ECML European Conference on Machine Learning and
PKDD Principles and Practice of Knowledge Discovery in Databases
Joint Workshop on Large-Scale Evolving Networks and Graphs

Design and implementation issues of a time-dependent shortest path algorithm for multimodal transportation network

Abdelfettah IDRI[1], Mariyem OUKARFI[2], Azedine BOULMAKOUL[2] and Karine ZEITOUNI[3]

[1]LIM Lab. IOS, National School of business and Management, Casablanca, Morocco

[2]LIM Lab. IOS, Computer Sciences Department, FSTM, Morocco

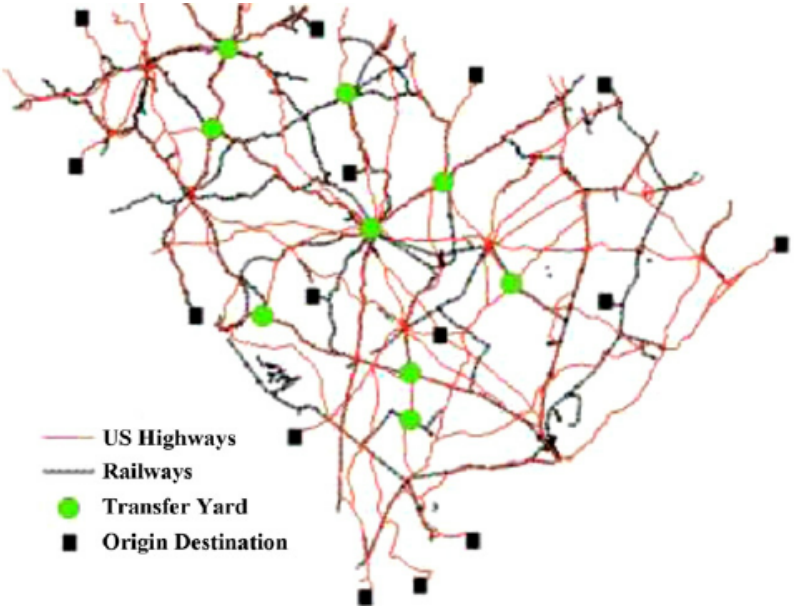
[3]David Lab. University of Versailles Saint-Quentin-en-Yvelines, Versailles, France

*This work was supported by the **MESRSFC** and the **CNRST** – Morocco*

Outline

- Scope
- Constraint Based Shortest Path Algorithm (CBSPA)
- Search process
- Search dimensions
- Building process of the solution
- Experiments
- Large scale networks
- Preliminary Results
- Conclusion & future work

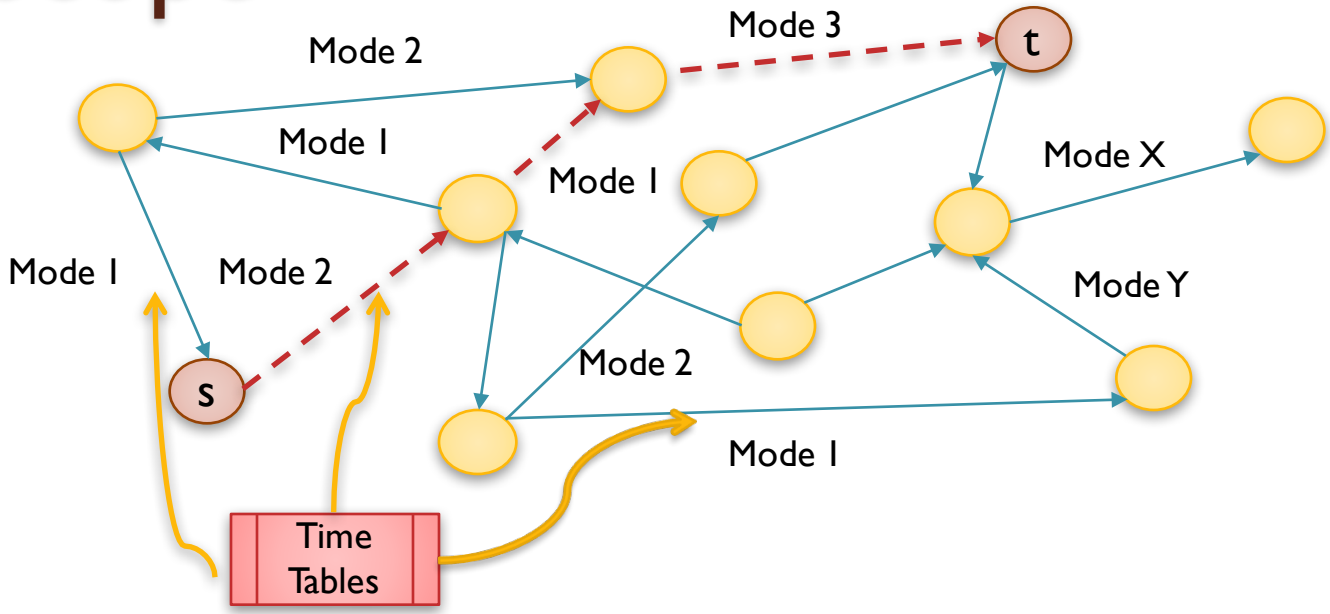
Scope



- Transport planning problem ?
  - Travel time
  - Financial costs
  - Multimodality
  - …

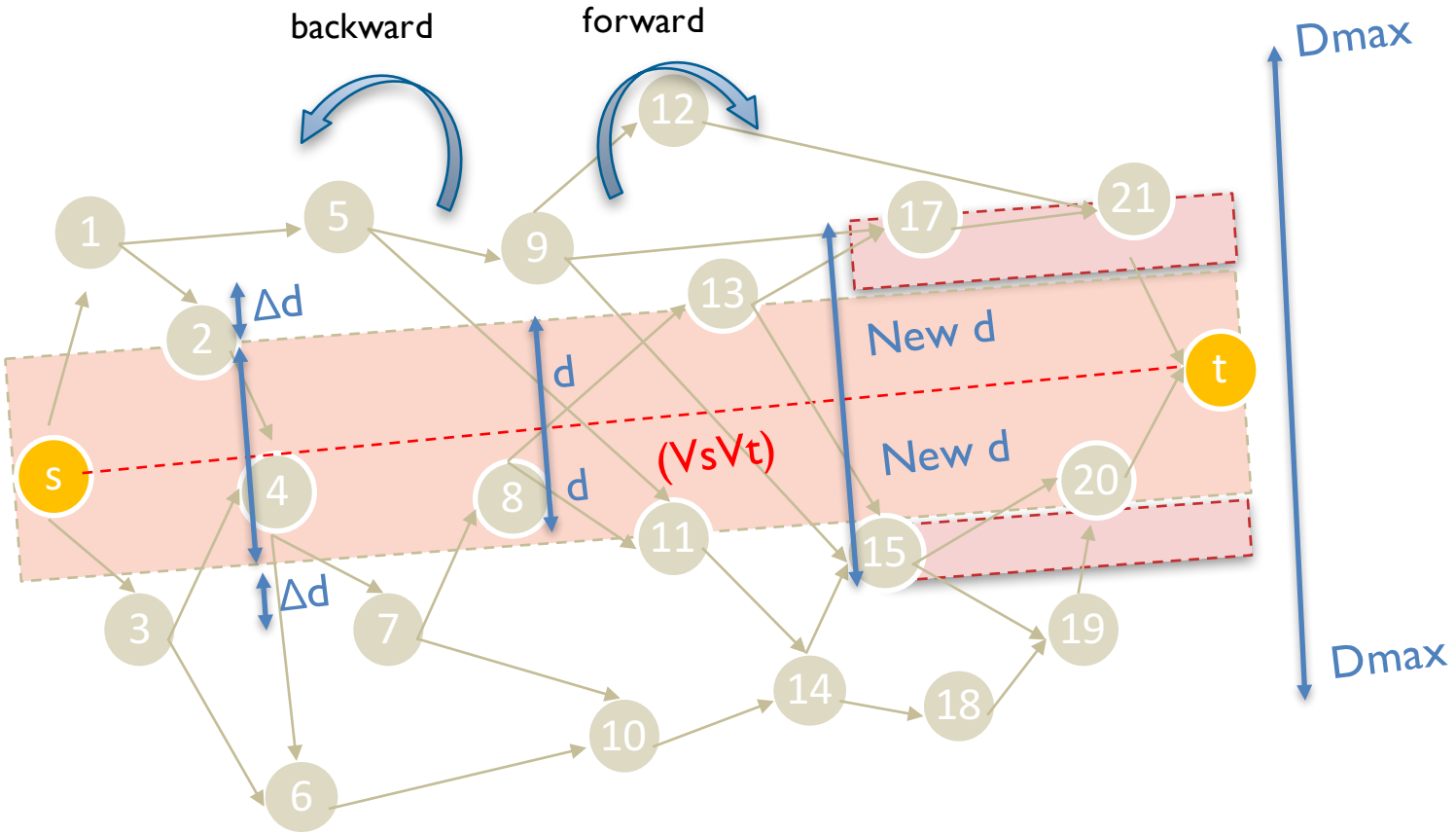
➔ Shortest path problem ?

Scope



- How to efficiently compute the shortest path from s to t ?
- Taking both Time variation and multimodality of the network raises the problem of **big search space**

CBSPA : Constraint Based Shortest Path Algorithm



CBSPA : Constraint Based Shortest Path Algorithm

- Virtual path from s to t ($V_s V_t$)
- Search space restriction by mean of the user-defined constraint "d"
- Increasing d with Δd whenever it is necessary without surpassing Dmax
- Possibility of navigating back and forth to capture more nodes

CBSPA Algorithm

Algorithm CBSPA (u, d, path ,Q , t)

Output: shortest path

// **($V_s V_t$):** virtual path which is calculated based on the coordinates (Euclidean case: straight line between V_s and V_t) or the minimal/mean value of the cost function (in our case, the travel time)

// **Q**: the set of the neighbors of u satisfying the constraint control, a parameter needed to forward the intermediate results

// **t:** start time; u: start node

// **d** : represents the mean distance(cost) from all nodes to the virtual path

$$d = \sum_{1}^{n} dist\left(v_i, (VsVt)\right)/n$$

// **Dmax**: represent the maximum value of d, that's the distance (cost) to the farthest node of the network

CBSPA Algorithm

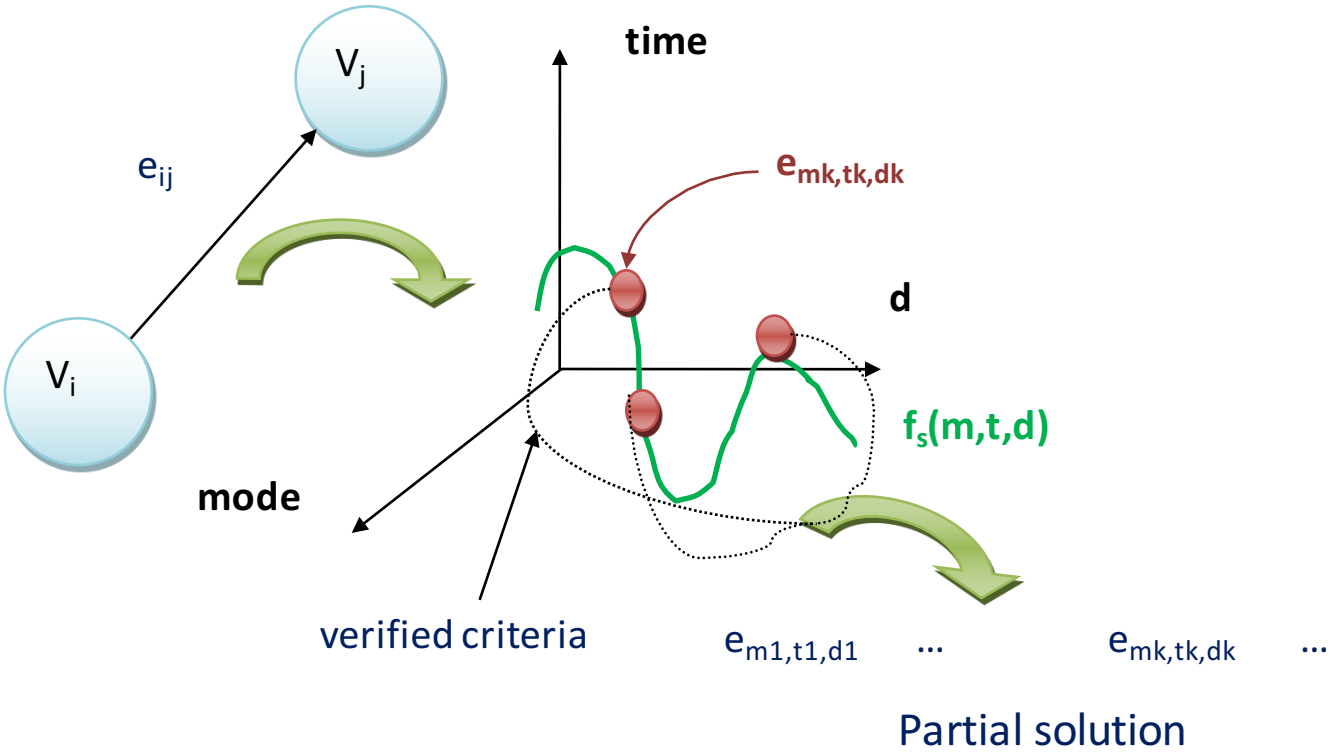
```
1    If u = V_t then
2       Return path
3    Else
4    Let Q = {v  Neighbors(u)/ dist(v,(V_s V_t)) ≤ d}
5                if Q = Ø then
6                    If d < Dmax then
7                       CBSPA (u, d+Δd ,path,Q,t)
8                    Else
9                       Let w = predecessor(u)
10                      CBSPA(w,d+Δd, path\{u},Q,t)
11               Else
12                      Let Q_new =
13                      Let Q = {v  (∐_{v∈Q} OneStepMMTDSP(v,t)
14                      ∀ v ∈ Q, CBSPA(v,d,path{v},Q\{v},t)
```

*The function OneStepMMTDSP(v,t) generates the next iteration candidates based on the timetable of the vertex v from Vs to Vt.

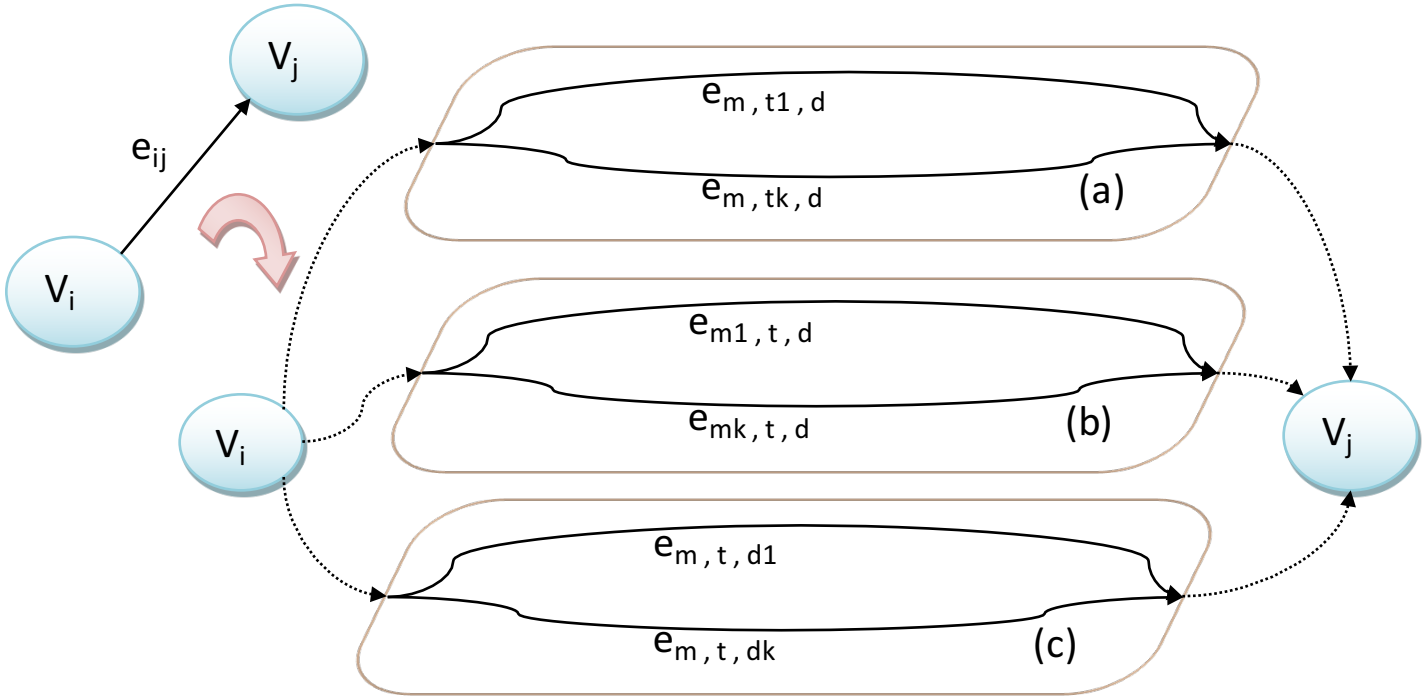
Search process

- Target oriented
- Driven by three search dimensions
  - Mode
  - Time
  - User-defined constraint d
- Each physical edge generates instances in the 3 dimensional space based on the search function $\mathbf{f_s(m,t,d)}$

Search process



Search layers



(a) time layer (b) multimodality layer (c) constraint layer

Time dependency layer

- A timetable is assigned to each edge
- Edge scenario = departure-arrival case
- Set of edge instances from Time dependency view point = set of edge scenarios
- Each instance has to be handled while fixing (projection) the mode and the constraint

Multimodality layer

- modes are assigned to each edge
- Edge scenario = given mode
- Set of edge instances from Mode view point = set of edge scenarios
- Each instance has to be handled while fixing (projection) the time and the constraint

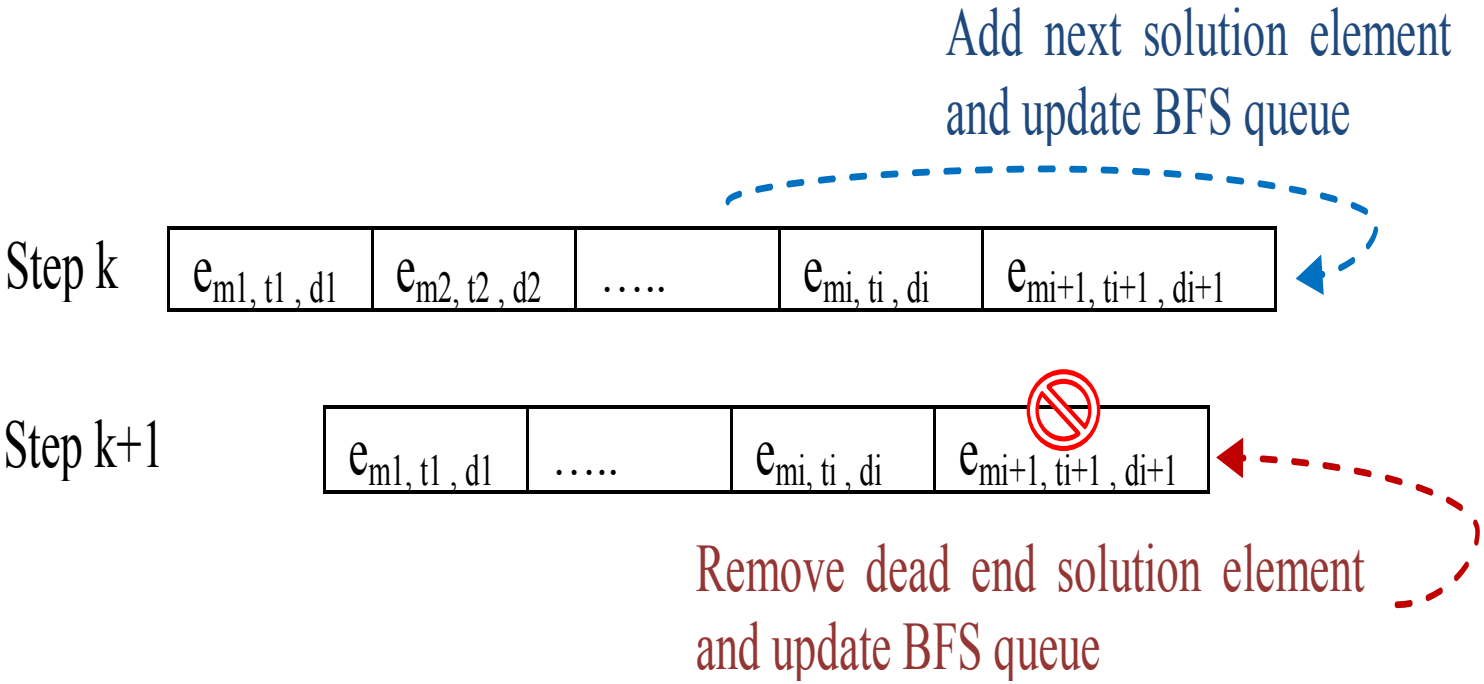
Constraint layer

- Only the edges respecting the constraint are considered for the search process
- Edge scenario = given constraint value
- Set of edge instances from Constraint view point = set of edge scenarios
- Each instance has to be handled while fixing (projection) the time and the mode

Building process of the solution

- Based on the partial / complete solution approach
- Backtracking mechanism
- Eliminating visited 3 dimensional instances (mark and unmark instances (not edges))

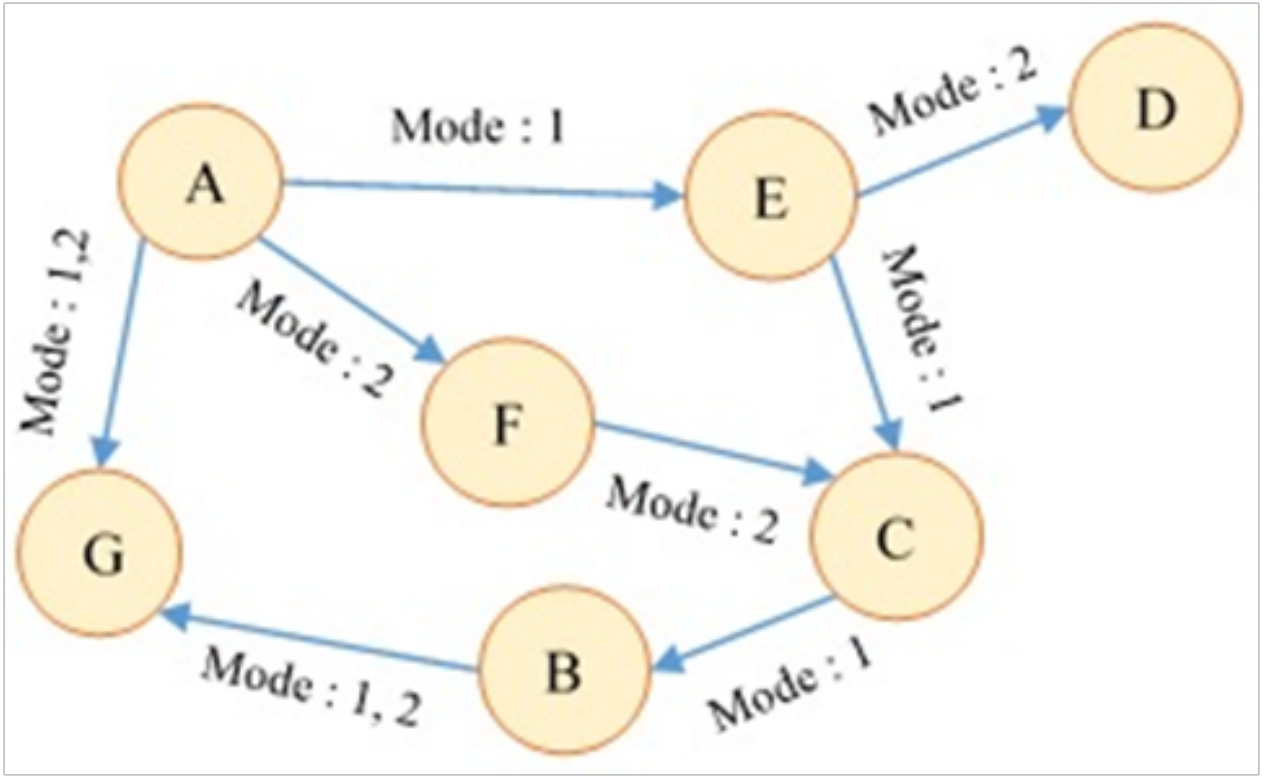
Building process of the solution



Example :

Mode 1			*Mode 2*		
Edge	Timetable	Cost function	Edge	Timetable	Cost function
a → e	1 → 3	2	e → d	1 → 4	7
	3 → 4	4		12 → 15	3
e → c	3 → 5	1	a → f	4 → 6	4
	4 → 8	1		7 → 9	1
c → b	6 → 8	5	f → c	6 → 7	1
	7 → 10	2		8 → 9	5
g → a	2 → 3	7	g → a	3 → 9	1
	9 → 11	6		8 → 11	10
b → g	9 → 10	1	b → g	10 → 11	5
	10 → 11	1		13 → 15	8

Example : network presentation



Example : results

Path to find :[[Node id: 1]======>[
Node id: 2]]

[

[start node: [Node id: 1]]

[end node: [Node id: 5]]

[travel : 1 ===> 3 ===> 2]

[mode : 1]

[start node: [Node id: 5]]

[end node: [Node id: 3]]

[travel : 3 ===> 5 ===> 1]

[mode : 1]

[start node: [Node id: 3]]

[end node: [Node id: 2]]

[travel : 6 ===> 8 ===> 5]

[mode : 1]

[Total cost : 8]

]

time :0 s

Final D: 10

Path to find :[[Node id: 3
]======>[Node id: 7]]

[

[start node: [Node id: 3]]

[end node: [Node id: 2]]

[travel : 6 ===> 8 ===> 5]

[mode : 1]

[start node: [Node id: 2]]

[end node: [Node id: 7]]

[travel : 9 ===> 10 ===> 1]

[mode : 1]

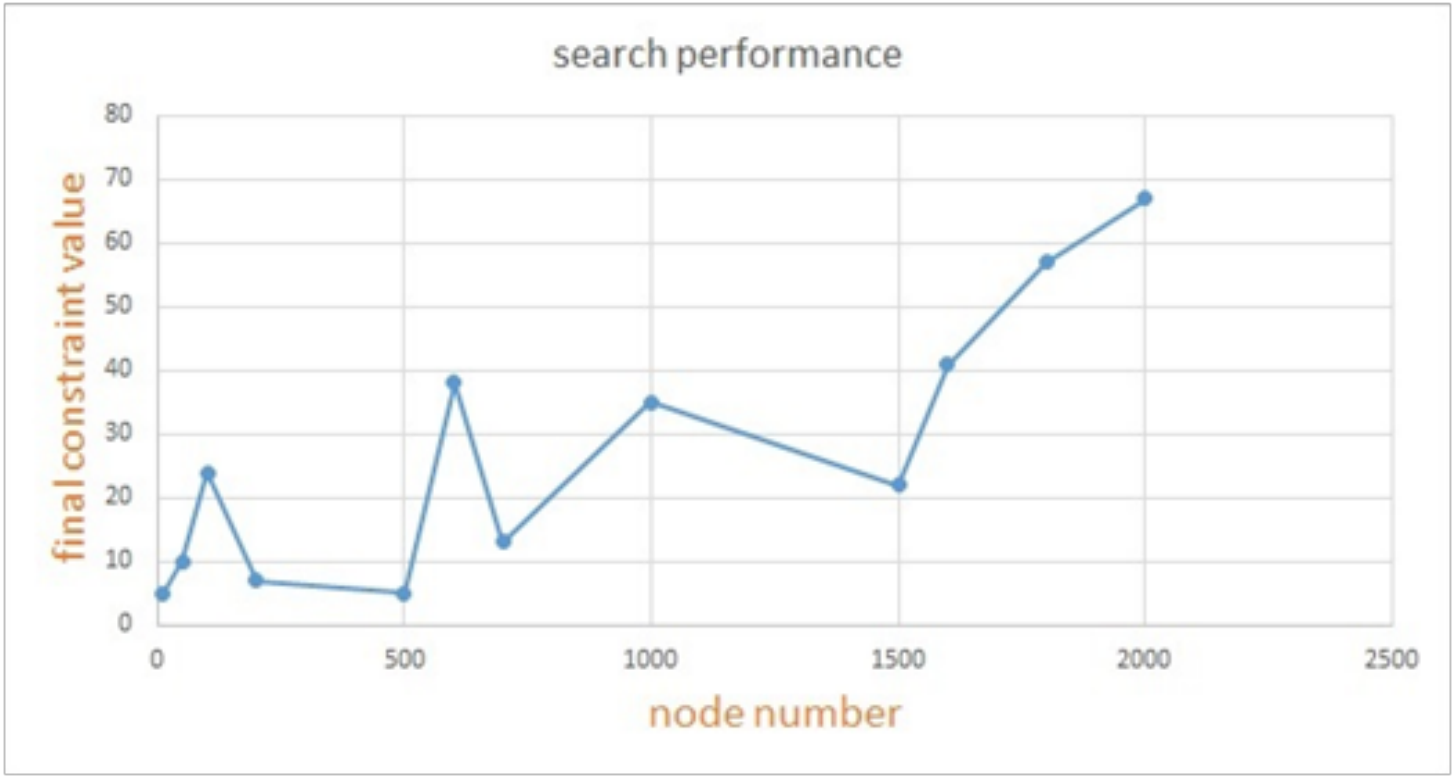
[Total cost : 6]

]

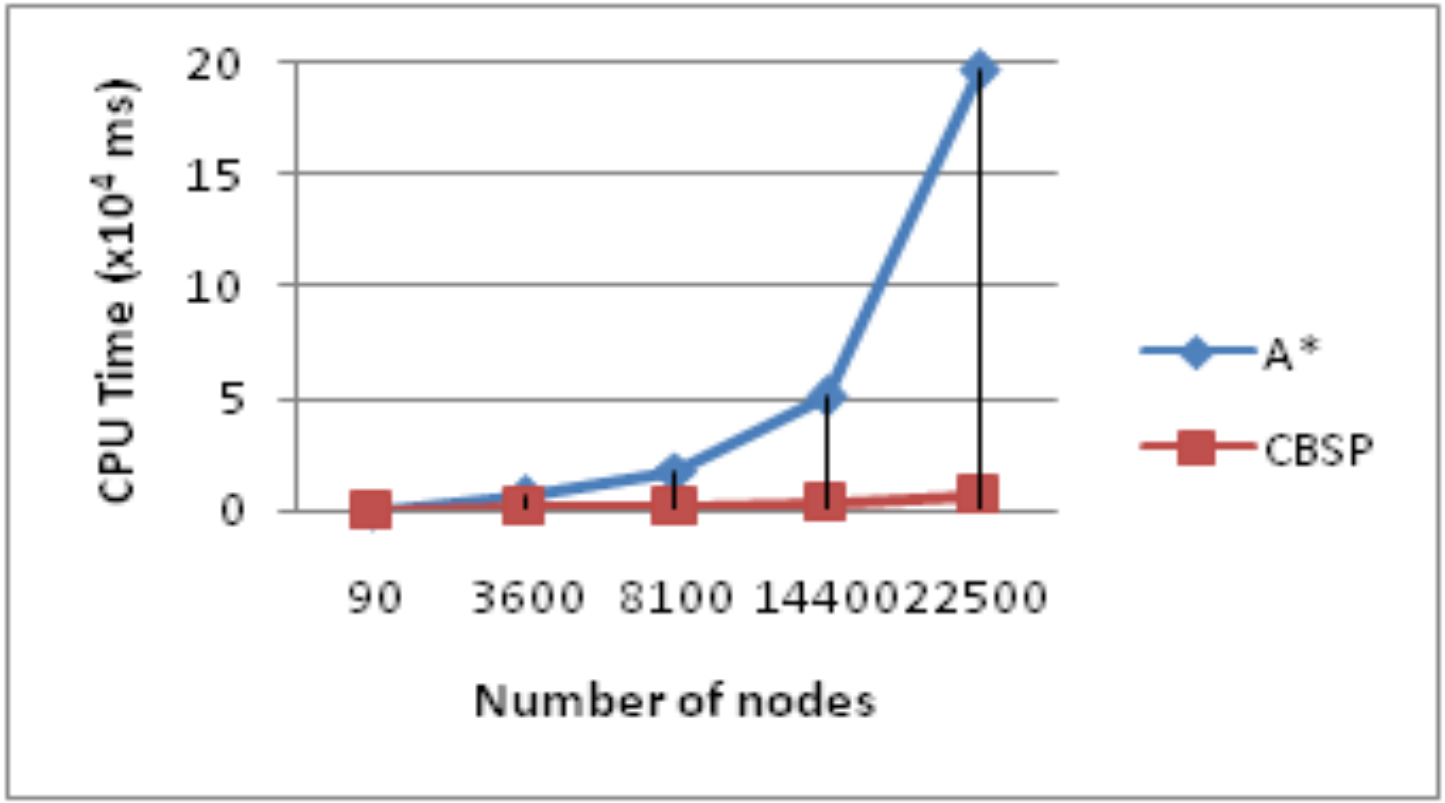
time :0 s

Final D: 5

Evaluation - Impact of the network density on the constraint parameter



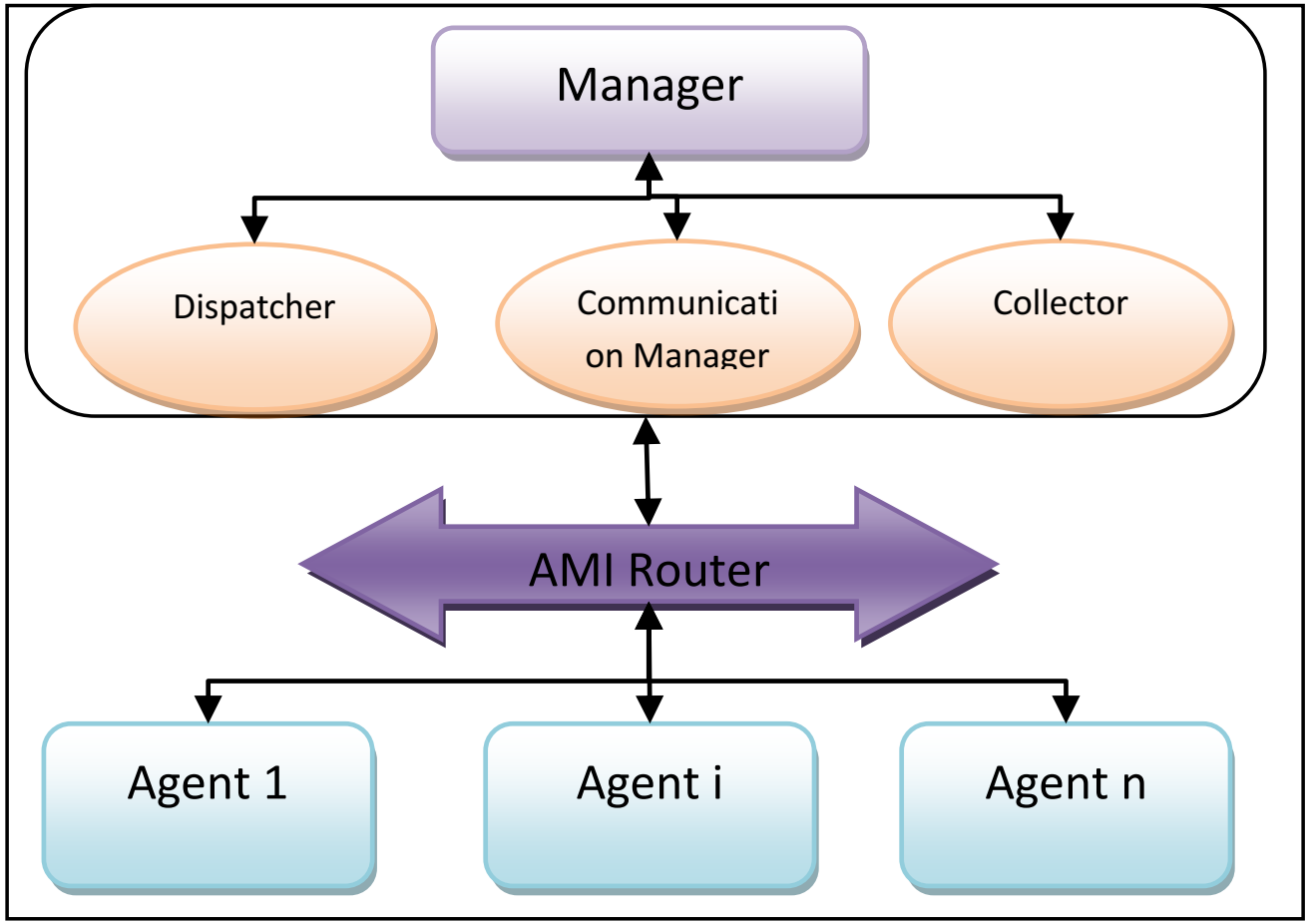
Evaluation - Performances wrt Network Size



Large scale networks

- Parallel distributed architecture relying on Manager / Agent model
- Scalability
- Performance
- CORBA based architecture
- AMI communication model

Large scale networks



Large scale networks

- Agent : computes elementary intermediate paths
- Manager: builds the complete solution, dispatches the tasks, collects the intermediate results and covers the communication issues

Preliminary Results

Table 2. CPU time comparison between the performance of CSPA in monolithic and distributed architecture

Graph instance			CPU Time ($x10^{-2}$ms)	
$\|V\|$	$\|E\|$	$\|M\|$	Monolithic arch.	Distributed arch.
100	230	3	0.8	0.77
500	800	3	1.66	1.62
1000	2600	3	2.85	1.5
2000	5000	3	3.01	1.86

Conclusions and future work

- ✓ Design and implementation aspects of our proposed solution dealing with the Time-dependant multimodal transport problem
- ✓ CBSPA algorithm to find the shortest path
- ✓ Design techniques to drive the search process in a 3 dimensional search space
- ✓ A CORBA parallel distributed architecture to address the big data issue
- ➢ Thorough experimental evaluation is still to be conducted
- ➢ Investigation of the integration of our approach in existing algorithms

- THANK YOU