2G1517 Advanced Formal Methods

TAKE HOME EXAM | *Mads Dam*
18 April 2006, 09.00–17.00 | KTH/CSC

Give solutions in English or Swedish, each problem beginning on a new sheet. Write your name on all sheets. The maximal number of points is given for each problem. The exam is pass or fail. Cut-off level is 50. Collaboration is not allowed. Answers to be handed in Tuesday 18 April 2006 at 17.00 at the latest, either physically, or by email to `mfd@kth.se`.

---

1. Recall that concurrent processes (def. 4.1 in Milner's book) is defined using a generalized guarded sum construction Σ. Prove that the familiar CCS sum construct, $+$, is not definable in the language of concurrent processes. **[20p]**
(Hint: Try to find a property which is preserved by all constructs in the language of concurrent processes, but which is violated by $+$.)

---

2. The data type of strings has three constructors: ε is the empty string, $[a]$ is the string consisting of the single letter a, and $s_1 \cdot s_2$ is string concatenation. Give a π-calculus representation of the (ephemeral) string data type, show how it can be sorted, and prove that string concatenation is associative in your representation (up to weak bisimulation equivalence). **[20p]**

---

3. A ring buffer consists of a number of cells, organized in a ring topology as indicated in the example on figure `http://www.imit.kth.se/courses/2G1517/05-06/www/ringBuffer.{ps,gif}`. One cell is the buffer start cell, another is the buffer end cell. Cells located between the start and end cells (both included) hold a value. Whether this statement is interpreted clockwise or counterclockwise is up to you. The ring has three external ports: *get* retrieves the value at the end of the buffer and updates the buffer accordingly, *put* appends a value to the front of the buffer, and *alloc* allocates a new empty cell. No garbage collection facility is required (but you are free to add one if you want). You are also free to make other sensible design decisions. Code such a ring buffer in π-calculus. **[20p]**

---

4. Formalize the property "Infinitely often an a and a b action are simultaneously enabled" as a modal mu-calculus formula ϕ. Prove that ϕ holds of the process term *new* c $(P|Q)$ where $P = c.P + a.P$ and $Q = (\overline{c}.\overline{c}.Q) + (b.Q)$ **[20p]**

Please turn over!

5. Let f and g be monotone function on a complete lattice A. Assume that $f(a) \leq g(a)$ for all $a \in A$. Prove using fixed point induction that $\mu f \leq \mu g$. 20p

Good luck!