Solutions for CP Exercises, September 15

1. Solution for Andrews Ex. 3.2

```
var $l$ : $integer$ := 1;

process $P[i : 1..n]$ =
  var $s$ : $integer$;
  repeat
    non-critical section$_i$;
    $DEC(l, s)$;
    while $s > 0$ do {
      $INC(r, l)$;
      delay;
      $DEC(l, s)$;
    }
    critical section$_i$;
    $INC(l, s)$;
  forever;
```

Here, the lock l is used as in the test-and-set solution. However, if the lock is already "set" ($l < 1$), the effect of DEC must be undone by INC, before trying again. The correctness argument (or proof) follows the same line as for the test-and-set solution.

2. Solution for Andrews Ex. 3.13

A first attempt to be able to "use the barrier again" would be to let the last processes that arrives reset the counter thereby releasing everybody:

```
var $count$ : $integer$ := 0;

process $Worker[i : 1..n]$ =
  repeat
    code to implement task $i$;
    $\langle\, count := count + 1 \,\rangle$;
    if $count = n$ then $count := 0$;
    $\langle$ await $count = 0 \,\rangle$;
  forever;
```

In this proposal, however, the processes first released by the reset may race to the next barrier round and increment *count* before all processes have seen $count = 0$. The result is a deadlock.

The solution is to use two counters ensuring that the first counter is not incremented until everybody have seen the reset and vice versa:

```
var $count_1, count_2$ : $integer$ := 0;
```

```
process $Worker[i : 1..n]$ =
  repeat
    code to implement task $i$;
    $\langle\, count_1 := count_1 + 1 \,\rangle$;
    if $count_1 = n$ then $count_1 := 0$;
    $\langle$ await $count_1 = 0 \,\rangle$;
    $\langle\, count_2 := count_2 + 1 \,\rangle$;
    if $count_2 = n$ then $count_2 := 0$;
    $\langle$ await $count_2 = 0 \,\rangle$;
  forever;
```

Using the *fetch-and-add* instruction, this can be readily implemented:

```
process $Worker[i : 1..n]$ =
  repeat
    code to implement task $i$;
    if $FA(count_1, 1) = n - 1$ then $count_1 := 0$;
    while $count_1 \neq 0$ do skip;
    if $FA(count_2, 1) = n - 1$ then $count_2 := 0$;
    while $count_2 \neq 0$ do skip;
  forever;
```