International Journal of Control
Vol. 84, No. 3, March 2011, 476–495

International Journal of Control
Vol. 84, No. 3, March 2011, 476–495

Research Paper

Results in Cooperative Control and Implicit Consensus

Florian Knorn[a*], Martin Corless[b], and Robert N. Shorten[a]

[a]*Hamilton Institute, National University of Ireland Maynooth, Maynooth, Co. Kildare, Ireland*;
[b]*School of Aeronautics & Astronautics, Purdue University, West Lafayette, IN 47907, USA*

(*Received 18 August 2010; final version received 5 February 2011*)

This paper is concerned with a class of decentralised control problems that arise in contemporary applications where agents cooperate to control and regulate a global quantity, are limited in the manner in which they communicate with each other, and are required to reach consensus on some implicit variable (for example, CO_2 emissions). Three algorithms are presented for achieving this goal. The convergence of these algorithms are proven, and their efficacy is demonstrated in simulations.

Keywords: Implicit consensus; cooperative control; consensus algorithms with inputs; distributed averaging; constrained consensus; implicit consensus

1 Introduction

Survey papers such as Olfati-Saber *et al.* (2007), Ren *et al.* (2005) and the wealth of work published in the recent past reflect the large amount of interest in coordination, cooperation and consensus in multi-agent systems. As we shall see in the literature review at the end of this introduction, most of the previous work assumes bi-directional communications (undirected communication graphs) between agents, often does not cater for time changing topologies in the communication network and, in some cases, does not consider dynamics (or only very specific types of linear dynamics for specific applications) involved in state changes. Most importantly, however, while many consensus schemes will correctly produce a consensus, it appears little work has been done to use this consensus value to achieve a common goal. Namely, in many applications, the aggregate behaviour of the system (assumed to depend on *all* the network's states) is of interest as well, and a certain amount of cooperation between the agents in the network is desired in order for the network to achieve a common goal ("cooperative control").

In the present paper we not only try to be free of the above-mentioned pre-assumptions common in the literature, but aim at additionally influencing the consensus value reached in order to meet a global objective. We use an external input into the consensus protocol to regulate the consensus value according to a global performance measure (or *global value*) that depends on the entirety of the network's states. In other words, we would like to develop decentralised protocols and algorithms that allow a network to reach consensus on one quantity of interest, while additionally meeting a separate, global goal. The main contributions of this paper will be three algorithms that achieve this objective, operating in a variety of settings where certain key properties of the network may or may not be known *a priori*, and where different amounts of information and communication is available to the nodes.

*Corresponding author. Email: florian@knorn.org

1.1 *Motivating examples*

Let us give two potential applications as motivating examples for our work. First, consider a network of cars driving around in a city where the city council is trying to implement some form of emission control. The global objective would be that the aggregate emissions of all cars participating in the scheme do not exceed a prescribed level. Fairness dictates that no car should be allowed to pollute more than others, so the cars should adjust their behaviour so that they all produce the same CO_2 emissions (in other words, reach consensus on the emissions). But assuming that the emissions are a direct function of the cars' speed (and that different cars have different efficiency levels, depending on their weight, engine, *etc.*) some cars may be able to drive faster than others for a given level of emission. In order to implement the emission control scheme, the council may place a number of monitoring units around the city to measure the overall emission level and broadcast that (global) information to all the cars in the network, along with the value of the desired or allowable emission level. Clearly, the cars need to cooperate to achieve the desired emission level. For that, we assume that the cars are able to broadcast their own emission level to vehicles in their vicinity. The so-established communication network can then be used to reach a consensus among the cars on a common emission level. Additionally, incorporating the information from the global broadcast, the cars should now be able to adjust their speed (e.g. by using penalties for cars exceeding the resulting speed threshold or confining cars to different lanes with different speeds) so that the resulting emissions match those of other cars in the network, and also so that the overall emissions produced throughout the city reach the admissible level. To highlight some of the particularities of this setting, we note that the topology of the resulting communication network would be constantly changing as the cars drive around and move in and out of range from each other; the communication network will not necessarily be symmetric — some cars may not be able to broadcast as far as others, or some of the transmissions may be lost; and the dependence of emissions on the car speed will be non-linear.[1]

A second example is a wireless sensor network. Assume that a number of battery powered motes are dropped roughly uniformly distributed over a defined area. The (usually identical) motes would be equipped with a battery, a radio, one or more sensors and some kind of processing unit. The objective would be to adjust the broadcast power of their radios so that the network reaches a prescribed level of (algebraic) connectivity, but also so that all nodes last equally long in terms of battery power. The first objective would be important for certain types of algorithms whose rate of convergence depends on the algebraic connectivity of the graph they evolve on, and the second objective guarantees maximum life-time of the network without node failures (due to power shortage). Clearly, the power used by the radio directly influences the time-to-live (TTL) of a node. However, the overall power consumption may vary among nodes depending on their individual workload, and the batteries may also have slightly varying initial charges. Assuming that the radio is the biggest power consumer in each mote, they will be able to influence their TTL by varying the power setting of their radios. But now, depending on the power used, each node can broadcast information to more or fewer nodes in its vicinity. As different nodes will use different power settings, the resulting topology of the communication graph will generally be asymmetric, and changing over time. In this setting, we would like to find a decentralised algorithm that adjusts the node's power setting so that all nodes eventually have equal TTLs, but additionally guaranteeing a certain algebraic connectivity level of the resulting communication network. As reported in Knorn *et al.* (2009), there are decentralised algorithms that can be used to estimate the global connectivity level of a graph, so contrary to the previous example no external broadcast of the global quantity of interest would be needed.

[1]For reasons of simplicity we assume, however, a monotonic relationship between speed and emission levels, as would roughly be the case for cars with continuously variable transmissions (CVT), i. e. modern gear boxes with no "discrete" gears.

1.2 *Our contributions*

These are just two situations where the objective is twofold — reaching consensus on a local utility value that depends on the physical state of each node, combined with an additional global criterion to be met. Naturally, depending on the application, different amounts of information will be available to the network designer, and later the implemented network. For instance, there will be cases where not all the nodes have access to the value of the global quantity, or the relationship between physical state and the utility value may only be known approximately. The scenarios will also be different whether nodes can adjust their state and the utility value directly (i. e. infinitely fast), or only with certain dynamics. In this paper we will present three different decentralised algorithms that are designed to achieve the objectives in three different situations.

(1) Physical state: No dynamics involved, can be changed directly.
Utility functions: Must be perfectly known.
Global value: All nodes must have access.

(2) Physical state: No dynamics involved, can be changed directly.
Utility functions: Lower and upper growth bounds must be known.
Global value: Not all nodes must have access.

(3) Physical state: Dynamics may be involved in state change.
Utility functions: Only approximate knowledge required, can be filtered values.
Global value: Not all nodes must have access.

Additionally, in each case the underlying communication network can be directed and time varying, both the utility functions as well as the global quantity's dependence on the network states can be non-linear, and the state updates in the network must not necessarily be performed synchronously (in other words, asynchronous communications are covered by our approach as well).

1.3 *Previous work*

As mentioned above, much work has recently been done in the field of consensus and coordination in multi-agent systems. For a great introduction into the field and examples of its many, diverse applications see for instance the surveys by Olfati-Saber *et al.*, Olfati-Saber *et al.* (2007), and Ren *et al.*, Ren *et al.* (2005), as well as the collection of references at Reynolds (2001). The general problem of consensus finding can come in many "flavours" depending on the application. These variations include whether the topology of the communication graph remains fixed or changes over time; it is undirected or directed; the agents can manipulate the state on which to reach consensus instantly or only with certain dynamics; if each node's state is scalar or multidimensional; whether there are delays in the information exchange; or if all nodes update their states in a synchronous fashion or on their own pace. While initial work Borkar and Varaiya (1982), Tsitsiklis (1984), Tsitsiklis *et al.* (1986), Reynolds (1987), Vicsek *et al.* (1995) on consensus and coordination was based on bi-directional information exchange between neighbouring nodes (leading to undirected communication graphs) with rigorous convergence proofs given in Jadbabaie *et al.* (2003), this has been extended to include directed communication graphs for instance in Olfati-Saber and Murray (2004), Moreau (2005), Ren and Beard (2005), Fang *et al.* (2005), Beard and Stepanyan (2003). Another generalisation allowed asynchronous consensus protocols so that not all nodes had to perform state updates at the same instant, Olfati-Saber and Murray (2004), Fang *et al.* (2005), Cao *et al.* (2006), Hatano and Mesbahi (2005), Blondel *et al.* (2005). Closely related was the work that also considered changing graph topologies, Jadbabaie *et al.* (2003), Ren and Beard (2005), Tanner *et al.* (2003b), Olfati-Saber (2006), Beard and Stepanyan (2003). Further generalisations of the problem allowed the inclusion of agent dynamics (typically linear, second order systems) in the consensus problem, Tanner *et al.* (2003a,b), Olfati-Saber and Murray (2003), Olfati-Saber (2006), which play an important role in networks of mobile agents that move with

finite dynamics. In some situations the consensus variable may not be directly altered by the nodes, but only implicitly. Such a situation is dealt with in Stanojević and Shorten (2008, 2009).

However, most of these papers only focus on so-called *unconstrained* consensus applications. When the consensus, that the system is to reach, should fulfil external conditions (such as a common heading of a flock of agents, but in a particular direction), three approaches are usually taken, see Beard *et al.* (2001), Lawton *et al.* (2003), Ren and Beard (2004) and citations therein: leader-following (Wang 1989, Singh *et al.* 2000, Mesbahi and Hadaegh 1999, Fax and Murray 2004, Ji *et al.* 2006), virtual structure based (Lewis and Tan 1997, Beard *et al.* 2000, Shi *et al.* 2005) or behaviour based (Balch and Arkin 1998, Anderson and Robbins 1998, Lawton *et al.* 2003, Parker 1998, Chen and Luh 1994, Veloso *et al.* 2000) approaches. The first concept presents a common technique used typically to make formations of autonomous mobile agents follow desired trajectories. However, the problem with these architectures is usually that they not only depend heavily on the leader, but it appears that little discussion of the case where the leader adjusts its state based on feedback of the totality of the states of the network has taken place, and most of the systems dealt with in that context are linear.

In the virtual structure approach, the entire network of agents is treated as a single entity, the virtual structure. The desired behaviour is then assigned to the virtual structure relative to which each member controls its own behaviour.

In the behavioural approach, each agent's behaviour is based on a weighted sum of a number of desired behaviours, such as goal seeking, formation keeping, obstacle and collision avoidance, *etc.* A typical application of these techniques are *rendez-vous* problems with obstacle and collision avoidance, where the agents are to meet in a certain place, but avoid running into obstacles or crashing into each other during the approach.

It is in this third class that our work could be placed, as the desired behaviour of the nodes in our network is also a combination of localised constraints (consensus on the utility values) and global constraints.

1.4 *Structure of this paper*

The remainder of the paper is structured as follows: The next section will introduce the problem setting more concretely and define some necessary notation. This is followed by three algorithms (together with a number of comments and simulations) which form the main contributions of the paper. Finally, after extending our results to the case of asynchronous communications, we will draw some conclusions, discuss open questions and suggest some future directions.

2 Overall setting

2.1 *Problem setting*

Concretely, we consider the following situation. In a network with $n > 1$ agents or "nodes" and a number of directed communication links[1] that may change over time, each node i has a *physical state* (or just "state") that it can change, either directly or indirectly through certain dynamics. Furthermore, associated to each node is also what we call a *utility value*: This value directly depends on the node's physical state and represents some particular quantity of interest that is somehow related to, but usually different from, the physical state. This dependence is given by each node's *utility function*, which is generally assumed to differ between nodes.

Additionally, we define a certain *global value* that depends directly on *all* the nodes' physical states through the *global function*. By suitable means of communication (or decentralised

[1]This could be due to each node broadcasting information about its state at regular intervals, and other nodes in proximity picking up this broadcast — but these nodes do not necessarily have to communicate back.

estimation) either all or just some nodes in the network have access to this global value.[2] Finally, we assume that the agents (locally) share their current utility value through (directed) communication links. This set-up is illustrated in Figure 1.

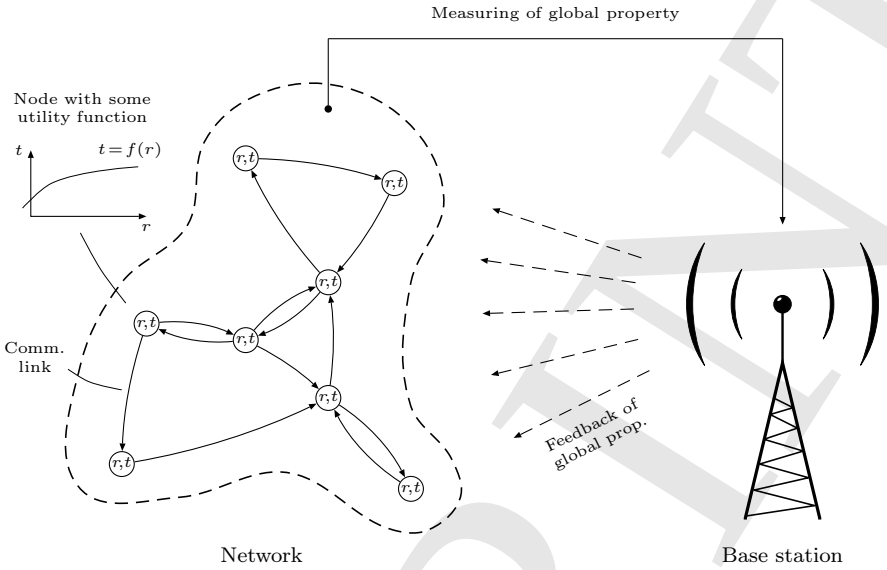


Figure 1. The basic setting.

2.2 *Problem statement*

The objective is now for all nodes in the network to reach consensus on their utility values, while also, jointly, driving the global value to a prescribed, "desired" value. This should be achieved in a fully decentralised way, using simple algorithms that will operate in a variety of settings, including non-linear utility functions that are only known approximately, when not all nodes have access to the global value and when the state updates are not necessarily performed synchronously.

In order to solve this problem, the three results given in this paper present recursive update rules for each node's physical state. These update rules share the same basic idea: Use classic consensus techniques together with an additional term that takes into account the global objective.

2.3 *Notation*

Throughout, we use superscripts in parentheses to denote individual elements in a vector or matrix, and subscripts (usually) denote the time index. Vectors and matrices are typeset in bold letters to improve readability. We use $\mathbf{1}$ resp. $\mathbf{0}$ to denote the all ones resp. zero vector or matrix of appropriate dimension; $\boldsymbol{e}_i$ defines the ith unit vector of appropriate dimension.

The standard situation we will be interested in in this paper is best described using typical notions from graph theory. Let $\mathcal{V} = \{1, \dots, n\}$ be the vertex set of the network and let $\mathcal{E}_k \in \mathcal{V} \times \mathcal{V}$ be the edge set representing the (directed) communication links at time $k = 0, 1, \dots$ between the nodes. The overall directed graph describing the communication structure of the network at time k is the pair $\mathcal{G}_k = (\mathcal{V}, \mathcal{E}_k)$, where we explicitly assume that the communication links may change over time, but not the node set. The set of (in-)neighbours of node i is called $\mathcal{N}_k^{(i)}$; it contains

[2]That is, either the global value can be measured or estimated locally by the nodes, or it will be communicated to them by some form of "external" broadcast (for instance sent from some form of base station that itself can estimate or measure that value).

all the nodes j that can send information to node i, i. e. $\mathcal{N}_k^{(i)} = \{\, j \,|\, (j,i) \in \mathcal{E}_k \,\}$. We say that $\mathcal{G}_k$ is the graph of a non-negative square matrix $\boldsymbol{S}$ if for each $j \neq i$, $s^{(ij)} \neq 0$ if and only if $j \in \mathcal{N}_k^{(i)}$.

The network is called *connected* (in the literature also referred to as *strongly connected*) if there exists a path from every node to every other node in the network, respecting the orientation of the edges. This is the case if and only if the adjacency matrix is *irreducible*, (Horn and Johnson 1985, Th. 6.2.24). We shall either assume in the following that all networks dealt with are strongly connected, or if this is not the case, we use the concept of joint connectivity: A set of graphs is called *jointly strongly connected* if the union of those graphs is strongly connected.[1]

A matrix $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ is called *row-stochastic* if all its entries are non-negative and all its row-sums equal one, in other words $p^{(ij)} \geq 0$ and $\boldsymbol{P}\mathbf{1} = \mathbf{1}$. Similarly, *row sub-stochastic* matrices are defined to be real valued, non-negative matrices whose row-sums are less than or equal to one (but with at least one row-sum strictly less than one). A *strictly row sub-stochastic matrix* is a row sub-stochastic matrix where *all* row-sums are strictly less than one.

Let $r_k^{(i)} \in \mathbb{R}$ be the *physical state* of node i at time k where $k = 0, 1, \ldots$, so that $\boldsymbol{r}_k$ forms the state vector of the network. Node i's *utility value* $t^{(i)} \in \mathbb{R}$ depends on the physical state *via* a continuous and strictly increasing *utility function* $f^{(i)} : \mathbb{R} \to \mathbb{R}$, that is $t_k^{(i)} = f^{(i)}\big(r_k^{(i)}\big)$. Further properties of the utility functions (such as invertibility) will be assumed where necessary. Note that for convenience we will often write the utility functions in vector form, i. e. we use $\boldsymbol{r}_k = \boldsymbol{f}(\boldsymbol{t}_k)$ to represent $t_k^{(i)} = f^{(i)}\big(r_k^{(i)}\big)$ for each i. Furthermore, let $g : \mathbb{R}^n \to \mathbb{R}$ be a *global function* that depends on all the states, which we assume to be element-wise strictly increasing. Desired values are usually denoted with subscript asterisks, so that, for example, the desired value for the global function is always denoted by g_*. Based on this desired value, the solution to the problem thus consists of a vector $\boldsymbol{r}_*$ for which $f^{(i)}\big(r_*^{(i)}\big) = t_*$ for all i and $g(\boldsymbol{r}_*) = g_*$.

2.4 *Growth conditions*

Throughout we shall assume that the utility functions and the global function are continuous and satisfy the following growth conditions: There are positive constants $\underline{d}^{(i)}, \bar{d}^{(i)}, \underline{h}^{(i)}, \bar{h}^{(i)}$ such that for all $i = 1, \ldots, n$

$$\underline{d}^{(i)} \leq \frac{f^{(i)}(r_a) - f^{(i)}(r_b)}{r_a - r_b} \leq \bar{d}^{(i)} \qquad \text{for all } r_a, r_b \in \mathbb{R} \text{ with } r_a \neq r_b \tag{1a}$$

$$\underline{h}^{(i)} \leq \frac{g(\boldsymbol{r} + \Delta r \boldsymbol{e}_i) - g(\boldsymbol{r})}{\Delta r} \leq \bar{h}^{(i)} \qquad \text{for all } \boldsymbol{r} \in \mathbb{R}^n \text{ and all } \Delta r \in \mathbb{R} \text{ with } \Delta r \neq 0 \tag{1b}$$

Loosely speaking, the growth conditions require the utility functions to be strictly increasing with a rate that is bounded away from zero and upper bounded; the global function must also be strictly increasing with a non-zero but also upper bounded rate coordinate-wise.

2.5 *Feasibility*

The conditions on the continuous utility functions guarantee that they are strictly monotone increasing and unbounded; hence each utility function has a continuous inverse $\phi^{(i)}$; thus

$$\phi^{(i)}\big(f^{(i)}(r)\big) = r \qquad \text{and} \qquad f^{(i)}\big(\phi^{(i)}(t)\big) = t \tag{2}$$

[1]The union of a set of graphs on a common vertex set is defined as the graph consisting of that vertex set and whose edge set is the union of the edge sets of the constituent graphs.

for all $t, r \in \mathbb{R}$. We let $\boldsymbol{\phi}(\boldsymbol{t}) = \left[\phi^{(1)}(t^{(1)}), \ldots, \phi^{(n)}(t^{(n)})\right]^{\mathsf{T}}$. Thus $\boldsymbol{\phi}$ is the inverse of $\boldsymbol{f}$. If $\boldsymbol{t} = t_*\mathbf{1}$ then

$$g\big(\boldsymbol{\phi}(\boldsymbol{t})\big) = g(\boldsymbol{\phi}(t_*\mathbf{1})) =: \theta(t_*) \tag{3}$$

In order for our problem to have a solution, it is necessary and sufficient that the equation $\theta(t_*) = g_*$ has a solution for t_* for all g_*, that is, the function θ is indeed invertible.

When a solution for t_* exists, the solution for the state vector is given by $\boldsymbol{r}_* = \boldsymbol{\phi}(t_*\mathbf{1})$. We now show that, as a consequence of the growth conditions, the function θ is invertible. Using the result in Appendix A, we obtain that, for any $t_a, t_b \in \mathbb{R}$,

$$\theta(t_a) - \theta(t_b) = g\big(\boldsymbol{\phi}(t_a\mathbf{1})\big) - g\big(\boldsymbol{\phi}(t_b\mathbf{1})\big) = \sum_{i=1}^{n} c^{(i)}(t_a - t_b) \qquad \text{where} \qquad \frac{\underline{h}^{(i)}}{\bar{d}^{(i)}} \le c^{(i)} \le \frac{\bar{h}^{(i)}}{\underline{d}^{(i)}} \tag{4}$$

From this it follows that θ satisfies the growth condition

$$0 < \underline{c} \le \frac{\theta(t_a) - \theta(t_b)}{t_a - t_b} \le \bar{c} \qquad \text{for all } t_a, t_b \in \mathbb{R} \text{ with } t_a \neq t_b \tag{5}$$

where

$$\underline{c} = \sum_{i=1}^{n} \frac{\underline{h}^{(i)}}{\bar{d}^{(i)}} \qquad \text{and} \qquad \bar{c} = \sum_{i=1}^{n} \frac{\bar{h}^{(i)}}{\underline{d}^{(i)}} \tag{6}$$

Satisfaction of the above growth condition implies that θ is invertible, hence, our problem always has a unique feasible solution.

With all these definitions given, we are now ready to present the main contributions of this paper. At the heart of each of the algorithms presented in the following sections will be a recursive update law according to which the nodes are to adjust their physical state. We would like to emphasise the fact that these update laws indeed represent a decentralised approach — they only require locally available information from neighbouring nodes, and the global term (which is assumed, ideally, to be estimated in a decentralised fashion as well).

3 Algorithm 1: Complete information

The first algorithm provides a control law that will be suitable for situations where the utility functions are invertible functions and are perfectly known to the designer. Also, the value of the global function must also be accessible to all nodes at all times.

The idea then consists of running a classical consensus scheme directly on the utility values with an additional global term added at each node. The actual state update is then calculated from these new utility values using the inverse utility function.

Theorem 3.1 : *Consider the standard situation as described in the Notation section and assume that the utility functions $f^{(i)}$ and the global function g are continuous and satisfy the growth condition. Furthermore, assume that each node, using the inverse of its utility function, can calculate its physical state corresponding to a particular utility value.*

For any initial condition $\boldsymbol{r}_{k=0} = \boldsymbol{r}_0 \in \mathbb{R}^n$*, and any sequence of strongly connected communication graphs, suppose that the nodes iteratively update their physical states based on*

$$t_{k+1}^{(i)} = t_k^{(i)} + \eta \sum_{j \in \mathcal{N}_k^{(i)}} \left(t_k^{(j)} - t_k^{(i)} \right) + \mu \big(g_* - g(\boldsymbol{r}_k) \big) \tag{7}$$

$$r_{k+1}^{(i)} = \phi^{(i)} \big(t_{k+1}^{(i)} \big) \tag{8}$$

for some

$$0 < \eta < \frac{1}{n-1} \qquad \textit{and} \qquad 0 < \mu < \frac{2}{\bar{c}} \tag{9}$$

Then, the physical state vector $\boldsymbol{r}_k$ *converges asymptotically to the vector* $\boldsymbol{r}_*$ *for which* $f^{(i)}\big(r_*^{(i)}\big) = t_*$ *for all* i *and* $g(\boldsymbol{r}_*) = g_*$*.*

Proof The control equation (7) can be expressed as

$$\boldsymbol{t}_{k+1} = \boldsymbol{S}_k \boldsymbol{t}_k + \mu \big[g_* - g\big(\phi(\boldsymbol{t}_k)\big) \big] \mathbf{1} \tag{10}$$

where

$$s_k^{(ij)} = \begin{cases} 1 - \sum_{j \in \mathcal{N}_k^{(i)}} \eta & \text{if } j = i \\ \eta & \text{if } j \in \mathcal{N}_k^{(i)} \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

Clearly $\boldsymbol{S}_k$ is a row-stochastic matrix. The bounds in (9) on η guarantee that, for all i, the elements $s_k^{(ii)}$ and $s_k^{(ij)}$ are positive for $j \in \mathcal{N}_k^{(i)}$. Thus the graph corresponding to $\boldsymbol{S}_k$ is the (strongly) connected communication graph at time step k; this implies that $\boldsymbol{S}_k$ is irreducible. Furthermore, since the main diagonal entries of $\boldsymbol{S}_k$ are all strictly positive, this matrix is *primitive*, (Horn and Johnson 1985, Lem. 8.5.5). Noting that the number of strongly connected graphs on n nodes is finite, it follows that all the $\boldsymbol{S}_k$ matrices are contained in a finite set.

Having shown these properties of the $\boldsymbol{S}_k$ matrices we can now apply Theorem 4.1 of Knorn *et al.* (2009) which guarantees that $t_k^{(i)} - t_k^{(j)} \to 0$ as k grows. This implies that the evolution of each utility $t^{(i)}$ will eventually be described by

$$\bar{t}_{k+1} = \underbrace{\bar{t}_k + \mu \big[g_* - g\big(\boldsymbol{\phi}(\bar{t}_k \mathbf{1})\big) \big]}_{=:\psi(\bar{t}_k)} \tag{12}$$

It is well know that such one-dimensional iterated maps have a unique and globally asymptotically stable fixed point $t_* = \psi(t_*)$ if

$$\left| \frac{\psi(t_a) - \psi(t_b)}{t_a - t_b} \right| \leq \beta < 1 \tag{13}$$

for any $t_a, t_b \in \mathbb{R}$ and $t_a \neq t_b$, Hilborn (1994). So let us determine suitable bounds for μ so that the above inequality is satisfied and the system will indeed converge to a fixed point. Considering any $t_a, t_b \in \mathbb{R}$ with $t_a \neq t_b$, we have

$$\psi(t_a) - \psi(t_b) = t_a - t_b - \mu \big[\theta(t_a) - \theta(t_b) \big] \tag{14}$$

where $\theta(t) = g\big(\boldsymbol{\phi}(t\mathbf{1})\big)$. We have already shown that

$$0 < \underline{c} \leq \frac{\theta(t_a) - \theta(t_b)}{t_a - t_b} \leq \bar{c} \tag{15}$$

from which the following bounds can be established

$$1 - \mu\bar{c} \leq \frac{\psi(t_a) - \psi(t_b)}{t_a - t_b} \leq 1 - \mu\underline{c} < 1 \tag{16}$$

Thus, condition (13) holds if $1 - \mu\bar{c} > -1$, that is, $\mu < 2/\bar{c}$ which is one of the hypotheses of the theorem. Convergence of the one-dimensional system (13) to t_* corresponds to all the utility values of all nodes converging to the same value t_*; since $\psi(t_*) = t_*$ can only be the case if $g\big(\boldsymbol{\phi}(t_*\mathbf{1})\big) = g_*$, we obtain the result that $g(\boldsymbol{r}_*) = g_*$ where $r_*^{(i)} = f^{(i)}(t_*)$, i. e. the original system converges to the desired solution. □

3.1 *Remarks*

Our first result can be seen as a direct generalisation of the work reported in Knorn *et al.* (2009): On the one hand, the algorithm presented here allows the inclusion of utility functions, and on the other the global term can be taken from a class of functions rather than one specific, concrete function stemming from the particular application Knorn *et al.* (2009) was focusing on.

The control law (7) has two components: One aimed at achieving consensus on the utility values, the other at regulating the global value. In order to make this control law easier to understand and implement we suggested a very specific form for the consensus achieving part — it only involves one parameter (the gain η) together with the summation over the differences between a node's own utility value and those of its neighbours. As we stated earlier, the bounds on the gain η are used to make sure that this formulation results in primitive row-stochastic averaging matrices $\boldsymbol{S}_k$ so that we can apply Theorem 4.1 of Knorn *et al.* (2009). Clearly, this specific formulation does not necessarily have to be used. In fact, Theorem 4.2 of the same publication also provides the required contraction property but for a much broader class of averaging schemes.

The theorem requires a very precise setting where perfect knowledge of the utility functions (and their inverses in particular) is required. Additionally, every node needs to have access to the value of the global term which may not be possible in all applications. In that regard, the algorithm and its generalisation in the next section requires weaker assumptions on the setting and thus is relevant to a much larger class of applications.

3.2 *Simulations*

To produce time varying graphs for our simulations, we made use of random geometric graphs with time varying *connection radii* (or *distance parameters*), see Penrose (2003), Santi (2005). A geometric graph is created by distributing nodes over a defined area (typically, the unit square is used), associating a connection radius to each node i and then connecting it to all the nodes j that lay within node i's connection radius (which could be thought of as a "broadcast radius", that is an area within which other nodes j can receive information from node i). In all the examples here, each node's physical state is interpreted as its connection radius,[1] and thus, as the states change so will the network's topology. All examples use graphs with $n = 25$ nodes.

The global and utility functions used for the simulation of Algorithm 1 were of the quadratic type, see Appendix B and Fig. B1. For these functions it is straightforward to determine the

[1]However, if a state is less than 0 or larger than 1.5, it is interpreted as 0 or 1.5 respectively to determine the graph topology.

growth-bounds as required by the theorem and calculate the bounds on the gains μ and η used in the update equation.

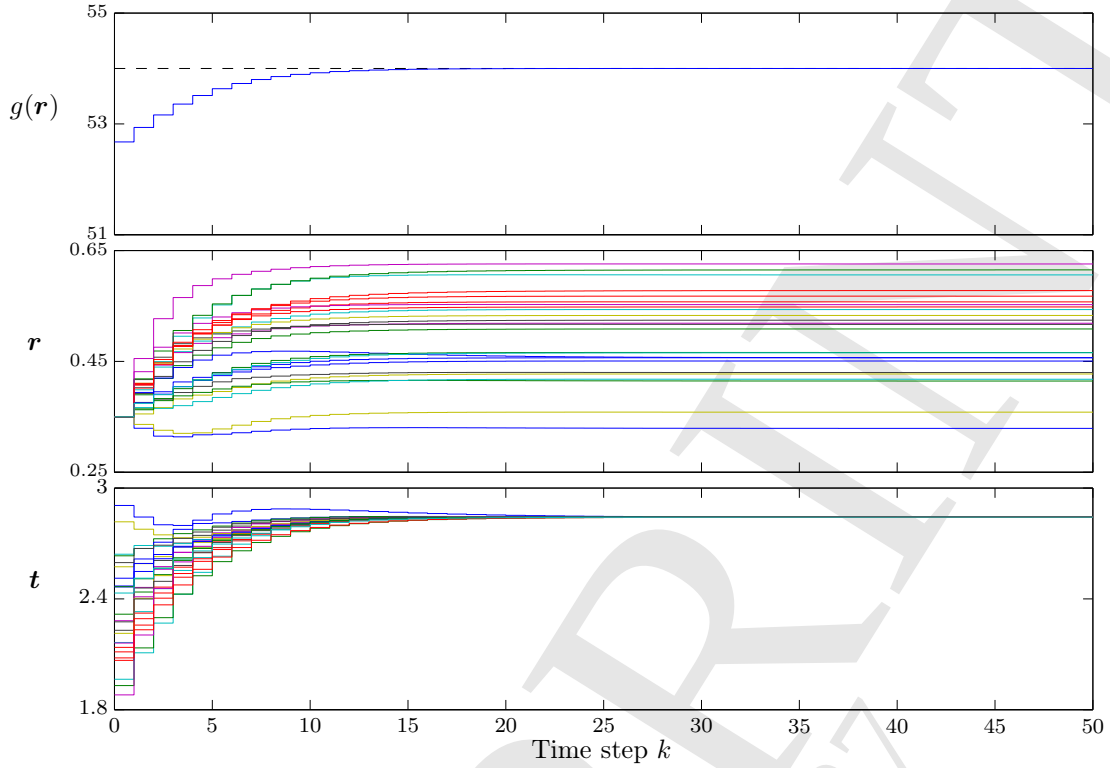


Figure 2. Simulation of Algorithm 1.

Figure 2 shows the results for a desired global value of $g_* = 54$, when the network was initialised with a common physical state of $r_0^{(i)} = 0.35$ for $i = 1, \ldots, 25$. The subplots show the evolution over time of the value of the global term (with the desired value marked by the dashed line), the physical states and the utility values, respectively. As can be seen, the network quickly reaches consensus on the utility values. The general increase in the physical state values is driven by the, initially, lower than desired global value, which then pushes the global value towards its target value. The physical states (interpreted as the connection radii for the underlying communication graph) remained large enough for the network to be strongly connected in each time step.

4 Algorithm 2: Partial information

4.1 *Simple form*

In this section we present a second, more general algorithm for consensus and cooperative control of a global goal. The implementation of this method only requires limited knowledge of the utility functions as well as the global function. Also, as shown later, it can be easily extended to be able to deal with situations where not all nodes have access to the global term and where the communication network is not necessarily strongly connected (that is, asynchronous communications or a certain amount of communication failures can be tolerated).

Theorem 4.1 : *Consider the standard situation as described in the Notation section and assume that the utility functions $f^{(i)}$ and the global function g are continuous and satisfy the growth condition. For any initial condition $\boldsymbol{r}_{k=0} = \boldsymbol{r}_0 \in \mathbb{R}^n$ and any sequence of strongly connected*

communication graphs, suppose that the nodes iteratively update their physical states based on

$$\boxed{r_{k+1}^{(i)} = r_k^{(i)} + \sum_{j\in\mathcal{N}_k^{(i)}} \eta_k^{(ij)}\left(t_k^{(j)} - t_k^{(i)}\right) + \mu_k^{(i)}\sigma_k} \tag{17}$$

where

$$\sigma_k = \begin{cases} g_* - g(\boldsymbol{r}_{k+1-M}) & \text{if } k+1 \text{ is a multiple of } M := n-1 \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

and there exist constants $\varepsilon_1, \varepsilon_2, \underline{\mu}, \bar{\mu} > 0$ *such that*

$$\eta_k^{(ij)} \geq \varepsilon_1 \quad \text{for} \quad j \in \mathcal{N}_k^{(i)}, \qquad \text{and} \qquad \sum_{j\in\mathcal{N}_k^{(i)}} \eta_k^{(ij)} \leq \frac{1}{\bar{d}^{(i)}} - \varepsilon_2 \tag{19}$$

and

$$0 < \underline{\mu} \leq \mu_k^{(i)} \leq \bar{\mu} \tag{20}$$

Then, if $\bar{\mu} > 0$ *is sufficiently small, the state vector* $\boldsymbol{r}_k$ *converges asymptotically to the vector* $\boldsymbol{r}_*$ *for which* $f^{(i)}\big(r_*^{(i)}\big) = t_*$ *for all* i *and* $g(\boldsymbol{r}_*) = g_*$.

Proof Using the growth properties of the utility functions, we have

$$t_{k+1}^{(i)} - t_k^{(i)} = d_k^{(i)}\left(r_{k+1}^{(i)} - r_k^{(i)}\right) \qquad \text{where} \qquad 0 < \underline{d}^{(i)} \leq d_k^{(i)} \leq \bar{d}^{(i)}\,. \tag{21}$$

Hence, multiplication of update law (17) by $d_k^{(i)}$ results in

$$t_{k+1}^{(i)} = t_k^{(i)} + d_k^{(i)} \sum_{j\in\mathcal{N}_k^{(i)}} \eta_k^{(ij)}\left(t^{(j)} - t^{(i)}\right) + d_k^{(i)}\mu_k^{(i)}\sigma_k \tag{22}$$

that is,

$$t_{k+1}^{(i)} = s_k^{(ii)} t_k^{(i)} + \sum_{j\in\mathcal{N}_k^{(i)}} s_k^{(ij)} t^{(j)} + d_k^{(i)}\mu_k^{(i)}\sigma_k \tag{23}$$

where

$$s_k^{(ij)} = \begin{cases} 1 - d_k^{(i)} \sum_{j\in\mathcal{N}_k^{(i)}} \eta_k^{(ij)} & \text{if } j = i \\ d_k^{(i)}\eta_k^{(ij)} & \text{if } j \in \mathcal{N}_k^{(i)} \\ 0 & \text{otherwise} \end{cases} \tag{24}$$

Using the result in Appendix A again, we obtain that

$$g_* - g(\boldsymbol{r}_{k+1-M}) = g\big(\boldsymbol{\phi}(\boldsymbol{t}^*)\big) - g\big(\boldsymbol{\phi}(\boldsymbol{t}_{k+1-M})\big) = \sum_{i=1}^{n} c_k^{(i)}\big(t_*^{(i)} - t_{k+1-M}^{(i)}\big) \tag{25}$$

where

$$0 < \frac{\underline{h}^{(i)}}{\bar{d}^{(i)}} \leq c_k^{(i)} \leq \frac{\bar{h}^{(i)}}{\underline{d}^{(i)}} \tag{26}$$

This allows us to rewrite Equation (22) as

$$\boldsymbol{t}_{k+1} = \begin{cases} \boldsymbol{S}_k \boldsymbol{t}_k - \bar{\mu} \boldsymbol{Q}_k (\boldsymbol{t}_{k+1-M} - \boldsymbol{t}_*) & \text{if } k+1 \text{ is a multiple of } M \\ \boldsymbol{S}_k \boldsymbol{t}_k & \text{otherwise} \end{cases} \tag{27}$$

where, for each $i = 1, \ldots, n$,

$$q_k^{(ij)} = \mu_k^{(i)} d_k^{(i)} c_k^{(j)} / \bar{\mu} \tag{28}$$

Since $\boldsymbol{S}_k \boldsymbol{t}_* = \boldsymbol{S}_k \mathbf{1} t_* = \mathbf{1} t_* = \boldsymbol{t}_*$, we subtract $\boldsymbol{t}_*$ from both sides of (27) and define $\Delta \boldsymbol{t}_k := \boldsymbol{t}_k - \boldsymbol{t}_*$ to get the following reformulation of (17)

$$\Delta \boldsymbol{t}_{k+1} = \begin{cases} \boldsymbol{S}_k \Delta \boldsymbol{t}_k - \bar{\mu} \boldsymbol{Q}_k \Delta \boldsymbol{t}_{k+1-M} & \text{if } k+1 \text{ is a multiple of } M \\ \boldsymbol{S}_k \Delta \boldsymbol{t}_k & \text{otherwise} \end{cases} \tag{29}$$

We can now use this expression to show that $\Delta \boldsymbol{t}_k$ converges to the zero vector — which, of course, implies that the states converge to the desired solution of the problem.

If the system starts at $k = 0$ then, after M iterations, Eq. (29) results in

$$\Delta \boldsymbol{t}_M = \overbrace{\boldsymbol{S}_{n-2} \boldsymbol{S}_{n-3} \ldots \boldsymbol{S}_0}^{=: \bar{\boldsymbol{S}}_0} \Delta \boldsymbol{t}_0 - \bar{\mu} \boldsymbol{Q}_{n-2} \Delta \boldsymbol{t}_0 \tag{30}$$

$$= \underbrace{\left(\bar{\boldsymbol{S}}_0 - \bar{\mu} \boldsymbol{Q}_{n-2} \right)}_{=: \boldsymbol{Z}_0} \Delta \boldsymbol{t}_0 \tag{31}$$

and after another $n-1$ steps

$$\Delta \boldsymbol{t}_{2n-2} = \bar{\boldsymbol{S}}_1 \Delta \boldsymbol{t}_{n-1} - \bar{\mu} \boldsymbol{Q}_{2n-3} \Delta \boldsymbol{t}_{n-1} \tag{32}$$

$$= \boldsymbol{Z}_1 \Delta \boldsymbol{t}_{n-1} \tag{33}$$

In general, for $l = 0, 1, \ldots$, we have

$$\Delta \boldsymbol{t}_{(l+1)M} = \boldsymbol{Z}_l \Delta \boldsymbol{t}_{lM} \tag{34}$$

where

$$\boldsymbol{Z}_l = \bar{\boldsymbol{S}}_l - \bar{\mu} \boldsymbol{Q}_{(l+1)M-1} \qquad \text{and} \qquad \bar{\boldsymbol{S}}_l = \boldsymbol{S}_{(l+1)(n-1)-1} \ldots \boldsymbol{S}_{l(n-1)} \tag{35}$$

The evolution of the $\Delta \boldsymbol{t}_k$ vectors is thus governed by the product of $\boldsymbol{Z}_l$ matrices, at which we must hence take a closer look.

To this end, we first show that the $\bar{\boldsymbol{S}}_l$ matrices are row-stochastic and positive. To do this we first show that the $\boldsymbol{S}_k$ matrices are primitive and thus *fully indecomposable* row-stochastic matrices whose non-zero elements are uniformly bounded away from zero. It is clear from (24)

that $\boldsymbol{S}_k$ is a row-stochastic matrix. Now note that

$$s_k^{(ii)} = 1 - d_k^{(i)} \sum_{j \in \mathcal{N}_k^{(i)}} \eta_k^{(ij)} \geq 1 - \bar{d}^{(i)} \sum_{j \in \mathcal{N}_k^{(i)}} \eta_k^{(ij)} \geq \bar{d}^{(i)} \varepsilon_2 > 0 \tag{36}$$

Also, when $j \in \mathcal{N}_k^{(i)}$ we have

$$s_k^{(ij)} = d_k^{(i)} \eta_k^{(ij)} \geq \underline{d}^{(i)} \varepsilon_1 \geq \underline{d} \varepsilon_1 > 0 \tag{37}$$

where

$$\underline{d} := \min_i \left\{ \underline{d}^{(i)} \right\} \qquad \text{and} \qquad \bar{d} := \max_i \left\{ \bar{d}^{(i)} \right\} \tag{38}$$

The above positive lower bounds on the elements $s_k^{(ij)}$ for $j \in \mathcal{N}_k^{(i)}$ imply that the graph corresponding to $\boldsymbol{S}_k$ is the (strongly) connected communication graph at time step k. Since the diagonal elements of $\boldsymbol{S}_k$ are positive this implies that $\boldsymbol{S}_k$ is primitive, (Horn and Johnson 1985, Lem. 8.5.5). Applying (Brualdi and Ryser 1991, Th. 4.2.3) we can thus note that $\boldsymbol{S}_k$ is fully indecomposable for all k. However, a product of the $n-1$ fully indecomposable $n \times n$ matrices yields a strictly positive matrix, (Hartfiel 2002, Cor. 2.5), and hence the $\bar{\boldsymbol{S}}_l$ are all strictly positive (row-stochastic) matrices.

We now obtain a lower bound on the elements of every $\bar{\boldsymbol{S}}_l$. It follows from (36) and (37) that the non-zero elements $s_k^{(ij)}$ of $\boldsymbol{S}_k$ must satisfy

$$s_k^{(ij)} \geq s_{\min} \qquad \text{where} \qquad s_{\min} := \min \left\{ \underline{d} \varepsilon_1, \, \bar{d}^{(1)} \varepsilon_2, \ldots, \bar{d}^{(n)} \varepsilon_2 \right\} \tag{39}$$

Since each element of $\bar{\boldsymbol{S}}_l$ is the sum of a number of positive terms, where each term is the product of at most M elements of $\boldsymbol{S}_k$ matrices, and $s_{\min} \leq 1$, we must have

$$\bar{s}_k^{(ij)} \geq (s_{\min})^M =: \bar{s}_{\min} \tag{40}$$

for all i, j and k.

Regarding the $\boldsymbol{Q}_k$ matrices, it follows from (28) that, for all k,

$$0 < \underline{d} \left(\frac{\underline{\mu}}{\bar{\mu}} \right) \frac{\underline{h}^{(j)}}{\bar{d}^{(j)}} \leq q_k^{(ij)} \leq \bar{d} \, \frac{\bar{h}^{(j)}}{\underline{d}^{(j)}} \tag{41}$$

or

$$0 < q_{\min} \leq q_k^{(ij)} \leq q_{\max} \tag{42}$$

where

$$q_{\min} := \underline{d} \left(\frac{\underline{\mu}}{\bar{\mu}} \right) \min_i \left\{ \frac{\underline{h}^{(i)}}{\bar{d}^{(i)}} \right\} \qquad \text{and} \qquad q_{\max} := \bar{d} \max_i \left\{ \frac{\bar{h}^{(i)}}{\underline{d}^{(i)}} \right\} \tag{43}$$

Thus, provided

$$0 < \bar{\mu} \leq \frac{\bar{s}_{\min}}{q_{\max}} = \frac{(s_{\min})^M}{q_{\max}} \tag{44}$$

every $\boldsymbol{Z}_l$ matrix will be non-negative; furthermore, since $\bar{\boldsymbol{S}}_l$ is row-stochastic the row sum of every row of $\boldsymbol{Z}_l$ will be bounded above by

$$\kappa := 1 - \bar{\mu} q_{\min} < 1 \tag{45}$$

This implies that the $\boldsymbol{Z}_l$ matrices are strictly row *sub*-stochastic and thus satisfy

$$\|\boldsymbol{Z}_l \Delta \boldsymbol{t}\|_\infty \leq \kappa \|\Delta \boldsymbol{t}\|_\infty \qquad \text{where} \qquad \|\Delta \boldsymbol{t}\|_\infty = \max_i \{|\Delta t^{(i)}|\} \tag{46}$$

It now follows from (34) that

$$\|\Delta \boldsymbol{t}_{(l+1)M}\|_\infty \leq \kappa \|\Delta \boldsymbol{t}_{lM}\|_\infty \tag{47}$$

for all l; hence

$$\|\Delta \boldsymbol{t}_{lM}\| \leq \kappa^l \|\Delta \boldsymbol{t}_0\|_\infty \tag{48}$$

Since each $\boldsymbol{S}_k$ matrix is non-negative and row-stochastic, it satisfies $\|\boldsymbol{S}_k \Delta \boldsymbol{t}\|_\infty \leq \|\Delta \boldsymbol{t}\|_\infty$; hence

$$\|\Delta \boldsymbol{t}_k\|_\infty \leq \kappa^l \|\Delta \boldsymbol{t}_0\|_\infty \qquad \text{when} \qquad lM \leq k \leq (l+1)M - 1\,. \tag{49}$$

Thus $\Delta \boldsymbol{t}_k$ converges to zero as k goes to infinity. This concludes the proof of Theorem 4.1. □

4.2 *Simulations of Algorithm 2*

For the simulations of the algorithm based on Theorem 4.1 we used piecewise linear utility functions, and the global function was selected to be of affine form, see again Appendix B and Fig. B1(b). The parameter bounds were again chosen within certain bounds to produce the growth-bounds as required by the theorem.

As described in Theorem 4.1, the states only incorporate the value of the global term every $n - 1 = 24$ time steps. These updates are marked by the dashed, vertical lines in the second subplot of Figure 3.

While in each time step the averaging scheme pulls together the utility values, each update with the global term pulls them apart again (but brings the global value closer to its desired value). As the targeted value is approached, however, the influence of the global term gets smaller and smaller and eventually the averaging scheme brings a "lasting" consensus to the utility values, at a point where the global term has reached the desired value.

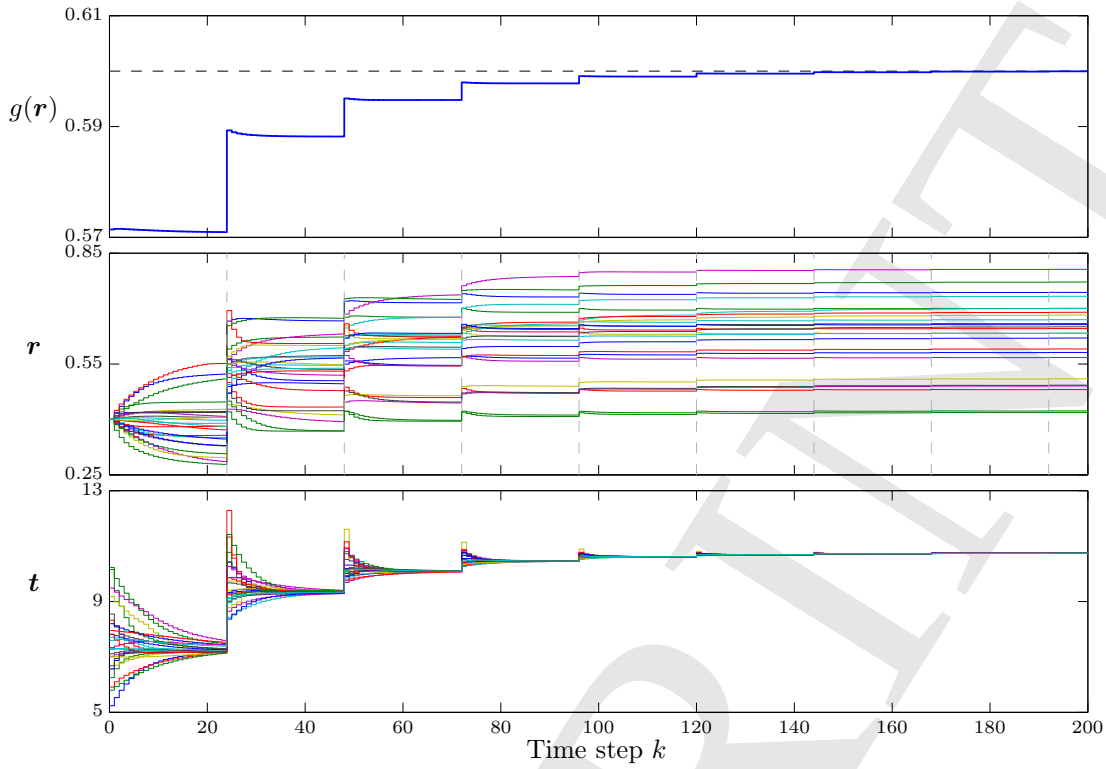


Figure 3. Simulation of Algorithm 2.

4.3 *When not all nodes have access to the global term*

The previous result assumes that all nodes always have access to the global value. In order to make our results more relevant to actual applications, where this assumption may not always be practical or possible to guarantee (for instance in the presence of communication failures), we provide the following corollary to Theorem 4.1. It relaxes the assumption to the more general setting where not all nodes must have access to the global term. In fact, it is sufficient for only one node to have access to the global value. This "special" node could be placed in a strategic place to measure or receive the global value from an external source.

To model this more general scenario, consider any time step k where the global term $g(\boldsymbol{r}_{k+1-M})$ is needed and let $\mathcal{I}_k \subseteq \{1,\dots,n\}$ be the non-empty set of nodes which incorporate the global term in their state update at time k. Then, recalling the original algorithm in (17), the more general algorithm under consideration is modelled by

$$\boxed{r_{k+1}^{(i)} = r_k^{(i)} + \sum_{j\in\mathcal{N}_k^{(i)}} \eta_k^{(ij)}\left(t_k^{(j)} - t_k^{(i)}\right) + \mu_k^{(i)}\sigma_k^{(i)}} \tag{50}$$

where

$$\sigma_k^{(i)} = \begin{cases} g_* - g(\boldsymbol{r}_{k+1-M}) & \text{if } k{+}1 \text{ is a multiple of } M = n{-}1 \text{ and } i \in \mathcal{I}_k \\ 0 & \text{otherwise} \end{cases} \tag{51}$$

for all $k = 0, 1, \dots$. We have now the following result.

Corollary 4.2: *The results of Theorem 4.1 still hold when not all, but at least one node includes the global term in the state update whenever it is required.*

Proof The proof of the corollary is almost identical to that of Theorem 4.1; only some small modifications are needed. Proceeding as in the proof of Theorem 4.1, the algorithm can still be described by (29) where $\boldsymbol{S}_k$ is the same as before and the rows of $\boldsymbol{Q}_k$ corresponding to the nodes which update with the global term at k are the same as before; however the rows of $\boldsymbol{Q}_k$ corresponding to those nodes which cannot incorporate the global term at k are zero. Thus $\boldsymbol{Q}_k$ is not necessarily strictly positive. However, since the assumptions of the corollary guarantee at least one positive row, the $\boldsymbol{Z}_l$ matrices defined in (35) will still be row sub-stochastic but not necessarily *strictly* row sub-stochastic (as they were under the hypotheses of Theorem 4.1).

However, as we show now, products of the form $\boldsymbol{Z}_{l+1}\boldsymbol{Z}_l$ are strictly row sub-stochastic. To this end suppose that $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n\times n}$ are positive, row-stochastic or row sub-stochastic matrices, and at least one row-sum in $\boldsymbol{B}$ is strictly less than one. We show that then the product $\boldsymbol{AB}$ must be strictly row sub-stochastic. Let $\boldsymbol{b} = \boldsymbol{B1}$ and $\boldsymbol{w} = \boldsymbol{AB1}$ be the vectors containing the row-sums of $\boldsymbol{B}$ and the product $\boldsymbol{AB}$ respectively. Since $\boldsymbol{B}$ is row-stochastic or sub-stochastic, we have $b^{(j)} \leq 1$ for all j and, by assumption, there is at least one j_0 for which $b^{(j_0)} < 1$. Since $\boldsymbol{w} = \boldsymbol{AB1} = \boldsymbol{Ab}$, it follows from the definition of the matrix product that for each $i = 1, \ldots, n$, $w^{(i)} = \sum_{j=1}^{n} a^{(ij)} b^{(j)}$. As all elements in $\boldsymbol{A}$ are positive, $\sum_{i=1}^{n} a^{(ij)} \leq 1$, $b^{(j)} \leq 1$ for all j and $b^{(j_0)} < 1$, we must have

$$\begin{aligned} w^{(i)} &= \sum_{j=1}^{n} a^{(ij)} - \sum_{j=1}^{n} a^{(ij)}(1 - b^{(j)}) \\ &\leq 1 - a^{(ij_0)}(1 - b^{(j_0)}) \\ &< 1 \end{aligned} \tag{52}$$

In other words, the product $\boldsymbol{AB}$ is strictly row sub-stochastic.

Using (34) we obtain that for $l = 0, 2, 4, \ldots,$

$$\Delta \boldsymbol{t}_{(l+2)M} = \big(\boldsymbol{Z}_{l+1}\boldsymbol{Z}_l\big)\Delta \boldsymbol{t}_{lM} \tag{53}$$

Since the elements of each matrix $\boldsymbol{Z}_l$ are uniformly bounded away from zero and each matrix has at least one row whose sum is uniformly bounded above by a number less than one, it follows that the matrix product $\boldsymbol{Z}_{l+1}\boldsymbol{Z}_l$ is positive, strictly row sub-stochastic with row sums uniformly bounded above by some $\kappa < 1$. As demonstrated in proof of Theorem 4.1, one can now prove again convergence of $\Delta \boldsymbol{t}_k$ to zero. This concludes the proof of the corollary. □

Simulations of Algorithm 2 where only a small number of nodes have access to the global value are given at the end of Section 6. We shall now move on to our third main result.

5 Algorithm 3: Dynamics and controllers

While the third proposed algorithm shares some similarities with the previous two, it differs conceptually from them in that it is more abstract, modular and allows different nodes to use different controllers to adjust their physical state. In fact, the combination of controller and utility function (the "control loop") may even have a dynamic behaviour, and can be heterogeneous (that is, different nodes may use completely different controller types or utility functions).

The following approach can be interpreted as "decoupling" the adjusting of the physical states (control action) from the (iterative) calculation of "desired utility values", designed to converge to the actual solution of the problem. As is the case with the previous two algorithms, this algorithm is also intended to be implemented in a fully decentralised way.

Concretely, we envisage the following structure: First, in a distributed averaging step the current utility values are averaged using some distributed averaging scheme. To this, if k is

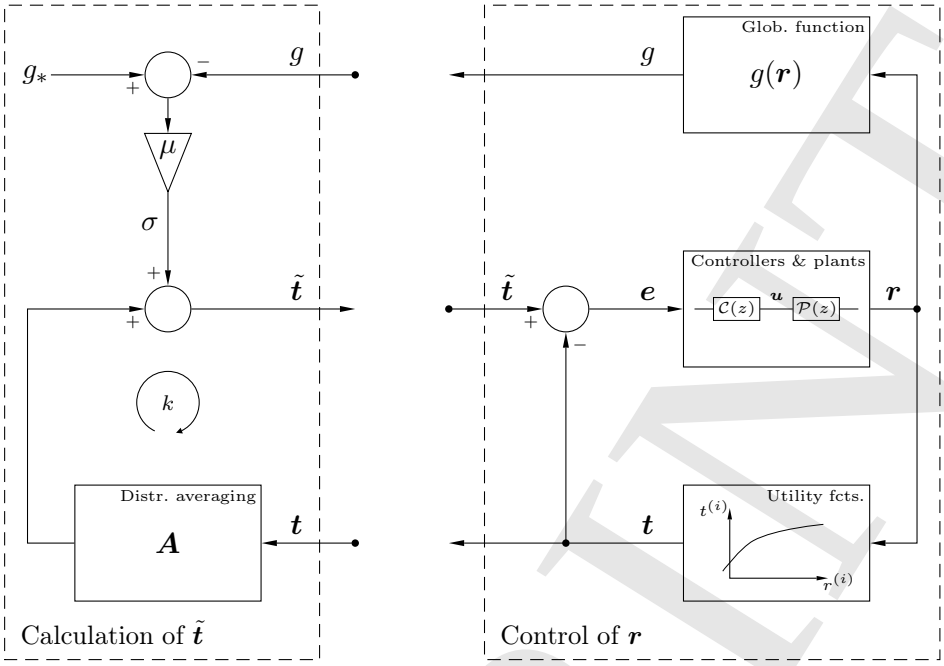


Figure 4. Illustration of the interplay of calculation of the target utility values $\tilde{\boldsymbol{t}}$ and the control action to adjust the physical states $\boldsymbol{r}$ accordingly.

a multiple of $M = n - 1$, a term $\mu\sigma_k^{(i)}$ which is proportional to the error between desired global value and actual global value is added. This yields the *target utility values* $\tilde{t}^{(i)}$. The calculation of the target values thus has the form

$$\tilde{t}_{k+1}^{(i)} = a_k^{(ii)} t_k^{(i)} + \sum_{j \in \mathcal{N}_k^{(i)}} a_k^{(ij)} t_k^{(i)} + \mu\sigma_k^{(i)} \tag{54}$$

where $\boldsymbol{A}_k = \left(a^{(ij)}\right)$ represents the distributed averaging scheme, $\sigma_k^{(i)}$ is as defined in (51) for Algorithm 2 and $\mu > 0$ is a sufficiently small gain which is to be determined.

Each node then passes its target utility value to its controller, which (over a certain finite time span) manipulates the physical state $r^{(i)}$ in order to drive the node's utility value toward its target value. After that control action, new target values will be calculated based on the resulting new utility values as well as the value of the global function, and so on. This interplay of calculating the target values and then adjusting the states accordingly is shown in Figure 4.

To leave this third approach as modular as possible, we will not specify any specific averaging scheme or controller type. All that will be required for convergence is that it must be possible to express the averaging scheme as multiplication by row-stochastic matrices with non-zero entries uniformly bounded away from zero, and that the controllers reduce the control error to within some specified range.

As in the previous two algorithms the questions is again: Does there exist a gain μ such that the resulting system is stable and converges to the desired solution?

Theorem 5.1 : *Consider the standard situation as described in the Notation section and assume that the utility functions $f^{(i)}$ and the global function g are continuous and satisfy the growth condition. Suppose that the communication structure of the network allows it to run a distributed averaging scheme on the utility values. Furthermore, each node is assumed to use a controller that is designed to adjust the node's physical state in such a way as to drive its utility value towards the target utility value.*

If the averaging scheme can be represented in each time step as a non-negative row-stochastic matrix $\boldsymbol{A}_k$ whose graph is strongly connected and with all non-zero elements uniformly bounded away from zero by some $\gamma > 0$, and if the controllers guarantee

$$\underline{\alpha}_i\big(t_k^{(i)} - \tilde{t}_{k+1}^{(i)}\big) \;\le\; t_{k+1}^{(i)} - \tilde{t}_{k+1}^{(i)} \;\le\; \bar{\alpha}_i\big(t_k^{(i)} - \tilde{t}_{k+1}^{(i)}\big) \tag{55}$$

in each control phase for some constant $\underline{\alpha}^{(i)}, \bar{\alpha}^{(i)}$ which satisfy

$$-\gamma/(1-\gamma) < \underline{\alpha}^{(i)} \le \bar{\alpha}^{(i)} < 1 \tag{56}$$

then a positive gain μ can be found for any initial condition $\boldsymbol{r}_{k=0} = \boldsymbol{r}_0$, the system converges so that $\boldsymbol{t}_k \to t_*\mathbf{1}$ and $g(\boldsymbol{r}_k) \to g_*$.

Proof We will show that any algorithm under consideration here can be reduced to one considered in Corollary 4.2. Satisfaction of the inequalities in (55) is equivalent to writing

$$t_{k+1}^{(i)} - \tilde{t}_{k+1}^{(i)} = \beta_k^{(i)}\big(t_k^{(i)} - \tilde{t}_{k+1}^{(i)}\big) \qquad \text{with} \qquad \underline{\alpha}^{(i)} \le \beta_k^{(i)} \le \bar{\alpha}^{(i)} \tag{57}$$

that is,

$$t_{k+1}^{(i)} = t_k^{(i)} + \big(1 - \beta_k^{(i)}\big)(\tilde{t}_{k+1}^{(i)} - t_k^{(i)}) \tag{58}$$

Recall that

$$\tilde{t}_{k+1}^{(i)} = a_k^{(ii)} t_k^{(i)} + \sum_{j \in \mathcal{N}_k^{(i)}} a_k^{(ij)} t_k^{(j)} + \mu\sigma_k^{(i)} \tag{59}$$

Since $\boldsymbol{A}_k$ is row-stochastic, we must have $a_k^{(ii)} = 1 - \sum_{j\in\mathcal{N}_k^{(i)}} a_k^{(ij)}$ and hence

$$\tilde{t}_{k+1}^{(i)} - t_k^{(i)} = \sum_{j \in \mathcal{N}_k^{(i)}} a_k^{(ij)}(t_k^{(j)} - t_k^{(i)}) + \mu\sigma_k^{(i)} \tag{60}$$

Recalling (58) now results in

$$t_{k+1}^{(i)} = t_k^{(i)} + \sum_{j \in \mathcal{N}_k^{(i)}} \eta_k^{(ij)}(t_k^{(j)} - t_k^{(i)}) + \mu_k^{(i)}\sigma_k^{(i)} \tag{61}$$

where

$$\eta_k^{(ij)} = (1 - \beta_k^{(i)})a_k^{(ij)} \qquad \text{and} \qquad \mu_k^{(i)} = (1 - \beta_k^{(i)})\mu \tag{62}$$

Thus the algorithm is an example of those considered in considered in Corollary 4.2. We now show that the hypotheses of Corollary 4.2 hold. First note that $\eta_k^{(ij)} \ge (1 - \bar{\alpha}^{(i)})\gamma > 0$; hence

$$\eta_k^{(ij)} \ge \varepsilon_1 \qquad \text{where} \qquad \varepsilon_1 = \gamma \min_i \Big\{(1 - \bar{\alpha}^{(i)})\Big\} > 0 \tag{63}$$

We also note that

$$\sum_{j\in\mathcal{N}_k^{(i)}} \eta_k^{(ij)} = \big(1-\beta_k^{(i)}\big) \sum_{j\in\mathcal{N}_k^{(i)}} a_k^{(ij)} \tag{64}$$

$$= \underbrace{\big(1-\beta_k^{(i)}\big)}_{\leq 1-\underline{\alpha}^{(i)}} \underbrace{\big(1-a_k^{(ii)}\big)}_{\leq 1-\gamma} \tag{65}$$

$$\leq 1-\big[\gamma+\underline{\alpha}^{(i)}(1-\gamma)\big] \tag{66}$$

Since $\gamma+\underline{\alpha}^{(i)}(1-\gamma)>0$ for $j\in\mathcal{N}_k^{(i)}$, we obtain the desired result that

$$\sum_{j\in\mathcal{N}_k^{(i)}} \eta_k^{(ij)} \leq 1-\varepsilon_2 \qquad \text{where} \qquad \varepsilon_2=\gamma+(1-\gamma)\min_i\big\{\underline{\alpha}^{(i)}\big\}>0 \tag{67}$$

We also obtain that

$$0<\underline{\mu}\leq\mu_k^{(i)}\leq\bar{\mu} \tag{68}$$

where

$$\underline{\mu}=\mu\min_i\Big\{1-\bar{\alpha}^{(i)}\Big\} \qquad \text{and} \qquad \bar{\mu}=\mu\max_i\Big\{1-\underline{\alpha}^{(i)}\Big\} \tag{69}$$

So, clearly, $\bar{\mu}$ can be made sufficiently small by choosing μ sufficiently small. Application of Corollary 4.2 concludes the proof of Theorem 5.1. □

Comment: It is easy to see that the lower bound in (56) is automatically fulfilled if the controllers are designed to produce no overshoot.[1]

5.1 *Simulations of Algorithm 3*

The set-up used for our simulations of the third algorithm was the following. In a network on $n=10$ nodes the global functions were again of affine type (as for the simulations of Algorithm 1), the utility functions, in turn, were of quadratic type (as for Algorithm 2).

The averaging scheme in this example was based on random, strongly connected row-stochastic matrices with non-zero entries uniformly bounded below by $\gamma=0.02$.

The controllers used were discrete-time implementations of PID and PI controllers,[2] randomly assigned to nodes, Visioli (2006). For both controller types the parameters were tuned as to guarantee that the resulting closed loop system would not produce any overshoot. The gains in the PI controllers were intentionally reduced somewhat in order to produce a slightly slower step response and increase the heterogeneity between the controllers.

As for the plants (that is, the physical state updates) we chose first order low-pass filters (see for instance Oppenheim *et al.* (1996)) with randomly chosen smoothing parameter $\zeta\in[0.55\,,\,0.85]$ to simulate a system were the physical state cannot be changed instantly.[3]

[1]By "no overshooting" we mean that during each control phase the utility values never exceed the target values, that is if for instance $t_k^{(i)}<\tilde{t}_{k+1}^{(i)}$ then the utility value during that control phase will always be less than or equal to $\tilde{t}_{k+1}^{(i)}$.

[2]For easier implementation, we used the "velocity formulation", that is the output of each controller is calculated recursively with: $u_k^{(i)}=u_{k-1}^{(i)}+k_{\mathrm{p}}\big(e_k^{(i)}-e_{k-1}^{(i)}\big)+k_{\mathrm{i}}\,e_k^{(i)}+k_{\mathrm{d}}\big(e_k^{(i)}-2e_{k-1}^{(i)}+e_{k-2}^{(i)}\big)$. For the PID-controllers, the parameters were set to $k_{\mathrm{p}}=0.10$, $k_{\mathrm{i}}=0.09$ and $k_{\mathrm{d}}=0.03$; for the PI-controllers in turn, $k_{\mathrm{p}}=0.02$, $k_{\mathrm{i}}=0.05$ and $k_{\mathrm{d}}=0$.

[3]Specifically, the new states were calculated as $r_{k+1}^{(i)}=\zeta^{(i)}u_k^{(i)}+(1-\zeta^{(i)})r_k^{(i)}$.

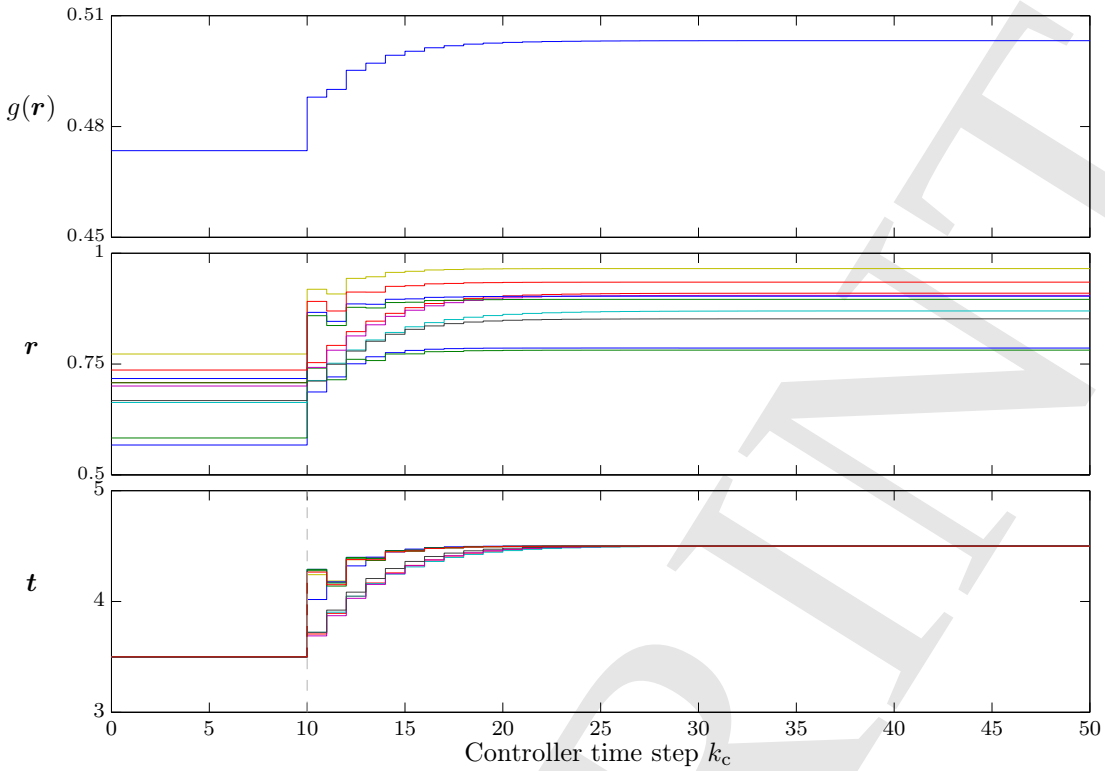


Figure 5. Step-response of the closed loop control part in the simulation of Algorithm 3.

To illustrate the behaviour of the resulting controller-plant combination (together with the non-linear utility functions), a step response of the closed loop system is shown in Fig. 5: The system was initialised with a physical state distribution such that all the utility values would be equal. At $k_c = 10$ the target utility values were then set to $\tilde{t}^{(i)} = 4.5$. While the first two subplots showing the global value and physical states are not of particular interest here, the third subplot clearly reveals the two "groups" of nodes — those with the slower PI controllers and those with the faster PID controllers. At $k_c = 30$ (that is, after 20 control iterations), the error between actual utility value and target value relative to the initial value is less than 0.1% for each node.

While this observation does not guarantee that the control error is less than 0.1% at the end of *every* control phase (since the system is not necessarily in steady-state at the beginning of each control phase) it is still reasonable to assume that the error is reduced sufficiently in order to guarantee the bounds (56).

This closed loop based on 20 control iterations was then also used in the actual simulation of a system operating according to Algorithm 3, shown in Fig. 6.

The dashed vertical lines in the third subplot indicate each time a new target utility value was calculated. The global term was incorporated every $(n-1)\cdot 20 = 180$ time steps. Again, consensus is reached on the utility values and the global term reaches its target value of $g_* = 0.44$ as desired.

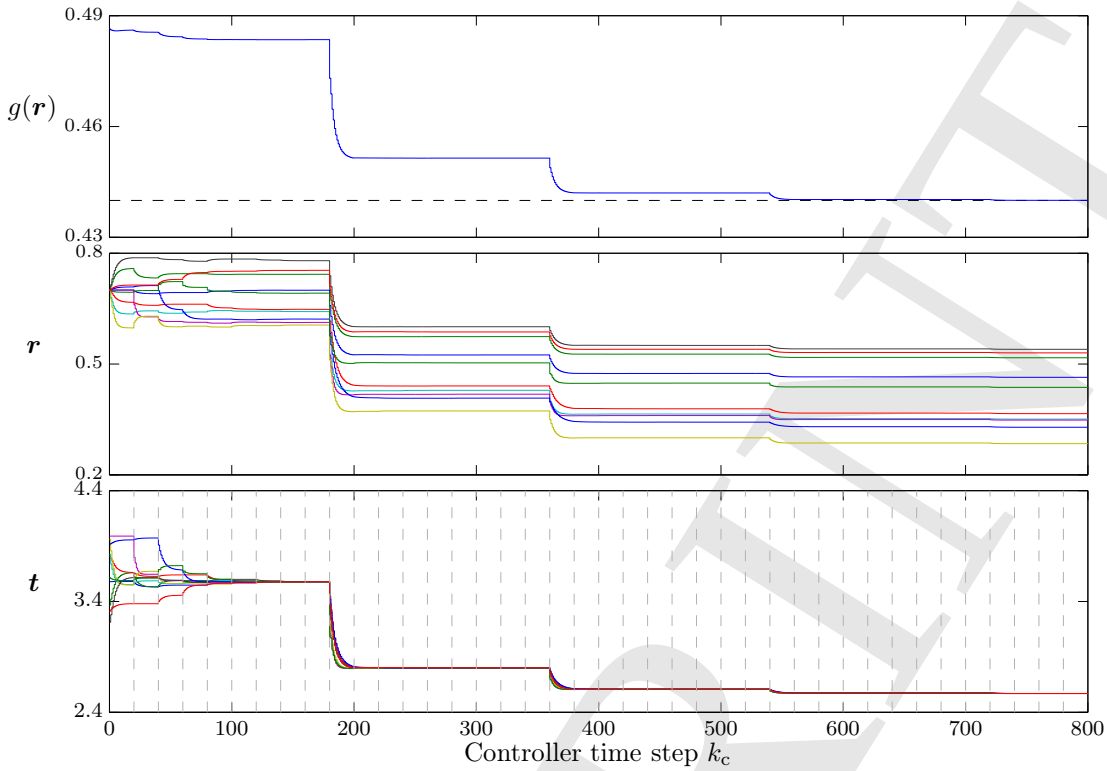


Figure 6. Simulation of Algorithm 3.

6 Extension to asynchronous networks

Until now we have only considered networks where the state updates are all performed in a synchronised fashion. That is, for a given time step k, the nodes first exchanged all the relevant state information with each other, and then, jointly, performed the update based on the state information at time k to reach the new state value at $k+1$. However, this perfectly synchronised way of performing the updates may not always be easy to implement, or even guarantee at all.

In that regard, we now extend our above results to asynchronous communications and state updates by no longer requiring the communication graphs representing the information flow in the network to be strongly connected in each time step (as above), but rather only *jointly* strongly connected over time, with a fixed and constant time horizon $m \geq 1$. In other words, it is only required that the union of any m consecutive graphs taken from that sequence must yield a strongly connected graph. That way, the communication between nodes can be "staggered out", with nodes updating their state right after they have received information from a neighbour, rather than having to wait until they have received the states from all their neighbours and until all the other nodes are also "ready" to perform the (synchronised) update.

In each of our three results above, the update equations (or their transformed versions in the proofs) contain a consensus term based on row-stochastic and primitive matrices. In case of asynchronous updates, these matrices would also be row-stochastic, but not necessarily primitive. Rather, they would contain a number of rows that only have a 1 in the main diagonal entry and 0 everywhere else (corresponding to nodes that have not received any state information from any other nodes).

6.1 *Asynchronous version of Algorithm 1*

Corollary 6.1: *The results of Theorem 3.1 still hold if the sequence of communication graphs is jointly strongly connected over some finite and constant time horizon $m \geq 1$.*

Proof Recall that the algorithm can be described by

$$\boldsymbol{t}_{k+1} = \boldsymbol{S}_k \boldsymbol{t}_k + \mu\big[g^* - g(\boldsymbol{r}_k)\big]\mathbf{1} \tag{70}$$

The proof of Theorem 3.1 relies on a convergence result given in Knorn *et al.* (2009), in particular Theorem 4.1 from that reference. The proof of the result in Knorn *et al.* (2009) involves transforming (70) into

$$\boldsymbol{y}_{k+1} = \boldsymbol{S}_k \boldsymbol{y}_k \tag{71}$$

where $\boldsymbol{y}_k$ has the form $\boldsymbol{y}_k = \boldsymbol{t}_k - \hat{t}_k\mathbf{1}$ with $\hat{t}_k = \sum_{i=0}^{k-1} \mu\big[g^* - g(\boldsymbol{r}_i)\big]$, and then applying (Hartfiel 1998, Th. 1.9) (for which primitivity of the $\boldsymbol{S}_k$ is needed) to show that the elements in the state vector approach each other over time and eventually become all equal. In the case of only jointly strongly connected graphs we need to interpret the product of the $\boldsymbol{S}_k$ matrices as *blocks* of m matrices multiplied together, since these "sub-products" yield primitive, row-stochastic matrices (as the main diagonal entries in each matrix $\boldsymbol{S}_k$ are strictly positive). Additionally, since there are only finitely many possible graph topologies on n nodes, there can only be finitely many different m-blocks of $\boldsymbol{S}_k$ matrices, which implies a uniform, non-zero lower bound on the non-zero matrix elements in all these m-blocks. Both properties allow us again to apply Theorem 1.9 from Hartfiel (1998) to (71) to show that the entries in $\boldsymbol{y}_k$ approach each other over time and eventually all become equal, which ultimately implies that the original system (10) converges to a scalar system. The rest of the proof then follows again the lines of the proof of Theorem 3.1. □

6.2 *Asynchronous version of Algorithms 2 and 3*

Corollary 6.2: *The results of Theorem 4.1 and Corollary 4.2 still hold if the sequence of communication graphs is jointly strongly connected over some finite and constant time horizon $m \geq 1$, provided $M = n - 1$ is replaced with $M = m(n-1)$.*

Proof Only a small modification to the proof of Theorem 4.1 and Corollary 4.2 is needed to show the above result. For $\tilde{k} = 0, 1, \ldots,$ let

$$\tilde{\boldsymbol{S}}_{\tilde{k}} = \boldsymbol{S}_{(m+1)\tilde{k}-1} \cdots \boldsymbol{S}_{\tilde{k}m} \tag{72}$$

Since all the $\boldsymbol{S}_k$ matrices are non-negative row-stochastic matrices with strictly positive diagonal elements, each matrix $\tilde{\boldsymbol{S}}_{\tilde{k}}$ is row-stochastic, has positive diagonal elements and its graph corresponds to the collection of communication graphs from time step $\tilde{k}m$ to $(m+1)\tilde{k} - 1$. As any collection of m consecutive graphs is assumed to be jointly strongly connected, it follows that $\tilde{\boldsymbol{S}}_{\tilde{k}}$ is irreducible, and since it has positive diagonal elements, it is primitive and thus fully indecomposable. The algorithm under consideration still satisfies (34) where

$$\bar{\boldsymbol{S}}_l = \boldsymbol{S}_{(l+1)M-1} \cdots \boldsymbol{S}_{lM}\,; \tag{73}$$

However, here $M = m(n-1)$. Thus,

$$\bar{\boldsymbol{S}}_l = \tilde{\boldsymbol{S}}_{(l+1)(n-1)-1} \cdots \tilde{\boldsymbol{S}}_{l(n-1)} \tag{74}$$

Having established the above properties of the $\tilde{\boldsymbol{S}}_{\tilde{k}}$ matrices, the remainder of the proof follows Theorem 4.1 or Corollary 4.2. □

Comment: The generalised forms of Algorithm 4.1 given by Corollaries 4.2 and 6.2 are designed to tolerate certain communication problems. In the case of Corollary 4.2 this robustness

is achieved at the cost of very small gains $\mu^{(i)}$ on the global term; see (44). It is not difficult to see that smaller gains produce slower convergence; see (45) and (49). However, various simulations using sufficiently "general" graphs (rather than pathological cases like the directed n-cycle) have shown that those gains can, in fact, be set significantly larger than required by the theoretical results above, which suggests that these bounds are loose and may be improved on.

Our third algorithm can also be modified to accommodate for asynchronous communications in the same manner as described above; simply let $M = m(n-1)$.

6.3 *Simulations of Algorithm 2 with asynchronous communications and limited access to global term*

The communication graphs in the previous two simulations were, by design, all strongly connected. For the simulations of the modification of the second algorithm as presented in the corollary, we also used state-dependant disc graphs, but randomly removed, in each time step, a number of edges in order to deliberately disconnect the graphs. The amount of edges removed (in average 75% of the edges), however, was experimentally chosen in order to guarantee (almost always) that every set of $m = 3$ consecutive graphs would form a jointly connected graph, as required by the corollary. Thus, the updates using the global term were performed only every $3(n-1) = 72$ time steps.

Additionally, we also randomly prevented nodes from accessing the value of the global term (in average, only 25% of the nodes were allowed to use the global term during at each global term update step).

The results from the simulation under these harder conditions are shown in Fig. 7 and closely resemble that of the previous case. Due to the less frequent updates, however, convergence to the desired global value takes much longer but is achieved nonetheless.

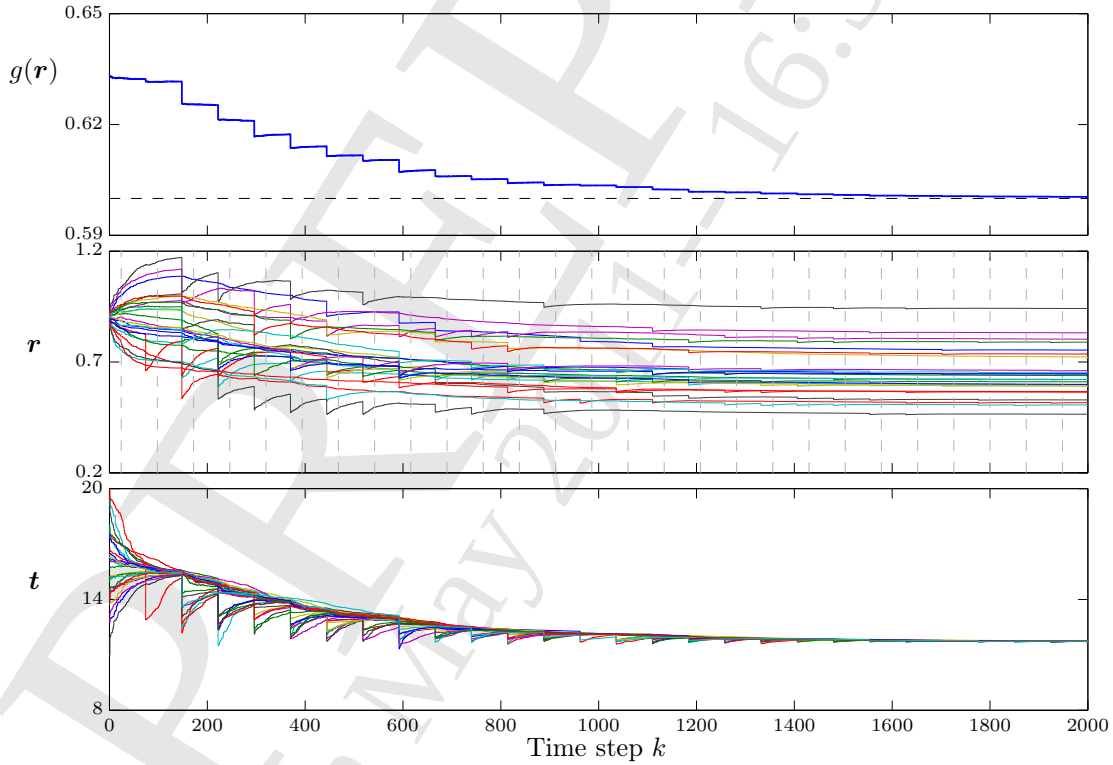


Figure 7. Simulation of Algorithm 2 (corollary).

7 Conclusion

As shown in the literature review, consensus problems have attracted a large amount of attention in recent years. The present paper's contribution in that area are three different fully decentralised cooperative control algorithms that not only allow a network to reach consensus either directly or indirectly (that is, with or without utility functions involved), but also enable the nodes in the network to cooperate and achieve a global, common goal that depends on all the aggregate behaviour of the network.

Our first result concerned the well-controlled case where the utility functions and their inverses are perfectly known *a priori* as the nodes need the inverse utility functions to calculate the state updates. We noted that the algorithm presents a generalisation of the work reported in Knorn *et al.* (2009) through the addition of a more general global term and the introduction of utility functions.

The second contribution consisted of an algorithm that is less demanding on the problem setting. All that needed to be known are upper and lower bounds on the growth rates of the global- and utility functions, but not the functions themselves. Also, through Corollary 4.2, we allowed for an even broader class of applications where not all nodes need to have access to the global value.

Our third piece of work took a somewhat different approach. The idea consisted of decoupling the adjusting of the physical state from the iterative calculation of desired values for the utility values. This enabled us to cater for networks where the state cannot change instantly, where only filtered versions of the state are available, but, probably most importantly, where different nodes may have completely different dynamics and controllers. The key property required for convergence in these networks was that the controllers must be designed so that they drive the physical states / utility values (in finite time) to within a certain range of the calculated target utility values.

Each of the three algorithms was accompanied by simulation results that demonstrated the effectiveness of our approach, and they were then extended to the case of asynchronous communications and state updates.

While our results are certainly promising, a number of open questions remain and should be the subject of further investigations. For instance, the gain μ in the second and third algorithm may become very small in larger networks, and there is much experimental evidence that the bounds presented here tend to be rather conservative. This can be explained, in part, by the fact that for sufficiently connected graphs (and not pathological worst-case scenarios such as, for example, directed n-cycles) significantly less than $n-1$ multiplications in (30) would be required to produce strictly positive $\bar{\boldsymbol{S}}$ matrices — which in turn means that the corresponding $s_{\min}$ value in (40) would be much larger and ultimately allows μ to be increased. One possible future extension of our work that accounts for unlikely topological effects is *via* a stochastic formulation of this problem. Here, expected quantities are controlled rather than deterministic ones.

Also, in the second and third algorithm, the nodes incorporate the global term in their state updates term every M steps. We will show in that in fact nodes can include the global term in *every* time step, which should speed up convergence. Notions of "eventually positive" matrices are required in this context.

A third important open question concerns intermittent feedback of the global quantity of interest. Thus, the graph may operate either in an open- or closed loop setting. This immediately draws parallels to classical Lur'e problems, see for instance Khalil (1992). Future work will involve investigating whether such a formulation will indeed lead to useful insights.

Other open questions in a problem setting as encountered here concern the effect of communication delays, quantisation effects when using finite precision arithmetics (for instance when implementing our algorithms on digital processors without floating point precision) or the effect of nodes joining or leaving the network. These issues may possibly be addressed using ideas from

previous (unconstrained consensus) literature such as Kashyap *et al.* (2007), Nedić and Ozdaglar (2010).

Also, the present work only considers a single physical state and single utility value associated with each node; in a more general setting, nodes could have two or more states associated with them. This leads to a MIMO-like formulation of our problem.

Acknowledgements

This work was supported by *Science Foundation Ireland* PI Award 07/IN.1/1901.

References

Anderson, M.R., and Robbins, A.C. (1998), "Formation flight as a cooperative game," in *Proceedings of the AIAA Guidance, Navigation, and Control Conferences*, Boston, MA, USA, August, Vol. 1, pp. 244–251.

Balch, T., and Arkin, R.C. (1998), "Behavior-based formation control for multirobot teams," *IEEE Transactions on Robotics and Automation*, 14, 926–939.

Beard, R.W., Beard, A.W., Lawton, J.R.T., and Hadaegh, F.Y. (2001), "A Coordination Architecture for Spacecraft Formation Control," *IEEE Transactions on Control Systems Technology*, 9, 777–790.

Beard, R.W., Lawton, J.R.T., and Hadaegh, F.Y. (2000), "A feedback architecture for formation control," in *ACC'00: Proceedings of the 2000 American Control Conference*, Chicago, IL, USA, June, Vol. 6, pp. 4087–4091.

Beard, R.W., and Stepanyan, V. (2003), "Information Consensus in Distributed Multiple Vehicle Coordinated Control," in *CDC'03: Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, HI, USA, Dec., Vol. 2, pp. 2029–2034.

Blondel, V.D., Hendrickx, J.M., Olshevsky, A., and Tsitsiklis, J.N. (2005), "Convergence in Multiagent Coordination, Consensus, and Flocking," in *CDC-ECC '05: Proceedings of the Joint 44th IEEE Conference on Decision and Control, and the European Control Conference*, Seville, Spain, Dec., pp. 2996–3000.

Borkar, V., and Varaiya, P.P. (1982), "Asymptotic Agreement in Distributed Estimation," *IEEE Transactions on Automatic Control*, 27, 650–655.

Brualdi, R.A., and Ryser, H.J., *Combinatorial Matrix Theory*, Vol. 39 of *Encyclopedia of Mathematics and its Applications*, Cambridge, UK: Cambridge University Press (1991).

Cao, M., Morse, A.S., and Anderson, B.D.O. (2006), "Agreeing Asynchronously: Announcement of Results," in *CDC'06: Proceedings of the 45th IEEE Conference on Decision and Control*, San Diego, CA, USA, Dec., pp. 4301–4306.

Chen, Q., and Luh, J. (1994), "Coordination and control of a group of small mobile robots," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, San Diego, CA, USA, May, Vol. 3, pp. 2315–2320.

Fang, L., Antsaklis, P.J., and Tzimas, A. (2005), "Asynchronous Consensus Protocols: Preliminary Results, Simulations and Open Questions," in *CDC-ECC '05: Proceedings of the Joint 44th IEEE Conference on Decision and Control, and the European Control Conference*, Seville, Spain, Dec., pp. 2194–2199.

Fax, J.A., and Murray, R.M. (2004), "Information Flow and Cooperative Control of Vehicle Formations," *IEEE Transactions on Automatic Control*, 49, 1465–1476.

Hartfiel, D.J., *Markov Set-Chains*, Vol. 1695 of *Lecture Notes in Mathematics*, Springer Berlin / Heidelberg, Germany (1998).

Hartfiel, D.J., *Nonhomogeneous Matrix Products*, Singapore: World Scientific Publishing Co. Pte. Ldt. (2002).

Hatano, Y., and Mesbahi, M. (2005), "Agreement over Random Networks," *IEEE Transactions on Automatic Control*, 50, 1867–1872.

Hilborn, R.C., *Chaos and Nonlinear Dynamics*, New York, NY, USA: Oxford University Press (1994).

Horn, R.A., and Johnson, C.R., *Matrix Analysis*, Cambridge, UK: Cambridge University Press (1985).

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

Ji, M., Muhammad, A., and Egerstedt, M. (2006), "Leader-based Multi-agent Coordination: Controllability and Optimal Control," in *ACC'06: Proceedings of the 2006 American Control Conference*, Minneapolis, MN, USA, Jun., pp. 1358–1363.

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

Khalil, H.K., *Nonlinear Systems*, New York, NY, USA: Macmillan Publishing Company (1992).

Knorn, F., Stanojevic, R., Corless, M., and Shorten, R. (2009), "A framework for Decentralised Feedback Connectivity Control with Application to Sensor networks," *International Journal of Control*, 82, 2095–2114.

Lawton, J.R.T., Beard, R.W., and Young, B.J. (2003), "A Decentralized Approach to Formation Maneuvers," *IEEE Transactions on Robotics and Automation*, 19, 933–941.

Lewis, M.A., and Tan, K.H. (1997), "High Precision Formation Control of Mobile Robots Using Virtual Structures," *Autonomous Robots*, 4, 387–403.

Mesbahi, M., and Hadaegh, F.Y. (1999), "Formation Flying Control of Multiple Spacecraft via Graphs, Matrix Inequalities, and Switching," in *CCA'99: Proceedings of the 1999 IEEE International Conference on Control Applications*, Kohala Coast-Island of Hawai'i, HI, USA, Aug., Vol. 2, pp. 1211–1216.

Moreau, L. (2005), "Stability of Multiagent Systems with Time-Dependent Communication Links," *IEEE Transactions on Automatic Control*, 50, 169–182.

Nedić, A., and Ozdaglar, A. (2010), "Convergence Rate for Consensus with Delays," *Journal of Global Optimization*, 47, 437–456.

Olfati-Saber, R. (2006), "Flocking for Multi-Agent Dynamic Systems: Algorithms and Theory," *IEEE Transactions on Automatic Control*, 51, 401–420.

Olfati-Saber, R., Fax, J.A., and Murray, R.M. (2007), "Consensus and Cooperation in Networked Multi-Agent Systems," *Proceedings of the IEEE*, 95, 215–233.

Olfati-Saber, R., and Murray, R.M. (2003), "Consensus Protocols for Networks of Dynamic Agents," in *ACC'03: Proceedings of the 2003 American Control Conference*, Denver, CO, USA, Jun., Vol. 2, pp. 951–956.

Olfati-Saber, R., and Murray, R.M. (2004), "Consensus Problems in Networks of Agents with Switching Topology and Time-Delays," *IEEE Transactions on Automatic Control*, 49, 1520–1533.

Oppenheim, A.V., Willsky, A.S., and Nawab, S.H., *Signals & Systems*, 2 ed., Upper Saddle River, NJ, USA: Prentice-Hall, Inc. (1996).

Parker, L.E. (1998), "ALLIANCE: an architecture for fault tolerant multirobot cooperation," *IEEE Transactions on Robotics and Automation*, 14, 220–240.

Penrose, M., *Random Geometric Graphs*, Vol. 5 of *Oxford Studies in Probability*, New York, NY, USA: Oxford University Press (2003).

Ren, W., and Beard, R.W. (2004), "Formation feedback control for multiple spacecraft via virtual structures," *IEEE Proceedings of Control Theory and Applications*, 151, 357–368.

Ren, W., and Beard, R.W. (2005), "Consensus Seeking in Multiagent Systems under Dynamically Changing Interaction Topologies," *IEEE Transactions on Automatic Control*, 50, 655–661.

Ren, W., Beard, R.W., and Atkins, E.M. (2005), "A Survey of Consensus Problems in Multi-agent Coordination," in *ACC'05: Proceedings of the 2005 American Control Conference*, Portland, OR, USA, June, Vol. 3, pp. 1859–1864.

Reynolds, C., "Boids (Flocks, Herds, and Schools: a Distributed Behavioral Model)," Website: http://www.red3d.com/cwr/boids/ (2001).

Reynolds, C.W. (1987), "Flocks, Herds, and Schools: A Distributed Behavioral Model," in *SIGGRAPH'87: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, Anaheim, CA, USA, Jul., pp. 25–34.

Santi, P., *Topology Control in Wireless Ad Hoc and Sensor Networks*, Chichester, UK: John Wiley & Sons, Inc. (2005).

Shi, H., Wang, L., and Chu, T. (2005), "Virtual Leader Approach to Coordinated Control of Multiple Mobile Agents with Asymmetric Interactions," in *CDC-ECC '05: Proceedings of the Joint 44th IEEE Conference on Decision and Control, and the European Control Conference*, Seville, Spain, December, pp. 6250–6255.

Singh, S.N., Chandler, P., Schumacher, C., Banda, S., and Pachter, M. (2000), "Nonlinear Adaptive Close Formation Control of Unmanned Aerial Vehicles," *Dynamics and Control*, 10, 179–194.

Stanojević, R., and Shorten, R. (2008), "Fully Decentralized Emulation of Best-Effort and Processor Sharing Queues," in *SIGMETRICS'08: Proceedings of the 2008 ACM International Conference on Measurement and Modeling of Computer Systems*, Annapolis, MD, USA, Jun., ACM, pp. 383–394.

Stanojević, R., and Shorten, R. (2009), "Generalized distributed rate limiting," in *IWQoS'09: Proceedings of 17th IEEE International Workshop on International Workshop on Quality of Service*, Charleston, SC, USA, July, pp. 1–9.

Tanner, H.G., Jadbabaie, A., and Pappas, G.J. (2003a), "Stable Flocking of Mobile Agents Part I: Fixed Topology," in *CDC'03: Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, HI, USA, Dec., Vol. 2, pp. 2010–2015.

Tanner, H.G., Jadbabaie, A., and Pappas, G.J. (2003b), "Stable Flocking of Mobile Agents Part II: Dynamic Topology," in *CDC'03: Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, HI, USA, Dec., Vol. 2, pp. 2016–2021.

Tsitsiklis, J.N. (1984), "Problems in Decentralized Decision Making and Computation," PhD Thesis, Massachusetts Institute of Technology, Laboratory for Information and Decision Systems, Cambridge, MA, USA, Department of Electrical Engineering and Computer Science.

Tsitsiklis, J.N., Bertsekas, D.P., and Athans, M. (1986), "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Transactions on Automatic Control*, 31, 803–812.

Veloso, M., Stone, P., and Han, K. (2000), "The CMUnited-97 Robotic Soccer Team: Perception and Multi-agent Control," *Robotics and Autonomous Systems*, 29, 133–143.

Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., and Shochet, O. (1995), "Novel Type of Phase Transition in a System of Self-Driven Particles," *Physical Review Letters*, 75, 1226–1229.

Visioli, A., *Practical PID Control*, Advances in Industrial Control, Springer London (2006).

Wang, P.K.C. (1989), "Navigation Strategies For Multiple Autonomous Mobile Robots Moving In Formation," in *IROS'89: Proceedings of the IEEE/RSJ International Workshop on Intelligent Robots and Systems*, Tsukuba, Japan, September, pp. 486–493.

Appendix A: An expression for the global term

Given global and utility functions which satisfy the growth conditions, we show here that, for any $\boldsymbol{t}, \boldsymbol{t}_* \in \mathbb{R}^n$,

$$g\big(\boldsymbol{\phi}(\boldsymbol{t}_*)\big) - g\big(\boldsymbol{\phi}(\boldsymbol{t})\big) = \sum_{i=1}^{n} c^{(i)}\big(t_*^{(i)} - t^{(i)}\big) \qquad \text{where} \qquad \frac{\underline{h}^{(i)}}{\bar{d}^{(i)}} \leq c^{(i)} \leq \frac{\bar{h}^{(i)}}{\underline{d}^{(i)}} \tag{A1}$$

Letting $\boldsymbol{r} = \boldsymbol{\phi}(\boldsymbol{t})$ and $\boldsymbol{r}_* = \boldsymbol{\phi}(\boldsymbol{t}_*)$, we start by showing that $\Delta g := g\big(\boldsymbol{\phi}(\boldsymbol{t}_*)\big) - g\big(\boldsymbol{\phi}(\boldsymbol{t})\big) = g(\boldsymbol{r}_*) - g(\boldsymbol{r})$ can be expressed as

$$\Delta g = \sum_{i=1}^{n} \tilde{c}^{(i)} \Delta r^{(i)} \qquad \text{where} \qquad \underline{h}^{(i)} \leq \tilde{c}^{(i)} \leq \bar{h}^{(i)} \tag{A2}$$

and $\Delta r^{(i)} = r_*^{(i)} - r^{(i)}$. The change Δg corresponds to the change of the value of the global function when moving from $\boldsymbol{r}$ to $\boldsymbol{r}_*$. Now, instead of going "directly" from $\boldsymbol{r}$ to $\boldsymbol{r}_*$ we can also reach $\boldsymbol{r}_*$ by only changing one coordinate at a time, that is we basically break up the "cumulative change" Δg into the changes caused by moving along each coordinate. To express this mathematically, we recursively define the vectors $\boldsymbol{r}_0, \ldots, \boldsymbol{r}_n$ by

$$\boldsymbol{r}_0 = \boldsymbol{r} \qquad \text{and} \qquad \boldsymbol{r}_i = \boldsymbol{r}_{i-1} + \Delta r^{(i)} \boldsymbol{e}_i \qquad \text{for} \quad i = 1, \ldots n\,. \tag{A3}$$

Clearly, the $\boldsymbol{r}_i$ vectors correspond to the "corner points" of the "path" if one starts at $\boldsymbol{r}$ and then moves by $\Delta r^{(1)}$ along the first dimension, then by $\Delta r^{(2)}$ along the second and so on. By construction, in the end $\boldsymbol{r}_n = \boldsymbol{r}_*$.

As a consequence of the growth properties of g, we have

$$g(\boldsymbol{r}_i) - g(\boldsymbol{r}_{i-1}) = g(\boldsymbol{r}_{i-1} + \Delta r^{(i)} \boldsymbol{e}_i) - g(\boldsymbol{r}_{i-1}) = \tilde{c}^{(i)} \Delta r^{(i)} \qquad \text{where} \qquad \underline{h}^{(i)} \leq \tilde{c}^{(i)} \leq \bar{h}^{(i)} \tag{A4}$$

and since

$$\Delta g = g(\boldsymbol{r}_n) - g(\boldsymbol{r}_0) = \sum_{i=1}^{n} \big[g(\boldsymbol{r}_i) - g(\boldsymbol{r}_{i-1}) \big] \tag{A5}$$

the result (A2) now follows.

Next, we replace the difference $\boldsymbol{r}_* - \boldsymbol{r}$ by the corresponding difference $\boldsymbol{t}_* - \boldsymbol{t}$. As a consequence of the growth properties of the utility functions $f^{(i)}$, we have

$$f^{(i)}(r_*^{(i)}) - f^{(i)}(r^{(i)}) = d^{(i)}(r_*^{(i)} - r^{(i)}) = d^{(i)} \Delta r^{(i)} \qquad \text{where} \qquad 0 < \underline{d}^{(i)} \leq d^{(i)} \leq \bar{d}^{(i)} \tag{A6}$$

Since $t^{(i)} = f^{(i)}(r^{(i)})$ and $t_*^{(i)} = f^{(i)}(r_*^{(i)})$ we see that $t_*^{(i)} - t^{(i)} = d^{(i)} \Delta r^{(i)}$; hence

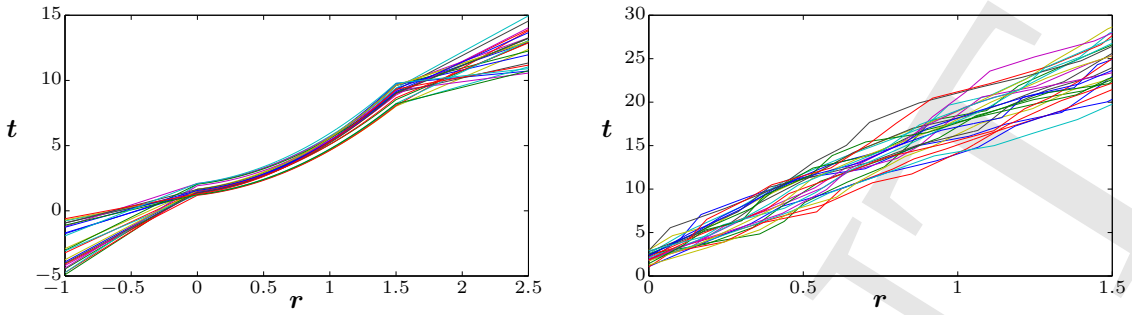
$$\Delta r^{(i)} = \frac{t_*^{(i)} - t^{(i)}}{d^{(i)}} \qquad \text{where} \qquad 0 < \underline{d}^{(i)} \leq d^{(i)} \leq \bar{d}^{(i)} \tag{A7}$$

Combining (A2) and (A7) now yields the desired result (A1).

Appendix B: Global- and utility functions used in our simulations

For some simulations, the utility functions were chosen to be of quadratic form on the interval $[\,0\,,\,1.5\,]$, and linear outside this range. Specifically, the functions were of the form

$$t^{(i)} = \begin{cases} \alpha_1^{(i)} (r^{(i)})^2 + \alpha_2^{(i)} r^{(i)} + \alpha_3^{(i)} & \text{if } 0 \leq r^{(i)} \leq 1.5 \\ \beta_1^{(i)} r^{(i)} + \beta_2^{(i)} & \text{if } r^{(i)} < 0 \\ \beta_3^{(i)} r^{(i)} + \beta_4^{(i)} & \text{otherwise} \end{cases} \tag{B1}$$



(a) Quadratic type (b) Piecewise linear

Figure B1. Utility functions $t^{(i)} = f^{(i)}(r^{(i)})$ used in the simulations demonstrating Algorithms 1 and 2.

where the coefficients $\alpha_1^{(i)}, \alpha_2^{(i)}, \alpha_3^{(i)}$ were chosen within appropriate bounds to guarantee invertibility on the interval $[\,0\,,\,1.5\,]$. The coefficients $\beta_1^{(i)}, \ldots, \beta_4^{(i)}$ where also chosen randomly, but in such a way as to guarantee that the overall function would be continuous (i. e. that the linear segments join up with the quadratic part). A set of 25 randomly generated functions of this type are shown in Fig. B1(a).

In other simulations we used piecewise linear utility functions, shown in Fig. B1(b), also chosen at random.

The global functions used were also either of quadratic form $g(\boldsymbol{r}) = d + \sum_{i=1}^{n} q^{(i)}\big(r^{(i)}\big)$ where the $q^{(i)}$ were of a similar type as (B1), or of affine form $g(\boldsymbol{r}) = d + \sum_{i=1}^{n} c^{(i)} r^{(i)}$, also with randomly chosen parameters $c^{(i)} > 0$ and $d > 0$.

All random parameters were chose within bounds, from which the required growth conditions for the theorems were derived.