Fast Shortest-Path Queries

Laurent Viennot

MPRI – Theory of practical graph algorithms

Fast shortest-path queries

Shortest path queries

Problem :

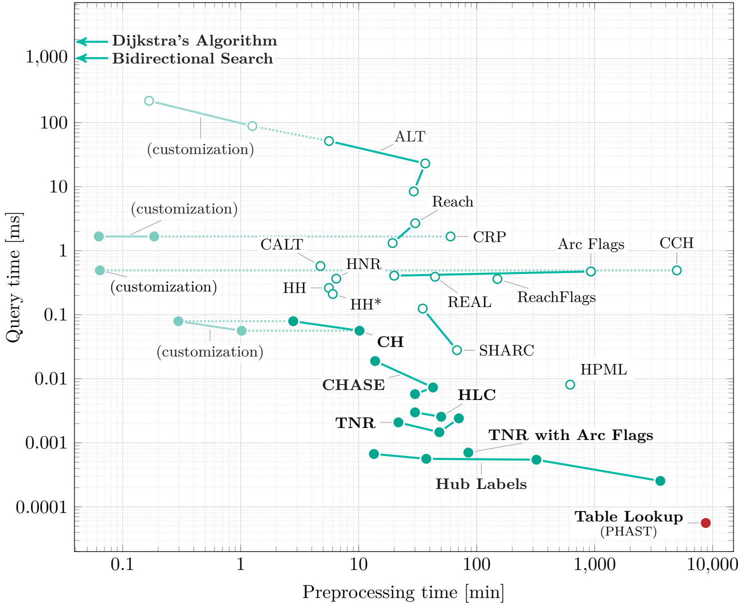
- A graph G is given.
- Answer queries : shortest path from s to t ?

Trivial solution : pre-compute for all s, t.

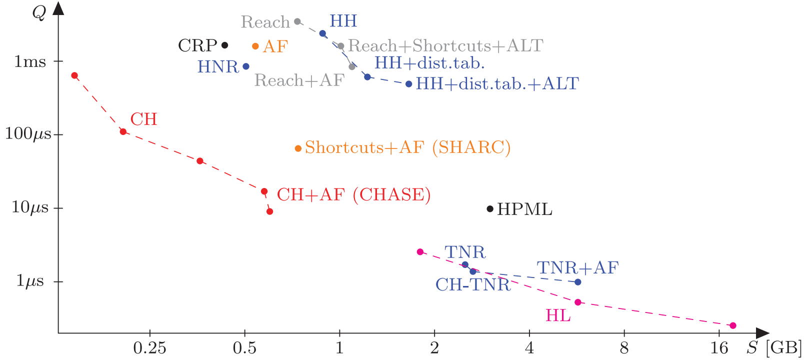
Recent progress [BDG+15], e.g. in road networks ($n = 20$M) :

- Dijkstra : 4s
- Bidirectional Dijkstra : 1s
- Bidirectional A^* : 100ms
- Reach-Pruning, Contraction Hierarchies : 10 ms
- Hub labeling : 10 μs

Query time vs preprocessing time (exact) [BDG+15]



Query time vs space (exact) [Sommer 14]



Distance oracles

Def : Given a graph G with n nodes, compute a data-structure of size S allowing to answer queries "what is $d_G(u,v)$?" for $u, v \in V(G)$ in time t.

Th : [Cohen Porat 2010] $t = O(1)$ requires space $S = \Omega(n^2)$ under Fast-Set-Intersection Hypothesis (given $S_1, \dots, S_n \subseteq [\log^c n]$, answering queries "does S_i intersects S_j ?" in constant time requires $\Omega(n^2)$ space). This holds even for $2-\varepsilon$-approximation.

Best algorithm : $O(t)$ time with $O(n^2/t^2)$ space. [Cohen Porat 2010].

A short history of shortest paths

Dijkstra [Dijkstra '59]

Procedure Dijkstra (G, s, t)

  Distance label $d(u) := 0$ if $u = s$, ∞.

  Radius $r := 0$.

  Repeat

    Pick unvisited u with $d(u)$ min. // $d(u) = d_G(s, u)$

    Visit u :

    For $v \in N_G(u)$ **do** $d(v) := \min\{d(v), d(u) + \ell(uv)\}$

    $r := \min_{\text{unvisited } v} d(v)$

  until $d(t) \leq r$

  Return $d(t)$

Bidirectional Dijkstra

Procedure BidirDijkstra (G, s, t)

Alternate Dijkstra (G, s, t) and Dijkstra $(\overleftarrow{G}, t, s)$.
Stopping condition?

Estimation $\mu := \infty$ of $d(s,t)$.
When scanning edge uv : $\mu := \min \left\{ \mu, d(u) + \ell(uv) + \overleftarrow{d}(v) \right\}$.
Stop if $\mu \leq r + \overleftarrow{r}$.

Exercise : show correctness.
Exercise : explain 1s vs 4s in road networks.

A^* [Hart, Nilsson, Raphael '68]

Shortest path algorithm with prediction :

Potential function $\pi(u) \approx d(u,t)$.
Dijkstra (G_π, s, t) with $\ell_\pi(uv) = \ell(uv) - (\pi(u) - \pi(v))$.
  $d_\pi(s,t) = d(s,t) - (\pi(s) - \pi(t))$.
  Visit u with $d(u) + \pi(u)$ min.

π feasible if $\forall uv \in E(G)$, $\ell_\pi(uv) \geq 0$

Exercise : Sufficient condition for using $\pi(u) = D(u,t)$ for a metric D ?

Bidirectional A^* : ALT [Goldberg, Harelsson '05]
(ALT = A^*, Landmarks, Triangle inequality)

- $\ell_\pi(uv) = \overleftarrow{\ell}_{\overleftarrow{\pi}}(vu) \iff \pi + \overleftarrow{\pi} = cte$
  (ex : $\pi' = (\pi - \overleftarrow{\pi})/2$ and $\pi'' = (\overleftarrow{\pi} - \pi)/2$
  or π and $\pi''(v) = \max\{\overleftarrow{\pi}(v), \pi(t) - \pi(v) + \alpha\overleftarrow{\pi}(s)\}$)
- or different stopping condition ($\overleftarrow{r} \geq \mu$)
- π from landmarks (better than using coordinates) :

$\pi(u) = \max_{x \in X} d_G(u,x) - d_G(t,x)$

Reach pruning [Gutman '04] revisited [Goldberg, Kaplan, Werneck '05]

$\text{Reach}(u) = \max_{(s,t) | u \in P_{st}} \min \{d(s,u), d(u,t)\}$

In bidir. Dijkstra, when scanning u :

- Prune v s.t. $\text{Reach}(v) < \min \{d(u) + \ell(uv), \overleftarrow{r}\}$.

Add shortcuts :

- Tie break : fewer links is shorter.

Exercise : how to get shortest path from s to t ?

Pre-compute reach upper bounds :

- Eliminate nodes with reach $\leq \delta$.
- Shortcut paths with degree 2 nodes.
- Repeat with larger δ.

Contraction Hierarchies [Geisberger, Sanders, Shultes, Delling '05-08]

Node ordering $\pi : u_1 < \cdots < u_n$

Contract successively u_i :

- add shortcut vw for $v, w \in N(u_i)$ (if needed),
- remove u_i (distances are preserved in remaining graph).

Query : bidir. Dij. in $G^+\uparrow$ and $\overleftarrow{G}^+\uparrow$.

- G^+ : graph + shortcuts
- $\uparrow$: follow uv if $u <_\pi v$

Finding π :

- small degree + levels (MIS),
- min fill-in (greedy treewidth dec.),
- small balanced separators ($O(n \log n)$ shortcuts if planar).

Exercise : bound the number of shortcuts if any subgraph of G has an $O(n^\varepsilon)$ balanced separator and maximum degree Δ.

Open pb : link between small Reach and small CH?

CH complexity for planar graphs 1/3.

Theorem [Lipton, Tarjan '79] Every planar graph G has a 2/3-balanced separator S_0 of size $O(\sqrt{n})$.

Elimination ordering π : recursively order each connected component and then add nodes in S_0 in any order. This results in a tree of separators of depth $O(\log n)$. (Nested dissection as in [Gilbert, Tarjan '87].)

Rq : all shortcuts occur between a tree-node and an ancestor (possibly the tree-node itself).

Corollary : The nodes visited during a pruned Dijkstra from a node s at depth k are all in the separators in the branch of s and query time is $O(\sum_{i=0}^{k} s_i) = O(\sqrt{n})$ (where $s_i = c\sqrt{(2/3)^i n}$ is the maximum size of a separator at depth i).

Lemma : There are $O(n)$ shorcuts with an extremity in S_0.
Rq1 : At most $|S_0|^2 = O(n)$ inside S_0.
Rq2 : Each shortcut with a node u at depth k is the result of the contraction of nodes in the subtree rooted at the separator S containing u.
Def : define the bipartite graph G_k with vertex set $S_0 \cup D_k$ where D_k is the set of separators at depth k and an edge (v, S) if there is a node $v \in S_0$ and a separator $S \in D_k$ such that there exists a shortcut from $u \in S$ to v.
Rq3 : G_k is planar because it can be obtained from G by removing edges inside S_0, removing nodes at depth within 1 and $k-1$ and contracting edges between nodes at depth k or deeper. We thus have $m(G_k) \le 3n(G_k) - 6$ (Euler & $3f \le 2m$).
Rq4 : The number of shortcuts with an extremity at depth k is $N_k \le \sum_{S \in D_k} |S| \deg_{G_k}(S)$. If $D'_k = \{S : \deg_{G_k}(S) > 3\}$, then $N_k \le \sum_S 3|S| + \sum_{S \in D'_k} s_k(\deg_{G_k}(S) - 3) \le 3n_k + s_k(m(G'_k) - 3|D'_k|)$ where $G'_k = G_k[D'_k \cup S_0]$ satisfies $m(G'_k) \le 3n(G'_k)$ implying $N_k \le 3n_k + 3s_k|S_0|$. The Lemma follows from $\sum_k s_k = O(\sqrt{n})$.

CH complexity for planar graphs 3/3.

Corollary : The number of shortcuts generated by contracting G according to π is $O(n \log n)$.

Rq1 : Similarly to the previous Lemma, for each separator S root of a subtree of n' nodes, there are $O(n')$ shortcuts with an extremity in S.

Rq2 : When summing all subtree sizes, a node is counted $O(\log n)$ times.

Transit node routing [Bast, Funke, Matjevic '07 ; Sanders, Schultex '09]

Transit nodes T so that any long distance path goes through a transit node $x \in T$. Pre-compute all distances $d(x, y)$ for $x, y \in T$.

Access nodes $A(v) \subseteq T$: any long path from/to v goes through $x \in A(v)$.

Long distance query (s, t) :
$\min_{x \in A(s), y \in A(t)} d(s, x) + d(x, y) + d(y, t)$

Local query (s, t) : Use bidirectional Dijkstra or even CH.

Highway dimension [Abraham, Delling, Fiat, Goldberg, Werneck '10-13]

Graph property ensuring efficient ordering for CH and fast pruned bidir. Dijkstra.

Definition

Highway dimension $h = \max_{u,r} \min_{H \text{ hitting set of } \mathcal{P}_{ur}} |H|$
where $\mathcal{P}_{ur} = \{P \in \mathcal{P}_r \mid \overline{P} \cap B(u, 2r) \neq \emptyset\}$, $\mathcal{P}_r = \{P \mid \ell(\overline{P}) > r\}$, and $\overline{P}$ is any shortest path extending P by 0 or 1 edge at each extremity.

Theorem

Any graph G with highway dimension h and diameter D admits a node ordering π s.t. CH_π produces at most $O(nh \log D)$ shortcuts and $CH_\pi + RP$ bidir. Dij. visits $O(h \log D)$ nodes. (RP : prune v s.t. $d(v) > 2^{\text{level}(v)}$)

Lemma

For all r, G has an (h, r)-sparse hitting set, i.e. a set C s.t. $C \cap P \neq \emptyset$ for all $P \in \mathcal{P}_r$ and $|B(u, 2r) \cap C| \leq h$ for all $u \in V(G)$.

Proof Lemma : Define C as a minimum hitting set for $\mathcal{P}_r$.

Idea Theorem : Construct a $(h, 2^{i-1})$-sparse hitting set C_i for $i = 0, \ldots, \lceil \log D \rceil$.
Define $C'_i = C_i \setminus \cup_{j>i} C_j$ and $\pi = C'_0, \ldots, C'_{\lceil \log D \rceil}$.

Rk1 : For each shortcut vw with $v \in C'_i$ and $w \in C'_j$ with $i \leq j$, we have $d(v, w) \leq 2^i$.

Cor1 : Each node v has at most $h \lceil \log D \rceil$ shortcuts.

Rk2 : In CH bidir. Dijkstra query, prune $v \in C'_i$ s.t. $d(v) > 2^i$.

Cor2 : At most h nodes of C'_i are visited.

A striking remark on pruned Dijkstra

Rk : The $O(h \log D)$ bound on pruned Dijkstra from s holds even without the bidir. stopping condition.

Idea : Pre-compute this pruned Dijkstra D_v for all v!

Result : For any s, t the distance (and the shortest path) from s to t can be computed from D_s and D_t.

The collection $(D_v)_{v \in V}$ form a distance labeling of G (subject of next lecture).

Further reading

[Schild, Sommer 2015]
On balanced separators in road networks.
SEA 2015.
https://aschild.github.io/papers/roadseparator.pdf

[Delling, Goldberg, Pajor, Werneck 2017]
Customizable route planning in road networks.
Transportation Science 2017.
https://www.microsoft.com/en-us/research/wp-content/uploads/2013/01/crp_web_130724.pdf

Open pb : Analyze CH in H-minor-free graphs.

Exercise

I : Bidirectional A^*.

II : Transit node routing in graphs of highway dim. $h = O(1)$:

- transit nodes : $X = C_q$ with smallest q s.t. $|C_q| \leq \sqrt{m}$,
- $A(v) = \{x \in T_v \cap X \text{ S.t. } P_{vx} \cap X = \{x\}\}$,
- TN algorithm (for a well chosen distance R) :

$g(s,t) = \min_{x \in A(s), y \in A(t)} \ell(P_{sx}) + \ell(P_{xy}) + \ell(P_{yt})$;
if $g(s,t) \geq R$ return $g(s,t)$ else return bidirDijCH(s,t).

II.1 Show $d(v,x) \leq 2^{q-1}$ for any $v \in V$, $x \in A(v)$.

II.2 Show $g(s,t) = d(s,t)$ when $d(s,t) > 2^{q-1}$.

II.3 Upper bound $g(s,t)$ when $d(s,t) \leq 2^{q-1}$ and deduce an appropriate value for R.

II.4 Bound $|A(v)|$ for any $v \in V$.

II.5 What is the time complexity of a TN query when $g(s,t) \geq R$ when allowing linear space ?

Exercise for next week

Contraction order matters : Propose a bounded-degree tree graph of n nodes and a contraction order that produce contraction hierarchies of size $\Theta(n^2)$. (Nodes are contracted one after an other in order. When contracting a node v of the current graph G, we add shortcuts uw to G for neighbors u, w of u such that $d_G(u, w) = d_G(u, v) + d_G(v, w)$ and remove v. The size of the contraction hierarchies is counted as the number of shortcuts added.)

Send a short argumented answer to `laurent.viennot@inria.fr`.