dConc Aflevering 4 - DA6

Christian Zhuang-Qing Nielsen

201504624, christian@czn.dk

December 3, 2016

Step 4

How can you express the notions of first-come-first-served and 1-bounded and 2-bounded waiting in LTL? (Hint: you need to use the U or W operators).

Jeg har valgt at udtrykke First-in-first-out (FIFO), 1- og 2-bounded waiting vha. labels i promela-koden, der gør det nemmere at beskrive hvor trådene er på givne tidpunkter.

For FIFO if we create a section in the promela code in the beginning, we can make test when one of the threads is at the end of the doorway labeled *door*, while we know the other one is at the beginning labeled *play*.

Jeg skabt en sektion i starten af promelakoden kaldet *play*, således kan vi teste hvor vi ved at p_0 er i slutningen af doorway'en med ediketten *label*, mens p_1 er i *play*. Så kan vi definere at dette implikerer at p1 ikke kan være ved critical section (ediket *cs*), indtil før p_0 har været i sin, hvilket er følger definition af fifo (eftersom p_0 fuldender sin doorway først). LTL input for spin er i dette tilfælde:

$$G((p_0@door \ \wedge \ p_1@play) \implies (\neg p_1@cs \cup p_0@cs))$$

1-bounded waiting betyder at hvis en tråd gennemfører sin doorway før en anden given tråd, så vil den første tråd maksimalt blive overhalet én gang før den får lov til at køre sin cs. Dette viser jeg ved at "neste" et ekstra lag på formen (!P W (P W r_i)), hvor i = antallet af gange den må vente. For 1-bounded waiting vil det derfor se ud som følgende:

$$G((p_0@door \wedge p_1@lock) \implies (\neg p_1@cs \, W \, ((p_1@cs) \, W \, ((\neg(p_1@cs)) \, W \, (p_0@cs)))))$$

2-bounded waiting is in my opinion just another step. So I add another nesting. This looks as follows:

$$G((p_0@door \ \wedge \ p_1@lock) \implies (\neg(p_1@cs) \ W \ ((p_1@cs) \ W \ (\neg(p_1@cs)$$

$$W\ ((p_1@cs)\ W\ ((\neg(p_1@cs))\ W\ (p_0@cs)))))))$$

Step 5

Let Spin solve AMP, exercise 9 for you, for the two different natural definitions of doorway of Peterson's algorithm (1. the doorway is the code up to and including the first write to a shared variable, i.e., "flag[i] = true" and 2. the doorway is the code up to and including "victim = i").

Jeg har omskrevet peterson-algoritmen fra bogen om til Promela. Herefter har jeg brugt ovenstående LTL-formler (Efter promela-syntaks, f.eks G = [], $\wedge$ = osv.) og kørt dem med min peterson algoritme. I min 1-bounded har skrevet den således at den tjekker worst case scenario (hvor den ene lige er blevet færdig med doorway og den anden lige skal til at gå ind i sin doorway). Jeg har her (i FIFO og 1-bounded) fået ingen fejl i Spin ved at køre de pågældende LTL-input på Promelafilen.

Jeg har ikke kunnet køre min LTL-formel for 2-bounded waiting eftersom min bærbar ikke har nok udregningskraft til dette. Kørselstiden for Spin er jo $O(4^{|\phi|} * |M|$, hvor ϕ er LTL-formlen og M er det underliggende transitionssystem. Der er også lidt ekstra kommentering i den vedhæftede kode. LTL-formlerne er udkommenteret, så hvis de skal køres skal de afkommenteres.