Brief Announcement: On Connectivity in the Broadcast Congested Clique*

Tomasz Jurdziński[1] **and Krzysztof Nowicki**[2]

1 **Institute of Computer Science, University of Wrocław, Poland**
2 **Institute of Computer Science, University of Wrocław, Poland**

Abstract

Recently, very fast deterministic and randomized algorithms have been obtained for connectivity and minimum spanning tree in the unicast congested clique. In contrast, no solution faster than a simple parallel implementation of the Boruvka's algorithm has been known for both problems in the broadcast congested clique. In this announcement, we present the first sub-logarithmic deterministic algorithm for connected components in the broadcast congested clique.

1998 ACM Subject Classification F.1.2 Modes of Computation, F.2.3 Tradeoffs between Complexity Measures, F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases congested clique, broadcast, connected components, bandwidth

Digital Object Identifier 10.4230/LIPIcs.DISC.2017.54

1 Introduction

In the congested clique model, each pair of n nodes of a network is connected by a separate communication link. Communication is synchronous, each node in each round can send message of $O(\log n)$ bits to each other node of the network. The main purpose of such a model is to understand the role of congestion in distributed computation, separately from limitations of locality. In the *unicast* congested clique, a node can send (possibly) different message to each other node of the network. In contrast, in the *broadcast* congested clique, each node can only send a single (the same) message to all other nodes in a round.

Graph problems in the congested clique model are considered under the assumption that the input is an undirected n-node weighted graph $G(V, E, w)$, where each node corresponds to a node of the communication network which initially knows the network size n, its unique ID in $[n]$, the IDs of its neighbors in the input graph and the weights of its incident edges. In the connected components problem, the set of edges inducing connected components of the input graph has to be determined.

The main complexity measure is *round complexity*, equal to the number of rounds in an execution of an algorithm. A natural generalization parametrizes the size (in bits) of messages transmitted in a round, called *bandwidth* and denoted by b. Yet another generalization is that the size of messages in various rounds might be different, but not larger than the bandwidth. Then, *bit complexity* is defined as the sum of sizes of messages in all rounds.

Formal study of the congested clique model was initiated in [4], where a $O(\log\log n)$ round deterministic algorithm for minimum spanning tree (MST), and therefore also for the connected components problem, in the unicast congested clique is presented. The best known randomized solution for MST in the unicast model works in $O(\log^* n)$ rounds [2],

* This work was supported by the Polish National Science Centre grant DEC-2012/07/B/ST6/01534.

31st International Symposium on Distributed Computing (DISC 2017).
Editor: Andréa W. Richa; Article No. 54; pp. 54:1–54:4
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

improving the $O(\log\log\log n)$ bound [3]. If the bandwidth is $b = \sqrt{n}\log n$, one can compute deterministically Connectivity in $O(1)$ of rounds, even in the broadcast congested clique [5]. In an extreme scenario of one-round protocols in the broadcast congested clique, connected components can be computed with public random bits using $\Theta(\log^3 n)$-size messages [1].

2 Connected Components Algorithm

In this section we describe our algorithm for the connected components problem. In the following $[p]$ denotes the set $\{1, 2, \ldots, p\}$. Given a partition $\mathcal{C}$ of a graph $G(V, E)$ into connected components and $v \in V$, C^v denotes the component containing v. We define $\deg_{\mathcal{C}}(v)$ for a vertex v wrt a partition $\mathcal{C}$ as the number of components connected with v, i.e., $\deg_{\mathcal{C}}(v) = |N_{\mathcal{C}}(v)|$, where $N_{\mathcal{C}}(v) = \{C \in \mathcal{C} \,|\, \exists u \in C \text{ such that } (v, u) \in E \text{ and } C \neq C^v\}$. For a component $C \in \mathcal{C}$, we define $\deg_{\mathcal{C}}(C) = \max_{v \in C}\{\deg_{\mathcal{C}}(v)\}$. Given a partition $\mathcal{C}$ of the graph into components, we define the linear ordering $\succ$ of components, where $C \succ C'$ iff $\deg_{\mathcal{C}}(C) > \deg_{\mathcal{C}}(C')$ or $\deg_{\mathcal{C}}(C) = \deg_{\mathcal{C}}(C')$ and $\text{ID}(C) > \text{ID}(C')$. A component C is a *local maximum* if all its neighbors are smaller with respect to the $\succ$ ordering.

The algorithm consists of the main part and the *playoff* (see Alg. 1). The main part is split into *phases*. In a phase, edges connecting currently build connected components are reported. The edges which connect nodes to the components of large degree are preferred. The intended result of a phase is that each component either has a small degree (smaller than s) or it is connected to some "host" of large degree (directly or by a path). As the number of such "hosts" will be relatively small, we obtain significant reduction of the number of components of large degree in each phase. Moreover, we separately deal with components of small degree by allowing them to broadcast all their neighbours in the playoff.

At the beginning of phase 1 of the main part, each node is *active* and it forms a separate component. During an execution of the algorithm, nodes from components of small degree (smaller than s) are *deactivated*. At the beginning of a phase, a partition $\mathcal{C}$ of the graph of active nodes is known. In Round 1 of a phase, each node v determines $N_{\mathcal{C}}(v)$ and announces its degree $\deg_{\mathcal{C}}(v)$. With this information, each node v knows the ordering of components of $\mathcal{C}$ according to $\succ$. Then, each *active* node v (except of members of local maxima) broadcasts its incident edge to the largest active component from $N_{\mathcal{C}}(v)$ according to $\succ$ (Round 2). Next, each node v of each local maximum C checks whether edges connecting C to all components from $N_{\mathcal{C}}(v)$ have been already broadcasted. If it is not the case, an edge connecting v to a new component C' is broadcasted by v (Round 3). Based on broadcasted edges, new components are determined and their degrees are computed (Round 4). Each new component with degree smaller than s is *deactivated* at the end of a phase.

The playoff lasts s rounds in which each node v of each deactivated component broadcasts an edge going to each component connected to v at the moment of deactivation (there are at most s such components for each deactivated node).

The key property of the algorithm is that each active component C of degree $\geq s$ is either connected during a phase to all its neighbors or to a component which is larger than C according to $\succ$. Thus, the number of active components decreases s times in each phase.

▶ **Theorem 1.** *Alg. 1 solves the spanning forest problem in $O(s + \log_s n)$ rounds.*

For $s = \frac{\log n}{\log\log n}$ Algorithm 1 gives the claimed $o(\log n)$ result.

▶ **Corollary 2.** *It is possible to solve the connected components problem in the broadcast congested clique in $O\left(\frac{\log n}{\log\log n}\right)$ rounds.*

Algorithm 1 BroadcastCC(v, s) ▷ s is the threshold between small/large degree

$\mathcal{C} \leftarrow$ the partition into one-node components $\{v_1\}, \ldots, \{v_n\}$
while there are active components **do** ▷ execution at a node v
  Round 1: v broadcasts $\deg_{\mathcal{C}}(v)$
  if $\deg_{\mathcal{C}}(v) > 0$ **then** ▷ $\mathcal{C}$ – the current partition into connected components
    $C_{\max}(v) \leftarrow$ the largest element of $N_{\mathcal{C}}(v)$ wrt the ordering $\succ$
    Round 2:
    if C^v is not a local maximum **then** v broadcast an edge (u, v) such that $u \in C_{\max}$
    Round 3:
    if C^v is a local maximum **then**
      $N_{\text{lost}}(v) \leftarrow \{C \mid C \in N_{\mathcal{C}}(v)$ and no edge connecting C and C^v was broadcasted$\}$
      if $N_{\text{lost}}(v) \neq \emptyset$ **then**
        $u \leftarrow$ a neighbor of v such that $u \in C$ for some $C \in N_{\text{lost}}(v)$
        v broadcasts an edge (u, v)
  v computes the new partition $\mathcal{C}$ into components, using all broadcasted edges
  Round 4: v broadcasts $\deg_{\mathcal{C}}(v)$ ▷ degrees wrt the new components!
  if $\deg_{\mathcal{C}}(C^v) < s$ **then** deactivate v
  Remove deactivated components from the partition $\mathcal{C}$
Playoff (s rounds): deactivated nodes broadcast edges to neighboring components.

Now, assume that the bandwidth is $b = d \log n$. If $s = d$ in Algorithm 1, we get $\log_d n$ phases, each requiring $O(\log n)$ bits per node. Edges from deactivated nodes can be broadcasted during playoff in one round, using $O(d \log n)$ bits bandwidth. This gives $O(\log_d n)$ round algorithm using $O(\log n(d + \frac{\log n}{\log d}))$ bit complexity.

▶ **Corollary 3.** *It is possible to solve the connectivity problem in the broadcast congested clique with bandwidth $d \log n$ in $\log_d n$ rounds and $O(\log n(d + \frac{\log n}{\log d}))$ bit complexity.*

The above corollary gives an improvement over a result from [5], where bit complexity is $O\left(d\frac{\log^2 n}{\log d}\right)$ in $O(\log_d n)$ rounds. Moreover, our algorithm does not use number theoretic techniques as d-pruning or deterministic sparse linear sketches needed in [5].

Conclusions. We have shown the first sub-logarithmic algorithm for connected components in the broadcast congested clique. On the other hand, it is still not known whether MST can be computed in $o(\log n)$ rounds.

References

1 Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Analyzing graph structure via linear measurements. In *Proceedings of SODA 2012*, pages 459–467, 2012. URL: `http://dl.acm.org/citation.cfm?id=2095116.2095156`.

2 Mohsen Ghaffari and Merav Parter. MST in log-star rounds of congested clique. In *Proceedings of PODC 2016*, pages 19–28, 2016. `doi:10.1145/2933057.2933103`.

3 James W. Hegeman, Gopal Pandurangan, Sriram V. Pemmaraju, Vivek B. Sardeshmukh, and Michele Scquizzato. Toward optimal bounds in the congested clique: Graph connectivity and MST. In *Proceedings of PODC 2015*, pages 91–100, 2015. `doi:10.1145/2767386.2767434`.

4 Zvi Lotker, Boaz Patt-Shamir, Elan Pavlov, and David Peleg. Minimum-weight spanning tree construction in $O(\log\log n)$ communication rounds. *SIAM J. Comput.*, 35(1):120–131, 2005. `doi:10.1137/S0097539704441848`.

5 Pedro Montealegre and Ioan Todinca. Brief announcement: Deterministic graph connectivity in the broadcast congested clique. In *Proceedings of PODC 2016*, pages 245–247, 2016. doi:10.1145/2933057.2933066.