
# Road Network Upgrade Planning

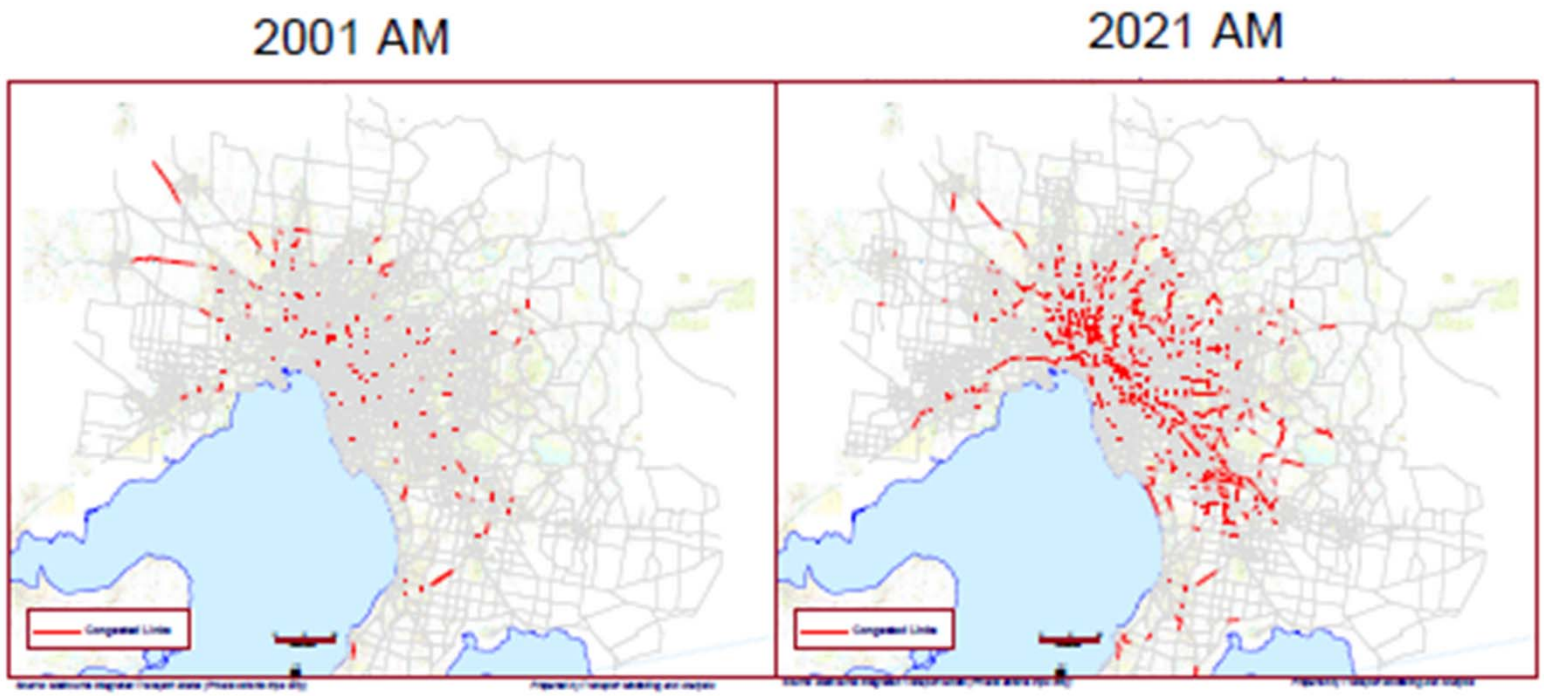
Australian Government
Department of Broadband, Communications and the Digital Economy
Australian Research Council

NICTA Funding and Supporting Members and Partners

# Traffic Assignment and Road Network Upgrade Planning

Alex Stivala (NICTA)
Supervised by
Prof. Peter J. Stuckey (NICTA, Melbourne Uni.)
& Prof. Mark Wallace (NICTA, Monash Uni.)

# CONGESTION OUTPUT

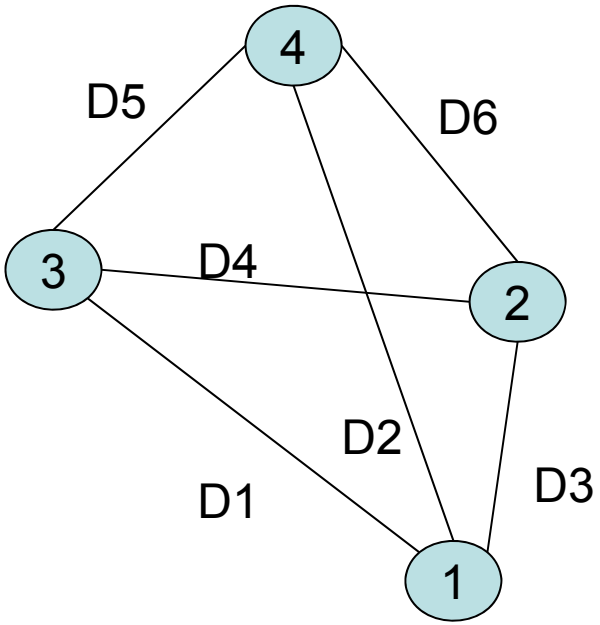



**Integrated Transport Modelling**
**Fotios Spiridonos**

# Introduction


- We want to choose the optimal subset of a set of potential road upgrades (added lanes, new roads)
- We need a benefit and a cost for each upgrade
- The cost is just the cost of building it
- How do we measure the benefit?

# Demands and Capacities

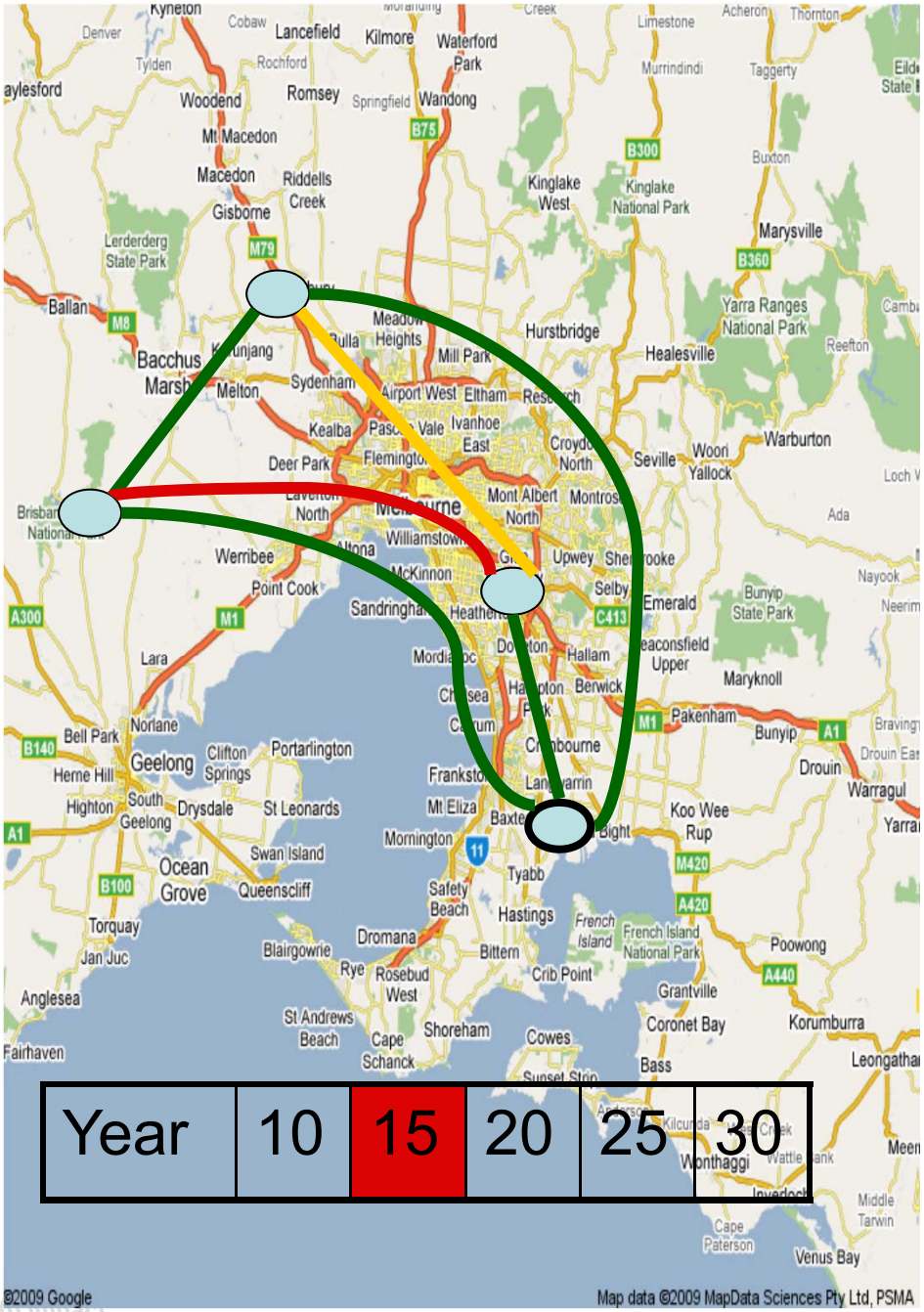



| Demand | 2010 | 2015 | 2020 | 2025 | 2030 |
|---|---|---|---|---|---|
| D1 | 10 | 12 | 16 | 20 | 24 |
| D2 | 10 | 12 | 16 | 20 | 24 |
| D3 | 10 | 12 | 16 | 20 | 24 |
| D4 | 14 | 16 | 18 | 20 | 22 |
| D5 | 14 | 16 | 18 | 20 | 22 |
| D6 | 14 | 16 | 18 | 20 | 22 |

| Capacity | Current | Option | New |
|---|---|---|---|
| A | 15 | 1 | 25 |
| B | 15 | 2 | 20 |
| C | 15 | 3 | 30 |
| D | 15 | 4 | 25 |
| E | 15 | 5 | 30 |
| F | 15 | 6 | 30 |
| G | 15 | 7 | 25 |
| H | 15 | 8 | 25 |
| J | 15 | 9 | 20 |
| K | 15 | 10 | 25 |

# Investment Options

| Option | 10 | 15 | 20 | 25 | 30 |
|---|---|---|---|---|---|
| A | | | X | | |
| B | X | | | | |
| C | | | | | |
| D | | | | | |
| E | | X | | | |
| F | | X | | | |
| G | | | | | |
| H | | | | X | |
| J | | | | | X |



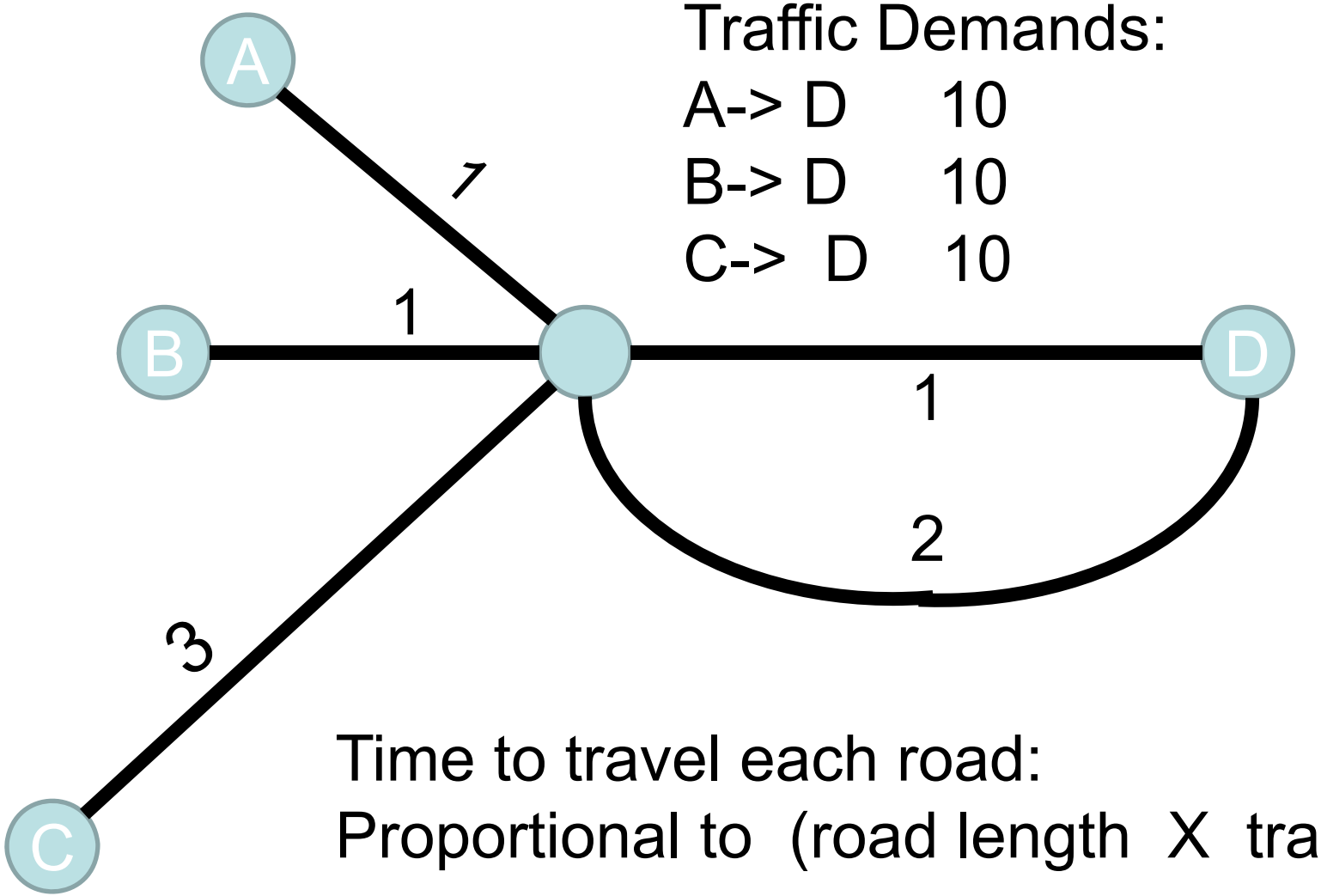
# Vehicle Hours Travelled (VHT)


- One measure of a road network is Vehicle Hours Travelled (VHT)
  - sum of all hours travelled by all vehicles
  - for some fixed demand for travel between origins and destinations
  - at some fixed time period (morning peak for example)
- Assess the benefit of a road upgrade by the reduction in VHT it induces
  - measure changes in travel time caused by changes in congestion patterns
  - ***not*** vehicle miles (or kilometres) travelled
  - ***not*** just geographically shorter or longer routes.

# The Traffic Assignment Problem (TAP)


- Given the road network and demands (trips between origins and destinations)
- *Predict the route choices of road users, giving the traffic flows on each road.*
- Assume that:
  - Time taken to travel a road is a (continuous non-decreasing) function of the volume of traffic on the road
  - Static assignment: the travel demands are given and not affected by travel times
  - Each traveller chooses the route minimizing his/her travel time
  - User Equilibrium is achieved when no traveller can reduce his/her travel time by unilaterally choosing a different route

# Simplified Example




Traffic Demands:

A-> D 10

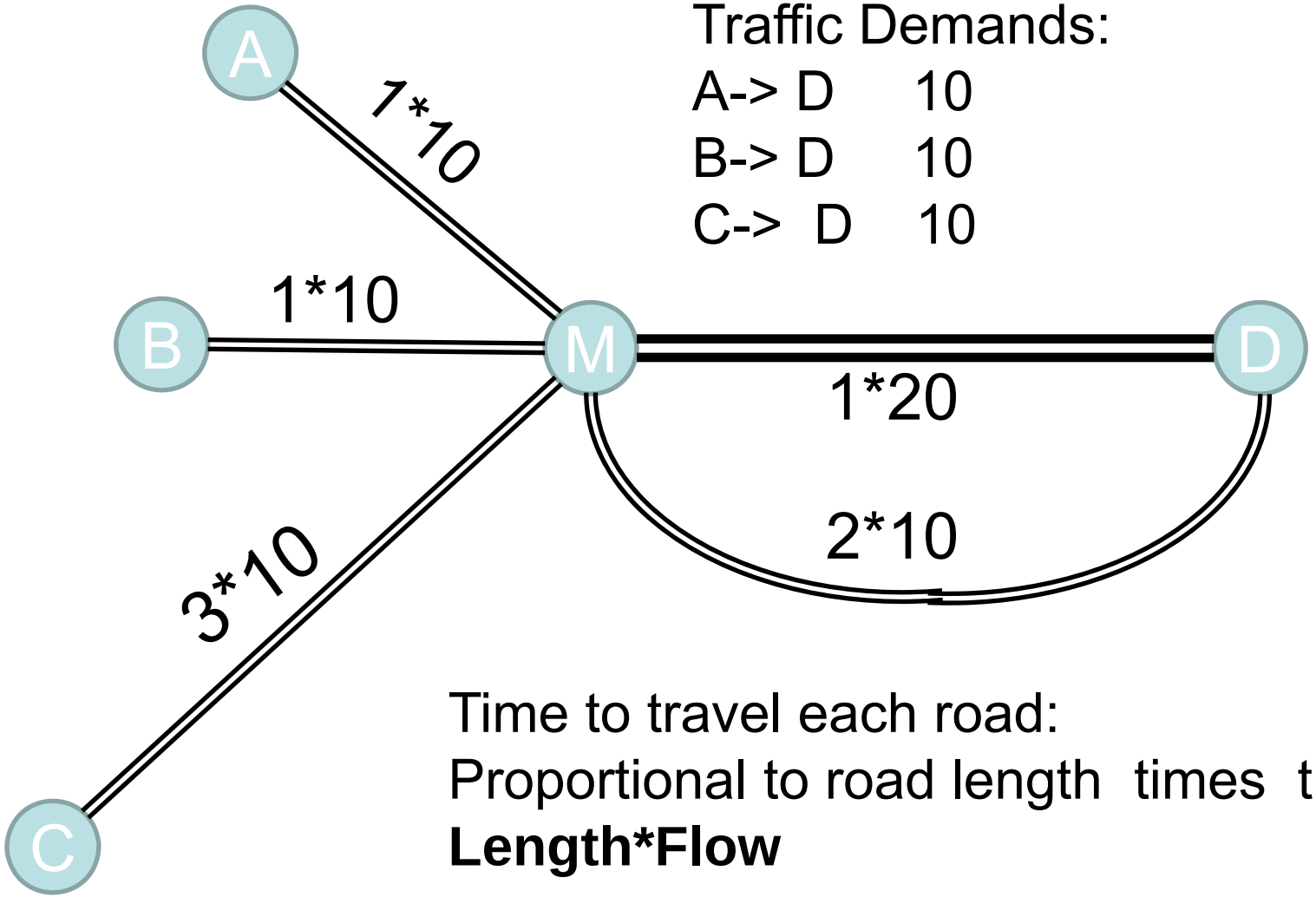
B-> D 10

C-> D 10

Time to travel each road:

Proportional to (road length X traffic flow)

# Simplified Example – User Equilibrium




Traffic Demands:

A-> D 10

B-> D 10

C-> D 10

Time to travel each road:

Proportional to road length times traffic flow:

**Length*Flow**

# Link travel time function

NICTA

The Bureau of Public Roads (BPR) function is commonly used for the latency on a link given its flow (as ratio of current volume to capacity)

$$l_a(f_a) = t_a \left(1 + \alpha \left(\frac{f_a}{Q_a^{\max}}\right)^{\beta}\right)$$

where α and β are parameters, for example α = 0.15 and β = 4.0.

# User Equilibrium formulation of TAP (1) — Definitions


- The road network is a graph $G = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of arcs (links representing road segments).
- $Q_{|\mathcal{O}|\times|\mathcal{D}|}$ is the O-D demand matrix such that $(q_{rs})$ gives the number of vehicles from origin $r \in \mathcal{O}$ to destination $s \in \mathcal{D}$
- $f \in \mathbb{R}^{|\mathcal{A}|}$ is the flow (the volume of cars on each link)
- $l \in \mathbb{R}^{|\mathcal{A}|}$ is the travel time (the latency on each link)
- We denote the flow on path $k$ connecting an O-D pair $(r, s) \in \mathcal{O} \times \mathcal{D}$ by $h_{rs}^k$, where $k$ is in $\mathbb{P}_{rs}$, the set of paths between origin $r$ and destination $s$.
- $\delta_{a,k}^{rs}$ is the indicator variable

$$\delta_{a,k}^{rs} = \begin{cases} 1, & \text{if link } a \text{ is on path } k \text{ between O-D pair } (r,s) \\ 0, & \text{otherwise} \end{cases}$$

# User Equilibrium formulation of TAP (2) — Convex quadratic program


$$\min \sum_{a \in \mathcal{A}} \int_0^{f_a} l_a(x)dx$$

subject to

$$\sum_{k \in \mathbb{P}_{rs}} h_{rs}^k = q_{rs} \qquad \forall (r,s) \in \mathcal{O} \times \mathcal{D}$$

$$h_{rs}^k \geq 0 \qquad \forall k \in \mathbb{P}_{rs}, \forall (r,s) \in \mathcal{O} \times \mathcal{D}$$

$$f_a = \sum_{(r,s) \in \mathcal{O} \times \mathcal{D}} \sum_{k \in \mathbb{P}_{rs}} h_{rs}^k \delta_{a,k}^{rs} \qquad \forall a \in \mathbb{A}$$

# Solution by Frank-Wolfe Algorithm


- Iteratively solve
  - a set of shortest-path problems (the “all-or-nothing” solutions) to find search direction
  - followed by a linear program to find search step size
- Terminate on some criteria, often the relative gap:

$$\frac{\sum f \cdot l(f) - \sum f_{aon} \cdot l(f)}{\sum f \cdot l(f)}$$

  - Relative gaps on the order of $10^{-5}$ (or smaller) are usually considered to indicate a very good solution.
  - This requires thousands of iterations, in total several hours for a reasonable sized city

# Application of Frank-Wolfe algorithm


- Although commercial products (OBA, LUCE) are claimed to be cleverer and faster
- Frank-Wolfe is still used often in practice (and in commercial products) as it is relatively simple, has low memory requirements, and has capacity to be straightforwardly parallelized
- Our algorithm is based on Frank-Wolfe

# Naïve algorithm for choosing an optimal set of road upgrades


- Solve the TAP for every possible subset of the set of proposed road upgrades
- For $N$ potential upgrades requires (worst-case) $2^N$ TAPs to be solved

*Not good enough to just solve TAP for each individual upgrade and add the resulting changes in VHT, they can interact (we need at least to solve each individually though)*

# Improved algorithm for choosing an optimal set of road upgrades


- Compute the VHT change for each individual upgrade
  - This basic computation is unavoidable
- Compute the VHT change for each pair of upgrades
  - Neglecting interactions that only appear between 3 or more individual upgrades
- Solve a quadratic number of TAP problems
  - still a quadratic number of TAP problems, which is also impractical

# Proposed algorithm for choosing an optimal set of road upgrades


- Compute VHT for each individual upgrade by solving TAP on modified network
- Estimate which pairs of upgrades will have a significant interaction
- Solve TAP on each such pair to get the pairwise VHT change
- Solve our constrained optimization problem with the individual VHT changes and adjustments for pairwise interactions

# Implementation


- We implemented the Frank-Wolfe algorithm in C
  - At each iteration, the single-source shortest path problem is solved for each origin
  - accounts for more than 90% of CPU time
- We solved a TAP for each upgrade

# Finding shortest paths efficiently


- “Everybody knows” that Dijkstra’s algorithm (with Fibonacci heap) is among the asymptotically fastest single-source shortest path algorithms
- But in practice, especially on road networks, the “label-correcting” algorithms take less time to run
- Such algorithms do not determine the shortest path for each node at a time, but are iterative and only have the optimal values for all nodes on completion
- These algorithms use much simpler data structures giving less overhead to find a pivot node, at the expense of needing multiple iterations
- We used the d’Esopo-Pape algorithm with the “large label last” (LLL) modification
- The graph (road network) is stored efficiently in memory in “packed adjacency list” format, which is advantageous for memory locality (caching)

# Performance of our implementation on the TAP

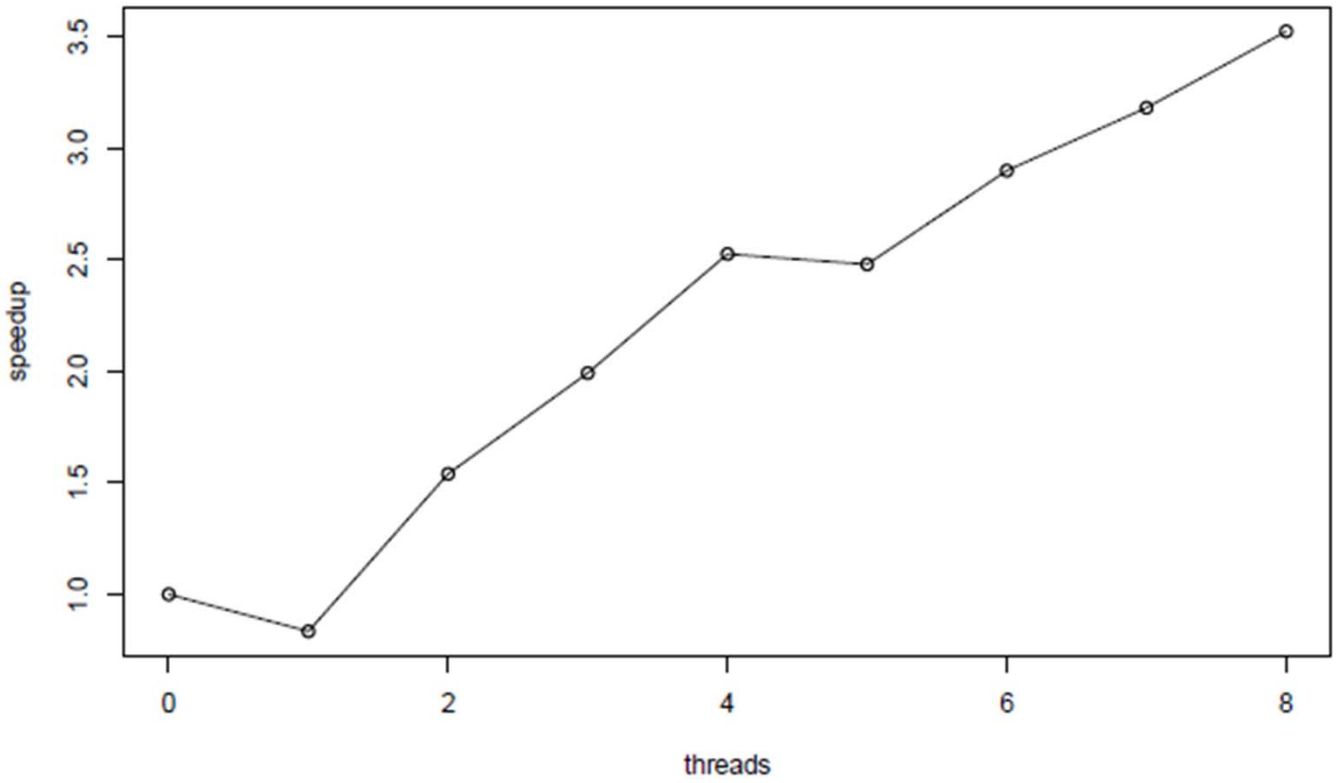

| Platform | Algorithm | Time (s) |
|---|---|---|
| CPU | d'Esopo-Pape LLL | 1.759 |
| GPU | d'Esopo-Pape | 1.890 |
| CPU | SLF LLL | 1.958 |
| CPU | d'Esopo-Pape | 3.017 |
| CPU | SLF | 5.965 |
| CPU | Bellman-Ford | 12.075 |
| CPU | Dijkstra | 14.541 |

Table: Times for 1790 O-D pair shortest path computations in Chicago Regional data. Each time in seconds is the average of 10 runs. CPU is 2.3 GHz quad core Opteron, 4 GB RAM. GPU is NVIDIA GTX 280 (1 GB RAM, 30 multiprocessors, 1.3 GHz).

# Parallel computation


- Run shortest path from each origin in parallel
  - update the link volume vectors in parallel (with appropriate lock-free synchronization)
  - use threads (GPU?)
- Solve TAP for each upgrade in parallel
  - use MPI on a suitable cluster

# Multithreading on a standard PC




4-core "hyperthreading". 5.4 hours unthreaded, 1.5 hours 8 threads.

# A model to determine the optimal subset of upgrades


$$\max \left( \sum_{1 \le i < j \le N} y_i y_j d_{ij} + \sum_{1 \le i \le N} y_i v_i \right)$$

subject to

$$\sum_{1 \le i \le N} y_i c_i \le B$$

- $v_i$ is the VHT change for each individual upgrade,
- $d_{ij} = \Delta\text{VHT}_{ij} - (v_i + v_j)$ gives the difference between the pairwise change and the sum of individual changes, where $\Delta\text{VHT}_{ij}$ is the VHT change for each pair of upgrades
- $c_i$ is the cost of upgrade $i$
- $B$ is the total budget
- $y_i \in \{0, 1\}$ is an indicator variable such that $y_i = 1$ if upgrade $i$ is in the selected subset.

# Test data


- Publically available data for Chicago regional road network
  - used in reported experiments
- Data provided by Victorian Dept of Transport
  - currently aligning node and link data for locally more meaningful experiments

# Experimental Data


- Chicago Regional road network from Dr Hillel Bar-Gera's Transportation Test Problems website
  - 39 018 links, 12 982 nodes and 1 790 zones
  - http://www.bgu.ac.il/~bargera/tntp/

| Project Id | Type | Cost ($000s) | Description |
|---|---|---|---|
| 03-02-9005 | capacity upgrade | 999 | add lanes on I-90 to I-294 |
| 07-06-0014 | new road | 472 | from Cottage Grove Ave. to Mark Collins Dr. |
| 07-94-0027 | new road | 700 | Joe Orr Rd. extension to Burnham Ave. |
| 07-96-0013 | new road | 748 | Joe Orr Rd. extension to Sheffield Ave. |
| 03-95-0001 | new road | 4000 | Elgin-O'Hare Expwy to Lake St. |
| 03-96-0024 | capacity upgrade | 1000 | widen lanes on Villa St./Lake St. |
| 03-03-0101 | capacity upgrade | 465 | add lanes on Meacham Rd. |
| 07-97-0055 | capacity upgrade | 4000 | add lanes on I-57 from I-80 to I-294 |

Table: Potential upgrades, based on data from the Chicago Metropolitan Agency for Planning Transportation Improvement Program website http://www.cmap.illinois.gov/tip

# Optimal Solution

- We solve the TAP for all subsets of the set of upgrades
  - 255 TAPs (8 upgrades)
- We measure the relative error

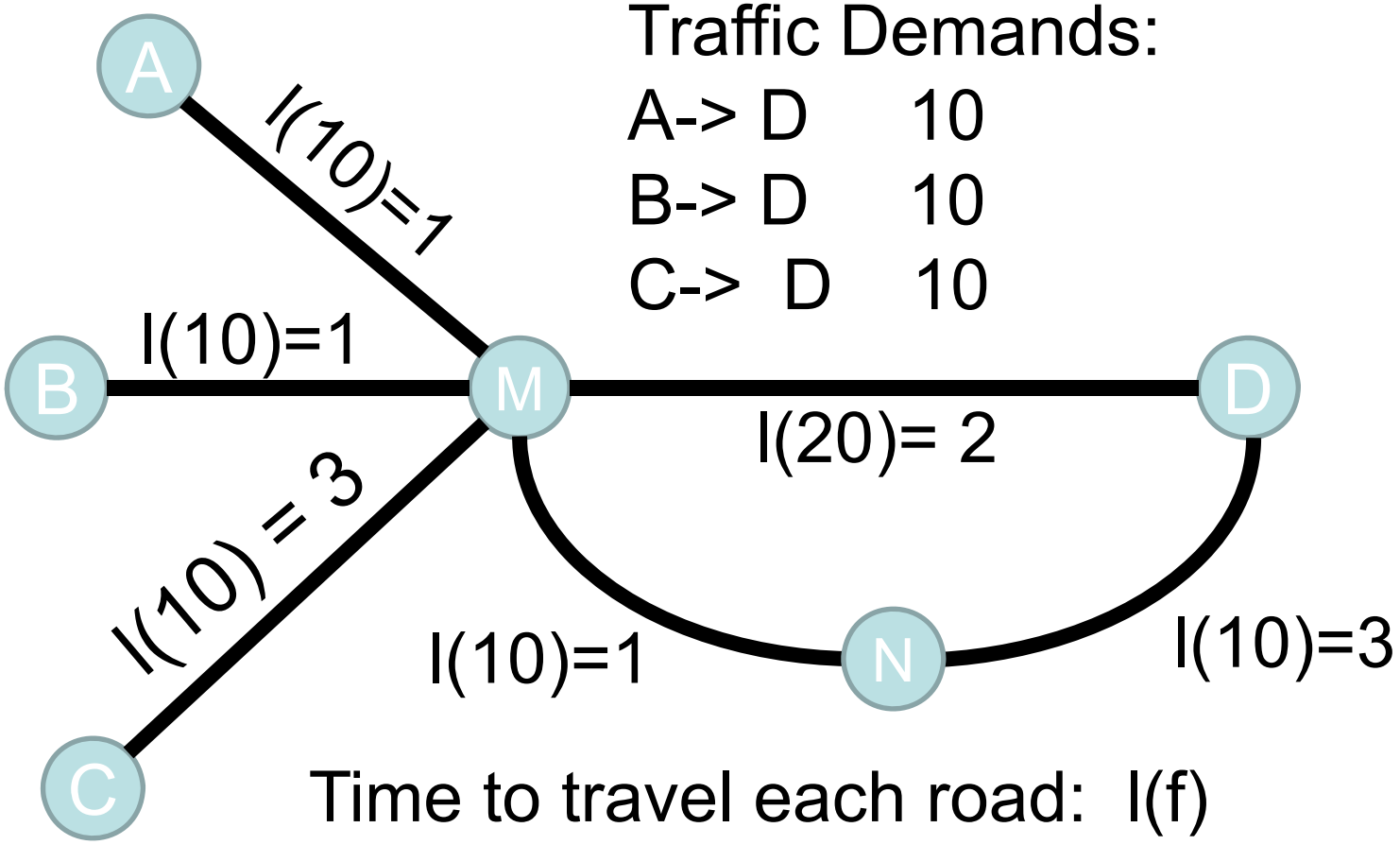
$$\frac{\left|\sum_{i \in S} v_i + \sum_{i \neq j \in S} d_{ij} - \Delta \mathrm{VHT}_S\right|}{\Delta \mathrm{VHT}_S}$$

between the actual VHT computed for each subset and the estimates computed by only using pairwise interactions

# Results: Relative Errors


- All pairwise upgrade VHT values computed
  - 36 TAPs to solve
  - 0.76%
- Individual upgrade VHT values computed
  - 8 TAPs to solve
  - 4.0%
- Individual upgrades plus the one "significant" pairwise VHT value
  - 9 TAPs to solve
  - 1.0%

# Predicting which upgrades need pairwise solutions

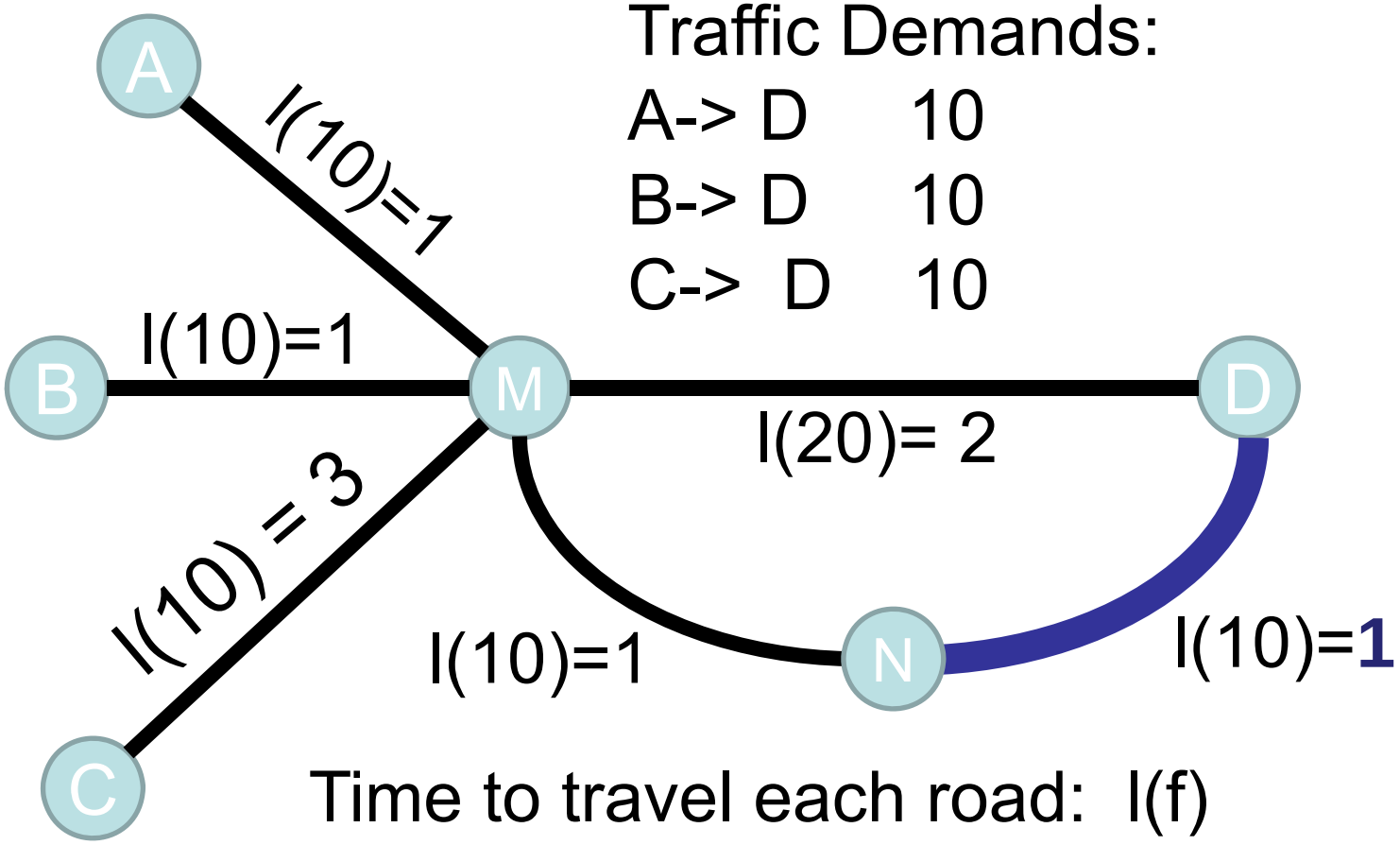



Traffic Demands:

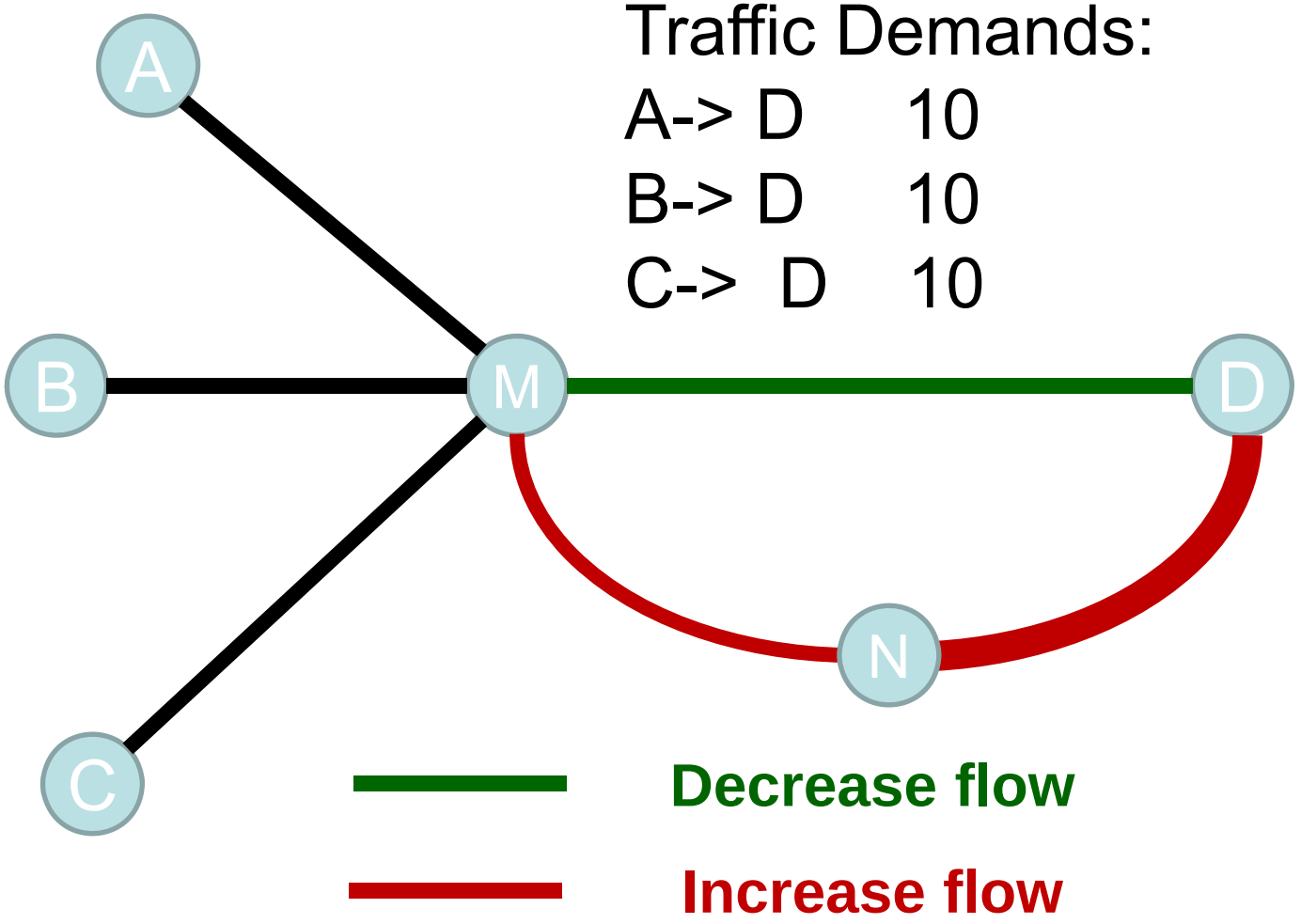
| | |
|---|---|
| A-> D | 10 |
| B-> D | 10 |
| C-> D | 10 |

Time to travel each road: l(f)
(latency, dependent on flow)

# Predicting which upgrades need pairwise solutions – ***reduce latency link of ND***



# Predicting which upgrades need pairwise solutions – ***reduce latency link of ND***



# Interaction between two upgrades – naïve hypotheses


- Sets of links with changed flow due to each upgrade disjoint
  - VHT change is the sum of VHT changes due to individual upgrades?
- Links with increased flow due to one upgrade and decreased flow due to the other
  - VHT change less than the sum of individual changes?
- Links with increased flow due to both upgrades
  - VHT change greater than the sum of individual changes?

# Experiments on 1000 randomly generated sets of upgrades


- Sets of links with changed flow due to each upgrade disjoint
  - VHT change is the sum of VHT changes due to individual upgrades?
  - *Largely confirmed (but not 100%)*
- Links with increased flow due to one upgrade and decreased flow due to the other
  - VHT change less than the sum of individual changes?
  - *Unconfirmed*
- Links with increased flow due to both upgrades
  - VHT change greater than the sum of individual changes?
  - *Unconfirmed*

# Predicting which upgrades need pairwise solutions – further hypotheses


- We can use certain a priori knowledge of the upgrades and also information from solving the TAP individually:
  - Distance (geographic) between the upgrades
  - Number of links that have significant flow changes in both upgrades (either reinforcing or cancelling)
  - Number of shortest paths that pass through both upgrades (this requires extra work since we are using here a link not path based method)
- And use machine learning to predict which upgrades will have a pairwise VHT change significantly different from the sum of individual VHT changes

# Learning Outcomes

***The best predictor of the change in VHT due to a pair of upgrades is the sum of the VHT changes due to the individual upgrades***

***For optimised planning, find the optimum combination of individual upgrades, for each year planned into the future, and then test the combination.***

# Conclusions

- Solving the optimal upgrade subset problem exactly requires an infeasibly large number (exponential) of TAP problems to be solved
- But we can reduce this by ignoring interactions, and running a TAP on the optimum combinations to check the change in VHT
- *We have gone "from simulation to optimization", using the results from the TAP solutions as input to a model giving the optimal subset of road upgrades.*
- Open problem: in case the check on the optimum combination reveals unexpected interactions, which alternative combinations should we check?