FE 8828 Assignment2.1

Fu Zhuxuan (G1800418B)

Make Choice for one simulation

The function works as below:

1. This function will generate a list of integers from 1 to N in random sequence.
2. Then it splits the list into evaluation group and selection group.
3. Finally, it would return the choice from this simulation group.

We assume the larger the number, the better is the candidate, hence N is the best amoung all numbers. If the best choice N is already included in the evaluation group, the function would return the last candidate.

An example output of 10 numbers with evalution group of size 5 is shown below.

```
make_choice <- function(N, split_number){
  input_list <- as.list(sample(1:N, N, replace = FALSE))
  evaluation <- input_list[1:split_number]
  selection <- input_list[-(1:split_number)]
  eval_best <- max(unlist(evaluation))
  result <- 0
  for (i in selection){
    if (i>=eval_best){
      result <- i
      break
    }
  }
  if (result == 0){
    result <- unlist(selection[length(selection)])
  }
  choice <- c(result, input_list)
  return(choice)
}
```

```
## [1] "For example, the choice from simulation group: 7,5,10,3,4,9,1,8,6,2 with eval
uation group of size 5 is 2"
```

Find the probability and average number

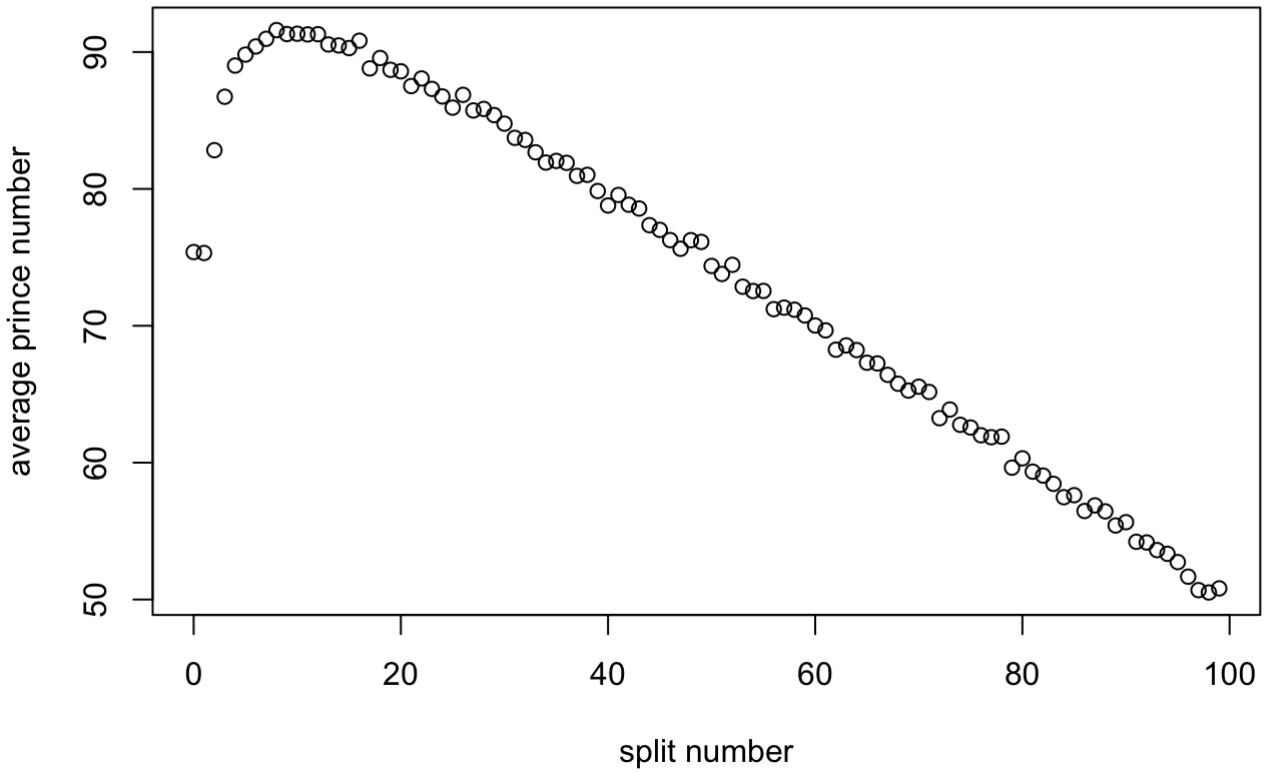
This function returns the probability of finding best prince and the average princes' number for N princes and k (split_number) princes as observation group under m (run_times) simulations.

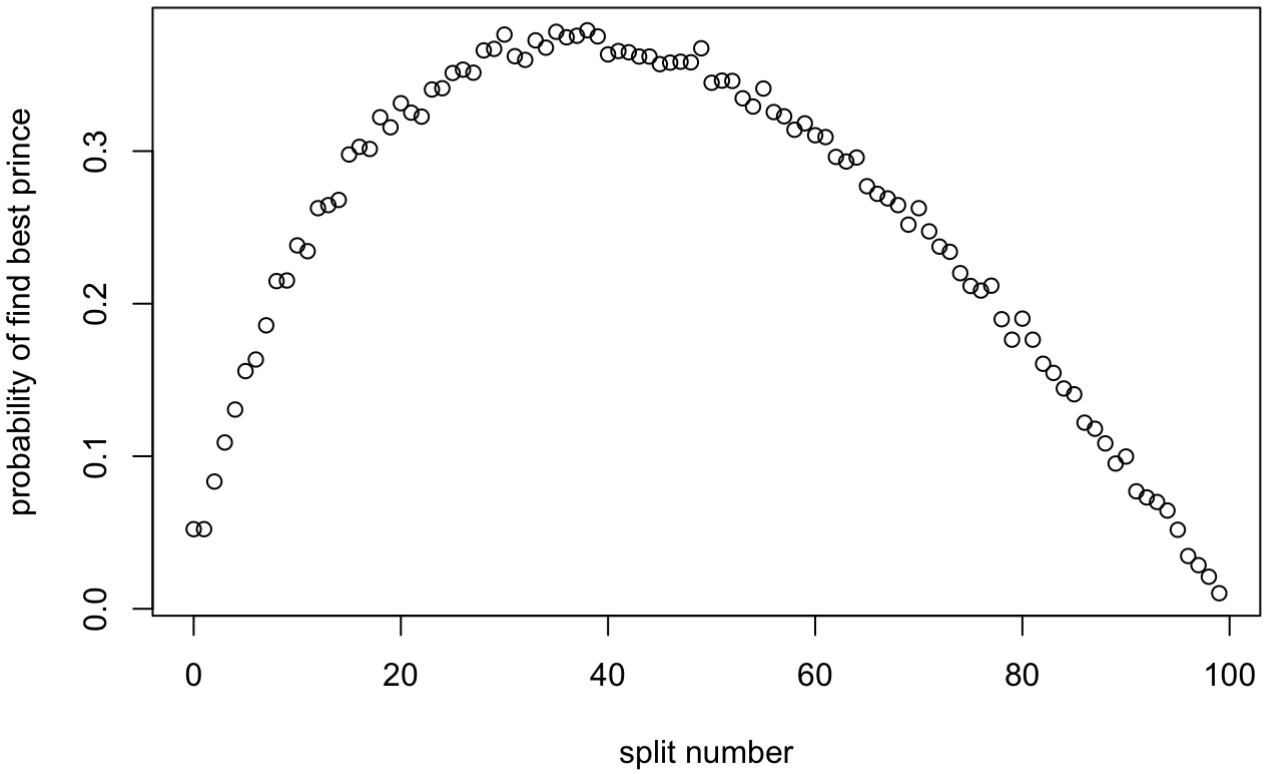
```
find_prob <- function(N, split_number, run_times){
  success <- 0
  sum <- 0
  for (i in 1:run_times){
    choice <- make_choice(N, split_number)
    choice_number = unlist(choice[1])
    if (choice_number == N){
      success <- success+1
    }
    sum <- sum + choice_number
  }
  prob = success/run_times
  prince_avg <- sum/run_times
  result <- c(prob, prince_avg)
  return(result)
}
```

Find the optimal split number

Below are the plots of average princes' number and probablity of finding best prince with obersation group size from 0 to N-1.

```
find_optimal <- function(N, run_times){
  optimal_split <- 0
  optimal_choice <- find_prob(N, optimal_split, run_times)
  optimal_prob <- optimal_choice[1]
  prince_ave_list <- rep(0, N)
  prob_list <- rep(0, N)
  prince_ave_list[1] <- optimal_choice[2]
  prob_list <- optimal_prob
  for (i in 1:N-1){
    curr_choice <- find_prob(N, i, run_times)
    curr_prob <- curr_choice[1]
    prince_ave_list[i+1] <- curr_choice[2]
    prob_list[i+1] <- curr_prob
    if (curr_prob>optimal_prob){
      optimal_prob <- curr_prob
      optimal_split <- i
    }
  }
  x <- seq(from = 0, to = N-1, by = 1)
  plot(x, prince_ave_list, xlab = 'split number', ylab = 'average prince number')
  plot(x, prob_list, xlab = 'split number', ylab = 'probability of find best prince')
  result <- c(optimal_split, optimal_prob)
  return(result)
}
```






```
## [1] "For 5000 simulations, the split number with highest probability of finding be
st princt is 38 with probability of 0.3792"
```