CS552 Assignment 2

Yuepei Li

October 11, 2019

1 Question 2

1.1 Version 1

Algorithm 1 Cross Bridge, Version 1, PV

```
Input: int pass[2] = {0, 0}
Input: semaphore mutex[2] = {1, 1}, mutexBridge = 1
  procedure Cross(i)
      P(mutex[i])
      if pass[i] == 0 then
          pass[i] += 1
          P(mutexBridge)
      else
          pass[i] += 1
      end if
      V(mutex[i])

      Cross Bridge

      P(mutex[i])
      pass[i] -= 1
      if pass[i] == 0 then
          V(mutexBridge)
      end if
      V(mutex[i])
  end procedure
```

Algorithm 2 Cross Bridge, Version 1, Monitor

```
Input: int pass[2] = {0, 0}
Input: condition OKtoPass[2]

  procedure STARTCROSS(i)
      if pass[1-i] > 0 then
          OKtoPass[i].wait
      end if
      pass[i] += 1
      OKtoPass[i].signal
  end procedure

  procedure ENDCROSS(i)
      pass[i] -= 1
      if pass[i] == 0 then
          OKtoPass[1-i].signal
      end if
  end procedure
```

1.2 Version 2

Algorithm 3 Cross Bridge, Version 2, Simultaneous P/V

```
Input: semaphore n[2] = {N}                  ▷ N is the capability of the bridge
  procedure CROSS(i = 0)
      SP(n[i], 1, 1)
      SP(n[1-i], N, 0)                      ▷ block, if car from 1 runing on bridge
      CrossBridge
      SV(n[i], 1, 1)
  end procedure
  procedure CROSS(i = 1)
      SP(n[1-i], N, 0)                      ▷ block, if car waiting at 0.
      SP(n[i], 1, 1)
      CrossBridge
      SV(n[i], 1, 1)
  end procedure
```

Algorithm 4 Cross Bridge, Version 2, Monitor

```
Input: int pass[2] = {0, 0}
Input: condition OKtoPass[2]

  procedure STARTCROSS(i)
      if pass[1-i] > 0 then
          OKtoPass[i].wait
      end if
      if i == 1 and OKtoPass[1-i].queue then
          OKtoPass[i].wait
      end if
      pass[i] += 1
      OKtoPass[i].signal
  end procedure

  procedure ENDCROSS(i)
      pass[i] -= 1
      if pass[i] == 0 then
          OKtoPass[1-i].signal
      end if
  end procedure
```

1.3 Version 3

Algorithm 5 Cross Bridge, Version 3, Monitor

```
Input: int pass[2] = {0, 0}
Input: condition OKtoPass[2]

  procedure STARTCROSS(i)
      if OKtoPass[1-i].queue or pass[1-i] > 0 then
          OKtoPass[i].wait
      end if
      pass[i] += 1
      if !OKtoPass[1-i].queue then          ▷ if there is no car waitting opposite.
          OKtoPass[i].signal
      end if
  end procedure

  procedure ENDCROSS(i)
      pass[i] -= 1
      if pass[i] == 0 then
          OKtoPass[1-i].signal
      end if
  end procedure
```

Algorithm 6 Cross Bridge, Version 3, Serializer

Input: **queue** q[2]
Input: **crowd** crowd[2]

```
procedure CROSS(i)
    enqueue(q[i]) until ((empty(crowd[i]) and empty(crowd[1-i]))
                                                    ▷ no one on the bridge
      or (empty(q[1-i]) and !empty(crowd[i])))
                                                    ▷ flow car on same direction
    joincrowd(crowd[i])
     cross bridge
    end
end procedure
```