What Can We Compute in a Single Round of the Congested Clique?

PETER ROBINSON, Augusta University, Georgia, USA

We show that any one-round algorithm that computes a minimum spanning tree (MST) in the unicast congested clique must use a link bandwidth of $\Omega(\log n)$ bits in the worst case. Consequently, computing an MST under the standard assumption of $O(\log n)$-size messages requires at least 2 rounds. This is the first round complexity lower bound in the unicast congested clique for a problem where the output size is small, i.e., $O(n \log n)$ bits. Our lower bound holds as long as every edge of the MST is output by an incident node. To the best of our knowledge, all prior lower bounds for the unicast congested clique either considered problems with large output sizes (e.g., triangle enumeration) or required every node to learn the entire output.

CCS Concepts: • Theory of computation → Distributed algorithms.

Additional Key Words and Phrases: Congested clique, distributed graph algorithm, lower bound

1 INTRODUCTION

In this work, we investigate the necessary link bandwidth for computing a Minimum Spanning Tree (MST) in the congested clique model [12], where there are $n$ nodes that are associated with the vertices of an input graph $G$. The nodes can communicate with each other over the links of the clique-topology in synchronous rounds, and the goal is to compute the solution to a graph problem on $G$. We consider the powerful unicast congested clique, where each node can send a (possibly distinct) message to every other node in the network. Our work aims at understanding the computational power of the congested clique with respect to algorithms that compute their output based on the received messages after just a single round, while assuming that nodes have access to a shared source of randomness. We use UCC$_1$ to denote the one-round variant of the unicast congested clique.

We assume that the nodes are assigned unique IDs chosen from $\{1, \ldots, n\}$. In addition, we equip the nodes with a crucial amount of initial knowledge of the input graph, in the sense that each node knows not only its own ID but also the IDs of all of its neighbors, which is known as the KT$_1$ assumption in previous literature (see [2, 10, 15, 19]).

2 OUR CONTRIBUTIONS AND RELATED WORK

The problem of computing a minimum spanning tree is, without doubt, one of the most-studied graph problems in distributed computing and, in particular, in the congested clique model. Here, we focus our discussion on the prior work in the congested clique and refer the reader to the survey of [17] for a more comprehensive discussion of distributed MST algorithms in general networks. [12] gave a deterministic $O(\log \log n)$-round algorithm, and, more recently, [6] used the linear graph sketches of [1] to obtain a $O(\log \log \log n)$-round randomized algorithm. Subsequently, [5] showed that $O(\log^* n)$ rounds are possible by using sketches that are sensitive to the node degrees. Shortly after, [9] presented an
O(1)-round randomized algorithm, and [14] showed that O(1) rounds are possible even deterministically. These results crucially rely on the fast routing algorithm of [11], which itself requires 16 rounds (see Theorem 3.7 in [11]), and it is unclear whether any of these algorithms can be adapted to requiring just a single round, without increasing the necessary link bandwidth to \( \Omega(n) \) bits.

**Theorem 2.1.** Any randomized algorithm that, with probability at least \( \frac{3}{4} \), computes an MST on an \( \mathcal{U} \)-node graph in a single round of the unicast congested clique, requires a link bandwidth of \( \Omega(\log^3 n) \) bits. This holds even if nodes have access to shared randomness, and when each edge of the MST is output by at least one of its two endpoints.

An immediate consequence of Theorem 2.1 is that computing an MST requires at least two rounds under the standard assumption of limiting the link bandwidth to \( O(\log n) \) bits. We are not aware of any previous round complexity lower bounds for problems in the unicast congested clique where each node may output only part of the solution and the total size of the output is \( O(n \log n) \) bits, which is significantly smaller than the available per-round bandwidth of \( O(n^2 \log n) \).

The lack of existing lower bounds comes as no surprise: the seminal work of [3] proved that lower bounds on the required number of rounds in the (multi-round) unicast congested clique is notoriously difficult for problems with small output sizes, as even slightly super-constant lower bounds (i.e. \( \Omega(\log \log n) \)) would yield new lower bounds on the circuit complexity of threshold gates. Consequently, the only existing lower bounds for the unicast congested clique are shown for problems with large outputs. For instance, in previous work [18], we showed a lower bound of \( \tilde{\Omega}(n^{1/3} / \log^3 n) \) rounds for triangle listing, which, subsequently, was improved by Izumi and Le Gall [7] to \( \tilde{\Omega}(n^{1/3} \log n) \) rounds. We point out that triangle listing has an output size of \( \tilde{\Omega}(n^{7/3}) \) in some graphs, and hence the round complexity bound stems from the fact that the per-round bandwidth is limited to \( \tilde{O}(n^2) \) bits in the congested clique.

The work of [13] gives exact bounds on the time complexity of the graph reconstruction problem in the broadcast congested clique, under the assumption that every node must output all edges of the input graph. In particular, they show that reconstructing the edges of an arbitrary input graph \( G \) requires 2 rounds in the broadcast congested clique. However, we note that their result does not hold under the relaxed assumption that each edge that is part of the solution must be output by some incident node, i.e., the solution to the problem is the union of the nodes’ outputs, which is what we assume in Theorem 2.1.

### 3 PROOF OF THEOREM 2.1

In this section, we prove that computing an MST in just one round of the unicast congested clique model is impossible under the standard assumption of \( O(\log n) \) bits per message.

Consider any UCC\(_1\) algorithm \( \mathcal{A} \) that correctly outputs an MST with probability \( \frac{3}{4} \), after just one round, while sending at most \( L \) bits over each link. We now describe how to use \( \mathcal{A} \) in the distributed graph sketching (SKETCH) model to solve graph connectivity, where the input graph is assigned to \( n \) players, such that each player obtains one vertex and
What Can We Compute in a Single Round of the Congested Clique?

Fig. 1. The hard input graph $H$ on which computing an MST requires messages of $\Omega(\log^3 n)$ bits for one-round algorithms in the unicast congested clique. The green-shaded regions are the blocks $G_1$, $G_2$, and $G_3$, which form the lower bound graph $G$ of [20]. (Note that the figure shows a simplified version of this graph where we omitted the parts that are not relevant for understanding our proof.) A snake-shaped line corresponds to a path of 0-weight edges. The orange vertices and edges are added to $G$ to form graph $H$. In each block $G_i$, vertex $s_1^{(i)}$ is either connected to $t_1^{(i)}$ or $t_2^{(i)}$. As shown in [20], the graph induced by these blocks (i.e. without $a$ and $b$) is connected if and only if the latter case happens in an odd number of blocks. Since all edges except $\{a, b\}$ have weight 0, the MST of $H$ will contain $\{a, b\}$ if and only if the subgraph induced by $G_1, G_2$, and $G_3$ is disconnected.

the incident edges as its input. The referee, on the other hand, receives a single message (“sketch”) from each player and must compute the output.

Our lower bound graph $H$ contains a certain subgraph $G$, which corresponds to the construction of Yu [20] and was used to show that a sketch length of $\Omega(\log^3 n)$ bits is required for deciding graph connectivity in the SKETCH model. In more detail, graph $G$ consists of $\beta = \Theta(\sqrt{n})$ interconnected blocks $G_0, \ldots, G_\beta$, comprising $\Theta(\sqrt{n})$ vertices each. A block $G_i$ contains four special vertices, $s_1^{(i)}$, $s_2^{(i)}$, $t_1^{(i)}$, and $t_2^{(i)}$ that connect $G_i$ to $G_{i+1} \mod \beta$. 

For simulating $\mathcal{A}$, we define $H$ to consist of a copy of $G$ and two additional nodes $a$ and $b$ with fixed unique IDs $id(a)$ and $id(b)$. Node $a$ is connected to $s_1^{(1)} \in G_1$, and $b$ is connected to $s_2^{(1)} \in G_1$. Moreover, there is an edge between $a$ and $b$. We assign weight 0 to all edges except for the edge $\{a, b\}$, which has weight 1. Figure 1 shows the resulting graph, assuming just three blocks.

Our proof makes use of the following properties of graph $G$:

**Lemma 3.1 ([20]).** The graph $G$ has at most 2 components. Vertices $s_1^{(1)}$ and $s_2^{(1)}$ lie in distinct components if and only if $G$ is not connected.

From the definition of our graph $H$, we are able to tie together the connectivity of $G$ with the inclusion of the edge $\{a, b\}$ in the MST of $H$:

**Observation 1.** Graph $H$ is connected. Moreover, any MST of $H$ contains the edge $\{a, b\}$ if and only if the subgraph $G$ induced by the vertices in $V(H) \setminus \{a, b\}$ is disconnected.

We now describe the details of simulating the given UCC$_1$ algorithm $\mathcal{A}$ in the SKETCH model: To determine the messages sent in the first (and only) round, each node simulates the local computation of $\mathcal{A}$ for its assigned vertex of $H$, by taking into account its neighborhood information and the shared randomness. Note that the referee knows the neighborhoods of nodes $a$ and $b$, because the IDs of their neighbors $s_1^{(1)}$ and $s_2^{(1)}$ are fixed in advance (see Sec. 4.1 in [20]). As a result, the referee can locally simulate the first (and only) round of algorithm $\mathcal{A}$ for nodes $a$ and $b$.

Every other node $u$ simply discards all messages produced by the simulation of $\mathcal{A}$, except for the messages that are addressed to either $a$ or $b$, which $u$ sends as its sketch to the referee instead. Recall that $u$ knows the IDs of these vertices since they are fixed in advance. Moreover, the messages that are produced by simulating algorithm $\mathcal{A}$ for a vertex $v$ only depend on $v$’s neighborhood and the (shared) randomness, both of which are readily available to the node in the SKETCH model who is responsible for $v$.

After receiving all messages from the players, the referee has learned all messages that were sent to $a$ and $b$, and it also knows the complete neighborhood information of these vertices. Thus, the referee can correctly simulate the output that $a$ and $b$ produce in the UCC$_1$ model after receiving their first round messages. Finally, the referee answers “connected” if and only if the edge $\{a, b\}$ is not included in the MST edges output by $a$ and $b$.

It follows from Lemma 3.1, Observation 1, and the fact that $\mathcal{A}$ computes an MST with probability at least $\frac{3}{4}$ that the referee is able to correctly decide whether $G$ is connected with precisely the same probability. Theorem 1 in [20] tells us that the worst case sketch length of any protocol that solves connectivity in SKETCH is $\Omega(\log^3 n)$. We conclude that the link bandwidth used by algorithm $\mathcal{A}$ must be at least $\Omega(\log^3 n)$ bits.

4 DISCUSSION AND OPEN PROBLEMS

Our work suggests several interesting avenues for future exploration. Since we have shown that multiple rounds are necessary for computing an MST in the congested clique with $O(\log n)$ bandwidth. On the other hand, the recent line of work [5, 6] culminated in the
O(1)-rounds algorithm of [9] for computing an MST in the unicast congested clique. In fact, [4] prove that we can solve even \( n^{1-\varepsilon} \) many MST instances in \( O(1) \) rounds. These results raise the question regarding the precise dependency between the necessary link bandwidth and the number of rounds.

**Open Problem 1.** Is there an algorithm that computes an MST in just 2 rounds of the unicast congested clique with \( O(\log n) \) bandwidth? Alternatively, is there a tradeoff between the number of rounds and the link bandwidth?

A problem closely related to constructing a spanning tree is testing the connectivity of the graph. Several previous works have considered this problem in the congested clique, see [8, 9, 16]. In the congested clique, graph connectivity has been studied under the assumption that all nodes need to output “yes” if the graph is connected and at least one node outputs “no” otherwise. Under this assumption, the lower bound technique of this paper extends to graph connectivity even for the unicast congested clique. However, it would be interesting to study connectivity under the more realistic assumption that only some node outputs the decision while all other nodes may remain undecided. Algorithms for this setting can truly leverage the power of the unicast congested clique:

**Open Problem 2.** What is the minimum link bandwidth required for graph connectivity in the one-round unicast congested clique, for algorithms where some arbitrary node outputs the decision?

**REFERENCES**

[1] Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. 2012. Analyzing graph structure via linear measurements. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*. SIAM, 459–467.

[2] Baruch Awerbuch, Oded Goldreich, David Peleg, and Ronen Vainish. 1990. A Trade-Off between Information and Communication in Broadcast Protocols. *J. ACM* 37, 2 (1990), 238–256. https://doi.org/10.1145/77600.77618

[3] Andrew Drucker, Fabian Kuhn, and Rotem Oshman. 2014. On the power of the congested clique model. In *ACM Symposium on Principles of Distributed Computing, PODC ’14, Paris, France, July 15-18, 2014*. 367–376.

[4] Mohsen Ghaffari and Krzysztof Nowicki. 2018. Congested Clique Algorithms for the Minimum Cut Problem. In *PODC 2018*. 357–366.

[5] Mohsen Ghaffari and Merav Parter. 2016. MST in log-star rounds of congested clique. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*. 19–28.

[6] James W Hegeman, Gopal Pandurangan, Sriram V Pemmaraju, Vivek B Sardeshmukh, and Michele Squazzato. 2015. Toward optimal bounds in the congested clique: Graph connectivity and MST. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing*. 91–100.

[7] Taisuke Izumi and François Le Gall. 2017. Triangle Finding and Listing in CONGEST Networks. In *PODC 2017*. 381–389.

[8] Tomasz Jurdziński and Krzysztof Nowicki. 2017. Brief announcement: on connectivity in the broadcast congested clique. In *DISC 2017*. Schloss Dagstuhl-Leibniz-Zentrum f. Informatik.

[9] Tomasz Jurdziński and Krzysztof Nowicki. 2018. MST in \( O(1) \) rounds of congested clique. In *SODA 2018*. SIAM, 2620–2632.

[10] Valerie King, Shay Kutten, and Mikkel Thorup. 2015. Construction and Impromptu Repair of an MST in a Distributed Network with \( o(m) \) Communication. In *PODC 2015*. Chryssis Georgiou and Paul G. Spirakis (Eds.). ACM, 71–80.
[11] Christoph Lenzen. 2013. Optimal deterministic routing and sorting on the congested clique. In *PODC 2013*. 42–50.

[12] Zvi Lotker, Elan Pavlov, Boaz Patt-Shamir, and David Peleg. 2003. MST construction in O (log log n) communication rounds. In *Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*. 94–100.

[13] Pedro Montealegre, Sebastian Perez-Salazar, Ivan Rapaport, and Ioan Todinca. 2020. Graph reconstruction in the congested clique. *J. Comput. Syst. Sci.* 113 (2020), 1–17. https://doi.org/10.1016/j.jcss.2020.04.004

[14] Krzysztof Nowicki. 2021. A deterministic algorithm for the MST problem in constant rounds of congested clique. In *STOC 2021*, Samir Khuller and Virginia Vassilevska Williams (Eds.). ACM, 1154–1165.

[15] Shreyas Pai, Gopal Pandurangan, Sriram V. Pemmaraju, and Peter Robinson. 2021. Can We Break Symmetry with o(m) Communication?. In *PODC 2021: ACM Symposium on Principles of Distributed Computing*. ACM, 247–257.

[16] Shreyas Pai and Sriram V Pemmaraju. 2020. Connectivity Lower Bounds in Broadcast Congested Clique. In *40th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*.

[17] Gopal Pandurangan, Peter Robinson, and Michele Scquizzato. 2018. The Distributed Minimum Spanning Tree Problem. *Bull. EATCS* 125 (2018). http://eatcs.org/beatcs/index.php/beatcs/article/view/538

[18] Gopal Pandurangan, Peter Robinson, and Michele Scquizzato. 2021. On the distributed complexity of large-scale graph computations. *ACM Transactions on Parallel Computing (TOPC)* 8, 2 (2021), 1–28.

[19] Peter Robinson. 2021. Being Fast Means Being Chatty: The Local Information Cost of Graph Spanners. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, January 10 - 13, 2021*. SIAM, 2105–2120.

[20] Huacheng Yu. 2021. Tight Distributed Sketching Lower Bound for Connectivity. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*. 1856–1873.