Lab9

Lingchen Lou

2018/11/13

Q1

```
library(microbenchmark)

odd_count<-function (x){
  odd_num <- 0
  for (i in 1:length(x)){
    if (x[i] %% 2 ==1)
      odd_num <- odd_num +1
  }
  return(odd_num)
}

odd_count2 <- function(x){
  odd_num <- 0
  odd_num <- sum(x %% 2 == 1)
  return(odd_num)
}

microbenchmark(odd_count(1:1000), odd_count2(1:1000))
```

```
## Unit: microseconds
##                expr     min       lq     mean  median       uq       max
##   odd_count(1:1000) 183.721 249.0650 341.1863 272.443 313.6310  4039.244
##  odd_count2(1:1000)   9.277  10.1005 189.8077  14.951  17.5065 17439.439
##  neval
##    100
##    100
```

We found that the Vectorized one is faster.

Q2

```
sort_vec<-function (x, ascending = TRUE){
  if(length(x) <2){
    return (x)
  }
  else{
    for (last in length(x):2){
      for (first in 1:(last -1)){
        if (x[first]> x[first+1]){
          temp <-x[first]
          x[first] = x[first +1]
          x[first +1] = temp
        }
      }
```

```
    }
  }
  if (ascending == FALSE){
    for (last in length(x):2){
      for (first in 1:(last-1)){
        if(x[first]<x[first +1]){
          temp<-x[first]
          x[first] = x[first +1]
          x[first +1] = temp
        }
      }
    }
  }
  return (x)
}

sort_vec(c(3, 1, 2), ascending = TRUE)
```

```
## [1] 1 2 3
```

```
sort_vec(c(3, 1, 2), ascending = FALSE)
```

```
## [1] 3 2 1
```

Q3

```
#1000

N = 1000
#dynamically allocated memory
data_series = 0
system.time({for (i in 2:N){
  data_series[i] = data_series[i-1] + sample(c(-1, 1), 1)
  }
})
```

```
##    user  system elapsed
##   0.009   0.000   0.009
```

```
##preallocated memory


system.time({
  data_series2 = length(N)
  for (i in 2:N) {
    data_series2[i] = data_series2[i-1] + sample(c(-1, 1), 1)
  }
})
```

```
##    user  system elapsed
##   0.007   0.000   0.007
```

There is not obvious difference between two methods.

10000

```r
N = 10000
#dynamically allocated memory
data_series = 0
system.time({for (i in 2:N){
  data_series[i] = data_series[i-1] + sample(c(-1, 1), 1)
}
})
```

```
##    user  system elapsed
##   0.054   0.008   0.061
```

```r
##preallocated memory

system.time({
  data_series2 = length(N)
  for (i in 2:N) {
    data_series2[i] = data_series2[i-1] + sample(c(-1, 1), 1)
  }
})
```

```
##    user  system elapsed
##   0.045   0.006   0.053
```

When N = 10000, we find that the second one is faster than the first one but there only a tiny difference, since the second one is 0.003 seconds faster than the first one.

1000000

```r
N = 1000000
#dynamically allocated memory
data_series = 0
system.time({
  for (i in 2:N){
  data_series[i] = data_series[i-1] + sample(c(-1, 1), 1)
}
})
```

```
##    user  system elapsed
##   3.971   0.292   4.282
```

```r
##preallocated memory
system.time({
  data_series2 = vector("numeric",N)
  for (i in 2:N) {
    data_series2[i] = data_series2[i-1] + sample(c(-1, 1), 1)
  }
})
```

```
##    user  system elapsed
##   3.750   0.165   3.965
```

When N = 1000000, we find that there is obvious difference between two that the second one is much faster than the first one.