Very few benchmarks comparisons

Author: Lukasz Zaniewicz

1 Task 1 - the mode function

At first we assume that microbenchmark package is loaded.

```
require(microbenchmark)
```

I was surprised that there is no ready-made function calculating mode i R, so in order to perform those benchmarks I use first google result solution

```
mode_r <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}
```

```
x <- sample(c(1:50), 100, replace=TRUE)
microbenchmark(mode(x), mode_r(x))

## Unit: nanoseconds
##       expr   min    lq     mean median    uq    max neval cld
##    mode(x)   810  1216  1698.08   1621  2026  10941   100  a
##  mode_r(x) 23502 25123 35294.40  25934 40319 343218   100   b
```

```
x <- sample(c(1:5), 100, replace=TRUE)
microbenchmark(mode(x), mode_r(x))

## Unit: nanoseconds
##       expr   min    lq     mean median    uq    max neval cld
##    mode(x)   810  1216  2650.31   1216  1621  95226   100  a
##  mode_r(x) 22692 23908 26096.09  24313 24921 104141   100   b
```

```
x <- sample(c(1:50), 1000, replace=TRUE)
microbenchmark(mode(x), mode_r(x))
```

```
## Unit: nanoseconds
##       expr   min    lq     mean  median      uq    max neval cld
##    mode(x)   810  1216  1661.65  1418.5  2026.0   4458   100  a
##  mode_r(x) 56325 61999 71310.05 64429.0 72534.5 164923   100   b
```

And we see that my function performs worse in case of large sample :(

2 Task 2 - the simplify2array function

Now we move on to benchmarks of list-to-array simplifying function. At first we create simple auxillary function thta generates random integer vector.

```
aux_fun <- function(n) {
  return(sample.int(50, n, TRUE))
}
x <- lapply(rep(5,5), aux_fun)
microbenchmark(hw6package::simplify2array(x), simplify2array(x))

## Unit: microseconds
##                           expr    min     lq     mean  median      uq
##  hw6package::simplify2array(x) 27.554 29.176 182.6592 33.8360 36.6725
##              simplify2array(x) 40.522 42.548  50.4738 48.8285 52.0705
##        max neval cld
##  14860.067   100   a
##    115.082   100   a
```

```
x <- lapply(rep(100,100), aux_fun)
microbenchmark(hw6package::simplify2array(x), simplify2array(x))

## Unit: microseconds
##                           expr     min      lq     mean   median       uq
##  hw6package::simplify2array(x) 141.015 148.512 307.1253 154.9955 224.0845
##              simplify2array(x) 175.863 183.968 253.1627 197.1375 292.7685
##       max neval cld
##  2975.093   100   a
##  2116.846   100   a
```

```
x <- lapply(rep(1000,1000), aux_fun)
microbenchmark(hw6package::simplify2array(x), simplify2array(x))

## Unit: milliseconds
##                            expr       min       lq     mean   median
##  hw6package::simplify2array(x) 26.286337 28.79847 47.18296 32.23997
##              simplify2array(x)  7.999362 11.65056 15.47916 12.93570
##        uq      max neval cld
##  79.61575 97.28619   100   b
##  14.43803 79.72253   100  a
```

We see that on smaller sample our function performs slowly, but it

3 Task 3 - the ass function

And now the last function. We compare our function to my own, very-fast-written survey assignment R function called ass2. It uses combn function from combinat package.

```
require(combinat)
ass2 <- function(n){
  tt <- t(combn(2*n,n))
  N <- nrow(tt)
  res <- matrix(0,ncol=2*n, nrow=N)

  for (i in 1:N){
    res[i,tt[i,]] <- 1
  }
  return(res)
}
```

```
microbenchmark(hw6package::ass(3), ass2(3))

## Unit: microseconds
##                expr     min       lq      mean  median       uq      max
##  hw6package::ass(3)  25.529  28.7710  45.62351  42.143  52.8810  316.068
##             ass2(3) 308.370 339.5715 412.80151 402.785 464.7825 1032.489
##  neval cld
##    100  a
##    100   b
```

```
microbenchmark(hw6package::ass(5), ass2(5))

## Unit: microseconds
##                expr      min       lq       mean    median       uq
##  hw6package::ass(5)   35.254   41.332   60.27601   52.4755   81.043
##             ass2(5) 2330.395 2576.158 2830.14763 2655.7835 2796.393
##       max neval cld
##   130.480   100  a
##  5975.715   100   b
```

```
microbenchmark(hw6package::ass(6), ass2(6))

## Unit: microseconds
##                expr      min        lq      mean   median        uq
##  hw6package::ass(6)   73.750   84.4875  124.8633  132.911   146.891
##             ass2(6) 8414.708 8997.0030 9839.3647 9265.052 10202.113
##        max neval cld
##    263.795   100  a
##  18383.011   100   b
```

And we see that performance gap is really huge.