Science of Computer Programming 167 (2018) 114–137

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

# Towards attack-resistant Aggregate Computing using trust mechanisms

Roberto Casadei [a,*], Alessandro Aldini [b], Mirko Viroli [a]

[a] Alma Mater Studiorum—Università di Bologna, Italy
[b] Università di Urbino Carlo Bo, Italy

ARTICLE INFO

*Article history:*
Received 26 December 2017
Received in revised form 26 July 2018
Accepted 31 July 2018
Available online 10 August 2018

*Keywords:*
Aggregate Computing
Computational fields
Collaborative P2P systems
Security
Trust

ABSTRACT

Recent trends such as the Internet of Things and pervasive computing demand for novel engineering approaches able to support the specification and scalable runtime execution of adaptive behaviour for large collections of interacting devices. Aggregate Computing is one such approach, formally founded in the field calculus, which enables programming of device aggregates by a global stance, through functional composition of self-organisation patterns that is turned automatically into repetitive local computations and gossip-like interactions. However, the logically decentralised and open nature of such algorithms and systems presumes a fundamental cooperation of the devices involved: an error in a device or a focused attack may significantly compromise the computation outcome and hence the algorithms built on top. For this reason, in this paper, we move the first steps towards attack-resistant aggregate computations. We propose *trust* as a framework to detect, ponder or isolate voluntary/involuntary misbehaviours, with the goal of mitigating the influence on the overall computation. On top of this, we consider recommendations in order to provide more reactivity and stability through the sharing of individual perceptions. To better understand the fragility of aggregate systems in face of attacks and investigate the extent of the mitigation afforded by the adoption of trust mechanisms, we consider the paradigmatic case of the *gradient* algorithm. Experiments are carried out to analyse the sensitivity of the adopted trust framework to malevolent actions and to study the impact of different factors on the error committed by trust-based gradients under attack. Finally, in a case study of the *spatial channel* algorithm, it is shown how the protection afforded by attack-resistant gradients can be effectively propagated to higher-level building blocks.



## 1. Introduction

The last decades have been feeding a process where large numbers of interconnected computing devices get densely deployed in our living and working environments. Such technological and social movements are implying a future of increasing pervasiveness and interconnection, where dense networks of computer-like nodes are overlaid on and tightly interact with our physical world and humans in it. Exploiting one such computational fabric is appealing but it does challenge current methods and tools in a paradigmatic way: the large-scale, situated and complex nature of this kind of systems makes open-loop approaches infeasible and pushes forward the need to embrace self-* properties, but hence a whole set of new challenges arises.

* Corresponding author.
*E-mail address:* roby.casadei@unibo.it (R. Casadei).

https://doi.org/10.1016/j.scico.2018.07.006
0167-6423/

The field of collective adaptive systems (CASs) [1] is devoted to the study of systems where large groups of entities jointly seek to reach their goals in a dynamic environment [2]. The main issues in this context include *(i)* how to provide an effective specification of self-organising and goal-oriented behaviour and *(ii)* how this can turn into efficient, resilient and distributed execution. Moreover, from the engineering side, trade-offs have to be carefully balanced to limit usage of resources and still provide the required quality of service. Given this complex setting and the impracticality of in-field tests, it is crucial to be able to count on ways to obtain certain guarantees on the correctness of implementations; for this purpose, both formal methods and simulations are invaluable.

Aggregate Computing [3] is one promising approach for the engineering of (possibly large-scale) distributed systems that need to resiliently adapt to local, environmental conditions. The idea is to directly express, by a single global program, the behaviour of an *aggregate* of devices, all of which interpret that program and correspondingly execute local actions encompassing continuous sense of the environment, computation of local data, and their sharing with neighbours. This approach takes an abstract, global stance in which one programs the desired collective behaviour, through a composition of self-organising coordination patterns [4], and lets the platform deal with the proper unfolding of the computation at the micro-level in a complex set of repetitive, weaved interactions and calculations. The field calculus [5–7], which builds on the idea of computational fields (spatially-distributed data structures), provides the formal underpinnings of Aggregate Computing and gives a concrete shape to the approach: in this framework, programs at the aggregate level are represented as functional compositions of dynamic fields that map devices to computational values in space-time. In practice, an aggregate system consists of a collection of networked devices where each device computes the same aggregate program and interacts with a subset of other devices known as its neighbourhood. That is, computation unwinds in a logically decentralised way based on locally sensed information and data received through peer-to-peer, gossip-like communications.

Among the various challenges behind the design and development of successful CASs, trust represents a fundamental aspect in a setting in which the rapid and continuous exchange and propagation of information is key. The need for cooperation is typically accompanied by the growth of potential (insider) threats, which may depend on selfish or malicious behaviours of nodes, either in isolation or in collusion. From the viewpoint of a node issuing a request to another node, which may refer to the communication of a simple detected value or the delivery of a complex paid service, trust can be defined as the belief perceived by the former node about the capability, intention, honesty, and reliability of the latter node in satisfying the request. The estimation of such a belief perception through automatic mechanisms supporting the decision-making processes improves the reliability of CASs and, therefore, it is quite natural to investigate their application also to the Aggregate Computing framework. However, implementing a trust model respecting the principles of Aggregate Computing is particularly challenging in a distributed setting in which the promptness and success of self-adaptation strongly depends on the accuracy and reliability of (partial) information exchanged among interacting devices.

Starting with these considerations, we propose the combination of trust and Aggregate Computing, with the goal of making collective, cooperative computations more resilient to voluntary or involuntary misbehaviours. The results of the experimental analysis of the proposed approach are promising: the use of trust mechanisms can effectively reduce or nullify the error caused by subversive behaviours, thus contributing to alleviate the fragility inherent to open, cooperative computations.

This paper is organised as follows. In Section 2, we recall the paradigm of Aggregate Computing, its formal instantiation in the field calculus framework, and describe the basics of programming with computational fields — using ScaFi as reference implementation. Here, we also introduce the "collective algorithms" considered in the evaluation part of the paper — namely, the self-healing, self-organising gradient and channel algorithms. In Section 3, we provide an overview of the security issues in Aggregate Computing, motivating this study and its focus. Then, in Section 4, we discuss and refactor the applicability of classical trust management techniques to the Aggregate Computing setting. Section 5, hence, describes a trust-based implementation of the gradient algorithm. Then, in Section 6, we present a comprehensive empirical study of the effectiveness of trust fields in mitigating or avoiding at all the consequences of (deliberate or not) misconducts of nodes in a gradient computation. An empirical study of the benefits of using trust-based gradients in a more complex computation — i.e., the channel — is considered in Section 7; this proves the effectiveness of the approach in aggregate algorithmic compositions that include attack-resistant building blocks. Finally, comparison with related work and future perspectives are discussed in Section 8.

Most specifically, this paper extends the work in [8] with: sensitivity analysis on the main factors affecting the trust algorithm (Sections 6.2, 6.3); description, implementation and analysis of recommendations on top of the plain trust algorithm (Sections 4, 5, 6.3); refactoring and description of the source code for the trust-based gradient algorithms (Section 5); empirical study of the response of the trust algorithms against mobile attackers (Section 12); and empirical study of the benefits of selecting a trust-based gradient algorithm in a more complex computation, namely the *self-organising channel* (Section 7).

## 2. Aggregate Computing

In what follows, we provide a general description of the Aggregate Computing paradigm, then we introduce the formal framework that makes it operational, and finally show a progression of examples covering the basic elements of the framework up to the components considered in the evaluation sections.