Bonus Material: Proof of Mutual Exclusion by Induction for Third Attempt

Material not Examinable

Third Attempt

boolean wantp ← false, wantq ← false	
p	q
`loop_forever`	`loop_forever`
`p1:  non-critical section`	`q1:  non-critical section`
`p2:  wantp = true`	`q2:  wantq = true`
`p3:  await wantq = false`	`q3: await wantp = false`
`p4:  critical section`	`q4:  critical section`
`p5:  wantp ← false`	`q5:  wantq ← false`

- Mutual Exclusion didn’t work so let’s swap the order.
- Now let’s try to prove mutual exclusion.

Inductive Proof of Mutual Exclusion

- Enumeration of state diagrams can be cumbersome if done by hand.
  - State diagrams get big very quickly for large programs.
- This time we will use proof by induction to show mutual exclusion for this algorithm.
  - Similar to mathematical induction on the natural numbers:
    - Show that the property is true for 1;
    - Show that, assuming it is true for n it holds for n+1;
    - By induction it is true for all natural numbers.

Mathematical Induction Example

- Show that: $1 + 3 + 5 + \ldots + (2n - 1) = n^2$
- Base case show for 1:
  - LHS = 1; RHS = $1^2 = 1$.
- Assume the formula holds for k;
  - $1 + 3 + 5 + \ldots + (2k - 1) = k^2$
- Given the assumption, show for k+1:
  - Show that: $1 + 3 + 5 + \ldots + (2k - 1) + (2(k+1) - 1) = (k+1)^2$
  - Use the assumption: $k^2 + (2(k+1) - 1) = (k+1)^2$
  - Expand Brackets: $k^2 + (2k + 2 - 1) = k^2 + 2k + 1$
  - $k^2 + 2k + 1 = k^2 + 2k + 1$: indeed it does, so the formula is true for k+1.

- Formula is true for 1, and if true for k, true for k+1, therefore true for 2.
- Formula is true for 2, and if true for k, true for k+1, therefore true for 3.
- Ad infinitum…

Inductive Proofs on Programs

- Induction on program states:
  - Slightly different variant of the same idea:
    - Show that the property holds for the initial state;
    - Show that if the property holds for a state it holds for all subsequent states.

- We will use propositional logic:
  - Propositions e.g. *p1* indicate the position of the program counter in a given state (i.e. we are executing p1);
  - Each process has a single program counter, therefore we assume that $pi \rightarrow \neg pj \; \forall \; j \neq i$.
  - Propositions e.g. *wantp* (*turn=1*) are true if the corresponding variable is true (holds the value 1) in that state.

- To prove mutual exclusion in our algorithm we have to show that for all states: $\neg(q4 \wedge p4)$.
  - That is in no state both programs are in their critical section.
  - We will start with something easier and then use that to prove the above: $p3 \vee p4 \vee p5 \rightarrow wantp$
    - Recall we want to prove this is true in all states.

Base Case: $p3 \vee p4 \vee p5 \rightarrow wantp$

boolean wantp ← false, wantq ← false	
p	q
`loop_forever`	`loop_forever`
`p1:  non-critical section`	`q1:  non-critical section`
`p2:  wantp ← true`	`q2:  wantq ← true`
`p3:  await wantq = false`	`q3: await wantp = false`
`p4:  critical section`	`q4:  critical section`
`p5:  wantp ← false`	`q5:  wantq ← false`

- Initial state (p1,q1,false).
  - Trivially true since $p3 \vee p4 \vee p5$ is false and false $\rightarrow$ anything is true.

Inductive Step

- Assume formula true: $p3 \vee p4 \vee p5 \rightarrow wantp$

- q cannot change the truth of the formula because it depends only on the program counter for p and the value of *wantp*, which is only ever modified by p.

- The only statements we need to check are those that could affect the truth value of the formula (make the antecedent true): p2 and p5.
  - Executing p2 (`wantp ← true`) makes $p3 \vee p4 \vee p5$ true, but also makes *wantp* true, so the formula remains true.
  - Executing p5 (`wantp ← true`) makes $p3 \vee p4 \vee p5$ false and therefore the formula is true, regrardless of the value of *wantp*.

- Therefore we have shown that $p3 \vee p4 \vee p5 \rightarrow wantp$ is always true (symmetric proof applies for $q3 \vee q4 \vee q5 \rightarrow wantq$)

The Reverse

- We showed: $p3 \vee p4 \vee p5 \rightarrow wantp$, now we will show $wantp \rightarrow p3 \vee p4 \vee p5$.

- Base case: (p1,q1,false), trivially true, antecedent false;

- The only statements that can falsify this are:
  - p2, which makes *wantp* true but it also makes the consequent ($p3 \vee p4 \vee p5$) true, so the statement remains true.
  - P5, which makes ($p3 \vee p4 \vee p5$) false, so the formula is trivially true following its execution.

- Now we have (by symmetry):
  - $p3 \vee p4 \vee p5 \leftrightarrow wantp \;\wedge\; q3 \vee q4 \vee q5 \leftrightarrow wantq$

Now to Prove Mutual Exclusion

- Recall we are proving ¬(q4 ^ p4) holds in all states, that (q4 ^ p4) is false in all states.

- Base case: initial state (p1,q1,false) therefore q4 ^ p4 is false.

- Only two statements could make it become true:
  - Completed execution of p3 (**`await wantq = false`**) while at q4 or q3 (**`await wantp = false`**) while at p4. These are symmetric so let's choose p3.
    - p3 can only complete execution if *wantq* is false
    - But we already showed that p*3* ∨ p*4* ∨ p*5* ↔ *wantp, so if we are at p3 then wantp is true.*
    - Therefore p3 cannot complete execution whilst in q.

- Therefore q4 ^ p4 is false in all states and mutual exclusion holds.

Mutual Exclusion for Peterson's Algorithm

- Show $p3 \vee p4 \vee p5 \leftrightarrow wantp \;\wedge\; q3 \vee q4 \vee q5 \leftrightarrow wantq$ just as above.
- We can also show last = 1 or last = 2 by inspection of the program.
- Now we want to show: $(p4 \wedge q5) \rightarrow (wantq \wedge last = 1)$.
  - Base case: initial state (p1,q1,1,false,false) antecedent false, therefore statement true.
- Only transition in p that can falsify is execution of p3 when $(wantq \wedge last = 1)$ is false.
  - But $q3 \vee q4 \vee q5 \leftrightarrow wantq$ so *wantq* is true, and executing p3 sets last to 1, so this is not possible.
- Transitions in q:
  - p4 holds and q4 is executed: wantp is true (because $p3 \vee p4 \vee p5 \leftrightarrow wantp$); therefore q4 can only complete if last = 1.
  - $(wantq \wedge last = 1)$ can become false if executing q3 or q6. This also means q5 is false, meaning the statement still holds.
- Mutual exclusion holds because if p4 and q5 hold, p4 cannot complete since its condition is the negation of $(wantq \wedge last = 1)$.