Routing Directions: Keeping it Fast and Simple

Dimitris Sacharidis [1] Panagiotis Bouros [2]

[1]Institute for the Management of Information Systems, R.C. Athena, Greece

[2]Department of Computer Science, Humboldt-Universität zu Berlin, Germany

motivation

the **fastest route** is not always the best option for routing directions

- tourist asking for walking directions to a landmark
  - unfamiliar neighborhood: simple instructions
- emergency evacuation plan
  - distress situation: concise and clear to follow instructions

the simplest route minimizes **complexity** of the routing directions in terms of turns (road changes)

- for simplicity, all turns have equal cost
- the generalization to non-uniform costs (right/left turns, orientation changes, etc.) is straightforward

goal of this work: study the trade-off between fastest and simplest routes

terminology

- a road r is a sequence of distinct nodes (road intersections)
- the road network is a directed graph $G_R(V, E)$
  - V is the set of nodes
  - E contains an edge $e_{ij} = (n_i, n_j)$ where n_i, n_j are consecutive nodes in some road.
- a route ρ is a sequence of nodes defined by a path on G_R
- the length $L(\rho)$ of a route is the sum of lengths of its edges
- the complexity $C(\rho)$ of a route is the number of turns (road changes)

note:

- path is a general term referring to a sequence of vertices of **some** graph
- a route is a path on the road network graph G_R

terminology

- a fastest route has the smallest length
- a near-fastest route has length at most $(1+\varepsilon)$ times the smallest length
- a simplest route has the lowest complexity
- a near-simplest route has complexity at most $(1+\varepsilon)$ times the lowest complexity

ε is a user defined parameter

note:

- a fastest route is essentially a **shortest path** on the road network graph G_R

problem definition

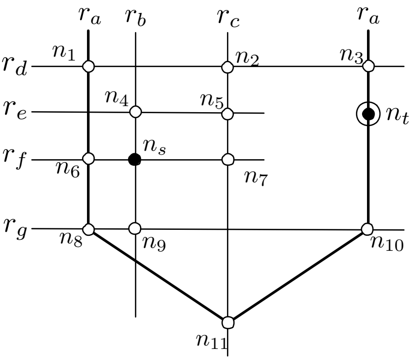
given a source node n_s and a target n_t

- **fastest simplest route** (FSR): find a route that has the smallest length among all simplest routes
- **simplest fastest route** (SFR) find a route that has the lowest complexity among all fastest routes
- **fastest near-simplest route** (FNSR): find a route that has the smallest length among all near-simplest routes
- **simplest near-fastest route** (SNFR) find a route that has the lowest complexity among all near-fastest routes

FSR is studied in a past work

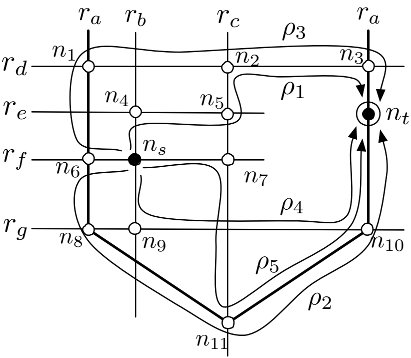
SFR, FNSR, SNFR are studied for the first time (to the best of our knowledge)

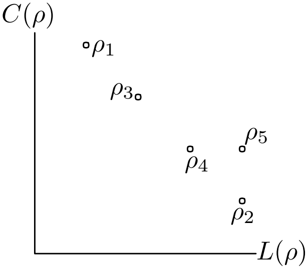
example



7 roads, 12 nodes

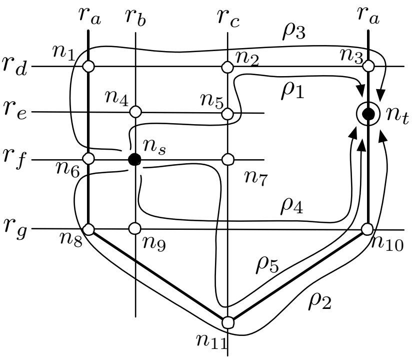
example

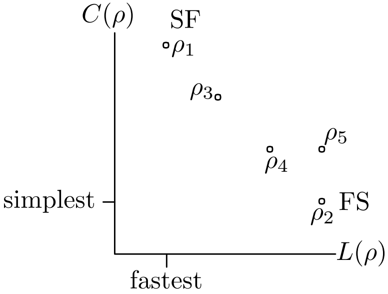




7 roads, 12 nodes , 5 routes from n_s to n_t

example

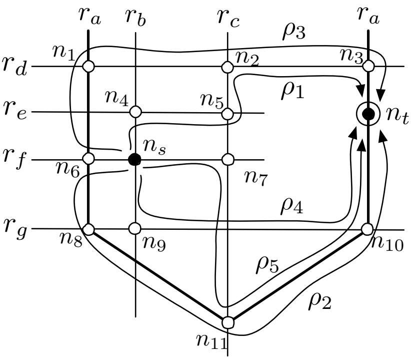


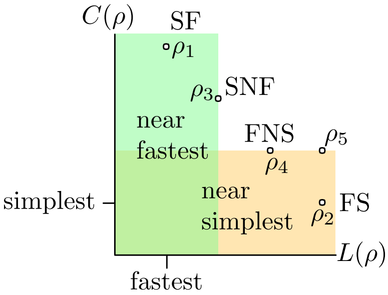


7 roads, 12 nodes , 5 routes from n_s to n_t

ρ_1 is simplest fastest, ρ_2 is fastest simplest

example





7 roads, 12 nodes , 5 routes from n_s to n_t

ρ_1 is simplest fastest, ρ_2 is fastest simplest

for $\varepsilon = 1$, ρ_3 is simplest near-fastest, ρ_4 is fastest near-simplest

fastest simplest route: baseline approach (BSL)

- intersection graph $G_I(R, I)$
  - R is the set of roads
  - I contains an **intersection** (n_x, r_i, r_j), where node n_x belongs to roads r_i, r_j
- multiple source/target roads associated with the source/target node n_s/n_t

BSL algorithm

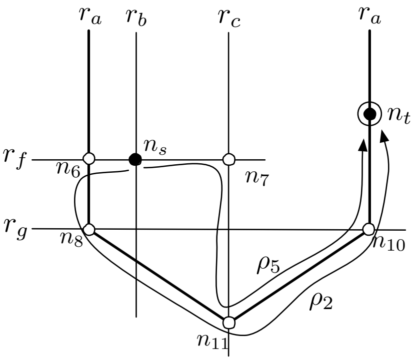
key idea: a simplest route is related to a shortest path on G_I

- find a shortest path on G_I from any source road to any target road
  - the number of vertices in this path is 1 plus the complexity of the simplest route
- enumerate all shortest paths on G_I
  - convert each path into a route and compute its length
- among the constructed routes, return the fastest

fastest simplest route: our approach (FS)

operates directly on the road network graph G_R

> but, principle of sub-route optimality does not hold: a fastest simplest route may not contain a fastest simplest sub-route



- sub-routes from n_s to n_{11} $\rho'_2 = (n_s, n_6, n_8, n_{11})$ and $\rho'_5 = (n_s, n_7, n_{11})$ have $L = 20$ and $C = 1$
- and extend to routes ρ_2, ρ_5, with $L(\rho_2) = L(\rho_5) = 40$ and $C(\rho_2) = 1$, $C(\rho_5) = 2$

fastest simplest route: our approach (FS)

idea: define a **conceptual** expanded graph $G_E(V', E')$ and a distance metric so that the optimality principle holds

- V' contains an expanded node (n_x, r_i)
- E' contains an edge from (n_x, r_i) to (n_y, r_j) if n_x is on both roads and n_y follows n_x on r_j (r_i and r_j could be the same road)
- an expanded route $\rho_{\mathcal{E}}$ is a path on G_E
  - length $L(\rho_{\mathcal{E}})$ and complexity $C(\rho_{\mathcal{E}})$ are defined appropriately
  - **important**: costs are additive, i.e., $L(\rho^1_{\mathcal{E}}\rho^2_{\mathcal{E}}) = L(\rho^1_{\mathcal{E}}) + L(\rho^2_{\mathcal{E}})$, $C(\rho^1_{\mathcal{E}}\rho^2_{\mathcal{E}}) = C(\rho^1_{\mathcal{E}}) + C(\rho^2_{\mathcal{E}})$ (it does not hold for paths on G_R)
- distance metric: $\rho^1_{\mathcal{E}}$ is FS-shorter than $\rho^2_{\mathcal{E}}$, if $\rho^1_{\mathcal{E}}$ is
  - simpler, i.e., $C(\rho^1_{\mathcal{E}}) < C(\rho^1_{\mathcal{E}})$, or
  - as simple but faster, i.e., $C(\rho^1_{\mathcal{E}}) = C(\rho^1_{\mathcal{E}})$ and $L(\rho^1_{\mathcal{E}}) < L(\rho^1_{\mathcal{E}})$

principle of expanded sub-route optimality holds for FS-shorter

fastest simplest route: our approach (FS)

one-to-many relationship between routes and expanded routes

- an expanded route $\rho_{\mathcal{E}}$ corresponds to a unique route ρ
  - an expanded node (n_x, r_y) of $\rho_{\mathcal{E}}$ corresponds to node n_x in ρ
- theorem: an FS-shortest expanded route corresponds to a fastest simplest route
  - there exist multiple expanded source and target nodes

FS algorithm

- find the FS-shortest expanded route, using any shortest-path algorithm
- convert FS-shortest expanded route to a route

analysis: the number of labels increases by a factor of δ, the number of edge relaxations by a factor of δ^2
(δ is the maximum number of roads a node belongs to)

simplest near-fastest route

no optimality principle, because the answer route is **not optimal**

only option: enumerate routes, use pruning criteria to eliminate sub-routes

consider a route ρ from n_s to n_x

- prune by **length**: $L(\rho) + L_{sf}(n_x \rightsquigarrow n_t) > (1+\varepsilon) \cdot L_{sf}(n_s \rightsquigarrow n_t)$
- prune by **complexity**: $C(\rho) + C_{fs}(n_x \rightsquigarrow n_t) > C^{+}_{snf}(n_s \rightsquigarrow n_t)$

where L_{sf} is the minimum length of the shortest fastest route, C_{fs} is the minimum complexity of the fastest shortest route,

- computed by running a single source shortest path from n_t

and C^{+}_{snf} is an **upper bound** on the complexity of the simplest near-fastest route

- computed as $C^{+}_{snf}(n_s \rightsquigarrow n_t) = C(\rho') + 1 + C_{fs}(n_x \rightsquigarrow n_t)$, for any ρ' such that $L(\rho') + L_{fs}(n_x \rightsquigarrow n_t) \leq (1+\varepsilon) \cdot L_{sf}(n_s \rightsquigarrow n_t)$
  - (L_{fs} is the minimum length of the fastest shortest route)

simplest near-fastest route: algos

SNF-DFS

- DFS-based route enumeration using a stack
- low memory footprint, but does not guide the search

SNF-A*

- A*-based route enumeration, label-setting using a heap
- a heap entry is a **label** representing a route ρ up to n_x
  - A* estimation on the costs of the FS-extension of ρ to n_t: $L(\rho) + L_{fs}(n_x \rightsquigarrow n_t), C(\rho) + C_{fs}(n_x \rightsquigarrow n_t)$
- labels sorted on FS-shorter order
- stop when a label for n_t is deheaped

note: multiple labels per node (no optimality principle)

- but, remove ρ' if $L(\rho') > L(\rho)$ and $C(\rho') > C(\rho) + 1$

experiments

- methods
  - fastest simplest route: BSL, FS
  - simplest near-fastest route: SNF-DFS, SNF-A*, SNF-A*-WB (without precomputed FS, SF routes or bounds)
- datasets

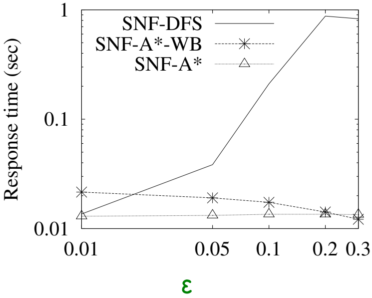
network	# roads	# nodes	avg. degree
OLB	1,672	2,383	2.09
BER	15,246	25,321	2.15
VIE	20,224	27,563	2.17
ATH	76,896	108,156	2.19

experiments - fastest simplest route

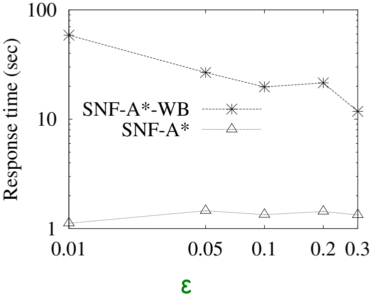
	BSL		FS	
road network	**response time (sec)**	**routes examined**	**response time (sec)**	**routes examined**
OLB	68.7	121,236,000	0.003	2,286
BER	—	—	0.055	27,226
VIE	—	—	0.057	29,301
ATH	—	—	0.346	117,973

BSL is impractical
FS is several orders of magnitude faster for the smallest network

experiments - simplest near-fastest route



OLB



ATH

as ε increases

- the number of candidate routes increases (SNF-DFS suffers)
- but it becomes easier to identify a good candidate (SNF-A* gains)

thank you!