Brief Announcement: Super-Fast t-Ruling Sets

Tushar Bisht
International Institute of Information Technology
Hyderabad, INDIA, 500032
tushar.bisht@research.iiit.ac.in

Kishore Kothapalli
International Institute of Information Technology
Hyderabad, INDIA, 500032
kkishore@iiit.ac.in

Sriram V. Pemmaraju*
Department of Computer Science
The University of Iowa
Iowa City, IA 52242-1419
sriram-pemmaraju@uiowa.edu

ABSTRACT

A t-ruling set of a graph $G = (V, E)$ is a vertex-subset $S \subseteq V$ that is independent and satisfies the property that every vertex $v \in V$ is at a distance of at most t hops from some vertex in S. A *maximal independent set (MIS)* is a 1-ruling set. Extending results from Kothapalli et al. (FSTTCS 2012) this note presents a randomized algorithm for computing, with high probability, a t-ruling set in $O(t \cdot \log^{\frac{1}{t-1}} n)$ rounds for $2 \leq t \leq \sqrt{\log\log n}$ and in $\exp(O(\sqrt{\log\log n}))$ rounds for $t > \sqrt{\log\log n}$.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Computations on Discrete Structures*

Keywords

Distributed Algorithms; LOCAL Algorithms; Maximal Independent Sets; Ruling Sets; Symmetry Breaking

1. INTRODUCTION

Symmetry breaking is a fundamental theme in distributed computing and a classic example of symmetry breaking arises in the computation of a *maximal independent set* (MIS) of a given graph. There is a well known 25-year old randomized algorithm [6] that solves the MIS problem in $O(\log n)$ communication rounds. Since then, all attempts to devise an algorithm for MIS that runs in *sub-logarithmic* rounds (for general graphs) have failed. However, there has been some exciting recent progress on the problem. Barenboim et al. [1] present MIS algorithms that run in $O(\log \Delta \sqrt{\log n})$ rounds on arbitrary graphs (with n vertices and maximum degree Δ) and in $\exp(O(\sqrt{\log\log n}))$ rounds when $\Delta = \text{poly}(\log n)$.

A natural generalization of MIS is the problem of computing t-ruling sets. For $t \geq 1$, a *t-ruling set* of a graph $G = (V, E)$ is an independent subset S of vertices with the property that every vertex $v \in V$ is at a distance of at most t hops from some vertex in S. Thus an MIS is a 1-ruling set[1]. The main result of this paper is a t-ruling set algorithm that runs in $O(t \cdot \log^{\frac{1}{t-1}} n)$ rounds for $2 \leq t \leq \sqrt{\log\log n}$ and in $\exp(O(\sqrt{\log\log n}))$ rounds for $t > \sqrt{\log\log n}$. Our algorithms are designed to run in $\mathcal{LOCAL}$ model, which is a standard synchronous, message passing model of communication in which each node, in each round, can send a possibly distinct message of unbounded size along each incident edge. Our technique is randomized and involves an iterative, rapid sparsification of the graph while ensuring that nodes that are removed from further consideration are within one hop of some remaining node. When we complete all iterations of the algorithm, we can apply the MIS result due to Barenboim et al. [1], mentioned earlier, taking advantage of the low maximum degree of the graph that remains.

1.1 Related Work

In previous work [3], we showed how to compute a 2-ruling set in $O(\log^{3/4} n)$ rounds, with high probability. Our current work extends this result to $t \geq 3$.

Our results should also be viewed in the context of results by Gfeller and Vicari [2]. These authors showed how to compute, in $O(\log\log n)$ rounds, a vertex-subset T of a given n-vertex graph $G = (V, E)$ such that (i) every vertex is at most $O(\log\log n)$ hops from some vertex in T, and (ii) the subgraph induced by T has maximum degree $O(\log^5 n)$. One can use the MIS algorithm from Barenboim et al. [1] on $G[T]$ and sparsify T into an $O(\log\log n)$-ruling set in an additional $\exp(O(\sqrt{\log\log n}))$ rounds. Thus, by combining the Gfeller-Vicari algorithm with the Barenboim et al. algorithm one can compute an $O(\log\log n)$-ruling set in general graphs in $\exp(O(\sqrt{\log\log n}))$ rounds. Our current work improves on the Gfeller-Vicari result by allowing arbitrary t and also by being able to compute an $O(\sqrt{\log\log n})$-ruling set in $\exp(\sqrt{\log\log n})$ rounds.

Schneider et al. [7] consider the problem of computing an (α, β)-*ruling set*, which they define as a vertex-subset S of the input graph G such that (i) pairs of vertices in S are at distance at least α away from each other and (ii) every vertex in G is at distance at most β from some vertex in S. Thus a t-ruling set as per our definition is a $(2, t)$-ruling set as per their definition. These authors present an algorithm

*The work of this author is supported in part by National Science Foundation grant CCF 1318166.

PODC'14, July 15–18, 2014, Paris, France.
ACM 978-1-4503-2944-6/14/07.
http://dx.doi.org/10.1145/2611462.2611512.

[1]In the definition of Gfeller and Vicari [2], a t-ruling set need not be independent, and what we call a t-ruling set, they call an *independent* t-ruling set.

that can compute a $(2,t)$-ruling set in $t \cdot d^{1/t}$ rounds, given a d-coloring of the input graph. Preprocessing the input with a distributed graph coloring algorithm and then providing the colored graph as input to the above algorithm yields a $(2,t)$-ruling set algorithm. For example, using the classical distributed algorithm of Linial [5] that computes an $O(\Delta^2)$-coloring in $O(\log^* n)$ rounds, one can obtain a $(2,t)$-ruling set algorithm that runs in $O(t \cdot \Delta^{2/t} + \log^* n)$ rounds. For rather small Δ, i.e., $\Delta < \sqrt{\log n}$, this algorithm has smaller asymptotic running time relative to our algorithm, but not otherwise.

2. RAPID SPARSIFICATION

In this section we first describe Algorithm SPARSIFY-GG, that rapidly sparsifies a given graph. This algorithm was used in [3] as a step in computing a 2-ruling set and so the material in this section has largely appeared in [3]. Let f be some parameter that can be chosen later and i^* be the smallest positive integer such that $f^{i^*+1} \geq \Delta$. Thus $i^* = \lceil \log_f \Delta \rceil - 1$. The run-time of the algorithm depends on the parameter f. At the end of the algorithm, the following properties are satisfied: (i) all the vertices of the graph are labeled with one of {`inactive`, `active`} and every `inactive` node has an `active` neighbor, and (ii) the degree of the graph induced by the `active` vertices is bounded by $O(f \cdot \log n)$.

The algorithm proceeds in *stages* and there are i^* stages, indexed by $i = 1, 2, \ldots, i^*$. In Stage i every vertex independently joins a set M_i with probability $\frac{6 \log n \cdot f^i}{\Delta}$ and is marked `active` (Line 4). It is shown in [3] that any vertex that is in V at the start of Stage i has degree at most Δ/f^{i-1}, with high probability. Therefore, it is easy to see that any vertex in the graph induced by M_i has expected degree at most $O(f \cdot \log n)$. In fact, this is true with high probability and this fact can be shown by appealing to the fact that vertices join M_i independently and using Chernoff bounds. Following the identification of the set M_i, all neighbors of M_i that are outside M_i are placed in a set W_i and marked `inactive` (Line 7). Both sets M_i and W_i are then deleted from the vertex set V. The sets W_i play a critical role in our algorithm. Given the probability $\frac{6 \log n \cdot f^i}{\Delta}$ of joining M_i, we can show that with high probability every vertex with degree more than Δ/f^i ends up either in M_i or in W_i. This establishes the degree-bound needed for next iteration of the algorithm. Also, the sets W_i act as "buffers" between the M_i's ensuring that there are no edges between M_i and $M_{i'}$ for $i \neq i'$. As a result the graph induced by $\cup_i M_i$ has degree $O(f \cdot \log n)$.

Algorithm SPARSIFY-GG($G = (V, E), f$)
1. **for** $i \leftarrow 1, 2, \ldots, i^*$ **do**
2. $\quad M_i \leftarrow \emptyset$; $W_i \leftarrow \emptyset$;
3. $\quad$ **for** each $v \in V$ *in parallel* **do**
4. $\quad\quad$ With probability $\frac{6 \log n \cdot f^i}{\Delta}$ add v to M_i and mark v as `active`
5. $\quad$ **for** each $v \in V$ *in parallel* **do**
6. $\quad\quad$ **if** $v \in$ Neighbors(M_i) $\setminus M_i$ **then**
7. $\quad\quad\quad$ Add v to W_i and mark v as `inactive`
8. $\quad V \leftarrow V \setminus (M_i \cup W_i)$
9. **end-for**(i)
10. Mark the vertices remaining in V as `active`
11. $S \leftarrow \{v \in V \mid v$ is `active`$\}$.
12. **return** S

The proof of the following theorem appears in [3].

THEOREM 1. *Let G be an arbitrary n-vertex graph with maximum degree Δ. With high probability, Algorithm Sparsify-GG with input G and f runs in $O(\log_f \Delta)$ rounds and produces a vertex-subset $S \subseteq V(G)$ such that $\Delta(G[S]) \in O(f \cdot \log n)$, and every vertex in V is either in S or has a neighbor in S.*

3. COMPUTING RULING SETS

The main idea behind out t-ruling set algorithm consists of iteratively sparsifying the given graph $t-1$ times using Algorithm SPARSIFY-GG and then computing an MIS. Let $G = (V, E)$ be the given graph and let $S_0 = V$. Let $f_0, f_1, \ldots$ be a sequence of parameters, whose values will be specified later. For $i = 1, 2, \ldots, t-1$, we execute SPARSIFY-GG($G[S_{i-1}], f_{i-1}$) in order to compute S_i. Finally, we compute MIS on $G[S_{t-1}]$ and return this as the t-ruling set of G. It turns out that setting f_{i-1} to $2^{(\log n)^{(t-1-i)/(t-1)}}$ for $i = 1, 2, \ldots, t-2$ and then setting f_{t-2} to $\log n$ nicely balances the running time of all of the sparsification steps so that each call to SPARSIFY-GG takes $(\log n)^{1/(t-1)}$ rounds. Pseudocode of this algorithm appears below in Algorithm t-RULINGSET-GG. Correctness of the algorithm follows from the next lemma.

LEMMA 1. *For each i, $0 \leq i \leq t$, with high probability every vertex in V is at most i hops from some vertex in S_i. Thus, with high probability, S_t is a t-ruling set.*

PROOF. Every vertex in V is 0 hops away from some vertex in $S_0 = V$. Suppose that every vertex in V is i hops away from S_i, for some $0 \leq i < t$. From Theorem 1 we know that every vertex in S_i is at most one hop away from some vertex in S_{i+1}. The result follows by induction. $\square$

Algorithm t-RULINGSET-GG($G = (V, E)$) (for $t \geq 2$)

$S_0 \leftarrow V$
for $i \leftarrow 1$ **to** $t-2$ **do**
$\quad f_{i-1} \leftarrow 2^{(\log n)^{(t-1-i)/(t-1)}}$
$\quad S_i \leftarrow$ SPARSIFY-GG($G[S_{i-1}], f_{i-1}$)
$f_{t-2} \leftarrow \log n$
$S_{t-1} \leftarrow$ SPARSIFY-GG($G[S_{t-2}], f_{t-2}$)
$S_t \leftarrow \mathtt{MIS}(G[S_{t-1}])$
return S_t

THEOREM 2. *With high probability, Algorithm t-RULINGSET-GG runs in time $O(t \cdot (\log n)^{1/(t-1)}) + \exp(O(\sqrt{\log \log n}))$.*

PROOF. For i, $0 \leq i \leq t$, let Δ_i be the maximum degree of a vertex in $G[S_i]$. Now consider i, $0 \leq i \leq t-2$. Since S_i is computed by calling SPARSIFY-GG($G[S_{i-1}], f_{i-1}$) and the value of f_{i-1} is $2^{(\log n)^{(t-1-i)/(t-1)}}$, it follows from Theorem 1 that Δ_i is bounded above by

$$O\left(\log n \cdot 2^{(\log n)^{(t-1-i)/(t-1)}}\right).$$

Also from Theorem 1 it follows that the call to subroutine SPARSIFY-GG($G[S_{i-1}], f_{i-1}$) takes $O(\log_{f_{i-1}} \Delta_{i-1})$ rounds. We obtain an upper bound on $\log_{f_{i-1}} \Delta_{i-1}$ as follows:

$$\begin{aligned} \log_{f_{i-1}} \Delta_{i-1} &\leq \frac{(\log n)^{(t-1-(i-1))/(t-1)} + \log \log n}{(\log n)^{(t-1-i)/(t-1)}} \\ &\leq (\log n)^{1/(t-1)} + \log \log n. \end{aligned}$$

Thus the call to Sparsify-GG($G[S_{i-1}], f_{i-1}$) takes

$$O((\log n)^{1/(t-1)} + \log\log n)$$

rounds. Also note that $\Delta_{t-2} = O(\log n \cdot 2^{(\log n)^{1/(t-1)}})$ and thus the call (in Line 6) to Sparsify-GG($G[S_{t-2}], f_{t-2}$) takes an additional

$$O(\log_{f_{t-2}} \Delta_{t-2}) = O((\log n)^{1/(t-1)} + \log\log n)$$

rounds. Thus all $t-1$ calls to the Sparsify-GG subroutine take a total of $O(t((\log n)^{1/(t-1)} + \log\log n))$ rounds. Now note that Δ_{t-1} is bounded by $O(f_{t-2} \cdot \log n) = O(\log^2 n)$ and therefore using the MIS algorithm (version 2) of Barenboim et al. (Theorem 4.3, [1]) we compute S_t in an additional $\exp(O(\sqrt{\log\log n}))$ rounds. The result follows. $\square$

Corollary 1. *For $t \leq \sqrt{\log\log n}$, we can compute a t-ruling set in $O(t \cdot (\log n)^{1/(t-1)})$ rounds and if $t > \sqrt{\log\log n}$, we can compute a t-ruling set in $\exp(O(\sqrt{\log\log n}))$.*

4. CONCLUSIONS

Kuhn et al. [4] claim a $\Omega(\sqrt{\log n})$ lower bound on the computation of an MIS of an n-vertex input graph in the $\mathcal{LOCAL}$ model. Our results show that even for small constant t, one can compute a t-ruling set faster than it is possible to compute an MIS.

There are no known lower bounds for the t-ruling set problem in the $\mathcal{LOCAL}$ model and so it is possible that our results can be improved. In light of our results, it is interesting to explore whether the lower bound technique of Kuhn et al. [4] can be extended to derive a lower bound for t-rulings set

5. REFERENCES

[1] Leonid Barenboim, Michael Elkin, Seth Pettie, and Johannes Schneider. The locality of distributed symmetry breaking. In *Proc. of IEEE FOCS*, 2012.

[2] Beat Gfeller and Elias Vicari. A randomized distributed algorithm for the maximal independent set problem in growth-bounded graphs. In *Proc. ACM PODC*, pages 53–60, 2007.

[3] Kishore Kothapalli and Sriram V. Pemmaraju. Super-fast 3-ruling sets. In *In Proc. of Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 136–147, 2012.

[4] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. Local computation: Lower and upper bounds. *CoRR*, abs/1011.5470, 2010.

[5] N. Linial. Locality in distributed graph algorithms. *SIAM Journal of Computing*, 21:193–201, 1992.

[6] M. Luby. A simple parallel algorithm for the maximal independent set. *SIAM Journal on Computing*, 15:1036–1053, 1986.

[7] Johannes Schneider, Michael Elkin, and Roger Wattenhofer. Symmetry breaking depending on the chromatic number or the neighborhood growth. *Theor. Comput. Sci.*, 509:40–50, October 2013.