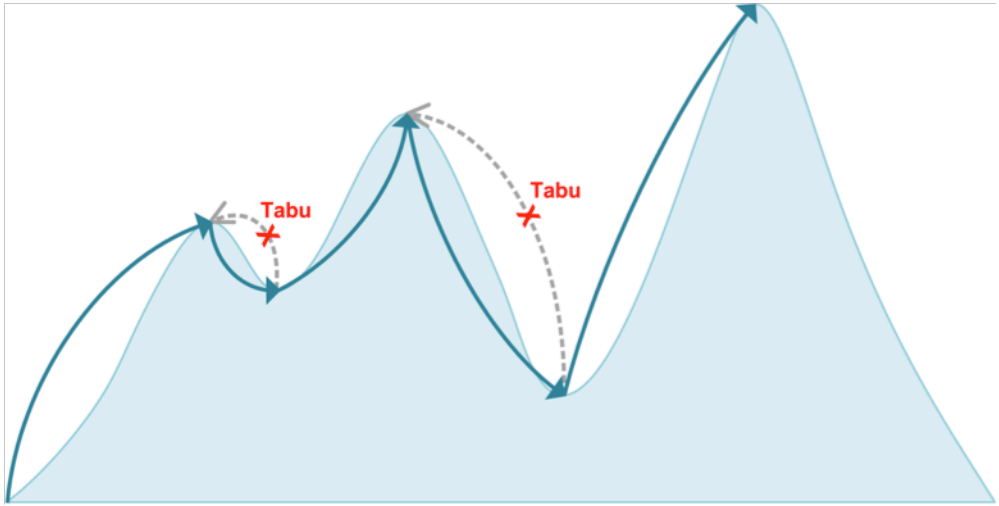
Εξελικτικοί Αλγόριθμοι Βελτιστοποίησης

Αναζήτηση με Απαγορευμένες Καταστάσεις (Tabu Search)



Υπεύθυνος μαθήματος : Αθανάσιος Χασιακός
Επιμέλεια παρουσίασης : Παναγιώτης Τσίκας

Αναζήτηση με απαγορευμένες καταστάσεις

Η μέθοδος εστιάζει σε τοπική αναζήτηση γύρω από μια αρχική λύση, αλλάζοντας μια μόνο μεταβλητή του διανύσματος λύσεων κάθε φορά κι αξιολογώντας τις νέες λύσεις που προκύπτουν από τη διαδικασία αυτή.

Βέβαια, η συνεχείς αλλαγές μεταβλητών οδηγεί προοδευτικά σε λύσεις αρκετά μακριά από την αρχική.

Σε αντίθεση με τις άλλες μεθόδους βελτιστοποίησης (οι οποίες υιοθετούν στοχαστική αναζήτηση νέων λύσεων), η μέθοδος αυτή εξελίσσεται ντετερμινιστικά (αυτό σημαίνει ότι αν ξεκινήσει κάποιος από συγκεκριμένη αρχική λύση, θα καταλήξει σε συγκεκριμένη τελική λύση).

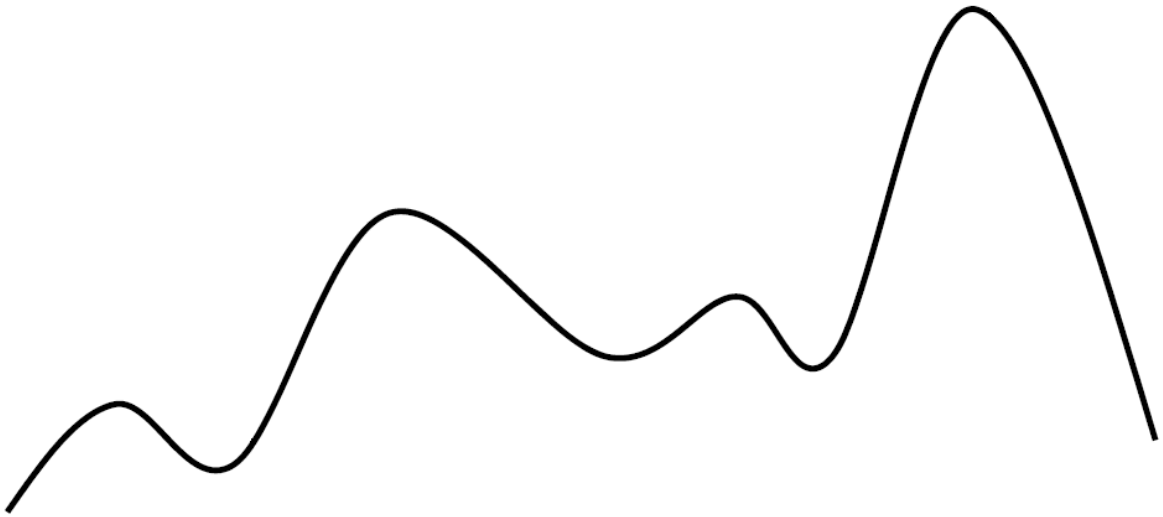
Η βασική ιδέα της μεθόδου (εξ ης και η ονομασία) είναι ότι αλλαγές στην τρέχουσα λύση που έχουν υλοποιηθεί στο (πρόσφατο) παρελθόν καταχωρούνται ως «απαγορευμένες» (tabu) και δεν αξιολογούνται στη συνέχεια για έναν αριθμό φάσεων της βελτιστοποίησης.

Ο αλγόριθμος Tabu Search αναπτύχθηκε από τον Glover το 1986.

Αναζήτηση με απαγορευμένες καταστάσεις

Βασικά σημεία της αναζήτησης με απαγορευμένες καταστάσεις:

1. Επιλογή αρχικής λύσης
2. Συνάρτηση καταλληλότητας
3. Τοπική («γειτονική») αναζήτηση νέας λύσης
4. Δημιουργία/ενημέρωση μνήμης «απαγορευμένων» λύσεων
5. Κριτήριο επιλογής νέας λύσης
6. Διαδικασία τερματισμού

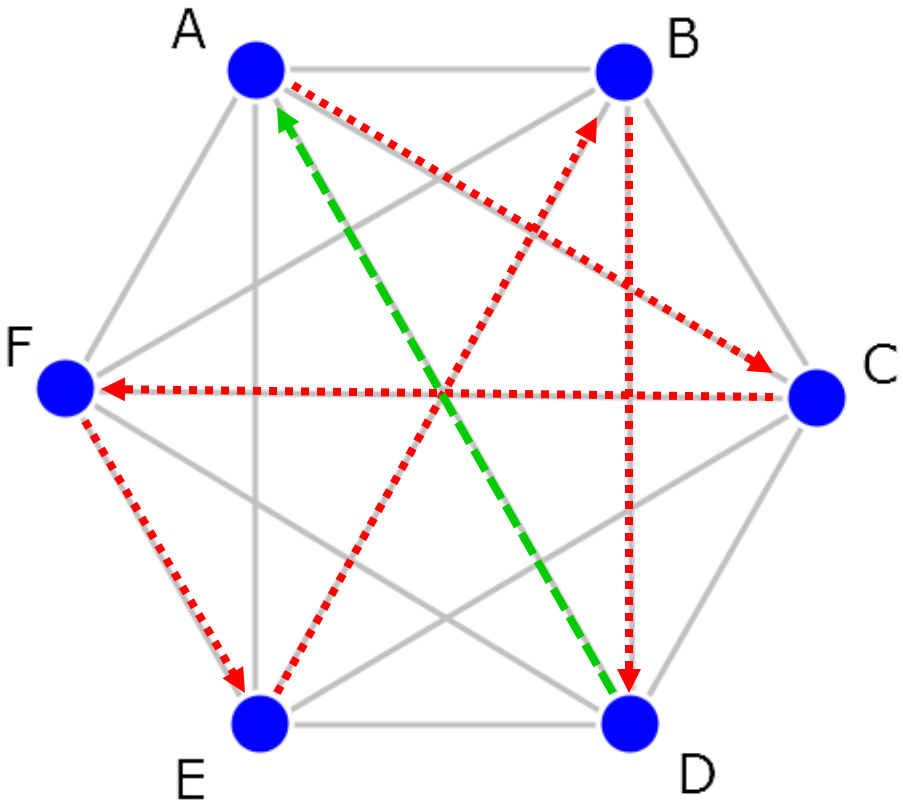


το Πρόβλημα του Πλανόδιου Πωλητή (TSP)

Ο πλανόδιος πωλητής πρέπει να περάσει από όλες τις πόλεις του προβλήματος διανύοντας τη μικρότερη δυνατή απόσταση (ή με το μικρότερο δυνατό κόστος) ξεκινώντας και καταλήγοντας στο ίδιο σημείο.

Πρόβλημα TSP

Τυχαία λύση



Πίνακας αποστάσεων

	1	2	3	4	5	6
1		150	190	200	160	60
2	150		150	60	160	180
3	190	150		140	90	190
4	200	60	140		110	80
5	160	160	90	110		110
6	60	180	190	80	110	

A	C	F	B	E	D	A	
1	3	6	5	2	4	1	910

TSP - Tabu Search

Βήμα 1. Επιλέγουμε μια αρχική λύση X_0 για το πρόβλημα

A	C	F	B	E	D	A
1	3	6	5	2	4	1

910

Βήμα 2. Μεταβάλουμε (τοπικά) τις μεταβλητές της αρχικής λύσης (τη σειρά μετάβασης των πόλεων) με αντιμετάθεση ενός ζεύγους μόνο πόλεων κάθε φορά (στο παράδειγμα είναι εφικτές 10 αντιμεταθέσεις).

Βήμα 3. Από τις νέες λύσεις που πρόεκυψαν από την τοπική αναζήτηση, επιλέγεται ως νέα λύση X_1 η καλύτερη (με βάση την τιμή καταλληλότητας) έστω κι αν δεν είναι καλύτερη από την αρχική (διατηρείται όμως και η βέλτιστη στη μνήμη).

Βήμα 4. Η αντιμετάθεση με βάση την οποία προέκυψε η βέλτιστη λύση (έστω η 3 με την 6, δες επόμενη σελίδα) καταχωρείται στη μνήμη και δεν θα υλοποιηθεί ξανά (η συγκεκριμένη αντιμετάθεση) στις επόμενες πέντε φάσεις της διαδικασίας.

	2	3	4	5	6
2					
3					5
4					
5					
6					

t=1

TSP - Tabu Search (t=1)

	A	C	F	B	E	D	A	
	1	3	6	5	2	4	1	910
I	1	2	6	5	3	4	1	870
II	1	3	6	5	4	2	1	810
III	1	3	6	2	5	4	1	910
IV	1	3	2	5	6	4	1	890
V	1	4	6	5	2	3	1	890
VI	1	5	6	3	2	4	1	870
VII	1	6	3	5	2	4	1	760
IIX	1	3	6	4	2	5	1	840
IX	1	3	4	5	2	6	1	840
X	1	3	5	6	2	4	1	830

t=1	2	3	4	5	6
2					
3					5
4					
5					
6					

Πίνακας μνήμης

Στο διπλανό πίνακα καταγράφονται όλες οι πιθανές αντιμεταθέσεις ζεύγους πόλεων γύρω από την αρχική λύση (μια αντιμετάθεση μόνο κάθε φορά).

Επιλέγεται η βέλτιστη λύση (VII) που αντιστοιχεί στην αντιμετάθεση 3↔6.

Στον ανωτέρω πίνακα καταγράφεται η «απαγορευμένη» αλλαγή για τις 5 επόμενες φάσεις της διαδικασίας.

TSP - Tabu Search (t=2)

	A	C	F	B	E	D	A	
	1	6	3	5	2	4	1	760
I	1	6	2	5	3	4	1	830
II	1	6	3	5	4	2	1	660
III	1	6	3	2	5	4	1	870
IV	1	2	3	5	6	4	1	780
V	1	6	4	5	2	3	1	750
VI	1	6	5	3	2	4	1	670
VII	1	3	6	5	2	4	1	910
IIX	1	6	3	4	2	5	1	770
IX	1	4	3	5	2	6	1	830
X	1	5	3	6	2	4	1	880

t=2	2	3	4	5	6
2			5		
3					4
4					
5					
6					

Η αντιμετάθεση 3↔6 είναι «απαγορευμένη».

Επιλέγεται η βέλτιστη λύση (II) που αντιστοιχεί στην αντιμετάθεση 2↔4.

Στον ανωτέρω πίνακα καταγράφονται «απαγορευμένες» αλλαγές για τις επόμενες φάσεις της διαδικασίας.

TSP - Tabu Search (t=3)

	A	C	F	B	E	D	A	
	1	6	3	5	4	2	1	660
I	1	6	2	5	4	3	1	840
II	1	6	3	5	2	4	1	760
III	1	6	3	2	4	5	1	730
IV	1	2	3	5	4	6	1	640
V	1	6	4	5	3	2	1	640
VI	1	6	5	3	4	2	1	610
VII	1	3	6	5	4	2	1	810
IIX	1	6	3	4	5	2	1	810
IX	1	4	3	5	6	2	1	870
X	1	5	3	6	4	2	1	730

	2	3	4	5	6	
			4			2
				5	3	3
						4
t=3						5
						6

Όπως προηγουμένως.

TSP - Tabu Search (t=4)

	A	C	F	B	E	D	A	
	1	6	5	3	4	2	1	610
I	1	6	5	2	4	3	1	720
II	1	6	5	3	2	4	1	670
III	1	6	2	3	4	5	1	800
IV	1	2	5	3	4	6	1	680
V	1	6	5	4	3	2	1	720
VI	1	6	3	5	4	2	1	660
VII	1	3	5	6	4	2	1	680
IIX	1	6	4	3	5	2	1	680
IX	1	4	5	3	6	2	1	920
X	1	5	6	3	4	2	1	810

	2	3	4	5	6	
			3		5	2
				4	2	3
						4
t=4						5
						6

Όπως προηγουμένως.

Επιλέγεται η βέλτιστη λύση (IV) που αντιστοιχεί στην αντιμετάθεση 2↔6, καίτοι με χαμηλότερη καταλληλότητα από την τρέχουσα λύση (680>610).

Η συνολικά βέλτιστη λύση ως τώρα (610) διατηρείται στη μνήμη (και θα είναι η τελική λύση αν δεν προκύψει κάποια καλύτερη αργότερα).

TSP - Tabu Search (t=5)

	A	C	F	B	E	D	A	
	1	2	5	3	4	6	1	680
I	1	3	5	2	4	6	1	640
II	1	4	5	3	2	6	1	790
III	1	5	2	3	4	6	1	750
IV	1	6	5	3	4	2	1	610
V	1	2	5	4	3	6	1	810
VI	1	2	3	5	4	6	1	640
VII	1	2	5	6	4	3	1	830
IIX	1	2	4	3	5	6	1	610
IX	1	2	5	3	6	4	1	870
X	1	2	6	3	4	5	1	930

	2	3	4	5	6	
			2		4	2
				3	1	3
				5		4
t=5						5
						6

Όπως προηγουμένως.

Η βέλτιστη λύση που προκύπτει εδώ (τυχαίνει να) είναι η ίδια (αλλά με αντίθετη φορά επίσκεψης πόλεων) με τη λύση σε t=3.

TSP - Tabu Search (t=6)

	A	C	F	B	E	D	A	
	1	2	4	3	5	6	1	610
I	1	3	4	2	5	6	1	720
II	1	4	2	3	5	6	1	670
III	1	5	4	3	2	6	1	800
IV	1	6	4	3	5	2	1	680
V	1	2	3	4	5	6	1	720
VI	1	2	4	5	3	6	1	660
VII	1	2	4	6	5	3	1	680
IIX	1	2	5	3	4	6	1	680
IX	1	2	6	3	5	4	1	920
X	1	2	4	3	6	5	1	810

	2	3	4	5	6
2			1		3
3				2	5
4				4	
5					
6					

t=6

Σημαντική σημείωση:

Αν στο διπλανό πίνακα προέκυπτε λύση καλύτερη από 610 και η οποία αντιστοιχούσε σε «απαγορευμένη» αντιμετάθεση πόλεων, επιλέγεται η νέα λύση καθ' υπέρβαση του κριτηρίου της απαγόρευσης.

TSP - Tabu Search (t=7)

	A	C	F	B	E	D	A	
	1	**2**	**4**	**6**	**5**	**3**	**1**	**680**
I	1	3	4	6	5	2	1	830
II	1	4	2	6	5	3	1	830
III	1	5	4	6	2	3	1	870
IV	1	6	4	2	5	3	1	640
V	1	2	3	6	5	4	1	910
VI	1	2	4	6	3	5	1	730
VII	1	2	4	3	5	6	1	610
IIX	1	2	5	6	4	3	1	830
IX	1	2	6	4	5	3	1	**800**
X	1	2	4	5	6	3	1	810

	2	3	4	5	6	
			0		2	**2**
				1	4	**3**
				3	5	**4**
t=7						**5**
						6

TSP - Tabu Search (t=8)

	A	C	F	B	E	D	A	
	1	2	6	4	5	3	1	800
I	1	3	6	4	5	2	1	880
II	1	4	6	2	5	3	1	900
III	1	5	6	4	2	3	1	750
IV	1	6	2	4	5	3	1	690
V	1	2	6	3	5	4	1	920
VI	1	2	6	4	3	5	1	800
VII	1	2	3	4	5	6	1	720
IIX	1	2	6	5	4	3	1	880
IX	1	2	4	6	5	3	1	680
X	1	2	5	4	6	3	1	880

t=8

	2	3	4	5	6
2			0	5	1
3				0	3
4				2	4
5					
6					

TSP - Tabu Search (t=9)

	A	C	F	B	E	D	A	
	1	**5**	**6**	**4**	**2**	**3**	**1**	**750**
I	1	5	6	4	3	2	1	790
II	1	5	6	2	4	3	1	840
III	1	5	6	4	2	3	1	750
IV	1	5	2	4	6	3	1	840
V	1	5	6	3	2	4	1	870
VI	1	3	6	4	2	5	1	840
VII	1	5	6	4	2	3	1	750
IIX	1	4	6	5	2	3	1	890
IX	1	5	4	6	2	3	1	870
X	1	6	5	4	2	3	1	**680**

t=9

	2	3	4	5	6
2			0	4	0
3				0	2
4				1	3
5					5
6					

TSP - Tabu Search (t=10)

	A	C	F	B	E	D	A	
	1	6	5	4	2	3	1	680
I	1	6	5	4	3	2	1	720
II	1	6	5	2	4	3	1	750
III	1	6	2	4	5	3	1	690
IV	1	2	5	4	6	3	1	880
V	1	6	5	3	2	4	1	670
VI	1	6	3	4	2	5	1	770
VII	1	3	5	4	2	6	1	690
IIX	1	6	4	5	2	3	1	750
IX	1	4	5	6	2	3	1	940
X	1	5	6	4	2	3	1	750

t=10

	2	3	4	5	6
2			0	3	0
3			5	0	1
4				0	2
5					4
6					

TSP - Tabu Search (t=11)

A	C	F	B	E	D	A	
1	6	5	3	2	4	1	670

I	1	6	5	2	3	4	1	820
II	1	6	5	3	4	2	1	610
III	1	6	2	3	5	4	1	790
IV	1	2	5	3	6	4	1	870
V	1	6	5	4	2	3	1	680
VI	1	6	3	5	2	4	1	760
VII	1	3	5	6	2	4	1	830
IIX	1	6	4	3	2	5	1	750
IX	1	4	5	3	2	6	1	790
X	1	5	6	3	2	4	1	870

t=11

	2	3	4	5	6
2			5	2	0
3			4	0	0
4				0	1
5					3
6					

TSP - Tabu Search (t=12)

	A	C	F	B	E	D	A	
	1	**6**	**5**	**3**	**4**	**2**	**1**	**610**
I	1	6	5	2	4	3	1	720
II	1	6	5	3	2	4	1	670
III	1	6	2	3	4	5	1	800
IV	1	2	5	3	4	6	1	680
V	1	6	5	4	3	2	1	720
VI	1	6	3	5	4	2	1	**660**
VII	1	3	5	6	4	2	1	680
IIX	1	6	4	3	5	2	1	680
IX	1	4	5	3	6	2	1	920
X	1	5	6	3	4	2	1	810

	2	3	4	5	6	
			4	1	0	**2**
			3	5	0	**3**
				0	0	**4**
t=12					2	**5**
						6

TSP - Tabu Search (t=13)

	A	C	F	B	E	D	A	
	1	**6**	**3**	**5**	**4**	**2**	**1**	**660**
I	1	6	2	5	4	3	1	840
II	1	6	3	5	2	4	1	760
III	1	6	3	2	4	5	1	730
IV	1	2	3	5	4	6	1	**640**
V	1	6	4	5	3	2	1	640
VI	1	6	5	3	4	2	1	610
VII	1	3	6	5	4	2	1	810
IIX	1	6	3	4	5	2	1	810
IX	1	4	3	5	6	2	1	870
X	1	5	3	6	4	2	1	730

	2	3	4	5	6	
			3	0	5	2
			2	4	0	3
				0	0	4
t=13					1	5
						6

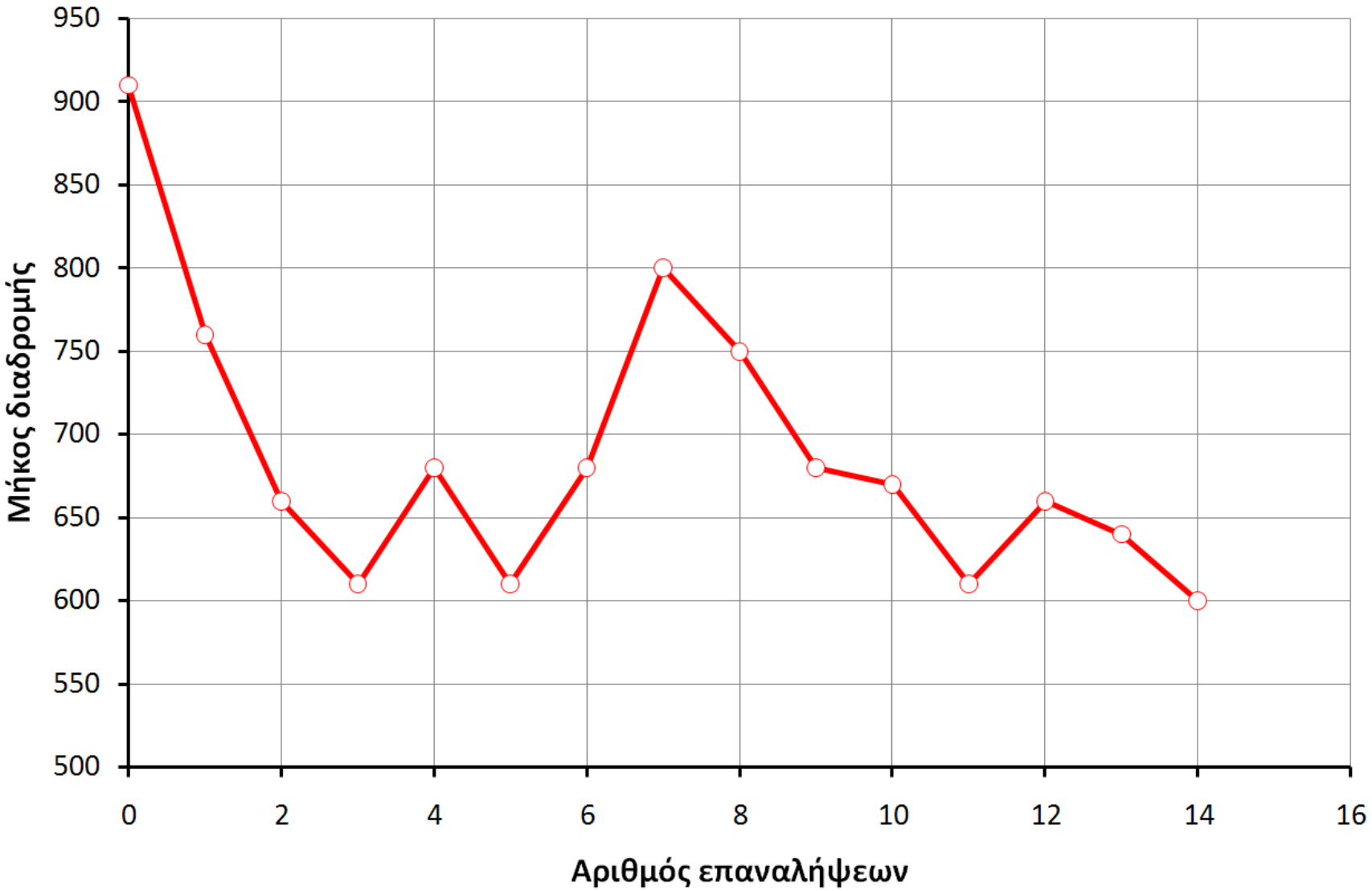
TSP - Tabu Search (t=14)

	A	C	F	B	E	D	A	
	1	2	3	5	4	6	1	640
I	1	3	2	5	4	6	1	750
II	1	4	3	5	2	6	1	830
III	1	5	3	2	4	6	1	600
IV	1	6	3	5	4	2	1	660
V	1	2	4	5	3	6	1	660
VI	1	2	5	3	4	6	1	680
VII	1	2	6	5	4	3	1	880
IIX	1	2	3	4	5	6	1	720
IX	1	2	3	5	6	4	1	780
X	1	2	3	6	4	5	1	840

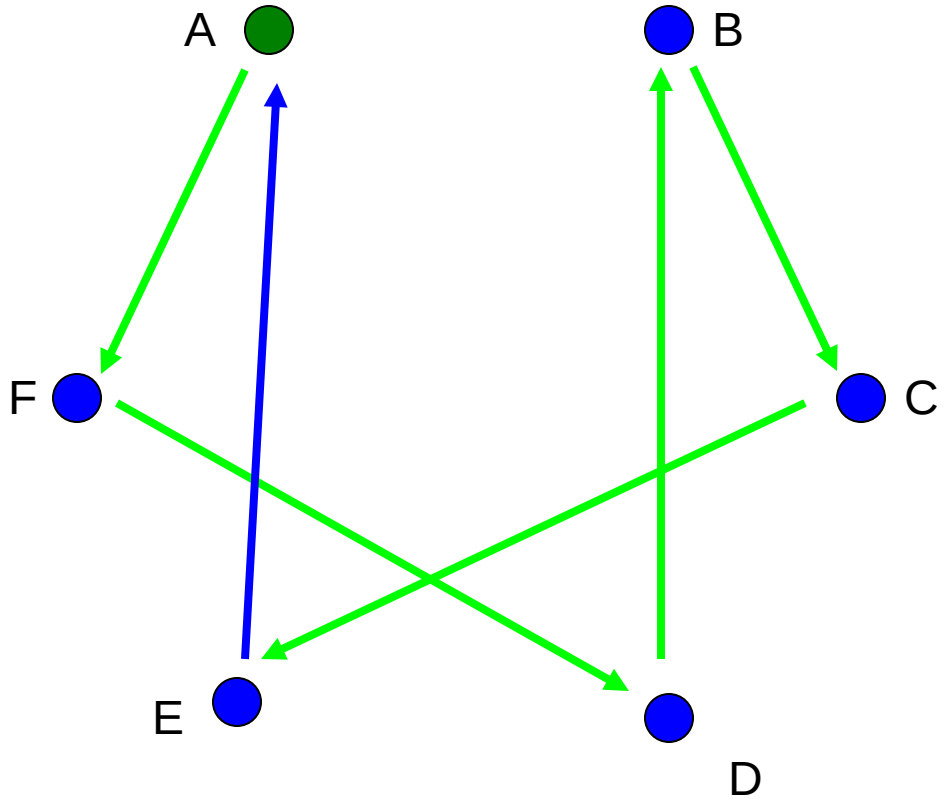
t=14

	2	3	4	5	6
2			2	5	4
3			1	3	0
4				0	0
5					0
6					

PSO - Tabu Search



Βέλτιστη Διαδρομή



A	F	D	B	C	E	A	
	60	80	60	150	90	160	**600**

Αναζήτηση με απαγορευμένες καταστάσεις

Όπως προαναφέρθηκε, η «απαγορευμένη» μνήμη μπορεί να παραβιαστεί (επιλέγοντας «απαγορευμένες» καταστάσεις) στην περίπτωση που εμφανιστεί πολύ καλό αποτέλεσμα (καλύτερο από την έως του σημείου βέλτιστη λύση) εφαρμόζοντας το κριτήριο υπέρβασης (aspiration criterion).

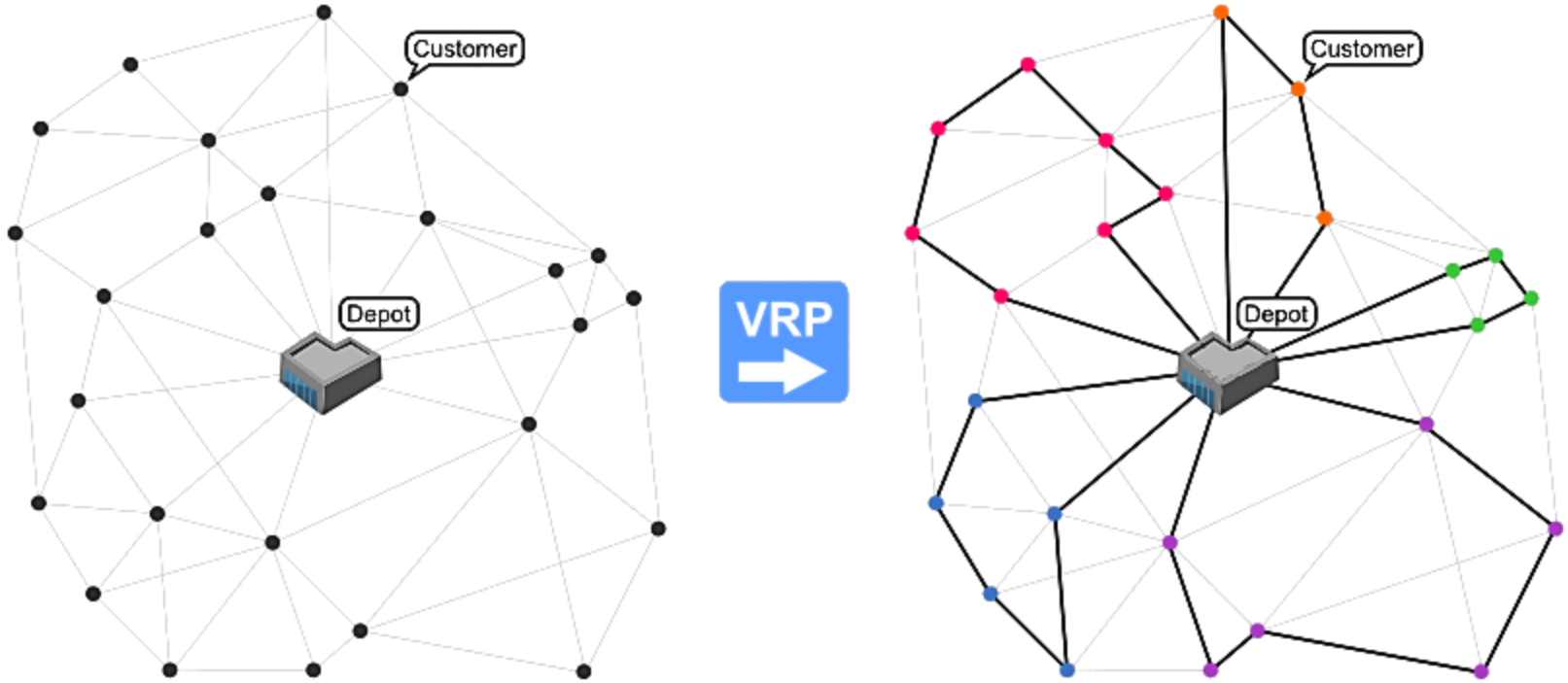
Πίνακας συχνότητας εναλλαγών

	2	3	4	5	6	
		2	1	2	1	2
			1	2	2	3
				2	1	4
t=...					1	5
						6

Πίνακας μνήμης παλαιότητας

	2	3	4	5	6	
		2		3	1	2
			1	3	2	3
				3	2	4
t=...					1	5
						6

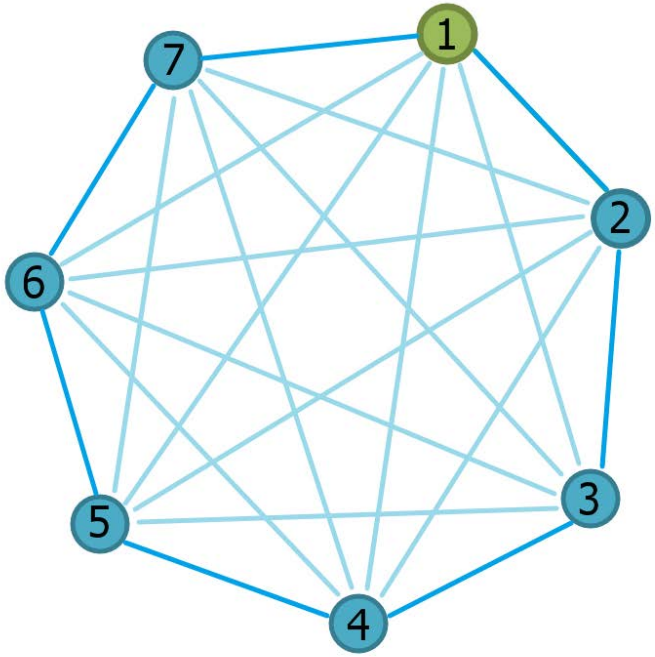
το Πρόβλημα Δρομολόγησης Οχημάτων



Το πρόβλημα Δρομολόγησης Οχημάτων επιδιώκει την εξυπηρέτηση ενός αριθμού πελατών (κομβόι) με εμπορεύματα (ποσότητα), χρησιμοποιώντας ένα αριθμό οχημάτων συγκεκριμένης χωρητικότητας και διανύοντας τη μικρότερη απόσταση (για το σύνολο των οχημάτων) κατά την εξυπηρέτηση των πελατών.

το Πρόβλημα Δρομολόγησης Οχημάτων

Δίκτυο πόλεων



Χωρητικότητα οχημάτων

25

25

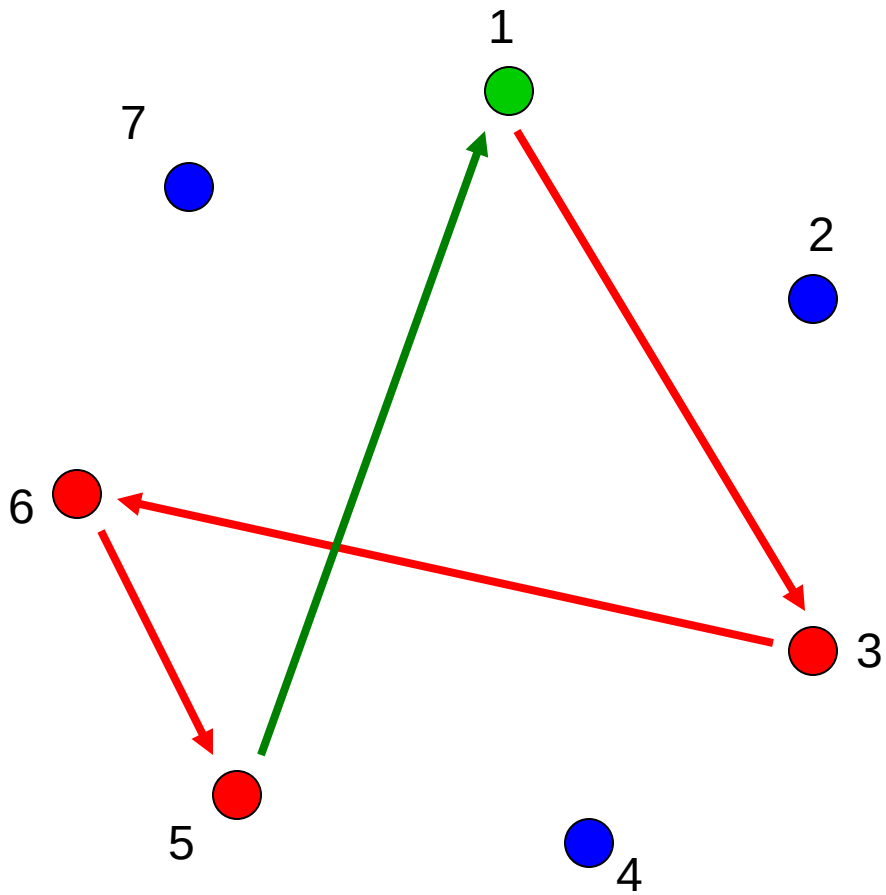
Ζήτηση πόλεων

	2	3	4	5	6	7
45	6	5	7	8	10	9

Αποστάσεις πόλεων

	1	2	3	4	5	6	7
1	--	26	17	26	22	10	22
2	26	--	12	17	11	21	19
3	17	12	--	23	27	25	18
4	26	17	23	--	23	19	27
5	22	11	27	23	--	29	21
6	10	21	25	19	29	--	20
7	22	19	18	27	21	20	--

Παράδειγμα - Vehicle Routing Problem VRP



A

Ζήτηση πόλεων

	2	3	4	5	6	7
45	6	5	7	8	10	9

Αποστάσεις πόλεων

	1	2	3	4	5	6	7
1	--	26	17	26	22	10	22
2	26	--	12	17	11	21	19
3	17	12	--	23	27	25	18
4	26	17	23	--	23	19	27
5	22	11	27	23	--	29	21
6	10	21	25	19	29	--	20
7	22	19	18	27	21	20	--

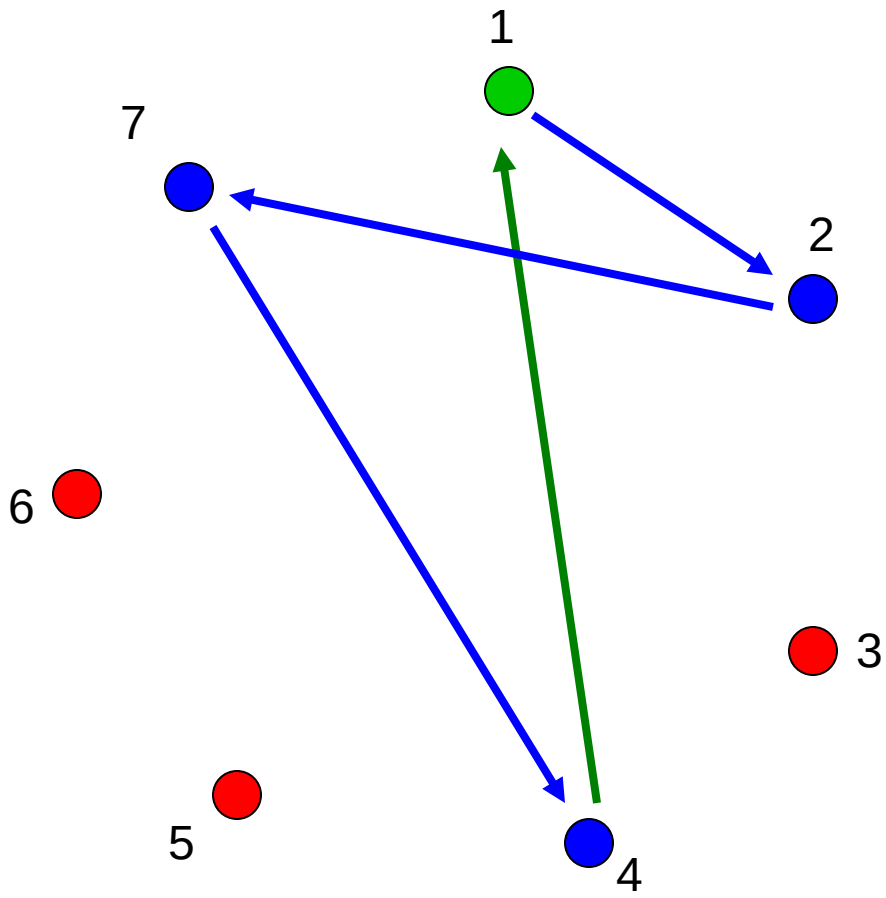
L	1	3	6	5	1
D	93	17	25	29	22
Z	23	5	10	8	

L: διαδρομή

D: απόσταση

Z: ζήτηση

Παράδειγμα - Vehicle Routing Problem VRP



B

Ζήτηση πόλεων

	2	3	4	5	6	7
45	6	5	7	8	10	9

Αποστάσεις πόλεων

	1	2	3	4	5	6	7
1	--	26	17	26	22	10	22
2	26	--	12	17	11	21	19
3	17	12	--	23	27	25	18
4	26	17	23	--	23	19	27
5	22	11	27	23	--	29	21
6	10	21	25	19	29	--	20
7	22	19	18	27	21	20	--

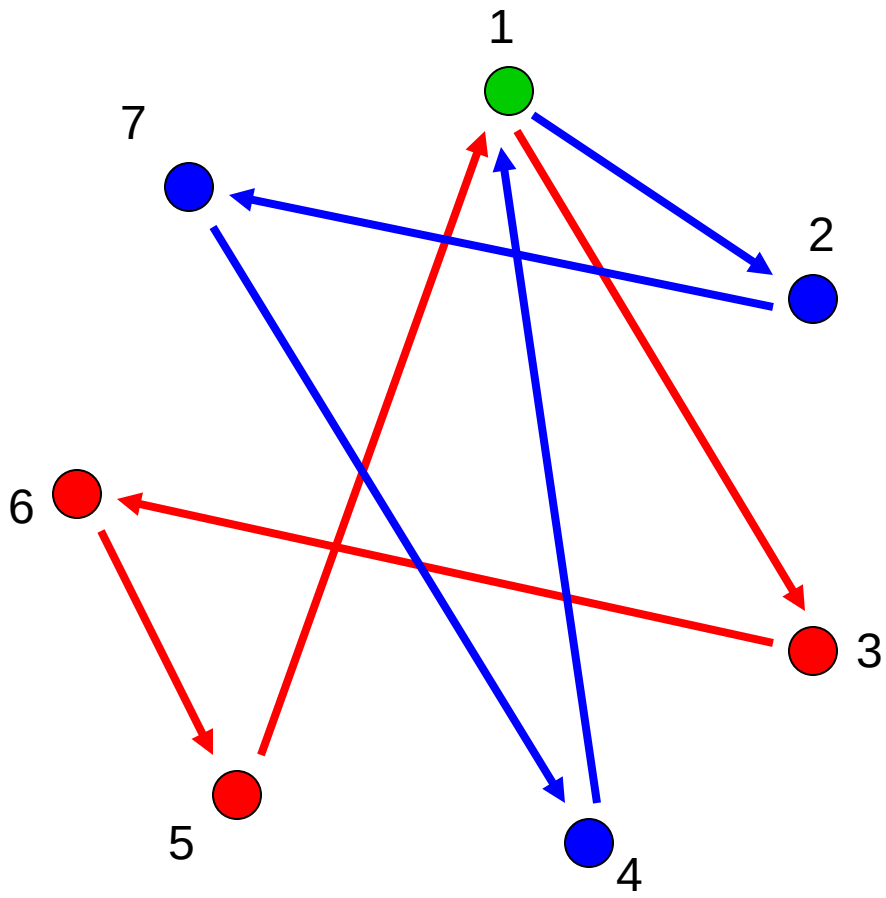
L: διαδρομή

D: απόσταση

Z: ζήτηση

L	1	2	7	4	1
D	98	26	19	27	26
Z	22	6	9	7	

Παράδειγμα - Vehicle Routing Problem VRP



Ζήτηση πόλεων

	2	3	4	5	6	7
45	6	5	7	8	10	9

Αποστάσεις πόλεων

	1	2	3	4	5	6	7
1	--	26	17	26	22	10	22
2	26	--	12	17	11	21	19
3	17	12	--	23	27	25	18
4	26	17	23	--	23	19	27
5	22	11	27	23	--	29	21
6	10	21	25	19	29	--	20
7	22	19	18	27	21	20	--

L	1	3	6	5	1
D	93	17	25	29	22
Z	23	5	10	8	

L	1	2	7	4	1
D	98	26	19	27	26
Z	22	6	9	7	

VRP - Tabu Search (t=1)

	1	3	6	5	1	23	93	1	2	7	4	1	22	98	45	191
I	1	2	6	5	1	24	98	1	3	7	4	1	21	88	45	186
II	1	3	6	5	1	23	93	1	4	7	2	1	22	98	45	191
III	1	3	6	2	1	21	89	1	5	7	4	1	24	96	45	185
IV	1	3	2	5	1	19	62	1	6	7	4	1	26	83		
V	1	3	6	5	1	23	93	1	7	2	4	1	22	84	45	177
VI	1	4	6	5	1	25	96	1	2	7	3	1	20	90	45	186
VII	1	5	6	3	1	23	103	1	2	7	4	1	22	98	45	201
IIX	1	6	3	5	1	23	84	1	2	7	4	1	22	98	45	182
IX	1	7	6	5	1	27	93	1	2	3	4	1	18	87		
X	1	3	6	4	1	22	87	1	2	7	5	1	23	88	45	175
XI	1	3	4	5	1	20	85	1	2	7	6	1	25	75	45	160
XII	1	3	6	5	1	23	93	1	2	4	7	1	22	92	45	185
XIII	1	3	5	6	1	23	83	1	2	7	4	1	22	98	45	181
XIV	1	3	6	7	1	24	84	1	2	5	4	1	21	86	45	170
XV	1	3	7	5	1	22	78	1	2	6	4	1	23	92	45	170

t=1

	2	3	4	5	6	7
2					5	
3						5
4					5	
5						
6						
7						

Σε κόκκινο φόντο εμφανίζονται μη εφικτές λύσεις (υπέρβαση της χωρητικότητας του οχήματος).

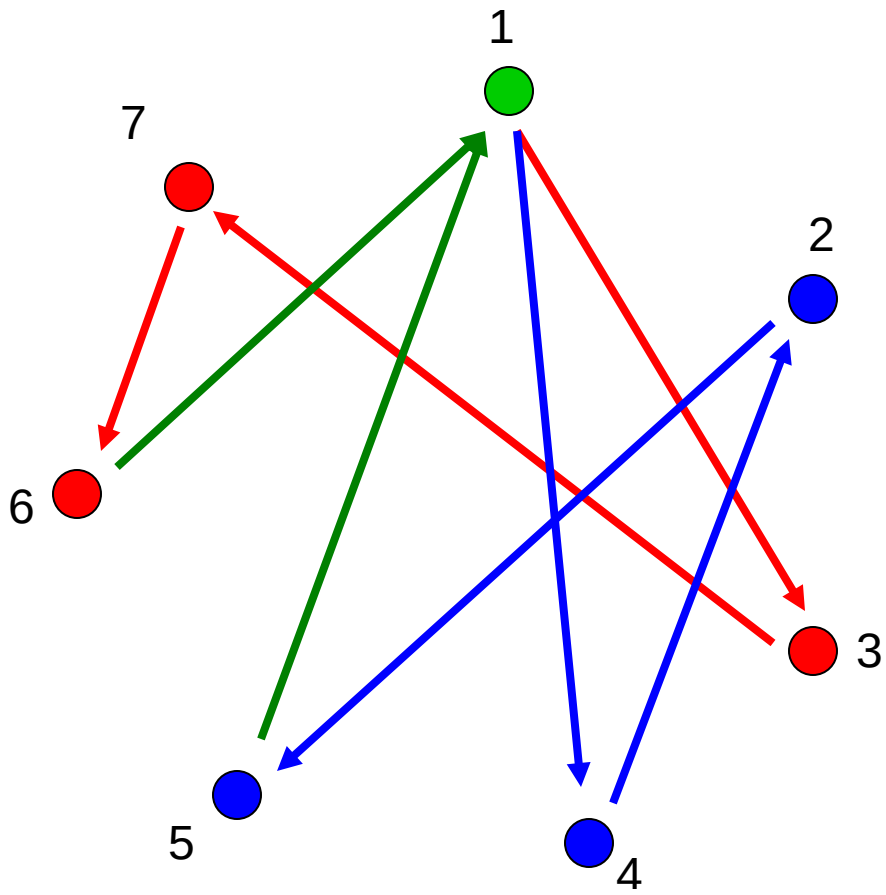
VRP - Tabu Search (t=2)

	1	3	4	5	1	20	85	1	2	7	6	1	25	75	45	160
I	1	2	4	5	1	21	88	1	3	7	6	1	24	65	45	153
II	1	3	2	5	1	19	62	1	4	7	6	1	26	83		
III	1	3	4	2	1	18	83	1	5	7	6	1	27	73		
IV	1	3	4	5	1	20	85	1	6	7	2	1	25	75	45	160
V	1	3	4	5	1	20	85	1	7	2	6	1	25	72	45	157
VI	1	4	3	5	1	20	98	1	2	7	6	1	25	75	45	173
VII	1	5	4	3	1	20	95	1	2	7	6	1	25	75	45	170
IIX	1	6	4	5	1	25	74	1	2	7	3	1	20	90	45	164
IX	1	7	4	5	1	24	94	1	2	3	6	1	21	73	45	167
X	1	3	5	4	1	20	93	1	2	7	6	1	25	75	45	168
XI	1	3	6	5	1	23	93	1	2	7	4	1	22	98	45	191
XII	1	3	7	5	1	22	78	1	2	4	6	1	23	72	45	150
XIII	1	3	4	6	1	22	69	1	2	7	5	1	23	88	45	157
XIV	1	3	4	7	1	21	89	1	2	5	6	1	24	76	45	165
XV	1	3	4	5	1	20	85	1	2	6	7	1	25	89	45	174

t=2

	2	3	4	5	6	7
2					4	
3						4
4					4	5
5						
6						
7						

Βέλτιστο Δρομολόγιο



A,B

	2	3	4	5	6	7
45	6	5	7	8	10	9

L	1	3	7	6	1
D	65	17	18	20	10
Z	24	5	9	10	

L	1	4	2	5	1
D	76	26	17	11	22
Z	21	7	6	8	

	1	2	3	4	5	6	7
1	--	26	17	26	22	10	22
2	26	--	12	17	11	21	19
3	17	12	--	23	27	25	18
4	26	17	23	--	23	19	27
5	22	11	27	23	--	29	21
6	10	21	25	19	29	--	20
7	22	19	18	27	21	20	--

Παράδειγμα: Βελτιστοποίηση συνάρτησης (t=0)

$$\min C(x) = 20X_1 + 25X_2 - 30X_3 - 45X_4 + 40X_5$$

Περιορισμοί:

$$X_1 + X_2 - X_3 + X_4 + X_5 \geq 1$$ Ποινή 70 (για υπέρβαση του)

$$X_1 + X_2 - X_4 + 2X_5 \geq 2$$ Ποινή 70

$$-X_2 + X_4 + X_5 \leq 1$$ Ποινή 100

$$X_2 + X_3 + X_5 \leq 2$$ Ποινή 100

Τιμές μεταβλητών: X_1, X_2, X_3, X_4, X_5, {0 ή 1} (δυαδική)

Έστω μια αρχική λύση {X_1, X_2, X_3, X_4, X_5,} → {1, 0, 0, 0, 0, 1} → C = 60

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	0	0	0	1	60	2	3	1	1
					60	0	0	0	0

Βελτιστοποίηση συνάρτησης (t=1)

<u>Τοπική αναζήτηση</u>

(αλλάζει κάθε φορά μια μόνο μεταβλητή)

Στα κουτάκια δεξιά:
Σε γαλάζιο φόντο οι τιμές των περιορισμών.
Σε λευκό φόντο η ποινή (αν υπάρχει υπέρβαση περιορισμού).

Στο μέσο:
Σε λευκό / κίτρινο φόντο η τιμή της αντικειμενικής συνάρτησης χωρίς / με ποινή αντίστοιχα.

X1=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	0	0	1	40	1	2	1	1
					40	0	0	0	0

X2=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	1	0	0	1	85	3	4	0	2
					85	0	0	0	0

X3=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	0	1	0	1	30	1	3	1	2
					30	0	0	0	0

X4=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	0	0	1	1	15	3	2	2	1
					115	0	0	100	0

X5=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	0	0	0	0	20	1	1	0	0
					90	0	70	0	0

Βελτιστοποίηση συνάρτησης (t=1)

Η καλύτερη λύση

X3=1

X1	X2	X3	X4	X5
1	0	1	0	1

C
30
30

Π1	Π2	Π3	Π4
1	3	1	2
0	0	0	0

Λίστα μνήμης : έχει ήδη τοποθετηθεί η προηγούμενη λύση (t=0)

{1, 0, 0, 0, 1} → C = 60

Η τρέχουσα λύση δεν είναι στην μνήμη οπότε την αποδεχόμαστε ως νέα λύση.

X1	X2	X3	X4	X5
1	0	1	0	1

C
30
30

Π1	Π2	Π3	Π4
1	3	1	2
0	0	0	0

Βελτιστοποίηση συνάρτησης (t=2)

Τοπική αναζήτηση

X1=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	1	0	1	10	0	2	1	2
					80	70	0	0	0

X2=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	1	1	0	1	55	2	4	0	3
					155	0	0	0	100

X3=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	0	0	0	1	60	2	3	1	1
					60	0	0	0	0

X4=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	0	1	1	1	-15	2	2	2	2
					85	0	0	100	0

X5=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	0	1	0	0	-10	0	1	0	1
					130	70	70	0	0

Το κόκκινο φόντο υποδεικνύει ότι η λύση έχει ήδη αξιολογηθεί (είναι στη μνήμη).

Βελτιστοποίηση συνάρτησης (t=2)

Η καλύτερη λύση

X1=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	1	0	1	10	0	2	1	2
					80	70	0	0	0

Λίστα μνήμης : **{1, 0, 0, 0, 1} → C = 60**
{1, 0, 1, 0, 1} → C = 30

Η τρέχουσα λύση δεν είναι στην μνήμη οπότε την αποδεχόμαστε ως νέα λύση.

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	1	0	1	10	0	2	1	2
					80	70	0	0	0

Βελτιστοποίηση συνάρτησης (t=3)

Τοπική αναζήτηση

Το κόκκινο φόντο υποδεικνύει ότι η λύση έχει ήδη αξιολογηθεί (είναι στη μνήμη).

X1=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	0	1	0	1	30	1	3	1	2
					30	0	0	0	0

X2=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	1	1	0	1	35	1	3	0	3
					135	0	0	0	100

X3=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	0	0	1	40	1	2	1	1
					40	0	0	0	0

X4=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	1	1	1	-35	1	1	2	2
					135	0	70	100	0

X5=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	1	0	0	-30	-1	0	0	1
					110	70	70	0	0

Βελτιστοποίηση συνάρτησης (t=3)

Η καλύτερη λύση

X3=0

X1	X2	X3	X4	X5
0	0	0	0	1

C
40
40

Π1	Π2	Π3	Π4
1	2	1	1
0	0	0	0

Λίστα μνήμης : **{1, 0, 0, 0, 1} → C = 60**
{1, 0, 1, 0, 1} → C = 30
{0, 0, 1, 0, 1} → C = 80

Η τρέχουσα λύση δεν είναι στην μνήμη οπότε την αποδεχόμαστε ως νέα λύση.

X1	X2	X3	X4	X5
0	0	0	0	1

C
40
40

Π1	Π2	Π3	Π4
1	2	1	1
0	0	0	0

Βελτιστοποίηση συνάρτησης (t=4)

Τοπική αναζήτηση

X1=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	0	0	0	1	60	2	3	1	1
					60	0	0	0	0

X2=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	1	0	0	1	65	2	3	0	2
					65	0	0	0	0

X3=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	1	0	1	10	0	2	1	2
					80	70	0	0	0

X4=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	0	1	1	-5	2	1	2	1
					165	0	70	100	0

X5=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	0	0	0	0	0	0	0	0
					140	70	70	0	0

Βελτιστοποίηση συνάρτησης (t=4)

Η καλύτερη λύση

X2=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	1	0	0	1	65	2	3	0	2
					65	0	0	0	0

Λίστα μνήμης : **{1, 0, 0, 0, 1} → C = 60**
{1, 0, 1, 0, 1} → C = 30
{0, 0, 1, 0, 1} → C = 80
{0, 0, 0, 0, 1} → C = 40

Η τρέχουσα λύση δεν είναι στην μνήμη οπότε την αποδεχόμαστε ως νέα λύση.

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	1	0	0	1	65	2	3	0	2
					65	0	0	0	0

Βελτιστοποίηση συνάρτησης (t=5)

Τοπική αναζήτηση

X1=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
1	1	0	0	1	85	3	4	0	2
					85	0	0	0	0

X2=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	0	0	0	1	40	1	2	1	1
					40	0	0	0	0

X3=1

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	1	1	0	1	35	1	3	0	3
					135	0	0	0	100

X4=1

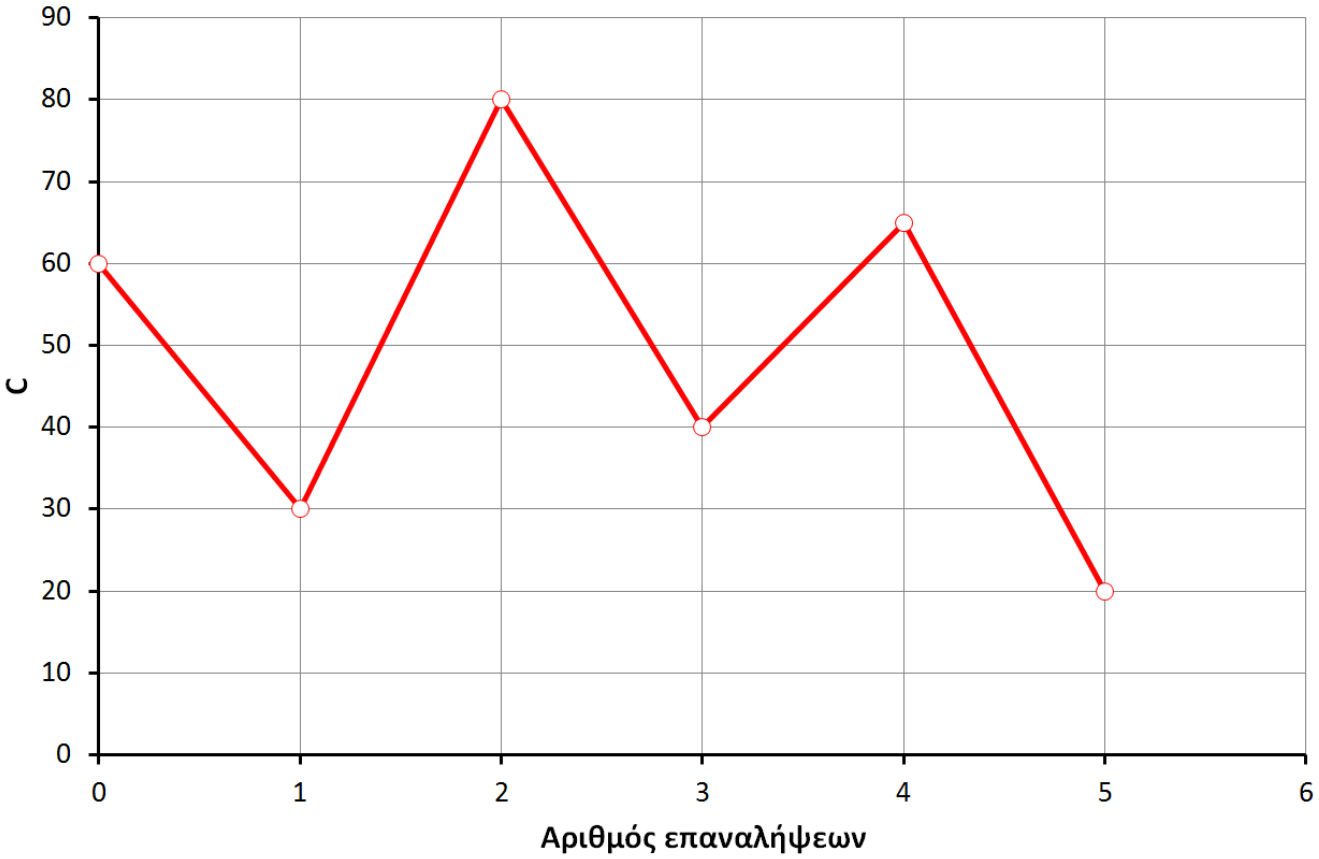
X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	1	0	1	1	20	3	2	1	2
					20	0	0	0	0

X5=0

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	1	0	0	0	25	1	1	-1	1
					95	0	70	0	0

Βελτιστοποίηση συνάρτησης – Βέλτιστη λύση

X1	X2	X3	X4	X5	C	Π1	Π2	Π3	Π4
0	1	0	1	1	20	3	2	1	2
					20	0	0	0	0



Αναζήτηση με απαγορευμένες καταστάσεις

Η Tabu Search βασίζεται σε μια σειρά επαναλήψεων όπου, σε κάθε στάδιο, υπάρχει μία τρέχουσα λύση υπό επεξεργασία, στη γειτονιά της οποίας καθορίζονται και αξιολογούνται γειτονικές λύσεις (τοπική αναζήτηση), αποφεύγοντας (για κάποιο διάστημα της διαδικασίας) επανάληψη της ίδιας μετακίνησης που επέφερε τη νέα λύση («απαγορευμένες» λύσεις).

Αλγόριθμος αναζήτησης με απαγορευμένες κατάστασης:

1. Επιλέγεται μια αρχική (τρέχουσα) λύση.
2. Η τρέχουσα λύση καθορίζει μια γειτονιά από τοπικές λύσεις.
3. Επιλέγεται η βέλτιστη από τις τοπικές λύσεις (νέα τρέχουσα λύση).
4. Χαρακτηριστικά της βέλτιστης λύσης τοποθετούνται στον πίνακα μνήμης.
5. Η διαδικασία επαναλαμβάνεται από τη νέα τρέχουσα λύση μέχρι να ικανοποιηθεί το κριτήριο τερματισμού της διαδικασίας.

Simulated Annealing και Tabu Search

Η προσομοιωμένη ανόπτηση (simulated annealing) και η αναζήτηση απαγορευμένων καταστάσεων (tabu search) έχουν σχεδιαστεί με κύριο στόχο τον απεγκλωβισμό από τοπικά βέλτιστα. Προς το σκοπό αυτό, και οι δύο κάνουν δεκτές και λύσεις μειωμένης καταλληλότητας σε σχέση με προηγούμενες λύσεις.

Ο αλγόριθμος αναζήτησης απαγορευμένων καταστάσεων χρησιμοποιείται συχνά σε συνδυασμό με άλλες μεθόδους για να βελτιώσει την ικανότητα τους στην τοπική αναζήτηση βέλτιστων λύσεων.

Τέλος παρουσίασης