Brief Announcement: Fast Approximate Counting and Leader Election in Populations*

Othon Michail[1], Paul G. Spirakis[1,2], and Michail Theofilatos[1]

[1] Department of Computer Science, University of Liverpool, UK
[2] Computer Engineering and Informatics Department, University of Patras, Greece
Email: {Othon.Michail, P.Spirakis, Michail.Theofilatos}@liverpool.ac.uk

Keywords: population protocol · epidemic · leader election · counting · approximate counting · polylogarithmic time protocol.

1 Introduction

Population protocols [2] are networks that consist of very weak computational entities (also called *nodes* or *agents*), regarding their individual capabilities and it has been shown that are able to perform complex computational tasks when they work collectively. *Leader Election* is the process of designating a single agent as the coordinator of some task distributed among several nodes. The nodes communicate among themselves in order to decide which of them will get into the *leader* state, starting from the same initial state q. An algorithm A solves the leader election problem if eventually the states of agents are divided into *leader* and *follower*, a unique leader remains elected and a follower can never become a leader. A randomized algorithm R solves the leader election problem if eventually only one leader remains in the system w.h.p.. *Counting* is the problem where nodes must determine the size n of the population. We call *Approximate Counting* the problem in which nodes must determine an estimation $\hat{n}$ of the population size, where $\frac{\hat{n}}{a} < n < \hat{n}$. We call a the estimation parameter. Consider the setting in which an agent is in an initial state a, the rest $n-1$ agents are in state b and the only existing transition is $(a,b) \to (a,a)$. This is the *one-way epidemic* process and it can be shown that the expected time to convergence under the uniform random scheduler is $\Theta(n \log n)$ (e.g., [3]), thus *parallel time* $\Theta(\log n)$.

2 Related Work

The framework of population protocols was first introduced by Angluin et al. [2] in order to model the interactions in networks between small resource-limited mobile agents. There are many solutions to the problem of leader election, such as in networks with nodes having distinct labels or anonymous networks [1,5,4]. In a recent work, Gasieniec and Stachowiak [5] designed a space optimal ($O(\log \log n)$ states) leader election protocol, which stabilizes in $O(\log^2 n)$ parallel time. They

* All authors were supported by the EEE/CS initiative NeST. The last author was also supported by the Leverhulme Research Centre for Functional Materials Design.

use the concept of phase clocks (introduced in [3] for population protocols), which is a synchronization and coordination tool in distributed computing. Regarding the counting problem, in a recent work, Michail [6] proposed a terminating protocol in which a pre-elected leader equipped with two n-counters computes an approximate count between $n/2$ and n in $O(n \log n)$ parallel time w.h.p..

3 Contribution

In this work we employ the use of simple epidemics in order to provide efficient solutions to approximate counting and also to leader election in populations. Our model is that of population protocols. Our goal for both problems is to get polylogarithmic parallel time and to use small memory per agent. *(a)* We start by providing a protocol which provides an upper bound $\hat{n}$ of the size n of the population, where $\hat{n}$ is at most n^a for some $a > 1$. This protocol assumes the existence of a unique leader in the population. The runtime of the protocol until stabilization is $\Theta(\log n)$ parallel time. Each node except the unique leader uses only a constant number of states. However, the leader is required to use $\Theta(\log^2 n)$ states. *(b)* We then look into the problem of electing a leader. We assume an approximate knowledge of the size of the population and provide a protocol (parameterized by the size m of a counter for drawing local random numbers) that elects a unique leader w.h.p. in $O(\frac{\log^2 n}{\log m})$ parallel time, with number of states $O(\max\{m, \log n\})$ per node. By adjusting the parameter m between a constant and n, we obtain a leader election protocol whose time and space can be smoothly traded off between $O(\log^2 n)$ to $O(\log n)$ parallel time and $O(\log n)$ to $O(n)$ states.

4 The model

In this work, the system consists of a population V of n distributed and anonymous (i.e., do not have unique IDs) agents, that are capable to perform local computations. Each of them is executing as a deterministic state machine from a finite set of states Q according to a transition function $\delta : Q \times Q \to Q \times Q$. Their interaction is based on the probabilistic (uniform random) scheduler, which picks in every discrete step a random edge from the complete graph G on n vertices. When two agents interact, they mutually access their local states, updating them according to the transition function δ. The transition function is a part of the population protocol which all nodes store and execute locally. *The time is measured as the number of steps until stabilization, divided by n (parallel time).*

5 Fast Counting with a unique leader

Our probabilistic algorithm for solving the approximate counting problem requires a unique leader who is responsible to give an estimation on the number of nodes. There is initially a unique leader l and all other nodes are in state q. The leader l stores two counters in its local memory, initially both set to 0, and after the first interaction it starts an epidemic by turning a q node into an a node. Whenever a q node interacts with an a node, its state becomes a. Whenever the leader l interacts with a q node, the value of the counter c_q is increased by one

and whenever l interacts with an a node, c_a is increased by one. The termination condition is $c_q = c_a$ and then the leader holds a constant-factor approximation of $\log n$. Chernoff bounds then imply that repeating this protocol a constant number of times suffices to obtain $n/2 \leq n_e \leq 2n$ w.h.p..

Analysis.

THEOREM 1. *Our Approximate Counting protocol obtains a constant-factor approximation of* $\log n$ *in* $O(\log n)$ *parallel time w.h.p..*

PROOF. We divide the process into two phases, with the first phase starting when the unique leader initiates the spreading of an epidemic, and the second phase starting when half of the agents become infected. During the first phase, c_q reaches $O(\log n)$, while c_a is increased by a small constant number w.h.p.. This means that our protocol does not terminate w.h.p. until more than half of the population has been infected. During the second phase, when the infected agents are in the majority, c_q is increased by a small constant number, while c_a eventually catches up the first counter. The termination condition ($c_q = c_a$) is satisfied and the leader obtains a constant-factor approximation of $\log n$. Finally, our protocol terminates after $\Theta(\log n)$ parallel time w.h.p.. After half of the population has been infected, it holds that $|c_q - c_a| = \Theta(\log n)$. When the a nodes are in the majority, this difference reaches zero after $\Theta(\log n)$ leader interactions. Thus, the total parallel time to termination is $\Theta(\log n)$.

6 Leader Election with approximate knowledge of n

We assume that the nodes know *an upper bound on the population size* n^b*, where* n *is the number of nodes and* b *is any big constant number.* All nodes store three variables; the round e, a random number r and a counter c and they are able to compute random numbers within a predefined range $[1, m]$ (m is *the maximum number that the nodes can generate*). We define two types of states; the leaders (l) and the followers (f). Initially, all nodes are in state l, indicating that they are all potential leaders. The protocol operates in rounds and in every round, the leaders compete with each other trying to survive (i.e., do not become followers). During the first interaction of two l nodes, one of them becomes follower, a random number in $[1, m]$ is being generated, the leader enters the first round and the follower copies the tuple (r, e) from the leader to its local memory. The followers are only being used for information spreading purposes among the potential leaders and they cannot become leaders again.

Information spreading. All leaders try to spread their tuple (r, e) throughout the population, but w.h.p. all of them except one eventually become followers. We say that a node x wins during an interaction with node y if: (a) $e_x > e_y$ or (b) if $(e_x = e_y)$, $r_x > r_y$. One or more leaders L are in the *dominant state* if their tuple (r_1, e_1) wins every other tuple in the population. Then, the tuple (r_1, e_1) is being spread as an epidemic throughout the population, independently of the other leaders' tuples (all leaders or followers with the tuple (r_1, e_1) always win their competitors). We also call leaders L the *dominant leaders.*

Transition to next round. After the first interaction, a leader l enters the first round. As long as a leader survives (i.e., does not become a follower), in every interaction it increases it's counter c by one. When c reaches $b \log n$, where n^b is the upper bound on n, it resets it and round e is increased by one. Finally, the followers can never increase their round or generate random numbers.

Stabilization. Our protocol stabilizes, as the whole population will eventually reach in a final configuration of states. To achieve this, when the round of a leader l reaches $\lceil \frac{2b \log n - \log(b \log^2 n)}{\log m} \rceil$, l stops increasing its round e, unless it interacts with another leader. This rule guarantees the stabilization of our protocol.

Analysis. The protocol proceeds by monotonously reducing the set of possible leaders, until only one candidate for a leader remains. There are initially $k_0 = n$ leaders in the population (round $e = 0$) and between successive rounds, the number of the dominant leaders is given by $k_e = \frac{n}{m^e}$.

THEOREM 2. *Our Leader Election protocol elects a unique leader in* $O(\frac{\log^2 n}{\log m})$ *parallel time w.h.p..*

PROOF. During a round e, the dominant tuple spreads throughout the population in $\Theta(\log n)$ parallel time. No leader can enter to the next round if their epidemic has not been spread throughout the whole population before, thus, for $m = b \log n$ the overall parallel time is $O(\frac{\log^2 n}{\log \log n})$. Finally, during an execution of the protocol, at least one leader will always exist in the population (i.e., a unique leader can never become follower) and a follower can never become leader again. The rule which says that leaders stop increasing their rounds if $e >= \frac{2b \log n - \log (b \log^2 n)}{\log m}$, unless they interact with another leader, implies that the population stabilizes in $O(\frac{\log^2 n}{\log m})$ parallel time w.h.p. and when this happens, there will exist only one leader in the population and eventually, our protocol always elects a unique leader.

References

1. Alistarh, D., Gelashvili, R.: Polylogarithmic-time leader election in population protocols. In: 42nd International Colloquium on Automata, Languages, and Programming (ICALP). Lecture Notes in Computer Science, vol. 9135, pp. 479 – 491. Springer, Berlin, Heidelberg (2015)
2. Angluin, D., Aspnes, J., Diamadi, Z., Fischer, M.J., Peralta, R.: Computation in networks of passively mobile finite-state sensors. Distributed Computing **18**(4), 235–253 (2006)
3. Angluin, D., Aspnes, J., Eisenstat, D.: Fast computation by population protocols with a leader. Distributed Computing **21**(3), 183–199 (2008)
4. Fischer, M., Jiang, H.: Self-stabilizing leader election in networks of finite-state anonymous agents. OPODIS 2006: International Conference on Principles of Distributed Systems **vol 4305** (2006)
5. Gasieniec, L., Stachowiak, G.: Fast space optimal leader election in population protocols. In: SODA 2018: ACM-SIAM Symposium on Discrete Algorithms. pp. 265–266 (2018)
6. Michail, O.: Terminating distributed construction of shapes and patterns in a fair solution of automata. In: Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing. pp. 37–46 (2015)