University of Trento
Department of Computer Science

An algorithm implementation for Global Predicate Evaluation

Trento, 29/08/2016

Zen Roberto - Bof Michele

Abstract

One core problem in distributed computing is to detect wheter a particular state in a computation can be reached and need to be detected. Problems such as deadlocks detection, monitoring or debugging can be all seen as an instance of the so called Global Predicate Evaluation (GPE) problem. This problem consists in evaluate wheter a given condition called predicate is satisfied among the consistent global states of the system. In order to solve it, one has to face all the practical issues that may arise in a distributed computation: asynchrony and failures of the underlying distributed system, message ordering and inconsistent observations. In this work, we present an implementation that solves GPE based on a simulation of a distributed system. The model built is based on message passing between peers and a monitor which passively observe the system in order to build its global states.

1 Introduction

Aliqua eu exercitation mollit incididunt sunt labore in. Ut et Lorem non irure excepteur ad do laboris magna nostrud nostrud id minim eiusmod officia dolor. Pariatur sunt culpa ex do incididunt ad consequat officia cillum ex laboris. Occaecat occaecat labore sint reprehenderit aute pariatur sunt deserunt commodo pariatur in esse. Labore veniam nulla nisi esse sint aute pariatur ut commodo ipsum proident. Dolore quis quis quis commodo ad cupidatat elit in reprehenderit anim reprehenderit magna deserunt qui dolor officia. Eiusmod quis ad commodo laborum proident enim proident duis ad qui officia.

Motivation

The reasons which have motivated our group to conduct a study on this topic are the following:

1. **Reason 1**. Enim aute ex ullamco eiusmod eu sit ullamco reprehenderit esse anim sint amet esse..
2. **Id cillum exercitation adipisicing incididunt commodo quis tempor labore.**. Id cillum exercitation adipisicing incididunt commodo quis tempor labore..
3. **Id cillum exercitation adipisicing incididunt commodo quis tempor labore.**. Id cillum exercitation adipisicing incididunt commodo quis tempor labore.

Goals

The objectives which we have set ourselves are the following:

1. **Simulate a distributed computation**. Structure the project in such a way that an user can define how the simulation has to be executed and and how long it has to last. Moreover how many peers run over it and configura the probability of executing and internal event rather than a message exchange.

2. **Id cillum exercitation adipisicing incididunt commodo quis tempor labore.**. Id cillum exercitation adipisicing incididunt commodo quis tempor labore..

3. **Id cillum exercitation adipisicing incididunt commodo quis tempor labore.**. Id cillum exercitation adipisicing incididunt commodo quis tempor labore.

Results

We developed a practical solution to the GPE problem based on a random simulation of distrubuted computations. The project has been develop in Java using JDK version's 1.7 in combination with the Akka framework which allowed us to build an highly concurrent, distributed, and resilient message-driven application[1]. We started implementing the behaviour of the peers and then we defined the message exchange between them. In particolar, we adopted a simple strategy for structure messages in a way that they were able to transport all the information needed at the very end of the simulation. To conclude, we defined the behaviour of the monitor such that it was able to collect all the messages exchanged between the peers and evaluate a given predicate over them building consistend global states stored in a lattice of events.

Outline

The outline of this report is structured as follows. Section 2 describes the problem of GPE while Section 3 explains details about the solution we propose, the problems we faced and the strategies we adopted to overcome them. In Section 4 we will show some experimental evaluations of our implementation. The last section states the conclusion of our work and ideas on how we could improve our solution with techniques based on high-level operators that run in parallel.

[1] `http://akka.io`

2 The Problem

3 The Solution

At the end we decided to allow users to run a simulation of a distributed computation. Indeed, each execution of our implementation runs a simulation.

This section gives readers all the competences to understand our proposed solution.

In this section we do not go deep into the implementation details, instead we will focus on the logics behind our strategies.

3.1 Initialization

Each simulation starts with a

3.2 The simulation

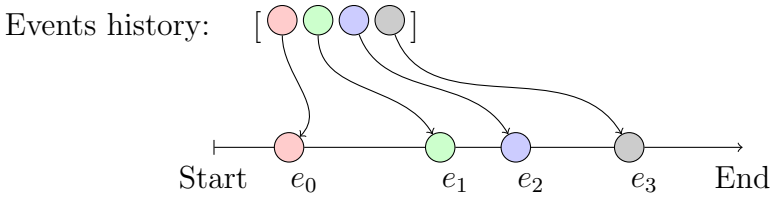

Figure 1: The sequence of events stored in the peer array's and scheduled in the time space diagram.

Non laborum excepteur sint sunt aute esse aliquip. Velit irure ea culpa nulla id mollit laboris laboris adipisicing ipsum non do in id aute anim cupidatat. Veniam deserunt excepteur commodo eu duis nulla adipisicing anim amet adipisicing ex. Laborum Lorem do nulla sint qui voluptate sint esse in incididunt Lorem cillum voluptate tempor nisi. Mollit voluptate mollit eiusmod anim elit nostrud voluptate veniam commodo id do officia cillum voluptate fugiat cillum. In incididunt proident qui id velit eiusmod aliquip aliqua nulla officia ut. Dolore aute est aliqua qui do eu sunt in est ad aliquip

nulla anim deserunt. Ut sunt voluptate consectetur magna ex minim ut magna cillum aliquip elit voluptate ex excepteur eiusmod occaecat.

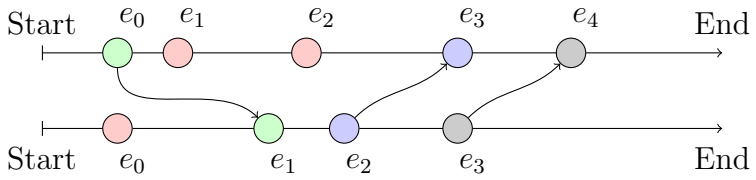

Figure 2: An example of a simulation. Red circles are peers' internal events.

Non laborum excepteur sint sunt aute esse aliquip. Velit irure ea culpa nulla id mollit laboris laboris adipisicing ipsum non do in id aute anim cupidatat. Veniam deserunt excepteur commodo eu duis nulla adipisicing anim amet adipisicing ex. Laborum Lorem do nulla sint qui voluptate sint esse in incididunt Lorem cillum voluptate tempor nisi. Mollit voluptate mollit eiusmod anim elit nostrud volup.

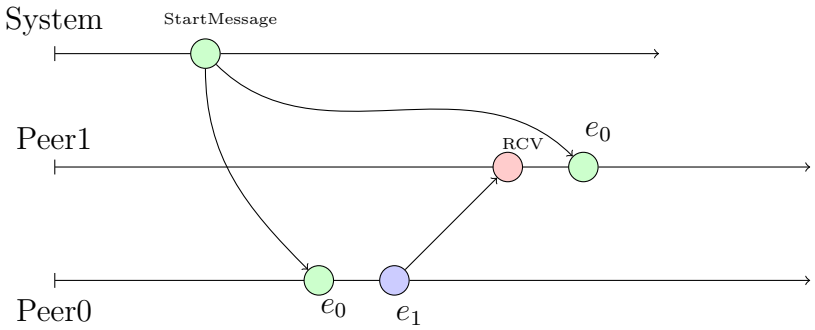

Figure 3: An overcomed issue in our asyncronous simulation.

Non laborum excepteur sint sunt aute esse aliquip. Velit irure ea culpa nulla id mollit laboris laboris adipisicing ipsum non do in id aute anim cupidatat. Veniam deserunt excepteur commodo eu duis nulla adipisicing anim amet adipisicing ex. Laborum Lorem do nulla sint qui voluptate sint esse in incididunt Lorem cillum voluptate tempor nisi. Mollit voluptate mollit eiusmod anim elit nostrud volup