Process Calculi

A Brief, Gentle Introduction

Jorge A. Pérez

university of groningen

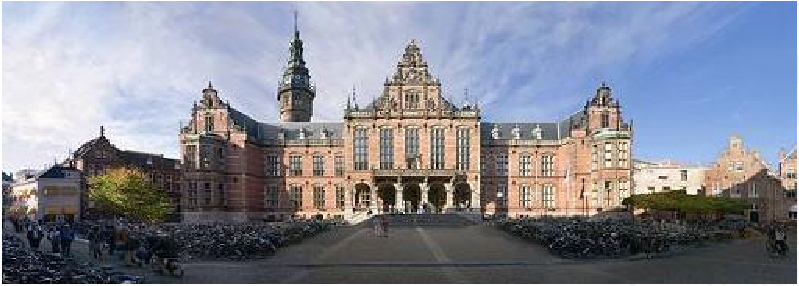
University of Brasilia
July 20, 2015

Outline

About Groningen

- 200000 inhabitants
- Cultural and economic capital of the northern Netherlands
- Vibrant student city
- About 2h from Amsterdam (high-speed railway connection)

University of Groningen



- Research-driven university, founded in 1614
- 30000 students / 5300 staff
- 9 Faculties and Graduate Schools
- #56 ESI Citations Impact
- #82 Academic Ranking of World Universities
- Partner of "Ciência sem Fronteiras" — see `www.rug.nl/swb`

Faculty of Math. and Natural Sciences

- 4200 students / 1350 staff (around 600 PhD students)
- 12 Research institutes - JBI (Mathematics and Comp Science)
- 13 Bachelor's degree programs (10 English-taught)
- 27 Master's degree programs

About Me

- Since 2014: Assistant professor, University of Groningen (NL).
- 2010-2014: Postdoc, Universidade Nova de Lisboa (PT), working with Luís Caires.
  Main topic: Types for Concurrency.
- 2007-2009: PhD student, University of Bologna (IT), under the supervision of Davide Sangiorgi.
  Main topic: Expressiveness of Process Calculi.
- 2005-2006: Undergraduate research assistant at Universidad Javeriana - Cali (CO), working with Camilo Rueda.
  Main topic: Declarative Models of Concurrency.

Outline

Presentation

Context and Motivation

Process Calculi

This Minicourse

Concurrency?

- "Things that happen at the same time"
- A set of agents or proceses which interact between them
- A concept that appears in many scenarios:
  - Biological systems
  - Social and human systems
  - Reactive systems
  - Distributed computing systems
  - ...
- In computer science, this is a relatively recent concept
- Computing originated and was developed from a sequential perspective (e.g., the notion of algorithm).

Concurrency?

- "Things that happen at the same time"
- A set of agents or proceses which interact between them
- A concept that appears in many scenarios:
  - Biological systems
  - Social and human systems
  - Reactive systems
  - Distributed computing systems
  - ...
- In computer science, this is a relatively recent concept
- Computing originated and was developed from a sequential perspective (e.g., the notion of algorithm).

Concurrency?

- "Things that happen at the same time"
- A set of agents or proceses which interact between them
- A concept that appears in many scenarios:
  - Biological systems
  - Social and human systems
  - Reactive systems
  - Distributed computing systems
  - ...
- In computer science, this is a relatively recent concept
- Computing originated and was developed from a sequential perspective (e.g., the notion of algorithm).

Concurrency?

- "Things that happen at the same time"
- A set of agents or proceses which interact between them
- A concept that appears in many scenarios:
  - Biological systems
  - Social and human systems
  - Reactive systems
  - Distributed computing systems
  - ...
- In computer science, this is a relatively recent concept
- Computing originated and was developed from a sequential perspective (e.g., the notion of algorithm).

Concurrency: A Challenge

Currently, a mismatch:

Information technologies
(predominantly concurrent and interactive)
vs.
computing foundations
(mostly sequential)

- Consequence: conceiving, designing, and implementing concurrent systems is difficult and error prone.
- These errors are often costly (even catastrophic) and have societal implications

Concurrency: A Challenge

Currently, a mismatch:

Information technologies
(predominantly concurrent and interactive)
vs.
computing foundations
(mostly sequential)

- Consequence: conceiving, designing, and implementing concurrent systems is difficult and error prone.
- These errors are often costly (even catastrophic) and have societal implications

Concurrency: A Challenge

Currently, a mismatch:

Information technologies
(predominantly concurrent and interactive)
vs.
computing foundations
(mostly sequential)

- Consequence: conceiving, designing, and implementing concurrent systems is difficult and error prone.
- These errors are often costly (even catastrophic) and have societal implications

Nasdaq glitches during Facebook's IPO

facebook NASDAQ UBS

THE WALL STREET JOURNAL.

UBS MAY HAVE \$350M FACEBOOK TRADE LOSSES (8/6/2012)

Technical problems kept investors from buying shares in the morning, or selling them later in the day, or even knowing whether their orders went through.

UBS placed an order for 1M shares but did not receive confirmations and repeated the order several times.
So UBS ended up with much more stock than it intended.

Which Concurrency?

A rough classification of the concurrency in computer science:

- Local concurrency
- Global concurrency

Local Concurrency

Focus on interaction that occurs in a shared state.

Multiple homogeneous tasks executing "nearby" with limited resources and operations.

Examples:

- A smartphone (or any mobile device)
- Multicore processors in modern tablets and laptops
- GPGPUs
- Concurrent data structures

Global Concurrency

Privileges the notion of computing as interaction, in distributed and highly dynamic scenarios.

Frequently found in systems which are built as the composition of heterogeneous components which communicate between them.

Examples:

- Twitter, Google, Facebook, Skype, ...
- Online services for booking flights and hotels
- Government information systems
- Web services, cloud computing, grid infrastructures

This talk: models for global concurrency

Global Concurrency

Privileges the notion of computing as interaction, in distributed and highly dynamic scenarios.

Frequently found in systems which are built as the composition of heterogeneous components which communicate between them.

Examples:

- Twitter, Google, Facebook, Skype, ...
- Online services for booking flights and hotels
- Government information systems
- Web services, cloud computing, grid infrastructures

This talk: models for global concurrency

Models for Concurrency

It is hard to specify and reason about the phenomena which are typical of concurrent computation.

This is because concurrent systems usually are

- Interactive/reactive
- Infinite
- Hard to predict (non deterministic)

Consequence: models/techniques for the design and construction of sequential systems are inadequate for concurrent systems.

Models for Concurrency

It is hard to specify and reason about the phenomena which are typical of concurrent computation.

This is because concurrent systems usually are

- Interactive/reactive
- Infinite
- Hard to predict (non deterministic)

Consequence: models/techniques for the design and construction of sequential systems are inadequate for concurrent systems.

Models for Concurrency

It is hard to specify and reason about the phenomena which are typical of concurrent computation.

This is because concurrent systems usually are

- Interactive/reactive
- Infinite
- Hard to predict (non deterministic)

Consequence: models/techniques for the design and construction of sequential systems are inadequate for concurrent systems.

Models for Concurrency

We require models tailored to concurrent computing.

In principle, we would like models which are at least

- general, based on a few, key principles
- expressive enough to represent relevant phenomena

Moreover, these models should be also precise and reliable.

To this end, we will find it reasonable to require models which

- are formal, that is, based upon solid mathematical foundations
- are endowed with reasoning techniques which allows us to discern about certain **aspects of interest**

Models for Concurrency

We require models tailored to concurrent computing.

In principle, we would like models which are at least

- general, based on a few, key principles
- expressive enough to represent relevant phenomena

Moreover, these models should be also precise and reliable.

To this end, we will find it reasonable to require models which

- are formal, that is, based upon solid mathematical foundations
- are endowed with reasoning techniques which allows us to discern about certain **aspects of interest**

Models for Concurrency

We require models tailored to concurrent computing.

In principle, we would like models which are at least

- general, based on a few, key principles
- expressive enough to represent relevant phenomena

Moreover, these models should be also precise and reliable.

To this end, we will find it reasonable to require models which

- are formal, that is, based upon solid mathematical foundations
- are endowed with reasoning techniques which allows us to discern about certain **aspects of interest**

Aspects of Interest?

There are many possible aspects that a formal model of concurrency may aspire to capture:

- communication discipline: point-to-point, broadcast
- synchronization mechanisms: synchronous, asynchronous
- message passing, shared variables
- timed (discrete, continuous) or untimed
- deterministic, or non deterministic
- ...

Following an abstraction principle, the intended models will focus only on a few aspects/concerns, ignoring the rest.

Aspects of Interest?

There are many possible aspects that a formal model of concurrency may aspire to capture:

- communication discipline: point-to-point, broadcast
- synchronization mechanisms: synchronous, asynchronous
- message passing, shared variables
- timed (discrete, continuous) or untimed
- deterministic, or non deterministic
- ...

Following an abstraction principle, the intended models will focus only on a few aspects/concerns, ignoring the rest.

Outline

Formal Models of Concurrency

We shall focus on a very concrete class of models of concurrency, the so-called process calculi (aka process algebras).

- Widely studied in the last 20-30 years.
- Formal languages in which the structure of terms represents (or reflects) the structure of computational processes
- Such a structure is given by a reduced set of process constructors
- Example: process $P \parallel Q$ represents the parallel execution of two processes P and Q which are combined using the $\parallel$ operator.
- Endowed with an operational semantics which represents steps of concurrent computation.
- Examples: CCS (Milner), CSP (Hoare), the π-calculus (Milner, Parrow, Walker), and several others.

Formal Models of Concurrency

We shall focus on a very concrete class of models of concurrency, the so-called process calculi (aka process algebras).

- Widely studied in the last 20-30 years.
- Formal languages in which the structure of terms represents (or reflects) the structure of computational processes
- Such a structure is given by a reduced set of process constructors
- Example: process $P \parallel Q$ represents the parallel execution of two processes P and Q which are combined using the $\parallel$ operator.
- Endowed with an operational semantics which represents steps of concurrent computation.
- Examples: CCS (Milner), CSP (Hoare), the π-calculus (Milner, Parrow, Walker), and several others.

Formal Models of Concurrency

We shall focus on a very concrete class of models of concurrency, the so-called process calculi (aka process algebras).

- Widely studied in the last 20-30 years.
- Formal languages in which the structure of terms represents (or reflects) the structure of computational processes
- Such a structure is given by a reduced set of process constructors
- Example: process $P \parallel Q$ represents the parallel execution of two processes P and Q which are combined using the $\parallel$ operator.
- Endowed with an operational semantics which represents steps of concurrent computation.
- Examples: CCS (Milner), CSP (Hoare), the π-calculus (Milner, Parrow, Walker), and several others.

Formal Models of Concurrency

We shall focus on a very concrete class of models of concurrency, the so-called process calculi (aka process algebras).

- Widely studied in the last 20-30 years.
- Formal languages in which the structure of terms represents (or reflects) the structure of computational processes
- Such a structure is given by a reduced set of process constructors
- Example: process $P \parallel Q$ represents the parallel execution of two processes P and Q which are combined using the $\parallel$ operator.
- Endowed with an operational semantics which represents steps of concurrent computation.
- Examples: CCS (Milner), CSP (Hoare), the π-calculus (Milner, Parrow, Walker), and several others.

Process Calculi: Some Features

- A compositional approach to specification:
  we define a concurrent system in terms of its sub-systems and the interaction between them
- The operators allow us to represent explicitly abstraction criteria in specifications
- They are defined as minimal models, able to represent interesting behaviors using a reduced set of elements.
- Typically, process calculi are able to represent
  - Atomic actions and their interaction
  - Explicit concurrency (e.g. $\parallel$)
  - Choices between different behaviors
  - Delimited interactions
  - Infinite behaviors (e.g. recursion)

Process Calculi: Some Features

- A compositional approach to specification:
  we define a concurrent system in terms of its sub-systems and the interaction between them
- The operators allow us to represent explicitly abstraction criteria in specifications
- They are defined as minimal models, able to represent interesting behaviors using a reduced set of elements.
- Typically, process calculi are able to represent
  - Atomic actions and their interaction
  - Explicit concurrency (e.g. $\parallel$)
  - Choices between different behaviors
  - Delimited interactions
  - Infinite behaviors (e.g. recursion)

Process Calculi: Some Features

- A compositional approach to specification:
  we define a concurrent system in terms of its sub-systems and the interaction between them
- The operators allow us to represent explicitly abstraction criteria in specifications
- They are defined as minimal models, able to represent interesting behaviors using a reduced set of elements.
- Typically, process calculi are able to represent
  - Atomic actions and their interaction
  - Explicit concurrency (e.g. $\parallel$)
  - Choices between different behaviors
  - Delimited interactions
  - Infinite behaviors (e.g. recursion)

Process Calculi: Purpose

- Basic models of concurrent computing (as the λ-calculus is the foundation of sequential programming)
- Formal foundations for modern programming languages and development tools
- Useful to define and study techniques for reasoning and verification
  - Simulators
  - Model checkers
  - Type systems

Outline

Process Calculi: This Minicourse

1. Basic notions on CCS and the π-calculus
2. Curry-Howard correspondences for the π-calculus
3. Expressiveness in concurrency

Process Calculi

A Brief, Gentle Introduction

Jorge A. Pérez

university of groningen

University of Brasilia
July 20, 2015