

Advanced School in Artificial Intelligence

Constraint Processing: Routing

Marco Gavanelli

marco.gavanelli@unife.it



Progetto di alta formazione in ambito tecnologico economico e culturale per una regione della conoscenza europea e attrattiva approvato e cofinanziato dalla Regione Emilia-Romagna con deliberazione di Giunta regionale n. 1625/2021



**Università
degli Studi
di Ferrara**

ROUTING: problem definition

- Routing concerns the problem of finding a set of routes to be covered by a set of vehicles visiting a set of cities/customers once starting and ending at one (n) depot(s).

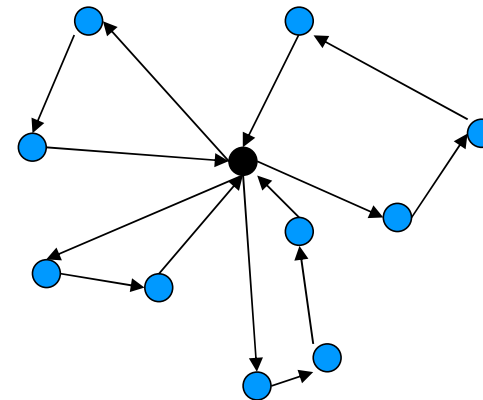
- Constraints

- temporal restrictions:
 - time windows
 - maximal duration of a travel
- vehicle capacity
- customer demands

- Optimization Criteria

- number of vehicles
- travel cost
- travel time

- Basic component: [Travelling Salesperson Problem](#) (TSP) and its variants.



Travelling Salesperson Problem

- Dati un insieme di **N** città
- `int :N;`
- `enum citta;`
- Una matrice di distanze
- `array [citta,citta] of int : distanza;`
- calcolare il percorso che visita ogni città esattamente una volta, percorrendo distanza minima

```
N=7;
```

```
citta = {Milano, Genova,  
Ferrara, Firenze, Perugia,  
Bari, Roma};
```

```
distanza =
```

```
[| 0 ,144,254,314,449,879,574  
|144, 0 ,339,230,383,910,505  
|254,343, 0 ,154,280,713,420  
|314,230,154, 0 ,151,678,273  
|449,383,280,151, 0 ,557,171  
|879,910,713,678,557, 0 ,432  
|574,505,420,273,171,432, 0 |];
```

```
citta.dzn
```

TSP: Permutation representation

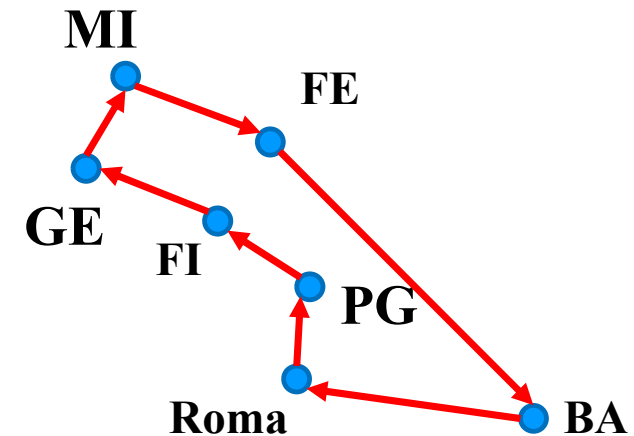
- Variabili: $P[i]$ = città visitata nella i -esima posizione

- Domini: insieme delle città

`array [1..N] of var città:P;`

- Constraints:

- **alldifferent**(P)



`solve minimize sum([distanza[P[i],P[i+1]]|i in 1..N-1])`

`+ distanza[P[N],P[1]];`

P	GE	MI	FE	BA	Roma	PG	FI
	1	2	3	4	5	6	7

TSP: successor representation

Variabili: $S[c]$ = città visitata dopo la città c

- Domini: insieme delle città

`array [città] of var città:S;`

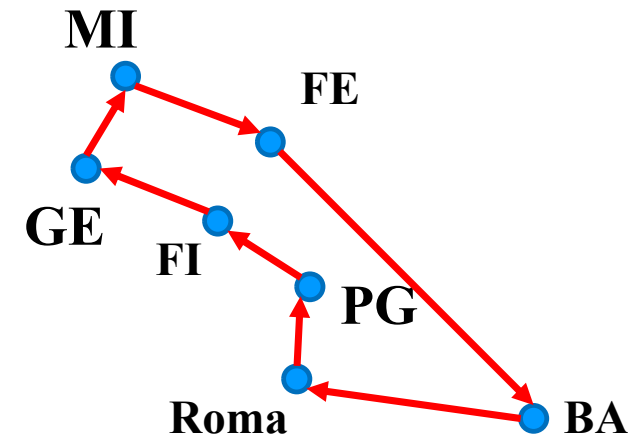
- Constraints:

- `alldifferent(S)`
- $S[c] \neq c$
- `circuit(N)`

solve minimize

`sum([distanza[i, S[i]] | i in città]);`

S	FE	MI	BA	GE	FI	Roma	PG
	MI	GE	FE	FI	PG	BA	Roma




TSP: successor representation

Variabili: $S[c]$ = città visitata dopo la città c

- Domini: insieme delle città

`array [citta] of var citta:S;`

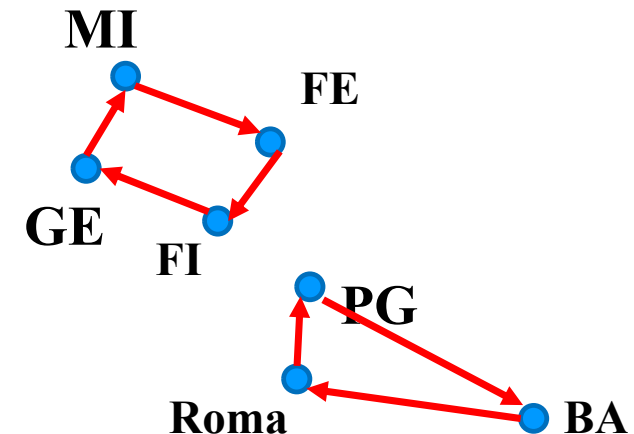
- Constraints:

- alldifferent(S)
- $S[c] \neq c$
- ~~circuit(N)~~ 

solve minimize

sum([distanza[i, S[i]] | i in citta]);

S	FE	MI	BA	GE	FI	Roma	PG
	MI	GE	FE	FI	PG	BA	Roma



TSP: successor representation

Variabili: $S[c]$ = città visitata dopo la città c

- Domini: insieme delle città

array [città] of var
città:S;

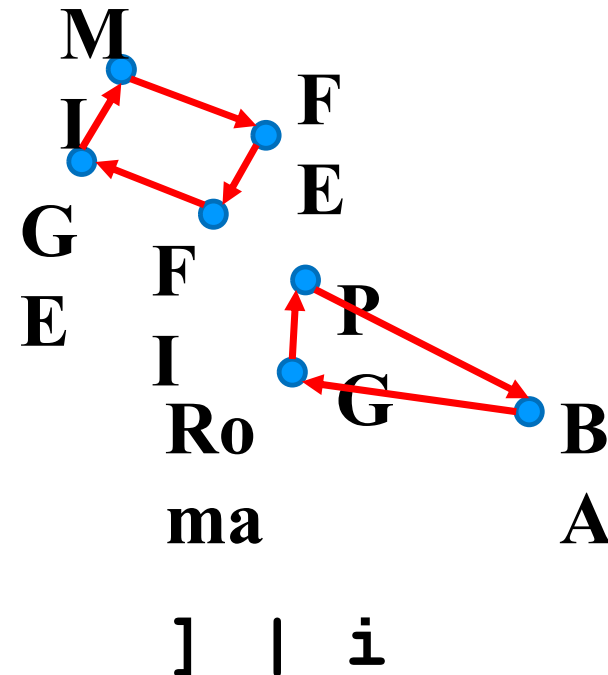
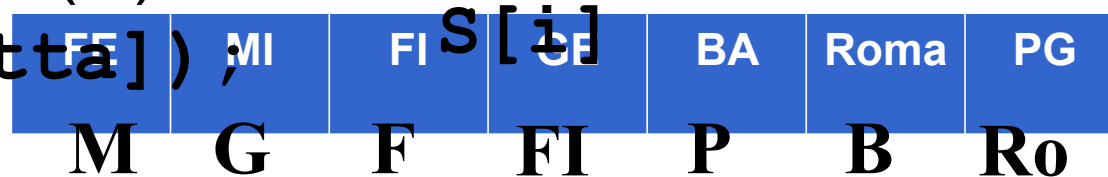
- Constraints:

- ~~all different~~(S)

~~solve~~ minimize

- circuit(N) [distanza[

in città] ;

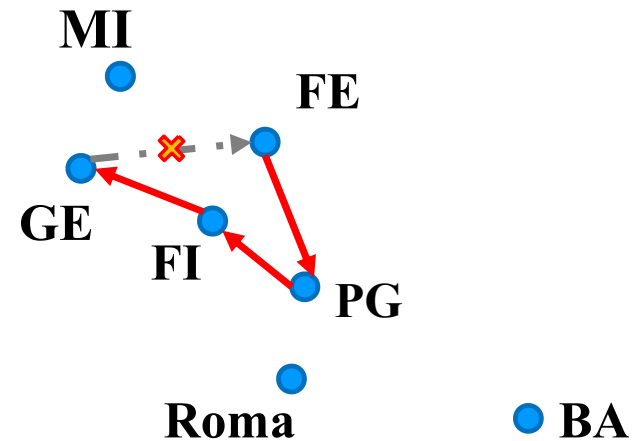


Propagazione vincolo circuit

- I vincoli **circuit** e **alldifferent** insieme risolvono il problema del circuito Hamiltoniano, che è un problema NP-completo
- Risulta quindi NP-completo ottenere la propagazione GAC dei due vincoli insieme
- In pratica, ci si accontenta di algoritmi di propagazione che garantiscano la correttezza della soluzione, anche se non raggiungono GAC o GBC

circuit constraint

- Algoritmo semplice
 - *cerca valore ground*
 - *segui path fino a una variabile*
 - *elimina dal dominio di questa (ultima) variabile la prima città*
 - *(sospenditi se non hai finito)*



	PG	GE			FI	
BA	FE	FI	GE	MI	PG	Roma

Esercizio

- *Dato il seguente CSP:*
- **var 2..5:X1; var 3..5: X2; var 1..4: X5;**
constraint circuit([X1,X2,4,1,X5]);
constraint alldifferent([X1,X2,4,1,X5]);
- *Si mostri la propagazione effettuata*

x1 **[2,3,4,5]**

x2 **[3,4,5]**

4

1

x5 **[1,2,3,4]**

Vincoli globali: table

- `table(array[int] of array[int, int] of TIPO: t) var TIPO: x,`
- Dove TIPO può essere bool o int.
- Impone che `x` sia una delle righe della tabella `t`.
- Utile per creare nuovi vincoli

```
include "globals.mzn";  
array [1..4] of var 0..9 : x;  
constraint table(x,  
  [|1,2,3,4  
   |5,6,7,8  
   |9,0,1,2  
   |3,4,5,6  
   |7,8,9,0|]);
```