Principles of software composition 2017/18

Mid-term exam – May 31, 2018

[**Ex. 1**] Consider the HOFL term

$$t \stackrel{\text{def}}{=} \mathbf{rec}\ f.\ \lambda x.\ \mathbf{if}\ x\ \mathbf{then}\ (x, \mathbf{fst}(f\ x))\ \mathbf{else}\ (1,2)$$

1. Find the principal type of t.
2. Compute the canonical form of the term $\mathbf{fst}(t\ 0)$.
3. Compute the (lazy) denotational semantics of t.

[**Ex. 2**] Let $\simeq$ denote strong bisimilarity and $\approx$ be weak bisimilarity. Given an action α:

1. Find two CCS processes p, q such that

$$(p|q)\backslash\alpha \not\simeq p\backslash\alpha|(q\backslash\alpha) \quad \text{but} \quad (p|q)\backslash\alpha \approx p\backslash\alpha|(q\backslash\alpha)$$

2. Find two (non inactive) CCS processes r, s such that

$$(r|s)\backslash\alpha \simeq r\backslash\alpha|(s\backslash\alpha)$$

[**Ex. 3**] Two elevators e_1, e_2 serve three floors f_1, f_2, f_3. Consider the atomic propositions (with with $i \in [1,2]$ and $j \in [1,3]$):

$at_{i,j}$: holds when elevator e_i is at floor f_j;
up_i: holds when elevator e_i is moving upwards;
$down_i$: holds when elevator e_i is moving downwards.

1. Write the property "whenever e_1 moves up, it will stop at f_2 or f_3 before it can move down" in LTL.
2. Write the property "it is possible to find both elevators at f_2" in CTL.
3. Write the property "it is always the case that if e_1 moves up then e_2 goes down or is at f_1" in the μ-calculus.

[**Ex. 4**] A process is *sequential* if it is written without using parallel composition. Consider the π-calculus process (with x, y, z pairwise different)

$$p \stackrel{\text{def}}{=} \overline{x}y.\mathbf{nil} \mid x(z).\mathbf{nil}$$

Write a sequential process q that is strongly (early and late) bisimilar to p.

[**Ex. 5**] Write a GoogleGo function `Split` that takes three channels `c`, `c1`, `c2` for passing integers and forwards each message arriving on `c` to either `c1` or `c2` depending on which one is ready to receive. When `c` is closed, `Split` closes `c1` and `c2` and terminates. When writing the function, type the channels according to their usages.