On Computing the Diameter of (Weighted) Link Streams

Andrea Marino
Joint work with: Marco Calamai and Pierluigi Crescenzi

Algorithmic Aspects of Temporal Graphs V,
Satellite workshop of ICALP 2022

Paris, France. Monday 4 July 2022

Distance Analysis in Facebook

In 2011, the average distance and the diameter of Facebook (721.1M nodes and 68.7G edges) have been computed (they were resp. 5.7 and 41).


The New York Times

Business Day
Technology

WORLD | U.S. | N.Y. / REGION | BUSINESS | TECHNOLOGY | SCIENCE | HEALTH

Separating You and Me? 4.74 Degrees

By JOHN MARKOFF and SOMINI SENGUPTA
Published: November 21, 2011

Paolo Boldi, Sebastiano Vigna: Four Degrees of Separation, Really. ASONAM 2012: 1222-1227

Average Distance Analysis

- The average distance of Facebook has been computed by applying HyperANF tool.
  - Paolo Boldi, Marco Rosa, Sebastiano Vigna: HyperANF: approximating the neighbourhood function of very large graphs on a budget. WWW 2011: 625-634
- It estimates the number of pairs of nodes at distance h in time $O(hm \log n)$.
- **It is possible to extend this approach to temporal graphs,** in order to estimate the pairs of vertices reachable within time t in time $O(m \log n)$, where m is the number of temporal edges.
  - Pierluigi Crescenzi, Clémence Magnien, Andrea Marino: Approximating the Temporal Neighbourhood Function of Large Temporal Graphs. Algorithms 12(10): 211 (2019)

Diameter Analysis

- The diameter of Facebook has been computed by applying *i*FUB.
- Even though computing the diameter requires $\Omega(n^2)$ unless SETH (Rodditty and V. Williams, 2013), in practice we can do it in less time.

  Crescenzi, Grossi, Habib, Lanzi, and Marino. On computing the diameter of real-world undirected graphs. Theoretical Computer Science, 2012.

- **It is possible to extend this approach to temporal graphs? Today's Talk.**

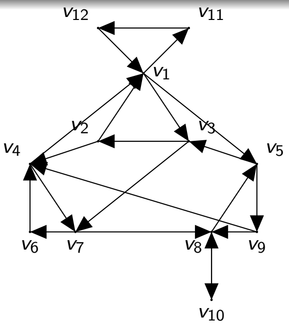
  Marco Calamai, Pierluigi Crescenzi, Andrea Marino: On Computing the Diameter of (Weighted) Link Streams. SEA 2021: 11:1-11:21

Part I

On computing the Diameter of Static (Directed) Graphs

Given a Strongly Connected Directed Graph

- The distance $d(u, v)$ is the number of edges along shortest path from u to v.
- Forward Eccentricity of u: $eccf(u) = \max_{v \in V} d(u, v)$
- Backward Eccentricity of u: $eccb(u) = \max_{v \in V} d(v, u)$
- Diameter: maximum *eccf* or *eccb*
- Computable with n BFSes.

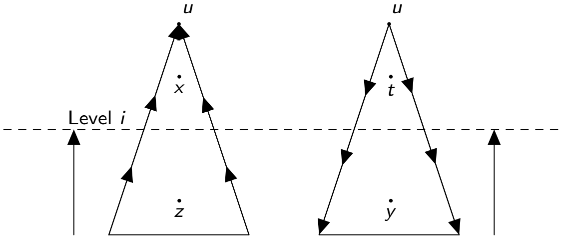


	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	ecc_F
v_1	0	2	1	3	1	3	2	3	2	4	1	2	4
v_2	1	0	2	1	2	3	2	3	3	4	2	3	4
v_3	2	1	0	2	3	2	1	2	4	3	3	4	4
v_4	1	3	2	0	2	2	1	2	3	3	2	3	3
v_5	3	2	1	2	0	3	2	2	1	3	4	5	5
v_6	2	4	3	1	3	0	2	3	4	4	3	4	4
v_7	3	4	3	2	2	1	0	1	3	2	4	5	5
v_8	4	3	2	3	1	4	3	0	2	1	5	6	6
v_9	2	4	3	1	2	3	2	1	0	2	3	4	4
v_{10}	5	4	3	4	2	5	4	1	3	0	6	7	7
v_{11}	2	4	3	5	3	5	4	5	4	6	0	1	6
v_{12}	1	3	2	4	2	4	3	4	3	5	2	0	5
ecc_B	5	4	3	5	3	5	4	5	4	6	6	7	

Main Observation of iFUB (iterative fringe upper bound)

Theorem (Sketch)

1. *All the nodes x above the level i in* $\text{BBFS}(u)$ *having eccf greater than* $2(i-1)$ *have a corresponding node y, whose eccb is greater or equal to* $eccf(x)$*, below or on the level i in* $\text{FBFS}(u)$.
2. *Symmetrically for t and z.*



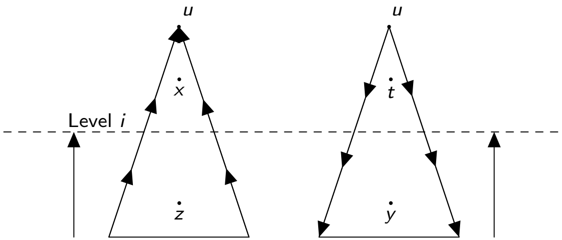
Corollary

- *Let lb be max among all the eccb of nodes y and all the eccf of nodes z.*
- *The eccb of nodes t and the eccf of nodes x are bounded by* $\max\{lb, 2(i-1)\}$.

Main schema of the algorithm

Perform a forward and a backward BFS from a node u. For decreasing values of i

- At level i we have computed all the *eccb* of nodes y and the *eccf* of nodes z. The maximum is our lower bound *lb*.
- If *lb* is bigger than $2(i-1)$, *lb* is the diameter.
  - No node to be examined can have *eccf* or *eccb* bigger than *lb*.



For Facebook, instead of doing $O(n)$ BFSes (the worst case of the algorithm is indeed $O(n)$ BFSes), it did 17 BFSes.

Part II

On Computing the Diameter of (Weighted) Link Streams

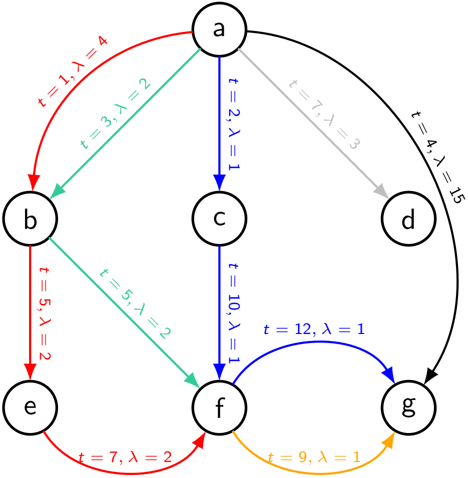
- (Temporal) graph $\mathbb{G} = (V, \mathbb{E})$ where V is set of nodes ($n = |V|$) $\mathbb{E}$ is set of (temporal) edges (u, v, t, λ) ($m = |\mathbb{E}|$).
- The graph is seen as a **stream of links**. The graph is not stored in memory but we have to scan a file of temporal edges.
- Path $\mathbb{P}$ from u to v is a sequence of edges

  $$(u = w_1, w_2, t_1, \lambda_1), (w_2, w_3, t_2, \lambda_2), \ldots, (w_k, w_{k+1} = v, t_k, \lambda_k)$$

  such that, for each i with $1 < i \leq k$, $t_i \geq t_{i-1} + \lambda_{i-1}$
  - Departure time: t_1
  - Arrival time: $t_k + \lambda_k$
  - Duration: $t_k + \lambda_k - t_1$
  - Travel time: $\lambda_1 + \cdots + \lambda_k$
  - $[t_{min}, t_{max}]$-compatible if departure time no earlier than t_{min} and arrival time no later than t_{max}

Given interval $\mathbb{I} = [t_{min}, t_{max}]$ (e.g. [0,19]):

- $d_{\text{eat}}(u, v)$: min arrival time of $\mathbb{I}$-compatible path minus t_{min} ($d_{\text{eat}}(a, g) = 10$)
- $d_{\text{ldt}}(u, v)$: t_{max} minus max departure time of $\mathbb{I}$-compatible path ($d_{\text{ldt}}(a, g) = 15$)
- $d_{\text{ft}}(u, v)$: min duration time of $\mathbb{I}$-compatible path (e.g. $d_{\text{ft}}(a, g) = 3$)
- $d_{\text{st}}(u, v)$: min travel time of $\mathbb{I}$-compatible path (e.g. $d_{\text{st}}(a, g) = 7$)



$d_*(u, v) = \infty$ if there exists no $\mathbb{I}$-compatible path from u to v
Eccentricity $\text{ecc}_*(u)$ of node u: max finite distance to any other node.
Diameter D_* of $\mathbb{G}$: max (defined) eccentricity of all its nodes

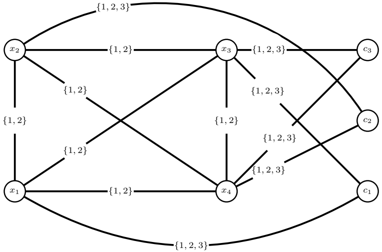
Negative Result

Theorem (Extending the result by Roditty and V. Williams for static)

For any $\mathrm{D} \in \{\mathrm{EAT}, \mathrm{FT}, \mathrm{ST}\}$, *computing the diameter of a link stream* $(V, \mathbb{E})$ *cannot be done in time* $\tilde{O}(|\mathbb{E}|^{2-\epsilon})$ *for any* $\epsilon > 0$, *unless the SETH is false, even if the link stream is unweighted and undirected.*

Quasi-linear Reduction from *k-Two Disjoint Sets* problem: Given a set X and a collection $\mathcal{C}$ of subsets of X such that $|X| < \log^k(|\mathcal{C}|)$, decide if there are two disjoint sets $c, c' \in \mathcal{C}$. For any k, the k-TDS problem is not solvable in time $\tilde{O}(|\mathcal{C}|^{2-\epsilon})$, unless the SETH is false.

For any distance, the diameter is 3, iff there are two disjoint sets (i.e. c_1 and c_2).



But let's try to do something that in practice is fast (as for static graphs)!

The time and space complexity of the single source best path and the single target best path algorithms with input $\mathbb{E}_{\downarrow}$ and $\mathbb{E}_{\uparrow}$, respectively (without considering the time and space necessary for sorting the link stream).

D	Single Source TIME	Single Source SPACE	Single Target TIME	Single Target SPACE
EAT	$O(m)$	$O(n)$	$O(m)$	$O(m)$
LDT	$O(m)$	$O(m)$	$O(m)$	$O(n)$
FT	$O(m)$	$O(m)$	$O(m)$	$O(m)$
ST	$O(m \log m)$	$O(m)$	$O(m \log m)$	$O(m)$

$\overrightarrow{\mathbb{E}}$: sorted in non-decreasing order w.r.t. the edge starting times
$\overleftarrow{\mathbb{E}}$: sorted in non-increasing order w.r.t. the edge starting times
All time complexities become $O(m \log m)$ if we pay sorting

Huanhuan Wu, James Cheng, Silu Huang, Yiping Ke, Yi Lu, Yanyan Xu: Path Problems in Temporal Graphs. Proc. VLDB Endow. 7(9): 721-732 (2014)

Pierluigi Crescenzi, Clémence Magnien, Andrea Marino: Finding Top-k Nodes for Temporal Closeness in Large Temporal Graphs. Algorithms 13(9): 211 (2020)

Connectivity and Pivots

- The iFUB approach in static graphs assumes that the graph is **connected**, i.e. all the pairs u, v are s.t. u reaches v.
- **No efficient algorithm** capable of computing temporal connected components, i.e. $R = \{(u, v) : u \text{ reaches } v\}$.

We restrict ourselves to the computation of **pivot-diameter**

Pivots P: a set of pairs (x, t), for instance, central stations at a given time

$R(P)$: set of pairs of nodes u and v such that u can reach v passing through some $p \in P$

- That is, $d_{\text{eat}}(u, p) < \infty$ in $[t_{min}, t]$ and $d_{\text{eat}}(p, v) < \infty$ in $[t, t_{max}]$

P-diameter: max distance from u to v for any $(u, v) \in R(P)$

- The original diameter consider R instead of $R(P)$.

If we choose P as the top-*logn* nodes with more out-neigh, taken at 4 times eq. spaced:

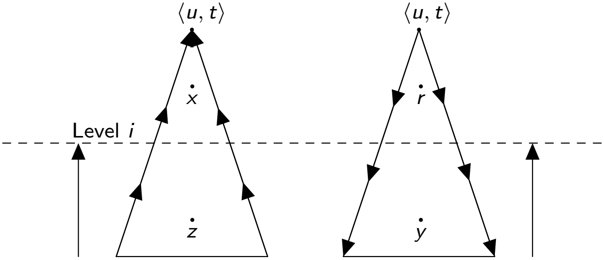
- $R(P)$ (pairs u, v such that u can reach v passing through a $p \in P$) **is almost** R (all reachable pairs)
- P-diameter D_*^P (max distance from u to v for any $(u, v) \in R(p)$) **is close to the diameter**.

This is an heuristic! Solving the perfect pivot set problem is NP-hard and not approximable within a factor $c \log n$, for some constant $c > 0$ (reduction from Hitting Set).

PUBLIC TRANSPORT NETWORKS	$\frac{\lvert R(P)\rvert}{\lvert \mathcal{R}\rvert}$	$\frac{D_{\text{FT}}^P}{D_{\text{FT}}}$	$\frac{D_{\text{ST}}^P}{D_{\text{ST}}}$
KUOPIO	98.18%	1	0.09
RENNES	98.55%	0.91	0.97
GRENOBLE	97.41%	1	1
VENICE	96.21%	1	0.95
BELFAST	98.52%	1	0.89
CANBERRA	99.28%	1	1
TURKU	98.21%	1	0.94
LUXEMBOURG	99.74%	1	1
NANTES	98.23%	1	1
DETROIT	99.08%	1	1
TOULOUSE	98.86%	1	1
PALERMO	100.00%	1	1
BORDEAUX	98.55%	1	0.97
WINNIPEG	97.85%	1	1
BRISBANE	98.68%	1	0.99
DUBLIN	99.20%	1	1
ADELAIDE	98.87%	1	0.93
LISBON	98.13%	1	1
PRAGUE	98.37%	1	1
HELSINKI	99.40%	1	0.95
BERLIN	99.58%	1	1
ROME	99.86%	1	1
MELBOURNE	98.22%	1	0.95
SYDNEY	96.22%	1	0.55
PARIS	99.54%	1	1

Computing the Pivot Diameter

- Consider a distance FT, ST (works also for EAT and LDT)
- **The same observation of iFUB applies.**
  - Given a pivot $\langle u, t \rangle$, if x has large forward eccentricity and $d_\star^{[t_{min}, t]}(x, u)$ is low, then there is y with large backward eccentricity such that $d_\dagger^{[t, t_{max}]}(u, y)$ is large.



  - Start from the farthest nodes from $\langle u, t \rangle$ and compute eccentricities, maintaining the maximum lb and an upper bound ub of the remaining ecc. Stop when $lb \geq ub$.
  - ub, $\dagger$, and $\star$ depend on the chosen kind of distance.

Computing the Pivot Diameter

Denote with F_* the number of visits performed wrt to the number of nodes (i.e. visits of the textbook algorithm) to compute pivot diameter D_*^P.

- The quadratic conditional lower bound we proved also applies to pivot diameter.
- Consistently, the worst case running time of this algorithm is $O(nm)$, but in practice it is often much faster.

PUBLIC TRANSPORT NETWORKS	F_{FT}	F_{ST}
KUOPIO	0.26	1.44
RENNES	0.15	0.69
GRENOBLE	0.55	0.60
VENICE	0.07	0.49
BELFAST	0.07	0.66
CANBERRA	0.32	0.32
TURKU	1.40	0.05
LUXEMBOURG	0.08	0.52
NANTES	0.05	0.04
DETROIT	0.06	0.12
TOULOUSE	0.10	0.58
PALERMO	1.62	0.06
BORDEAUX	0.07	1.32
WINNIPEG	0.07	0.51
BRISBANE	0.01	0.09
DUBLIN	0.03	0.74
ADELAIDE	0.07	0.36
LISBON	0.03	0.09
PRAGUE	0.03	0.36
HELSINKI	0.03	0.02
BERLIN	0.02	0.19
ROME	0.03	0.17
MELBOURNE	0.06	0.03
SYDNEY	0.01	0.07
PARIS	0.01	0.04

For EAT (simil. LDT), simpler lower/upper bound algorithm is quite effective by observing that the EAT diameter often traverses edges with high time appearance.

Thanks