



HAL
open science

Synchronizability of Communicating Finite State Machines is not Decidable

Alain Finkel, Etienne Lozes

► **To cite this version:**

Alain Finkel, Etienne Lozes. Synchronizability of Communicating Finite State Machines is not Decidable. 2017. hal-01474722v3

HAL Id: hal-01474722

<https://hal.science/hal-01474722v3>

Preprint submitted on 6 Mar 2017 (v3), last revised 7 Aug 2018 (v5)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

Synchronizability of Communicating Finite State Machines is not Decidable

Alain Finkel¹ and Etienne Lozes¹

¹ LSV, ENS Cachan, CNRS, France
{finkel,lozes}@lsv.fr

Abstract

A system of communicating finite state machines is *synchronizable* [1, 4] if its send trace semantics, i.e. the set of sequences of sendings it can perform, is the same when its communications are FIFO asynchronous and when they are just rendez-vous synchronizations. This property was claimed to be decidable in several conference and journal papers [1, 4, 3, 2] for either mailboxes ($*-1$) or peer-to-peer ($1-1$) communications, thanks to a form of small model property. In this paper, we show that this small model property does not hold neither for mailbox communications, nor for peer-to-peer communications, therefore the decidability of synchronizability becomes an open question. We close this question for peer-to-peer communications, and we show that synchronizability is actually undecidable. We show that synchronizability is decidable if the topology of communications is an oriented ring. We also show that, in this case, synchronizability implies the absence of unspecified receptions and orphan messages, and the channel-recognizability of the reachability set.

1998 ACM Subject Classification F.1.2 Modes of Computation

Keywords and phrases verification, distributed system, asynchronous communications, choreographies

Digital Object Identifier 10.4230/LIPIcs...

1 Introduction

Asynchronous distributed systems are error prone not only because they are difficult to program, but also because they are difficult to execute in a reproducible way. The slack of communications, measured by the number of messages that can be buffered in a same communication channel, is not always under the control of the programmer, and even when it is, it may be delicate to choose the right size of the communication buffers.

Slack elasticity of a distributed system with asynchronous communications is the property that the “observable behaviour” of the system is the same whatever the slack of communications is. There are actually as many notions of slack elasticity as there are notions of observable behaviours (and of distributed systems). Slack elasticity has been studied in various contexts: for hardware design [15], with the goal of ensuring that some code transformations are semantic-preserving, for parallel programming in MPI [17, 18], for ensuring the absence of deadlocks and other bugs, or more recently for web services and choreographies [1, 4, 2], for verifying various properties, among which choreography realizability [3].

This paper focuses on *synchronizability* [1], a special form of slack elasticity that was defined by Basu and Bultan for analyzing choreographies. Synchronizability is the slack elasticity of the send trace semantics of the system: a system of communicating finite state machines is synchronizable if any asynchronous trace can be mimicked by a synchronous one that contains the same send actions in the same order. Synchronizability was claimed