Metaheuristics

Fares BOUGOURZI, Ph.D
Associate professor
DSLS Team, H&E Dept., JUNIA ISEN Lille
CNRS UMR-8520 IEMN
Tél. : 33 (0)3 20 30 40 27
fares.bougourzi@junia.com
faresbougourzi@gmail.com

JUNIA ISEN

iemn
Institut d'Electronique, de Microélectronique et de Nanotechnologie
UMR CNRS 8520

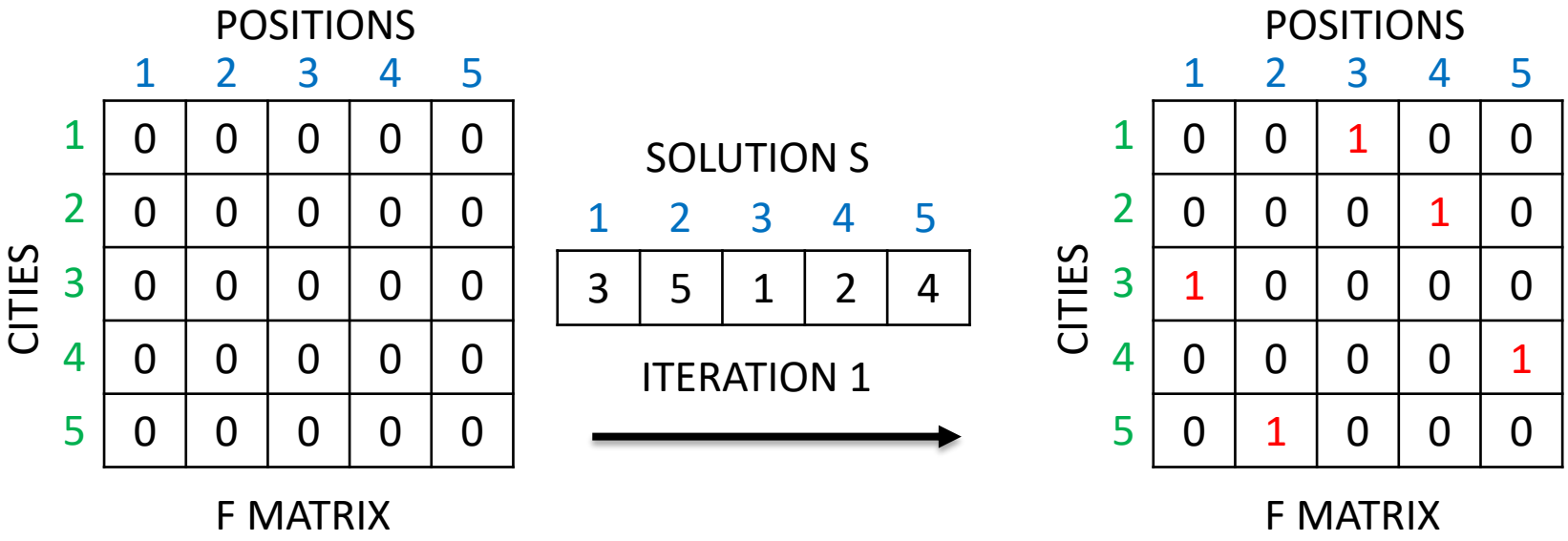
2.3 S-metaheuristics
Tabu Search (TS)

S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (long term memory)
  - Forcing search in unexplored regions
  - Frequency memory technic (problem dependent)
    - Focus search on less changed components in the search history
    - Example for TSP (Travelling Salesman problem)
      - Create a matrix *F* where *f(i,j)* corresponding to number of iterations where city *i* is positioned at *j* from the starting of algorithm
      - Diversification
        - Start search from a new initial solution *S* generated as follow
          - Use smallest values of *F* to replace components of *S*
          - Pursue search

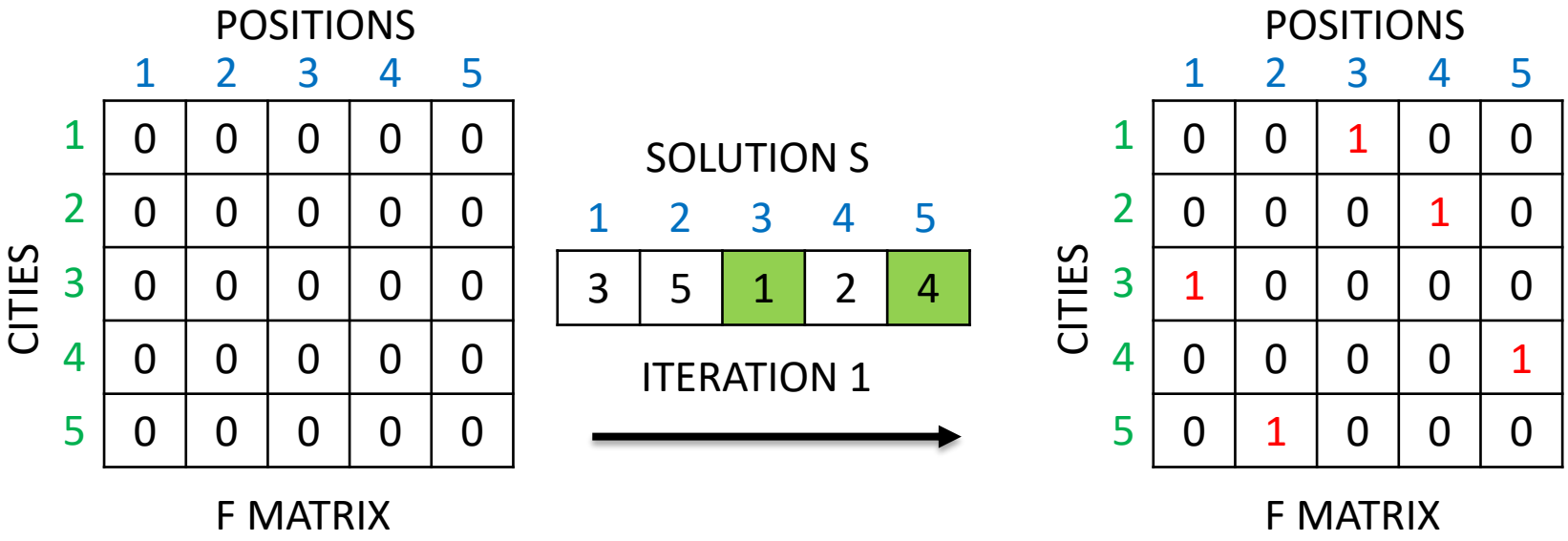
S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



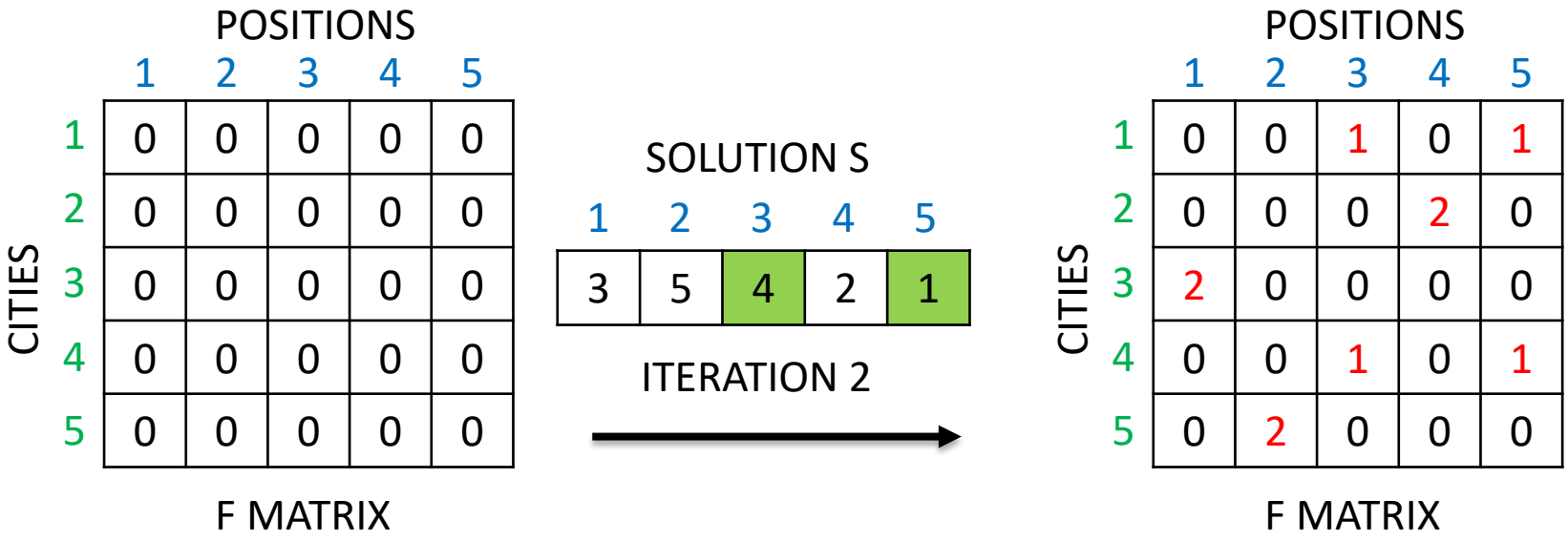
S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



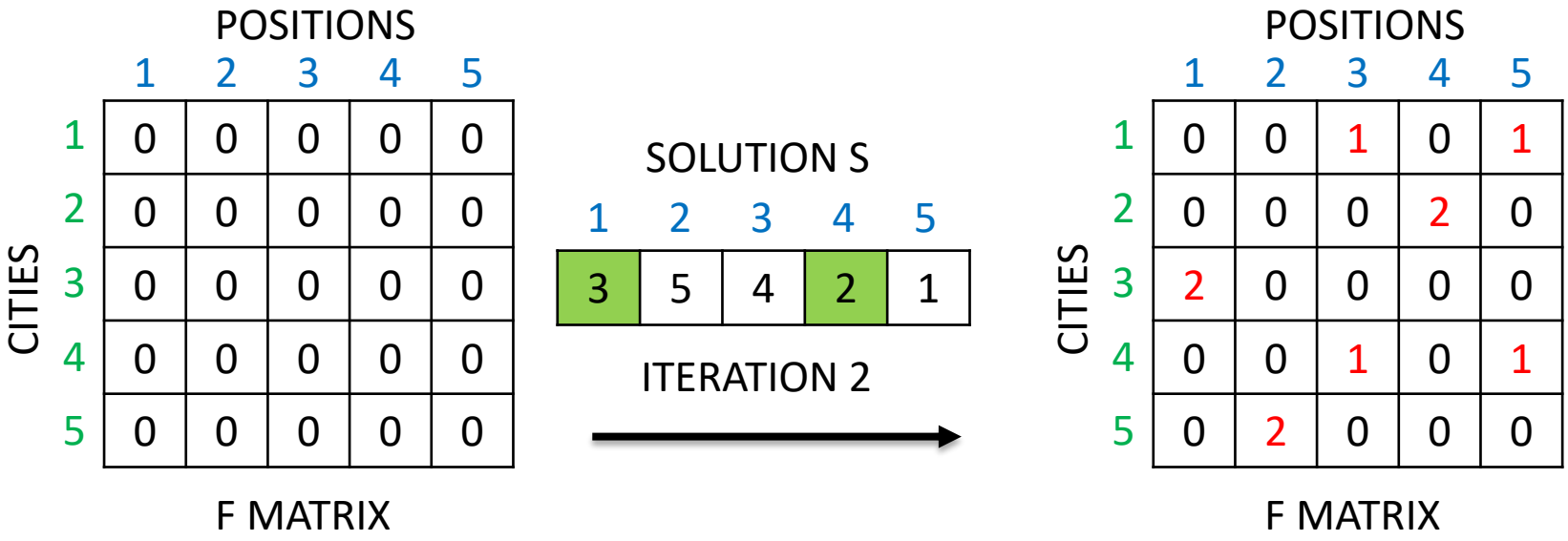
S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



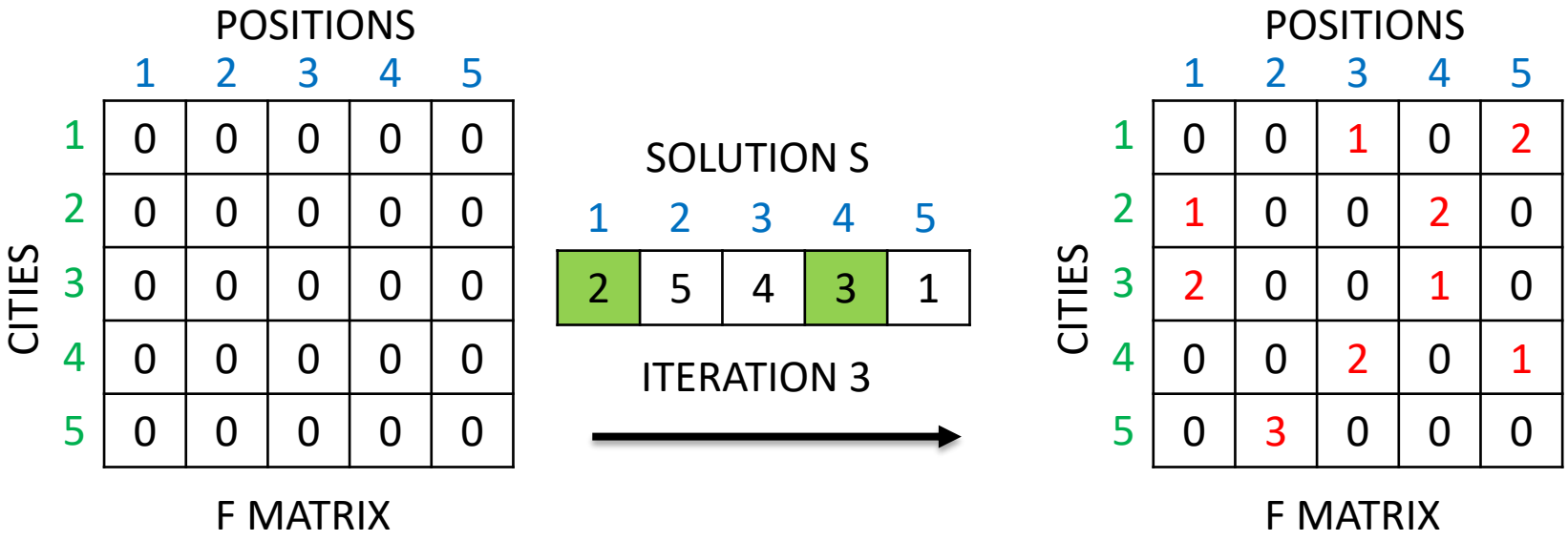
S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



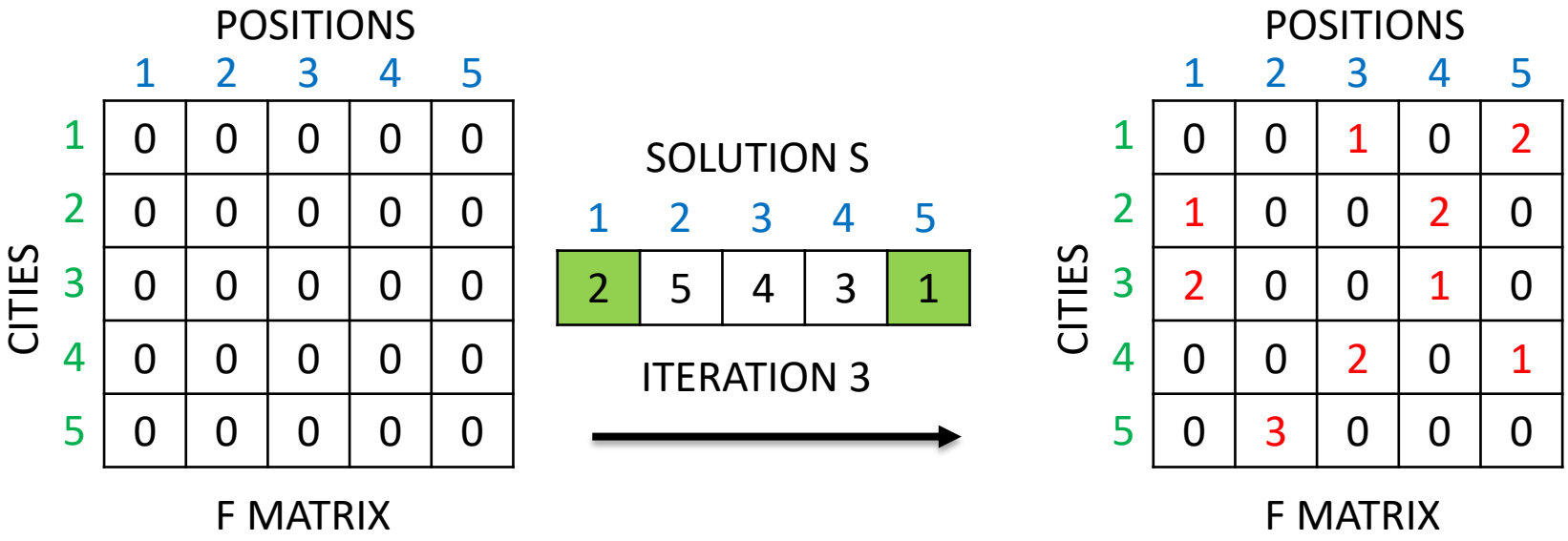
S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



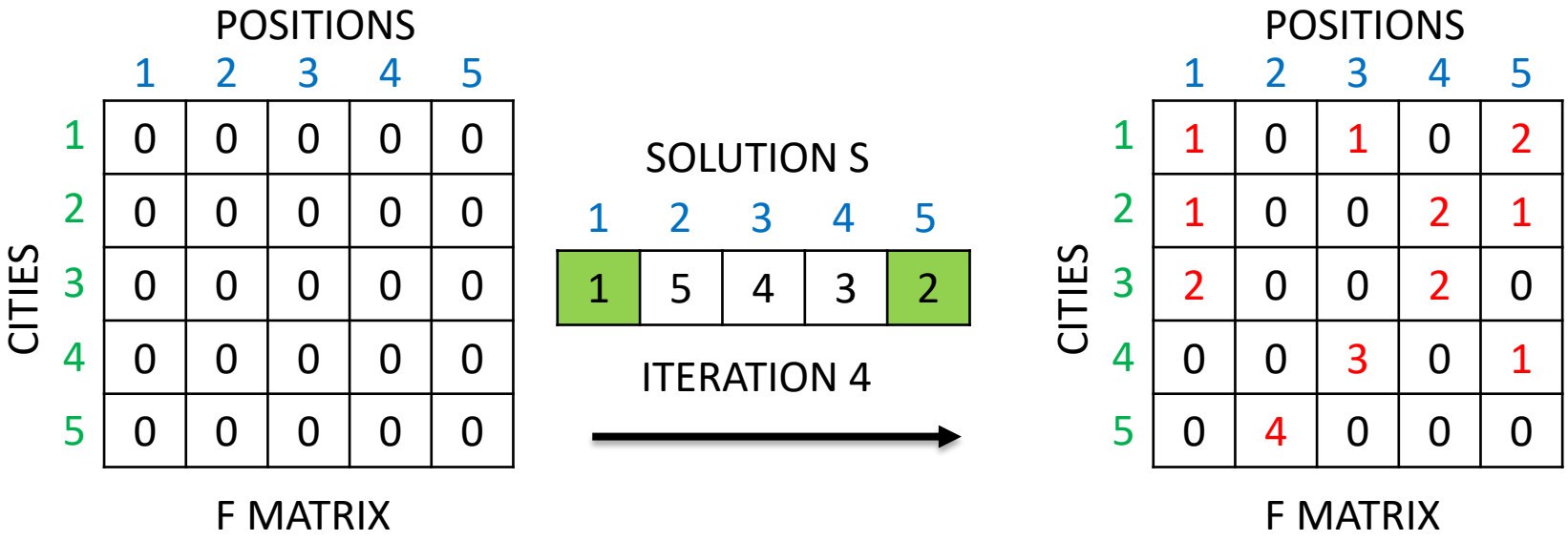
S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



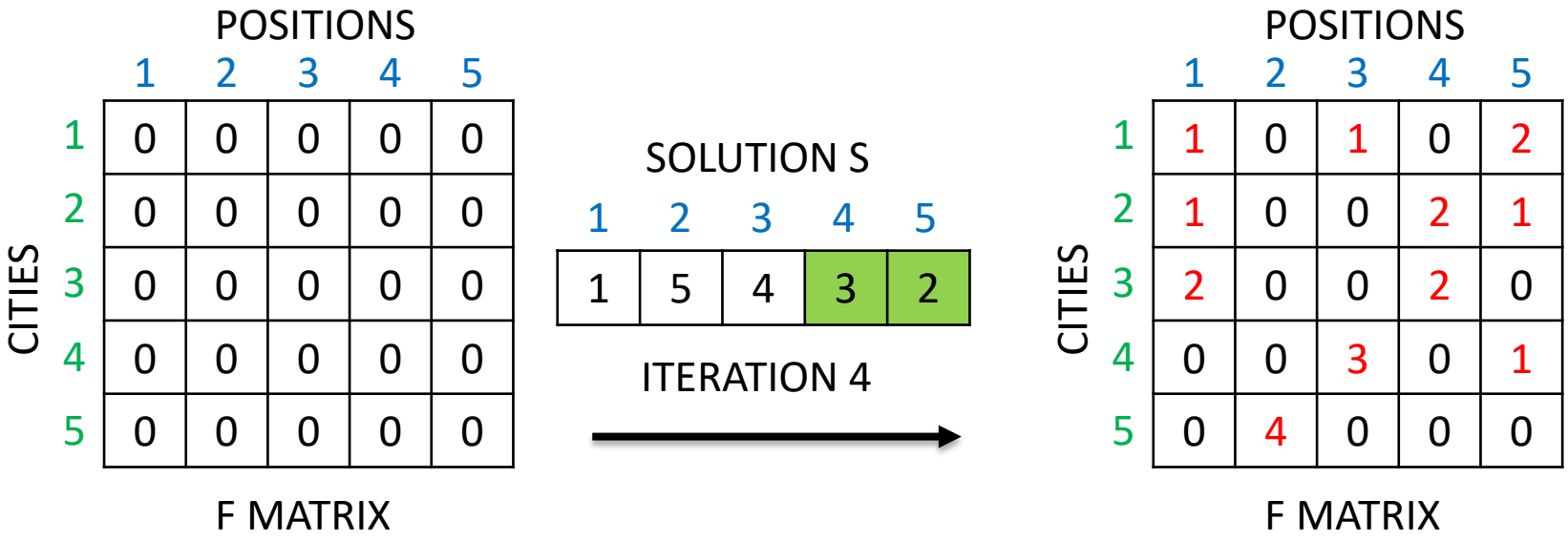
S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



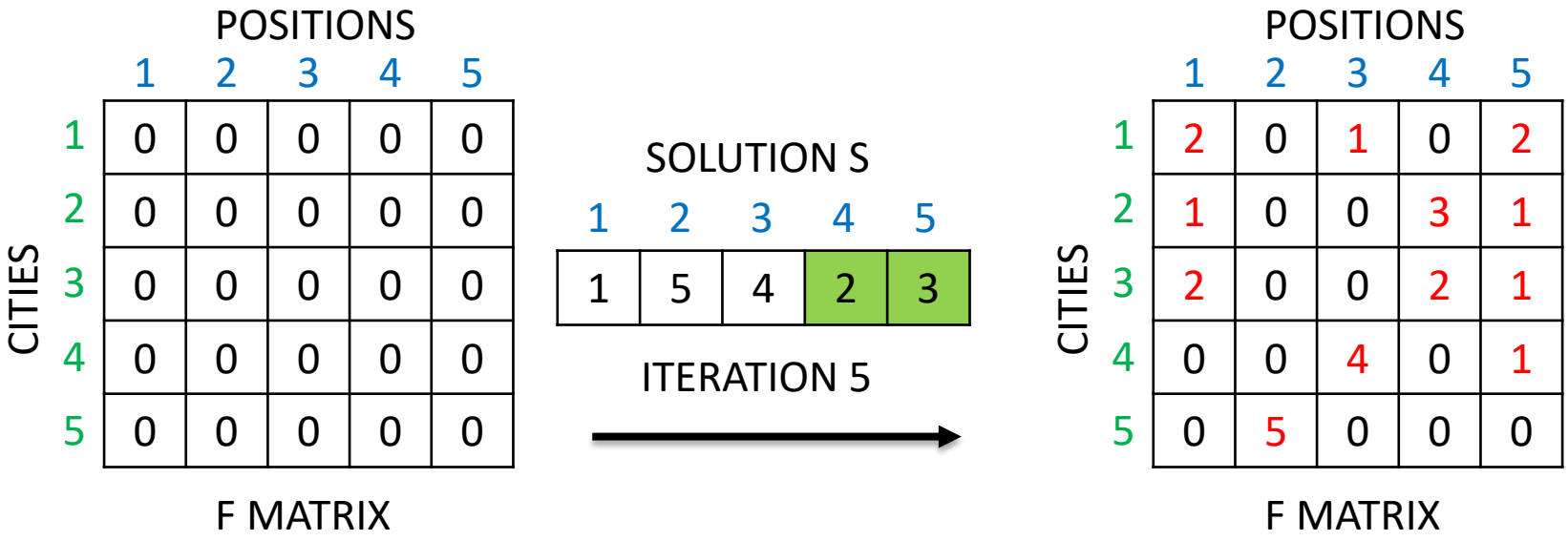
S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



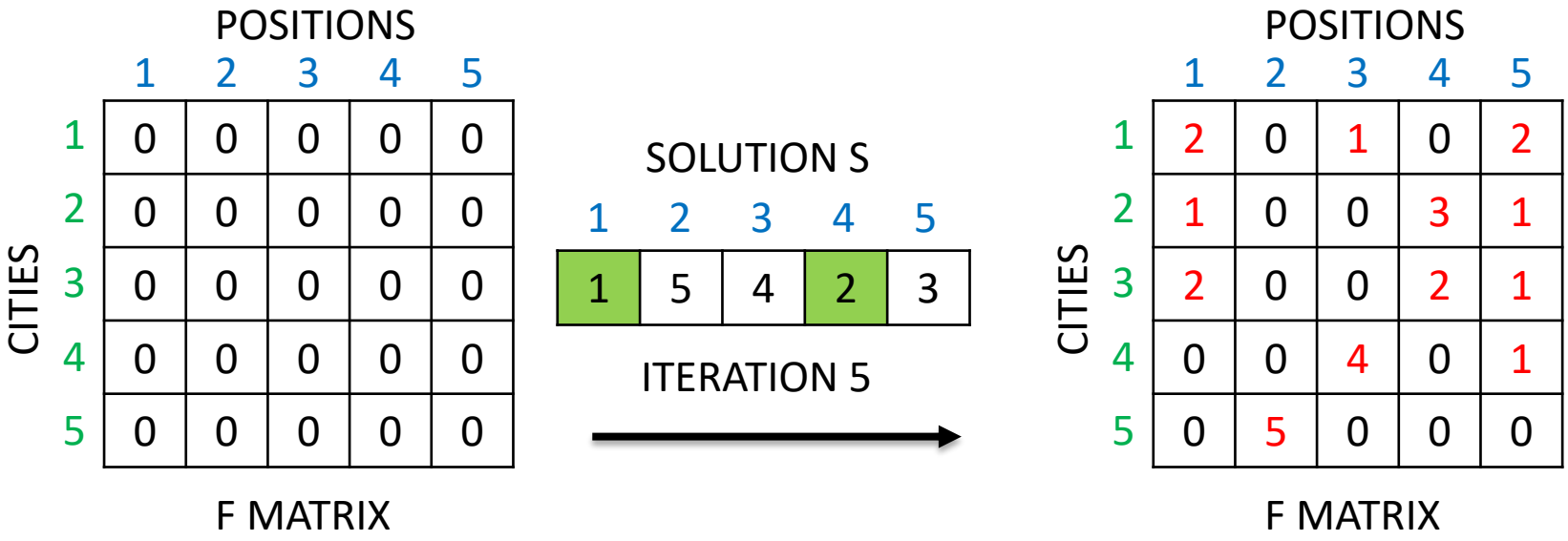
S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



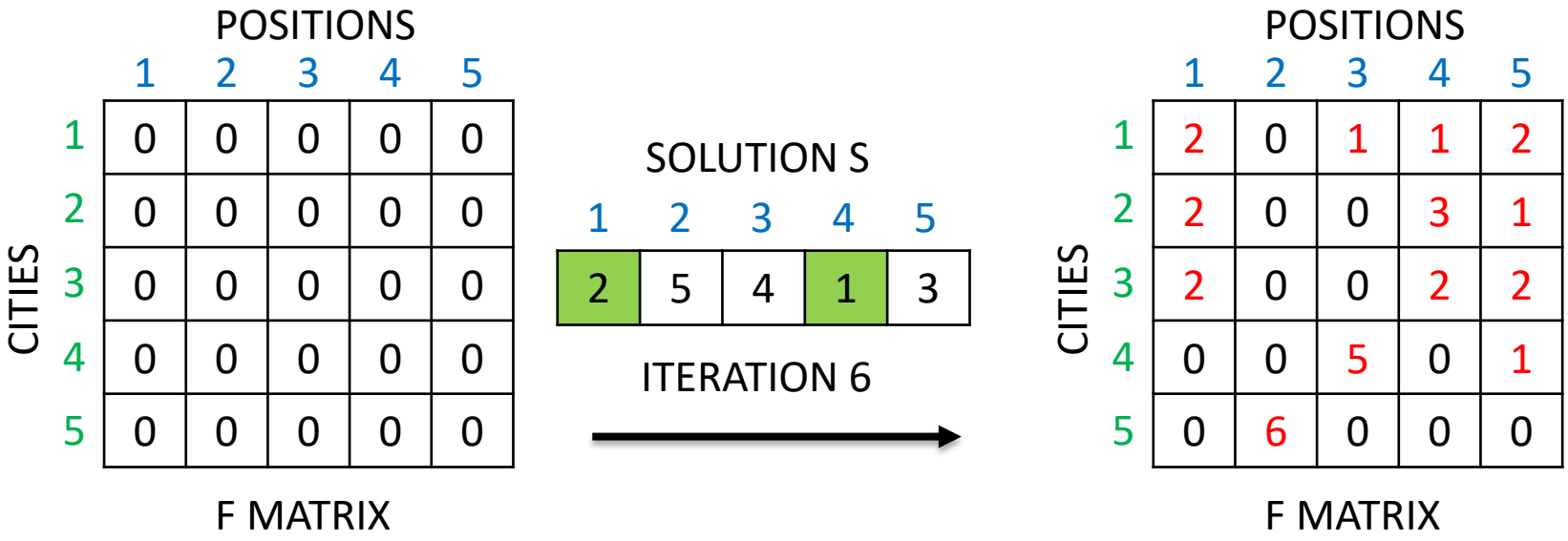
S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)



S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)

- Diversification
  - Start search from a new initial solution *S* generated as follow
    - Use smallest values of *F* to replace components of *S*
    - Pursue search

After a given number of iterations, start diversification

New solution S

1	2	3	4	5

+

CITIES \ POSITIONS	1	2	3	4	5
1	2	0	1	1	2
2	2	0	0	3	1
3	2	0	0	2	2
4	0	0	5	0	1
5	0	6	0	0	0

F MATRIX

=

S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)

- Diversification
  - Start search from a new initial solution *S* generated as follow
    - Use smallest values of *F* to replace components of *S*
    - Pursue search

After a given number of iterations, start diversification

New solution S

1	2	3	4	5
5				

POSITIONS

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

F MATRIX

S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)

- Diversification
  - Start search from a new initial solution *S* generated as follow
    - Use smallest values of *F* to replace components of *S*
    - Pursue search

After a given number of iterations, start diversification

New solution S

1	2	3	4	5
5				

+

F MATRIX

CITIES \ POSITIONS	1	2	3	4	5
1		0	1	1	2
2		0	0	3	1
3		0	0	2	2
4		0	5	0	1
5					

=

S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)

- Diversification
  - Start search from a new initial solution *S* generated as follow
    - Use smallest values of *F* to replace components of *S*
    - Pursue search

After a given number of iterations, start diversification

New solution S

1	2	3	4	5
5	1			

+

F MATRIX

CITIES \ POSITIONS	1	2	3	4	5
1		0	1	1	2
2		0	0	3	1
3		0	0	2	2
4		0	5	0	1
5					

=

S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)

- Diversification
  - Start search from a new initial solution *S* generated as follow
    - Use smallest values of *F* to replace components of *S*
    - Pursue search

After a given number of iterations, start diversification

New solution S

1	2	3	4	5
5	1			

+

POSITIONS

CITIES	1	2	3	4	5
1					
2			0	3	1
3			0	2	2
4			5	0	1
5					

F MATRIX

=

S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)

- Diversification
  - Start search from a new initial solution *S* generated as follow
    - Use smallest values of *F* to replace components of *S*
    - Pursue search

After a given number of iterations, start diversification

New solution S

1	2	3	4	5
5	1	3		

+

POSITIONS

CITIES	1	2	3	4	5
1					
2			0	3	1
3			0	2	2
4			5	0	1
5					

F MATRIX

=

S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)

- Diversification
  - Start search from a new initial solution *S* generated as follow
    - Use smallest values of *F* to replace components of *S*
    - Pursue search

After a given number of iterations, start diversification

New solution S

1	2	3	4	5
5	1	3		

+

POSITIONS

CITIES	1	2	3	4	5
1					
2				3	1
3					
4				0	1
5					

F MATRIX

=

S-metaheuristics – Tabu Search Algorithm (TS)

- Diversification (Frequency memory – TSP 5 cities)

- Diversification
  - Start search from a new initial solution *S* generated as follow
    - Use smallest values of *F* to replace components of *S*
    - Pursue search

After a given number of iterations, start diversification

New solution S

1	2	3	4	5
5	1	3	4	

POSITIONS

CITIES	1	2	3	4	5
1					
2				3	1
3					
4				0	1
5					

F MATRIX

S-metaheuristics – Tabu Search Algorithm (TS)

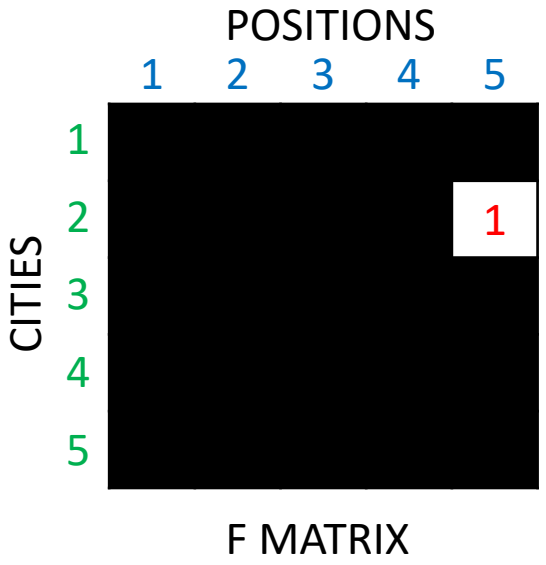
- Diversification (Frequency memory – TSP 5 cities)

- Diversification
  - Start search from a new initial solution *S* generated as follow
    - Use smallest values of *F* to replace components of *S*
    - Pursue search

After a given number of iterations, start diversification

New solution S

1	2	3	4	5
5	1	3	4	



S-metaheuristics – Tabu Search Algorithm (TS)

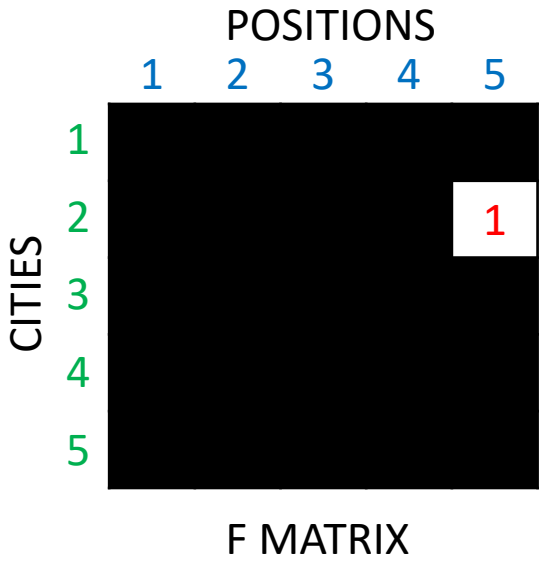
- Diversification (Frequency memory – TSP 5 cities)

- Diversification
  - Start search from a new initial solution *S* generated as follow
    - Use smallest values of *F* to replace components of *S*
    - Pursue search

After a given number of iterations, start diversification

New solution S

1	2	3	4	5
5	1	3	4	2



1	2	3	4	5
5	1	3	4	2

S-metaheuristics – Tabu Search Algorithm (TS)

Template of tabu search algorithm.

Save ***S*** as ***best_solution***

$s = s_0$; /* Initial solution */
Initialize the tabu list, medium-term and long-term memories ;
Repeat
Find best admissible neighbor s' ; /* non tabu or aspiration criterion holds */

Improve ***S***

Compare ***S*** with the ***best_solution*** and update if necessary

$s = s'$;
Update tabu list, aspiration conditions, medium and long term memories ;
If intensification_criterion holds **Then** intensification ;
If diversification_criterion holds **Then** diversification ;
Until Stopping criteria satisfied
Output: Best solution found.

S-metaheuristics – Algorithms review

Local search

- Selection strategies of the best neighbor
  - Best improvement (steepest descent)
  - First improvement
  - Random improvement

High probability to fall into local optima

Simulated annealing

- Accepting the degradation of a solution under some conditions
  - High temperatures promote accepting bad solutions
  - Static temperatures prevent accepting very bad solutions

Tabu search

- Accepting the degradation of a solution if and only if
  - Non tabu solution
- Memory usage to optimize the search process
  - Short term → tabu list
  - Medium term → recency for intensification
  - Long term → frequency for diversification

Strategies to escape from local optima

S-metaheuristics – Tabu Search Algorithm (TS)

- Lab session – first part 

Implement your third and last S-metaheuristic algorithm - the Tabu Search algorithm (TS) -

- Version 1, only Tabu list (recommended before next session)
- Version 2, add intensification process
- Version 3, add diversification process

Apply the 2 versions to

- The TSP problem – Data available on the campus
- Show the best solution and associated trajectory curve for each version