Most Pleasant Itinerary

Christopher Aykroyd
Ramon Sena

The problem

Given a Graph **G** and a query of two nodes **(u,v)**, find the path **P** in **G** that minimizes the maximum of all the weights along itself.

Results

- Report: *Total Runtime* = *File load* + *Preprocessing* + *Queries* + *File Save*
  - *This presentation: Separate columns (only preprocessing and query time considered)*
- *Randomness: Tree Root*
  - *Average, maximum and minimum experimental values for 50 iterations*
- Representation
  - M = #Edges, N = #Vertices, Q = #Queries

Results

FIRST APPROACH

Naive Approach				Kruskal			Query Time		
TEST #	n (10^5)	m (10^5)	q (10^5)	Min (ms)	Avg (ms)	Max (ms)	Min (ms)	Avg (ms)	Max (ms)
2	1	1	1	118	158	341	20	23	31
3	1	2	5	151	196	519	407	659	923
4	1	3	5	187	240	379	576	665	775
5	2	3	1	301	381	609	126	186	270
6	2	3	5	301	370	756	543	987	1506
7	2	3	5	306	384	361	631	958	1126
8	2	3	5	301	374	784	560	892	1145
9	2	3	5	308	364	459	597	870	1067
Extremes				118	308.4	756	20	655.0	1506

Results

EXPONENTIAL STEPS

Exponential Steps			Exp Preprocessing			Total Query Time		
TEST #	n (10^5)	q (10^5)	Min (ms)	Avg (ms)	Max (ms)	Min (ms)	Avg (ms)	Max (ms)
2	1	1	17	27	67	28	39	49
3	1	5	13	31	55	197	300	351
4	1	5	13	31	39	192	311	428
5	2	1	30	63	78	43	66	83
6	2	5	31	61	84	214	315	456
7	2	5	29	66	80	206	328	516
8	2	5	28	60	75	207	315	475
9	2	5	29	59	80	225	328	425
Extremes			13	49.8	84	28	250.3	516

Results

TARJAN

Tarjan			Tarjan's Algorithm		
TEST #	n (10^5)	q (10^5)	Min (ms)	Avg (ms)	Max (ms)
2	1	1	95	115	162
3	1	5	476	537	667
4	1	5	332	514	798
5	2	1	135	166	219
6	2	5	453	602	734
7	2	5	562	630	718
8	2	5	554	625	757
9	2	5	461	610	677
Extremes			95	474.8	798

Results

SUMMARY OF WORST / AVG / BEST CASES	Min (ms)	Avg (ms)	Max (ms)
Naive : Query Time	20	655.0	1506
Exponential Steps : Preprocessing + Query	45	300.1	569
Tarjan	95	474.8	798

ANALYSIS

- *Smallest overall time*
  - *Exponential Optimization*
- *Best Case: Similar performances*
- *Tarjan seems to have a smaller relative variance*
  - High constant?

Bibliography

Design and Analysis of Algorithms, Benjamin DOERR, 2017

Inverse Ackermann function: http://www.gabrielnivasch.org/fun/inverse-ackermann