Distributed Task Assignment Based on Gossip with Guaranteed Performance on Heterogeneous Networks ⋆

Mauro Franceschelli [*] **Alessandro Giua** [**,*] **Carla Seatzu** [*]

[*] *Dept. of Electrical and Electronic Engineering, University of Cagliari, Italy (e-mail: {mauro.franceschelli,giua,seatzu}@diee.unica.it).*
[**] *Aix Marseille Université, CNRS, ENSAM, Université de Toulon, LSIS UMR 7296,13397, Marseille, France (e-mail: alessandro.giua@lsis.org)*

Abstract: In this paper we propose a novel distributed algorithm for task assignment on heterogeneous networks. We consider a set of tasks with heterogeneous cost to be assigned to a set of nodes with heterogeneous execution speed and interconnected by a network with unknown topology represented by an undirected graph. Our objective is to minimize the execution time of the set of tasks by the networked system. We propose a local interaction rule which allows the nodes of a network to cooperatively assign tasks among themselves with a guaranteed performance with respect to the optimal assignment exploiting a gossip based randomized interaction scheme. We characterize the convergence properties of the proposed approach and provide simulation results.

Published as: M. Franceschelli, A. Giua, C. Seatzu, "Distributed Task Assignment Based on Gossip with Guaranteed Performance on Heterogeneous Networks," ADHS15: 5th IFAC Conference on Analysis and Design of Hybrid Systems (Atlanta, GA, USA), Oct 14-16, 2015.

The research leading to these results has received funding from Region Sardinia, LR 7/2007 (call 2010) under project SIAR (CRP-24709) and from Italian grant SIR "Scientific Independence of young Researchers", project CoNetDomeSys, code RBSI14OF6H, funded by the Italian Ministry of Research and Education (MIUR).

1. INTRODUCTION

In this paper we deal with the problem of distributing evenly a set of indivisible tasks over a network of agents who have to process them. To keep the presentation general, we consider tasks with different costs and agents with different execution speeds. This problem, which is denoted as *discrete consensus* in Franceschelli et al. (2010), is a generalization of the *quantized consensus* problem proposed in Kashyap et al. (2007) where, instead, all tasks (or tokens) have equal weight (or cost) and task execution speed is not considered. Moreover, we say that the assignment is performed on heterogeneous networks to emphasize the fact that each agent has its own execution speed. Agents are assumed to be interconnected by bidirectional communication network. Our goal is that of minimizing the execution time of the set of tasks by the networked system. The assignment is performed according to a novel distributed algorithm based on gossip-like asynchronous local state updates between the nodes. As a result of the proposed local interaction mechanism, the achievement of an optimal task assignment is not guaranteed. However, we are able to prove almost sure convergence in finite time to a bounded set containing optimal solutions. We guarantee that the worst case difference between the optimal value of the execution time and the performance of the proposed algorithm is bounded by a constant which does not depend on the network size.

The quantized consensus algorithm in Kashyap et al. (2007) and the discrete consensus algorithm proposed in Franceschelli et al. (2010) are based on a local balancing rule to redistribute tasks or tokens among selected pairs of nodes and a so called "swap" rule which updates the state of the nodes by simply swapping their set of tasks or tokens. The swap rule is necessary to avoid blocking configurations. It "shakes" the network state to redistribute the load and allows loads composed by discrete tasks to travel in the network, reaching a situation in which a new balancing may occur. The study of the convergence time of this process depends upon the meeting time of two random walkers in a graph and has received a significant attention by Zhu and Martínez (2011); Etesami and Başar (2013); Başar et al. (2014) among others.

When considering heterogeneous execution speed, methods developed for the homogeneous case may fail to converge to the desired convergence set. The approach in Franceschelli et al. (2010), which considers heterogeneous execution speeds, suffers the limitation that each pair of *swap domains*, namely connected subgraphs induced by nodes with the same speed, should be connected.

The main contribution of this paper is a novel distributed algorithm which allows to remove the limitation of Franceschelli et al. (2010) and considers arbitrary topologies modeled by connected undirected graphs with nodes of arbitrary speed. Furthermore, we characterize the convergence properties of the algorithm in terms of almost sure convergence to a set of task assignments which well approximate the optimal solution. We prove an absolute performance guarantee by showing that in the worst case scenario the execution time obtained by the proposed algorithm does not differ from the optimal one by more than a constant which does not depend on network size. Thus, the proposed approach is well suited to address the task assignment problem in large networks.

Finally, we propose numerical simulations to characterize the expected convergence time of the proposed algorithm in line graphs and random graphs and compare it with the performance of the algorithm in Franceschelli et al. (2010). We conclude the Introduction with a brief overview of the state of the art in this area.

Literature review: One of the first major contributions to the problem of *quantized consensus* which inspired several works in the following years, was proposed Kashyap et al. (2007). It consists in a gossip-based algorithm to steer a set of quantized state variables towards a common value. Gossip-based algorithms have been inspired by Boyd et al. (2006), among others, which considered a randomized averaging scheme to solve the distributed average problem in sensor networks. The issue of providing a characterization of the convergence time of quantized consensus algorithms is investigated in depth in Zhu and Martínez (2011), Lavaei and Murray (2012) and more recently in Etesami and Başar (2013).

A series of contributions in the framework of *discrete consensus* have been recently proposed. Apart from Franceschelli et al. (2010) that has already been recalled in the first part of this section, we want to mention Fanti et al. (2012) where a discrete consensus algorithm in networks affected by execution feasibility constraints has been considered. In Fanti et al. (2013) tasks of different cost and type with capacity and feasibility constraints are considered. Furthermore, authors impose a constraint on the maximum number of tasks executable by individual nodes. Almost sure convergence to a feasible and time-invariant configuration is proved. However, only preliminary results on the converge time are proposed. In Gravelle and Martinez (2014) a modification of the quantized consensus algorithm is proposed to solve a load balancing problem with tasks of identical size and nodes with limited capacity.

In Franceschelli et al. (2011) a discrete consensus algorithm with improved convergence time with respect to quantized consensus algorithms is proposed for Hamiltonian graphs. In Franceschelli et al. (2015) the expected convergence time of discrete and quantized consensus has been improved on arbitrary graphs to $\mathcal{O}(n)d(\mathcal{G})$, where n is the number of nodes and $d(\mathcal{G})$ is the diameter of graph $\mathcal{G}$ representing the network topology.

In Franceschelli et al. (2013) the distributed task assignment problem in a network of heterogeneous mobile robots with heterogeneous tasks is investigated. The authors exploit gossip based local optimizations to both assign tasks located in a plane and compute optimized routes for robots. Finally, in Chopra and Egerstedt (2014) heterogeneity in multi-robot

systems is investigated as the ability of robots with heterogeneous skill sets to serve spatially and temporally distributed tasks.

2. PROBLEM STATEMENT

Consider a network of n nodes whose pattern of interconnections can be described by an undirected connected graph $\mathcal{G} = (\mathcal{V}, E)$, where $\mathcal{V} = \{1, \dots, n\}$ is the set of nodes and $E \subseteq \{\mathcal{V} \times \mathcal{V}\}$ is the set of edges. We consider K indivisible tasks to be assigned to the nodes with execution cost $c_j \in \mathbb{N}^+$, $j = 1, \dots, K$ associated with each task. We define the *maximal cost* $c_{\max} = \max_{j=1,\dots,K} c_j$, and the *average load* $c_{ave} = \frac{1}{n}\sum_{j=1,\dots,K} c_j$. The tasks assigned to each node can be specified by n binary vectors $y_i \in \{0,1\}^K$ such that $y_{i,j} = 1$ if the j-th task is assigned to node i, $y_{i,j} = 0$ otherwise. The *load* assigned to node i is $c^T y_i$, i.e., it represents the total cost of tasks assigned to it (here $c \in \mathbb{N}^K$ is a vector whose j-th component is equal to c_j). The current *task assignment* of the network is thus $Y = [y_1\ y_2\ \dots\ y_n] \in \{0,1\}^{K\times n}$. With each node is associated an execution speed γ_i and we define the *minimal speed* $\gamma_{min} = \min_{i=1,\dots,n} \gamma_i$ and the *average speed* $\gamma_{ave} = \frac{1}{n}\sum_{i=1,\dots,n} \gamma_i$. The *execution time* of node i is therefore $x_i = \frac{c^T y_i}{\gamma_i}$. We define the *network execution time* as

$$F(Y) = \max_{i\in\mathcal{V}} \frac{c^T y_i}{\gamma_i}, \tag{1}$$

i.e., it corresponds to the maximum execution time among all nodes.

Our objective is to minimize the execution time of the network. An optimal assignment Y^* can be found as the solution of the following integer programming problem with binary variables

$$\begin{cases} \min & F(Y) = \max\limits_{i\in\mathcal{V}} \dfrac{c^T y_i}{\gamma_i} \\ s.t. & Y\mathbf{1}_n = \mathbf{1}_n \\ & y_{i,j} \in \{0,1\} \quad \forall i \in \mathcal{V},\ \ j = 1, \dots, K. \end{cases} \tag{2}$$

We denote by $\mathbf{1}_n$ a vector of ones with n elements. The constraints $Y\mathbf{1}_n = \mathbf{1}_n$ imposes that tasks are indivisible and can be assigned only to one node. For large number of nodes and tasks, the computational complexity of the integer programming problem (2) can be extremely high. In particular, (2) is a formulation of the makespan minimization problem on identical parallel machines. The complexity of finding an exact solution to this problem is known to be NP-hard, see Garey and Johnson (1979) for reference. Furthermore, it requires a centralized coordinator with full knowledge of the network state and ability to communicate with all the nodes.

In this manuscript, we aim to develop a distributed algorithm which by exploiting only local and asynchronous interactions between the nodes is able to achieve a task assignment with a guaranteed distance from the optimum. In particular we consider a target set

$$\mathcal{Y} = \left\{ Y \ \mid \ F(Y) \leq F(Y^*) + \frac{c_{max}}{\gamma_{min}} \right\}. \tag{3}$$

A solution $Y \in \mathcal{Y}$ provides an absolute performance guarantee with bounded error which does not depend on the size of the network, i.e., on the number of nodes. We point out that in the case tasks have unitary cost and nodes have unitary speed the set in eq. (3) contains the same set of task assignments that correspond to the quantized consensus state in Kashyap et al. (2007).

3. PROPOSED ALGORITHM

The triple $(\hat{c}_{ave,i}(t), \hat{\gamma}_{ave,i}(t), \mathcal{K}_i(t))$ represents the state of node i at time t, where:

- $\hat{c}_{ave,i}(t)$ denotes the current estimate at node i of the average load c_{ave} in the network;
- $\hat{\gamma}_{ave,i}(t)$ denotes the current estimate at node i of the average speed γ_{ave} in the network;
- $\mathcal{K}_i(t) = \{j \mid y_{i,j} = 1\}$ denotes the set of indices of tasks currently assigned to node i.

The first two components of the state are called *local estimation variables* while the last one is the *task assignment*.

We consider a gossip model of communication between agents, driven by a random edge selection process, described in Algorithm 1 (Heterogeneous Discrete Consensus). At each iteration an arbitrary edge (i, j) is selected, and nodes i and j communicate to update their state. First, the two nodes execute an averaging of the local estimation variables. In addition they execute the Balancing Rule described in Algorithm 2 to update their task assignment. We make the following common assumption concerning the network and the edge selection process.

***Assumption** 3.1.* The underlying undirected graph is connected and at each iteration all arcs have a non-null lower bounded probability of being selected. ◇

Note that in the following when it is not necessary we will omit the argument t: the current state will be denoted by $(\hat{c}_{ave,i}, \hat{\gamma}_{ave,i}, \mathcal{K}_i)$, while the updated state at each iteration will be denoted by $(\hat{c}^+_{ave,i}, \hat{\gamma}^+_{ave,i}, \mathcal{K}^+_i)$.

To simplify the presentation of our algorithms, we also denote the *execution time* of a node i with task assignment $\mathcal{K}_i$ as: $x_i(\mathcal{K}_i) = \frac{1}{\gamma_i}\sum_{r\in\mathcal{K}_i} c_r$.

Algorithm 1: Heterogeneous Discrete Consensus (HDC)

Input : Sets $\mathcal{K}_i(0)$, for $i \in \mathcal{V}$ (*initial assignment of tasks to nodes*).
Output: Sets $\mathcal{K}_{i,\infty}$, for $i \in \mathcal{V}$ (*final assignment of tasks to nodes*).
1 - Initialize: For $i \in \mathcal{V}$, let

$$\hat{\gamma}_{ave,i}(0) = \gamma_i \quad \text{and} \quad \hat{c}_{ave,i}(0) = \sum_{r\in\mathcal{K}_i(0)} c_r.$$

2 - while *NOT stop_criterion* **do**
3 - A random edge (i, j) is selected according to a given stochastic selection process.
4 - Update the local estimation variables according to

$$\begin{aligned}
\hat{c}^+_{ave,i} &= \tfrac{1}{2}(\hat{c}_{ave,i} + \hat{c}_{ave,j}) \\
\hat{c}^+_{ave,j} &= \tfrac{1}{2}(\hat{c}_{ave,i} + \hat{c}_{ave,j}) \\
\hat{\gamma}^+_{ave,i} &= \tfrac{1}{2}(\hat{\gamma}_{ave,i} + \hat{\gamma}_{ave,j}) \\
\hat{\gamma}^+_{ave,j} &= \tfrac{1}{2}(\hat{\gamma}_{ave,i} + \hat{\gamma}_{ave,j})
\end{aligned} \tag{4}$$

and let $\hat{x}_{ave,i} = \hat{c}^+_{ave,i}/\hat{\gamma}^+_{ave,i}$.
5 - Update the task assignment of nodes i and j according to

$$(\mathcal{K}_i^+, \mathcal{K}_j^+) = \textbf{Balancing rule}(\mathcal{K}_i, \mathcal{K}_j, \gamma_i, \gamma_j, \hat{x}_{ave,i})$$

as described in Algorithm 2.

We now discuss separately the two types of updates executed by Algorithm 1. The stopping condition will be discussed later.

Update of the local estimation variables. These variables are initialized, respectively, with the node initial load and with the node speed. The evolution of these two variables, that does not depend on the current task assignments, follows the well known *gossip averaging algorithm* whose properties have been investigated in Boyd et al. (2006). The computed value $\hat{x}_{ave,i} = \hat{c}_{ave,i}/\hat{\gamma}_{ave,i}$ is the estimate of the average execution time assuming it may be possible to assign to each node a fraction of total load in the network proportional to its speed (but this may not be possible due the task discretization).

Update of the task assignments. The task assignments of communicating nodes are updated as described in Algorithm 2. Initially (step 2) a simple heuristic is used to average the load of two nodes incident on the selected edge. This heuristic is a modification of the very well known algorithm for the 2-machine N job problem by Johnson (1954) and is completed in a number of steps proportional to the number of tasks contained in node i and j. Variations of this greedy and widely known heuristics have been investigated in the context of load distribution between two parallel machines and is a polynomial time approximation of the 2-partitioning problem (Babel et al. (1998)). This rule computes two updated assignments $\mathcal{K}_i^+$, $\mathcal{K}_j^+$. If the new assignments do not yield a smaller local execution time we revert to the original assignments (step 3). However, in such a case we also check if the maximum local execution time exceeds the estimated average time by a quantity greater than $c_{\max}/\gamma_{\min} + \varepsilon/2$ (step 4): if this is true we move one random task from one node to the other one to shake the network configuration and avoid being stuck in local minima. Here ε is a designer parameter that will discussed in the following section. Note also that we assume the exact value of $c_{\max}$ and $\gamma_{\min}$ to be known to all nodes: if these parameters are not available, it is possible to estimate them with max-consensus algorithms such as those developed by He et al. (2014).

Stopping condition.

The stopping criterion of the proposed algorithm is straightforward: when all nodes have an accurate estimation $\hat{x}_{ave,i}$ and a local execution time below the estimated threshold $\hat{x}_{ave,i} + \frac{c_{max}}{\gamma_{min}} + \frac{\varepsilon}{2}$, then task exchanges do not occur anymore. In this preliminary paper we do not discuss how to stop the edge selection process once a satisfactory task assignment has been achieved, we aim to address this problem in future work. We point out that the current literature on quantized consensus algorithms does not consider a stop criterion.

4. CONVERGENCE PROPERTIES

To study the convergence properties of the proposed algorithm, we first provide a lower bound to the optimal value $F(Y^*)$ of the execution time.

***Proposition** 4.1.* A lower bound on the optimal value of the objective function of Problem (2) is:

$$\frac{c_{ave}}{\gamma_{ave}} \leq F(Y^*). \tag{5}$$

Algorithm 2: Balancing rule

Input : $\mathcal{K}_i$, $\mathcal{K}_j$, γ_i, γ_j, $\hat{x}_{ave,i}$ *(current node task assignments, node speeds and estimated average execution time)*
Output: $\mathcal{K}_i^+$, $\mathcal{K}_j^+$ *(updated node task assignments)*
1 - Initialize: Let $\mathcal{K} = \mathcal{K}_i \cup \mathcal{K}_j$, let $\mathcal{K}_i^+ := \emptyset$ and $\mathcal{K}_j^+ := \emptyset$.
2 - while $\mathcal{K} \neq \emptyset$ **do**
  let $\delta := \text{argmax}_{j \in \mathcal{K}} c_j$;
  if $x_i(\mathcal{K}_i^+) + c_\delta/\gamma_i \leq x_j(\mathcal{K}_j^+) + c_\delta/\gamma_j$ **then**
    let $\mathcal{K}_i^+ := \mathcal{K}_i^+ \cup \{\delta\}, \quad \mathcal{K}_j^+ := \mathcal{K}_j^+$;
  else
    let $\mathcal{K}_i^+ := \mathcal{K}_i^+, \quad \mathcal{K}_j^+ := \mathcal{K}_j^+ \cup \{\delta\}$.
  (assign task δ so as to minimally increase the maximal execution time of the two nodes) $\mathcal{K} := \mathcal{K} \setminus \{\delta\}$.
3 - if $\max(x_i(\mathcal{K}_i^+), x_j(\mathcal{K}_j^+)) \geq \max(x_i(\mathcal{K}_i), x_j(\mathcal{K}_j))$ **then**
  $\mathcal{K}_i^+ := \mathcal{K}_i, \quad \mathcal{K}_j^+ := \mathcal{K}_j$;
  (the heuristic did not find a more balanced assignment and we revert to original one)
  4 - if $\max(x_i(\mathcal{K}_i), x_j(\mathcal{K}_j)) > \hat{x}_{ave,i} + \frac{c_{max}}{\gamma_{min}} + \frac{\varepsilon}{2}$ **then**
    Choose at random a task $\delta \in \mathcal{K}_i \cup \mathcal{K}_j$.
    if $\delta \in \mathcal{K}_i$ **then**
      $\mathcal{K}_i^+ = \mathcal{K}_i \setminus \{\delta\}, \quad \mathcal{K}_j^+ = \mathcal{K}_j \cup \{\delta\}$
    else
      $\mathcal{K}_i^+ = \mathcal{K}_i \cup \{\delta\}, \quad \mathcal{K}_j^+ = \mathcal{K}_j \setminus \{\delta\}$
    (move one random task from one node to the other one)
return Sets $\mathcal{K}_i^+$ and $\mathcal{K}_j^+$.

Proof: Consider a relaxed optimization problem where tasks are infinitely divisible so that each node has the same execution time. Then $\sum_{j \in V_i} c_j = x_{opt}\gamma_i$ for all $i \in V$. Therefore, summing up on all nodes it holds $\sum_{i \in V} \sum_{j \in V_i} c_j = \sum_{i \in V} x_{opt}\gamma_i$, thus $x_{opt} = \frac{\sum_{j=1}^{K} c_j}{\sum_{i=1}^{n} \gamma_i}$. By multiplying and dividing by n we can write equivalently $x_{opt} = \frac{c_{ave}}{\gamma_{ave}}$. □

Inspired by the previous result, we define a parameterized *relaxed target set* of task assignments:

$$\bar{\mathcal{Y}}_\varepsilon = \left\{ Y \;\mid\; F(Y) \leq \frac{c_{ave}}{\gamma_{ave}} + \frac{c_{max}}{\gamma_{min}} + \varepsilon \right\}. \tag{6}$$

It is obvious that $\bar{\mathcal{Y}}_\varepsilon \subseteq \bar{\mathcal{Y}}_{\varepsilon'}$ for $\varepsilon \leq \varepsilon'$.

From Proposition 4.1 derives the following result.

***Corollary** 4.2.* For $\varepsilon = 0$ the relaxed target set is contained in the target set (3), i.e., $\bar{\mathcal{Y}}_0 \subseteq \mathcal{Y}$. ◇

Given a task assignment Y we now introduce a new performance index $J(Y) = (F(Y), n_{\max}(Y))$ consisting of two terms. The first term $F(Y)$ is the network execution time, while the second one $n_{\max}(Y)$ denotes the set of nodes that have maximal execution time given Y. We impose a lexicographic ordering on the performance index, i.e., $J(Y') < J(Y)$ if either $F(Y') < F(Y)$ or $F(Y') = F(Y)$ and $n_{\max}(Y') < n_{\max}(Y)$.

***Proposition** 4.3.* Given a task assignment $Y \notin \bar{\mathcal{Y}}_\varepsilon$ there exists a new assignment Y' with $J(Y') < J(Y)$ that is identical to Y except for the transfer of one task from node i_0 with maximal execution time to another node i_k.

Proof: If $Y \notin \bar{\mathcal{Y}}_\varepsilon$ then $\max_{i \in V} x_i > \frac{c_{ave}}{\gamma_{ave}} + \frac{c_{max}}{\gamma_{min}} + \varepsilon$. This implies that there exists at least one node j such that $x_j < \frac{c_{ave}}{\gamma_{ave}}$. Therefore, if the number of nodes with execution time equal to x_i is greater than one, we can move one task from node i and put it in node j to lower this number by one. If only one node holds the maximum execution time x_i then moving a task from node i to node j lowers the maximum execution time, thus proving the statement. □

Note that in the previous proposition the new configuration Y' may not be reachable from Y in a single gossip iteration because node i_0 and i_k need not be connected by an edge.

Let us first discuss the evolution of local estimation variables $\hat{c}_{ave,i}(t)$ and $\hat{\gamma}_{ave,i}(t)$ initialized, respectively, with the node initial load and with the node speed. Their evolution follows (as we have already mentioned) the *gossip averaging algorithm* in Boyd et al. (2006). Under the stated Assumption 3.1 it has been proved that they asymptotically converge to consensus on the average of the initial values, i.e., for $i = 1, \ldots, n$:

$$\lim_{t\to\infty} \hat{c}_{ave,i}(t) = \frac{1}{n}\sum_{i=1}^{n} \hat{c}_{ave,i}(0) = \frac{1}{n}\sum_{j=1}^{K} c_j = c_{ave},$$
$$\lim_{t\to\infty} \hat{\gamma}_{ave,i}(t) = \frac{1}{n}\sum_{i=1}^{n} \hat{\gamma}_{ave,i}(0) = \frac{1}{n}\sum_{i=1}^{n} \gamma_i = \gamma_{ave}.$$

Note that convergence also has the following monotonicity property: if for a given time t it holds $\max_{i\in\mathcal{V}} |\hat{c}_{ave,i}(t) - c_{ave}| \leq \delta$ and $\max_{i\in\mathcal{V}} |\hat{\gamma}_{ave,i}(t) - \gamma_{ave}| \leq \delta$, then for all $t' > t$ it holds $\max_{i\in\mathcal{V}} |\hat{c}_{ave,i}(t') - c_{ave}| \leq \delta$ and $\max_{i\in\mathcal{V}} |\hat{\gamma}_{ave,i}(t') - \gamma_{ave}| \leq \delta$. On the basis of these known results we can state the following monotonicity property for the variable $\hat{x}_{ave,i} = \hat{c}_{ave,i}/\hat{\gamma}_{ave,i}$.

***Proposition** 4.4.* If at time t $\max_{i\in\mathcal{V}} |\hat{c}_{ave,i}(t) - c_{ave}| \leq \delta$ and $\max_{i\in\mathcal{V}} |\hat{\gamma}_{ave,i}(t) - \gamma_{ave}| \leq \delta$, with $0 < \delta < \gamma_{ave}$ then for all $t' > t$ $\max_{i\in\mathcal{V}} \left|\hat{x}_{ave,i}(t') - \frac{c_{ave}}{\gamma_{ave}}\right| \leq \frac{(\gamma_{ave}+c_{ave})\delta}{\gamma_{ave}(\gamma_{ave}-\delta)}$.

Proof: It can be proved by noticing that

$$\left|\hat{x}_{ave,i}(t') - \frac{c_{ave}}{\gamma_{ave}}\right| \leq \frac{c_{ave}+\delta}{\gamma_{ave}-\delta} - \frac{c_{ave}}{\gamma_{ave}}.$$ □

We can finally state the following results.

***Proposition** 4.5.* Given a task assignment $Y \notin \bar{\mathcal{Y}}_\varepsilon$ under Algorithm 1 if in the remaining evolution $|\hat{x}_{ave,i} - c_{ave}/\gamma_{ave}| \leq \varepsilon/2$ for all $i \in \mathcal{V}$, it is always possible to reach an assignment Y' with $J(Y') < J(Y)$ in a finite number of iterations $\alpha < 2d$, where d is the network diameter.

Proof: We provide only a proof sketch due to space constraints. We note that if an improvement of the objective function is possible, then it eventually occurs in a finite number of steps because nodes with loads exceeding the considered threshold swap tasks randomly. This random swap allows some tasks to execute a random walk in the graph and eventually find a node which allows a reduction in value of the objective function. □

Based on the previous proposition we characterize the following property of set Y.

***Corollary** 4.6.* Given a task assignment $Y \notin \bar{\mathcal{Y}}_\varepsilon$ under Algorithm 1 if in the future evolution $|\hat{x}_{ave,i} - c_{ave}/\gamma_{ave}| \leq \varepsilon/2$ for all $i \in \mathcal{V}$, it is always possible to reach an assignment $Y' \in \bar{\mathcal{Y}}_\varepsilon$ in a finite number of iterations, i.e., the task assignment set is $\bar{\mathcal{Y}}_\varepsilon$ is always reachable. ◇

Finally we prove that set $\bar{\mathcal{Y}}_0$ is invariant if the local estimate variables are sufficiently precise.

***Proposition** 4.7.* Given a task assignment $Y \in \bar{\mathcal{Y}}_0$ assume that in the future evolution holds $|\hat{x}_{ave,i} - c_{ave}/\gamma_{ave}| \leq \varepsilon/2$ for all $i \in \mathcal{V}$. If Y' is the new assignment determined by Algorithm 1 it holds $Y' \in \bar{\mathcal{Y}}_0$.

Proof: To prove the result, it is sufficient to prove that in the execution of the Balancing Rule step 4 is never executed. In fact for any two nodes i and j it holds:

$$\max(x_i, x_j) = F(Y) \leq \frac{c_{ave}}{\gamma_{ave}} + \frac{c_{max}}{\gamma_{min}} \leq \hat{x}_{ave,i} + \frac{c_{max}}{\gamma_{min}} + \frac{\varepsilon}{2}$$

□

The next proposition shows that any improvement of the objective function $F(Y)$ is lower bounded.

***Proposition** 4.8.* Given a task assignment $Y \in \bar{\mathcal{Y}}_0$ let Y' be a new assignment determined by Algorithm 1. If $F(Y') < F(Y)$, then $F(Y') \leq F(Y) - \frac{c_{min}}{\varrho}$, where ϱ is the maximum least common multiplier (lcm) among the pairs γ_i, γ_j for $(i,j) \in E$.

Proof: We omit the proof of this statement due to space constraints. □

We can finally characterize the convergence property of Algorithm 1 as follows.

***Theorem** 4.9.* The task assignment Y updated iteratively through Algorithm 1 with $0 < \varepsilon < \frac{c_{min}}{\varrho}$ converges almost surely in finite time to set $\mathcal{Y}$ defined in (3), i.e.,

$$Pr\left(\exists \tau : Y(\tau') \in \mathcal{Y}), \quad \forall \tau' \geq \tau\right) = 1.$$

Proof: We just provide a sketch of the proof based on the intermediate results previously collected. First, since $\varepsilon < \frac{c_{min}}{\varrho}$, it is possible to show using Proposition 4.8 that $\bar{\mathcal{Y}}_\varepsilon = \bar{\mathcal{Y}}_0$.

Also we observe that given the convergence property of the local estimated variables, we can be sure that there exists a finite time t such that for all $t' > t$ holds $\max_{i\in\mathcal{V}} \left|\hat{x}_{ave,i}(t') - \frac{c_{ave}}{\gamma_{ave}}\right| \leq \frac{\varepsilon}{2}$.

This means that after t, starting from any configuration $\mathcal{Y}$ set $\bar{\mathcal{Y}}_0 = \bar{\mathcal{Y}}_\varepsilon$ is reachable in a finite number of iterations by Corollary 4.6. Also this set is invariant by Proposition 4.7. Finally, from the fact that the target set $\mathcal{Y}$ is contained in $\bar{\mathcal{Y}}_0$, the result follows. □

5. NUMERICAL SIMULATIONS

In this section we corroborate the theoretical characterization of the convergence properties of Algorithm 1 with numerical simulations. First, we compare the proposed algorithm with the algorithm proposed in Franceschelli et al.

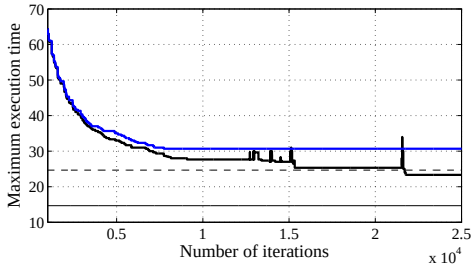

Fig. 1. Comparison of the evolution of the network maximum execution time according to Algorithm 1 (thick black line) and the Discrete Consensus Algorithm in Franceschelli et al. (2010) (blue line) in a line network of 30 nodes. The dashed line represents $F(Y) = \frac{c_{ave}}{\gamma_{ave}} + \frac{c_{max}}{\gamma_{min}}$ while the continuous thin line represents the lower bound to the optimal execution time.

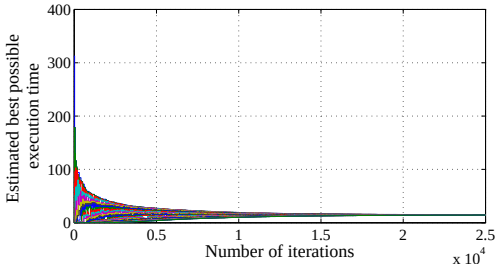

Fig. 2. Evolution of the estimated lower bound on the optimal execution time $F(Y^*)$ computed as $\frac{\hat{c}_{ave,i}}{\hat{\gamma}_{ave,i}}$ by each node.

(2010). We considered a network represented by a line graph composed by 30 nodes, each with an execution speed chosen uniformly at random in the interval $[1, 3]$. We considered a set of $K = 180$ tasks to be distributed among the nodes, each with a cost chosen uniformly at random in the interval $[1, 10]$.

We simulated the Discrete Consensus Algorithm (DCA) in Franceschelli et al. (2010) and the Heterogeneous Discrete Consensus (HDC) algorithm proposed in this paper with the same set of random initial conditions and with the same sequence of random edge selections. In these simulations we chose parameter $\varepsilon = 10^{-3} < \frac{c_{min}}{\varrho} = 0.11$.

In Figure 1 we show a direct comparison between the simulations of the evolution of the maximum execution time during the execution of the DCA and HDC algorithm. It can be seen that the execution time of DCA is non-increasing but since the network does not satisfy the condition of fully connected "swap domains" (Franceschelli et al. (2010)) it can not be guaranteed that the final task assignment is close to the optimal solution. On the contrary, in the chosen example the worst case performance may differ from the optimal value of the execution time by a quantity proportional to the number of nodes. The evolution of the HDC algorithm does not monotonically decrease because to overcome blocking configurations of tasks some of them are moved at random. Once each node estimates sufficiently well the lower bound to the optimal value of the objective function then after a sufficiently long time a task assignment in set $\mathcal{Y}$ is reached and the local interactions stop. It can be seen that Algorithm 1 outperforms the algorithm proposed in Franceschelli et al. (2010).

In Figure 2 we show the evolution of variables $\hat{x}_{ave,i}$, which evolve according to the gossip algorithm presented in Boyd et al. (2006). It can be noticed that despite Algorithm 1 involves quantized and randomized dynamics, its simulated convergence rate does not appear to be significantly different from the simple averaging gossip algorithm in Boyd et al. (2006). Finally, to corroborate our theoretical results we propose a set of numerical simulations to evaluate the expected convergence time of the proposed algorithm. In particular, in Figure 3 it is shown how the average convergence time of 10 simulations varies with respect to the number of nodes in a semi-logarithmic chart in line graphs (cotninuous line) and random graphs (dashed line). To allow fair comparisons we kept the average number of tasks constant and thus selected at each simulation a total number of tasks equal to $K = 6n$. It can be seen that the convergence time grows polynomially with respect to the number of nodes. Random graphs are generated with a probability of edge existence among pairs of nodes equal to $p = \frac{log(n)}{n}$. This probability of edge existence is chosen to generate graphs that with high probability have similar diameter. Comparing the simulations in Figure 3 it can be seen that the convergence time in random graphs is much smaller than in line graphs. A theoretical study of the convergence time will be carried out in future work.

6. CONCLUSIONS

In this paper we proposed a novel algorithm, the Heterogeneous Discrete Consensus (HDC) algorithm, which optimizes with guaranteed performance the execution time of a set of tasks by a network of nodes with heterogeneous execution

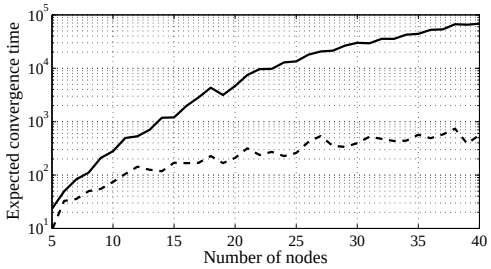

Fig. 3. Expected convergence time for line graphs (continuous line) and random graphs (dashed line) with increasing numbers of nodes.

speed exploiting only asynchronous and pairwise local state updates, i.e., gossip-based. The proposed algorithm extends the state of the art in that it guarantees the achievement of an assignment whose objective function value differs from the optimal one only by a constant function of the maximum task cost and minimum task execution speed. Therefore, the proposed distributed algorithm scales well with network size and is suitable to solve task assignment problems in large networks. We characterized the convergence properties of the algorithm and proved an absolute performance guarantee on the final computed task assignment. We discussed numerical simulations to further validate the proposed algorithm.

Future work will involve a theoretical characterization of the convergence time of the proposed algorithm.

REFERENCES

Babel, L., Kellerer, H., and Kotov, V. (1998). The k-partitioning problem. *Mathematical Methods of Operations Research*, 47(1), 59–82.

Başar, T., Etesami, S., and Olshevsky, A. (2014). Fast convergence of quantized consensus using metropolis chains. In *IEEE 53rd Annual Conference on Decision and Control*, 1330–1334.

Boyd, S., Ghosh, A., Prabhakar, B., and Shah, D. (2006). Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52(6), 2508–2530.

Chopra, S. and Egerstedt, M. (2014). Heterogeneous multi-robot routing. In *American Control Conference*, 5390–5395.

Etesami, S. and Başar, T. (2013). Convergence time for unbiased quantized consensus. In *IEEE 52nd Annual Conference on Decision and Control*, 6190–6195.

Fanti, M., Franceschelli, M., Mangini, A., Pedroncelli, G., and Ukovich, W. (2013). Discrete consensus in networks with constrained capacity. In *IEEE 52nd Annual Conference on Decision and Control*, 2012–2017.

Fanti, M., Mangini, A.M., and Ukovich, W. (2012). A quantized consensus algorithm for distributed task assignment. In *IEEE 51st Annual Conference on Decision and Control*, 2040–2045.

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

Franceschelli, M., Giua, A., and Seatzu, C. (2011). Quantized consensus in Hamiltonian graphs. *Automatica*, 47(11), 2495–2503.

Franceschelli, M., Giua, A., and Seatzu, C. (2015). Fast discrete consensus based on gossip for makespan minimization in networked systems. *Automatica*, 56, 60–69.

Franceschelli, M., Rosa, D., Seatzu, C., and Bullo, F. (2013). Gossip algorithms for heterogeneous multi-vehicle routing problems. *Nonlinear Analysis: Hybrid Systems*, 10(1), 156–174.

Garey, M.R. and Johnson, D.S. (1979). Computers and intractability: a guide to the theory of np-completeness. *WH Freeman & Co., San Francisco*.

Gravelle, E. and Martinez, S. (2014). Quantized distributed load balancing with capacity constraints. In *IEEE 53rd Annual Conference on Decision and Control*, 3866–3871.

He, J., Cheng, P., Shi, L., Chen, J., and Sun, Y. (2014). Time synchronization in wsns: A maximum-value-based consensus approach. *IEEE Transactions on Automatic Control*, 59(3), 660–675.

Johnson, S.M. (1954). Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*, 1(1), 61–68.

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

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

Zhu, M. and Martínez, S. (2011). On the convergence time of asynchronous distributed quantized averaging algorithms. *IEEE Transactions on Automatic Control*, 56, 386–390.