Brief Announcement: Efficient Computation in Congested Anonymous Dynamic Networks*

Giuseppe A. Di Luna†
g.a.diluna@gmail.com
Sapienza University of Rome
Rome, Italy

Giovanni Viglietta†
viglietta@gmail.com
University of Aizu
Aizuwakamatsu, Japan

ABSTRACT

An *anonymous dynamic network* is a network of indistinguishable processes whose communication links may appear or disappear unpredictably over time. Previous research has shown that deterministically computing an arbitrary function of a multiset of input values given to these processes takes only a linear number of communication rounds (Di Luna–Viglietta, FOCS 2022).

However, fast algorithms for anonymous dynamic networks rely on the construction and transmission of large data structures called *history trees*, whose size is polynomial in the number of processes. This approach is unfeasible if the network is *congested*, and only messages of logarithmic size can be sent through its links. In fact, it is known that certain basic tasks such as all-to-all token dissemination (by means of single-token forwarding) require $\Omega(n^2/\log n)$ rounds in congested networks (Dutta et al., SODA 2013).

In this work, we develop a series of practical and efficient techniques that make it possible to use history trees in congested anonymous dynamic networks. Among other applications, we show how to compute arbitrary functions in such networks in $O(n^3)$ communication rounds, greatly improving upon previous state-of-the-art algorithms for congested networks.

CCS CONCEPTS

• **Theory of computation → Distributed algorithms**; • **Computing methodologies → Distributed algorithms**.

KEYWORDS

anonymous dynamic network, congested network, history tree

ACM Reference Format:
Giuseppe A. Di Luna and Giovanni Viglietta. 2023. Brief Announcement: Efficient Computation in Congested Anonymous Dynamic Networks. In *ACM Symposium on Principles of Distributed Computing (PODC '23), June 19–23, 2023, Orlando, FL, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3583668.3594590

1 INTRODUCTION

In recent years, distributed computing has seen a remarkable increase in research on the algorithmic aspects of networks that constantly change their topology [7, 30, 32]. The study of these *dynamic networks* is motivated by technologies such as wireless sensors networks, software-defined networks, and networks of smart devices. Typically, the distributed system consists of n processes that communicate with each other in synchronous rounds. At each round, the network topology is rearranged arbitrarily, and communication links appear or disappear unpredictably.

There are efficient algorithms for various tasks that work under the assumption that processes have *unique IDs* [6, 27–29, 32, 34]. However, unique IDs may not be available due to operational limitations [34] or to protect user privacy; for instance, assigning temporary random IDs to users of COVID-19 tracking apps was not sufficient to eliminate privacy concerns [40]. Systems where processes are indistinguishable are called *anonymous*.

It is known that many fundamental problems for anonymous networks cannot be solved without additional "symmetry-breaking" assumptions: A notable example is the *Counting problem*, i.e., determining the total number of processes n. The most typical symmetry-breaking choice is assuming the presence of a single distinguished process in the system, called *leader* [1–4, 14, 19, 21, 23, 31, 39, 42]. A leader process may represent a base station in a sensor network, a super-node in a P2P network, etc.

Almost all previous research on anonymous dynamic networks pertains to the *LOCAL* model, which imposes no limit on the size of messages exchanged by processes [11, 15, 21–24, 26, 34, 36]. Unfortunately, in most mobile networks, sending small-size messages is not only desirable but also a necessity; for example, in sensor networks, short communication times significantly increase battery life. A more realistic model is *CONGEST*, which assumes the network to be "congested" and limits the size of every message to $O(\log n)$ bits, where n is the number of processes [38].[1]

A recent innovation in the study of anonymous dynamic networks with leaders was the introduction of *history trees* in [15], which led to an optimal deterministic solution to the *Generalized Counting problem*[2] in the LOCAL model. This problem is "complete" for a large class of functions called *multi-aggregate functions*, which in turn are the only computable functions in this model. The theory of history trees was extended in [17] to *leaderless* networks,

*The full version of this paper can be found in [16].

†Both authors contributed equally to this research. The second author was partially supported by JSPS KAKENHI grant number 23K10985.

PODC '23, June 19–23, 2023, Orlando, FL, USA

ACM ISBN 979-8-4007-0121-4/23/06…$15.00
https://doi.org/10.1145/3583668.3594590

[1]This $O(\log n)$ limit on message sizes does not imply that the processes have a-priori information about n. The size limit is not explicitly given to the processes, and it is up to the algorithm to automatically prevent larger messages from being sent.
[2]In the Generalized Counting problem, each process starts with a certain input, and the goal is to determine how many processes have each input.

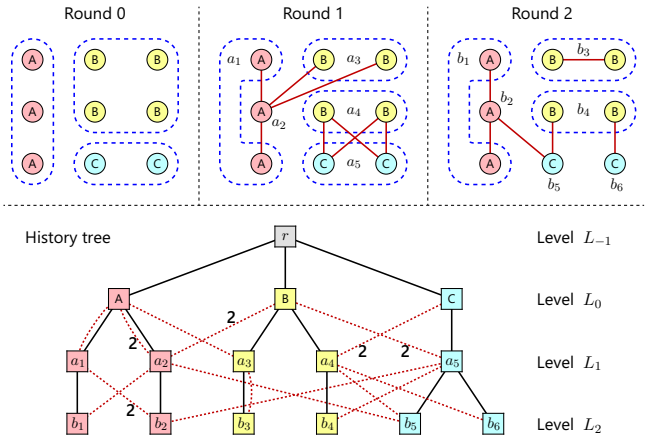

Figure 1: The first rounds of a dynamic network with $n = 9$ processes and the corresponding levels of the history tree.

providing optimal algorithms for the *Frequency problem*:[3] This problem is complete for the class of *frequency-based* multi-aggregate functions, which are the only computable functions in leaderless systems. Thus, the computational landscape for the LOCAL model is fully understood, and optimal linear-time algorithms are known for anonymous dynamic systems with and without leaders.[4] No previous research exists on the CONGEST model, except for a recent preprint that gives a Counting algorithm in $\widetilde{O}(n^{5+\epsilon})$ rounds for networks with leaders [25].

2 CONTRIBUTIONS AND TECHNIQUES

We provide a state-of-the-art general algorithmic technique for anonymous dynamic networks in the CONGEST model, with and without leaders. The resulting algorithms run in $O(n^3)$ rounds, where n is the (initially unknown) total number of processes.

Technical background. We model a dynamic network as an infinite sequence $\mathcal{G} = (G_t)_{t\geq 1}$, where $G_t = (V, E_t)$ is an undirected multigraph whose vertex set $V = \{p_1, p_2, \ldots, p_n\}$ is a system of n *anonymous processes* and E_t is a multiset of edges representing *links* between processes. Each process has an initial input and an internal state; at every round $t \geq 1$, all processes send each other messages through the links of G_t and update their states accordingly.

Informally, a *history tree* is a way of representing the history of a dynamic network in the form of an infinite tree. Each node in a history tree represents a set of anonymous processes that are "indistinguishable" at a certain round, where two processes become "distinguishable" as soon as they receive different sets of messages.

The theory of history trees developed in [15, 17] yields optimal general algorithms for anonymous dynamic networks with and without leaders in the LOCAL model. The idea is that processes can work together to incrementally construct the history tree by repeatedly exchanging and merging together their respective "views" of it. Once they have a sufficiently large portion of the history tree (specifically, $3n$ "levels"), each process can locally analyze its structure and perform arbitrary computations on the multiset of input values originally assigned to the processes.

Challenges. Unfortunately, implementing the above idea requires sending messages containing entire "views" of the history tree. The size of a view is $\Theta(n^3 \log n)$ bits in the worst case, and is therefore unsuitable for the CONGEST model [15]. There is a major difficulty in dealing with this problem deterministically, which stems from the lack of unique IDs combined with the dynamic nature of the network. Although a process can break down a large message into small pieces to be sent in different rounds, it is not clear how the original message can then be reconstructed, because the pieces carry no IDs and a process' neighbors may change at every round. This may result in messages from different processes being mixed up and non-existent messages being reconstructed.

Methodology. Our main contribution is a general method that allows history trees to be transmitted reliably and deterministically between anonymous processes in a dynamic network in the CONGEST model with a leader. To overcome the fundamental issues outlined above, we devised a basic protocol combining different techniques, as well as a number of extensions, including leaderless ones. Although the techniques we introduce are self-contained and do not rely on the results of [15], they effectively allow us to reduce the CONGEST model to the LOCAL one, making it possible to apply the Counting algorithm in [15] as a "black box".

Firstly, we developed a method for dynamically assigning temporary (non-unique) IDs to process; this method is an essential part of the history tree transmission algorithm. In fact, the nodes of our history trees are now augmented with IDs, meaning that each node represents the set of processes with a certain ID. When processes with equal IDs get disambiguated, they get new IDs.

The transmission of history trees occurs level by level, one edge at a time. Since the total ordering between IDs induces a total ordering on the history tree's edges, the processes can collectively transmit sets of edges with a method reminiscent of *Token Dissemination* [28].

Essentially, all processes participate in a series of broadcasts; the goal of each broadcast is to transmit the next "highest-value" edge to the whole network. The problem is that no upper bound on the *dynamic diameter* of the network is known, and therefore there is no way of knowing how many rounds it may take for all processes to receive the edge being broadcast.

We adopt a self-stabilizing approach to ensure that all messages are successfully broadcast. We give a communication protocol based on acknowledgments by the leader, where failure to broadcast a message alerts at least one process. Alerted processes start broadcasting error messages, which eventually cause a reset of the broadcast that caused the error. A mechanism that dynamically estimates the diameter of the network guarantees that no more than $O(\log n)$ resets are performed.

Moreover, in order to achieve a cubic running time, we do not construct the history tree of the actual network, but a more compact history tree corresponding to a *virtual network*. The virtual network is carefully derived from the real one in such a way as to amortize the number of edges in the resulting history tree and further reduce the final worst-case running time by a factor of n.

[3] In the Frequency problem, the goal is to determine the percentage of processes that have each input.

[4] By the word "optimal" we mean "asymptotically worst-case optimal as a function of the total number of processes n".

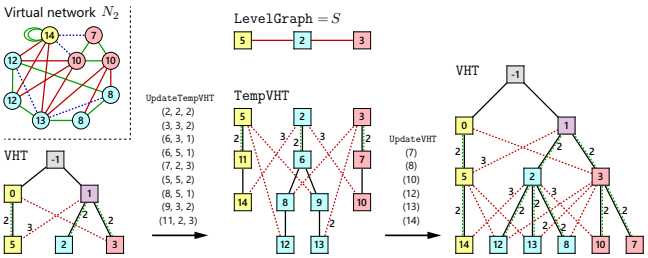

Figure 2: The virtual network N_2 is constructed by removing the blue edges from the real network G_2 and adding the green edges. The corresponding level L_2 of the virtual history tree (VHT) is constructed one red edge at a time, via several broadcast phases. The addition of a red edge to the VHT also causes some processes to update their temporary IDs.

3 ALGORITHM OUTLINE

Our main algorithm assumes the presence of a unique leader in the network, but it can also be generalized to leaderless networks. Each process has some private memory which is used to permanently store information in the form of internal variables.

Virtual history tree (VHT). The overall goal of the algorithm is for the processes to implicitly agree on the first $3n$ levels of a particular history tree, called *virtual history tree (VHT)*, which corresponds to a dynamic network $\mathcal{N}$ of n processes. Once the construction of a new level of the VHT is complete, the leader locally runs the Counting algorithm from [15] on the VHT. If this algorithm successfully returns a number (as opposed to "Unknown"), the leader outputs it; otherwise, the construction of a new level of the VHT is initiated.

Virtual network ($\mathcal{N}$). The dynamic network $\mathcal{N} = (N_1, N_2, \dots)$ represented by the VHT is in fact a *virtual network*, in the sense that none of the multigraphs N_t necessarily coincides with any multigraph of links actually occurring in the real communication network $\mathcal{G} = (G_1, G_2, \dots)$. However, each N_t is obtained by carefully adding and removing links from some G_{i_t} (see Figure 2). This manipulation has the purpose of reducing the size of the resulting VHT by a factor of n.

Temporary IDs. To cope with the fact that processes are anonymous and information can only be sent in small chunks of size $O(\log n)$, each process has a *temporary ID* stored in a local variable. Each node v in the VHT also has an ID, indicating that v represents all processes having that ID. Thus, a *red-edge triplet* of the form $(\texttt{ID1}, \texttt{ID2}, \texttt{Mult})$ can be used to unambiguously represent a red edge of multiplicity `Mult` between the nodes of the VHT whose IDs are `ID1` and `ID2`. Since a red-edge triplet has size $O(\log n)$, it can be included in a single message. Note that the local ID variable of each process may be modified over time as the VHT acquires more nodes.

Broadcast phases. The construction of the VHT is carried out level by level, and is done through several *broadcast phases*, which are indirectly coordinated by the leader. At first, each process knows the red edges incident to its corresponding node of the VHT. Then, ideally every two broadcast phases, the whole network learns a new red edge of the VHT. The broadcast phases continue until all processes know all red edges in the level.

Estimating the diameter. In order to guarantee the success of a broadcast phase, all processes must keep sending each other information for a certain number of rounds, which depends on the *dynamic diameter* of the network, and is $n-1$ in the worst case [28]. Since the processes have no information about the network, they can only estimate the dynamic diameter. The current estimate is stored by each process in a local variable, whose value is initially 1 and is doubled every time the processes detect a faulty broadcast.

Error phases. Detecting broadcasting errors and consistently reacting to them is by no means a trivial task. Our broadcasting technique ensures that, if some red-edge triplet fails to be broadcast to the entire network and does not become part of the local VHT of all processes, at least one process becomes aware of this fact. Such a process enters an *error phase*, sending a high-priority message at every round containing the level number at which the error occurred. Error messages supersede the regular ones and eventually reach the leader.

Reset phases. When the leader finally receives an error message, it initiates a *reset phase*, whose goal is to force the whole network to restore a previous state of the VHT and continue from there. This is achieved by broadcasting a high-priority reset message. Since the error must have occurred because the estimated diameter was too small, its value is doubled at the end of the reset phase.

Note that there is no obvious way for the leader to tell if any level of the VHT is actually missing some parts. Indeed, at any point in time, there may be processes in an error phase unbeknownst to the leader. One of the challenges of our method is to ensure that the leader will not terminate with an incorrect guess on n due to the VHT being incomplete. The interested reader may refer to the full version of this paper for extra details [16].

4 CONCLUDING REMARKS

We extended the theory of history trees by introducing the tools necessary for the distributed construction and transmission of history trees in the CONGEST model. This results in a new state of the art for computation in anonymous dynamic congested networks.

Our history tree construction technique leads to general algorithms whose running time is cubic in the size of the network. Specifically, if there is a unique leader in the network, the algorithm outlined in Section 3 runs in $O(n^3)$ communication rounds. If there is no leader but an upper bound D on the dynamic diameter is known, the running time becomes $O(Dn^2)$. Note that, without a leader, some knowledge about the network is required in order to solve the Counting problem and other basic problems [17].

An immediate open question is whether these running times can be reduced. Since our algorithm broadcasts information using a token-forwarding approach, and by virtue of the $\Omega(n^2/\log n)$ lower bound of [18], we believe that it would be unlikely to achieve a better running time without a radical change in the technique used.

Understanding whether the Counting problem has as a superlinear lower bound in congested networks is of special interest, as it would mark an inherent computational difference between anonymous dynamic networks in the LOCAL and CONGEST models.

REFERENCES

[1] D. Angluin, J. Aspnes, and D. Eisenstat. 2008. Fast Computation by Population Protocols with a Leader. *Distributed Computing* 21, 3 (2008), 61–75.

[2] J. Aspnes, J. Beauquier, J. Burman, and D. Sohier. 2016. Time and Space Optimal Counting in Population Protocols. In *Proceedings of the 20th International Conference on Principles of Distributed Systems (OPODIS '16)*. 13:1–13:17.

[3] J. Beauquier, J. Burman, S. Clavière, and D. Sohier. 2015. Space-Optimal Counting in Population Protocols. In *Proceedings of the 29th International Symposium on Distributed Computing (DISC '15)*. 631–646.

[4] J. Beauquier, J. Burman, and S. Kutten. 2011. A Self-stabilizing Transformer for Population Protocols with Covering. *Theoretical Computer Science* 412, 33 (2011), 4247–4259.

[5] D. P. Bertsekas and J. N. Tsitsiklis. 1989. *Parallel and Distributed Computation: Numerical Methods.* Prentice-Hall, Inc., USA.

[6] A. Casteigts, F. Flocchini, B. Mans, and N. Santoro. 2015. Shortest, Fastest, and Foremost Broadcast in Dynamic Networks. *International Journal of Foundations of Computer Science* 26, 4 (2015), 499–522.

[7] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. 2012. Time-Varying Graphs and Dynamic Networks. *International Journal of Parallel, Emergent and Distributed Systems* 27, 5 (2012), 387–408.

[8] B. Charron-Bost and P. Lambein-Monette. 2018. Randomization and Quantization for Average Consensus. In *Proceedings of the 57th IEEE Conference on Decision and Control (CDC '18)*. 3716–3721.

[9] B. Charron-Bost and P. Lambein-Monette. 2022. Computing Outside the Box: Average Consensus over Dynamic Networks. In *Proceedings of the 1st Symposium on Algorithmic Foundations of Dynamic Networks (SAND '22)*. 10:1–10:16.

[10] B. Chazelle. 2011. The Total s-Energy of a Multiagent System. *SIAM Journal on Control and Optimization* 49, 4 (2011), 1680–1706.

[11] G. A. Di Luna and G. Baldoni. 2015. Brief Announcement: Investigating the Cost of Anonymity on Dynamic Networks. In *Proceedings of the 34th ACM Symposium on Principles of Distributed Computing (PODC '15)*. 339–341.

[12] G. A. Di Luna, R. Baldoni, S. Bonomi, and I. Chatzigiannakis. 2014. Counting in Anonymous Dynamic Networks Under Worst-Case Adversary. In *Proceedings of the 34th IEEE International Conference on Distributed Computing Systems (ICDCS '14)*. 338–347.

[13] G. A. Di Luna, S. Bonomi, I. Chatzigiannakis, and R. Baldoni. 2013. Counting in Anonymous Dynamic Networks: An Experimental Perspective. In *Proceedings of the 9th International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics (ALGOSENSORS '13)*. 139–154.

[14] G. A. Di Luna, P. Flocchini, T. Izumi, T. Izumi, N. Santoro, and G. Viglietta. 2019. Population Protocols with Faulty Interactions: The Impact of a Leader. *Theoretical Computer Science* 754 (2019), 35–49.

[15] G. A. Di Luna and G. Viglietta. 2022. Computing in Anonymous Dynamic Networks Is Linear. In *Proceedings of the 63rd IEEE Symposium on Foundations of Computer Science (FOCS '22)*. 1122–1133.

[16] G. A. Di Luna and G. Viglietta. 2023. Efficient Computation in Congested Anonymous Dynamic Networks. *arXiv:2301.07849 [cs.DC]* (2023). Full version of this paper.

[17] G. A. Di Luna and G. Viglietta. 2023. Leaderless and Multi-Leader Computation in Disconnected Anonymous Dynamic Networks. *arXiv:2207.08061 [cs.DC]* (2023).

[18] C. Dutta, G. Pandurangan, R. Rajaraman, Z. Sun, and E. Viola. 2013. On the Complexity of Information Spreading in Dynamic Networks. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '13)*. 717–736.

[19] P. Fraigniaud, A. Pelc, D. Peleg, and S. Pérennes. 2000. Assigning Labels in Unknown Anonymous Networks. In *Proceedings of the 19th ACM Symposium on Principles of Distributed Computing (PODC '00)*. 101–111.

[20] B. Haeupler and F. Kuhn. 2012. Lower Bounds on Information Dissemination in Dynamic Networks. In *Proceedings of the 26th International Symposium on Distributed Computing (DISC '12)*. 166–180.

[21] D. R. Kowalski and M. A. Mosteiro. 2018. Polynomial Counting in Anonymous Dynamic Networks with Applications to Anonymous Dynamic Algebraic Computations. In *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP '18)*. 156:1–156:14.

[22] D. R. Kowalski and M. A. Mosteiro. 2019. Polynomial Anonymous Dynamic Distributed Computing Without a Unique Leader. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP '19)*. 147:1–147:15.

[23] D. R. Kowalski and M. A. Mosteiro. 2020. Polynomial Counting in Anonymous Dynamic Networks with Applications to Anonymous Dynamic Algebraic Computations. *J. ACM* 67, 2 (2020), 11:1–11:17.

[24] D. R. Kowalski and M. A. Mosteiro. 2021. Supervised Average Consensus in Anonymous Dynamic Networks. In *Proceedings of the 33rd ACM Symposium on Parallelism in Algorithms and Architectures (SPAA '21)*. 307–317.

[25] D. R. Kowalski and M. A. Mosteiro. 2022. Efficient Distributed Computations in Anonymous Dynamic Congested Systems with Opportunistic Connectivity. *arXiv:2202.07167 [cs.DC]* (2022).

[26] D. R. Kowalski and M. A. Mosteiro. 2022. Polynomial Anonymous Dynamic Distributed Computing Without a Unique Leader. *J. Comput. System Sci.* 123 (2022), 37–63.

[27] F. Kuhn, T. Locher, and R. Oshman. 2011. Gradient Clock Synchronization in Dynamic Networks. *Theory of Computing Systems* 49, 4 (2011), 781–816.

[28] F. Kuhn, N. Lynch, and R. Oshman. 2010. Distributed Computation in Dynamic Networks. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC '10)*. 513–522.

[29] F. Kuhn, Y. Moses, and R. Oshman. 2011. Coordinated Consensus in Dynamic Networks. In *Proceedings of the 30th ACM Symposium on Principles of Distributed Computing (PODC '11)*. 1–10.

[30] F. Kuhn and R. Oshman. 2011. Dynamic Networks: Models and Algorithms. *SIGACT News* 42, 1 (2011), 82–96.

[31] O. Michail, I. Chatzigiannakis, and P. G. Spirakis. 2013. Naming and Counting in Anonymous Unknown Dynamic Networks. In *Proceedings of the 15th International Symposium on Stabilizing, Safety, and Security of Distributed Systems (SSS '13)*. 281–295.

[32] O. Michail and P. G. Spirakis. 2018. Elements of the Theory of Dynamic Networks. *Commun. ACM* 61, 2 (2018), 72.

[33] A. Nedić, A. Olshevsky, A. E. Ozdaglar, and J. N. Tsitsiklis. 2009. On Distributed Averaging Algorithms and Quantization Effects. *IEEE Trans. Automat. Control* 54, 11 (2009), 2506–2517.

[34] R. O'Dell and R. Wattenhofer. 2005. Information Dissemination in Highly Dynamic Graphs. In *Proceedings of the 5th Joint Workshop on Foundations of Mobile Computing (DIALM-POMC '05)*. 104–110.

[35] A. Olshevsky. 2017. Linear Time Average Consensus and Distributed Optimization on Fixed Graphs. *SIAM Journal on Control and Optimization* 55, 6 (2017), 3990–4014.

[36] A. Olshevsky and J. N. Tsitsiklis. 2009. Convergence Speed in Distributed Consensus and Averaging. *SIAM Journal on Control and Optimization* 48, 1 (2009), 33–55.

[37] A. Olshevsky and J. N. Tsitsiklis. 2011. A Lower Bound for Distributed Averaging Algorithms on the Line Graph. *IEEE Trans. Automat. Control* 56, 11 (2011), 2694–2698.

[38] D. Peleg. 2000. *Distributed Computing: A Locality-Sensitive Approach.* Society for Industrial and Applied Mathematics, USA.

[39] N. Sakamoto. 1999. Comparison of Initial Conditions for Distributed Algorithms on Anonymous Networks. In *Proceedings of the 18th ACM Symposium on Principles of Distributed Computing (PODC '99)*. 173–179.

[40] T. Sharma and M. Bashir. 2020. Use of Apps in the COVID-19 Response and the Loss of Privacy Protection. *Nature Medicine* 26, 8 (2020), 1165–1167.

[41] J. N. Tsitsiklis. 1984. *Problems in Decentralized Decision Making and Computation.* Ph. D. Dissertation. Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science.

[42] M. Yamashita and T. Kameda. 1996. Computing on Anonymous Networks. I. Characterizing the Solvable Cases. *IEEE Transactions on Parallel and Distributed Systems* 7, 1 (1996), 69–89.

[43] Y. Yuan, G.-B. Stan, L. Shi, M. Barahona, and J. Goncalves. 2013. Decentralised Minimum-Time Consensus. *Automatica* 49, 5 (2013), 1227–1235.