EDA387: Computer Networks - Lab 2.3

Mats Högberg

October 3, 2019

Self-stabilizing vertex coloring

Here's the pseudo code for a self-stabilizing algorithm for vertex coloring in the given graph:

```
01 i = 1: do forever:
02          lr_n := read r_n
03          color_i := 1 if lr_n = 0 else 2
04          write r_i := color_i
05        end

06 i ≠ 1: do forever:
07          lr_{i-1} := read r_{i-1}
08          color_i := 0 if lr_{i-1} = 1 or lr_{i-1} = 2 else 1
09          write r_i := color_i
10        end
```

In this algorithm, the root sets its color to either 1 or 2, depending on the color of node n. All other processors look at the processor prior to them in the ring, and set its color to 0 if the neighbour's color is 1 or 2, and 1 otherwise.

For the task of vertex coloring, the set of legal executions are all executions in which, in every configuration, the values of registers of neighbouring processors are not equal. A configuration $c = \{r_1, r_2, \ldots, r_n\}$ is safe with regard to the set of legal executions and the above algorithm if:

$$r_1 = \begin{cases} 1, & \text{if } n \bmod 2 = 0 \\ 2, & \text{otherwise} \end{cases}$$

$$\forall i, i \neq 1 : r_i = i \bmod 2$$

For example, for $n = 5$ we would have $c = \{2, 0, 1, 0, 1\}$ and for $n = 6$ we would have $c = \{1, 0, 1, 0, 1, 0\}$. If a system is in such a configuration, the values of registers of neighbouring processors are not equal, and no register will ever change its value. It is therefore a safe configuration.

To prove that the algorithm will end up in this configuration, regardless of the initial configuration of the system, we will start by taking a closer look at p_1. Assuming read/write atomicity, p_1 will write either $r_1 = 1$ or $r_1 = 2$ within 2 asynchronous rounds, depending on the value of r_n. After 4 rounds, p_2 will have read the value written to r_1 and written $r_2 = 0$. An important thing to note here is that p_2 will write $r_2 = 0$ regardless of whether $r_1 = 1$ or $r_1 = 2$. After this happens, the value of r_2 will never change, since the value of r_1 will only ever be set to either 1 or 2.

After 6 rounds, p_3 will have read $r_2 = 0$ and written $r_3 = 1$. Once again, since the value stored in r_2 will never change after 4 rounds, the value of r_3 will never change again after this. This pattern will continue,

and within $2n$ rounds we will have $\forall i, i \neq 1 : r_i = i \bmod 2$, and most notably we will have $r_n = n \bmod 2$. Within 2 additional rounds, p_1 will have read $r_n = n \bmod 2$ and written $r_1 = 1$ if $r_n = 0$ or $r_1 = 2$ otherwise. At this point, the system will be in the safe configuration described above, and thus we have shown that the algorithm will stabilize within $2n + 2$ rounds.

This algorithm will use 2 colors if $n \bmod 2 = 0$ and 3 otherwise. This is the optimal number of colors for vertex coloring in a ring. As for the states of the processors, each processor can be in one of three states: unknown (its starting state), set to an incorrect color (which can happen before the system has stabilized), or set to its correct color (which is its color in the safe configuration).