Concurrent Algorithms December 8, 2020

Exercise 10

(Given as an exam problem in 2016-2017)

Problem 1. Consider the following implementation of an obstruction-free consensus object from atomic multi-valued MRMW shared registers in a system of n processes. A process's id is known to itself as i.

```
Using: an array of atomic multi-valued MRMW shared registers T[1, 2, ..., n],
       initialized to 0;
Using: an array of atomic multi-valued MRMW shared registers V[1, 2, ..., n],
       initialized to (⊥, 0);

propose(v) {
    ts := i;
    while (true) do{
        T[i].write(ts);

        maxts := 0;
        val := ⊥;
        for j = 1 to n do
            (t, vt) := V[j].read();
            if maxts < t then
                maxts := t;
                val := vt;

        if val = ⊥ then val := v;

        maxts := 0;
        for j = 1 to n do
            t := T[j].read();
            if maxts < t then maxts := t;

        if ts = maxts then
            V[i].write(val, ts);
            return(val);
        ts := ts + n;
    }
}
```

Recall that obstruction-free consensus ensures the property of *obstruction-freedom* instead of *wait-freedom*.

Your tasks:

1. Explain what is obstruction-freedom and what is the difference between obstruction-freedom and wait-freedom.
2. Answer whether the implementation satisfies obstruction-freedom. Justify your answer.
3. Is the algorithm correct? If the algorithm is correct prove its correctness. Otherwise, provide an execution that shows the algorithm is incorrect.