simulating_timepoints

Milla Juntunen

2022-06-29

*** SIMULATING EMERGENCY PROCESS AND PREHOSPITAL BLOOD TRANSFUSION TIMES ***

This code tries to calculate the total time spent in emergency process and the time between starting prehospital transfusion and arrival to hospital.

(For practical reasons the actual file that is called to simulate these times is .R-file. Contents of the files are same except the last chunk of the code (in .R-file that is also a function).)

```
library("readxl")
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

*** parse_file.function ***

This function reads and parses information from an excel file which contains the flowcharts.

Useful information (start node, end node, edge between (from start to end), lower bound of the time distribution, upper bound of the time distribution, probability of the edge) is stored in a list.

At the end the list is returned. For clarity, the excel sheet is also printed, and it can be found at the end of this file.

```
parse_file.function <- function(file_location, sheet_name) {

  # Reading the file and deleting empty rows
  df <- na.omit(read_excel(file_location, sheet_name))
  print(df)

  # Making empty list
  list_data <- list()

  # Counting rows
  rows = nrow(df)
  row <- 1

  # Reading the excel row by row
  for (row in 1:rows){
```

```
    # Reading data from a row
    row_data <- df[row,]

    # Defining the edge between nodes in flowchart and/or excel
    from <- as.character(select(row_data, from)) # start node
    to <- as.character(select(row_data, to)) # end node
    edge_name <- paste(from, to, sep = " - ") # edge between start node and end node

    # Limits for the times between events
    lower <- as.numeric(select(row_data, start)) # lower bound of the time distribution
    upper <- as.numeric(select(row_data, end)) # upper bound of the time distribution

     # Probability for the edge
    probability <- as.numeric(select(row_data, probability))

    # Listing parsed information
    list_data[row] <- list(c(from, to, edge_name, lower, upper, probability))
  }
  return(list_data)
}
```

*** check_vehicle.function ***

This function defines a vehicle used in emergency process. It is called in certain places of the flowchart - times to check are when 'to' includes "leaves" or "transport" or when 'from' includes "Packing". It is assumed that vehicle used in getting in the scene doesn't affect on the vehicle used in transportation and if infusion is started during transportation, the vehicle is assumed to stay the same as before.

If no vehicle exists (in other words it is NULL), vehicle to be used is included in 'to' (which is a string). By comparing the string with strings in vehicle list, the correct vehicle can be defined and vehicle is no longer NULL.

If the vehicle exists (for example 'to' includes "transport"), correct 'to' is defined again by comparing strings. Also storing correct bounds for time distribution at the same time.

Returning (corrected) vehicle, and (corrected) to and distribution limits.

```
check_vehicle.function <- function(vehicle, to, possible_rows, lower, upper){

  vehicles <- c("Ambulance", "Ambulance and doctor", "Helicopter", "Fire truck")

  # Going through list_data to make a correct match between vehicle and edge
  # If vehicle is NULL, should check the to
  if(is.null(vehicle)){

    for (i in 1:length(vehicles)){
      vehicle_to_test <- as.character(vehicles[i])

      # comparing data_to and vehicle if it is a match
      match_found <- grepl(vehicle_to_test, to, fixed = TRUE)

      # Must check that the correct vehicle is in the "to"
      if(match_found){
        comp_check <- grepl("Ambulance and doctor", to, fixed = TRUE)
        if(vehicle_to_test == "Ambulance" && comp_check == FALSE){
          vehicle <- vehicle_to_test
```

```
        } else {vehicle <- vehicle_to_test}
      }
    }
  } else {
    # If vehicle exists, should get a match from possible to (to_to_test) instead
    for (i in 1:length(possible_rows)){
      row <- possible_rows[i]
      to_to_test <- as.character(lapply(row,'[[',2))

      # Comparing to_to_test and vehicle if it is a match
      match_found <- grepl(vehicle, to_to_test, fixed = TRUE)

      # Must check that the correct vehicle is in the "to", returning correct limits at the same time..
      if(match_found){
        comp_check <- grepl("Ambulance and doctor", to_to_test, fixed = TRUE)

        if(vehicle == "Ambulance" && comp_check == FALSE){
          to <- to_to_test
          lower <- as.numeric(lapply(row,'[[',4))
          upper <- as.numeric(lapply(row,'[[',5))
        } else if (vehicle != "Ambulance") {
        to <- to_to_test
        lower <- as.numeric(lapply(row,'[[',4))
        upper <- as.numeric(lapply(row,'[[',5))
        }
      }
    }
  }
  # Returning matched vehicle and 'to'
  ret <- c(to, vehicle, lower, upper)
  return(ret)
}
```

*** draw_next_node_function ***

This function returns correct 'to' and time distribution limits based on the 'from' included in the function call. If multiple possible next nodes exist, then next node is drawn with probabilites of the edges taken in consideration. Also calling check_vehicle.function if needed.

```
draw_next_node.function <- function(from, list_data, vehicle){

  # Getting the next node, this is not the best way to do it
  # Must find the next "to", done below. Checking if multiple next nodes exist and drawing one of them

  match_count <- 0
  possible_to <- list()
  probs <- list()
  possible_rows <- list()

  for (j in 1:length(list_data)){
    row <- list_data[j]
    data_from <- as.character(lapply(row,'[[',1))

    if(data_from == from){
```

```
      # Match found, so increasing the match_count and saving the row in a list
      match_count <- match_count + 1
      to <- as.character(lapply(row,'[[',2))
      lower <- as.numeric(lapply(row,'[[',4))
      upper <- as.numeric(lapply(row,'[[',5))
      possible_rows <- append(possible_rows, row)
      prob <- as.numeric(lapply(row,'[[',6))
      probs <- append(probs, prob)
    }
  }
  # If multiple options from next node exist, drawing a row from the 'rows' to get the next 'to' and th
  if(match_count > 1){
    row <- sample(possible_rows, 1, replace = FALSE, probs)
    to <- as.character(lapply(row,'[[',2))
    lower <- as.numeric(lapply(row,'[[',4))
    upper <- as.numeric(lapply(row,'[[',5))
  }
  # Checking if check_vehicle is needed
  # Times to check are when 'to' includes "leaves" or "transport" or 'from' includes "Packing"
  check_1 <- grepl("Transport", to, fixed = TRUE)
  check_2 <- grepl("leav", to, fixed = TRUE)
  # If leaving the scene, new vehicle will be drawn, thus setting it NULL
  if(check_3 <- grepl("Packing", from, fixed = TRUE)){vehicle <- NULL}
  if(check_1 || check_2 || check_3){
    node <- check_vehicle.function(vehicle, to, possible_rows, lower, upper)
    to <- node[1]
    vehicle <- node[2]
    lower <- node[3]
    upper <- node[4]
  }
  return(c(from, to, vehicle, lower, upper))
}
```

*** time_calculator.function ***

This function calculates the time that has passed from the start of the emergency process and the time after starting the infusion.

In every round, the passed time between nodes is drawn between the time distribution limits and total_time_count (and infusion_time_count) is updated. The time distribution is assumed to follow the uniform distribution.

After each round check_next_node.function is called to get the next nodes and distribution limits.

Returns from, to, total_time_count, infusion_time_count and infusion_starts after the patient is transported to “Shock Room”.

```
time_calculator.function <- function(list_data){

  # Calculating the time between events (nodes)
  # stop_sign <- "Shock Room"

  # Drawing first node, assuming that the first from is 'Risk analysis' and vehicle is not known
  first_node <- draw_next_node.function(from <- "Risk analysis", list_data, vehicle <- NULL)
  from <- as.character(first_node[1])
  to <- as.character(first_node[2])
```

```
  vehicle <- as.character(first_node[3])
  lower <- as.numeric(first_node[4])
  upper <- as.numeric(first_node[5])

  # Going through the chart, drawing nodes when needed
  while(from != "Shock Room"){

    # Checking if it is time to stop and exit the loop
    #if(from == stop_sign) {break}
    # Updating the total_time_count
    random_time <- runif(1, lower, upper)
    total_time_count <- total_time_count + random_time

    # Checking if the infusion has started
    if(from == "Infusion starts"){infusion_starts <- TRUE}

    #If infusion has started, updating also infusion_time_count
    if(infusion_starts == TRUE) {
      infusion_time_count <- infusion_time_count + random_time
    }

    # Some printing to see if this works at all
    cat("\n")
    cat(from, "-", to, "\n")
    cat(lower, " is the lower limit for the time distribution\n")
    cat(upper, " is the upper limit for the time distribution\n")
    cat(random_time, " is the randomized time between limits\n")
    cat(total_time_count, " is the total time that has passed\n")
    cat(infusion_time_count, " is the time that has passed since the infusion started\n\n")

    # "to" is the next "from", getting forward in the chart
    from <- to
    if (from != "Shock Room"){
    next_node <- draw_next_node.function(from, list_data, vehicle)
    to <- as.character(next_node[2])
    vehicle <- as.character(next_node[3])
    lower <- as.numeric(next_node[4])
    upper <- as.numeric(next_node[5])
    }
 }
  ret <- c(from, to, total_time_count, infusion_time_count, infusion_starts)
}
```

*** “Main function” ***

This part of the code is a function in the .R-file and it is called make_timepoint.function.

The emergency process is assumed to follow the flowcharts that are stored in excel files. Chart A represents the situation in which the infusion starts on the scene and the chart B the situation in which the infusion starts during transportation to hospital. The probabilities of the charts used are assumed to be equal.

In this chunk simulating the time points is started by calling time_calculator.function. Now only one point is simulated, in R.-file this is done with for-loop so that as many simulated time points as wanted can be generated. For more information see the .R-file!

At the end results (total_time_count and infusion_time_count for each simulation round) are printed, in

.R-file infusion_time_count is returned in the file that calls the code.

```
### "MAIN" STARTS HERE
set.seed(2)
infusion_starts <- FALSE
total_time_count <- 0
infusion_time_count <- 0
file_location <- "C:\\Projektit\\whole_blood_research\\excel\\EmergencyProcess_EdgeTimes.xlsx"
sheet_name_a <- "test_times_a"
sheet_name_b <- "test_times_b"
from <- "Risk analysis"

# Reading the file, listing information from the sheets
list_data_a <- parse_file.function(file_location, sheet_name_a)
```

```
## # A tibble: 21 x 5
##    from                          to                       start   end probability
##    <chr>                         <chr>                    <dbl> <dbl>       <dbl>
##  1 Risk analysis                 Ambulance leaving            1   1.5         0.4
##  2 Risk analysis                 Ambulance and doctor le~     1   1.5         0.3
##  3 Risk analysis                 Helicopter leaving           2   5           0.25
##  4 Risk analysis                 Fire truck leaving           1   1.5         0.05
##  5 Ambulance leaving             On the scene                20  40           1
##  6 Ambulance and doctor leaving  On the scene                20  40           1
##  7 Helicopter leaving            On the scene                10  20           1
##  8 Fire truck leaving            On the scene                20  40           1
##  9 On the scene                  Meeting the patient          0  10           1
## 10 Meeting the patient           Infusion starts              1  20           1
## # ... with 11 more rows
```

```
list_data_b <- parse_file.function(file_location, sheet_name_b)
```

```
## # A tibble: 26 x 5
##    from                          to                       start   end probability
##    <chr>                         <chr>                    <dbl> <dbl>       <dbl>
##  1 Risk analysis                 Ambulance leaving            1   1.5         0.4
##  2 Risk analysis                 Ambulance and doctor le~     1   1.5         0.3
##  3 Risk analysis                 Helicopter leaving           2   5           0.25
##  4 Risk analysis                 Fire truck leaving           1   1.5         0.05
##  5 Ambulance leaving             On the scene                20  40           1
##  6 Ambulance and doctor leaving  On the scene                20  40           1
##  7 Helicopter leaving            On the scene                10  20           1
##  8 Fire truck leaving            On the scene                20  40           1
##  9 On the scene                  Meeting the patient          0  10           1
## 10 Meeting the patient           Packing                      1  20           1
## # ... with 16 more rows
```

```
# Drawing whether the infusion starts on the scene (chart_a) or during transportation (chart_b)
charts <- c("Chart A", "Chart B")
chart_to_follow <- charts[sample(1:length(charts),1)]
if(chart_to_follow == "Chart A"){
  list_data <- list_data_a
} else {list_data <- list_data_b}
cat(chart_to_follow, "is the chart to follow\n")
```

```
## Chart A is the chart to follow
```

```r
# Starting the time-calculator
current_stage <- time_calculator.function(list_data)
```

```
##
## Risk analysis - Helicopter leaving
## 2  is the lower limit for the time distribution
## 5  is the upper limit for the time distribution
## 3.719979  is the randomized time between limits
## 3.719979  is the total time that has passed
## 0  is the time that has passed since the infusion started
##
##
## Helicopter leaving - On the scene
## 10  is the lower limit for the time distribution
## 20  is the upper limit for the time distribution
## 11.68052  is the randomized time between limits
## 15.4005  is the total time that has passed
## 0  is the time that has passed since the infusion started
##
##
## On the scene - Meeting the patient
## 0  is the lower limit for the time distribution
## 10  is the upper limit for the time distribution
## 9.438393  is the randomized time between limits
## 24.83889  is the total time that has passed
## 0  is the time that has passed since the infusion started
##
##
## Meeting the patient - Infusion starts
## 1  is the lower limit for the time distribution
## 20  is the upper limit for the time distribution
## 18.92602  is the randomized time between limits
## 43.76492  is the total time that has passed
## 0  is the time that has passed since the infusion started
##
##
## Infusion starts - Packing
## 1  is the lower limit for the time distribution
## 5  is the upper limit for the time distribution
## 1.516636  is the randomized time between limits
## 45.28155  is the total time that has passed
## 1.516636  is the time that has passed since the infusion started
##
##
## Packing - Helicopter leaving the scene
## 1  is the lower limit for the time distribution
## 2  is the upper limit for the time distribution
## 1.468019  is the randomized time between limits
## 46.74957  is the total time that has passed
## 2.984654  is the time that has passed since the infusion started
##
##
## Helicopter leaving the scene - At hospital
## 10  is the lower limit for the time distribution
```

```
## 20  is the upper limit for the time distribution
## 15.49984  is the randomized time between limits
## 62.24941  is the total time that has passed
## 18.48449  is the time that has passed since the infusion started
##
##
## At hospital - Shock Room
## 1  is the lower limit for the time distribution
## 10  is the upper limit for the time distribution
## 5.974067  is the randomized time between limits
## 68.22347  is the total time that has passed
## 24.45856  is the time that has passed since the infusion started
```

```
# Information from the calculators:
total_time_count <- as.numeric(current_stage[3])
infusion_time_count <- as.numeric(current_stage[4])
cat("Patient transported to Shock Room!\n")
```

```
## Patient transported to Shock Room!
```

```
cat(total_time_count, "is the total time from the 'Risk Analysis'\n")
```

```
## 68.22347 is the total time from the 'Risk Analysis'
```

```
cat(infusion_time_count, "is the total time from the 'Infusion Starts'\n\n")
```

```
## 24.45856 is the total time from the 'Infusion Starts'
```