

## Decentralised decision making in heterogeneous teams using anonymous optimisation

George M. Mathews<sup>a,b,\*</sup>, Hugh Durrant-Whyte<sup>a</sup>, Mikhail Prokopenko<sup>c</sup>

<sup>a</sup> ARC Centre of Excellence for Autonomous Systems, The University of Sydney, Sydney NSW, Australia

<sup>b</sup> CSIRO Industrial Physics, Lindfield NSW, Australia

<sup>c</sup> CSIRO ICT Centre, North Ryde NSW, Australia

### ARTICLE INFO

#### Article history:

Available online 19 November 2008

#### Keywords:

Asynchronous optimisation  
Team decision making  
Multi-agent systems

### ABSTRACT

This paper considers the scenario where multiple autonomous agents must cooperate in making decisions to minimise a continuous and differentiable team cost function. A distributed and asynchronous optimisation algorithm is presented which allows each agent to incrementally refine their decisions while intermittently communicating with the rest of the team. A convergence analysis provides quantitative requirements on the frequency agents must communicate that is prescribed by the structure of the decision problem. In general the solution method will require every agent to communicate to and have a model of every other agent in the team. To overcome this, a specific subset of systems, called Partially Separable, is defined. These systems only require each agent to have a combined summary of the rest of the team and allows each agent to communicate locally over an acyclic communication network, greatly increasing the scalability of the system.

© 2008 Elsevier B.V. All rights reserved.

### 1. Introduction

This paper is part of an on going research program into decision making and control algorithms for large distributed active sensor networks. The vision of this work is a system of interconnected robotic agents, such as a team of unmanned air and ground vehicles. This type of system has applications in environmental monitoring, searching and rescue, surveillance, bush firefighting, target tracking, mapping and exploration, etc. Utilising an autonomous system to perform these tasks allows human operators to be removed from possibly dangerous (or simply mundane) situations.

The decision problems that are encountered in these types of multi-agent systems generally fall into two categories, discrete (e.g. task allocation) and continuous (e.g. motion control). This work is primarily focused on the latter. Current work in this area has mainly focused on the consensus problem [1–4], whereby multiple agents, each starting out with different local values of a common decision variable, must collaboratively reach agreement.

However, what is often overlooked in consensus theory is that some decisions are explicitly better than others (there exists a common cost function). To include this it is necessary to cast the decision problem as a distributed optimisation problem.

Existing approaches to this optimisation problem are either: (1) Fully centralised [5,6] where the system as a whole is modelled and the optimal decision found using conventional algorithms. (2) Distributed or hierarchical [7] which utilise the distributed computational capacity of the multiple platforms but require a single facility to fuse information or resolve global constraints, or (3) Fully decentralised which do not require any centralised facility.

The decentralised approach can be further broken down into systems that require each platform to have global information about the system and those that only require local information from a small subset [8,9]. It is the latter that is of interest here.

This paper presents a general asynchronous gradient-based distributed optimisation algorithm that can be used to solve the team decision problem. However, this algorithm requires every agent to communicate to every other agent, limiting the scalability of the system. To overcome this issue a specific class of systems, called *Partially Separable*, is defined that enable each agent to communicate in an anonymous fashion over an acyclic communication network.

The paper is organised as follows: Section 2 presents the general decentralised and asynchronous optimisation algorithm for smooth and differentiable cost functions. A convergence analysis quantitatively specifies the communication requirements

\* Corresponding address: Advanced Technology Centre, BAE SYSTEMS, Sowerby Building, FPC 267, PO BOX 5, Filton, Bristol, United Kingdom. Tel.: +44 0 117 302 8149; fax: +44 0 117 302 8007.

E-mail addresses: [george.mathews@baesystems.com](mailto:george.mathews@baesystems.com) (G.M. Mathews), [h.durrant-whyte@cas.edu.au](mailto:h.durrant-whyte@cas.edu.au) (H. Durrant-Whyte), [mikhail.prokopenko@csiro.au](mailto:mikhail.prokopenko@csiro.au) (M. Prokopenko).

of each agent. The scalability implications of these requirements are discussed in Section 3 and the Partial Separability condition defined. Section 4 presents a decentralised system architecture for partially separable systems and gives numerical results for a simulated feature localisation task. Section 5 present conclusions and future work.

## 2. Decentralised optimisation

This section introduces the team decision problem and presents a general asynchronous decentralised optimisation algorithm based on the work of Bertsekas and Tsitsiklis [10–12].

### 2.1. Problem definition

Consider a team of  $p$  agents, where each agent  $i$  is in charge of a local decision or control variable  $u_i \in \mathcal{U}_i$ . For simplicity, this work assumes the decision spaces are scalar and thus  $\mathcal{U}_i = \mathfrak{R}$ . For an extension to multi-dimensional decisions see [13].

The agents are required to select their decisions such that a given team cost function  $J : \mathcal{U}_1 \times \dots \times \mathcal{U}_p \rightarrow \mathfrak{R}$  is minimised

$$\mathbf{u}^* = \arg \min_{\mathbf{u} \in \mathcal{U}} J(\mathbf{u}) \quad (1)$$

where

$$\mathbf{u} = [u_1, u_2, \dots, u_p]^T \in \mathcal{U}_1 \times \dots \times \mathcal{U}_p \triangleq \mathcal{U} \quad (2)$$

is the team decision vector and  $\mathbf{u}^*$  is the desired optimal team decision.

**Assumption 1a (Objective Function).** The objective function  $J$  is continuous, twice differentiable and bounded from below.

**Assumption 1b (Convexity).** In addition to Assumption 1a, the objective function  $J$  is convex.

Under the strong convexity condition of Assumption 1b, the optimal decision vector obeys the sufficient condition

$$\frac{\partial J}{\partial u_i}(\mathbf{u}^*) = 0 \quad \forall i. \quad (3)$$

If the weaker condition of Assumption 1a holds, (3) represents the first order necessary condition for a local minimum.

### 2.2. Asynchronous optimisation

To find a decision obeying (3) an asynchronous gradient descent algorithm will be developed. During the optimisation process each agent maintains a local copy of the current team decision vector which, due to the asynchronous requirements, may contain outdated information about other agents proposed decisions. It is given at discrete time  $t$  for agent  $i$  as

$$\mathbf{u}^i(t) = [u_1^i(t), \dots, u_p^i(t)]^T. \quad (4)$$

In general superscripts refer to an agent, while subscripts represent a component of the decision vector (e.g.  $u_j^i(t)$  represents agent  $i$ 's local copy of agent  $j$ 's component of the team decision vector).

The time variable  $t$  is simply used to represent when discrete events take place (such as when an agent computes an update or communicates) and does not require each agent have access to a global clock or perform a periodic synchronisation.

To quantify the effects of communication delays, the variable  $\tau_j^i(t)$  will denote when agent  $i$ 's local copy  $u_j^i(t)$  was generated by agent  $j$  and hence  $u_j^i(t) = u_j^j(\tau_j^i(t))$ . It is assumed that  $\tau_j^i(t) = t$  and thus agent  $i$  always has the latest copy of its decision variable. Using this notation (4) can be written as

$$\mathbf{u}^i(t) = [u_1^i(\tau_1^i(t)), \dots, u_p^i(\tau_p^i(t))]^T. \quad (5)$$

### 2.3. Local update

Each agent employs a local update rule  $f_i : \mathcal{U} \rightarrow \mathcal{U}_i$  that modifies its proposed decision based on what it knows about the decisions of the rest of the team. To allow each agent to perform updates asynchronously, a set of times  $T_U^i \subseteq \{1, 2, \dots\}$  will be associated with each agent  $i$  that represent when the agent computes a local update. It is assumed the set  $T_U^i$  has an infinite number of elements and thus each agent never stops computing updates

$$u_i^i(t+1) = \begin{cases} f_i(\mathbf{u}^i(t)) & \text{if } t \in T_U^i \\ u_i^i(t) & \text{else.} \end{cases} \quad (6)$$

For a general gradient descent type algorithm, the update function has the form

$$f_i(\mathbf{u}^i(t)) = u_i^i(t) + \gamma_i s_i(\mathbf{u}^i(t)) \quad (7)$$

where  $\gamma_i$  is a constant step size to be defined and  $s_i : \mathcal{U} \rightarrow \mathcal{U}_i$  is the update direction, e.g. for steepest descent

$$s_i(\mathbf{u}^i(t)) = -\frac{\partial J}{\partial u_i}(\mathbf{u}^i(t)). \quad (8)$$

### 2.4. Communication

Communication is initiated by an agent  $i$  sending a message, at some time  $t \in T_C^{ij} \subseteq \{1, 2, \dots\}$ , to agent  $j$  containing its latest control  $u_i^i(t)$ , where the set  $T_C^{ij}$  is assumed to contain an infinite number of elements. After some communication delay  $b_{ij}(t)$  agent  $j$  receives it and incorporates it into its local copy of the team decision vector, thus  $u_j^j(t + b_{ij}(t)) = u_i^i(t)$  and  $\tau_j^j(t + b_{ij}(t)) = t$ .

Although this only explicitly models direct communication between two nodes, the actual implementation may employ information routing or forwarding via intermediate nodes.

### 2.5. Convergence results

This section presents a sufficient condition for the above algorithm to converge that quantitatively relates the communication frequency, update direction and the structure of the objective function to a maximum allowable step size for each agent.

Convergence is proved through ensuring cost reduction. This can be accomplished if a bound on the cost function's Hessian is known. This allows the effects of the outdated information one agent has about another to also be bounded.

**Assumption 2 (Local Descent).** For every  $i$  and  $t \in T_U^i$

$$(i) \quad s_i(\mathbf{u}^i(t)) \frac{\partial J}{\partial u_i}(\mathbf{u}^i(t)) \leq 0. \quad (9)$$

(ii) There exists positive constants  $C_i$  and  $D_i$  such that

$$C_i \left| \frac{\partial J}{\partial u_i}(\mathbf{u}^i(t)) \right| \leq |s_i(\mathbf{u}^i(t))| \leq D_i \left| \frac{\partial J}{\partial u_i}(\mathbf{u}^i(t)) \right|. \quad (10)$$

This assumption ensures the updates made by individual agents are in a direction of decreasing cost (based on local information), are bounded and may only be zero if the gradient of the cost function is zero.

**Assumption 3 (Bounded Delays).** There exists positive constants  $B_{ij}$  such that

$$t - \tau_j^i(t) \leq B_{ij} \quad \forall i, j, t. \quad (11)$$

This restricts the possible values of  $T_C^{ij}$  and  $b_{ij}(t)$  by requiring the age of agent  $j$ 's local copy of  $i$ 's decision to be bounded. Informally, this can be relaxed such that  $B_{ij}$  represents the time difference, measured in the number of updates computed by agent  $i$ , between  $u_i^i(t)$  and  $u_i^j(t)$ .

**Assumption 4 (Coupling).** There exist positive constants  $K_{ij}$  such that

$$\left| \frac{\partial^2 J}{\partial u_i \partial u_j}(\mathbf{u}) \right| \leq K_{ij} \quad \forall i, j, \mathbf{u} \in \mathcal{U}. \quad (12)$$

This imposes a limit on the maximum coupling between the decisions made by different agents, represented by bounds on the second derivatives of the objective function.

**Theorem 1 (General Convergence).** *Assumptions 1a and 2–4 provide sufficient conditions for the distributed optimisation algorithm defined by (6) and (7) to converge with*

$$\lim_{t \rightarrow \infty} \frac{\partial J}{\partial u_i}(\mathbf{u}^G(t)) = 0 \quad \forall i \quad (13)$$

where  $\mathbf{u}^G(t) = [u_1^1(t), \dots, u_p^p(t)]$  is the global decision vector formed with the most recent local decision from each agent and  $\gamma_i \in (0, \Gamma_i)$ , where

$$\Gamma_i = \frac{2}{D_i \left( K_{ij} + \sum_{j \neq i} K_{ij}(1 + B_{ij} + B_{ji}) \right)}. \quad (14)$$

See Appendix for proof.

This theorem guarantees that the algorithm will converge to the optimal decision if the cost function is convex. For nonconvex functions, convergence is still guaranteed but only to a local minimum. This general result provides a unified way to relate communication requirements and problem structure for a general cooperative multi-agent system.

### 2.6. Example 1: Quadratic cost

To demonstrate the convergence properties and communication requirements this section will consider a set of decision makers minimising the simple quadratic objective function

$$J(\mathbf{u}) = \frac{1}{2} \mathbf{u}^T \mathbf{A} \mathbf{u}. \quad (15)$$

For some positive definite matrix  $\mathbf{A} = \{A_{ij}\}$ . Although this has a trivial solution it will highlight the issues involved.

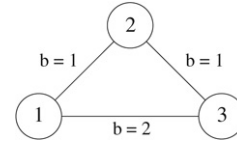
To completely specify a multi-agent algorithm to solve this, the local update equation (7) and the communication network/protocol must be specified.

- **Local update:** Each agent  $i$  computes an update at each time step  $T_U^i = \{0, 1, 2, \dots\}$ , where the update direction is defined by the steepest decent direction. Thus

$$u_i^i(t+1) = u_i^i(t) - \gamma_i \mathbf{A}_i \mathbf{u}^i(t), \quad \forall i \quad (16)$$

where  $\mathbf{A}_i$  is the  $i$ th row of  $\mathbf{A}$ . For this equation, the constant  $D_i = 1$ .

- **Communication:** Each agent  $i$  communicates after each local iteration to all other agents  $j \neq i$  and thus  $T_C^{ij} = \{1, 2, \dots\}$ . It is assumed that the message delivery delays are constant for each communication link and given by  $b_{ij}$ . This results in  $u_i^j(t) = u_i^i(t - b_{ij})$ . It is noted that for this particular setup, the delivery delay  $b_{ij}$  corresponds to the delay terms  $B_{ij}$ .



**Fig. 1.** Fully connected communication network between 3 agents with associated bi-directional communication delays.

The last term that must be specified is the step size  $\gamma_i$ . Based on Theorem 1  $\gamma_i$  is given by

$$\gamma_i = \frac{\beta}{K_{ii} + \sum_{j \neq i} K_{ij}(1 + B_{ij} + B_{ji})} \quad (17)$$

where  $\beta \in (0, 2)$  and for this example is set to a conservative 1.5. The delays terms  $B_{ij}$  are defined above, while the coupling terms are given by the matrix elements

$$K_{ij} = \left| \frac{\partial^2 J}{\partial u_i \partial u_j}(\mathbf{u}) \right| = |A_{ij}| \quad (18)$$

for all  $i$  and  $j$ .

Consider the 3 dimensional case consisting of 3 agents with an  $\mathbf{A}$  matrix defined as

$$\mathbf{A} = \begin{bmatrix} 3 & 2 & 1.5 \\ 2 & 3 & 2 \\ 1.5 & 2 & 3 \end{bmatrix}. \quad (19)$$

The delay terms are specified by a similar matrix  $\mathbf{B}$

$$\mathbf{B} = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{bmatrix} \quad (20)$$

and is shown in Fig. 1.

To demonstrate the full convergence properties, the local copy of each agents decision vector is initialised to the sum of the eigen vectors  $\mathbf{v}_i$  of  $\mathbf{A}$ ,

$$\mathbf{u}^i(0) = \sum_{i=1}^3 \mathbf{v}_i \approx \begin{bmatrix} -0.89 \\ -1.25 \\ 0.53 \end{bmatrix}, \quad \text{for all } i \in \{1, 2, 3\}. \quad (21)$$

The resulting evolution of the local decision of each agent and the associated value of the objective function is shown in Fig. 2. It is noted from Fig. 2(b) that this distributed algorithm maintains the linear convergence rate typical of steepest descent algorithms approaching a non-singular stationary point. This property has been proved for a similar distributed algorithm in [14].

### 3. Heterogeneity and system structure

This section examines the general structure of heterogeneous team decision problems and explicitly considers what information each agent requires about the team.

To start, consider a general heterogeneous team of agents. For each agent to evaluate the objective function (or any higher order derivatives) it will not only require the decision of all the other agents but also require a model of how their decision will modify or impact the system.

For large systems, containing many agents, the computational costs of storing and operating with these models, along with the bandwidth required to communicate the local decisions will become prohibitive. To deal with this issue, a particular form of the objective function will be defined that will enable the effect or impact of a group of agents to be described in the same manner as the impact of a single agent. For systems with this type of cost function it will also be shown that each agent will only be required to communicate over an acyclic network.

**Fig. 2.** Typical convergence results of the distributed optimisation algorithm for a quadratic objective function. (a) Local decision variables  $u_i^l(t)$  versus iterations  $t$ . (b) Objective function value. The dashed line shows the value of the global team decision,  $J(\mathbf{u}^c(t))$  at each iteration. The solid lines represent the objective function value of the local decision vectors of each agent,  $J(\mathbf{u}^i(t))$  for each  $i = 1, 2, 3$ .

### 3.1. Partial separability

This section defines a class of systems where each agent does not require specific information about every other agent. In general terms, it will be shown that if the effect or impact of a group of agents on the team objective or cost function, can be described in the same manner as the impact of a single agent. Then, each agent can consider the rest of the team as a single entity, described in the same manner as any other agent, irrespective of how large the team is.

The key concept behind this is the ability to describe the impact of a group of agents in the same manner (i.e. description length) as the impact of a single agent. Systems that exhibit this property will be called *Partially Separable*.

This enables each agent to independently evaluate its future impact for a proposed local decision. These impacts are then communicated around the team and combined such that each agent passes on an anonymous summary of what it knows about the impact of other agents. This concept will now be formalised.

**Definition 1 (Impact Function).** An impact function  $\Upsilon_i$ , of a given agent  $i$ , abstracts the local sensor and actuator models and maps a decision  $u_i$  onto a common team impact space  $\mathcal{S}$

$$\Upsilon_i : \mathcal{U}_i \rightarrow \mathcal{S}. \quad (22)$$

Given the impacts from two agents,  $\alpha_\mu = \Upsilon_\mu(u_\mu)$  and  $\alpha_\nu = \Upsilon_\nu(u_\nu)$ , the combined impact will be generated using a binary composition operator  $*$  defined on the impact space  $\mathcal{S}$ . This allows two impacts to be combined without losing any task related information.

**Definition 2 (Impact Properties).** For all non-empty and non-overlapping sets of agents  $\mu, \nu \subset \{1, \dots, p\}$ , such that  $\mu \cap \nu = \emptyset$  and  $\sigma = \mu \cup \nu$ , there exists

(i) A commutative and associative binary operator  $*$  defined for the impacts  $\alpha_\mu, \alpha_\nu \in \mathcal{S}$  such that the combined impact  $\alpha_\sigma$  is given by

$$\alpha_\sigma = \alpha_\mu * \alpha_\nu. \quad (23)$$

(ii) An inverse of all elements of  $\mathcal{S}$  and a unity element  $\alpha_\emptyset$ , such that the impact  $\alpha_\mu$  can be recovered by

$$\alpha_\sigma * \alpha_\nu^{-1} = \alpha_\mu * \alpha_\nu * \alpha_\nu^{-1} = \alpha_\mu * \alpha_\emptyset = \alpha_\mu. \quad (24)$$

This formally defines an *Abelian group*.

A partially separable objective function of the team is defined as a mapping from the combined team impact to the real line. This is in contrast to the previous definition, which defines it as a mapping directly from the team controls  $u_i$  to  $\mathfrak{R}$ .

**Definition 3 (Partial Separability).** The partially separable objective function  $J$  is equivalent to the mapping  $\psi : \mathcal{S} \rightarrow \mathfrak{R}$ , given by

$$J(\mathbf{u}) = \psi(\alpha_T) \quad (25)$$

where

$$\alpha_T = \Upsilon_1(u_1) * \Upsilon_2(u_2) * \dots * \Upsilon_p(u_p). \quad (26)$$

To demonstrate this concept, an reconnaissance scenario will be examined for the case when all uncertainties are Gaussian.

### 3.2. Example 2: Reconnaissance with Gaussian Uncertainty

A typical scenario of interest in this paper is the reconnaissance or information gathering task. This scenario requires the agents to decide on a joint decision that minimises the uncertainty about a particular random variable  $\mathbf{x} \in \mathcal{X}$  given some prior information. This state variable may include positions of targets, terrain properties of a given region, or surface information of a remote planet, for example.

This section will show that the reconnaissance scenario, when all uncertainties are described by Gaussian distributions, has a partially separable objective function. For a detailed reference on Gaussian filtering and the extended Kalman filter see [15].

#### 3.2.1. Prior

It is assumed that all agents have access to the prior information about probably values of  $\mathbf{x}$ , defined as a probability density function  $P(\mathbf{x})$  and assumed Gaussian with mean  $\hat{\mathbf{x}}$  and covariance  $\mathbf{P}$

$$P(\mathbf{x}) \sim N(\mathbf{x}; \hat{\mathbf{x}}, \mathbf{P}). \quad (27)$$

#### 3.2.2. Observation model

Agent  $i$ 's observation is modelled by

$$\mathbf{z}_i = \mathbf{h}_i(\mathbf{x}, u_i) + \mathbf{w}_i \quad (28)$$

where  $\mathbf{w}_i$  is Normally distributed with zero mean and covariance given by  $\mathbf{R}_i(\mathbf{x}, u_i)$ . In the implementation of the extended Kalman













