Sara Subedi - V00986656
Julia Hoang - V00974641
Angus Bews - V00980371

# Project 3 — SMV Model Checking

## phil.smv (basic solution)

The strategy we chose for this task was to simply have a variable noting which philosopher currently had permission to eat

**permission : {0,1,2,3,4};**

This only allows for one philosopher to be eating at a time so this is a sequential solution.
We initialize permission to philosopher 0 to start.

**init(permission) := 0;**

From there, we only allow a given philosopher to pick up their respective chopsticks if they have permission.

We also have to deal with passing permission onto the next philosopher by simply saying that if the current philosopher is thinking or done eating, pass the permission on to the next philosopher.

-- If the philosopher is thinking but has permission, pass it over to the next philosopher
**(state=thinking) & chosen : (i+1) mod 5;**
-- If the philosopher finished eating, pass permission to the next philosopher
**(state=done) & chosen : (i+1) mod 5;**

Lastly, we have:

- Our FAIRNESS constraints say we should be running and staying in the eating state as little as possible.
  **FAIRNESS running**
  **FAIRNESS !(state = eating)**
- Our safety states that we must have both forks to be eating.
  **SPEC AG((state = eating) -> (gotleft & gotright))**

- Our liveness states that if hungry, then we'll eventually get to be eating.
  **SPEC AG((state = hungry) -> AF (state = eating))**

## phil.extended.smv (extended solution)

The strategy here was to reuse the methodology for the basic solution but add an additional permission condition along with all new specs.

```
-- If a philosopher is currently eating (they got permission and started eating), then we can give a chance to eat to the next person!
        (state=eating) & chosen : (i+1) mod 5;
```

We implemented the permission condition by allowing currently eating philosophers who were chosen to pass on their permission, allowing for a philosopher currently eating to technically let another philosopher start eating at the same time (i.e. concurrency).