Optimizing Distributed Graph Coloring

Dante Durrman and Erik Saule

The University of North Carolina at Charlotte

INRIA Seminar; May 28, 2024

Table of Content

1. Dataflow Graph Algorithms
2. An experimental study [PDCO 2023]
3. Towards a Framework to Evaluate Orderings Theoretically
4. Conclusion

Parallel Dataflow Graph Algorithms

From 3000ft

- Every vertex needs to be processed once
- Two vertices can not process simultaneously
- A total order on the vertices
- After processing, vertices unlocks their neighbors

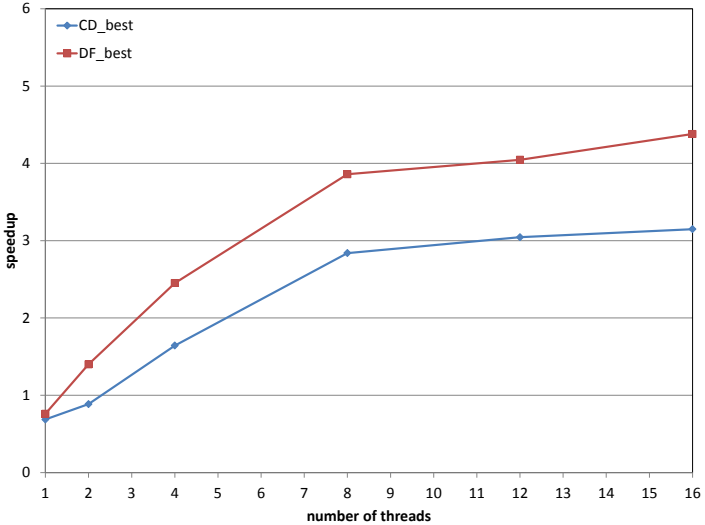
Applications

- Independent Set
- Graph Coloring
- Matching problems

Why care?

- Inherently distributed
  - Only use local information
- They implement well in a Gather-Apply-Scatter model
- Even in shared memory, they may perform better than other approaches
  - Often have optimal complexity
  - Ordering but not locking
  - Optimistic alternative double cost

Dataflow coloring can be faster than Optimistic in Shared Memory



Maximum Independent Set in Dataflow

MIS Dataflow Algorithm

- Each vertex v picks unique random number $r(v)$ uniformly in $[0; 1)$
- v sends $r(v)$ to each of its neighbors u
- v notes which neighbors u have the property $r(u) > r(v)$
- v marks its own state as `unknown`
- v awaits a message from each of its neighbors u if $r(u) < r(v)$
- If the state of u is `marked`, the state of v is changed to `unmarked`
- After receiving messages from all neighbors u, if the state of v is `unknown`, the state of v is changed to `marked`
- v sends its state to all neighbors u, such that $r(u) > r(v)$
- All vertices in the `marked` state are a maximal independent set

Partial Orders in Dataflow Algorithms

Partial Orders

- Processing order of vertices is generated randomly
- Once the order is picked the vertices are processed from low to high in each neighborhood
- Execution time of algorithm depends on length of critical path
- Cost to determine other desirable properties is too high

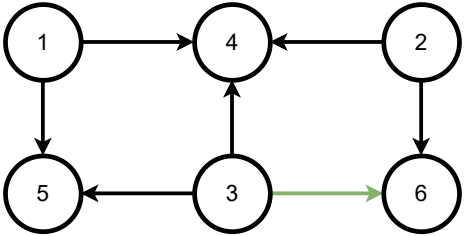


Figure 1: Lucky Draw

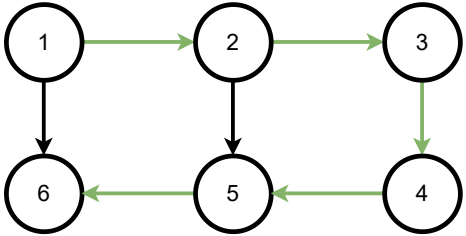


Figure 2: Unlucky Draw

Evaluating Partial Orders for Distributed Graph Algorithms

Critical Path

The length of the critical path is the longest weighted path in the graph. Since the weight of each vertex can be assigned differently depending on the model, the weight function is non-trivial.

Objective

We want to minimize the length of the critical path. That is our criteria for evaluating effectiveness of the partial order generated by the algorithm.

The Special Case of $w(v) = \delta(v)$

We use $w(v) = \delta(v)$, so the complexity of each algorithm for a given vertex is assumed to be $O(n(v))$, which is common to many dataflow algorithms.

The Special Case of $w(v) = 1$

Easier to address and model algorithms where the communication latency dominates the calculation.

Dataflow Algorithms and the Interval Coloring Problem

Gallai–Hasse–Roy–Vitaver Theorem

Finding a partial order that minimizes the critical path is equivalent to constructing a coloring that minimizes the number of colors used.

- Longest chain is the same as classical coloring
- Longest weighted chain is the same as coloring vertices with intervals

The true problem

Select an order on the graph, that implies a dependency graph that minimizes the length of the (weighted) longest path.
Ideally that orientation selection is cheaper than executing the graph. So ideally, a distributed algorithm that is $O(1)$ on each vertex.
So really, this is a distributed graph coloring problem.

Table of Content

1. Dataflow Graph Algorithms
2. An experimental study [PDCO 2023]
3. Towards a Framework to Evaluate Orderings Theoretically
4. Conclusion

Deriving Better Partial Orders for Distributed Graph Algorithms

`Uniform` $[0; 1)$

- Existing method of random number generation in dataflow algorithms

`Linear` $[0; \delta(v))$

- v is guaranteed to be after all vertices u, such that $\delta(u) = \delta(v) - 1$ with probability $\frac{1}{\delta(v)}$
- Good approximation of Largest Degree First with vertices of dramatic difference in degrees
- Poor approximation when $\Delta(G)$ is large and G has many vertices of large degrees

`Exponential` $[0; 2^{\delta(v)})$

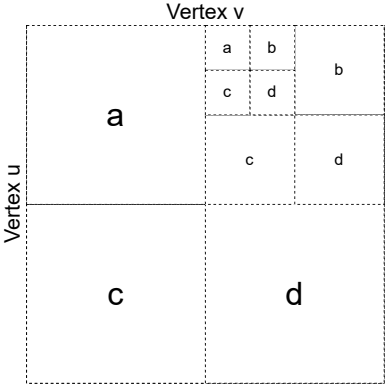
- v is guaranteed to be after all vertices u, such that $\delta(u) = \delta(v) - 1$ with probability $> \frac{1}{2}$
- Better approximation of Largest Degree First

- The communication and computational cost is the same for each algorithm.
- We are still sampling uniformly in those intervals despite the naming convention.

Introduction to RMAT Graphs

Construction of RMAT Graphs

- 2^n nodes
- Recursively split square matrix into 4 quadrants: a, b, c, d
- Each quadrant has an associated probability that a given edge will fall into that quadrant: $a + b + c + d = 1$
- Edges are generated one at a time and placed in a quadrant recursively following those probabilities until the edge is placed in a 1×1 submatrix.
- $ef * 2^n$ edges



RMAT Graphs Study

Methodology

- Sampled RMAT parameter space with constant *ef*
- Computed critical path length for each algorithm
- Calculated 95% confidence intervals
- Computed pairwise ratios of critical paths
- Conducted Z-Test to validate statistical significance

Results

- `Exponential` path < `Linear` path < `Uniform` path
- `Exponential` was never worse than `Uniform`
- At best, `Exponential` was 50% better
- On average, `Exponential` was about 10% better

a	b	c	d	Uniform CI	Exponential CI	Linear CI	U/E	U/L	L/E
0.30	0.28	0.28	0.14	[2444; 2447]	[2623; 2625]	[2850; 2853]	1.123	1.033	1.087
0.40	0.24	0.24	0.12	[4950; 4953]	[4680; 4683]	[4859; 4862]	1.058	1.019	1.038
0.50	0.20	0.20	0.10	[6968; 6971]	[6643; 6647]	[6845; 6848]	1.049	1.018	1.030
0.60	0.16	0.16	0.08	[8353; 8357]	[7949; 7952]	[8175; 8178]	1.051	1.022	1.028
0.70	0.12	0.12	0.06	[9211; 9214]	[8738; 8740]	[8987; 8990]	1.054	1.025	1.029
0.30	0.28	0.17	0.25	[2111; 2113]	[2064; 2066]	[2103; 2105]	1.023	1.004	1.019
0.30	0.28	0.25	0.17	[2633; 2635]	[2437; 2439]	[2583; 2585]	1.080	1.019	1.060
0.30	0.28	0.34	0.08	[3782; 3785]	[3112; 3115]	[3510; 3513]	**1.215**	1.077	1.128
0.30	0.35	0.14	0.21	[2625; 2627]	[2047; 2049]	[2402; 2404]	**1.282**	1.093	**1.173**
0.30	0.35	0.21	0.14	[3064; 3067]	[2464; 2467]	[2825; 2827]	**1.243**	1.085	1.146
0.30	0.35	0.28	0.07	[3959; 3962]	[3221; 3225]	[3644; 3647]	**1.229**	1.086	1.131
0.30	0.42	0.11	0.17	[3632; 3635]	[2181; 2183]	[2880; 2883]	**1.665**	**1.261**	**1.321**
0.30	0.42	0.17	0.11	[3821; 3824]	[2433; 2436]	[3061; 3064]	**1.570**	**1.248**	**1.258**
0.30	0.42	0.22	0.06	[4343; 4346]	[3110; 3113]	[3685; 3688]	**1.396**	**1.178**	**1.185**
0.30	0.49	0.08	0.13	[4852; 4855]	[2245; 2249]	[3437; 3441]	**2.160**	**1.411**	**1.530**
0.30	0.49	0.13	0.08	[4775; 4778]	[2505; 2508]	[3325; 3330]	**1.906**	**1.435**	**1.328**
0.30	0.49	0.17	0.04	[5052; 5056]	[3151; 3155]	[3883; 3887]	**1.603**	**1.301**	**1.232**
0.40	0.24	0.14	0.22	[3246; 3249]	[3185; 3188]	[3242; 3245]	1.019	1.001	1.018
0.40	0.24	0.22	0.14	[4571; 4574]	[4377; 4380]	[4509; 4512]	1.044	1.014	1.030
0.40	0.24	0.29	0.07	[5946; 5950]	[5500; 5504]	[5759; 5763]	1.081	1.032	1.047
0.40	0.30	0.12	0.18	[4026; 4029]	[3457; 3460]	[3811; 3814]	**1.165**	1.056	1.102
0.40	0.30	0.18	0.12	[5003; 5006]	[4551; 4555]	[4821; 4825]	1.099	1.038	1.059
0.40	0.30	0.24	0.06	[6141; 6144]	[5652; 5656]	[5932; 5935]	1.086	1.035	1.049
0.40	0.36	0.10	0.14	[4903; 4906]	[3767; 3771]	[4333; 4336]	**1.301**	1.132	1.150
0.40	0.36	0.14	0.10	[5506; 5509]	[4637; 4641]	[5071; 5075]	**1.187**	1.086	1.094
0.40	0.36	0.19	0.05	[6383; 6387]	[5684; 5688]	[6045; 6049]	1.123	1.056	1.063
0.40	0.42	0.07	0.11	[5667; 5670]	[3901; 3906]	[4639; 4643]	**1.452**	**1.221**	**1.189**
0.40	0.42	0.11	0.07	[6196; 6199]	[4927; 4932]	[5478; 5483]	**1.257**	1.131	1.112
0.40	0.42	0.14	0.04	[6670; 6674]	[5681; 5685]	[6132; 6136]	**1.174**	1.088	1.079
0.50	0.20	0.12	0.18	[5009; 5012]	[4881; 4884]	[4981; 4984]	1.026	1.006	1.020
0.50	0.20	0.18	0.12	[6489; 6492]	[6213; 6216]	[6399; 6402]	1.044	1.014	1.030
0.50	0.20	0.24	0.06	[7813; 7816]	[7385; 7368]	[7616; 7620]	1.061	1.026	1.034
0.50	0.25	0.10	0.15	[5687; 5690]	[5230; 5234]	[5515; 5519]	1.087	1.031	1.054
0.50	0.25	0.15	0.10	[6920; 6924]	[6550; 6554]	[6748; 6751]	1.065	1.026	1.038
0.50	0.25	0.20	0.05	[7984; 7987]	[7503; 7507]	[7766; 7770]	1.064	1.028	1.035
0.50	0.30	0.08	0.12	[6294; 6297]	[5524; 5528]	[5943; 5946]	1.139	1.059	1.076
0.50	0.30	0.12	0.08	[7263; 7267]	[6644; 6648]	[6969; 6972]	1.093	1.042	1.049
0.50	0.30	0.16	0.04	[8073; 8076]	[7495; 7499]	[7786; 7789]	1.077	1.037	1.039
0.50	0.35	0.06	0.09	[6812; 6815]	[5778; 5781]	[6278; 6281]	**1.179**	1.085	1.087
0.50	0.35	0.09	0.06	[7537; 7540]	[6729; 6732]	[7108; 7111]	1.120	1.060	1.056
0.50	0.35	0.12	0.03	[8135; 8138]	[7420; 7423]	[7754; 7757]	1.096	1.049	1.045
0.60	0.16	0.10	0.14	[6491; 6495]	[6204; 6208]	[6406; 6409]	1.046	1.013	1.032
0.60	0.16	0.14	0.10	[7774; 7777]	[7418; 7422]	[7631; 7635]	1.048	1.019	1.029
0.60	0.16	0.19	0.05	[9077; 9080]	[8613; 8616]	[8843; 8846]	1.054	1.026	1.027
0.60	0.20	0.08	0.12	[6942; 6945]	[6427; 6431]	[6741; 6744]	1.080	1.030	1.049
0.60	0.20	0.12	0.08	[8257; 8260]	[7803; 7806]	[8044; 8048]	1.058	1.026	1.031
0.60	0.20	0.16	0.04	[9253; 9256]	[8754; 8757]	[8997; 9000]	1.057	1.028	1.028
0.60	0.24	0.06	0.10	[7261; 7264]	[6516; 6519]	[6926; 6929]	1.114	1.048	1.063
0.60	0.24	0.10	0.06	[8597; 8600]	[8041; 8044]	[8308; 8311]	1.069	1.035	1.033
0.60	0.24	0.13	0.03	[9283; 9286]	[8731; 8734]	[8987; 8990]	1.063	1.033	1.029
0.60	0.28	0.05	0.07	[7829; 7832]	[6958; 6962]	[7380; 7383]	1.125	1.061	1.061
0.60	0.28	0.07	0.05	[8537; 8540]	[7838; 7841]	[8157; 8160]	1.089	1.047	1.041
0.60	0.28	0.10	0.02	[9249; 9252]	[8631; 8634]	[8900; 8903]	1.072	1.039	1.031
0.70	0.12	0.07	0.11	[7229; 7232]	[6852; 6856]	[7101; 7105]	1.055	1.018	1.036
0.70	0.12	0.11	0.07	[8854; 8857]	[8424; 8427]	[8659; 8662]	1.051	1.023	1.028
0.70	0.12	0.14	0.04	[9813; 9816]	[9197; 9200]	[9514; 9517]	1.067	1.031	1.034
0.70	0.15	0.06	0.09	[7797; 7800]	[7252; 7255]	[7573; 7576]	1.075	1.030	1.044
0.70	0.15	0.09	0.06	[9093; 9096]	[8589; 8592]	[8850; 8853]	1.059	1.027	1.030
0.70	0.15	0.12	0.03	[10032; 10035]	[9347; 9350]	[9694; 9696]	1.073	1.035	1.037
0.70	0.18	0.05	0.07	[8267; 8270]	[7587; 7591]	[7941; 7945]	1.090	1.041	1.047
0.70	0.18	0.07	0.05	[9183; 9186]	[8599; 8602]	[8878; 8879]	1.068	1.035	1.032
0.70	0.18	0.10	0.02	[10074; 10076]	[9370; 9372]	[9700; 9702]	1.075	1.039	1.035
0.70	0.21	0.04	0.05	[8667; 8669]	[7899; 7902]	[8263; 8266]	1.097	1.049	1.046
0.70	0.21	0.05	0.04	[9161; 9171]	[8523; 8506]	[8798; 8800]	1.078	1.042	1.035
0.70	0.21	0.07	0.02	[9644; 9846]	[9198; 9200]	[9470; 9472]	1.070	1.039	1.030

Understanding the Results of the RMAT Graphs Study

Why is `Exponential` better on RMAT Graphs

- RMAT Graphs have the same properties of a social network
- Social networks are "Onion-like" - dense core, but outer layers become less dense
- `Exponential` is similar to Largest First

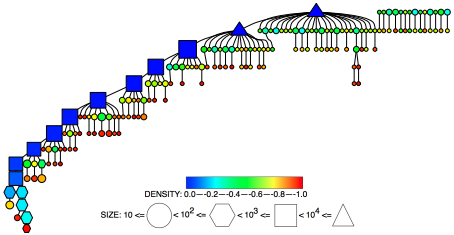


Figure 3: Hierarchy of Dense Subgraphs by Sariyuce et al. (2015)

Real World Applications

Real World Application

- Conducted similar experiment on real world graphs from SNAP
- All graphs have small world properties, except roads of Pennsylvania
- `Exponential` was better except on ca-HepPh and roadNet-PA

Name	Vertices	Edges	Max Degree	Clustering Coefficient	Diameter	Uniform CI	Exponential CI	Linear CI	U/E	U/L	L/E
CA-HepPh	89,209	118,521	491	0.6115	13	[1030; 1036]	[1040; 1045]	[1032; 1037]	0.991	0.999	0.992
Email-Enron	36,692	183,831	1,383	0.4970	11	[43437; 43720]	[38836; 38982]	[40688; 41002]	1.120	1.067	1.050
p2p-Gnutella04	10,879	39,994	103	0.0062	9	[911; 925]	[568; 575]	[728; 740]	**1.606**	**1.251**	**1.284**
roadNet-PA	1,090,920	1,541,898	9	0.0465	786	[49; 49]	[49; 50]	[48; 49]	0.990	1.010	0.980
soc-Epinions1	75,888	405,740	3,044	0.1378	14	[94793; 95270]	[88297; 88593]	[89488; 90034]	1.074	1.059	1.015
soc-pokec-relationships	1,632,804	22,301,964	14,854	0.1094	11	[118924; 119528]	[96958; 97239]	[100836; 101775]	**1.228**	**1.177**	1.043
web-Google	916,428	4,322,051	6,332	0.5143	21	[80466; 81618]	[18166; 18192]	[20577; 21084]	**4.458**	**3.891**	1.146
WikiTalk	2,394,385	4,659,565	100,029	0.0526	9	[1352414; 1357165]	[1101248; 1103043]	[1145942; 1151894]	**1.229**	**1.179**	1.042

Longest Path of the Real World Application Graphs

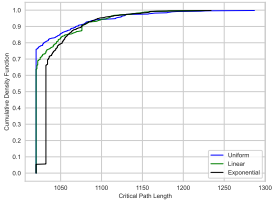


Figure 4: ca-HepPh

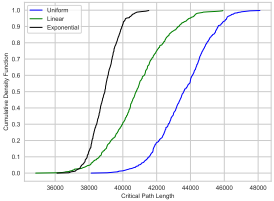


Figure 5: email-Enron

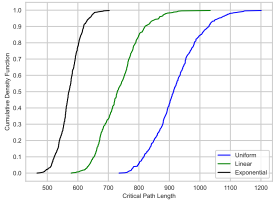


Figure 6: p2p-Gnutella04

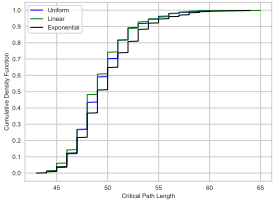


Figure 7: roadNet-PA

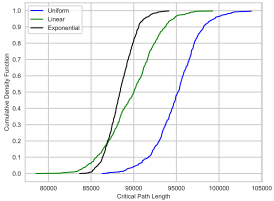


Figure 8: soc-Epinions1

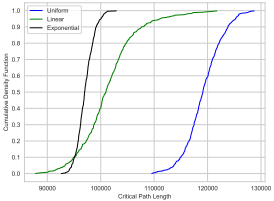


Figure 9: soc-pokec

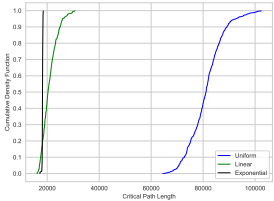


Figure 10: web-Google

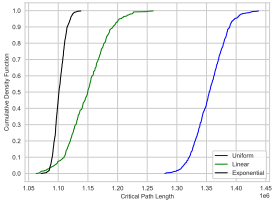


Figure 11: wiki-Talk

Summary of Work on Distributed Dataflow Algorithms

Summary of Work

- We provided a model for dataflow algorithms as a distributed graph coloring problem.
- We presented new ways to generate partial orderings and provide a theoretical argument for why they are sound.
- We studied the behavior of algorithms using different partial orders on both randomly generated RMAT graphs and graphs from real world applications.
- We provided an argument using statistical evidence to show that our methods perform usually better.

Summary of Work on Distributed Dataflow Algorithms

Summary of Work

- We provided a model for dataflow algorithms as a distributed graph coloring problem.
- We presented new ways to generate partial orderings and provide a theoretical argument for why they are sound.
- We studied the behavior of algorithms using different partial orders on both randomly generated RMAT graphs and graphs from real world applications.
- We provided an argument using statistical evidence to show that our methods perform usually better.

What we don't know

Why?

Can we prove that Exponential is always better in average?
Can we identify some types of graphs where this is true?

Table of Content

1. Dataflow Graph Algorithms
2. An experimental study [PDCO 2023]
3. Towards a Framework to Evaluate Orderings Theoretically
4. Conclusion

Goal

Ideally

For graphs that are social network (that exhibit small world properties; diameter in $\log n$).
For arbitrary random order generators.
Derive close-form formula of the length of the longest chain

Goal

Ideally

For graphs that are social network (that exhibit small world properties; diameter in $\log n$).
For arbitrary random order generators.
Derive close-form formula of the length of the longest chain

What we'll actually show

For a particular type of graph: $G_{N,P}$ graphs, with average degree in $\Theta(\log n)$
For random uniform rank generator.
The longest chain is in $\Theta(\log N)$.

Theorem

$G_{N,P}$ Graphs

- A $G_{N,P}$ graph is constructed by connecting N labeled nodes randomly
- Each edge is generated independently with probability P
- We can choose P so that the average is degree is proportional to $\log N$

Theorem

In $G_{N,P}$ graphs with a uniformly random total order, as long as $P = \frac{c_1 \log N}{N}$, w.h.p there is no path L, such that $L \in \omega(\log N)$ for sufficiently large N. (That is to say $L > c_2 \log N$)

$E(L, N, P)$

Definition

$E(L, N, P)$ is an event, such that G contains a path of size exactly L

Definition

$E_i(L, N, P)$ is an event where the path starts on vertex v_i

Remark

The probability of having a path of size L is the probability of the event E for large L. We need to show that the sum over all large L goes to 0 as N goes to ∞.

It is OK to relabel according to total order

Lemma

$\mathbb{P}(E(L, N, P)) \leq \sum_{i=1}^{N} \mathbb{P}(E_0(L, N-i, P))$

Key Idea

$\mathbb{P}(E(L, N, P)) \leq \sum_{i=0}^{N} \mathbb{P}(E_i(L, N, P))$

We can sort vertices based on the total order while retaining the original properties of our $G_{N,P}$ graph. Hence, we can make all edges go from low index to high index without loss of generality. So there are no vertices between 0 and $i-1$ which are part of the path in $E_i(L, N, P)$. Since all edges are equiprobable, remove v_0 and relabel the vertex with the lowest remaining index to v_0. We have a graph that preserves our path of size L with $N-i$ vertices. Therefore, $E_i(L, N, P)$ is the same event as $E_0(L, N-i, P)$.

$F(N)$, a bound of $\mathbb{P}(E_0(L, N, P))$

Definition: $F(L, N, P)$, an upper bound of $\mathbb{P}(E_0(L, N, P))$

$F(L, N, P) = \sum_{i=1}^{N} P * F(L-1, N-i, P)$
with base case $F(L > 0, N = 0, P) = 0$ and $F(L = 1, N > 0, P) = 1$

Refining F

$$
\begin{aligned}
F(L, N, P) &= \sum_{i=1}^{N} P * F(L-1, N-i, P) && (1)\\
&= P^{L-1} \sum_{i=1}^{N} H(L-1, N-i) && (2)\\
&= P^{L-1} H(L, N) && (3)\\
&= P^{L-1} \binom{N}{L-1} && (4)
\end{aligned}
$$

- (2) because of fixed tree depth of a 0/1 product function
- H is a 0,1 function independent of P
- $H(L, N)$ is the number of possible path of length exactly L in a graph of N vertices
- H boils down to how many ways to pick $L-1$ 1s in a sequence of N values

Lemma: $G(N) = \sum_{L > c_2 \log N} t(L, N)$, such that $t(L, N) = \left(\frac{c_1 \log N}{N}\right)^{L-1} \binom{N+1}{L}$

Definition: G(N), an upper bound on the probability of a long path

$G(N) = \sum_{L > c_2 \log N} \sum_{M \leq N} F(L, M, \frac{c_1 \log N}{N})$

Lemma

If $G(N)$ goes to 0 as N approaches ∞, then the probability of a long path event also goes to 0.

Refining G

$$G(N) = \sum_{L > c_2 \log N} \sum_{M \leq N} F(L, M, \frac{c_1 \log N}{N}) \tag{5}$$

$$= \sum_{L > c_2 \log N} \sum_{M \leq N} \left(\frac{c_1 \log N}{N}\right)^{L-1} \binom{M}{L-1} \tag{6}$$

$$= \sum_{L > c_2 \log N} \left(\frac{c_1 \log N}{N}\right)^{L-1} \binom{N+1}{L} \tag{7}$$

$G(N)$ is dominated by first term

Lemma

If $c_2 > c_1$, $G(N) \leq Bt(c_2 \log(N), N)$ where B is a constant

Key Ideas

- t is a rapidly decreasing function of L
  - We can show $t(L+1, N) \leq \frac{1}{c} t(L, N)$ for $L > c_2 \log N$ and $c > 1$
  - This implies $B = \frac{1}{1-\frac{1}{c}}$ is constant
- We can easily choose c_1, c_2 so that $c_2 > c_1$
- Since $G(N)$ is dominated by t, we only really care about the t function

Study of $t'(N) = t(c_2 \log N, N)$

Lemma

$\lim_{N\to\infty} t'(N) = 0$.

Key Idea

It is sufficient to show $\lim_{N\to\infty} \frac{t'(N)}{t'(2N)} \geq constant > 1$.

Study of $t'(N) = t(c_2 \log N, N)$

Lemma

$\lim_{N\to\infty} t'(N) = 0$.

Key Idea

It is sufficient to show $\lim_{N\to\infty} \frac{t'(N)}{t'(2N)} \geq constant > 1$.

Expression of this ratio after a decent amount of manipulation

$$\frac{t'(N)}{t'(2N)} = \left(\frac{2N}{c_1 \log_2(N)+c_1}\right)^{c_2} \left(\frac{2\log_2(N)}{\log_2(N)+1}\right)^{c_2 \log_2(N)-1} \frac{\binom{N+1}{c_2 \log_2(N)}}{\binom{2N+1}{c_2 \log_2(N)+c_2}}$$

Study of $t'(N) = t(c_2 \log N, N)$

Expression of this ratio after a decent amount of manipulation

$$\frac{t'(N)}{t'(2N)} = \left(\frac{2N}{c_1 \log_2(N)+c_1}\right)^{c_2} \left(\frac{2\log_2(N)}{\log_2(N)+1}\right)^{c_2 \log_2(N)-1} \frac{\binom{N+1}{c_2 \log_2(N)}}{\binom{2N+1}{c_2 \log_2(N)+c_2}}$$

Approximation of $\binom{n}{k}$

For $k \in o(n)$, we can use the following approximation derived from Stirling's formula:

$$\binom{n}{k} \sim \left(\frac{ne}{k}\right)^k (2\pi k)^{-1/2} e^{\left(-\frac{k^2}{2n}(1+o(1))\right)}$$

We get this after clever rewriting and the approximation.

$$\frac{t'(N)}{t'(2N)} = \left(\frac{1}{2}\right) \left(\frac{c_2}{e \cdot c_1}\right)^{c_2} \left(\frac{2N}{2N+1}\right)^{c_2} \left(\frac{2N+2}{2N+1}\right)^{c_2 \log_2(N)} \left(\frac{\log_2(N)+1}{\log_2(N)}\right)^{\frac{3}{2}} \left(e^{\frac{(c_2 \log_2(N)+c_2)^2}{2N+1} - \frac{(c_2 \log_2(N))^2}{N+1}}\right)$$

Study of $t'(N) = t(c_2 \log N, N)$

We get this after clever rewriting and the approximation.

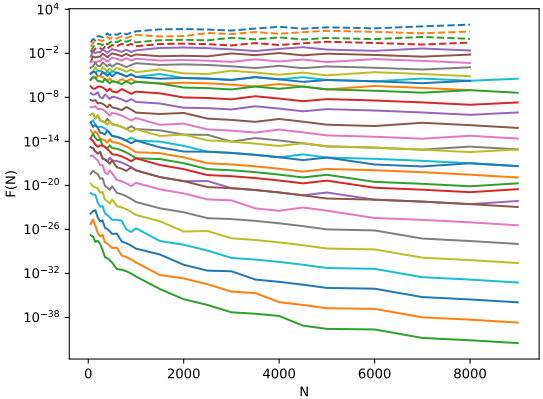
$$\frac{t'(N)}{t'(2N)} = \left(\frac{1}{2}\right)\left(\frac{c_2}{e \cdot c_1}\right)^{c_2}\left(\frac{2N}{2N+1}\right)^{c_2}\left(\frac{2N+2}{2N+1}\right)^{c_2 \log_2(N)}\left(\frac{\log_2(N)+1}{\log_2(N)}\right)^{\frac{3}{2}}\left(e^{\frac{(c_2 \log_2(N)+c_2)^2}{2N+1} - \frac{(c_2 \log_2(N))^2}{N+1}}\right)$$

Taking the limit as n goes to infinity we get.

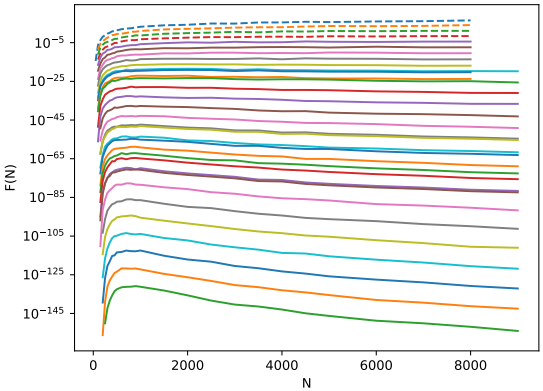
$$\lim_{N \to \infty} \frac{t'(N)}{t'(2N)} = \left(\frac{1}{2}\right)\left(\frac{c_2}{e \cdot c_1}\right)^{c_2} \tag{9}$$

Any choice of c_1, c_2, such that $\left(\frac{c_2}{e \cdot c_1}\right)^{c_2} > 2$ yields a constant limit for $\frac{t'(N)}{t'(2N)}$ greater than 1.

Numerical Results



$c_1 = 1.1$



$c_1 = 5$

Will this generalize?

Putting this all together...

Given a uniform edge orientation of a $G_{N,P}$ graph

- We are interested in the probability of a long path
- An individual path probability is $F(N)$
- We define $G(N)$, an upper bound of this probability
- We show that $G(N)$ is dominated by its first term
- We show that this term goes to 0 as N gets sufficiently large
- Therefore, the longest path is in $\Theta(\log N)$

Some issues

- The reordering only works because $G_{N,P}$ graphs have identical independent probability edges (necessary for F derivations)
- The extraction of combinatorial function H only works because the random orderings are uniform
- The technique only shows that likely long path are in $\Theta(\log n)$ and it is unlikely that exponential would be in a lower order of magnitude

Table of Content

1. Dataflow Graph Algorithms
2. An experimental study [PDCO 2023]
3. Towards a Framework to Evaluate Orderings Theoretically
4. Conclusion

Conclusion

Statistical study for distributed edge orientations

- For very large, arbitrary graphs, we want a model that gives a decent solution quickly
- We showed that statistical tuning can yield better results
- If we know more about the graph we can use other properties to increase accuracy

Unweighted distributed problem on random graph models

- We looked at a particular graph generator: $G_{N,P}$
- We provided theoretical bounds on path length

Thank you!

In general, I am interested in problems where you have control in the instances of scheduling problems that get executed on a parallel system.