Optimizing City Traffic using Time-Expanded Graphs

Orange Labs

Sébastien FELIX, Research & Development

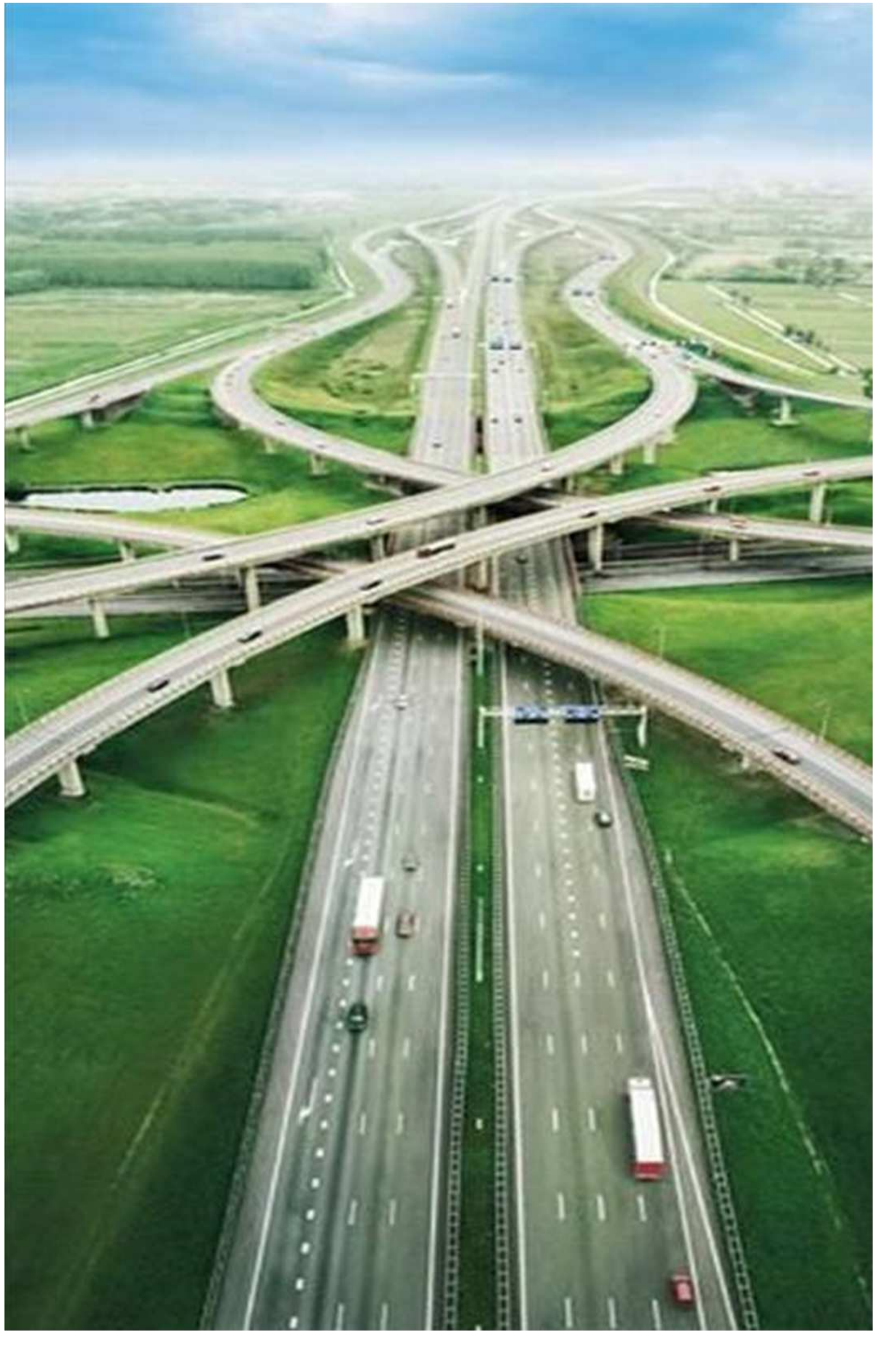
July 26th 2013, Recent Advances in Optimization

Optimizing city traffic using Time-Expanded Graphs

Introduction

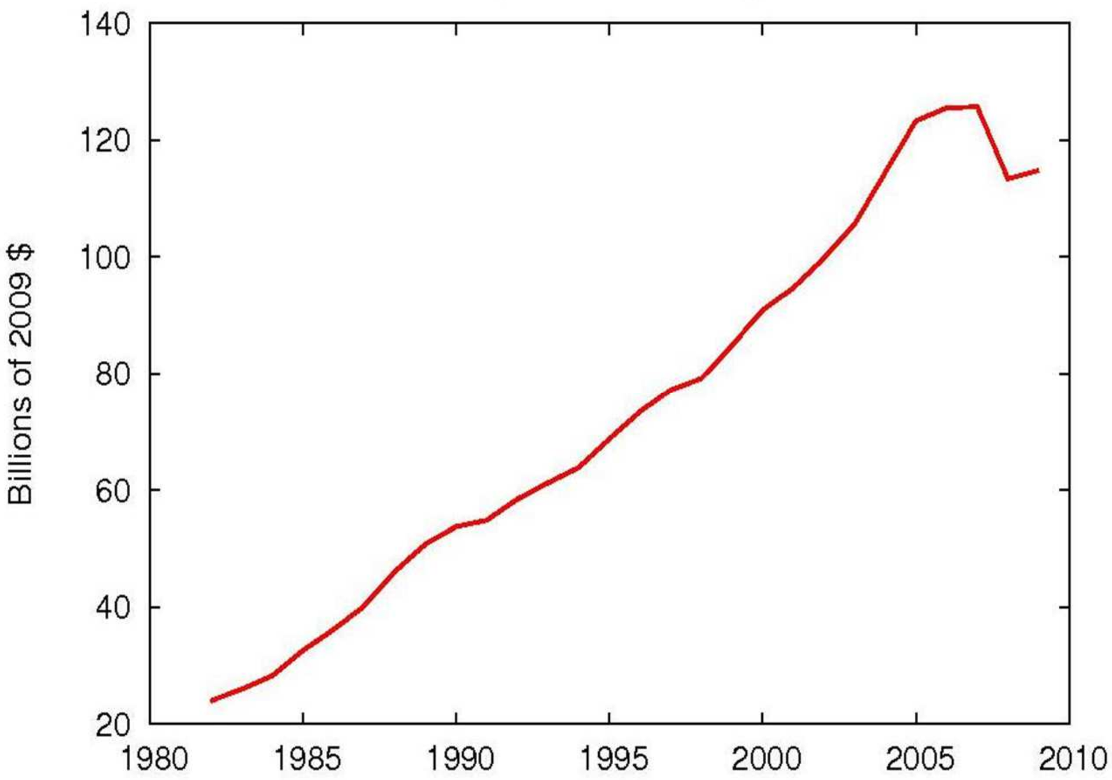
Introduction

- Growing number of vehicles and demands.
- Lot of data available (G.P.S, magnetic loops, data from telecommunication networks, ...).
- Automatization of driving (Google cars, vehicle pools, ...).



Introduction

- Congestion cost was estimated at over 100 billion dollars in the United States in 2009.



Introduction

- Full automatic cars :
  - Google automatic car
  - Volkswagen
- Connected cars :
  - Apple
  - Google API
- Pool of shared vehicles

(picture from http://www.KeyDatabase.in)

Introduction

- Goal : devellop a framework to manage a fleet of automatic cars in a city and reduce the transportation cost.

(picture from http://images.math.cnrs.fr)

Existing approaches

Existing approaches (Lighthill, Whitham and Richards model)

- Model based on fluid mechanics and Partial Differential Equations [1] :
  - Relation between the density and the speed of trafic

$$\partial_t \rho + \partial_x f(\rho) = 0 \quad \text{with} \quad f(\rho) = \rho\, v(\rho)$$

- Traffic is modelled as a continuous density function
- Trajectories are fixed
- Speed is variable
- No overtakings allowed

[1] R.M. Colombo,P. Goatin and M.D Rosini, On the modeling and Management of Traffic

Existing approaches (Lighthill, Whitham and Richards model)

- Pros :
  - Method based on P.D.E (efficient and stable framework)
  - Solution given is close to a real-world trafic behaviour
  - Work efficiently on Highway network to reduce congestion [2]
- Cons :
  - Can't be applied on a large city
  - Trajectories are fixed
  - No available data for a single car

[2] Serge Hoogendoorn, Effective Dynamic Speed Limit Control approaches

Existing approaches (multicommodity flow model)

- Model based on graph theory [3]
- Traffic is modelled as a set of commodities associated with an amount of flow
- Trajectories are not fixed
- Speed could vary with a fixed rule over time
- No overtakings allowed

[3] A. Hall, S. Hippler and M. Skutella, Multicommodity flows over time : Efficient algorithms and complexity

Existing approaches (multicommodity flow model)

- Pros :
  - Method based on Linear Programming (existence of polynomial complexity algorithms)
  - Work efficiently on large graphs
- Cons :
  - Don’t takes into account the relation between density and speed
  - No available data for a single car

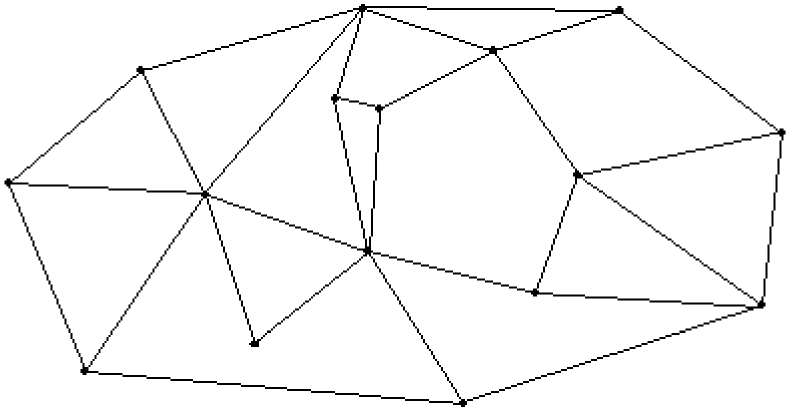
Proposed model

Proposed model

- Each car is represented by a particle
- Trajectories are NOT fixed
- Speed is variable
- No overtakings allowed
- Model based on Graph theory (vertex disjoint path problem)

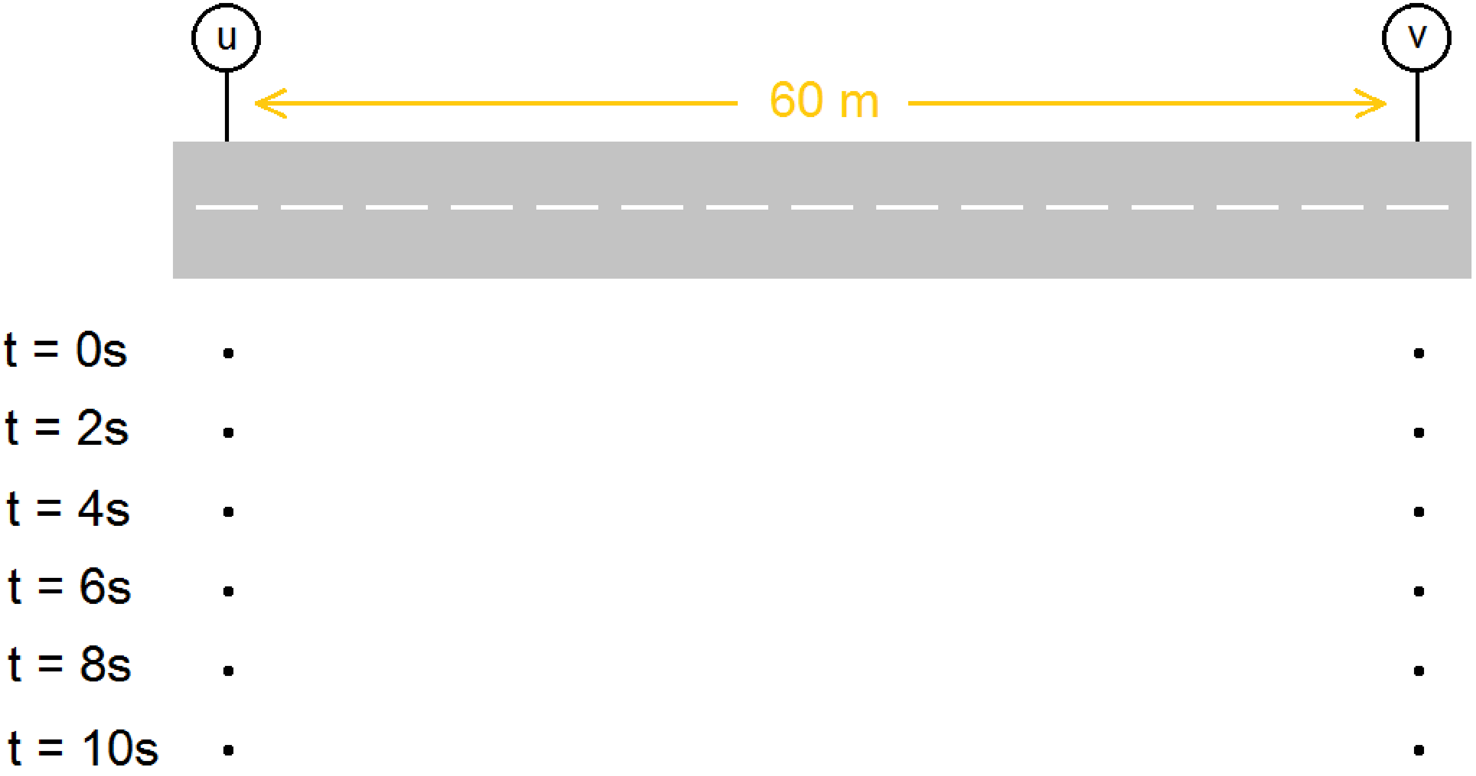
Time-Expanded Graphs

- g=(V,E) with |V|=n and |E|=m



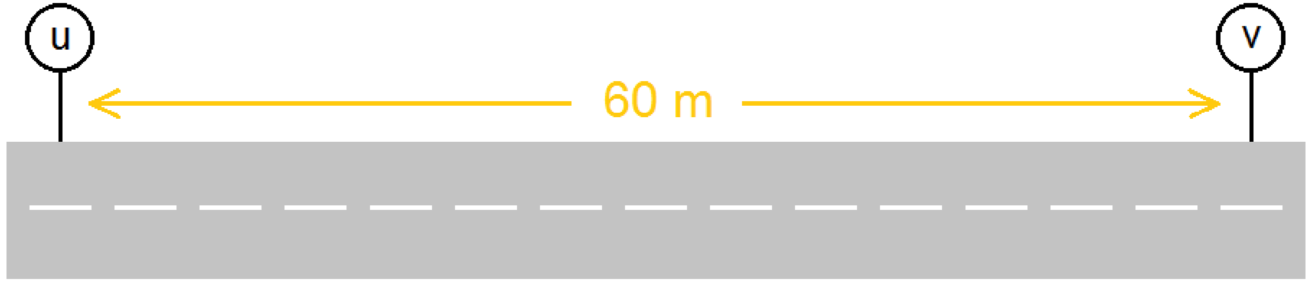
- For each e=(u,v) in E, the OpenStreetMap database provides :
  - Length
  - Maximum speed
  - Number of lanes
  - Direction
- Security distance (delta) = 2s between two cars (not depending on speed)

Time-Expanded Graphs



- T=10

Time-Expanded Graphs



(u,0) • • (v,0)

(u,2) • • (v,2)

(u,4) • • (v,4)

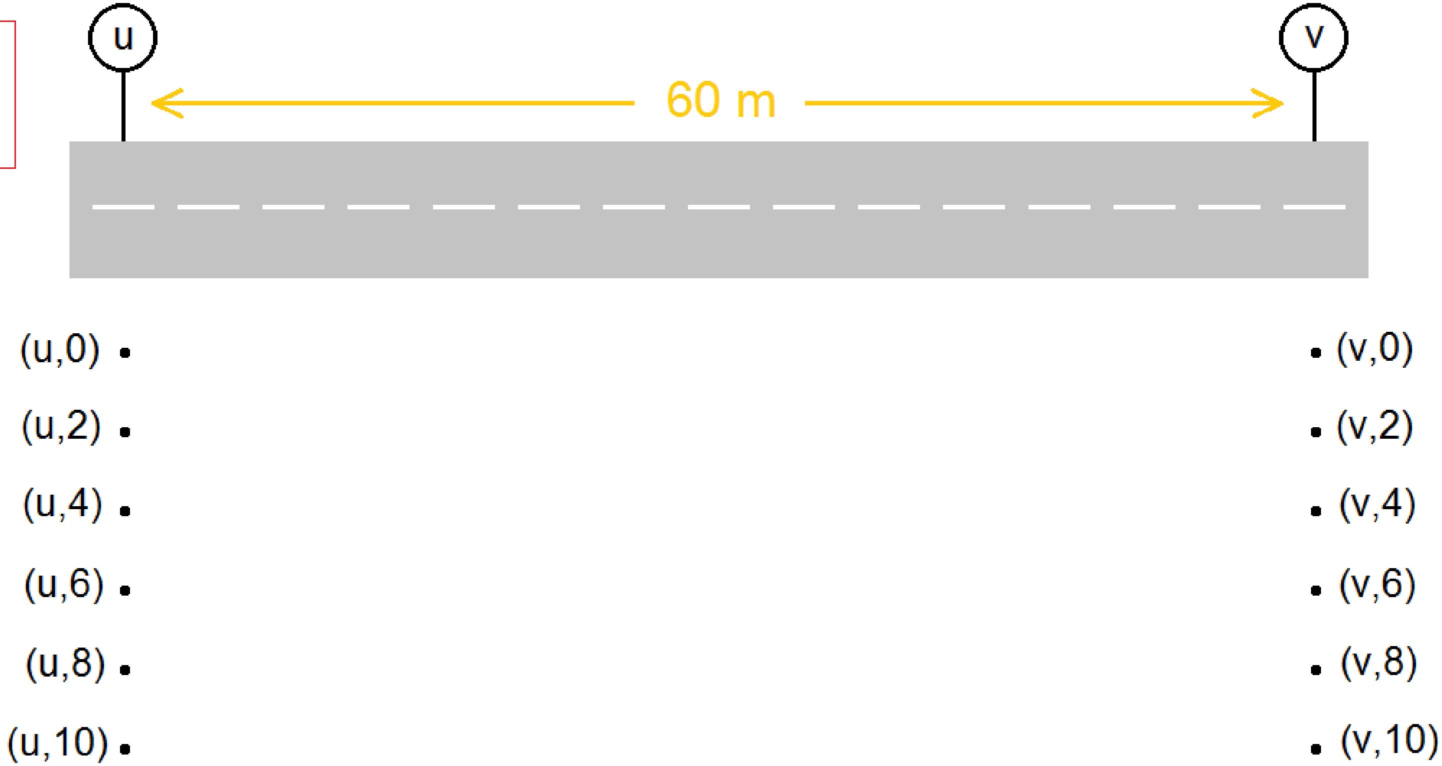
(u,6) • • (v,6)

(u,8) • • (v,8)

(u,10) • • (v,10)

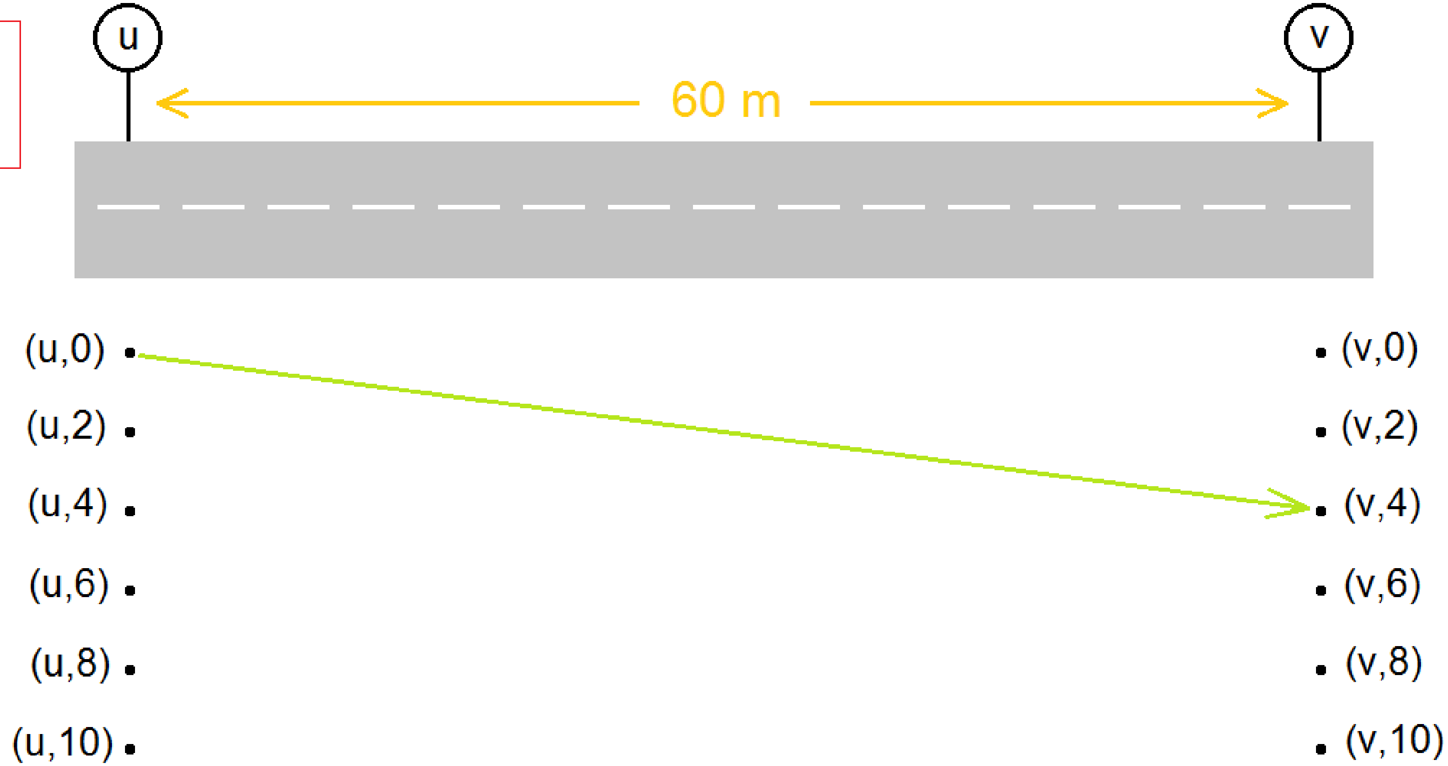
Time-Expanded Graphs

v = 54 km/h
= 15 m/s



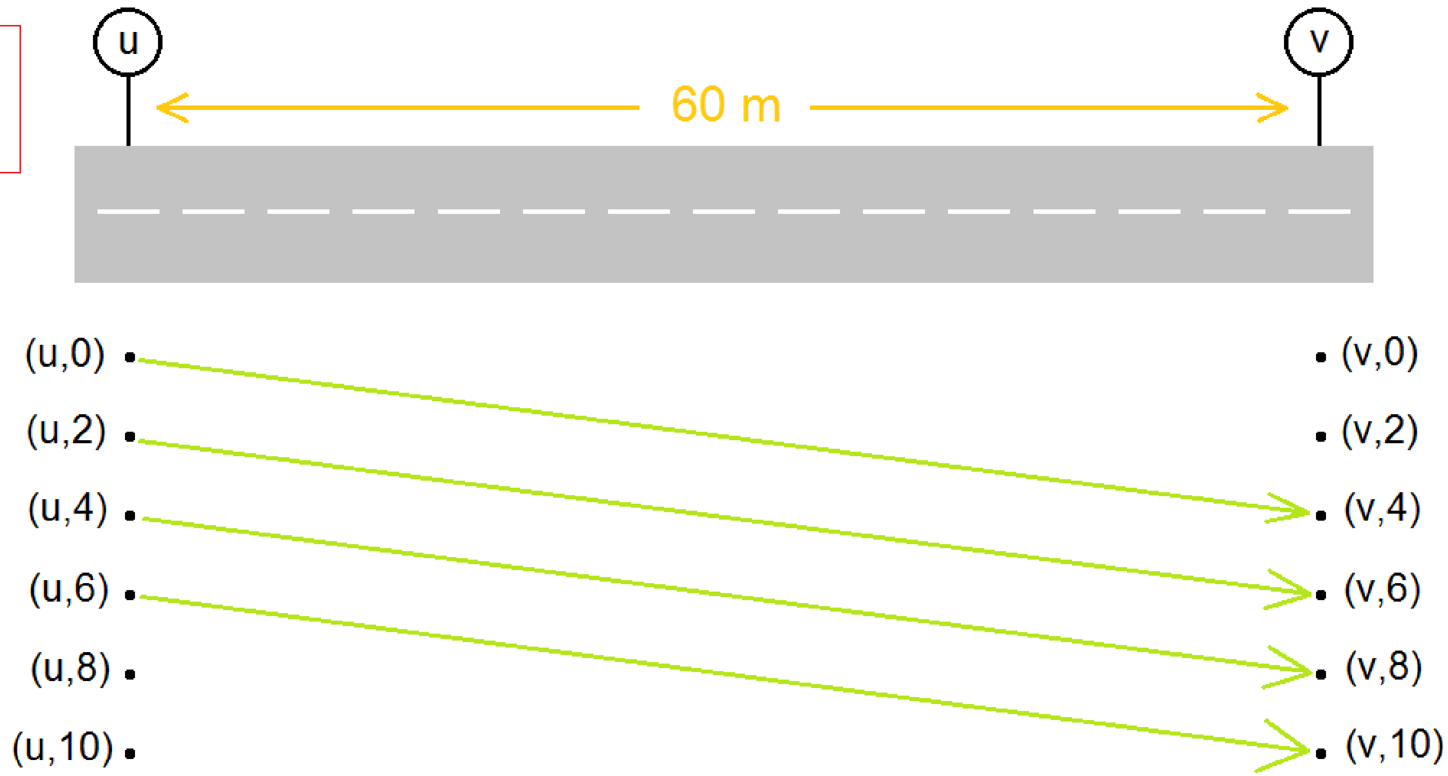
Time-Expanded Graphs

v = 54 km/h
= 15 m/s



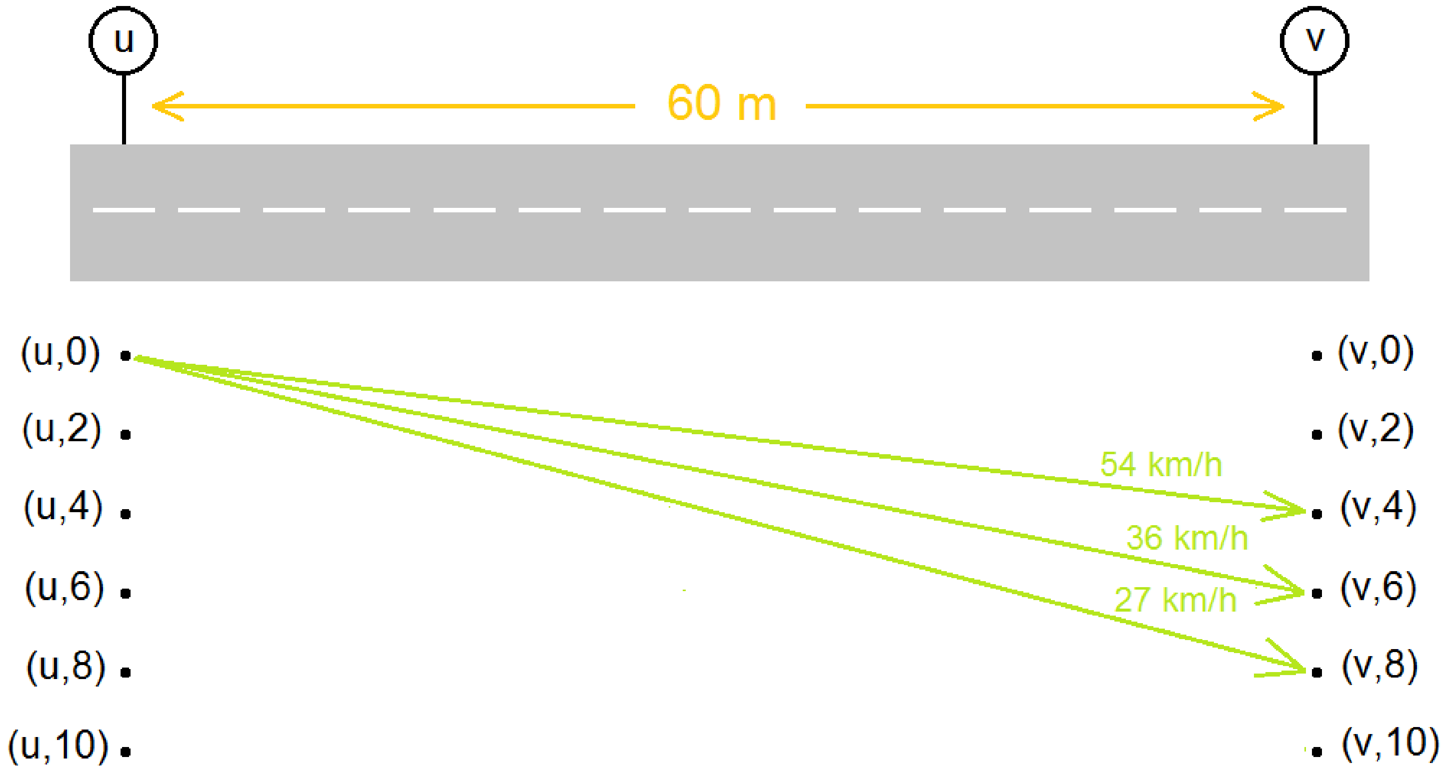
Time-Expanded Graphs

v = 54 km/h
= 15 m/s



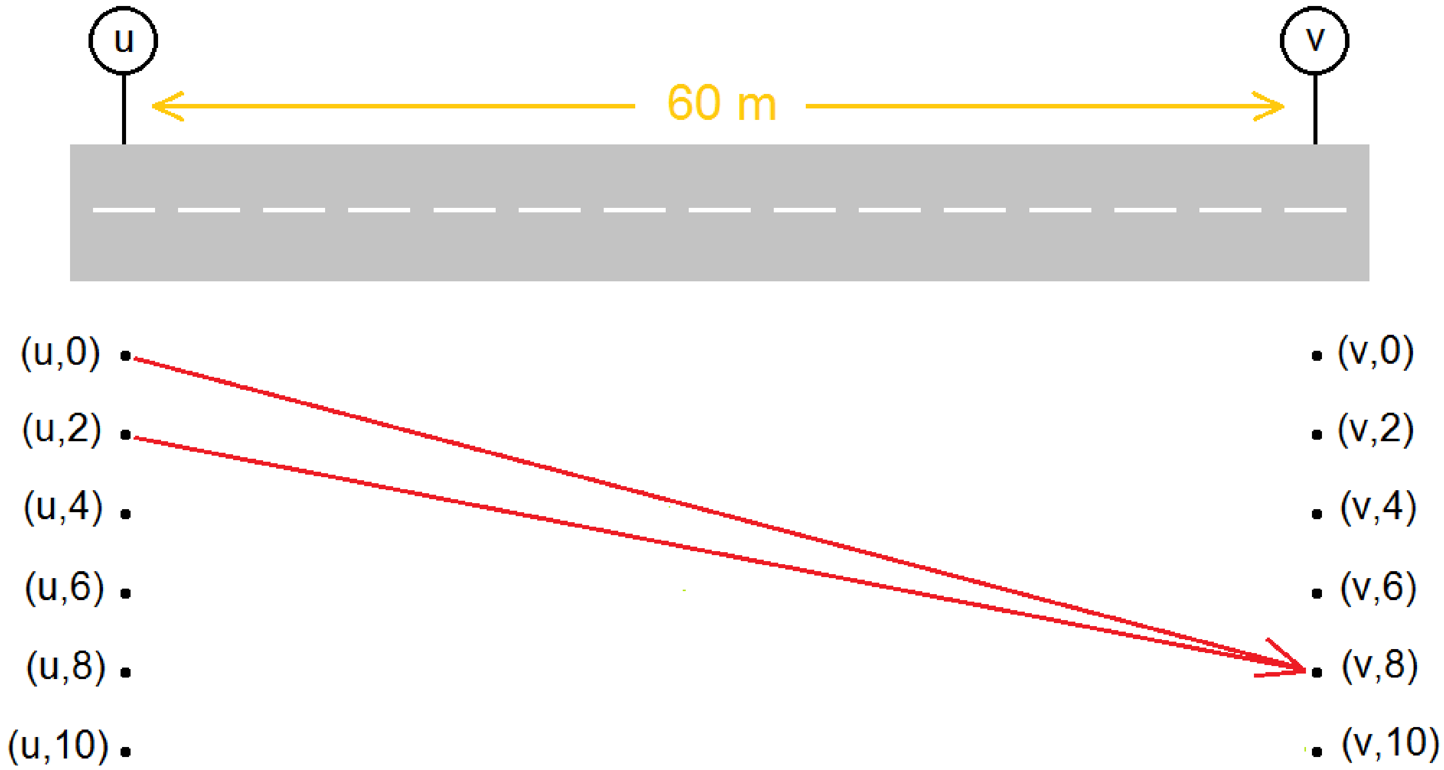
unrestricted

Time-Expanded Graphs

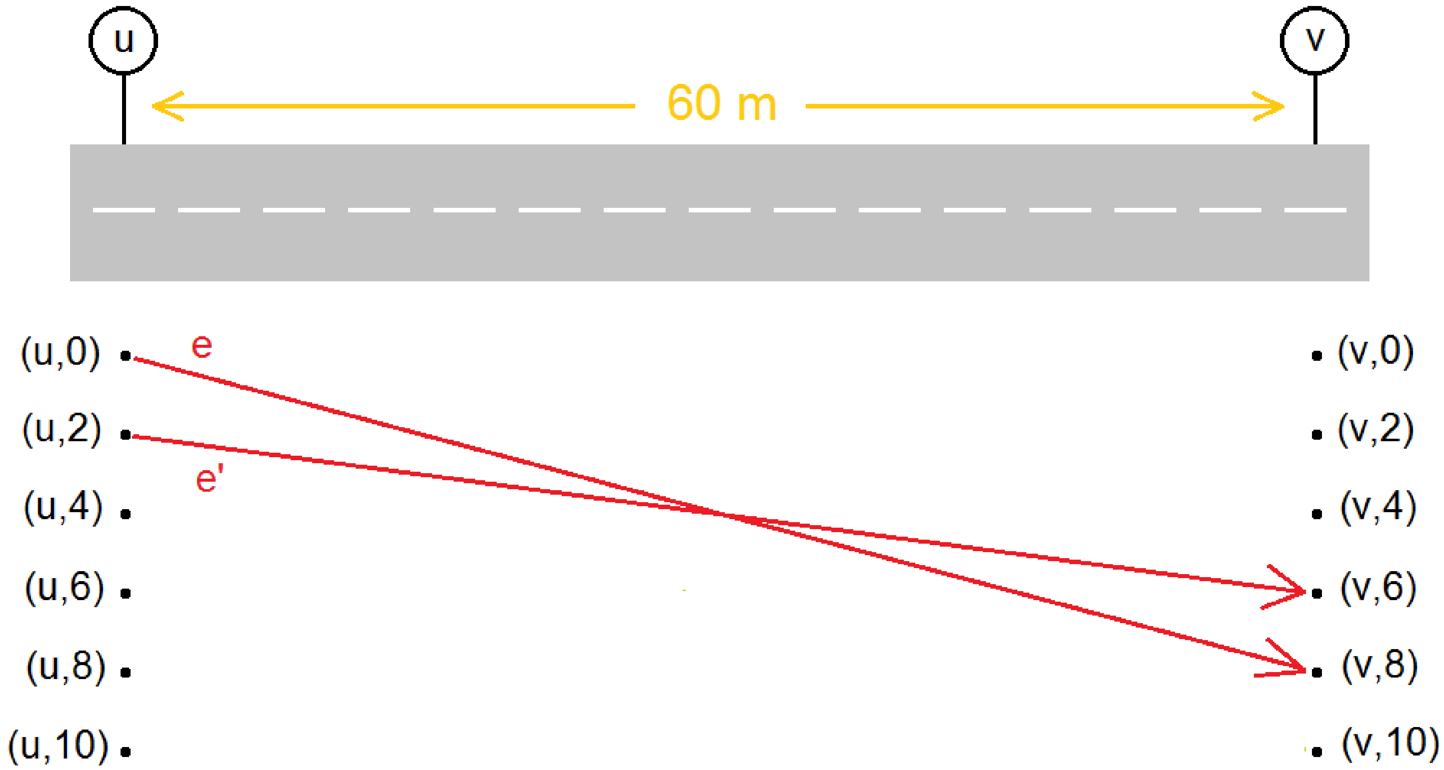


- Θ_{max}=3

Time-Expanded Graphs



Time-Expanded Graphs



- e’ overtakes e

Incompatibilities graph

- Let G be a Time-Expanded graph, we note $\mathcal{I}$ (G) the incompatibilities graph of G.
- *e* and *e'* in E(G) and incompatible if one of these conditions is verified:

  head(*e*) = head(*e'*)

  tail(*e*) = tail(*e'*)

  e overtakes *e'*

  e' overtakes *e*
- We have :
  - V($\mathcal{I}$ (G))=E(G) and (*e*,*e'*) belongs to E($\mathcal{I}$ (G)) if *e* and *e'* are incompatible.

Linear Programming formulation

$$\min \sum_{k\in\mathcal{K}} \sum_{P\in\mathcal{P}_k} \sum_{e\in P} c(e)\phi_P$$

under constraints

$$\sum_{k\in\mathcal{K}} \sum_{\substack{P\in\mathcal{P}_k \\ e\in P \text{ or } e'\in P}} \phi_P \leq 1, \forall (e,e') \in E(\mathcal{I}(G))$$

$$\phi_P \in \{0,1\}, \forall k \in \mathcal{K}, \forall P \in \mathcal{P}_k$$

- Where *K* is the set of request (a request *k* is represented by a origin vertex o_k in V(g) a starting time t_k and a destination vertex d_k in V(g)), and P_k is the set of paths from o_k to d_k in g (which corresponds to a set of pathes in G).

Clique covering problem

- Let C be a clique of $\mathcal{I}$ (G), we have :

$$\sum_{k \in \mathcal{K}} \sum_{\substack{P \in \mathcal{P}_k \\ P \cap C \neq \emptyset}} \phi_P \leq 1$$

- If we find a collection of cliques $\mathcal{C}$ such that every *e* in E($\mathcal{I}$(G)) belongs to a clique in $\mathcal{C}$ ($\mathcal{C}$ is a clique cover of $\mathcal{I}$(G)), then the two following statements are equivalent :

$$(1) \sum_{k \in \mathcal{K}} \sum_{\substack{P \in \mathcal{P}_k \\ P \cap C \neq \emptyset}} \phi_P \leq 1, \ \forall C \in \mathcal{C}$$

$$(2) \sum_{k \in \mathcal{K}} \sum_{\substack{P \in \mathcal{P}_k \\ e \in P \ or \ e' \in P}} \phi_P \leq 1, \forall (e, e') \in E(\mathcal{I}(G))$$

Clique covering problem

- <u>Hall Theorem :</u>

  Let G be a graph, we can find a clique cover of G such that

  $$|C| \leq \left\lfloor \frac{|V(G)|^2}{4} \right\rfloor$$

- Let C be a clique covering of $\mathcal{I}$ (G), thanks to Hall theorem, we have :

  $$|C| \leq \left\lfloor \frac{T^2\, m^2\, \Theta_{max}^2}{4} \right\rfloor \leq \frac{T^2\, m^2\, \Theta_{max}^2}{4}$$

- <u>Our result :</u>

  $$|C| \leq 2nT + m(T + \Theta_{max})\left(\frac{\Theta_{max}^2 \log(\Theta_{max})}{2} + \frac{5\Theta_{max}^2}{4}\right)$$

- Since T>> Θ_{max} , we reduced the number of constraints by a factor T/log(Θ_{max})

Linear Programming formulation using clique covering

$$\min \sum_{k\in\mathcal{K}} \sum_{P\in\mathcal{P}_k} \sum_{e\in P} c(e)\phi_P$$

under constraints

$$\sum_{k\in\mathcal{K}} \sum_{\substack{P\in\mathcal{P}_k \\ P\cap C\neq\emptyset}} \phi_P | \leq 1, \ \forall C \in \mathcal{C}$$

$$\phi_P \in \{0,1\}, \forall k \in \mathcal{K}, \forall P \in \mathcal{P}_k$$

Linear Programming formulation using clique covering : relaxing to fractionary flows

$$\min \sum_{k \in \mathcal{K}} \sum_{P \in \mathcal{P}_k} \sum_{e \in P} c(e)\phi_P$$

under constraints

$$\sum_{k \in \mathcal{K}} \sum_{\substack{P \in \mathcal{P}_k \\ P \cap C \neq \emptyset}} \phi_P \leq 1, \ \forall C \in \mathcal{C}$$

$$\phi_P \in [0,1] \ , \forall k \in \mathcal{K}, \forall P \in \mathcal{P}_k$$

- In fractionary flows, most values are integers.
- Apply column generation in the dual linear program (subroutine is a shortest path problem)

Experimentations and results

Experimentations and results

		vertex	edges	theta	constraints
Nice	Nice_reduced_4	26360	28491	4	142344000
	Nice_reduced_2	26405	28536	2	142587000
	Nice_reduced_3	26369	28500	3	142392600
	Nice_reduced_4_contracted_Infinity	5648	7779	56	30499200
	Nice_reduced_4_contracted_10	5721	7852	10	30893400
	Nice_reduced_4_contracted_8	5750	7881	8	31050000
	Nice_reduced_4_contracted_6	5812	7943	6	31384800
	Nice_reduced_4_contracted_5	5890	8021	5	31806000
	Nice_reduced_4_contracted_4	6007	8138	4	32437800
	Nice_reduced_3_contracted_Infinity	5648	7779	56	30499200
	Nice_reduced_3_contracted_10	5721	7852	10	30893400
	Nice_reduced_3_contracted_8	5750	7881	8	31050000
	Nice_reduced_3_contracted_6	5812	7943	6	31384800
	Nice_reduced_3_contracted_5	5889	8020	5	31800600
	Nice_reduced_3_contracted_4	6005	8136	4	32427000
	Nice_reduced_3_contracted_3	6205	8336	3	33507000
	Nice_reduced_2_contracted_Infinity	5648	7779	56	30499200
	Nice_reduced_2_contracted_10	5721	7852	10	30893400
	Nice_reduced_2_contracted_8	5750	7881	8	31050000
	Nice_reduced_2_contracted_6	5812	7943	6	31384800
	Nice_reduced_2_contracted_5	5889	8020	5	31800600
	Nice_reduced_2_contracted_4	6004	8135	4	32421600
	Nice_reduced_2_contracted_3	6200	8331	3	33480000
	Nice_reduced_2_contracted_2	6705	8836	2	36207000

- delta=2 and T=2700

Experimentations and results

		vertex	edges	theta	constraints
Toulouse	Toulouse_reduced_11	62746	70087	11	338828400
	Toulouse_reduced_2	62925	70266	2	339795000
	Toulouse_reduced_3	62774	70115	3	338979600
	Toulouse_reduced_4	62756	70097	4	338882400
	Toulouse_reduced_11_contracted_Infinity	20057	27398	45	108307800
	Toulouse_reduced_4_contracted_Infinity	20057	27398	45	108307800
	Toulouse_reduced_4_contracted_10	20077	27418	10	108415800
	Toulouse_reduced_4_contracted_8	20094	27435	8	108507600
	Toulouse_reduced_4_contracted_6	20136	27477	6	108734400
	Toulouse_reduced_4_contracted_5	20195	27536	5	109053000
	Toulouse_reduced_4_contracted_4	20315	27656	4	109701000
	Toulouse_reduced_3_contracted_Infinity	20057	27398	45	108307800
	Toulouse_reduced_3_contracted_10	20077	27418	10	108415800
	Toulouse_reduced_3_contracted_8	20094	27435	8	108507600
	Toulouse_reduced_3_contracted_6	20136	27477	6	108734400
	Toulouse_reduced_3_contracted_5	20195	27536	5	109053000
	Toulouse_reduced_3_contracted_4	20314	27655	4	109695600
	Toulouse_reduced_3_contracted_3	20601	27942	3	111245400
	Toulouse_reduced_2_contracted_Infinity	20057	27398	45	108307800
	Toulouse_reduced_2_contracted_10	20077	27418	10	108415800
	Toulouse_reduced_2_contracted_8	20094	27435	8	108507600
	Toulouse_reduced_2_contracted_6	20136	27477	6	108734400
	Toulouse_reduced_2_contracted_5	20195	27536	5	109053000
	Toulouse_reduced_2_contracted_4	20312	27653	4	109684800
	Toulouse_reduced_2_contracted_3	20601	27942	3	111245400
	Toulouse_reduced_2_contracted_2	21390	28731	2	115506000

- delta=2 and T=2700

Conclusion and further work

Conclusion and further work

- Using clique covering makes problems tractable.

- Incorporate multi-lane roads.

- Smarter crossroads management (increase the number of constraints).

- Use divide and conquer (Arora's techniques for TSP : recursive partitionning and dynamic programming).

thank you