Implementing high performance Synchronous Message Exchange
University of Copenhagen

Truls Asheim

April 17, 2015

Abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla laoreet lobortis erat, ac fringilla quam accumsan ut. Nam eget tortor neque. Nulla dictum dapibus venenatis. Vestibulum interdum sem vestibulum enim vulputate ullamcorper nec ut velit. Suspendisse rhoncus a nisi eget viverra. Curabitur porttitor hendrerit elit vel porta. Phasellus elementum nulla aliquet, rhoncus sapien sit amet, maximus libero. Morbi ipsum sapien, faucibus et tempor sed, commodo at enim. Curabitur maximus pretium felis, ac elementum nisi tincidunt et. Sed facilisis lacus purus, sit amet blandit elit ultricies sit amet. Maecenas et cursus libero, ac tincidunt arcu. In at tincidunt ante. Pellentesque lacinia dui nec nulla accumsan, auctor faucibus elit suscipit. Pellentesque quis justo turpis.

Praesent in dignissim lacus, in auctor felis. Phasellus iaculis finibus sapien, vitae vestibulum leo rutrum ac. Etiam in tellus eget sapien suscipit ornare eget sit amet lectus. Vestibulum sagittis consectetur varius. Aenean consectetur ante quis dui iaculis fringilla. Etiam enim sem, facilisis et interdum sit amet, congue nec metus. Fusce eu imperdiet enim, non sollicitudin sem. Vestibulum malesuada mattis justo, vel dictum massa tristique ac. Vivamus pretium turpis ut ligula porta, eget facilisis eros pulvinar. Aenean elit mi, semper vitae ornare vel, pretium sit amet ex. Cras commodo justo quis dolor gravida auctor. Vivamus ultrices arcu dolor, vitae porta ante viverra et. Quisque at aliquet massa, vel gravida felis. Donec eu imperdiet mi. Aliquam in tortor a libero iaculis luctus.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Project description

In this report, we describe a synchronous message passing framework intended to aid simulation of applications whose operating semantics are compatible with the SME model. An i

1.2 bar

Sed orci augue, fringilla ac ipsum a, accumsan lacinia neque. Cras sed odio in odio tempor cursus. Integer consectetur erat turpis, at ullamcorper ex suscipit vitae. Nulla ultricies nunc ac elementum tempus. Nunc porttitor eros eget nisi tempor ultrices. Curabitur non velit luctus tortor pretium volutpat ac at tortor. Praesent ornare, arcu eu varius porttitor, ex metus pulvinar odio, vel porta turpis tellus vel turpis. Donec vel pretium orci, ac condimentum ligula. Aliquam elementum diam ante, ac dapibus nisi suscipit et. Aliquam porttitor sagittis elementum. Aenean turpis neque, maximus et tellus quis, porttitor varius elit.

baz

In ac justo lorem. Aliquam ornare porttitor metus id consectetur. Praesent porttitor nisl a lorem scelerisque, ut scelerisque metus fringilla. Pellentesque lacinia eros eget metus mollis, a eleifend dui malesuada. Donec vestibulum varius efficitur. Nam tellus dolor, convallis at massa vel, dignissim pellentesque ligula. Integer vitae diam vel felis pellentesque consequat. Donec varius mi id ex hendrerit rhoncus. Nam feugiat porttitor accumsan. Nunc suscipit a ante vitae laoreet. Proin ut viverra sapien.

Aliquam eget enim finibus, ornare est a, lobortis ligula. Fusce sollicitudin ligula. Phasellus quis tempor nibh. Integer ut justo ac quam sollicitudin ultricies et id turpis. Sd a gravida metus. Fusce facilisis ac felis in ullamcorper. Proin porta urna enim, non ornare urna mollis id. Vestibulum mattis nulla ac lacus suscipit euismod. Etiam odio sapien, posuere tristique scelerisque a, tempor ut ex. Maecenas sed dui ex.

Chapter 2

Background and Analysis

2.1 Motivation

2.2 Synchronous Message Exchange

2.3 Problem characteristics

2.4 Towards paralelization

Praesent in dignissim lacus, in auctor felis. Phasellus iaculis finibus sapien, vitae vestibulum leo rutrum ac. Etiam in tellus eget sapien suscipit ornare eget sit amet lectus. Vestibulum sagittis consectetur varius. Aenean consectetur ante quis dui iaculis fringilla. Etiam enim sem, facilisis et interdum sit amet, congue

2.5 Static process orchestration

2.6 Monte Carlo orchestration

2.7 Optimization-based orchestration

2.8 Adaptive process orchestration

2.9 Adaptive Monte Carlo process orchestration

2.10 Adaptive Optimization-based process orchestration

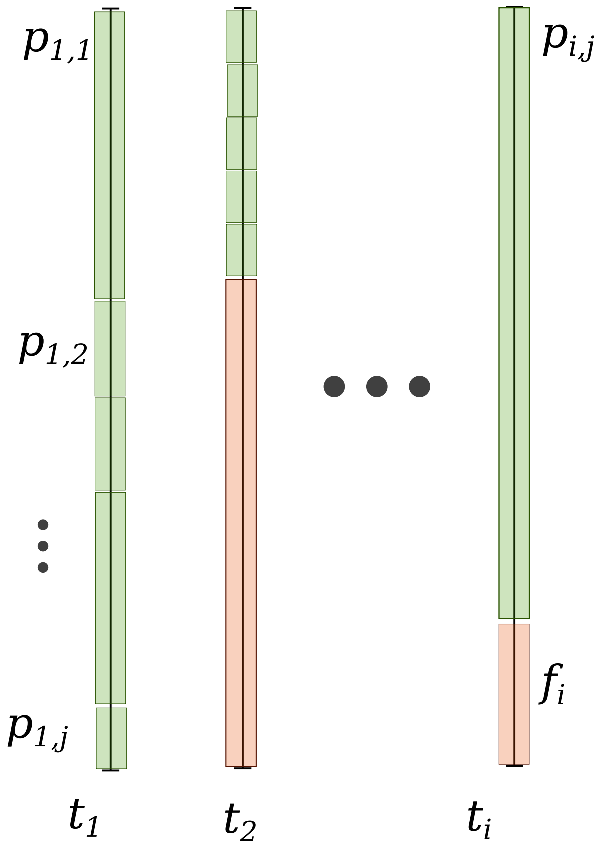

Figure 2.1: Example of suboptimal distribution of processes across processing threads. Green blocks represents processes while red blocks represents thread idle time. Threads are named $p_{i,j}$ where i is the number of the thread the process has been assigned to and f_i is the combined idle time for each thread. Threads are named t_i.

Chapter 3

Implementation

Chapter 4

Discussion

Appendix A

One appendix

foo

Appendix B

Two appendix

bar