Lecture 6: Coordination Pt.2

Adam Hawley

February 5, 2019

Contents

1 Shared Variable Coordination **1**
- 1.1 Monitors . . . 1
- 1.2 Pascal-FC Implementation of Monitors . . . 1
- 1.3 Ada Implementation of Monitors . . . 2
  - 1.3.1 Barriers . . . 2
  - 1.3.2 Attributes . . . 2

1 Shared Variable Coordination

1.1 Monitors

A monitor is a control structure that provides **mutual exclusion**. It can also be considered to be "An extension of the conditional critical sections".

1.2 Pascal-FC Implementation of Monitors

See lecture for live coding.

Pascal-FC uses a low-level condition variable mechanism with three operators:

delay immediately blocks caller task, and releases the monitor

resume makes at most one blocked task runnable again

empty return true if no blocked tasks exist

1.3 Ada Implementation of Monitors

In Ada, monitors are called **protected types**. There are no condition variables like there are in Pascal-FC. Instead, synchronisation comes from the use of barriers (guards).

Specifications of protected types can contain:

- Procedures executed under **mutual exclusion**
- Pure functions that are **read only** can execute concurrently with other functions.
- Entries specify **barriers** (or guards).
- Private data of any type.

1.3.1 Barriers

Barriers are boolean expressions that must evaluate to `true` for the entry to execute. If `false`, the task is blocked. Barriers are re-evaluated on task entry and exit from the **protected object** (**PO**). At most one task can proceed through the barrier.

1.3.2 Attributes

Attributes in Ada give information about the behaviour of the program, e.g:

```
entry await when await ‘count = 4 is ...
```

This releases one task (non-deterministically) when there are 4 waiting, otherwise blocks.