Effective Caching of Shortest Paths for Location-Based Services

Jeppe Rishede Thomsen[1], Man Lung Yiu[1], Christian S. Jensen[2]

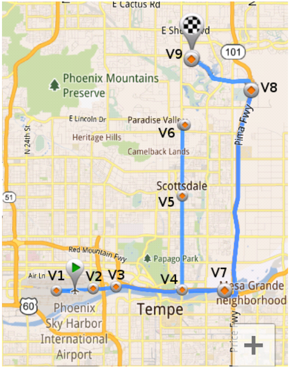
[1]Department of Computing
Hong Kong Polytechnic University

[2]Department of Computer Science
Aarhus University

May 19, 2012

Motivation

- Many Shortest Path Queries
  - 40 % of Google Maps usage is mobile
  - 200+ million active mobile users as of may 2011
  - Users drive 12 billion miles a year with Google Maps Navigation
  - 20 % of Google searches are for local information



http://techcrunch.com/2011/03/11/marissa-mayer-40-of-google-maps-usage-is-mobile-and-there-are-150-million-mobile-users/
http://techcrunch.com/2011/05/25/google-maps-for-mobile-stats/
http://blog.hubspot.com/blog/tabid/6307/bid/10829/5-Google-Local-Stats-Every-Marketer-Should-Know-Data.aspx

Why is caching necessary

- Shortest Path calculation is expensive

Source	Target	Travel time (s)	Response time (ms)
Capitol Building	The Smithsonian	372	101.24
The Smithsonian	Washington, DC	419	110.94
White house	War Memorials	41831	168.44
White house	Capitol Building	75805	278.44
White house	Statue of Liberty	88947	362.8
Capitol Building	Mount Rushmore	99456	364.68
White house	Golden Gate Bridge	108353	342.8

Google Directions API

Setting

Web search scenario:

[Maarkatos et al., Computer Communications 2001]

- Can have a cache at either a proxy or server site
- Saves response- or computation- time
- Cache stores web search results
- Existing cache algorithms include:

  —*Least Recently Used*

  —*Highest Query Frequency*

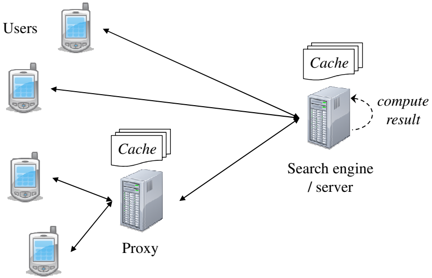


Figure: Web Search

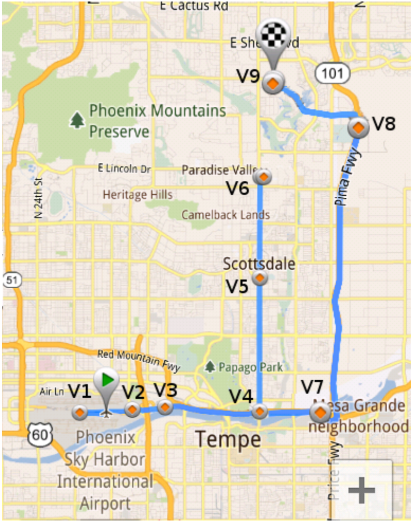
How does a Shortest Path Cache Work?

Cache Content

Path	Shortest Path
1	V1,V2,V3,V4,V7,V8,V9
2	V1,V2,V3,V4,V5,V6
3	V5,V4,V7

Queries

Query	Result	Path
$Q_{V1,V9}$	HIT	1
$Q_{V2,V5}$	HIT	2
$Q_{V5,V9}$	MISS	N/A



How do we express the Cache Performance?

Benefit is expected cost saved:

- On server: computation time
- On proxy: communication time

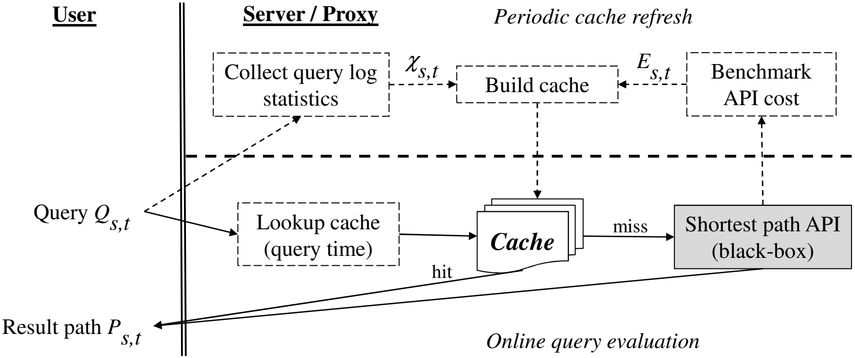
We need to answer:

- Which queries $Q_{s,t}$ can be answered by the path $P_{a,b}$?
- For query $Q_{s,t}$, what are the cost savings?

Given a:

- Cache Size Budget
- Query Log

Then build a cache Ψ with max benefit $\gamma(\Psi)$

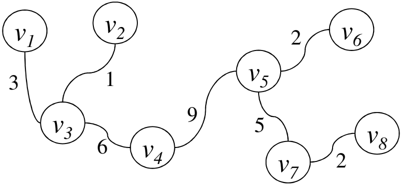


- Systematic model for benefit
- Techniques for query log statistics extraction
- Efficient caching structure
- Shortest Path benchmarking techniques

Statistics Extraction

Timestamp	Query
T_1	$Q_{3,6}$
T_2	$Q_{1,6}$
T_3	$Q_{2,7}$
T_4	$Q_{1,4}$
T_5	$Q_{4,8}$
T_6	$Q_{2,5}$
T_7	$Q_{3,6}$
T_8	$Q_{3,6}$

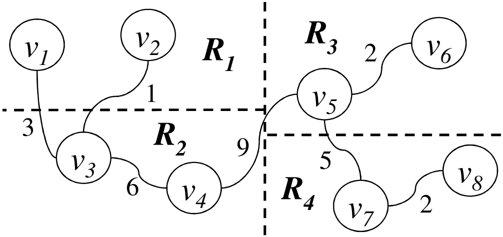
$\chi_{s,t}$	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
v_1	/	0	0	1	0	1	0	0
v_2	0	/	0	0	1	0	1	0
v_3	0	0	/	0	0	3	0	0
v_4	1	0	0	/	0	0	0	1
v_5	0	1	0	0	/	0	0	0
v_6	1	0	3	0	0	/	0	0
v_7	0	1	0	0	0	0	/	0
v_8	0	0	0	1	0	0	0	/



Grouping

$\chi_{s,t}$	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
v_1	/	0	0	1	0	1	0	0
v_2	0	/	0	0	1	0	1	0
v_3	0	0	/	0	0	3	0	0
v_4	1	0	0	/	0	0	0	1
v_5	0	1	0	0	/	0	0	0
v_6	1	0	3	0	0	/	0	0
v_7	0	1	0	0	0	0	/	0
v_8	0	0	0	1	0	0	0	/

χ_{R_i,R_j}	R_1	R_2	R_3	R_4
R_1	0	1	2	1
R_2	1	0	3	1
R_3	2	3	0	0
R_4	1	1	0	0



SP Call Cost Estimation

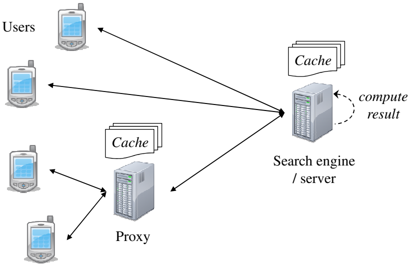
Proxy Scenario

$E_{s,t}(Proxy) = 1$

Server Scenario

Intuition: Longer query results incur higher cost.

– Cost only estimated in server scenario
– Estimation methods developed for Server scenario



Incremental Benefit per size

Benefit formula

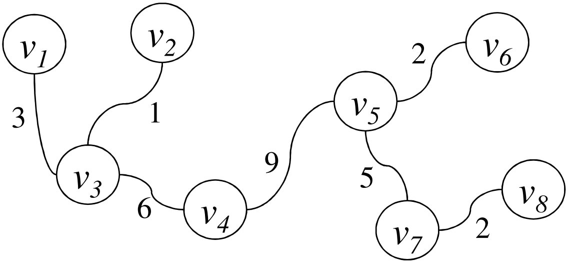
$$\Delta\overline{\gamma}(P_{a,b},\Psi) = \sum_{P_{s,t} \in \mathfrak{U}(P_{a,b}) - \mathfrak{U}(\Psi)} \frac{\chi_{s,t} * E_{s,t}}{|P_{a,b}|}$$

- $P_{a,b}$: a shortest path
- Ψ: the cache
- $\mathfrak{U}(P_{a,b})$: all sub-paths of $P_{a,b}$.
- $\chi_{s,t}$ frequency of query s to t.
- $E_{s,t}$: cost of calculating $P_{s,t}$

Cache: SP Result Ranking

Greedy algorithm

Timestamp	Query
T_1	$Q_{3,6}$
T_2	$Q_{1,6}$
T_3	$Q_{2,7}$
T_4	$Q_{1,4}$
T_5	$Q_{4,8}$
T_6	$Q_{2,5}$
T_7	$Q_{3,6}$
T_8	$Q_{3,6}$

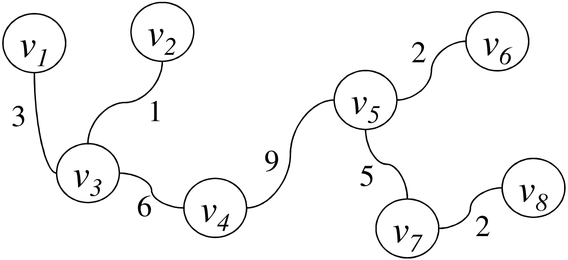


Round	Path						Cache Ψ	
	$P_{1,4}$	$P_{1,6}$	$P_{2,5}$	$P_{2,7}$	$P_{3,6}$	$P_{4,8}$	Before	After
1	1/3	$\boxed{\mathbf{5/5}}$	1/4	2/5	3/4	1/4	empty	$P_{1,6}$
2							$P_{1,6}$	$P_{1,6}, ?$

Cache: SP Result Ranking - Incremental benefit

Incremental benefit calculation

Timestamp	Query
T_1	$Q_{3,6}$
T_2	$Q_{1,6}$
T_3	$Q_{2,7}$
T_4	$Q_{1,4}$
T_5	$Q_{4,8}$
T_6	$Q_{2,5}$
T_7	$Q_{3,6}$
T_8	$Q_{3,6}$



Round	Path						Cache Ψ	
	$P_{1,4}$	$P_{1,6}$	$P_{2,5}$	$P_{2,7}$	$P_{3,6}$	$P_{4,8}$	Before	After
1	1/3	$\boxed{\mathbf{5/5}}$	1/4	2/5	3/4	1/4	empty	$P_{1,6}$
2	0	0	1/4	$\boxed{\mathbf{2/5}}$	0	1/4	$P_{1,6}$	$P_{1,6}, P_{2,7}$

Efficient Data Structure for the Cache - Faster lookup

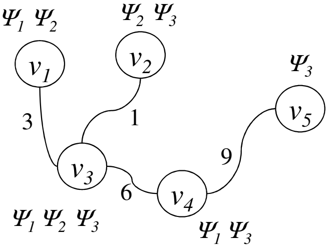
- Lookup time grows with size cache
- Support return of full or partial cache items

Ψ_1	v_1, v_3, v_4
Ψ_2	v_1, v_3, v_2
Ψ_3	v_2, v_3, v_4, v_5

Paths

v_1	Ψ_1, Ψ_2
v_2	Ψ_2, Ψ_3
v_3	Ψ_1, Ψ_2, Ψ_3
v_4	Ψ_1, Ψ_3
v_5	Ψ_3

Inverted List



Visualization

Efficient Data Structure for the Cache - Efficient storage

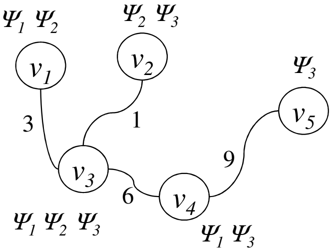
- Support efficient lookup and return of full or partial cache items
- Compact storage of shortest paths

v_1	v_3
v_2	v_3
v_3	v_1, v_2, v_4
v_4	v_3, v_5
v_5	v_4

Graph representation

	content	parent
v_1	Ψ_1, Ψ_2	NIL
v_2	$\cdots$	$\cdots$
v_3	Ψ_3	v_1
v_4	$\cdots$	$\cdots$
v_5	$\cdots$	$\cdots$

Prefix compressed



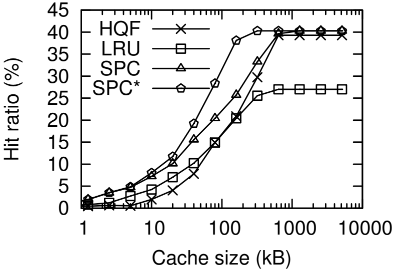
Visualization

Experimental Setup

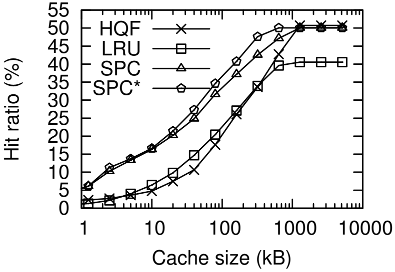
- Query logs divided equally into 2 sets:
  - historical
  - query workload
- Comparison with:
  - Least Recently Used (LRU)
  - Higest Query Frequency (HQF)

Dataset	Trajectories	Road network
Aalborg	Infati GPS data 4,401 trajectories	From *downloads.cloudmade.com* 129k nodes, 137k edges
Beijing	Geo-Life GPS data 12,928 trajectories	From *downloads.cloudmade.com* 76k nodes, 85k edges

Hit ratio — Proxy Scenario



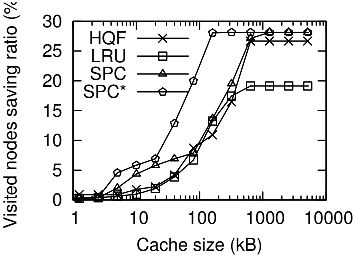
Aalborg



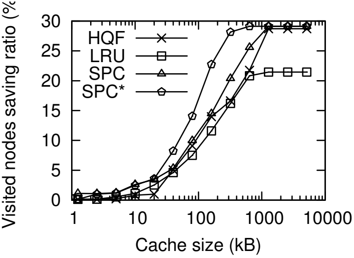
Beijing

Performance Savings — Server Scenario

Dijkstra

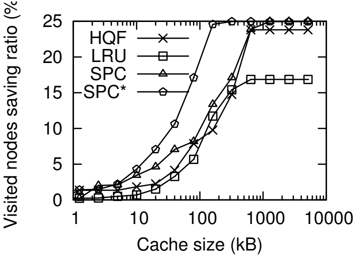


Visited nodes savings, Aalborg

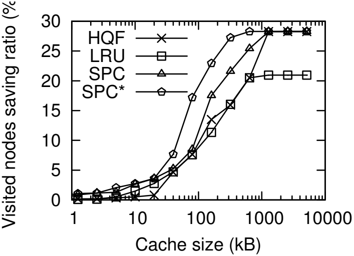


Visited nodes savings, Beijing

A*



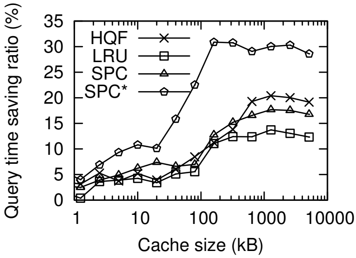
Visited nodes savings, Aalborg



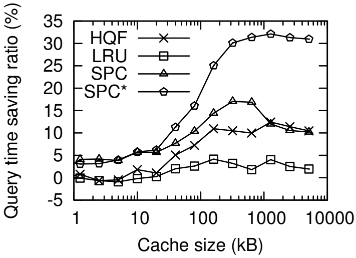
Visited nodes savings, Beijing

Performance Savings — Server Scenario

Dijkstra

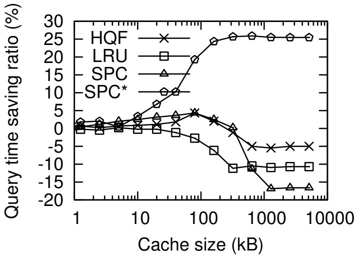


Query time savings, Aalborg

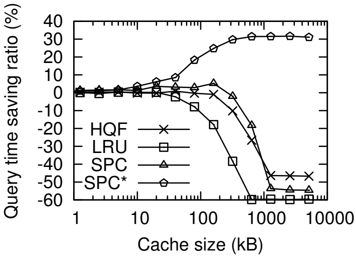


Query time savings, Beijing

A*



Query time savings, Aalborg



Query time savings, Beijing

Conclusion

- Introduced benefit model, capturing the benefit of adding a shortest path result, relative to other results.
- Designed statistics extraction techniques
- Developed techniques to estimate cost of calculating a shortest path
- Designed efficient data structure for the cache storage

- Experimental results show:
  - High hit ratio
  - Small lookup overhead
  - Low query time

Thank You For Listening