Quantized Distributed Load Balancing with Capacity Constraints

Evan Gravelle Sonia Martínez

Abstract— **Current research in the field of distributed consensus algorithms fails to adequately address physical limitations of real systems. This paper proposes a new algorithm for quantized distributed load balancing over a network of agents subject to upper-limit constraints. More precisely, loads are integer values, and nodes are constrained to remain under maximum load capacities at all times. Convergence to a set of desired states is proven for all connected graphs, any feasible initial load distribution, and separation and connectivity conditions on nodes with small maximum capacities. Simulations illustrate our results.**

I. Introduction

Motivation. A significant problem in distribution and service networks consists of splitting tasks among a group of providers or nodes which can offer certain network commodities to users. Examples include task distribution among a set of processors or robots, energy provision by a set of generators, or traffic control by a group of network managers. The nodes in the network may be subject to capacity constraints, which can restrict the amount of load they can handle; e.g. a finite memory bank in a processor or a maximum generating capacity of a generator. Another restriction is the indivisibility of jobs that need be completed by nodes in the network. Motivated by this, we design and analyze a new distributed algorithm that can achieve load balancing while satisfying capacity and integer load constraints. One advantage of this algorithm is that it can be stopped at any time while guaranteeing the constraints hold and leading to an approximately balanced configuration.

Literature review. The problem of distributed consensus has received considerable attention in the past. It has roots in parallel computation [2] and has been addressed in [3], [9], [13], and [12]. Load balancing is a constrained version of the consensus problem, and there exists many gossip-based algorithms for distributed load balancing over different types of graphs with various constraints and objectives [5], [6]. More recently there has been a focus on quantized consensus to address the problems of finite capacity of communication channels and finite precision in computation. An example of this work studies distributed gossip algorithms which converge to an integer approximation of the average of the initial values and satisfies the integer constraint at all times [10]. An efficient method of quantization using symbols instead of numbers as exchanged data in solving the distributed average consensus problem is seen in [4]. Dithered forms of quantization are also used [1]. An algorithm solving the distributed quantized consensus problem over heterogeneous networks which also minimizes time given different processor speeds is found in [7].

The rate of convergence of gossip-based distributed quantized consensus algorithms is studied in [8], [**?**] while non-gossip based algorithms are studied in [14], [11]. None of these works considers constraints on load capacity for each node, and most focus on gossip-based algorithms which are easy to analyze but do not converge quickly.

Statement of contributions. In this paper we design a new distributed algorithm to balance quantized loads among nodes with maximum load capacities. A node filled to capacity behaves differently from a node that is below its capacity, and this distinction is crucial for performance. A major difference between our algorithm and standard quantized consensus algorithms is that the nodes which reach their maximum capacity can make passes that unbalance the network, complicating the analysis. We prove convergence of this algorithm assuming certain conditions on the graph and on the distribution of nodes subject to maximum capacities. We then simulate the algorithm over a variety of graphs and initial load configurations to test our results, one example is provided.

Paper Organization. In Section II, we provide preliminary notations and ideas used throughout the paper. In Section III, we describe the problem we wish to solve and formulate it in a useful way. In Section IV, we describe the solution set that we wish to achieve. In Section V, we detail our load balancing algorithm. In Section VI, we prove convergence of our algorithm to our desired solution set under some basic assumptions. In Section VII, we simulate our algorithm and provide results. In Section VIII, we summarize our results and discuss possible ways to extend our work.

II. Preliminaries

This section presents some basic mathematical notations and concepts used throughout the paper.

A. Notation

We let $\mathbb{R}_{\geq 0}$ denote the set of non-negative real numbers, $\mathbb{N}$ denote the set of natural numbers, and $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. Similarly, $\mathbb{R}^n_{\geq 0}$ (resp. $\mathbb{N}^n, \mathbb{N}^n_0$) denotes the product space of n copies of $\mathbb{R}_{\geq 0}$ (resp. $\mathbb{N}, \mathbb{N}_0$). The identity matrix of dimension $n \times n$ is designated by I_n. The transpose of matrix

This work was supported by NSF-CMMI 1434819 and AFOSR 11RSL548. The authors are with the Department of Mechanical and Aerospace Engineering, University of California, San Diego, CA 92093, USA. {egravell,soniamd}@ucsd.edu

B is denoted by $B^\top$. The n-dimensional vector of all ones is represented as $\mathbf{1}_n = [1,\ldots,1]^\top$. The cardinality of set A is denoted by $|A|$. The floor (resp. ceiling) operator on $\mathbb{R}$ is denoted by $\lfloor\cdot\rfloor$ (resp. $\lceil\cdot\rceil$) : $\mathbb{R}_{\geq 0} \to \mathbb{N}_0$. Assigning the value y to a variable x is denoted $x \leftarrow y$.

B. Graph-theoretic notions

We summarize here some basic notions from algebraic graph theory. A *graph* $G = (V, E)$ is composed of a vertex set V indexed from $\{1, \ldots, n\}$ and an edge set $E \subset V \times V$ consisting of ordered pairs of vertices. A graph is undirected if $\forall\,(i, j) \in E$, $(j, i) \in E$. For undirected graphs, $\mathcal{N}_i$ denotes the set of neighbors of node i in V. The graph G is *connected* if for any pair of nodes (i, j) there exists a sequence of edges $(i, i_1), (i_1, i_2), \ldots, (i_k, j)$ connecting i with j. A node i is within *d-hops* of j if there exists a sequence of d or less edges connecting i with j.

III. Problem statement

In this section we define our system and describe its objective. Consider a network of n agents described by an undirected graph $G = (V, E)$. The state of the network, denoted by $x = [x_1, x_2, \ldots, x_n]^\top \in \mathbb{N}_0^n$, is given by the load at each node, which is initially $x(0)$. Each node i has a maximum load capacity $c_i \in \mathbb{N} \cup \{\infty\}$, constraining $x_i(k) \leq c_i$ for all times $k \in \mathbb{N}_0$.

The objective of the network is to spread out loads as evenly as possible across every node of the network. This translates into achieving a state that is a solution to the following minimization problem, Problem (1):

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & W(x) = \sum_{i\in V}(x_i - \overline{x})^2, \\ \text{subject to} \quad & \sum_{i\in V} x_i = n\overline{x}, \\ & x_i \leq c_i,\ \forall\, i \in V, \\ & x_i \in \mathbb{N}_0,\ \forall\, i \in V, \end{aligned} \tag{1}$$

where $\overline{x} = \frac{1}{n}\mathbf{1}_n^\top x_i(0)$. This function $W(x)$ is the *variance* of x, which is a distance function between x and $\overline{x}\mathbf{1}_n$. A state is *feasible* if it satisfies both capacity and integer load constraints as well as conserves all initial loads.

IV. Characterization of balanced states

We will characterize the set of solutions to Problem (1). If a state x^* is in this solution set then we call it a *balanced state*. Assuming some initial state $x(0)$, the following lemma will be useful:

Lemma 4.1: If there exists $p \in \operatorname{argmin}_{i\in V} c_i$ with $c_p \leq \lfloor\overline{x}\rfloor$, then any solution to Problem (1) requires $x_p = c_p$.

This is true because if $x_p < c_p \leq \lfloor\overline{x}\rfloor$, then $W(x^+) < W(x)$ if $x_p^+ = x_p + 1$ and $x_q^+ = x_q - 1$ for $q \in \operatorname{argmax}_{i\in V} x_i$. Note, complete proofs of all lemmas and claims will appear in a future publication, due to space constraints.

Apply Lemma 4.1 to $x(0)$, and if such a p exists, pick one and define $\tilde{V}_1 = \{p\}$, $V_1 = V \setminus \tilde{V}_1$, and $\overline{x}_1 = \frac{n\overline{x}-c_p}{n-1}$. With a slight abuse of notation, we can redefine x as the vector of remaining free states (both here and later on).

Lemma 4.1 can again be applied to this new problem with reduced state. Define a new $p \in \operatorname{argmin}_{i\in V_1} c_i$. If $c_p \leq \lfloor\overline{x}_1\rfloor$ then we know the solution contains $x_p = c_p$. We repeat this process for ℓ iterations, defining $\tilde{V}_\ell = \{p \cup \tilde{V}_{\ell-1}\}$, $V_\ell = V \setminus \tilde{V}_\ell$, and $\overline{x}_\ell = \frac{(n-\ell+1)\overline{x}_{\ell-1}-c_p}{n-\ell}$, until we have $\min_{i\in V_\ell} c_i > \lfloor\overline{x}_\ell\rfloor$. This results in $\tilde{V}_\ell = \{i \in V \mid c_i \leq \lfloor\overline{x}_\ell\rfloor\}$. After defining $m = |V_\ell|$ and labeling our node indices from 1 to m, Problem (1) reduces to

$$\begin{aligned} \underset{x\in\mathbb{R}^m}{\text{minimize}} \quad & W(x) = \sum_{i=1}^{m}(x_i - \overline{x})^2, \\ \text{subject to} \quad & \sum_{i=1}^{m} x_i = m\overline{x}_\ell, \\ & x_i \leq c_i,\ \forall\, i \in V_\ell, \\ & x_i \in \mathbb{N}_0,\ \forall\, i \in V_\ell, \end{aligned} \tag{2}$$

and we know that $c_i > \lfloor\overline{x}_\ell\rfloor$, $\forall\, i \in V_\ell$. To solve this problem we will first temporarily relax the last two constraints so we are left with

$$\begin{aligned} \underset{x\in\mathbb{R}^m}{\text{minimize}} \quad & W(x) = \sum_{i=1}^{m}(x_i - \overline{x})^2, \\ \text{subject to} \quad & \sum_{i=1}^{m} x_i = m\overline{x}_\ell. \end{aligned}$$

We can reduce dimensionality and eliminate the last constraint by substituting $x_m = m\overline{x}_\ell - \sum_{i=1}^{m-1} x_i$, so we are left with

$$\underset{x\in\mathbb{R}^{m-1}}{\text{minimize}} \quad W(x) = \sum_{i=1}^{m-1}(x_i - \overline{x})^2 + (m\overline{x}_\ell - \sum_{i=1}^{m-1} x_i - \overline{x})^2.$$

Setting the gradient to zero and solving the resulting system gives a unique solution in $x = \overline{x}_\ell\mathbf{1}_m$. Because the Hessian is positive definite, this critical point is the global minimum.

In order to meet our integer constraint and find a feasible solution to Problem (2), for each node $i \in \{1, \ldots, m-1\}$ we will choose $x_i = \lfloor\overline{x}_\ell\rfloor$ or $x_i = \lceil\overline{x}_\ell\rceil$. To meet our conservation of loads constraint, we solve

$$\begin{aligned} &\alpha + \beta = m, \\ &\alpha\lfloor\overline{x}_\ell\rfloor + \beta\lceil\overline{x}_\ell\rceil = m\overline{x}_\ell, \end{aligned} \tag{3}$$

where α and β are the number of nodes at $\lfloor\overline{x}_\ell\rfloor$ and $\lceil\overline{x}_\ell\rceil$, respectively. In the event that $\overline{x}_\ell$ is an integer, we define $\alpha = m$ and $\beta = 0$. Otherwise, it must hold that $\alpha =$

$m(\lceil \overline{x}_\ell \rceil - \overline{x}_\ell)$, $\beta = m(\overline{x}_\ell - \lfloor \overline{x}_\ell \rfloor)$. This satisfies our other relaxed constraint $x_i \leq c_i, \forall i \in V$ because we know that $c_i \geq \lceil \overline{x}_\ell \rceil, \forall i \in V_\ell$. Permuting the set of nodes which have $x_i = \lfloor \overline{x}_\ell \rfloor$ or $x_i = \lceil \overline{x}_\ell \rceil$ has no effect on our function $W(x)$, so there can be multiple solutions. Any integer configuration for which $x_q > \lceil \overline{x}_\ell \rceil$ or $x_p < \lfloor \overline{x}_\ell \rfloor$ for some $q, p \in V_\ell$ is not a solution to Problem (2). With the same reasoning as in Lemma 4.1, we can achieve a lower $W(x)$ by increasing x_p and decreasing x_q. With this, we have determined our optimal set of states. We summarize our results with the following theorem:

Theorem 4.2: A state x which solves Problem (1) and is therefore in the set of balanced states must satisfy $x_i = c_i, \forall i \in \tilde{V}_\ell$, $x_i = \lfloor \overline{x}_\ell \rfloor$ for $\alpha = m(\lceil \overline{x}_\ell \rceil - \overline{x}_\ell)$ nodes in V_ℓ and $x_i = \lceil \overline{x}_\ell \rceil$ for the rest, where $\tilde{V}_\ell = \{i \in V \mid c_i \leq \lfloor \overline{x}_\ell \rfloor\}$, $V_\ell = V \setminus \tilde{V}_\ell$, $\overline{x}_\ell$ is previously defined in this section, and $m = n - |\tilde{V}_\ell|$.

V. Load Balancing Algorithm

This section describes our proposed load balancing algorithm that satisfies integer and capacity contraints at all times as well as conserves all initial loads. We have designed a synchronous algorithm which can be divided into three main sections. In the Offering Phase, a node selects among its neighbors with lightest loads to make an offer to. If this node is at its capacity, it may select a node with a larger load than its own, in order to maintain passing connectivity of the graph. In the Accepting Phase, a node selects among its neighbors that offered, usually selecting the node with maximal load but sometimes giving priority to a node at its capacity. In the Passing Phase, passes are made to nodes that have accepted offers, and the amount passed is calculated such that the offering node expects to minimize the difference between the two nodes' loads after the pass. Initially define time $k = 0$, each node $i \in \{1, \ldots, n\}$ runs Offering Phase, Accepting Phase and Execution Phase, and the algorithm terminates when $k = K_f$.

Algorithm 1: Offering Phase for Node i

Inputs : $\forall j \in \mathcal{N}_i \cup i$, $x_j(k)$ and c_j. K_f.

```
x_i^min(k) ← min_{j∈N_i} x_j(k) ;
J_i(k) ← {j ∈ N_i | x_j(k) = x_i^min(k)} ;
With probability p_i = 1/|J_i(k)| choose h ∈ J_i(k) ;
if ⌈(x_i(k) − x_h(k))/2⌉ > c_h − x_h(k) then
    x_i^min*(k) ← min_{j∈N_i\h} x_j(k);
    J_i^*(k) ← {j ∈ N_i \ h | x_j(k) = x_i^min*(k)};
    With probability p_i^* = 1/|J_i^*(k)|, choose h^* ∈ J_i^*(k);
    if x_{h^*}(k) ≤ x_i(k) − 2(c_j − x_j(k)) then
        h ← h^*;
    end
end
if x_i(k) > x_h(k) or (x_i(k) = c_i and k < K_f) then
    send a message to h consisting of (i, c_i, x_i(k));
end
```

Algorithm 2: Accepting Phase for Node i

Inputs : $M_1 = \{(j_1, c_{j_1}, x_{j_1}(k)), \ldots, (j_m, c_{j_m}, x_{j_m}(k))\}$ is the set of messages received, $m_1 \triangleq |M_1|$.

```
if m_1 > 0 then
    x_i^max(k) ← max_{s∈{1,...,m}} x_{j_s}(k) ;
    if ∃ c_{j_s} < x_i(k), s ∈ {1,...,m} and
    x_i^max(k) ≤ x_i(k) + 1 then
        h ← j_s;

    else
        Q_i(k) ← {j_s ∈ M_1 | x_{j_s}(k) = x_i^max(k)}, s ∈
        {1,...,m};
        With probability p_i = 1/|Q_i(k)|, choose
        h ∈ Q_i(k);
    end
    Send message to h consisting of (i);
end
```

VI. Convergence Analysis

In this section we prove convergence of our algorithm to the set of balanced states, under the following assumptions:

Assumption 6.1 (Graph Connectivity): The undirected graph $G = (V, E)$ is connected.

Assumption 6.2 (Capacity Distribution): For each node i with finite capacity $c_i < \max_{j \in V} x_j(0) + 1$, there is no node h with $c_h < \max_{j \in V} x_j(0) + 1$ within 2-hops of i.

Assumption 6.3 (Neighbor Condition): Each node i with $c_i < \max_{j \in V} x_j(0) + 1$ has at least two neighbors.

We use these assumptions to state the following theorem, the main result of this paper:

Theorem 6.4: Consider a network of nodes modeled by an undirected graph $G = (V, E)$, subject to the constraints found in Problem (1). Given Assumptions 6.1 (Graph Connectivity), 6.2 (Capacity Distribution), 6.3 (Neighbor Condition), and applying the dynamics of our quantized distributed load balancing algorithm, any feasible initial state $x(0)$ converges to a balanced state in finite expected time.

Proof: We prove convergence of our algorithm to the set of balanced states by using ideas from Lyapunov stability theory. Initially define $k = 0$, $z = 0$ where z is a subsequence of k . Define $q_z = \min_{i \in V}\{x_i(0) \mid x_i(0) \leq c_i - 2\}$ and define $A_z = \{i \in V \mid x_i(0) = q_z,\ x_i(0) \leq c_i - 2\}$. For every node j s.t. $x_j(0) = c_j$, remove one node $h \in \mathcal{N}_j$ from A_z (if it exists), define a q-coin at every other node in A_z, define $S(k) = \{i \in V \mid i \text{ has a } q\text{-coin at time } k\}$, and define $s_k = |S(k)|$. A q-coin (or simply coin) at a node represents that this node has or can create a minimum load value across the network. We will later define q-coin dynamics for all

Algorithm 3: Passing Phase for Node i

Inputs : $M_2 = \{(j_1), \ldots, (j_p)\}$ is the set of messages received, where $m_2 \triangleq |M_2|$. K_f.

if $m_2 > 0$ **then**
  if $x_i(k) > x_j(k)$ **then**
    $\delta \leftarrow \min\left(\lceil \frac{x_i(k)-x_j(k)}{2} \rceil,\, c_j - x_j(k)\right)$;
  else
    $\delta \leftarrow 1$;
  end
  Pass δ loads to j ;
end
if $k < K_f$ **then**
  increment k.
else
  Terminate algorithm.
end

times $k \geq 0$. Over time, q-coins will be destroyed, and once there are none we will redefine new q-coins but at a greater q_z value; this will be useful in proving convergence. This set $S(k)$ will be defined at each time step, and q_z and A_z are only redefined when $s_k = 0$.

We define

$$\hat{W}(x) \triangleq W(\hat{x}) = \sum_{i \in V} (\hat{x}_i - \overline{x})^2,$$

where $\overline{x} = \frac{1}{n}\mathbf{1}^\top x(0)$ is the average of the initial states and $\hat{x}$ is a modification of state x. We define $\hat{x}$ by the following rule: if for some $i \in V$, $x_i = c_i$ and $\forall\, j \in \mathcal{N}_i$, $x_j \geq x_i$, then $\hat{x}_i = x_i - 1$ and $\hat{x}_h = x_h + 1$ for some $h \in \operatorname{argmin}_{j \in \mathcal{N}_i} x_j$. Otherwise, $\hat{x}_i = x_i$. Note that the value of $\hat{W}$ is independent of the choice of $h \in \operatorname{argmin}_{j \in \mathcal{N}_i} x_j$. Intuitively, $\hat{x}$ is a feasible state where all *capped* nodes (nodes in the set $\tilde{V}_\ell$) with only higher loaded neighbors have passed a load to temporarily unbalance the system; this represents a sort of worst-case scenario in terms of minimizing W.

We define $\tilde{W}(x) = \hat{W}(x) + W_c(x)$, where $W_c(x) = (n+1)(\lfloor \overline{x}_\ell \rfloor - q_z) + s_k$. Recall that $\overline{x}_\ell$ was found in Section IV and corresponds to the average of the excess load that needs to be divided among the remaining non-capped nodes. We know that $0 \leq q_z \leq \lfloor \overline{x}_\ell \rfloor$ and $0 \leq s_k \leq n$, so $W_c(x)$ is positive, implying $\tilde{W}(x(k))$ is positive for all $k \geq 0$. With a slight abuse of notation we denote $\tilde{W}(x(k)) = \tilde{W}(k)$, with similar conventions for $\hat{W}$ and W_c. When a q-coin is destroyed from time k to $k+1$ and if $s_k, s_{k+1} \geq 1$ then $W_c(k+1) < W_c(k)$. If s_{k+1} reaches zero, then at that time step we will redefine q_z and q-coins until $q_z = \lfloor \overline{x}_\ell \rfloor$. While s_k may increase by some amount between zero and n after redefining coins, $(n{+}1)(\lfloor \overline{x}_\ell \rfloor {-} q_z)$ decreases by at least $n{+}1$ so overall $W_c(k)$ decreases. We will later prove that $\hat{W}(k)$ is non-increasing over time, and we will use the composite $\tilde{W}(k)$ to prove convergence.

Given the existence of coins on the graph, they can be passed or destroyed. Zero coins on the graph implies that the graph no longer contains that particular minimum value, so that once this happens, we can increment z by one, define $q_z = q_{z-1} + 1$, and redefine coins in a similar way as done at step $k = 0$ but with respect to the new value of q_z and allowing for a node to be at $q_z - 1$. We repeat this process until $q_z = \lfloor \overline{x}_\ell \rfloor$. We will precisely characterize the set of nearly balanced states later in order to conclude the proof.

First, we will provide an overview on the motion of q-coins. Most of the time, a coin is passed from a node i to a node j at time k if $x_j(k+1)$ is reduced to either q_z or $c_j - 1$ due to it passing one load to node i. This rule does not precisely hold around capped nodes or given chains of passing, but the idea is that if a pass does not decrease $\tilde{W}(k)$, then a coin is being passed, and if a pass does decrease $\tilde{W}(k)$, then a coin is likely being destroyed. Precise details of the coin motion have been omitted.

To prove convergence using $\tilde{W}(k)$, we need to prove that these coins are expected to disappear in finite time if the state is not nearly balanced, and to do so we prove that a q-coin has a finite expected meeting time with a node h s.t. $x_h(k) \geq q_z + 2$. A meeting between a coin and a node h s.t. $x_h(k) \geq q_z + 2$ at time k is defined when they are neighbors. Note, the number of coins can be determined for any state $x(k)$ without knowledge of previous states, so we use both $W(x)$ and $W(k)$ interchangeably.

A few useful claims will help us characterize coins and the function $\hat{W}(x)$. In the rest of the claims in this paper, we assume both Assumption 6.2 (Capacity Distribution) and Assumption 6.3 (Neighbor Condition) hold.

Claim 6.5: Any node i s.t. $x_i(k) = q_z$, $c_i \geq q_z + 2$ has a q-coin or has a neighbor j s.t. $x_j(k) = c_j$, $c_j \leq q_z$, for all $k \geq 0$.

This follows from the definition of coins and their dynamics.

Claim 6.6: Any node i s.t. $x_i(k) = q_z - 1$, $c_i \geq q_z + 1$ must contain a q-coin for all $k \geq 0$.

This also follows from the definition of coins and their dynamics.

Claim 6.7: One node cannot contain two or more coins at any time k.

The coin dynamics along with rules for passing never allow this to occur.

Claim 6.8: The state of every node is upper bounded for all time by $\overline{m} = \max_{i \in V} x_i(0) + 1$.

Given that a node can have at most one capped neighbor, there does not exist a sequence of states such that some node i achieves $x_i(k) \geq \max_{i \in V} x_i(0) + 2$.

Claim 6.9: Any node i s.t. $c_i \geq q_z$ is lower bounded by $q_z - 1$, forward in time.

Given that a node can have at most one capped neighbor, there does not exist a sequence of states such that some node

i achieves $x_i(k) \leq \max_{i \in V} q_z - 2$.

The next claim requires a precise definition of the set of nearly balanced states. By Claim 6.11, these are the states partly characterized by the minimum value of $\tilde{W}$. Intuitively, this set contains states which can be balanced in one time step and pairs load values at $\lfloor \overline{x}_\ell \rfloor$ with capped nodes.

Definition 6.10 (Nearly Balanced States): Consider Problem (3). Let α be the number of nodes at $\lfloor \overline{x}_\ell \rfloor$ for some balanced state given $x(0)$ and $|\tilde{V}_\ell| = n - |V_\ell|$, where $|V_\ell|$ is the number of nodes whose capacities do not affect the solution to Problem (1). A *nearly balanced state* satisfies either one of the following conditions:

(i) If $\alpha < |\tilde{V}_\ell|$, then there are α capped nodes which are either capped and have exactly one neighbor at $\lfloor \overline{x}_\ell \rfloor$ with the rest at $\lfloor \overline{x}_\ell \rfloor + 1$ or are below their cap by one unit and have all neighbors at $\lfloor \overline{x}_\ell \rfloor + 1$. The remaining $|\tilde{V}_\ell| - \alpha$ capped nodes are either at their cap with all neighbors at $\lfloor \overline{x}_\ell \rfloor + 1$ or below their cap by one unit with one neighbor at $\lfloor \overline{x}_\ell \rfloor + 2$ and the rest of the neighbors at $\lfloor \overline{x}_\ell \rfloor + 1$. All other nodes are at $\lfloor \overline{x}_\ell \rfloor$ or $\lceil \overline{x}_\ell \rceil$.

(ii) If $\alpha \geq |\tilde{V}_\ell|$ then each capped node below its cap by one unit has a neighbor at $\lfloor \overline{x}_\ell \rfloor + 1$ and each capped node at its cap has a neighbor at $\lfloor \overline{x}_\ell \rfloor$. All other nodes are at $\lfloor \overline{x}_\ell \rfloor$ or $\lceil \overline{x}_\ell \rceil$.

Claim 6.11: Let $\tilde{W}^*$ be the minimum value of $\tilde{W}$ over all feasible x and recall α, $\lfloor \overline{x}_\ell \rfloor$, and $\tilde{V}_\ell$ from Section IV. We can say that $x(k)$ is a nearly balanced state if and only if $\tilde{W}(x(k)) = \tilde{W}^*$.

Proof: The proof of this is done by comprehensively showing that any state s.t. $\tilde{W}(k) > \tilde{W}^*$ is not nearly balanced, and every state with $\tilde{W}(k) = \tilde{W}^*$ is nearly balanced. ■

Claim 6.12: It holds that $\hat{W}(k) \geq \hat{W}(t) \geq W(t), \forall\, t \in [k, \ldots, K_f - 1]$.

Proof: We know $\hat{W}(t) \geq W(t)$ because any difference between $\hat{x}(t)$ and $x(t)$ will lead to $\hat{W}(t) \geq W(t)$. To prove $\hat{W}(k) \geq \hat{W}(t), \forall\, t \in [k, \ldots, K_f - 1]$, we show that every feasible pass of loads that the algorithm can make from any feasible state does not increase $\hat{W}(k)$. ■

Claim 6.13: Given Assumption 6.1 (Graph Connectivity), and any feasible initial state $x(0)$, we have convergence to the set of nearly balanced states in finite expected time.

Proof: Because of Claim 6.12, we can use $\tilde{W}(k) = \hat{W}(k) + W_c(k)$ as a Lyapunov function for this system. We treat $W_c(k)$ as the main component and use $\hat{W}(k)$ to show balancing for some cases where $W_c(k)$ does not capture it. The function $\tilde{W}$ is positive, nonincreasing under the dynamics, and any decrease has a fixed lower bound. Proving that $\tilde{W}(k)$ will decrease in finite expected time from any state $x(k)$ that is not nearly balanced is enough to prove convergence to the set of nearly balanced states.

We will provide an outline for the proof here, the complete proof will appear in a forthcoming publication. We first show that when $W_c(k) > W_c^*$, the expected time of a decrease in W_c is finite. To ensure this, we prove that there is sufficient randomness of coin motion over the graphs, sufficient coverage of the coins, and we know that when a coin meets a load with large enough load this decrease will occur. Once $W_c(k) = W_c^*$, we prove that if $\hat{W}(k) > \hat{W}^*$ this will also decrease in finite expected time, using a similar argument but without coins. Once both $W_c(k) = W_c^*$ and $\hat{W}(k) = \hat{W}^*$, then we know $\tilde{W}(k) = \tilde{W}^*$ and we are in the set of nearly balanced states.

■

Claim 6.14: The set of nearly balanced states is invariant under the dynamics of our algorithm.

Any feasible load transfer our algorithm performs results in a state that satisfies the constraints on the set of nearly balanced states.

Claim 6.15: If $x(k)$ is a nearly balanced state for some $k < K_f$, then $x(K_f)$ is balanced.

Proof: We require $x(k)$ to be a nearly balanced state for some $k < K_f$. Due to Claim 6.14 we know $x(K_f - 1)$ is also nearly balanced. The last run of our algorithm ensures that for our last iteration K_f, if $x_i(K_f - 1) < c_i \leq \lfloor x_\ell \rfloor$, then $x_i(K_f) = c_i$ and no node i at its capacity is permitted to offer to a node j s.t. $x_j(K_f - 1) \geq x_i$. In conclusion, our final state is: for some valid $\tilde{V}_\ell$, $\forall\, i \in \tilde{V}_\ell$, $x_i(K_f) = c_i$, and $\forall\, i \in V_\ell$, $x_i(K_f) \in \{\lfloor \overline{x}_\ell \rfloor, \lceil \overline{x}_\ell \rceil\}$. ■

We have proven convergence to the set of balanced states in finite expected time.

■

VII. Simulations

We have simulated this algorithm on a wide range of graphs, each with various capacity configurations and initial load conditions. One sufficiently complex graph is seen in Figure 1. The square vertices represent nodes with finite capacities and the circular vertices represent nodes with infinite capacities. We can see that nodes with finite capacities are located at central points on the graph, and if removed will disconnect the graph. The initial load configuration is shown in Figure 2 and the final load configuration is shown in Figure 3, after K_f iterations of our algorithm. In both of these figures, finite capacity values are shown as triangles. We can see in Figure 4 the nonincreasing behavior of $\tilde{W}(k)$ and convergence towards the optimal value.

VIII. Conclusions and Future Work

We have designed a new quantized distributed load balancing algorithm that accounts for maximum node capacities. We proved convergence to a set of balanced states in expected finite time under some assumptions on the location as well as a neighbor condition of nodes with small capacities. We

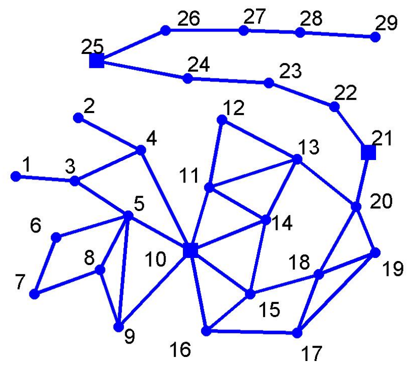

Fig. 1: Example graph

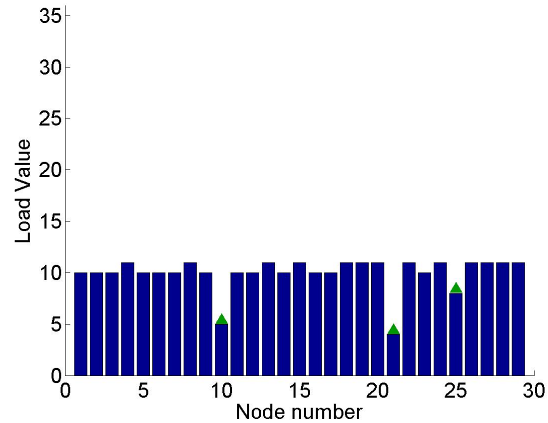

Fig. 3: Final Load Distribution

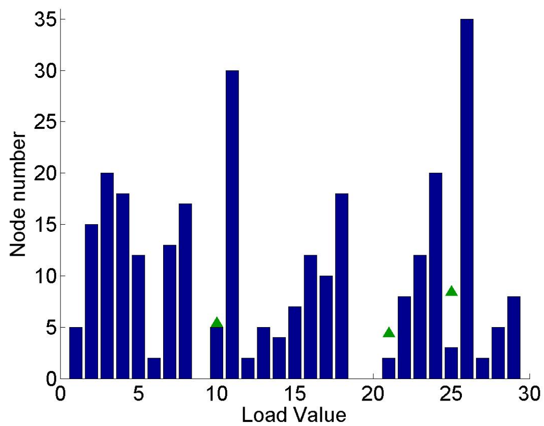

Fig. 2: Initial Load Distribution

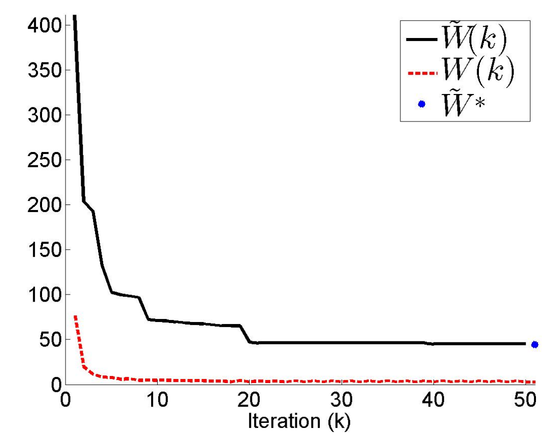

Fig. 4: $W(x)$ over K_f iterations

tested this algorithm over a variety of graphs and initial conditions, and provided one such example.

Future work could include relaxing the listed assumptions for convergence, which can be done under certain initial conditions. Relaxing the assumptions and restructuring the proof could lead to more general results, such as classifying for which initial load configurations and capacity values do we not require the two-hop constraint on nodes with capacities. This algorithm can be extended to account for both minimum and maximum node capacities over a graph. It is also desirable to find upper and lower bounds on the convergence rate.

References

[1] T. C. Aysal, M. J. Coates, and M. G. Rabbat. Distributed average consensus with dithered quantization. *IEEE Transactions on Signal Processing*, 56(10):4905–4918, 2008.

[2] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[3] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis. Convergence in multiagent coordination, consensus, and flocking. In *IEEE Int. Conf. on Decision and Control and European Control Conference*, pages 2996–3000, Seville, Spain, December 2005.

[4] R. Carli and S. Zampieri. *Advances in Control Theory and Applications*. Springer Berlin Heidelberg, 2007.

[5] M. Franceschelli, A. Giua, and C. Seatzu. Load balancing on networks with gossip-based distributed algorithms. In *IEEE Int. Conf. on Decision and Control*, pages 500–505, 2007.

[6] M. Franceschelli, A. Giua, and C. Seatzu. Load balancing over heterogeneous networks with gossip-based algorithms. In *American Control Conference*, pages 1987–1993, 2009.

[7] M. Franceschelli, A. Giua, and C. Seatzu. A gossip-based algorithm for discrete consensus over heterogeneous networks. *IEEE Transactions on Automatic Control*, 55(5):1244–1249, 2010.

[8] P. Frasca, R. Carli, F. Fagnani, and S. Zampieri. Average consensus by gossip algorithms with quantized communication. In *IEEE Int. Conf. on Decision and Control*, pages 4831–4836, Cancún, México, December 2008.

[9] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003.

[10] A. Kashyap, T. Başar, and R. Srikant. Quantized consensus. *Automatica*, 43(7):1192–1203, 2007.

[11] J. Lavaei and R. M. Murray. Quantized consensus by means of gossip algorithm. *IEEE Transactions on Automatic Control*, 57(1):19–32, 2012.

[12] A. Nedić, A. Olshevsky, and A. Ozdaglar. On distributed averaging algorithms and quantization effects. *IEEE Transactions on Automatic Control*, 54(11):2506–2517, 2009.

[13] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[14] M. Zhu and S. Martínez. On the convergence time of asynchronous

distributed quantized averaging algorithms. *IEEE Transactions on Automatic Control*, 56(2):386 – 390, 2011.