***Proyecto de Simulation:***

* *Franco, Federico*
* *Gastaldi, Diego*

***Semaphore Module:***

* Parameters:
  + time\_green: Real+ (semaphore time in green).
  + time\_red: Real+ (semaphore time in red).
  + time\_yellow: Real+ (semaphore time in yellow).
* Model:
  + S = R+  >< {red, yellow, green} => (clock, colors)
  + Y = {green, yellow, red}

(time\_green, green) if (color == red)

* + dint(clock, color) = (time\_yellow, yellow) if (color == green)

(time\_red, red) if (color == yellow)

* + ta(clock, color) = clock
  + Lambda(clock, color) = color
* Initial state: (some\_time, some\_color)

***Córner Module:***

* Parameters:
  + size\_cars: R+ (size of the cars).
  + speed\_cars: R+ (speed of the cars).
  + size\_corner: R+ (size of the corner).
  + out\_percent0: R+ (percentage of output on port 0).
  + out\_percent1: R+ (percentage of output on port 1) (not used)
* Model:
  + S = R+ >< [R+] >< R+ => amount\_cars >< [distance] >< sigma
  + X = {0, 1} => port
  + Y = {1} >< {0, 1}

error if (amount\_cars\*size\_cars >= size\_corner)

* + dext((amount\_cars, list, s), e, p) = (amount\_cars, update\_list (list, e), s-e) if (amount\_cars > 0 ^ size\_corner-head (list) < size\_cars)

(amount\_cars+1, update\_list (list, e) ● (size\_corner), s-e) if (amount\_cars >= 1 ^ (amount\_cars+1)\*size\_cars < size\_corner ^ size\_corner-head (list) >= size\_cars)

(1, size\_corner, speed\_cars/size\_corner) if (amount\_cars == 0)

Error if (amount\_cars <= 0)

* + dint((amount\_cars, list, s)) = (amount\_cars-1, update\_list(tail (list), s), speed\_cars/head (tail (list)))

if (amount\_cars >=2)

(0, empty, infinite) if (amount\_cars ==1)

* + ta((amount\_cars, list, s)) = s

(1, 0) if (random (100) <= out\_percent0)

* + Lambda((amount\_cars, list, s)) =

(1, 1) otherwise

* Notes:
  + In dext, the symbol “●” indicates the concatenation to final of the list.
  + As all cars have the same speed, the first to enter the corner will be the first out.
* Initial state: (0, empty, infinite)

***Street Module:***

* Parameters:
  + size\_cars: R+ (size of the cars).
  + speed\_cars: R+ (speed of the cars).
  + size\_street: R+ (size of the street).
* Model:
  + S = R+ >< [R+] >< R+ >< Bool=> (amount\_street, [distance], sigma, output)
  + X = {1} >< {0} U {green, red, yellow} >< {1}

The Zero port (corner input)

The One port (semaphore input)

* + Y = {1} >< {0}

(a\_street, update (dist, e, o), s-e, true) if ((xp == 1) ^ (xv == green) ^ (a\_street > 0))

(a\_street, dist, infinite, xv == green) if ((xp == 1) ^ (a\_street == 0))

* + dext ((a\_street, dist, s, o), e, (xv, xp)) = (a\_street, update (dist, e, o), infinite, false) if ((xp == 1) ^ (xv != green) ^ (a\_street > 0))

(a\_street+1, update (dist, e, o) ● size\_street, s-e, o) if ((xp == 0) ^ (a\_street >= 1) ^ (a\_street+1)\*size\_cars<=size\_street ^ (size\_street –last (update (dist, e, o)) >= size\_cars)

Error if ((xp == 0) ^ (a\_street+1)\*size\_cars>size\_street)

(a\_street, update (dist, e, o), s-e, o) if ((xp == 0) ^ ((a\_street+1)\*size\_cars<=size\_street) ^ (a\_street > 0) ^ (size\_street –last (update (dist, e, o)) < size\_cars)

(1, size\_street, size\_street/speed\_cars, o) if ((xp == 0) ^ (a\_street == 0))

(a\_street - 1, update (tail (dist), s, o), head (tail (dist)) /speed\_cars, o) if (o ^ (a\_street! >= 2))

(0, empty, infinite, o) if (o ^ (a\_street == 1))

* + dint((a\_street, dist, s, o)) =

error if (!o || a\_street <= 0)

* + ta((a\_street, dist, s, o)) = s
  + Lambda((a\_street, dist, s, o)) = 1
* Initial state: (0, empty, infinite, true)

Model of the project:

S2

S4

E1

E2

M1

SM2

S1

M2

S3

SM1

Streets:

* + - * SM1:
        + size\_cars: 2
        + speed\_cars: 30
        + size\_street: 100
      * SM2:
        + size\_cars: 2
        + speed\_cars: 30
        + size\_street: 80
      * M2:
        + size\_cars: 2
        + speed\_cars: 65
        + size\_street: 300
      * M1:
        + size\_cars: 2
        + speed\_cars: 30
        + size\_street:?

Semaphores

* + - * S1:
        + time\_green: 17
        + time\_red: 20
        + time\_yellow: 3
      * S2:
        + time\_green: 17
        + time\_red: 20
        + time\_yellow: 3
      * S3:
        + time\_green: 17
        + time\_red: 20
        + time\_yellow: 3
      * S4:
        + time\_green: 17
        + time\_red:20
        + time\_yellow: 3

Corners

* + - * E1:
        + size\_cars: 2
        + speed\_cars: 30
        + size\_corner: 10
        + out\_percent0: 40
        + out\_percent1: 60
      * E2:
        + size\_cars: 2
        + speed\_cars: 30
        + size\_corner: 10
        + out\_percent0: 50
        + out\_percent1: 50