# Brief Announcement: Exponential Speed-Up of Local Algorithms using Non-Local Communication

Christoph Lenzen
Computer Engineering and Networks Laboratory
ETH Zurich
Zurich, Switzerland
lenzen@tik.ee.ethz.ch

Roger Wattenhofer
Computer Engineering and Networks Laboratory
ETH Zurich
Zurich, Switzerland
wattenhofer@tik.ee.ethz.ch

## ABSTRACT

We demonstrate how to leverage a system's capability for all-to-all communication to achieve an exponential speed-up of local algorithms despite bandwidth and memory restrictions. More precisely, if a network comprises $n$ nodes with all-to-all bandwidth $n^\varepsilon$ ($\varepsilon > 0$ constant) and nodes know their input and neighborhood with respect to a graph problem instance of polylogarithmic maximum degree, any local algorithm for this problem with running time $r \in \mathcal{O}(\log n)$ and reasonably small states can be simulated within $\mathcal{O}(\log r)$ rounds.

## Categories and Subject Descriptors

F.1.1 [**Theory of Computation**]: Computation by Abstract Devices—*Complexity Measures and Classes*; F.2.3 [**Theory of Computation**]: Computation by Abstract Devices—*Models of Computation*

## General Terms

Algorithms, Theory

## Keywords

local algorithms, simulation, all-to-all communication

## 1. MOTIVATION

When designing distributed algorithms for graph problems, one typically assumes that the input graph coincides with the communication graph, i.e., if—and only if—two nodes are neighbors with respect to the problem formulation, they are able to communicate directly as well. However, many distributed systems present a different communication structure. Supercomputers, for instance, are shipped with predefined interconnection networks and routing mechanisms, peer-to-peer systems build communication networks following various constraints that are not directly related to their designation, and in shared memory systems communication is carried out by means of common memory that can be accessed by all processes.

Frequently, such systems allow for all-to-all communication in the sense that each participant may use a certain bandwidth for communication with arbitrary partners. Consequently, locality of information is not an obstacle any more; yet, it may be inefficient (for lack of bandwidth) or even infeasible (for lack of memory or computational power) to solve problems in a fully centralized manner at a single node.

In this work, we show how to exploit a non-local communication model to achieve an exponential speed-up of local algorithms despite bandwidth and memory restrictions.


*PODC'10,* July 25–28, 2010, Zurich, Switzerland.
ACM 978-1-60558-888-9/10/07.

## 2. MODEL

We assume that all nodes $v \in V$, where $n := |V|$, can send (receive) up to $n^\varepsilon$ bits per round to (from) arbitrary destinations (sources), where $\varepsilon \in (0, 1]$. Nodes have unique identifiers of size $\mathcal{O}(\log n)$. Moreover, nodes may never store more than $n^\varepsilon$ many bits. A (possibly randomized) synchronous local algorithm $\mathcal{A}$ is given that runs on the graph $G = (V, E)$, where initially each node knows its input and its neighborhood in $G$. The maximum degree $\Delta$ of $G$ is in $\text{polylog}\, n$. When executing $\mathcal{A}$, nodes store never more than $n^{(1-\delta)\varepsilon}$ many bits for a constant $\delta \in (0, 1)$ (including random bits, which we assume to be part of the input). If the algorithm runs less than $\delta\varepsilon \log n/(2 \log \Delta)$ rounds, it is sufficient to impose this condition on the inputs only.

These assumptions are motivated by the following considerations. If $G$ had large degrees (e.g. $n^\varepsilon$), the all-to-all communication model would not represent a significant advantage over local communication. Likewise, if states were too large, they could not be exchanged quickly due to the limitations in bandwidth. Note that local algorithms typically maintain only small states (e.g. polylogarithmic) and terminate quickly (e.g. in $\mathcal{O}(\log n)$ rounds) and a large number of practically relevant graph families exhibit polylogarithmic degrees.

## 3. SIMULATING LOCAL ALGORITHMS

THEOREM 3.1. *Suppose the local Algorithm $\mathcal{A}$ terminates within $r$ rounds with probability $p$. Then we can simulate $\mathcal{A}$ within $\mathcal{O}(\log r + r \log\log n/(\varepsilon \log n))$ rounds with probability $p$. If $r \in \mathcal{O}(\varepsilon \log n)$, this simulation takes $\mathcal{O}(\log r)$ time.*

PROOF. Observe that within $d \leq \delta\varepsilon \log n/\log \Delta$ rounds of $\mathcal{A}$, nodes may receive information from at most $\Delta^d \leq n^{\delta\varepsilon}$ many other nodes. We set $d := \min\{2r, \delta\varepsilon \log n/\log \Delta\}$. Each node $v \in V$ collects the topology $T_v$ of (potential) communication partners in $G$ up to distance $2^{\lfloor \log d \rfloor} \in (d/2, d]$. To this end, in each round nodes send the part of the graph $G$ they currently know to all nodes in that part of the graph. Thus, starting with radius 1, in each step the radius of the neighborhood each node knows is doubled, i.e., this needs

to be repeated $\lfloor \log d \rfloor$ times. Such a neighborhood can be encoded with at most $\Delta^d \log n$ many bits: for each node, we need to list the identifiers of its at most $\Delta$ neighbors, which can be encoded by $\mathcal{O}(\log n)$ bits each. Transmitting this information is possible in a single round, since no node ever needs to send or receive more than $\mathcal{O}(\Delta^d \log n) \subseteq \mathcal{O}(n^{\delta\varepsilon} \log n) \subseteq o(n^{\varepsilon})$ bits; for the same reason, nodes do not violate their memory constraints. Thus, after $\lfloor \log d \rfloor$ rounds, each node knows its $2^{\lfloor \log d \rfloor}$-hop neighborhood.

After collecting $T_v$, $v$ sends its input to all of the at most $\Delta^d \leq n^{\delta\varepsilon}$ other nodes in $T_v$. This takes only one round, as nodes have to send and receive at most $n^{\delta\varepsilon} n^{(1-\delta)\varepsilon} = n^{\varepsilon}$ bits. Subsequently, each node locally simulates $d/2 \leq 2^{\lfloor \log d \rfloor}$ rounds of $\mathcal{A}$. If $r \leq \delta\varepsilon \log n/(2 \log \Delta)$, $\mathcal{A}$ terminates with probability $p$ within $d/2 \geq r$ rounds. If the running time is larger than $d/2$, each node $v$ sends its state after $d/2$ rounds of $\mathcal{A}$ to all nodes in $T_v$ and simulates another $d/2$ rounds of the algorithm; we repeat this until all nodes have terminated. We conclude that the total running time is in $\mathcal{O}(\log d + r/d) \subseteq \mathcal{O}(\log r + r \log\log n/(\varepsilon \log n))$ with probability $p$, which in case of $r \in \mathcal{O}(\varepsilon \log n)$ is in $\mathcal{O}(\log r)$. □

# 4. EXAMPLES

For simplicity, we take $\varepsilon$ to be a constant in the following.

COROLLARY 4.1. *A* maximal independent set *is a maximal set containing no neighbors. In the above model, a maximal independent set can be computed in* $\mathcal{O}(\log\log n)$ *rounds with high probability (w.h.p.).*

PROOF. Luby's algorithm ([7], see [8] for a recent variant) is local and takes $\mathcal{O}(\log n)$ time w.h.p. to solve this problem. States are small (in set/not in set/undecided and $\mathcal{O}(\log n)$ random bits for each round), thus Theorem 3.1 states that Luby's algorithm can be simulated in $\mathcal{O}(\log\log n)$ rounds as claimed. □

COROLLARY 4.2. *A* (minimum) dominating set *is a set (of minimum size) such that all nodes are in the set or have a neighbor in the set. In the presented model, a dominating set that is in expectation a factor* $\mathcal{O}(\log \Delta) \subseteq \mathcal{O}(\log\log n)$ *larger than one of minimum size can be found in* $\mathcal{O}(\log\log \Delta) \subseteq \mathcal{O}(\log\log\log n)$ *rounds.*

PROOF. We apply Theorem 3.1 to the distributed minimum dominating set algorithm from [5] (with bounded message size and subsequent randomized rounding) for $k_p = k_d = \log \Delta \in \mathcal{O}(\log\log n)$. □

It is well known that the minimum dominating set problem is NP-complete [2]; even approximations better than factor $c \log \Delta$ are NP-hard [9] (for some constant $c > 0$). Therefore, if we neither permit exponential computation nor P = NP, the approximation ratio of Corollary 4.2 is asymptotically optimal for any running time polynomial in $n$.

COROLLARY 4.3. *A matching is a mutually non-adjacent subset of the edges of a graph. If the edges have weights, the weight of a matching is the sum of the weights of its edges. In the presented model, for any constant* $\delta > 0$ *it is possible to find within* $\mathcal{O}(\log\log n)$ *rounds a matching that has w.h.p. weight at most a factor* $(2 + \delta)$ *smaller than the maximum. In the unweighted case, a* $(1 + \delta)$*-approximation can be achieved with the same running time.*

PROOF. We employ the algorithms from [6], which compute w.h.p. $(2+\delta)$ and $(1+\delta)$-approximations in the weighted and unweighted case, respectively. They run for $\mathcal{O}(\log n)$ rounds and use small messages, i.e., applying Theorem 3.1 yields the claimed result. □

COROLLARY 4.4. *A vertex cover is a subset of the nodes such that for each edge at least one of its endpoints is in the subset. If nodes have weights, the weight of a vertex cover is the sum of the weights of its nodes. In the presented model, it is possible to find a vertex cover that has at most twice the weight of a solution of minimum weight within* $\mathcal{O}(\log\log n)$ *rounds w.h.p.*

PROOF. We apply Theorem 3.1 to the respective algorithm from [4]. □

It is known to be NP-hard to approximate minimum vertex cover within a factor of 1.3606 [1]. Moreover, if the unique games conjecture holds, it is NP-hard to approximate it within any constant factor smaller than 2 [3].

# 5. REFERENCES


[1] I. Dinur and S. Safra. On the Hardness of Approximating Minimum Vertex Cover. *Annals of Mathematics*, 162:2005, 2004.

[2] R. M. Karp. Reducibility Among Combinatorial Problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[3] S. Khot and O. Regev. Vertex Cover Might be Hard to Approximate to within 2-$\epsilon$. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008.

[4] C. Koufogiannakis and N. E. Young. Distributed and Parallel Algorithms for Weighted Vertex Cover and other Covering Problems. In *Proc. of the 28th ACM symposium on Principles of distributed computing (PODC)*, pages 171–179, 2009.

[5] F. Kuhn, T. Moscibroda, and R. Wattenhofer. The Price of Being Near-Sighted. In *Proc. 17th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, January 2006.

[6] Z. Lotker, B. Patt-Shamir, and S. Pettie. Improved Distributed Approximate Matching. In *Proc. 20th annual Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 129–136, 2008.

[7] M. Luby. A Simple Parallel Algorithm for the Maximal Independent Set Problem. *SIAM J. Comput.*, 15(4):1036–1055, 1986.

[8] Y. Métivier, J. M. Robson, N. Saheb Djahromi, and A. Zemmari. An optimal bit complexity randomised distributed MIS algorithm. In *Proc. 16th International Colloquium on Structural Information and Communication Complexity*, pages 1–15, Piran, Slovenia, 2009.

[9] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proc. 29th annual ACM Symposium on Theory of Computing (STOC)*, pages 475–484, New York, NY, USA, 1997.