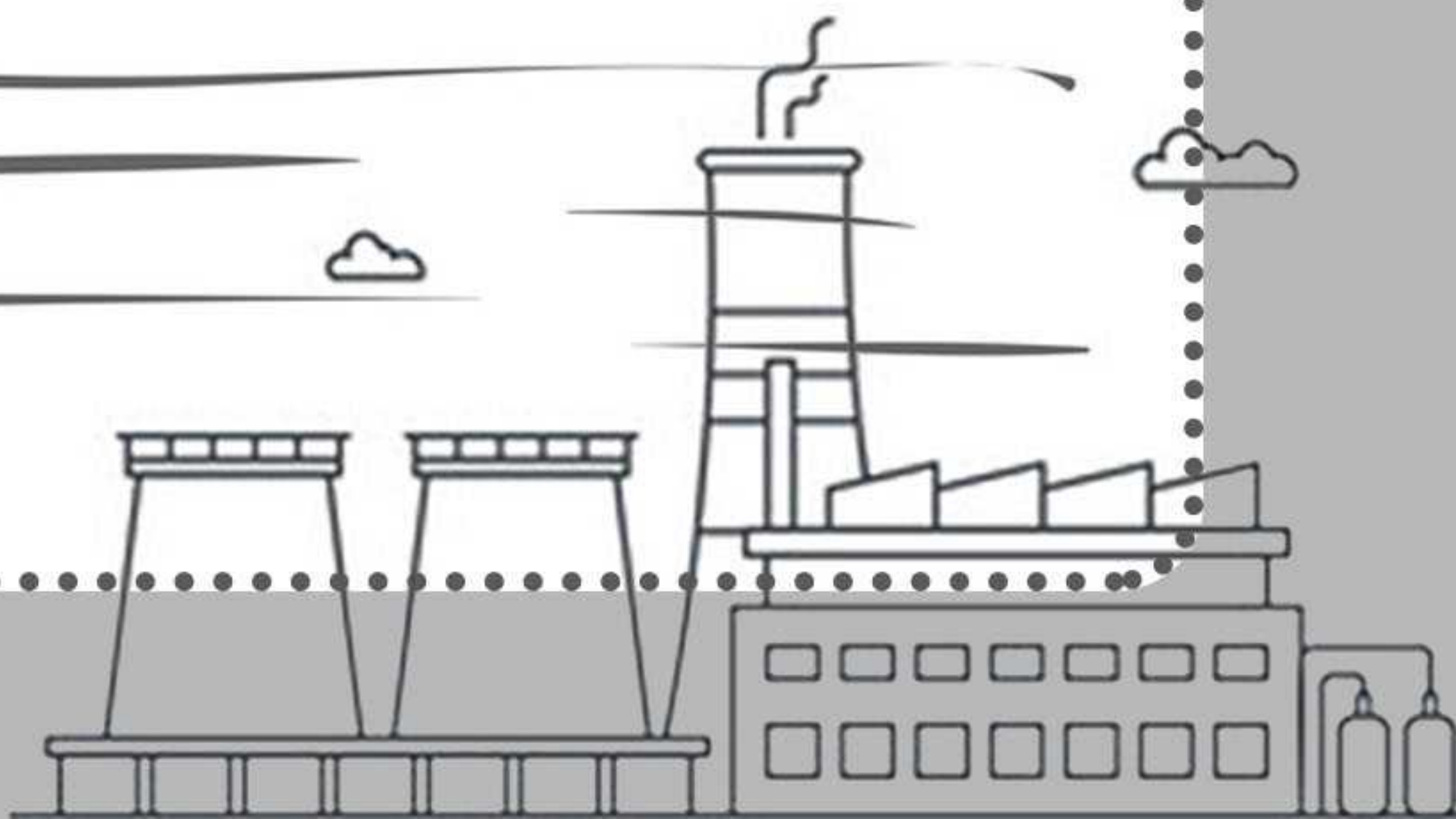


WEEKLY NOTEBOOK



Logistics

second semester
2021



logistics :-

supply chain : big umbrella

logistics : logistics is subsystem

Retailer البائع

Vendor المورد البائع

انتبه المصطلح

Assorted supply chain

- Retailer - merged التجار نفسه براقب المخزون
- Vendor المورد نفسه يكون له حاله مخزون في المحل براقبوا المخزون

which type of supply chain requires the least amount of inventory to be held?

- Make to assembly نفسه مصنع وفعلا
- Make to stock مصنع يسبقه براك فاعطاك
- Make to order ما ينتج الا لما يطلب

what are the main activity for logistics

- order processing
- Inventory Management
- Freight transportation

Work in process $\Rightarrow$ inside of facility

service level : نسبة الطلب التي تلبيها للزبائن بدون
ما يطلبوا أو يضايقوا -- الخ

So we need inventory

Cope with randomness : if they have variable
(demand, lead time) So we need more inventory

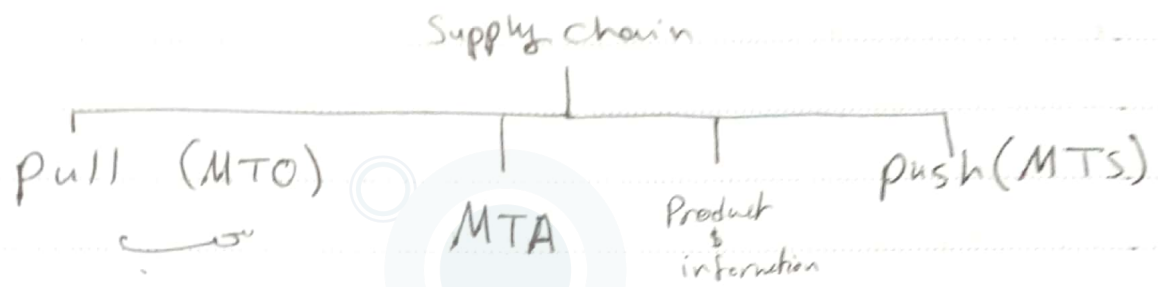
يعني يعني بسالة إذا زادت في Variable في demand في أي مكان
مساحة تخزين
وإذا زاد Variable في lead time في أي مكان
كأن ما يصير نقص

effective : مدى فاعلية الامر الانطباق في

efficient : مستوى الموارد الموجودة في بأفضل طريقة

- What is the relationship between supply chain
and logistic sys.?

logistic sys. is subsystem from supply chain



- انت بيتسأل الطلب -

على الاطارات

- lead times short اقل
- expensive اقل
- demand highly variable
- * trend

改善

KAIZEN

TEAM

Cost in logistics sys

For essential types of cost:-

1. transportation cost
2. Handling cost
3. Holding Cost (cost of capital / opportunity)
4. Facility rent cost

normalizing costs

- Cost per time period
- Cost // item

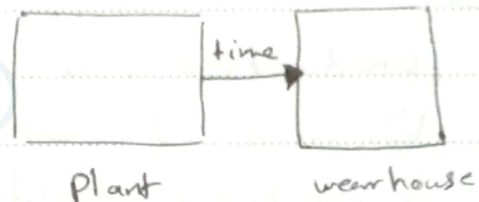
* Comparing the cost of carriers or modes the objective minimize tot. cost when selecting mode of transportation

⇒ often requires trading off inventories & transportation costs

Inventory cost — holding cost / cost of capital
— rent cost for space

Inventory held in 3 places

- at plant
- at warehouse
- in transit



Problem 1 :-

A company need to select between two transportation modes rail and truck, to move goods from plant to wh. with obj: minimizing tot. transportation and inventory costs on an annual bases

- Data

	<u>Plant</u>	<u>WH</u>
safety stock	0	100 units
Value of items	\$500 /unit	\$512 /units
spaces cost	\$ 20/unit year	\$ 25 /units

- Rail

- Car Capacity = 400 units
- Transit time = 20 days
- Cost per shipment = \$ 1200

Truck

Capacity = 100 units

transit time = 3 days

Cost per shipment = \$700

~~Set:~~

Production rate at plant 10 units/day

Demand of WH. 10 unit/day

Inventory holding cost rate 25% of value/year held

⇒ Goal: Determine the optimal shipment size q (# of units) for each mode, and the tot. annual cost associated with these shipment sizes

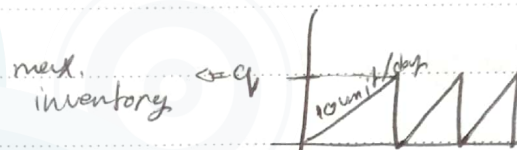
⇒ Assumption: shipper will never ship more than a single (rail) car load or single truckload (truck)

Steps

- ① annual inventory space cost at the plant
- ② " " holding " " "
- ③ " " space " " WH
- ④ " " holding " " "
- ⑤ Determine in-transit inventory cost

- ⑥ Determine transportation cost
- ⑦ tot. annual cost expression
- ⑧ Determine optimal q for each mode
- ⑨ choose a mode

Solution :-



① space cost at plant

$$\text{space cost} = \frac{\$20}{\text{unit} \cdot \text{year}} * q \text{ units} = \$20q / \text{year}$$

② Holding cost at plant

$$\begin{aligned} \text{rate of change per item of inventory} &= \$500 * 25\% \\ &= 125 \$ / \text{unit} \cdot \text{year} \end{aligned}$$

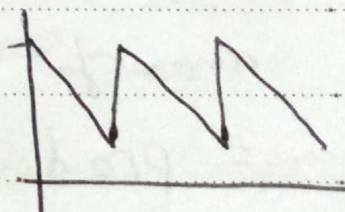
$$\text{Avg inventory level} = \frac{q}{2}$$

$$= 125 * \frac{q}{2} = 62.5q \$ / \text{year}$$

③ space cost at WH

$$\text{next inventory level} = q + 100$$

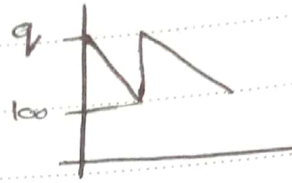
$$\text{space cost} = (25)(q + 100)$$



Holding cost at WH

Avg inventory level

$$= 100 + \frac{q}{2}$$



each item hold on yr charged at ~~regular~~

$$512 * 25\% = \$128 \text{ unit/year}$$

$$\text{Holding Cost} = \frac{\$128}{\text{unit}} * (100 + \frac{q}{2}) \text{ unit}$$

⑤ Determinant in transit inventory cost

$$\text{use } \frac{506\$}{\text{unit}} = \frac{500 + 512}{2}$$



$$\text{each item charge at } \frac{506\$}{\text{unit}} * 25\% \text{ year} = 126.5$$

Rail

each item space 20 days

amount to be shipped the whole yr.
= production rate

$$= \frac{10 \text{ unit}}{\text{day}} * 365 = 3650$$

$$\text{Inventory cost} = \frac{365 \text{ units}}{\text{yr}} \cdot 20 \text{ day} \cdot \frac{1 \text{ yr}}{365 \text{ day}} \cdot \$126.5 \frac{\text{unit}}{\text{yr}}$$

$$= 25300$$

Truck:

$$\text{In transit cost of truck} = \left(\frac{3650 \text{ units}}{\text{year}} \right) \cdot \left(3 \text{ days} \cdot \frac{1 \text{ yr}}{365 \text{ day}} \right) \cdot (126 \$ / \text{unit} \cdot \text{yr})$$

$$= 3,795 \$ / \text{yr}$$

6) Transportation Costs

Rail: every shipment cost \$1200

$$\# \text{ of shipments} = \frac{3650 \text{ shipment/yr}}{q} \quad \frac{\text{كم شحنة سنوية}}{\text{كم شحنة واحدة}}$$

$$\text{Transportation Cost Rail} = \frac{\$1200}{\text{shipment}} \cdot \frac{3650 \text{ shipment}}{q \text{ yr}}$$

$$= \frac{4380000}{q} \$ / \text{yr}$$

$$\Rightarrow \text{Transportation Cost Truck} = \frac{\$700}{\text{shipment}} \cdot \frac{3650 \text{ shipment}}{q \text{ yr}}$$

$$= \frac{2555000}{q} \$ / \text{yr}$$

⑦ Total annual Cost

$$TCR_{\text{Rail}}(q) = 20q + 62.5q + 25q + 2500 + 64q + 12800 + 25300 + \frac{4380000}{q}$$

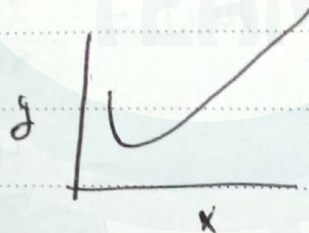
$$TCR_{\text{Rail}}(q) = 171.5q + 40600 + \frac{4380000}{q}$$

Inventory Cost Transport Cost

$$TCR_{\text{Truck}} = 20q + 62.5q + 25q + 2500 + 64q + 12800 + 3795 + \frac{2555000}{q}$$

$$TCR_{\text{Truck}} = 171.5q + 19095 + \frac{2555000}{q}$$

q	TCT	TCR
10	bs. Function	bs. Function
20		
30		



⑧ optimal q for each mode

$$TCR'(q) = 171.5 - \frac{4380000}{q^2} = 0$$

$$q_R = 160 \text{ units}$$

$$TCR(q_R = 160) = \$95415/\text{yr}$$

Is not possible $q_T = 122$ in one truck

$$TRT'(q) = 171.5q - \frac{2555000}{q^2} = 0$$

$$q_T = 122 \text{ units (هذه بـ 2 truck حتى انحدار)}$$

لا في الحقيقة حسب الجدول

$$\text{shipment cost} = 2 \times \frac{700}{\text{shipment}} \times \frac{3650}{q}$$

$$= \frac{5110000}{q}$$

$$TCT_{q=122} = 171.5 \times 122 + 19095 \neq \frac{5110000}{122} = \$81,903$$

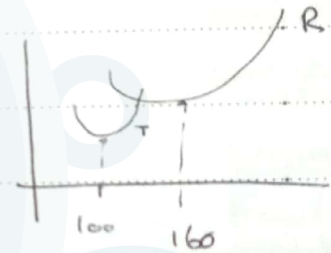
→ If the maximum truck Capacity was selected $q_T = 100$

so

$$TCT|_{q_T=100} = \$61795$$

تخمين إجمالي
Total Cost

$$(TCT|_{q_T=99}) = \$61881.58$$



Shipment size	Inventory Cost	Transportation cost	Total cost
160 rail	68,040	27,375	95,415
400	↑ 109,200	↓ 10,950	120,150
140	↓ 64,610	↑ 31,286	95,896

If $R \Rightarrow q = 160$ Cost = 95000

$T \Rightarrow q^* = 122$ Cost = 97000
2 shipment

$T \Rightarrow q = 100$ Cost = 61790
أقل
م
واظركم

改善

KAIZEN

TEAM

Transportation Rate and Cost by mode

Fixed on: infrastructure, Right of way
vehicles, equipment

variable cost (operating): fuel, maintenance
drive

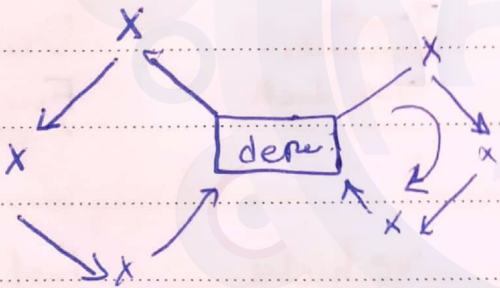
Mode	Fixed	Variable	Fixed Cost Component	variable Cost Component
water	M-H	L	vehicles	low speed high capacity fuel efficient
Rail	H	L	rail roads	Fuel eff. small crew
Truck	L	H	Trucks terminals cross dock	Drivers Fuel Maintenance
Air	H	M-H	vehicles terminals Facilities	Fuel Low capacity

Short-haul

short-haul transportation: pickup/delivery of goods in relatively small area using fleet of ~~vehicles~~ vehicles.

unless stated otherwise vehicles are stationed at single depot

- Route include several pickup and delivery point
- Routes performed in a single work shift



Decision in short haul transportation

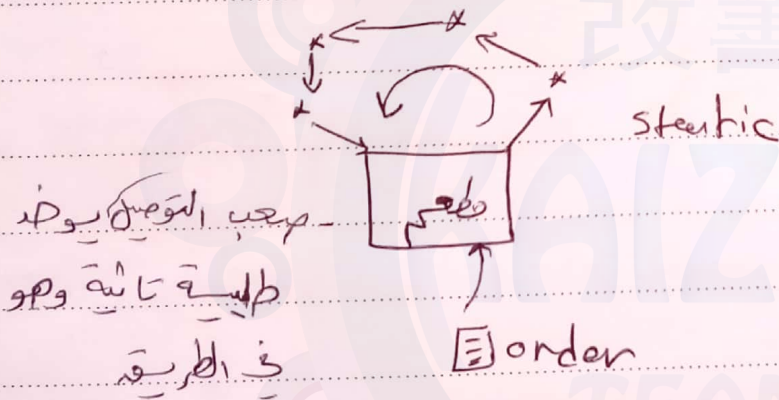
- strategic: where to locate depot/warehouse "terminal"
- tactical: fleet sizing (عدد) How many truck to have
- الترتيب operational: vehicle routing and scheduling problem
VRSP

- Important what data known before ?

↳ Static vehicle Route problem: Customer requests (location, amount of goods) is known in advance. ما في تغيير ووقت الزبون ثابتة

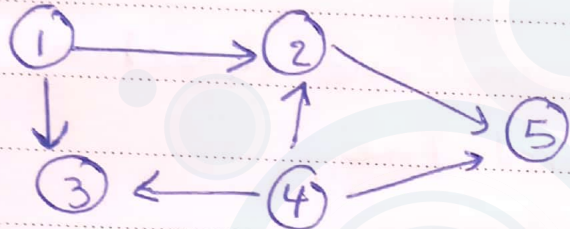
↳ Dynamic VRP: not an information is known in advance. ^{الوقت} _{مختلفة}

Routes Planned for set of requests known in advance, dynamic updated to incorporate new requests.



↳ order of a graph : $\#$ of vertices $|V|$ or nodes

↳ size of graph : $\#$ of edges $|E|$



$$G = (V, E)$$

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

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

order of $G = 5$ $|V|$

size of $G = 6$ $|E|$

structural properties of Graphs

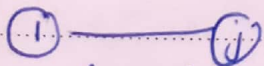
- undirected vs directed



network without
specified direction
- a connection between

i & j implies a
connection between i/j

symmetric
asymmetric

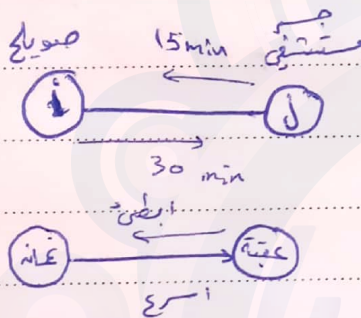


also referred to symmetric network

in transportation problem, sometime symmetry used to represent whether cost is same when transportation from i to j as when traveling from j to i

a symmetry example :-

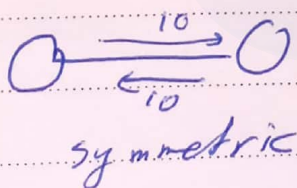
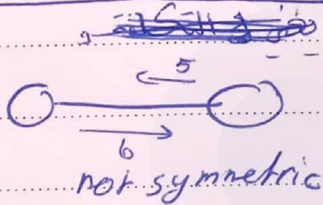
traffic may be higher in one direction resulting in higher travel time



→ symmetric or a symmetric
→ bidirectional

Cost = time

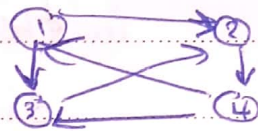
التي تكون الطريقة ذاتها في كلا الاتجاهين
symmetric cost



Completeness: a graph is complete if an edge exists between every ~~parties~~ pair of vertices (regardless of direction)



Completeness
undirected



↳ Path, Cycles, Trees

For the following definitions, assume there are (n) vertices in V , and (m) edges in E

Path: a path P in G is a sequence of vertices $\{i_1, i_2, i_3, \dots, i_n\}$

- $i_k \in V \quad \forall k = 1, \dots, p$
- i_k unique $\forall k = 1, \dots, p \Rightarrow$ no vertex visited more than once
- (i_k, i_{k+1}) or $(i_{k+1}, i_k) \in E$

↑
مسار
مسار
Path

$\{1, 4, 5\} \times$

$\{1, 3, 5\} \times$



~~paths {1, 2, 5}~~

Paths from 1-5

$\{1, 2, 5\}$ feasible ✓

$\{1, 2, 4, 5\}$ infeasible ✓

Directed Path

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

is a feasible
is a decision for

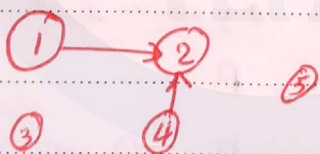
Path feasible in the previous example $\{1, 2, 5\}$ is directed path

Related concept: Connectedness

connected: two vertices $i \neq j$ are connected in G if there exists a path from i to j

regardless direct

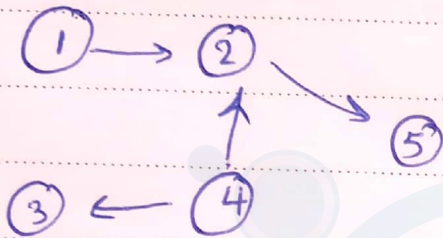
Adjacency: Connected ~~with~~ by path of ~~least~~ length



nodes 1 and 4 are connected but not adjacent

nodes 1 and 2 are adjacent
so it connected

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

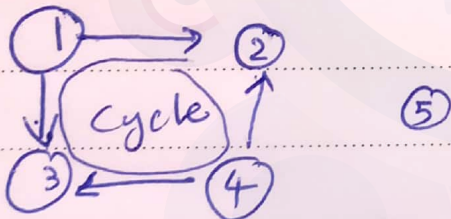


(1, 2, 3, 4) $\Rightarrow$ Path $\Rightarrow$ Connected but not complete

- Cycle



set of edges that form a closed loop in G
you can begin walking at some vertex, follow any edges in the cycle regardless of their direction, then return to ~~on~~ original vertex



it is cycle but is not feasible

not feasible لأنه انما يمرره
nodes

بالتفصيل في الفيديو

بالتفصيل في الفيديو

الحضارة

Dijkstra Alg.

$c_{ij} \geq 0$ does not allow negative cost

iter	S	A	B	C	D	t	P	PERM
init	0[0]	$\infty[-1]$	$\infty[-1]$	$\infty[-1]$	$\infty[-1]$	$\infty[-1]$	-	{ }
1		1[5]	3[5]				S cost	S
2				4[A]	5[A]		A	S, A
3							B	S, A, B
4						11[C]	C	C, S, A, B
5					10[D]		D	S, A, B, C, D

PERM

notes

S, A, B, C, D, t

A, B, C, D, t

B, C, D, t

C, D, t

D, t

t

1? (S) وبتكون

$$V(A) + C_{AB} < V(B)$$

$$1 + 2 < 3$$

^

$$\text{is } V(P) + C_{Pj} < V(j)$$

$$V(S) + C_{SA} < V(A)$$

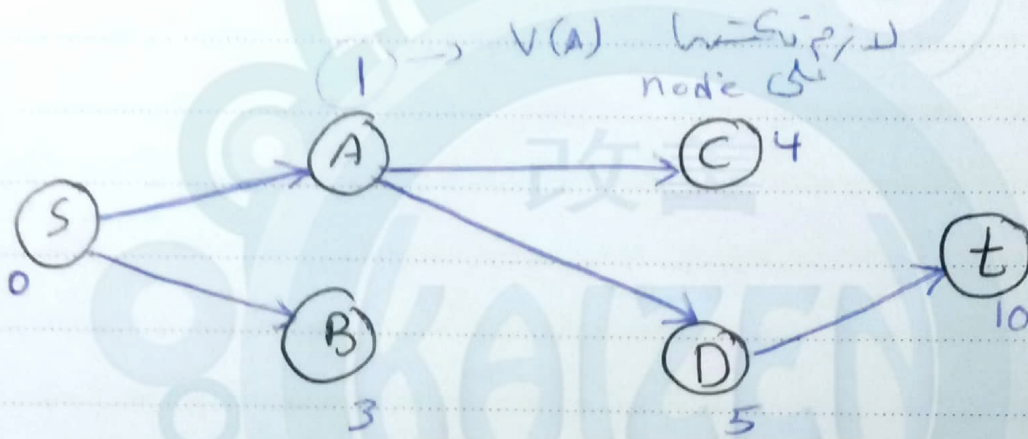
$$0 + 1 < \infty$$

yes, so $V(A) = V(A) + C_{AS}$

V_{ij} : is the cost from i to j

S A B ~~B~~C D t

0[S] 1[S] 3[S] 4[A] 5[A] 10[D]



shortest path to D ~~← 5~~ ⇒ S-A-D
 " " Cost " ⇒ 5

Multi-label algorithm for shortest path

- multiple objective: distance & time
- Vertices given labels with two component (d_k, t_k)

d_k : First cost component $\Rightarrow$ total distance required to reach this vertex along some path from s

t_k : second cost component, total time

- * vertices may have multiple labels resulting from different paths

$\hookrightarrow$ one path to vertex may have larger distance but shorter time than another path

$\hookrightarrow$ now, instead of a vertex having a predecessor

↳ one label each cost component label (d_k, t_k)
has associated predecessor ~~label~~ label
 $\text{pred}(k)$

↳ we only maintain non-dominated labels
at each vertex

• A label (d_q, t_q) is dominated by
 (d_k, t_k)

if both $d_q > d_k$ and $t_q > t_k$

• maintain a LIST of labels instead of
vertices:

remove label from LIST and examine
edges leaving from to vertex
associated with the label.

Initialization

• origin vertex s given the label $(0, 0)^s$
where sub script indicated label k ($k=1$)
and super script indicated the vertex
where the label exist (s)

$(d_k, t_k)^s$ Label k موقع s
 $(d_k, t_k)^k$ Label k موقع k

the predecessor for this label is 0 ~~pred(v)~~
 $\text{pred}(k) = \text{pred}(l) = 0$

- No labels are given to other vertices
- $\text{LIST} = \{(0,0)^s\}$
- $r=2$ (Keep track of # of next label generated)

Iterations

↳ while $\text{LIST} \neq \{\}$

I. remove label k from beginning of LIST
suppose label k has components
such as that it exists at vertex P

II. For every edge (P, j) leading from
vertex P :

a. Create a new label $\odot$ at vertex j
where $(d_r, t_r) = (d_k + d_{pj}, t_k + t_{pj})$

d. Define $\text{pred}(v) = k$

c. add label $\odot$ to end of LIST

d. Compare label r to each existing label (d_q, t_q) currently at vertex j

1. If label r is dominated by label q , discard the new label r .

and remove it from LIST

if $d_r \geq d_q$ and $t_r \geq t_q$

2. Else if label q is dominated by r discard q , and if necessary remove it from LIST

if $d_r \leq d_q$ and $t_r \leq t_q$

3. Else retain both Labels

e. $r = r + 1$

~~stopping~~

stopping criterion: when no labels remain in the LIST, no remaining label combinations can be found all non-dominated paths to all vertices have been identified.

existing vertex j

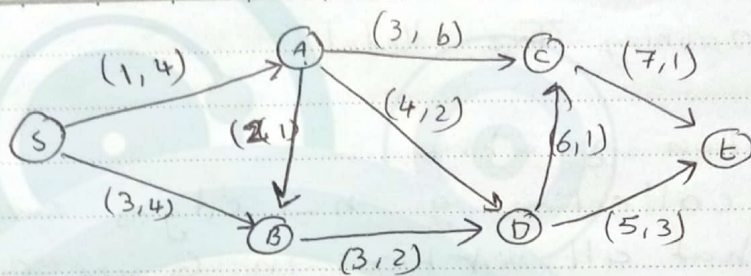
by

w

by

minimized

T

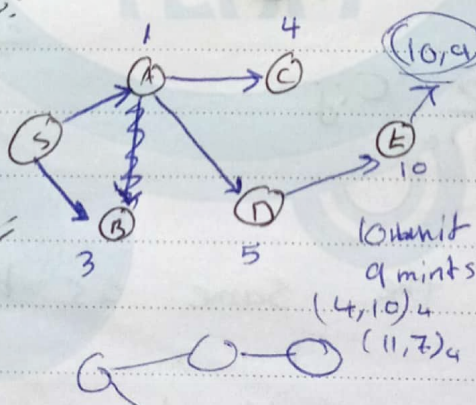


→ not a path
→ is labels

is dominated by k

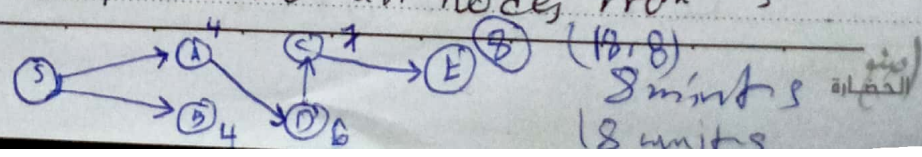
S	A	B	C	D	E	LIST	r
(0,0) ₁	(1,4) ₂	(3,4) ₃	(4,10) ₄	(5,6) ₅	(11,11) ₈	(10,0) ₁	2
		(3,5) ₆	(4,7) ₇	(6,6) ₇	(10,9) ₁₀	(1,4) ₁	3
					(18,8) ₁₁	(3,4) ₃	4
						(4,10) ₄	5
						(5,6) ₅	6
						(3,5) ₃	7
						(6,6) ₆	8
						(11,11) ₈	9
						(11,7) ₉	10
						(10,9) ₁₀	11
						(18,8) ₁₁	12

First Cost
Dominant



First Cost Component : distance shortest
distance path to all nodes from s

second Cost Component : time
shortest path to all nodes from s



Minimum spanning Tree Model

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

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

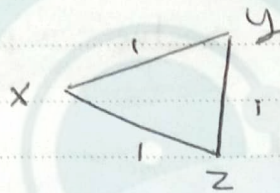
$$T^* = \arg \min_{T \subseteq G}$$

$$\sum_{(i,j) \in T} c_{ij}$$

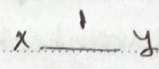
G : is MST the same as the SP tree?

No, shortest path tree minimizes the length shortest path(s) from some origin node to all other nodes

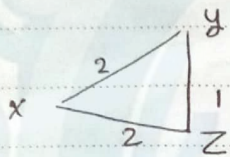
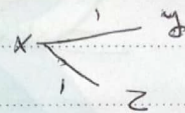
while the MST minimizes the total cost of all arcs in the tree



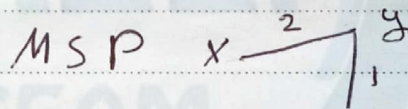
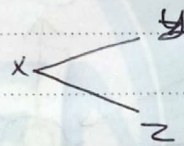
Shortest
Path from x-y
⇒



minimum span tree



SP



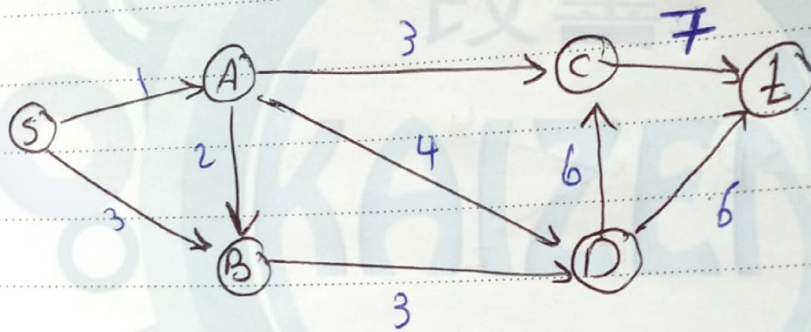
نکته: می‌توانیم از هر دو
مسیر به دست آوریم که
کمترین هزینه را دارد

Algorithm: steps, Procedure to solve a problem

↳ guarantee with optimum

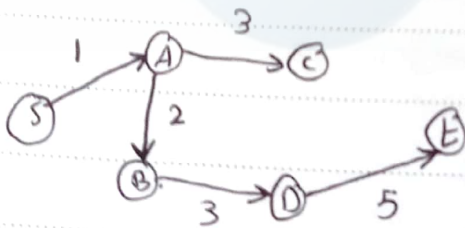
Heuristics

نفس لا بد آ بس
الفرق بينهم فارق proof
تعتبر من بس طاق ايتاار



MST $\Rightarrow$ require

TREE = { } $\rightarrow$ $< n-1$
 < 5 arcs
نوتف



~~S-A = C_{BA} = 1~~

~~A-B = 2~~

~~S-B = 3~~

~~A-C = 3~~

~~B-D = 3~~

~~A-D = 4~~

~~D-E = 5~~

~~D-E = 6~~

~~C-E = 7~~

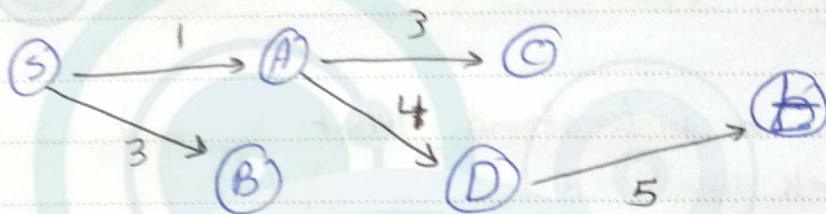
فانصفه لانه
ليس Cycle

IS(< 5) no
so Breaks

cost

$$C(MST) = \sum C_{ij} = 14$$

جميع arcs



Dijkstra's Algorithm

shortest path tree
rooted at S

not span path Tree

total cost on all arcs = 16

we must first understand what a cut is in a network

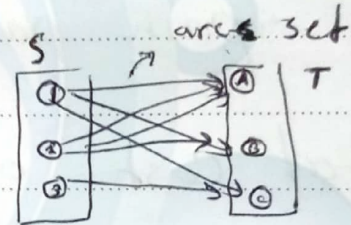
suppose vertices in G partitioned into two independent subsets (S) and (T)

The cut set (S, T) is given by all arcs (i, j)

such that:

$i \in S$ and $j \in T$

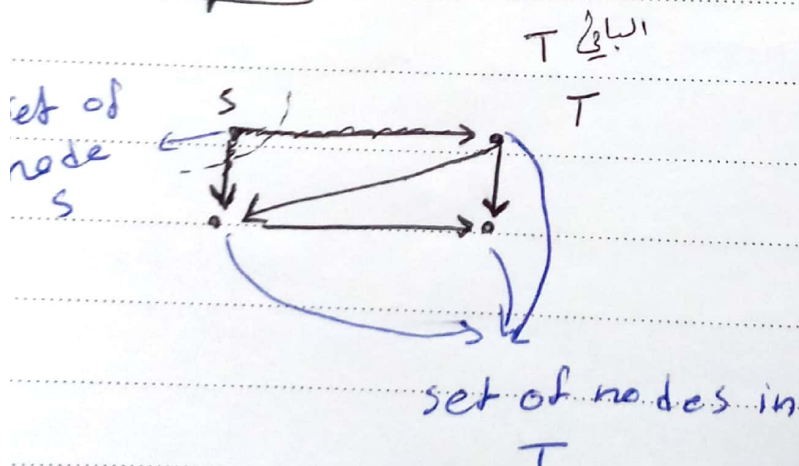
$j \in S$ and $i \in T$



$$S = \{1, 2, 3\}$$

$$T = \{A, B, C\}$$

$$\{(1, A), (1, B), (1, C), \dots\}$$



Algorithm

Initialization

• $TREE = \{ \}$

are ~~arc~~ 1 و 3 و 6

• $S = \{1\}$, $T = V \setminus S$

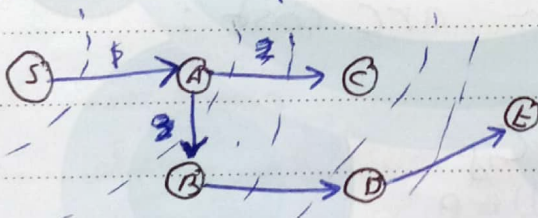
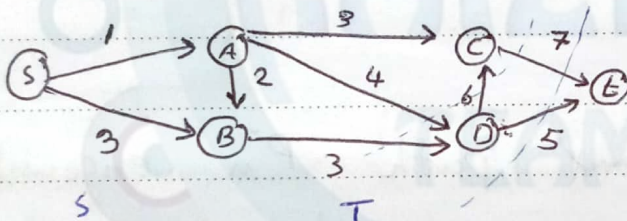
Iterations

• while $|TREE| < n-1$

→ Find arc $(i,j) \in (S,T)$ with min Cost C_{ij}

↳ add (i,j) to TREE

↳ Remove j from T and add to S



$C_{SA} = 1$

$C_{SB} = 3$

$C_{AC} = 3$

$C_{AD} = 4$

$C_{AB} = 2$

$C_{SB} = 3$

$(S, A, C, B, D) 4 < 5 \checkmark$

$5 < 5 \times$

$C(MST) = 14$

لحظة في
cut
arc
arc
cut
ب
منه

Minimum Spanning Tree application

standard minimum cost paths (shortest paths) are called mini-sum paths

بشكل عام مجموع التكاليف في المسار الأدنى

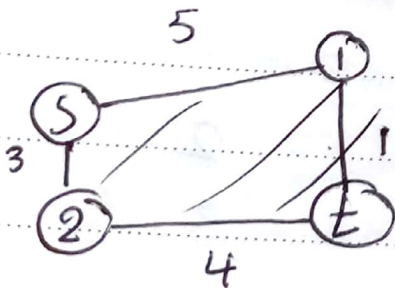
$$\begin{cases} \sum C_{ij} = \square \\ \sum C_{aj} = \square \end{cases}$$

because among all paths P among (s) to (t) the minimum path P_s^* has minimum total cost:

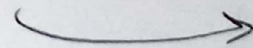
$$P_s^* = \arg \min_{P \in G} \sum_{(i,j) \in P} C_{ij}$$

Minimax Path: the minimax path P_m^* encounters the smallest maximum arc cost:

$$P_m^* = \arg \min \max_{(i,j) \in P} C_{ij}$$



Path From s to t
 $\sum C_{ij}$



Path From s to t

$\sum C_{ij}$

max C_{ij}

$\rightarrow s-1-t$

mini sum $\rightarrow 6$

5

$\rightarrow s-2-t$

7

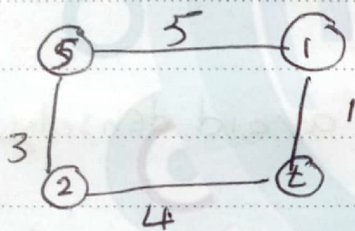
4

minimized

P_m

Jordan - USA $\Leftarrow$ Total time of trip
min

Jordan - USA $\Leftarrow$ max duration for
Connection min



spanning / mini max
time

Jordan⁽⁴⁾ - Germany⁽³⁾ - Canada⁽⁴⁾ - USA⁽⁴⁾ 16 hrs

shortest
path / mini sum

Jordan⁽²⁾ - Gater⁽¹⁰⁾ - USA⁽³⁾ 12 hrs

↑ انما بيني Connect. بيني Total Trip
اسرع فاعلى
mini sum

اهم اسمها افق بأي طيارة أقل ما يكون
(in Connection)

useful application of minimum Paths:-

- elevation change:

very steep road segments $\uparrow$

Fuel $\uparrow$, maintenance $\uparrow$

Cost, so you want to avoid segments with maximum elevation ~~Cost~~ Cause

MST $\leftarrow$

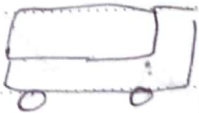
- Hazardous Materials

if spilled, population harmed, avoid densely populated road ways

$\uparrow$ (Minimize Maximum expected population exposed)

material $\leftarrow$
مادة

بأقل كمية
الخطر



minimize maximum people exposed

Review

Important Concepts :-

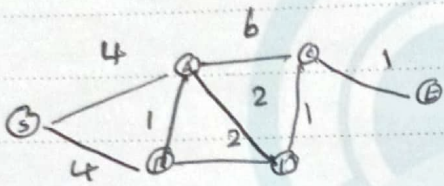
- the minimum path problem is solved by solving (shortest path problem)

- the min max problem is solved by solving a (minimum spanning tree problem)

⇒ solving minimum spanning tree problem solves the all-pairs minimax pathing problem.
~~the minimax~~

un directed path

کلیف محل



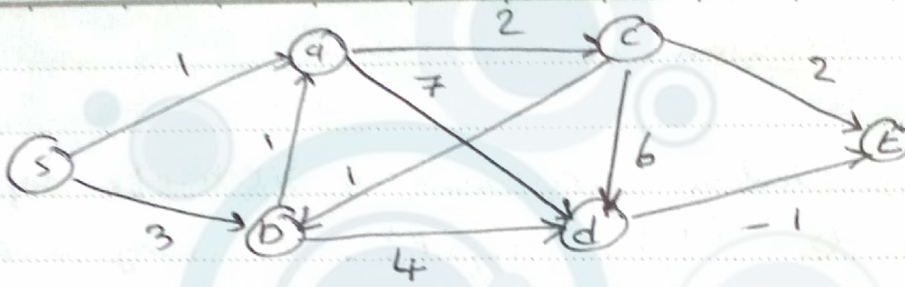
iteration	S	A	B	C	D	E	P	LIST
in	0[0]	$\infty[-1]$	$\infty[-1]$	$\infty[-1]$	$\infty[-1]$	$\infty[-1]$	-	S
1	-	4[S]	4[S]	-	-	-	S	A, B
2	-	-	-	10[A]	6[A]	-	A	B, C, D
3	-	-	-	-	-	-	B	A, C, D
4	-	-	-	-	-	-	A	
5	-	-	-	-	-	11[C]	C	D, E

S
ما بظہار لایہ
خارجہ سب سے اوپر
arc
الکاسٹ

① Arcs ~~6~~ A سب سے اوپر iteration 3 لایہ ما بظہار لایہ

Source ~~6~~ 6

Bellman Ford $\Leftarrow$ negative cost ~~6~~ 6



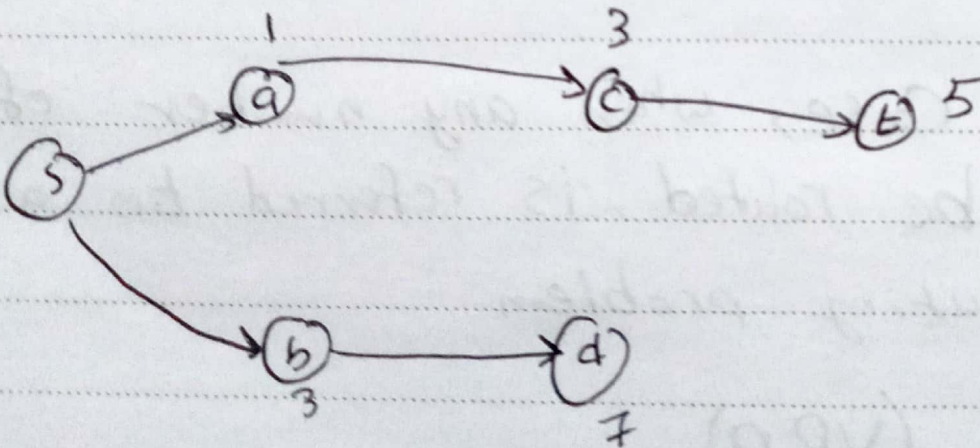
$-Z$ min $\rightarrow$ max $\rightarrow$ min

$$v(p) = \max \{ v(i), i \in \overline{PERM} \}$$

ما يتصل به p من i في $\overline{PERM}$

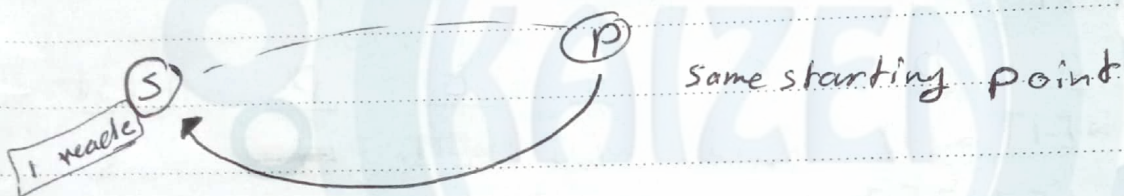
does not matter
لا يهم
Bitmask Form

iteration	s	a	b	c	d	e	p	LIST
ini.	0[0]	$\infty[-1]$	$\infty[-1]$	$\infty[-1]$	$\infty[-1]$	$\infty[-1]$	E	s
1		1[s]	3[s]				s	A, B
2				3[A]	8[A]		A	B, C, d
3					7[B]		B	C, d
4					-	5[C]	C	d, e
5	-	-	-	-	-	-	d	e
							t	



Vehicle Routing problem

In many transportation problem a vehicle departs from a node S (like depot), visiting a number of locations (eg 1, 2, ..., p) and returning to node S the objective is to choose the sequence of nodes 1, 2, 3, ..., p which minimize the cost of the circuit



when routing only a single vehicle s we refer to this as the Traveling sales man problem (TSP)

the more general case, when any number of vehicles they be routed, is referred to as the vehical routing problem

(VRP)

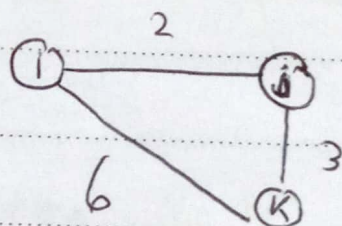
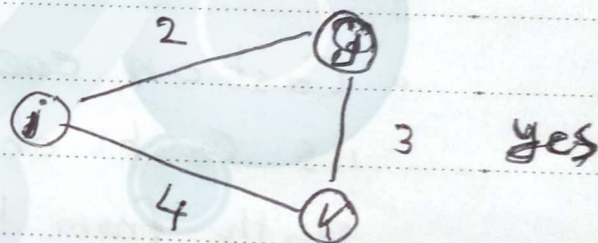
↳ the first step in solving routing problem is to develop a complete undirected network

$G(V, E)$ satisfying the triangle-inequality when V is limited to the nodes that will be visited (the start/end nodes s , and all nodes $1, 2, \dots, p$)

Triangle-inequality on a complete network:

A complete network $G=(V, E)$ satisfies the Δ -ineq iff $\forall i, j, k \in V$, the following relationship

$$C_{ik} \leq C_{ij} + C_{jk}$$



no not satisfy Δ -ineq

* Generating a complete undirected network satisfying the Δ -ineq.

* often not true

- procedure: Let $G' = (V', E')$ be a network that is not complete.

Let V be nodes of interest in routing problem, where

$$V \subset V'$$

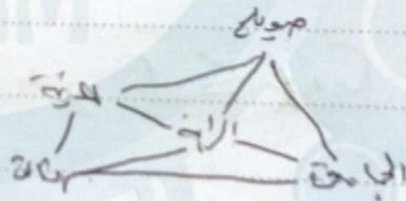
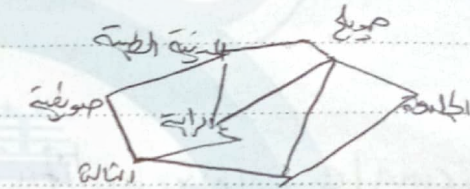
create $G(V, E)$ where

• E includes edge from each node in V to every other node in V

• Cost on each $(i, j) \in E$

is cost of minimum

path from i to j in G'



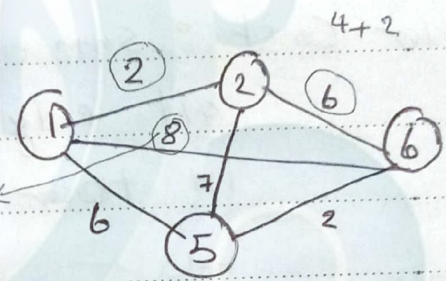
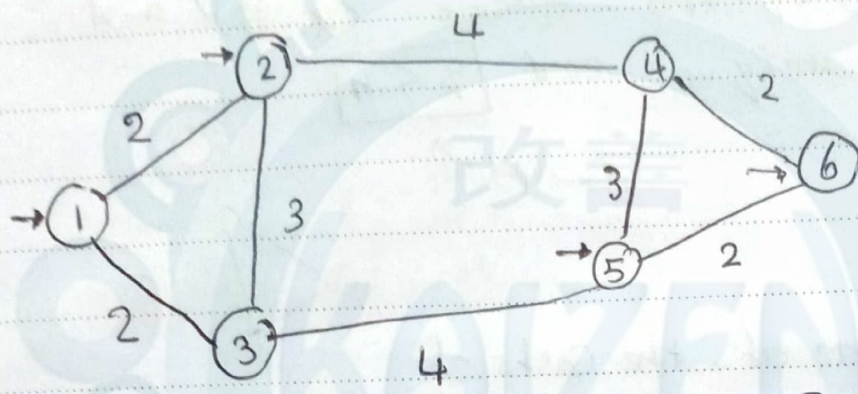
يعني Cost فيه صيغ للنوارثات minimum Path

G لازم تكون Compleat

G' مش لازم تكون Compleat

الخطوة
التي

assumptions: 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 .



$$G' = (V', E')$$

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

$$V = \{1, 2, 5, 6\} \quad \text{from } G'$$

symmetric & undirected متماثل و غير اتجاهي

TSP tour on complete undirected (symmetric) network $G = (V, E)$:

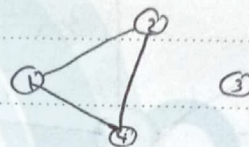
let n be the number of nodes in V . A TSP tour T on G is a sequence of all nodes in V , $T = \{i_1, i_2, i_3, \dots, i_n, i_1\}$: where $i_k \in V$ and each i_k unique to $k = 1, \dots, n$.

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

Sub tour sequence of nodes with common start and end node, such that not all nodes in V are visited

$\{i_1, i_2, \dots, i_q, i_1\}$ where $z_k \in V$ and each z_k unique and $q < n$

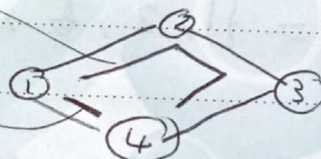
امثلة



Tour Cost: sum of the costs of all edges traversed nodes لازم زور كل nodes في tour

$q < n$ و

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



optimal TSP tour:

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

problem, with

one algorithm for TSP:

- Eliminate every tour in two network
- Choose the one with lowest cost

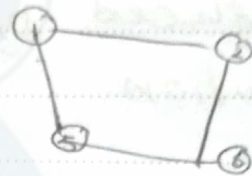
So inefficient algorithm

$n!$

are 1 are 2 are 3

$(n-1)(n-2)(n-3) \dots$

$(n-1)!$



tour (1)



tour (2)

if I can not proof $\Rightarrow$ Heuristic

Heuristic algorithms

- provides a feasible solution that may or may not be optimal

• Good heuristic:

1. provide solution close to optimal solution
2. Efficient in run time

Def

Desirables property of heuristic

• Ability to compare quality of solution produced by two heuristic to that optimal solution

• worst case quality of solutions

- these "guarantes" (Maximum deviation from optimality)

- If the cost of optimal solution is C^*
then heuristic finds a solution with cost no greater than αC^*

is constant α

* Christofid's Heuristic

Assumptions: complete network satisfying
 Δ -ineq

⇒ node degree: the degree of a node $i \in V$ with respect to an edge set E is the number of edges in E having i as an endpoint

⇒ Perfect node matching

Given a set of nodes S such that the number of nodes in S is even, a perfect node matching W is a set of $|S|/2$

nodes pairings

such that each node in S is in exactly one pairing

In a complete network, each node pairing (i, j) where $i, j \in S$ represent an arc in the network

the cost of matching W is given by $C(W) = \sum_{(i,j) \in W} C_{ij}$

Christofides heuristic

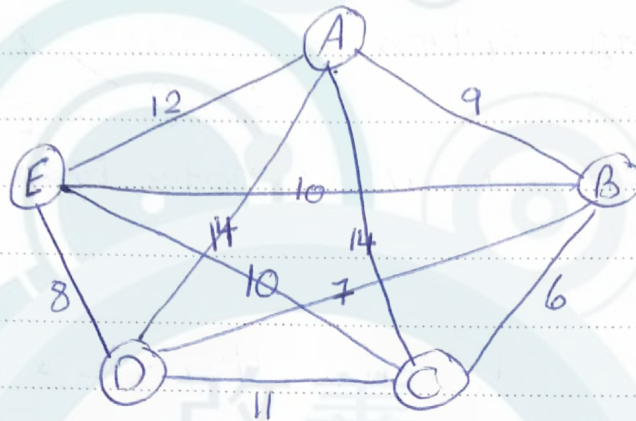
• Iterations:

Given an undirected complete network $G(V, E)$ satisfy the $\Delta-1$, Find a minimum spanning tree MST^* on G .

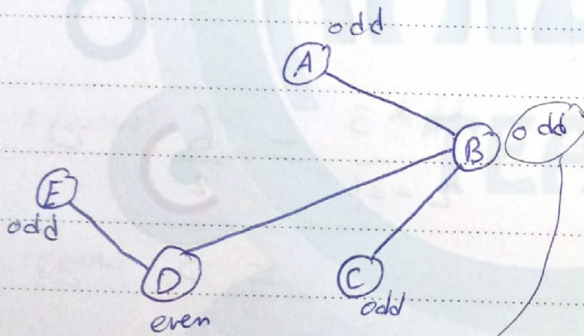
Let S be the nodes of G with odd degree with respect to MST^* .

• Find the minimum cost of perfect node matching W^* of the nodes in S .

• Generate a tour T^c of the nodes using each of the arcs in the set $MST^* \cup W^*$ exactly once (improve T^c if necessary by introducing short cuts to avoid revisiting nodes).



① minimum Spanning tree



② odd degree with respect to MST

A, B, C, E ④

|S| is even ✓

$S = \{A, B, C, E\}$ $|S| = 4$

③ A, B, C, E (pairs)

AB	9
CE	10
	19

AC	14
BE	10
	24

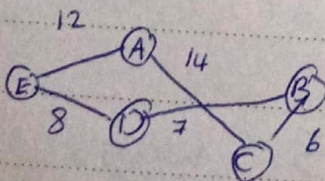
AE	12
BC	6
	18

edges $\rightarrow$ عدد الحواف
node $\rightarrow$ إذا كان عدد الحواف زوجي فبني

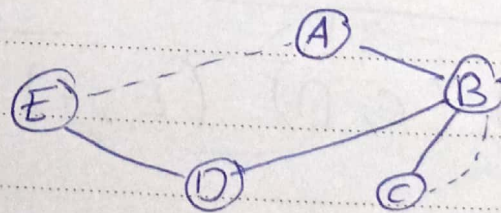
this is a perfect

④ $W^* \cup MST^*$

⑤ $\bar{A} - E - D - B - C - \bar{A}$
 $T = \{A, E, D, B, C, A\}$



$C(T) = 47$



Perfect node matching option

n nodes & nodes

2 step < 2 node

we need to set at 2 nodes from 4 than 2 from 2.

* of waest to divide 4 nodes into 2 groups

$$C_2^4 * C_2^2 = \frac{4!}{2! (4-2)!} * \frac{2!}{2! (2-2)!}$$

$$= \frac{4 * 3}{2 * 2!} = \frac{6}{2} \text{ ways}$$

$$= 3 \text{ ways}$$

ص لا متسا

6 nodes into groups of 2

$$C_2^6 * C_2^4 * C_2^2 = \frac{6 * 5 * 4!}{2 * 4!} * \frac{4 * 3 * 2!}{2! * 2!} * 1$$

(A, B) (C, D) (E, F)

بافرقه معي ادا وزعتم

$$= 15 \text{ ways}$$

A, B, C, E, F

Let T^c be tour produced by christofides

Theorem: worst-case performance Bound for Christofides:

chr. Heuristic is 1.5-optimal for every complete network G satisfy D-ity

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

↑
actual optimum of TSP
computation

أفضل tour ممكن 14

$$\frac{14}{1.5} = \text{best tour of TSP (Possible)}$$

Construction heuristics for TSP

A. Nearest neighbor heuristic

Assumption: Complete undirected network
Satisfying Δ -ineq.

Initialization, 2 - Options

- or
- Let $P = \{i, j\}$ where $C_{ij} = \min_{(i,j) \in E} C_{ij}$
 - Let i be arbitrary start node
Find $j \in V \setminus \{i\}$ such that C_{ij} minimized. Let $P = \{i, j\}$

Iterations

while $|P| < n$:

- Let s be the beginning node of the path and t be the end
- Choose the node $k \in V \setminus P$ to enter the path:
- Let k be such that C_{tk} or C_{ks} is

minimum $\Rightarrow$

معنى الحالة الأولى

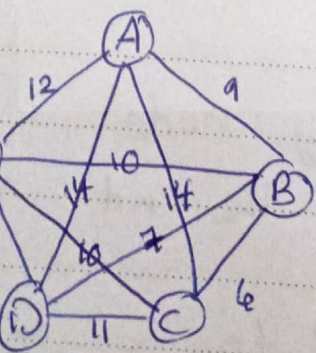
you have find the node closest to

- the new path is either $P + \{k\}$ or $\{k\} + P$ respectively

Let $\text{begin}\{P\}$ be the beginning node of the path

Let $T = P + \text{begin}(P)$

$\rightarrow$ If C_{Tk} is minimized, $P = P + \{k\}$
 If C_{ks} " " " " , $P = \{k\} + P$



Initialization

① $\min C_{ij} \Rightarrow B-C$

Iter 1

B: Closest is D 7

C: " " E 10

$P = \{D-B, C\}$



الحضارة

Iter 2

$P = \{D, B, C\}$

D: closest E 8

C: " E 10

نقطة البداية والنهاية

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

Iter 3

Tour $\Rightarrow$

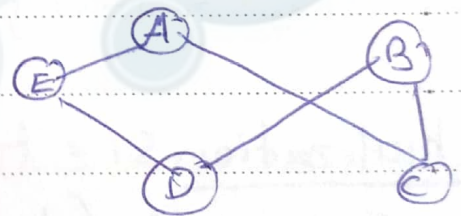
E: closest A 12

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

C: closest A 14

Cost = 47

$P = \{A-E-D-B, C\}$



Cost not include starting point

B. Nearest Insertion Heuristic

Insertion Cost: inserting node j between nodes L & K when L & K are a pair of successive nodes in a tour, has cost:

$$[C_{ij} + C_{jk} - C_{LK}]$$



Can never be negative if network satisfies Δ -ineq

Assume: undirected $G = (V, E)$ satis. Δ -ineq

Initialization

- Find arc (i, j) with minimum cost in E
- Let $T = \{i, j, i\}$

Iteration

while $|T| < n+1$

- Find the node not in the tour closest to a node in the tour

$\Rightarrow$ Let $C_j(T)$ be $\min C_{ij}$ for all nodes

$j \notin T \quad i \in T$

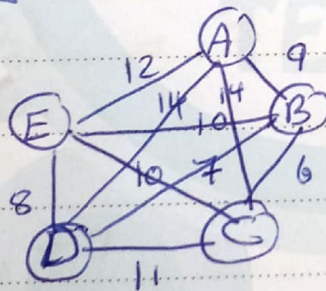
choose node j to insert whose $C_j(T)$ is $\min$

• Find insertion location in tour with minimum insertion

$\Rightarrow$ Find arc (L, K) in tour where $C_{ij} + C_{jk} - C_{ik}$ is minimized.

• Insert j b/w $L \& K$ in T .

example



initial -

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

$C(T) = 12$

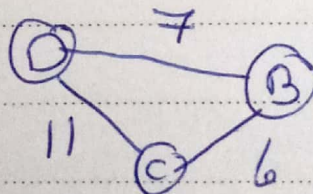
Iter 1

$j \notin T$	B	C
A	9	14
D	7	11
E	10	10

Cost of insertion = $C_{BD} + C_{DC} - C_{BC}$

$$= 7 + 11 - 6 = 12$$

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



Iter 2

$i \backslash j$	B	C	D
A	9	14	14
E	10	10	8

$$T = \{B - D - C - B\} \quad C(T) = 24$$

Insertion Location

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

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

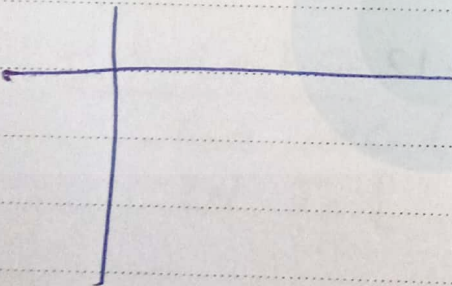
Iter 3

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

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

$$\text{Cost}(T) = 24 + 7 = 31$$

Iter 3



collection data : - Google map

- Drives

-  car

Iter 3

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

Insert A: Since it is the last node

location

← $(B, D) = 16 = C_{BA} + C_{AD} - C_{BD} = 9 + 14 - 7$

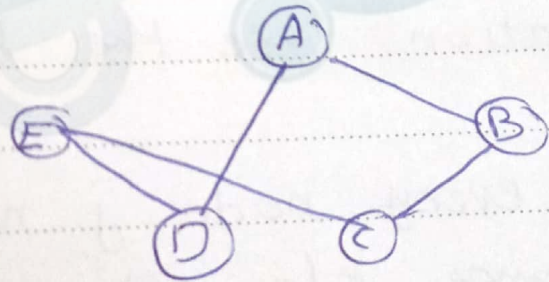
$(D, E) = 18$

← $(E, C) = 16$

$(C, B) = 17$

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

$C(T) = 47$



Co Farthlers Insertion Heuristic

Assumption undirected network $G = (V, E)$
satisfy Δ -ing

Initialization

- Find arc (i, j) with maximum cost in E .
- Let $T = \{i, j, i\}$

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 min. distance edge (i, j) connecting node j to a node in the tour that is:

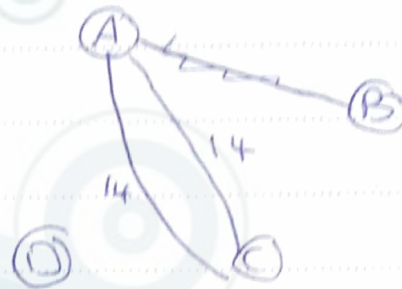
Let $C_j(T)$ be $\min_{i \in T} C_{ij}$ for $j \notin T$.

Pick node j for which $C_j(T)$ is maximum

- the node to insert is the one with maximum $C_j(T)$

initialization

init. $C(T) = 28 = 14 + 14$

$$\{A - C + A\}$$


ET ET	A	C
B	9	6
D	14	11 ←
E	12	10

3 min دو طرفه

row 5

عربی و قرآن

$$\{A-D-C-A\}$$

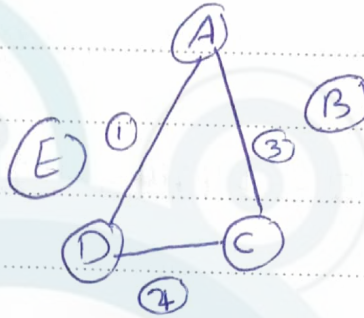
$$C_{AD} + C_{DC} - C_{AC} = 11$$

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



Iteration 2

set \ set	A	C	D
B	9	6	7
E	12	10	8



$$C_{AD} + C_{ED} - C_{AD} = 6$$

لنرمز T بكونه $sub-T$ من T بحيث يكون $nodes$ من T

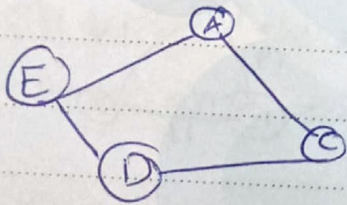
$$C_{DE} + C_{EC} - C_{DC} = 7$$

insert E b/w (A, D)

$$C_{CE} + C_{EA} - C_{CA} = 8$$

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

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

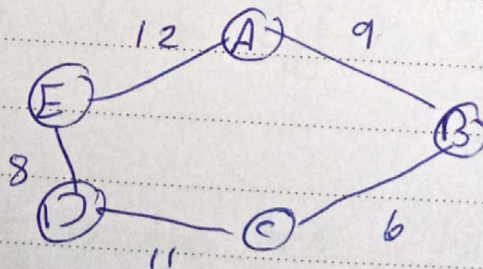


Final Iteration

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

Insert B between A & C $\Rightarrow$ insertion cost = 1

$$C(T) = 45 + 1 = 46$$



D. Cheapest insertion heuristic

Assumption: undirected network $G=(V, E)$
satisfying Δ -ineq.

Initialization

- Find the arc (i, j) with minimum Cost in E , $T = \{i, j, i\}$

Iteration

while $|T| \leq n+1$

- For every node j not in the tour, find the insertion cost for all possible insertion locations in the tour

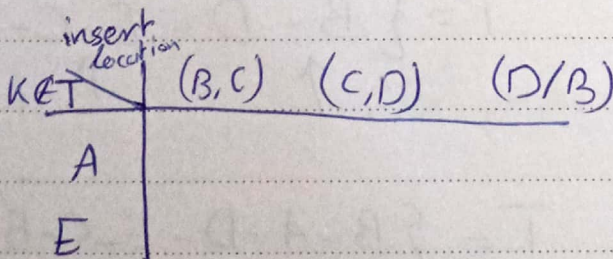
Find (i, j) in the current sub-tour and k not in the sub-tour, such that

$C_{jk} + C_{ij} - C_{ij}$ is ~~minimized~~ minimal

$\Rightarrow$ insert k between i & j

assume the current tour is

$B - C - D - B$



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

Iteration

insertion lock	(B, C)	
A	$9 + 14 - 6 = 17 = C_{BA} + A_{AC} - C_{BC}$	$T = \{B - D - C - B\}$
D	$7 + 11 - 6 = 12$	
E	$10 + 10 - 6 = 14$	$C(T) = 24$

insertion	(B, D)	(D, C)	(C, B)	
A	16	17	17	$T = \{B - D - E - C - B\}$
E	11	<u>7</u>	14	$C(T) = 31$

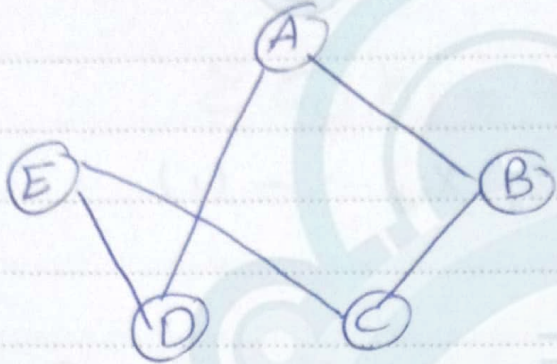
	(B, D)	(D, E)	(E, C)	(C, B)
A	<u>16</u>	18	<u>16</u>	17

نفس الـ 16، (E, C) و (B, D)

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

↑ ↑

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



$$C(T) = 47$$

Mathematical formulation of TSP

Complete undirected graph $G = (V, E)$ edge $(i, j) \in E$ has cost c_{ij} and cost satisfy Δ -ineq

$\Rightarrow$ The symmetric TSP is formulated as follows

note: that $i < j$ $\overset{\text{مثال}}{(1,2) (2,3)}$ for each $(i, j) \in E$

إذا كانت node اسم أماكن بـ i, j

Let $x_{ij} = \begin{cases} 1 & \text{if } (i,j) \text{ is included in the tour} \\ 0 & \text{o.w} \end{cases}$

$$\min \sum_{(i,j) \in E} c_{ij} x_{ij} \Rightarrow \sum_i \sum_j c_{ij} x_{ij} \quad \text{--- (1)}$$

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

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

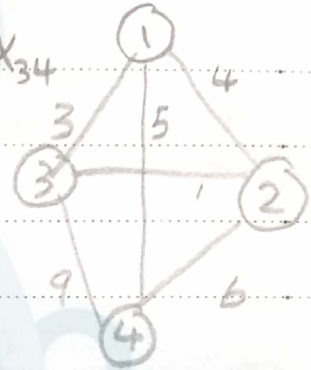
subset
يحتوي

$$2 \leq |S| \leq \left\lceil \frac{|V|}{2} \right\rceil$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in E$$

Objective (1) : minimize total cost

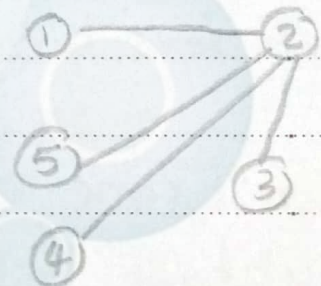
$$\sum C_{ij} X_{ij} = 4 X_{12} + 3 X_{13} + 5 X_{14} + 1 X_{23} + 6 X_{24} + 9 X_{34}$$



Constraint set (2): exactly 2 eds incident to each vertex (in, out) $\rightarrow$

For $j=2$: $(X_{12}) + (X_{23} + X_{24} + X_{25}) = 2$

For $j=3$: $(X_{13} + X_{23}) + (X_{34} + X_{35}) = 2$



An alternative Formulation to cost (2)

$$\sum_{i \in V : (i,j) \in E} X_{ij} = 1$$

$$\forall j \in V$$

Each node j has exactly one predecessor

$$\sum_{j \in V : (i,j) \in E} X_{ij} = 1$$

$$\forall i \in V$$

each node i has exactly one successor

both replace

cost set (2) in the original mathematical formulation

Const (2) to guarantee that each node is visited once



من يطلع من اول مرة يكون لا
not feasible

Constraint set (2) alone can result in subtour trap

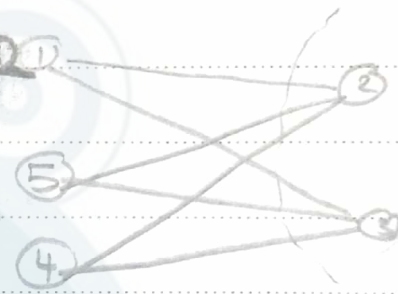
⇒ we need another const set to eliminate subtour

Constraint set (3) : subtour elimination we should have one such constraint for every possible subset of size 2 and 3
5 nodes
as a example

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

$$(X_{24} + X_{25} + X_{34} + X_{35}) + (X_{12} + X_{13}) \geq 2 \quad (1)$$

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



$$(X_{14} + X_{15} + X_{24} + X_{25} + X_{34} + X_{35})$$

$\forall S \subset V$

? $1, 4, 5$ $\hookrightarrow$ subset of V takes care of S of size 1

size 4 $\Rightarrow$ complement of size 1

cont (2)

size 5 $\Rightarrow$ this is what we looking for

$$2 \leq |S| \leq \lceil \frac{n}{2} \rceil$$

$$2 \leq |S| \leq 3$$

$$2 \leq |S| \leq \lceil \frac{5}{2} \rceil$$

Constraint set (3)

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

- why not of size 1? Constraint set (2) takes care of it

- For subset of size $> \lceil \frac{|V|}{2} \rceil$ these subsets are complements to some subset of size $< \lceil \frac{|V|}{2} \rceil$

- subsets of size $|V| \Rightarrow$ it is the tour we are looking for

Constraint set (4) introduced new, is an alternative for Constraint set (3) : would replace constraint set (3)

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

$(i,j) \in E, i \in S, j \in S$

Subtour elimination constraint

For $S = \{1, 2, 3\}$

$(i,j) \in S$

(5)



$$x_{12} + x_{13} + x_{23} \leq 3 - 1$$

Subtour elimination Const.

one vehicle one depo

Vehicle Routing Problem

TSP: all Customer request are served by single vehicle (tour)

In the VRP, m vehicle are based at a depot, and the problem is to find the set of m least cost tour, based at the depot, serving all (or a subset of) Customer request

Common operational constraints in vehicle routing problem

- * of vehicles m

⇒ can be fixed, or a decision variable

- Capacitated vehicles ⇒ Capacity Q ⇒

- * of unequal Volume that can fit in a truck

work shift duration → Capacity constraint

Time windows of customers → show up no earlier than 8am and no later 10pm

The classic VRP includes the capacitated vehicle constraint.

suppose a homogeneous fleet of vehicles, each with capacity Q , is based at the depot, and each of n customer demands that a quantity $q_i, i=1, \dots, n$

be picked up or delivered and demand cannot be split between vehicles.

Travel between location i & j incurs cost c_{ij} .

The VRP is to find a set of vehicle tours, each beginning and ending at the depot, such that each customer's demand is served, no vehicle violate capacity constraint, and total cost of all vehicles is minimized.

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

$$\begin{aligned} \text{TSP min } m &= 1 \\ Q &= \infty \end{aligned}$$

$$Q = \infty \equiv q_i = 0$$

⇒ Another problem with important application in transportation and logistics problem is a special case VRP

⇒ Bin Packing problem

- Assumes $C_{ij} = 0$ for all (i, j)

- ↳ this means the tour is negligible

- The capacity constraint remains

$$\text{VRP} = \text{BPP} + \text{TSP}$$

$\uparrow$ $\uparrow$
 $C_{ij} = 0$ $q_i = 0$

Bin Packing problem

Let m be the numbers of item to be loaded
and n be the number of bins.

Let p_i , $i = 1, \dots, m$ be the weight of item i

Let q_j , $j = 1, \dots, n$ " " Capacity of bin j

Items cannot be split between the Problem is
to assign each item to a bin such that:

• total weight of items in bin $j \leq q_j$

• Smallest # of bins is used

Veriance of picking problems

• offline $\rightarrow$ all item sizes known in advance
when you solve two problems

• outline $\rightarrow$ Input data is revealed over time

Mathematical formulation of the BPP

Let $y_j = \begin{cases} 1 & \text{if bin } j \text{ is used} \\ 0 & \text{o.w} \end{cases}$

Let $x_{ij} = \begin{cases} 1 & \text{if item } i \text{ is assigned to bin } j \\ 0 & \text{o.w} \end{cases}$

$$\min \sum_{j=1}^n y_j \quad (1)$$

$$\text{s.t.} \quad \sum_{j=1}^n x_{ij} = 1 \quad \forall \quad i=1, \dots, m \quad (2)$$

$$\sum_{i=1}^m p_i x_{ij} \leq q_j \quad \forall \quad j=1, \dots, n \quad (3)$$

$$p_1 x_{11} + p_2 x_{21} + \dots + p_m x_{m1} \leq q_1$$

Can not assign pin
not existed

$$x_{ij} \leq y_j \quad \forall \quad i=1, \dots, m \quad j=1, \dots, n \quad (4)$$

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

- objective (1): minimize # of bins used
- Constraint set (2): each item is assigned to exactly one bin
- Constraint set (3): total weight ~~to~~ bin capacity is not violated
- Constraint set (4): we can not assign an item to a bin unless the bin is opened

Lower bounds for BPP

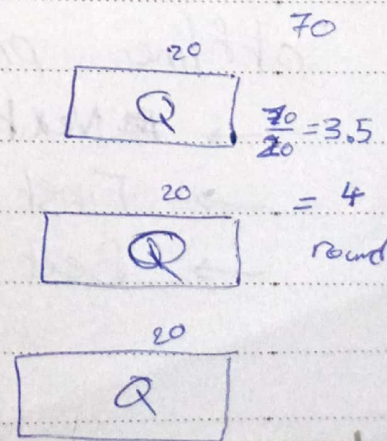
Assume $q_j = Q$ for all $j = 1, \dots, n$

then what is a valid lower bound on the # of bins required for items $i = 1, \dots, m$

$$q_i = Q \quad \forall i$$

$$\therefore LB \text{ is } \left\lceil \frac{\sum p_i}{Q} \right\rceil$$

Best solution



⇒ Bins have different sizes

$$q_i \neq 0$$

∴ take the largest bin ⇒ let $r = \max q_i$

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

* Bpp has no efficient ^{optimum solution} algorithm for its ~~st~~ solution

⇒ Heuristic

• Heuristic solution approach
outline problem

- next fit (NF), 2 optimal
- First fit (FF), 1.7 optimal
- Best Fit (BF), 1.7 optimal

offline problem

- ~~Next~~ Next Fit Decreasing (NFD)
- First Fit Decreasing (FFD), 1.5 optimal
- Best Fit Decreasing (BFD), 1.5 optimal

Variance of Baking Problem

offline: all in size known in advance

outline: Input ~~is~~ data is revealed
irrevocable decision made
with a some final limits

طابق المسألة
وقال في
تعاليم الشرح
بأنه لا

The difference in online & offline heuristics is that in the offline versions items are sorted in decreasing order under the Heuristic begins

ex: items weight/size 4, 9, 4, 7, 10, 3, 2, 11

become

11, 10, 9, 7, 5, 4, 3, 2

In each of the Heuristics belows

Let L be a list of item weights, $L = \{P_1, P_2, \dots, P_n\}$

The exam we will use for each heuristic has

$$q_1 = Q = 20$$

$$\text{and } L = \{16, 10, 14, 12, 4, 8, 3, 12, 7, 14, 9, 6, 3\}$$

$$\text{A quick LB} = \left\lceil \frac{\sum P_i}{Q} \right\rceil = \left\lceil \frac{118}{20} \right\rceil = 6 \text{ bins}$$

online

I. next Fit heuristic (NF)

Initialization: Place item in bin and remove fail

Let $j=1$, $i=2$

Iteration

→ If item i fits in bin j , Place i in j
otherwise open a new bin $j+1$ and place i in $j+1$

Let $j=j+1$

→ remove i from list L , Let $i=i+1$

→ while items ~~remain~~ remain in L , repeat
advantage of NF: allows immediate dispatch of bin j (vehicle j) once bin $j+1$ is opened.

								capacity
16	10	14	12, 4	8, 3	12, 7	14	9, 6, 3	
1	2	3	4	5	6	7	8	20

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

حسب الترتيب الذي يعطيه
للا online الخوارزمية

First Fit heuristic (FF)

- Initialization: Place item 1 in bin 1 and remove from L

Let $i=1$, $j=1$

Iteration

- place i in the lowest number bin for which it fits.
If item i does not fit in any open bin, open a new bin $j+1$ and place i in it.
- Remove i from L. Let $i=i+1$
- Until item ~~remove~~ remove in L repeat

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

$$\begin{array}{ccccccc} 16, 4 & 10, 8 & 14, 3, 3 & 12, 7 & 12, 6 & 14, 3 & 9 \\ \hline 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array}$$

$$Q = 20$$

لا يمكن وضع 8 في أي من الحاويات

Best Fit heuristic:

Initialization: Place item 1 in bin 1 and
remove From let $j=1$, $i=2$

Iteration

• Find the bin j whose contents (total weight of items)

are maximum but not greater than $Q - P_2$

if j fits in this bin

If item i does not fit in any open bin, open a new bin $j+1$ and place bin it.

- Remove i from L . let $i=i+1$
- while item i is not in L , repeat

16,4	10,9	14,3,3	12,8	12,7	14,6
1	2	3	4	5	6
x			x		x

$Q = 20$

Best Fit
بين كل الكيل بعد من خط الانسب

Practical issues: with FF & BF

If any open bin is completely full at
some iteration, it can be closed / dispatched

but ~~to~~ otherwise. so, in practice it is necessary to add some rule. similar to dispatch j once j reaches fill percentage from its full capacity. $Q = 20$

(16) ~~المسألة~~ ^{المسألة} ~~المسألة~~

or once bin j has been open for y time unit or a combination.

Practice:

create the solution to this instance of ~~BPP~~ BPP using NFD, FFD, BFD

Heuristic ^{نفس}
بن الفرقه

the only difference is that you will initialize each heuristic with L sorted in a decreasing ~~at~~ one.

الترتيب

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

NFD : 8 bin
FFD : 6 bin
BFD : 6 bin

} offline

Recall, in the VRP

- Fleet of homogenous vehicles, each with capacity Q , based at depot.
- n customers with demand q_i
- Find least cost set of vehicle tours (based at depot) serving all customers.

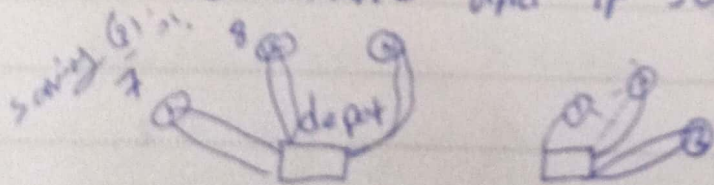
Heuristics for VRP.

1. Clarke - Wright savings heuristic
2. Route-First cluster second (RFS) Heuristics
3. Cluster " Route-second (CFRS) "

Clarke - Wright saving heuristic:

Begin with a separate tour for each tour.

Then combined tours, if feasible with respect to demand and if savings result.



$$Q = 20$$

Let T_i and T_j be two tours, and let $\text{end}(i)$ denotes the end of a tour i , and $\text{begin}(j)$ be the beginning of tour j

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

CWSTH:

- ① For each node $i \in N$, construct a tour T_i beginning at the depot, traveling to node i and returning to the depot

Let S be the set of tours

- ② calculate the saving S_{ij} for each pair of nodes i, j
$$S_{ij} = C_{0i} + C_{j0} - C_{ij}$$

- ③ Sort the savings in non-increasing ^{decreasing} order

- ④ walk down to saving combination list, and find the next feasible pair (i, j) such that:
 - i and j are on different tours, and i is the last node on its tour T_k and j is the first node on its tour T_c

- $S_{ij} \geq 0$

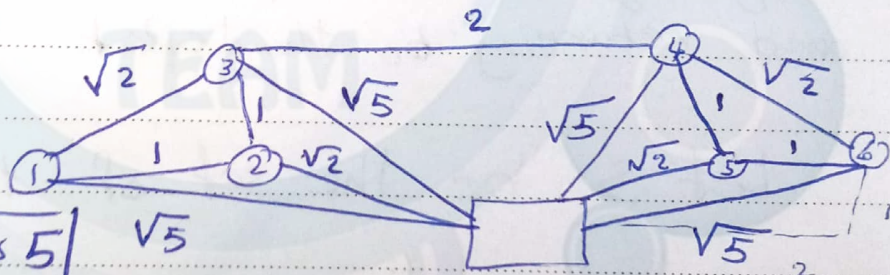
- $\sum_{T \in T_i, T_j} q_k \leq Q$

T_g

- Add the combined tour $T_i + T_j$ to S and remove T_i and T_j for S

- repeat this step as long as feasible combination remain the List.

example



Vehicle Capacity is 5
demand at customer
1, 2, 3 is 3

demand at customer
4, 5, 6 is 2

$$LB \text{ on vehicles} = \left\lceil \frac{\sum d_i}{Q} \right\rceil = \left\lceil \frac{9+6}{5} \right\rceil = 3$$