# ER01-quicksort-Report

## Introduction

The goal of this report is to compare the quicksort algorithm run on different machines, with different number of threads.

## Method

We used the algorithm from the link: http://mescal.imag.fr/membres/arnaud.legrand/teaching/2013/M2R_EP_archive_quicksort.tgz

We used 4 types of machines: 1. CPU with 2 cores and 4GB RAM 1. CPU with 4 cores and 4GB RAM 1. CPU with 4 cores and 8GB RAM 1. CPU with 8 cores and 16GB RAM

## Results

Loading the measurements form the csv files created by the algorithm.

```
## Warning: package 'ggplot2' was built under R version 3.2.1
```

Compute the average execution time for each size and type.

```
library(plyr)
```

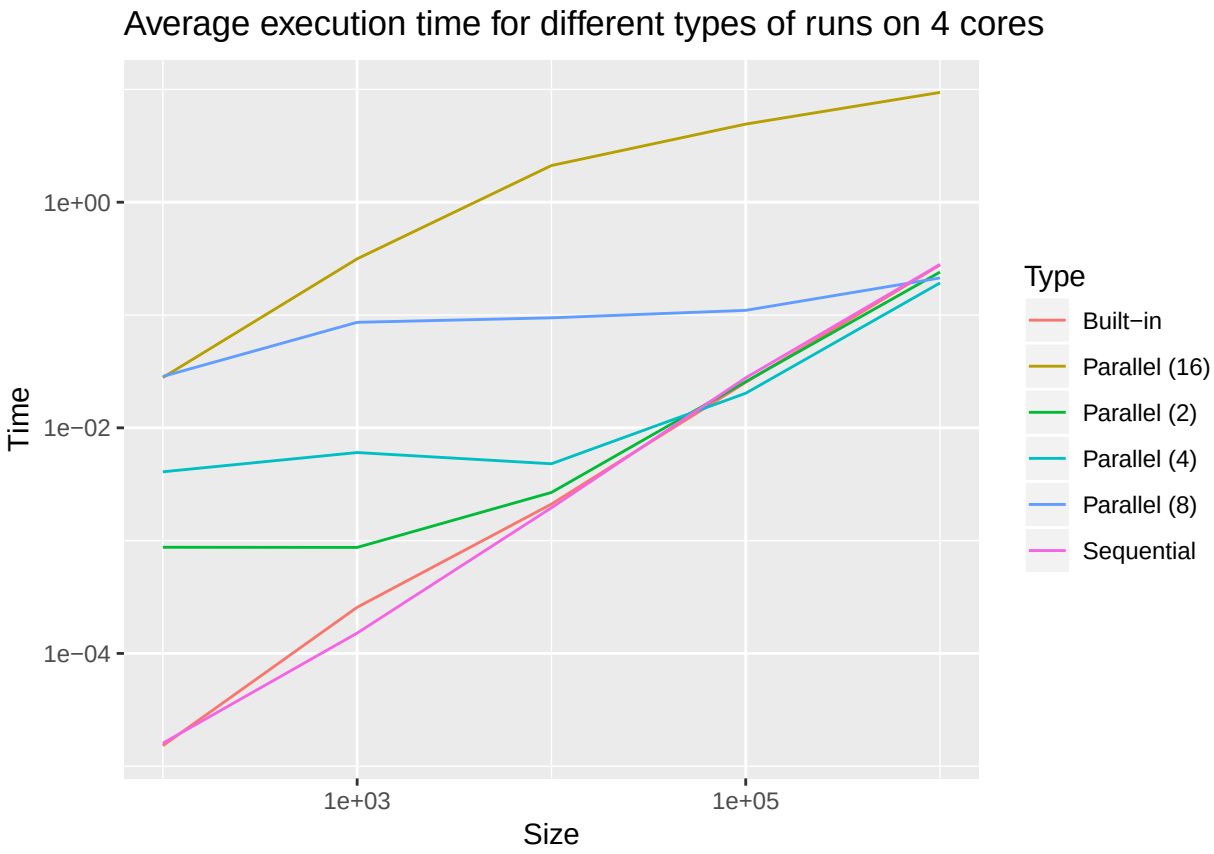
```
## Warning: package 'plyr' was built under R version 3.2.1
```

```
df3_avg <- ddply(df3,c("Size","Type"), summarise, Time = mean(Time))
df4_avg <- ddply(df4,c("Size","Type","Cores"), summarise, Time = mean(Time))
df5_avg <- ddply(df5,c("Size","Type","Cores"), summarise, Time = mean(Time))
df6_avg <- ddply(df6,c("Size","Type","Cores"), summarise, Time = mean(Time))
df7_avg <- ddply(df7,c("Size","Type","Cores"), summarise, Time = mean(Time))
```

### Average execution time

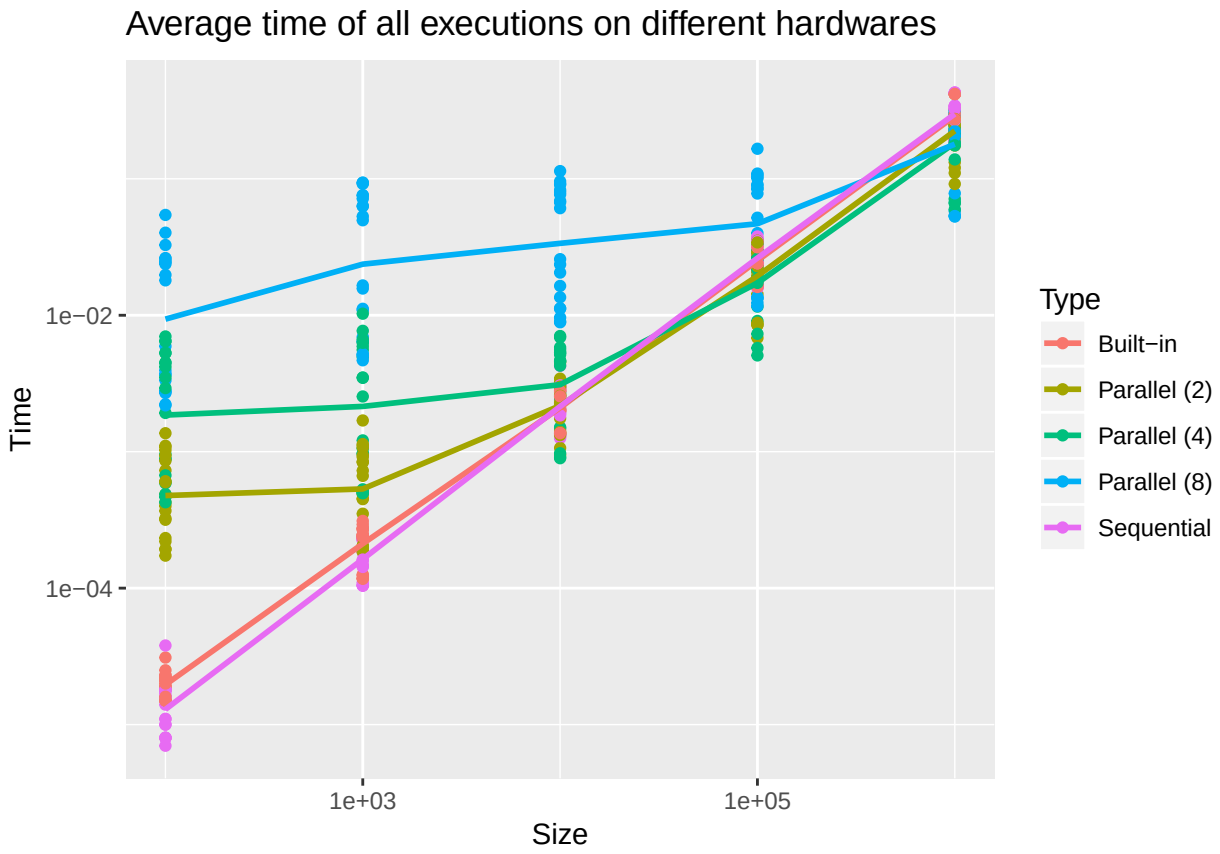
Plotting the average execution time for different types of runs, depending on the number of threads, using 4 cores.

```
pow2 <- c(1, 2, 4, 8, 16)
ggplot(data=df7_avg[df7_avg$Cores %in% pow2,],aes(x=Size,y=Time,color=Type))+geom_line() +
 scale_x_log10() + scale_y_log10() +
  ggtitle("Average execution time for different types of runs on 4 cores")
```



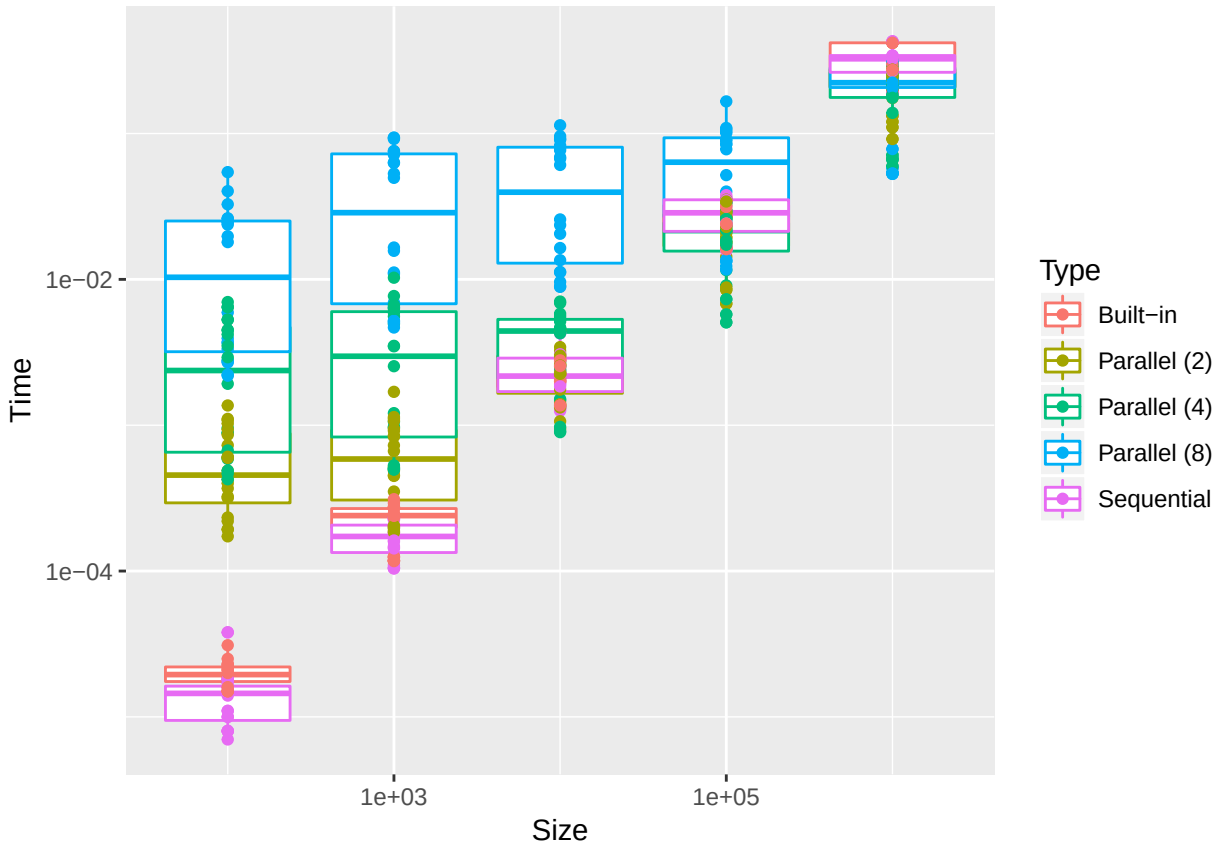

Plotting the average of each execution time on each hardware, depending on the type of the run (number of threads).

```
pow2 <- c(1, 2, 4, 8, 16)
all <- rbind(df4[df4$Cores < 11,], df5, df6, df7[df7$Cores < 11,])
ggplot(data=all[all$Cores %in% pow2,], aes(x=Size,y=Time,color=Type)) + geom_point() +
  scale_x_log10() + scale_y_log10() +  stat_summary(fun.y = mean, geom="smooth") +
  ggtitle("Average time of all executions on different hardwares")
```



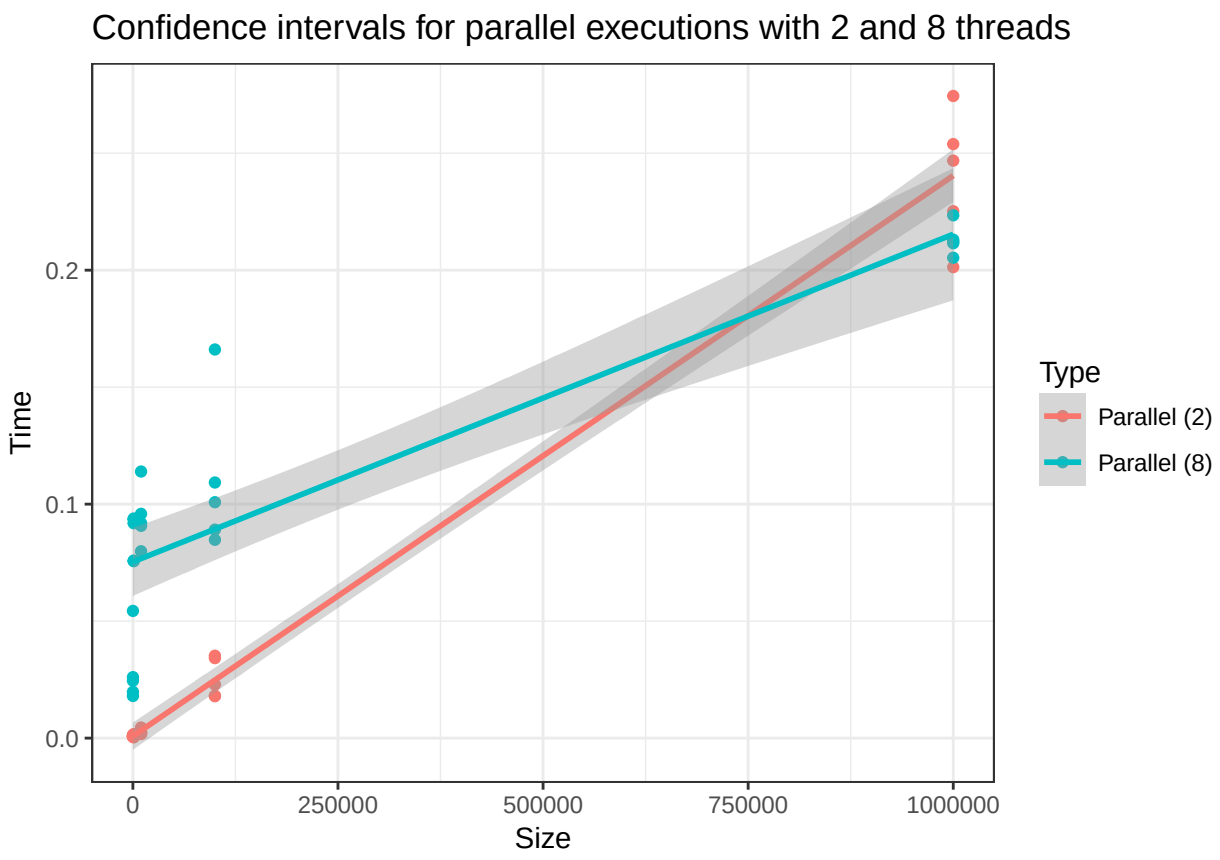

Plotting the average of each execution time on each hardware, depending on the size of the data being sorted.

```
pow2 <- c(1, 2, 4, 8, 16)
all <- rbind(df4[df4$Cores < 11,], df5, df6, df7[df7$Cores < 11,])
ggplot(data=all[all$Cores %in% pow2,], aes(x=Size,y=Time,color=Type,group=factor(Size):Type)) + geom_bo
  scale_x_log10() + scale_y_log10()
```




Plotting two types of parallel run, with confidence intervals.

```
cor = c(2, 8)
ggplot(data=df7[df7$Cores %in% cor,], aes(x=Size,y=Time,color=Type)) + geom_point() + theme_bw() + geom_
```



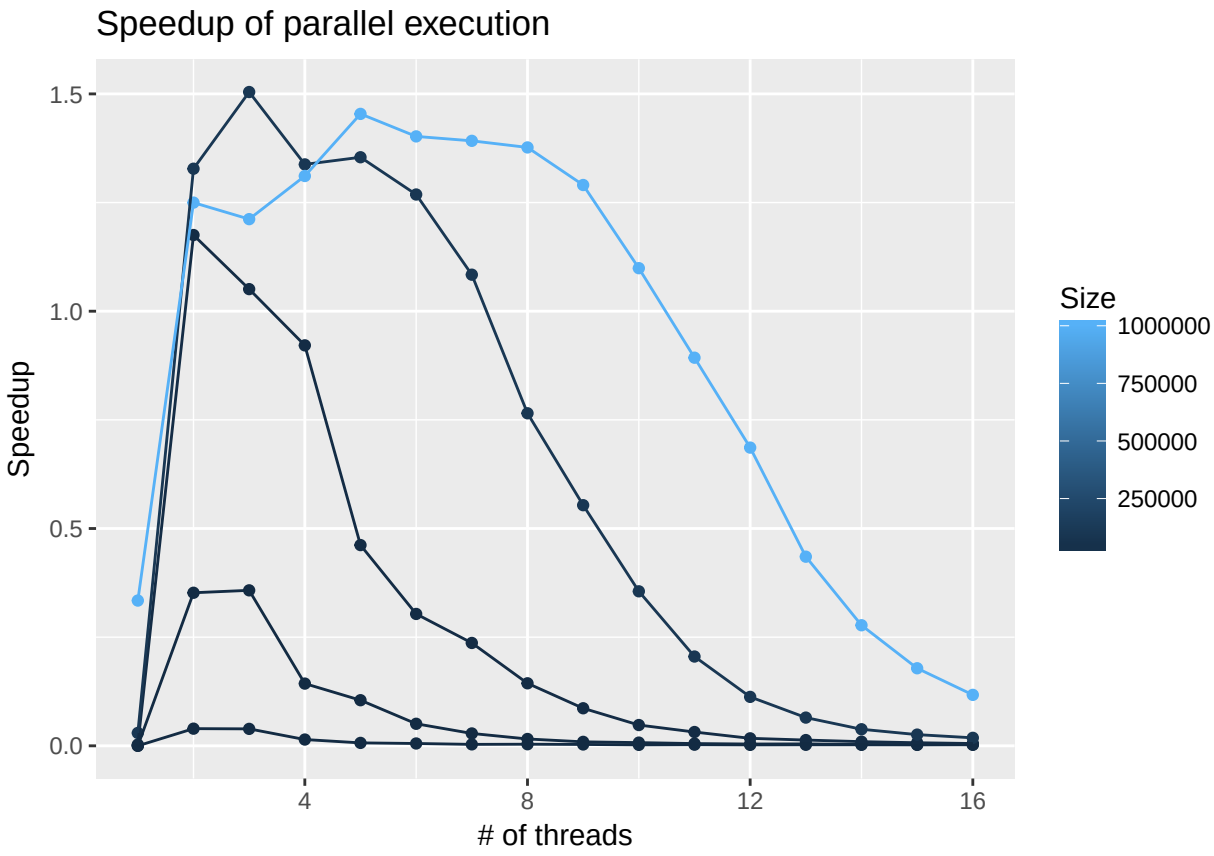

**Speedup**

Speedup for a parallel execution: speedup = sequential time / parallel time

Plotting the speedup for each type of hardware, using the average time for each size.

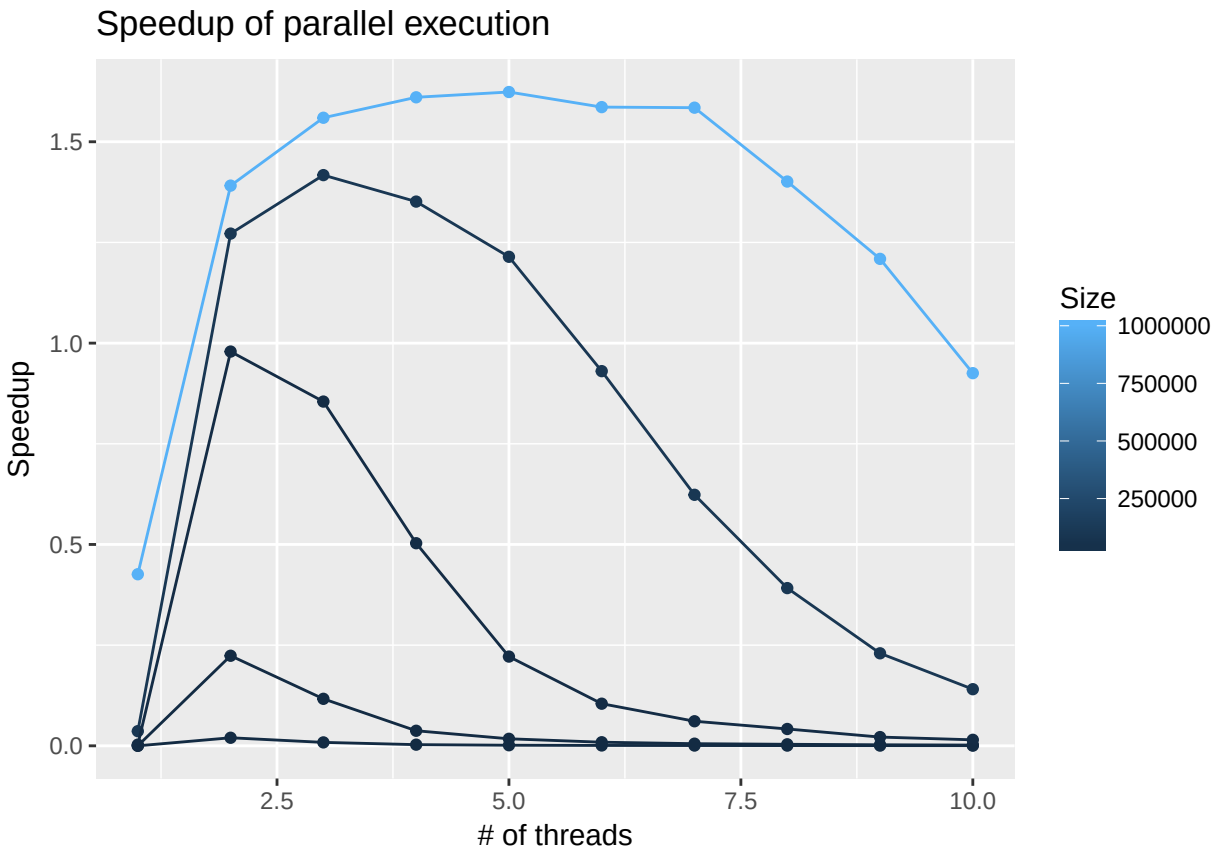
2 cores, 4GB RAM number of threads: 2-16

```
NR_THREADS <- 16
seq <- df4_avg[df4_avg$Type == " Sequential",]
df2_speedup <- rbind(seq)

for (t in c(2:NR_THREADS)){
  part <- df4_avg[df4_avg$Cores == t,]
  part["Time"] <- seq["Time"] / part["Time"]
  df2_speedup <- rbind(df2_speedup, part)
}

ggplot(data=df2_speedup, aes(x=Cores,y=Time,color=Size, group = Size))+ geom_point() + geom_line()+
  xlab("# of threads") + ylab("Speedup") + ggtitle("Speedup of parallel execution")
```




4 cores, 4GB RAM number of threads: 2-10

```
NR_THREADS <- 10
seq <- df5_avg[df5_avg$Type == " Sequential",]
df5_speedup <- rbind(seq)

for (t in c(2:NR_THREADS)){
  part <- df5_avg[df5_avg$Cores == t,]
  part["Time"] <- seq["Time"] / part["Time"]
  df5_speedup <- rbind(df5_speedup, part)
}

ggplot(data=df5_speedup, aes(x=Cores,y=Time,color=Size, group = Size))+ geom_point() + geom_line()+
  xlab("# of threads") + ylab("Speedup") + ggtitle("Speedup of parallel execution")
```

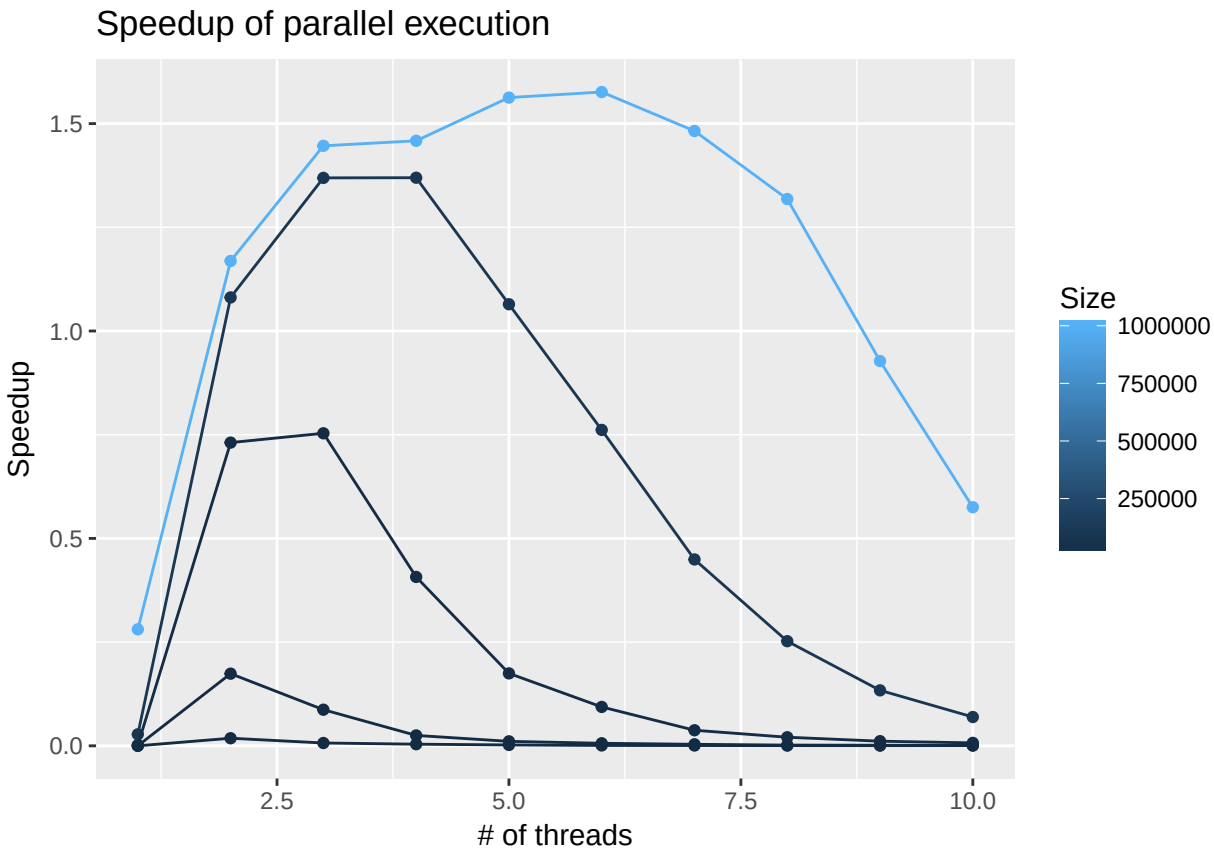



4 cores, 16GB RAM number of threads: 2-10

```
NR_THREADS <- 10
seq <- df6_avg[df6_avg$Type == " Sequential",]
df6_speedup <- rbind(seq)

for (t in c(2:NR_THREADS)){
  part <- df6_avg[df6_avg$Cores == t,]
  part["Time"] <- seq["Time"] / part["Time"]
  df6_speedup <- rbind(df6_speedup, part)
}

ggplot(data=df6_speedup, aes(x=Cores,y=Time,color=Size, group = Size))+ geom_point() + geom_line()+
  xlab("# of threads") + ylab("Speedup") + ggtitle("Speedup of parallel execution")
```


4 cores, 8GB RAM number of threads: 2-16

```
NR_THREADS <- 10
seq <- df7_avg[df7_avg$Type == " Sequential",]
df7_speedup <- rbind(seq)

for (t in c(2:NR_THREADS)){
  part <- df7_avg[df7_avg$Cores == t,]
  part["Time"] <- seq["Time"] / part["Time"]
  df7_speedup <- rbind(df7_speedup, part)
}

ggplot(data=df7_speedup, aes(x=Cores,y=Time,color=Size, group = Size))+ geom_point() + geom_line() +
  xlab("# of threads") + ylab("Speedup") + ggtitle("Speedup of parallel execution")
```



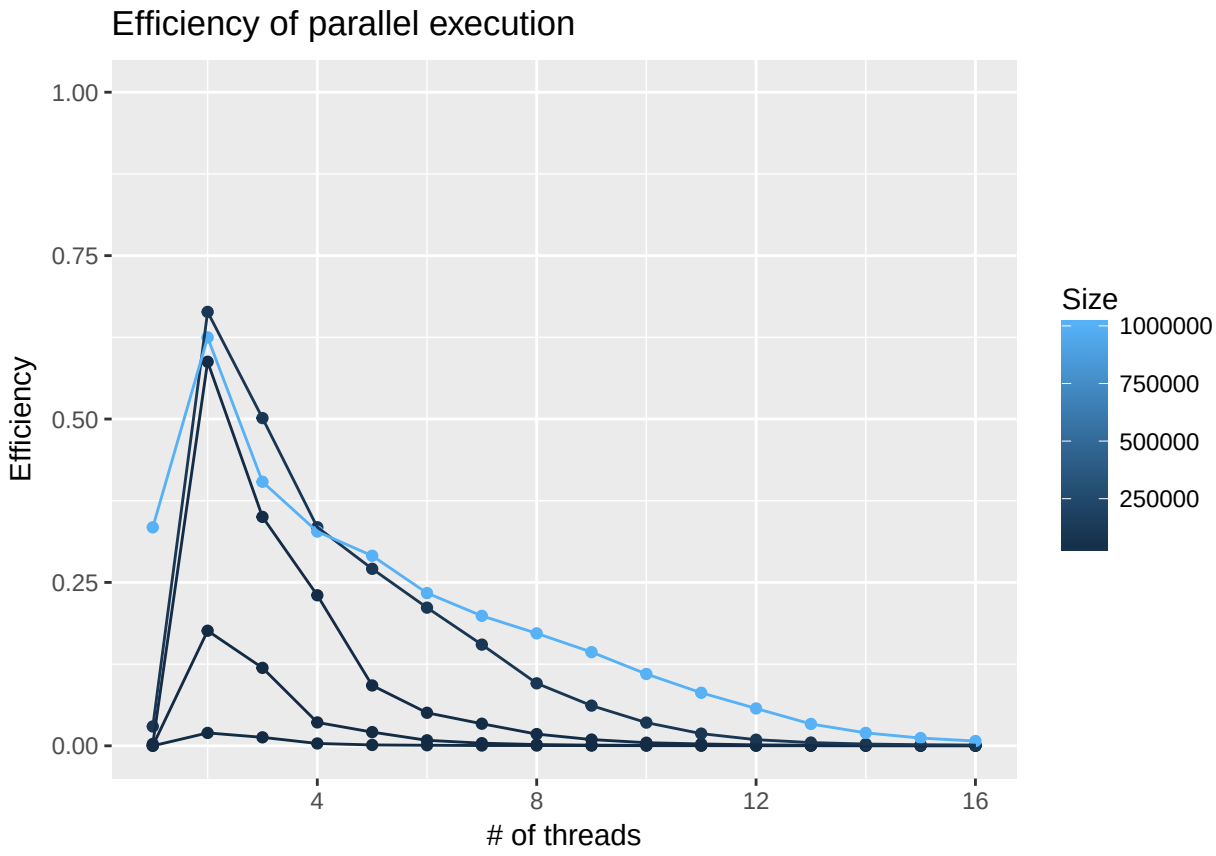

**Efficiency**

Efficiency for a parallel execution: efficiency = sequential time / (parallel time * # of threads)

Plotting the efficiency for each hardware, using the average execution time for each size.

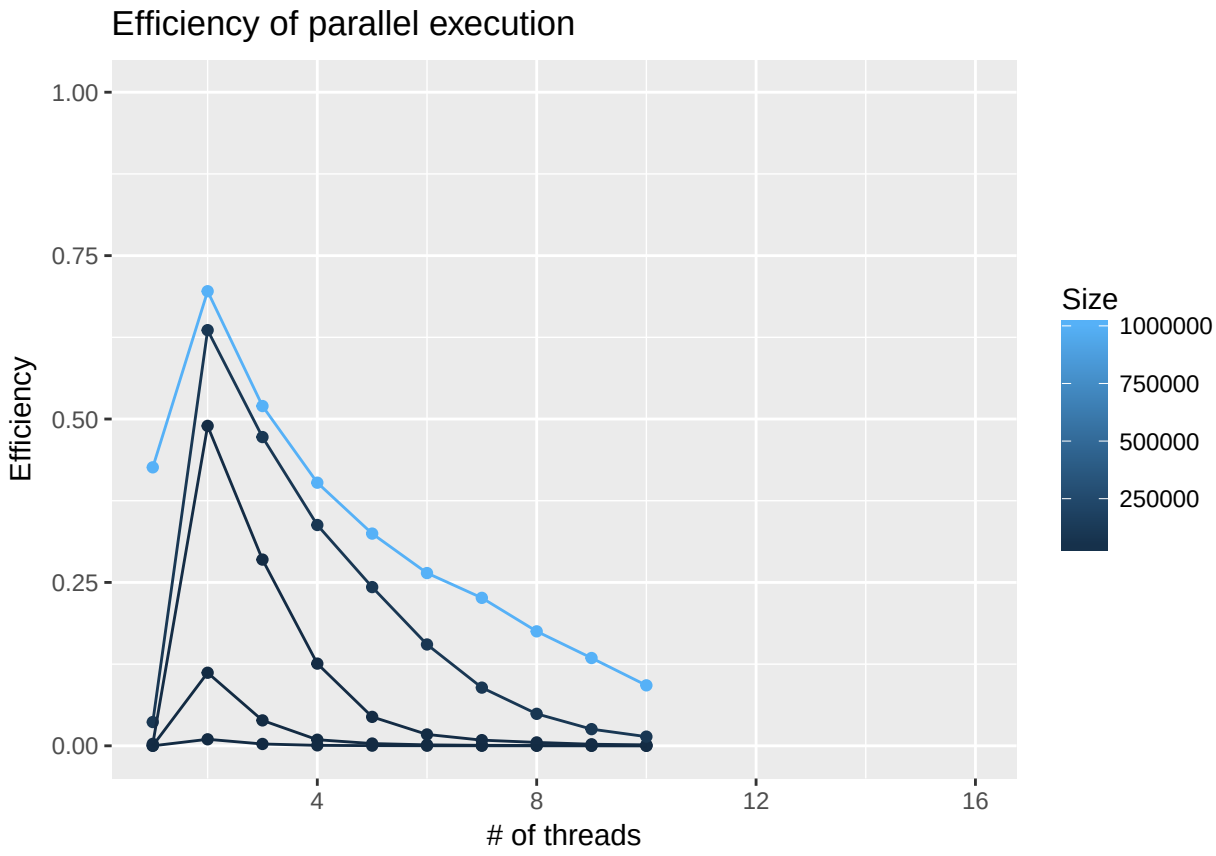
2 cores, 4GB RAM number of threads: 2-16

```
NR_THREADS <- 16
seq <- df4_avg[df4_avg$Type == " Sequential",]
df4_efficiency <- rbind(seq)

for (t in c(2:NR_THREADS)){
  part <- df4_avg[df4_avg$Cores == t,]
  part["Time"] <- seq["Time"] / part["Time"]
  part["Time"] <- part["Time"] / t
  df4_efficiency <- rbind(df4_efficiency, part)
}

ggplot(data=df4_efficiency, aes(x=Cores,y=Time,color=Size, group = Size))+ geom_point() + geom_line() +
  xlab("# of threads") + ylab("Efficiency") + ggtitle("Efficiency of parallel execution")
```



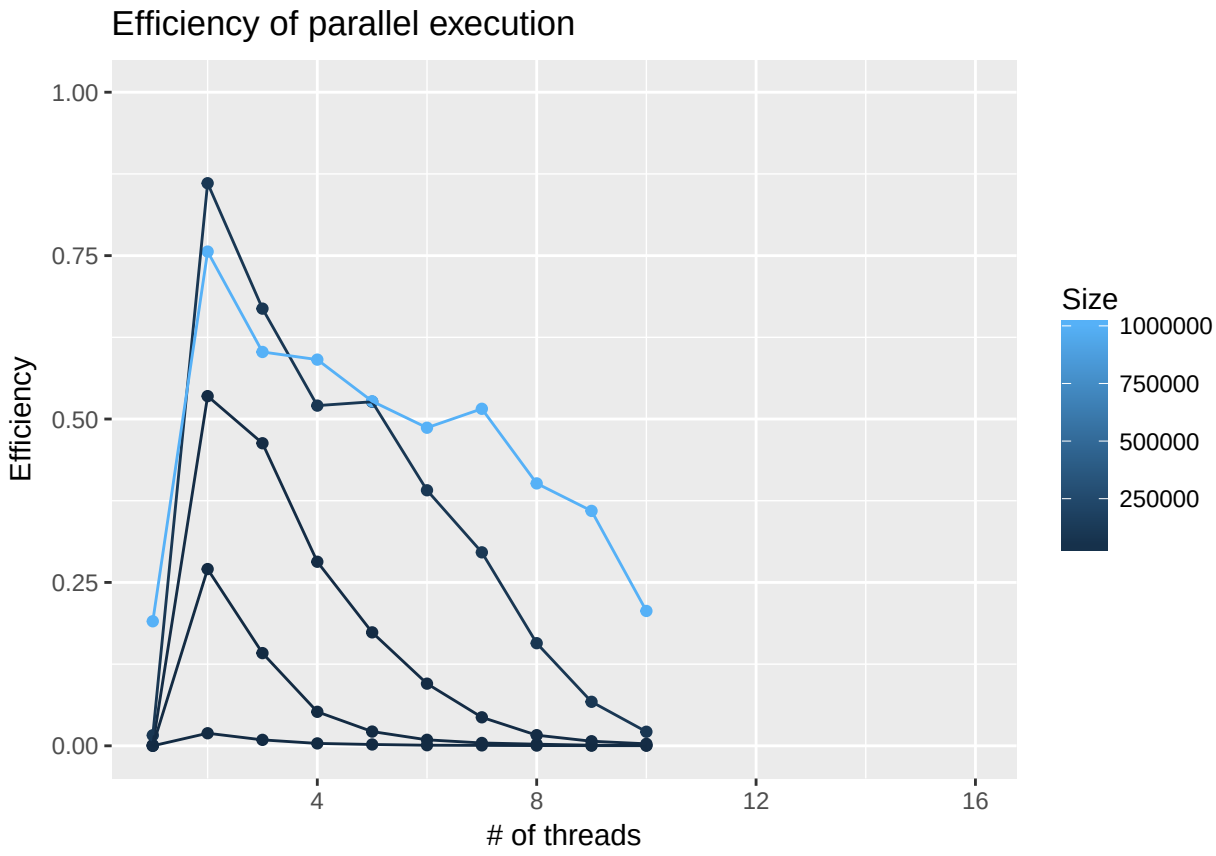

4 cores, 4GB RAM number of threads: 2-10

```
NR_THREADS <- 10
seq <- df5_avg[df5_avg$Type == " Sequential",]
df5_efficiency <- rbind(seq)

for (t in c(2:NR_THREADS)){
  part <- df5_avg[df5_avg$Cores == t,]
  part["Time"] <- seq["Time"] / part["Time"]
  part["Time"] <- part["Time"] / t
  df5_efficiency <- rbind(df5_efficiency, part)
}

ggplot(data=df5_efficiency, aes(c(1:16), x=Cores,y=Time,color=Size, group = Size))+ geom_point() + geom_
  scale_y_continuous(limits=c(0,1)) +
  xlab("# of threads") + ylab("Efficiency") + ggtitle("Efficiency of parallel execution")
```



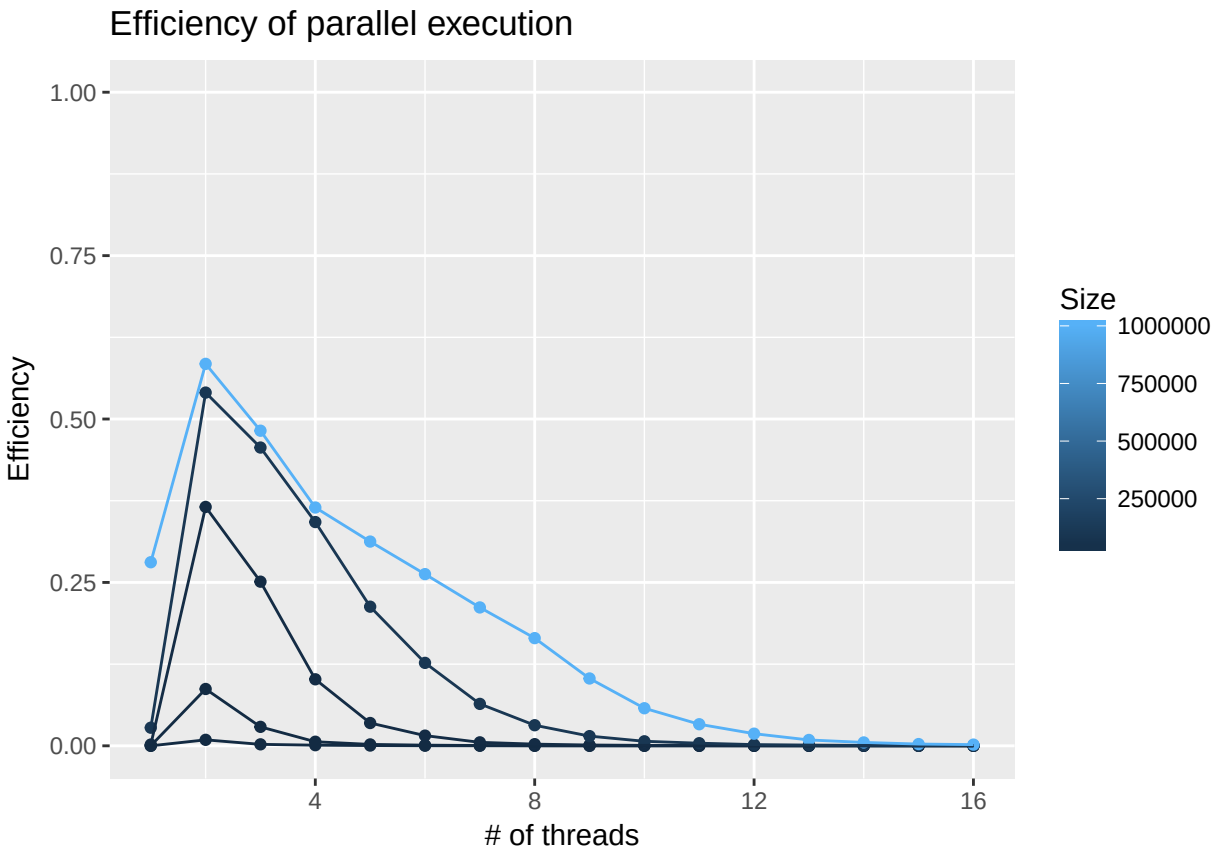

4 cores, 16GB RAM number of threads: 2-10

```
NR_THREADS <- 10
seq <- df6_avg[df6_avg$Type == " Sequential",]
df6_efficiency <- rbind(seq)

for (t in c(2:NR_THREADS)){
  part <- df6_avg[df6_avg$Cores == t,]
  part["Time"] <- seq["Time"] / part["Time"]
  part["Time"] <- part["Time"] / t
  df6_efficiency <- rbind(df6_efficiency, part)
}

ggplot(data=df6_efficiency, aes(c(1:16), x=Cores,y=Time,color=Size, group = Size))+ geom_point() + geom_
  scale_y_continuous(limits=c(0,1)) +
  xlab("# of threads") + ylab("Efficiency") + ggtitle("Efficiency of parallel execution")
```




4 cores, 8GB RAM number of threads: 2-16

```
NR_THREADS <- 16
seq <- df7_avg[df7_avg$Type == " Sequential",]
df7_efficiency <- rbind(seq)

for (t in c(2:NR_THREADS)){
  part <- df7_avg[df7_avg$Cores == t,]
  part["Time"] <- seq["Time"] / part["Time"]
  part["Time"] <- part["Time"] / t
  df7_efficiency <- rbind(df7_efficiency, part)
}

ggplot(data=df7_efficiency, aes(c(1:16), x=Cores,y=Time,color=Size, group = Size))+ geom_point() + geom_
  scale_y_continuous(limits=c(0,1)) +
  xlab("# of threads") + ylab("Efficiency") + ggtitle("Efficiency of parallel execution")
```

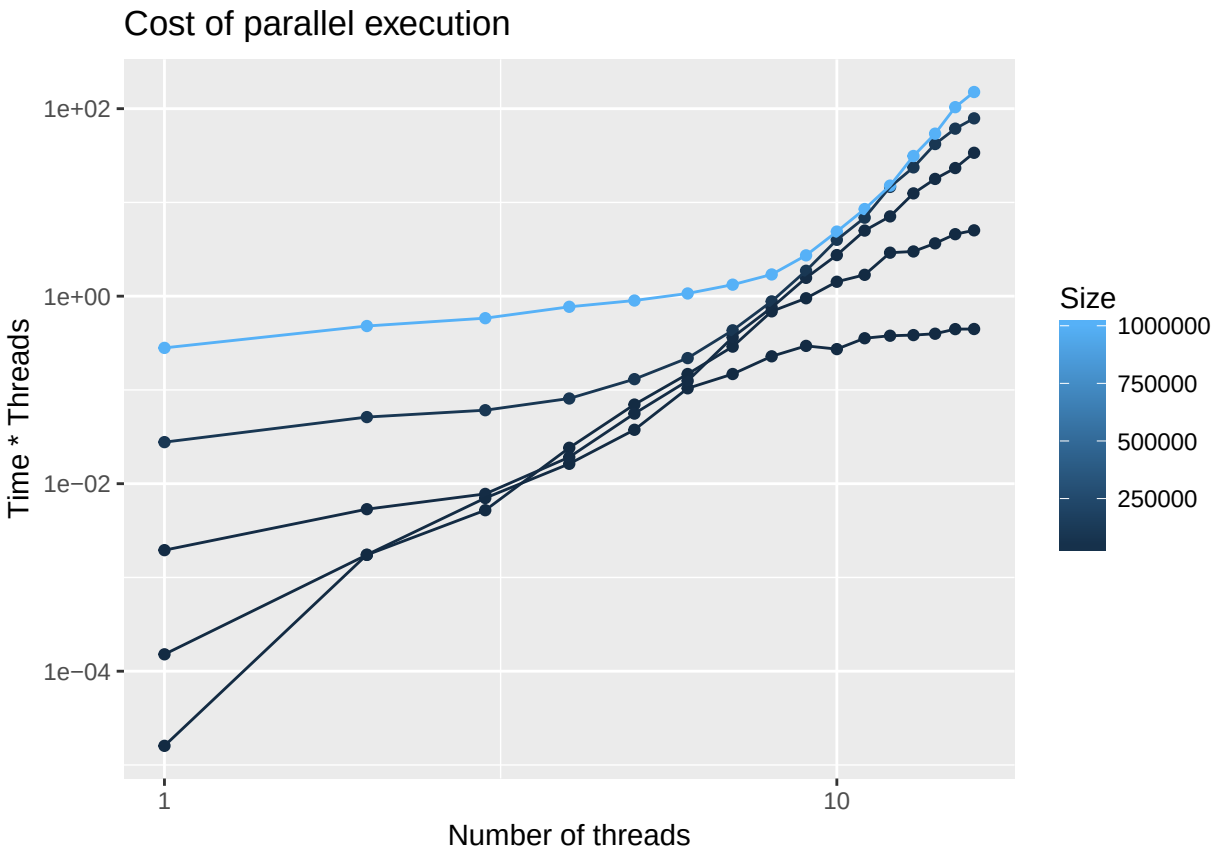



**Cost**

Cost = execution time * number of threads

4 cores, 8GB RAM number of threads: 2-16

```
NR_THREADS <- 16
seq <- df7_avg[df7_avg$Type == " Sequential",]
df7_efficiency <- rbind(seq)

for (t in c(2:NR_THREADS)){
  part <- df7_avg[df7_avg$Cores == t,]
  df7_efficiency <- rbind(df7_efficiency, part)
}

ggplot(data=df7_efficiency, aes(c(1:16), x=Cores,y=Time*Cores,color=Size, group = Size))+ geom_point() +
  xlab("Number of threads") + ylab("Time * Threads") + ggtitle("Cost of parallel execution")
```


Cost of parallel execution
Time * Threads
1e+02
1e+00
1e−02
1e−04
1
10
Number of threads
Size
1000000
750000
500000
250000