Models of Concurrent Computation

Philippa Gardner

http://www.doc.ic.ac.uk/~pg/

We meet in room 308 on Thursdays at 3-4 and in room 311 on Fridays at 4-6. I will give the first half of the course on CCS.

Nobuko Yoshida will give the second half of the course on the pi calculus.

Vasa Curcin and Rumyana Neykova will help with the tutorials.
We study **CCS**, a specification language for describing concurrent communicating processes.

It gives rise to model-checking tools for proving properties about concurrent systems.

For example, LOTUS used in circuit design.

We study **the pi calculus**, which evolved from CCS. It models the changing connectivity of interacting systems.

It forms the basis of languages for supporting distributed concurrent programming.

For example, it influenced the business process language BPEL.
Functional Behaviour

Turing machines, $\lambda$-calculus, recursive functions

$\lambda$-notation: $(\lambda x. x + x) \ 2 \rightarrow 2 + 2 = 4$

Haskell notation:

```
add :: Int -> Int
add x = x+x
```

...> add 2 apply add to 2

4 result

The behaviour of functions is determined by their results.
Processes interact continually with the environment and do not terminate. Systems exhibiting such interactive behaviour include:

- an operating system
- a telephone service
- a railway system
- biological cells

Even if a system as a whole has behaviour based on inputs and outputs, the constituent parts need not.
Finite Automata

Vending machine

Deterministic automata

Non-deterministic automata
Automata theory regards the behaviour of these automata as the same.
CCS

Some similarities with finite automata (labelled graphs, no distinguished final states).

Major differences

Process behaviour is different—our theory will distinguish the two Vending machines.

Interaction between processes is central—no interaction between individual automata.

The primitive interaction is to communicate with a handshake:

handshake on channel 1p

\[
\overline{1p}.P \mid 1p.Q \rightarrow P \mid Q
\]

handshake on channel send, passing message \( m \) to \( x \)

\[
\overline{send}\langle m \rangle.P \mid send(x).Q \rightarrow P \mid Q\{m/x\}
\]
Example: ISO standard CCR service

The service consists of one controller and several servers. All servers carry out a request by the controller or none do. CCS can specify the following concurrent interaction:

- the controller sends requests to all the servers;
- each server sends a reply to the controller saying whether or not it can carry out the request;
- if a server is able and the controller wishes to proceed, then the controller tells the server to prepare to carry out the request; if not, the controller tells the server to ignore the request;
- if the controller tells the server to act, the server carries out the request and then sends an acknowledgement to the controller.

This example will be an exercise in the course.
What does it mean to give a good CCS specification of the ISO standard CCR service?

We study

- a modal logic for describing properties: for example, either all services act or none of them act
- process behaviour: in particular, equivalences for determining when two processes behave in the same way
- the Concurrency Workbench, a tool for checking properties of processes.

**Fact**

Two processes behave in the same way if and only if they satisfy the same formulae.
The Pi Calculus

The pi calculus evolved from CCS.

Calculi based on the **pi calculus** provide primitives for describing and analysing global distributed infrastructure, including:

**Process migration between peers:**

\[
P \quad \text{migrate} \quad m. \quad P_1
\]

**Process interaction via dynamic channels:**

\[
\begin{align*}
P & = \text{migrate} \quad m. \quad P_1 \\
P_1 & = \overline{\text{send}}(p) \\
Q & = \text{send}(x).\overline{x}(\text{data}) \\
P_1 | Q & \rightarrow \overline{p}(\text{data})
\end{align*}
\]

**Private channel communication:**

\[
(\text{new} \; p) \; (P | R)
\]
Impact of the Pi calculus

Distributed Programming

It inspired the remote communication constructs of JoCaml (Inria, Paris) and Polyphonic C♯ (Microsoft Research Cambridge).

Web service orchestration

It is the basis of XLang, BPML and WS-CDL.

Security

The applied pi calculus is used to describe and analyse security protocols.

Systems Biology

The pi calculus is used to describe features of cell behaviour.

Academic

Lots of research, especially in Imperial, UK and Europe.
Pi Summary

The pi calculus part of the course includes

- the basic calculus

- the session calculus, an extension of the pi calculus to describe and verify web service (business and financial) protocols
Assessment

- An assessed exercise sheet on CCS, distributed on Tuesday 30th October. The answers must be submitted by Monday 12th November at 4 o’clock.

- An assessed exercise sheet on the pi calculus.

- A standard exam paper in the summer term: 2 hours, choose 3 questions out of 4.

In addition, unassessed exercises will be given throughout the course.