6CM504 Concurrency and communication

Understanding and modelling concurrency

Prof. Alistair A. McEwan

School of Computing
University of Derby

Table of Contents

1. Introduction
2. Introducing channels of communication
3. More about external choice
4. Determinism versus nondeterminism
5. More about choice: conditional choice
6. Some familiar processes
7. Summary

This week

So far we have looked at the basic building blocks of CSP—alphabets, events, and simple processes involving prefixing and choice. We developed a notion of *prefix choice* where we could offer events from a set of events.

This week we will develop this notion further—into one of communications, and a theory of inputs and outputs. This will then allow us to consider different types of choice, and introduce a theory of *nondeterminism*, allowing us to build much more sophisticated and descriptive models.

Table of Contents

1. Introduction
2. Introducing channels of communication
3. More about external choice
4. Determinism versus nondeterminism
5. More about choice: conditional choice
6. Some familiar processes
7. Summary

Introducing channels of communication

- We can extend the notion of prefix choice to allow us to describe channels of communication between processes. This will allow us to communicate higher order information between different processes in our models.
- If `c` is a channel name, and `T` is the type of an object communicated over the channel, we have

  $\texttt{c.T = }\{\texttt{c.x | x:T}\} \subset \Sigma$

- We can use this to think of processes inputting and outputting values of type `T` over channel `c`

Channels: input and output

- An inputting process would typically be able to communicate any element of `c.T`—that is to say, it does not restrict the input it receives

```
c?x:T -> P(x)
```

- An outputting process will typically communicate a single element of `c.T`—that is to say, the specific value it wishes to output

```
Copy = left?x:T -> right!x -> Copy
```

Table of Contents

1. Introduction
2. Introducing channels of communication
3. More about external choice
4. Determinism versus nondeterminism
5. More about choice: conditional choice
6. Some familiar processes
7. Summary

Recap: choice

- Recall that `P [] Q` (pronounced "P external choice Q") offers the environment the initials of `P` and the initials of `Q` and then behaves accordingly
- If the first event chosen by the environment is from the initials of `P` *only* then `P [] Q` behaves as `P`
- If the first event chosen by the environment is from the initials of `Q` *only* then `P [] Q` behaves as `Q`

External choice: example

- Thus the example process

```
(a -> A) [] (b -> B)
```

offers the environment the option of communicating `a`, after which it behaves as the process `A`, or event `b`, after which it behaves as the process `B`

External choice: introducing laws

- A fundamental aspect of CSP is that there are (algebraic) laws that allow us to reason about processes
- We will explore a few important laws in this module
- A very important property of external choice is shown in the following law

```
P = P [] STOP
```

- That is, to offer the environment the actions of `STOP` in addition to those of `P` has no effect on `P`: it is exactly the same as if we only offered `P`
- Question: can you explain intuitively why this is the case? Can you think of an analogy in arithmetic?

External choice: a puzzle!

- Our explanation of external choice assumed that the initial events were all different
  - In our example these were events `a` and `b`
  - And the subsequent behaviours were also different
- Question: what is going to happen in the following case, where the initial events `a` are not different?

```
(a -> A) [] (a -> B)
```

- Hint: when the environment agrees to communicate event `a`, what subsequent behaviour will be available?

External choice: a clearer puzzle!

- Consider the same question, but with the following processes:

```
(a -> a -> STOP) [] (a -> b -> STOP)
```

- Clearly, this process initially offers to communicate an `a`
- Question: after the initial `a`, it is free to offer either `a` or `b` but *not both*. How is this resolved?
- Question: how is this different to the following process?

```
a -> (a -> STOP [] b -> STOP)
```

Table of Contents

1 Introduction

2 Introducing channels of communication

3 More about external choice

4 Determinism versus nondeterminism

5 More about choice: conditional choice

6 Some familiar processes

7 Summary

Nondeterminism

- A *deterministic* process is one where the choice of events offered to the environment depends only on the things it has seen
- A process in *nondeterministic* when some internal decision leads to uncertainty about what may be offered
- Nondeterminism is a natural phenomenon—but we can identify it and reason about it cleanly and rigorously in CSP
- The distinction is important, and one we will later rely on in constructing specifications
  - We generally think of it as a specification construct, not an implementation one

There is nothing imprecise about nondeterminism!

More about choice: internal choice

- A nondeterministic choice between two processes `P` and `Q` is written as `(P |~| Q)`
- This is a process that can offer to behave either as `P` or as `Q`, but the environment cannot affect the choice of which is available
- When we communicate with the process we find that either `P` or `Q` is unavailable

External versus internal

- It is important to appreciate the difference between external and internal choice

```
ExtChoiceExample = a -> STOP [] b -> STOP

IntChoiceExample = a -> STOP |~| b -> STOP
```

- The process `ExtChoiceExample` is required to communicate either an a or a `b`, irrespective of which is offered
- The process `IntChoiceExample` will refuse to communicate either a or `b`, but we have no way of knowing which

Table of Contents

1. Introduction
2. Introducing channels of communication
3. More about external choice
4. Determinism versus nondeterminism
5. More about choice: conditional choice
6. Some familiar processes
7. Summary

Conditional choice

- To make out processes useful models of real world entities, we allow identifiers (variables) through input and process parameters
- This permits us to consider another form of choice: one where the availability of an event is determined based on the value of boolean expression
- These can be thought of as analogous to "if then" statements in programming

Conditional choice: an example

- This example descibes a counter that can move around a chess board
  - It has two parameters—representing the x and y co-ordinates
  - These parameters must be restricted to the range $0 \leq i \leq 7$
  - One will be changed by the events {`up`,`down`}, and the other by {`left`,`right`}

```
Counter(x,y) =
  x > 0 & down -> Counter(x-1, y)
  []
  x < 7 & up -> Counter (x+1,y)
  []
  y > 0 & left -> Counter (x,y-1)
  []
  y < 7 & right -> Counter(x,y+1)
```

Why does this example work?

- Just for clarity, lets translate it into an analogous `if then` syntax

```
C(x,y) =
  if(x>0) then down -> C(x-1,y) else STOP
  []
  if(x<7) then up -> C(x+1,y) else STOP
  []
  if(y>0) then left -> C(x,y-1) else STOP
  []
  if(y<7) then right -> C(x,y+1) else STOP
```

- Question: how does this relate to our unit law for STOP?

Table of Contents

1. Introduction
2. Introducing channels of communication
3. More about external choice
4. Determinism versus nondeterminism
5. More about choice: conditional choice
6. Some familiar processes
7. Summary

RUN and CHAOS

- There are a few common processes with very basic behaviour that we will see repeatedly

  ```
  RUN(A) = ?x:A -> RUN(A)
  ```

  is the process that for a set of events `A` can always communicate any member of `A`

  ```
  CHAOS(A) = STOP |~| ?x:A -> CHAOS(A)
  ```

  can choose to communicate or reject any member, or even all of, of `A`

RUN and CHAOS

- `RUN({})` and `CHAOS({})` are equivalent to `STOP`
- Question: why is this the case?
- Later, we will see two other important processes `div` and `SKIP`

Table of Contents

1. Introduction
2. Introducing channels of communication
3. More about external choice
4. Determinism versus nondeterminism
5. More about choice: conditional choice
6. Some familiar processes
7. Summary

Summary

- In this lecture we have expanded our notion of events into one of channels
- Channels allow us to communicate higher order information between processes. We have seen how external choice describes what channels may or may not be available, and used this to construct a theory of input and output
- Our definition of choice enabled us to introduce a theory of nondeterminism
- In the next lecture we shall use the constructs introduced so far to build and reason about models, and develop these models in the FDR tool

Summary

- Advance reading: Schneider, pages 467–482
- Advance reading: Roscoe, pages 423–429
- Advance reading: Woodcock and Davies, chapter 3