CSI 702 - High Performance Computing
George Mason University

Comparison of Random Forest and Ranger in R for Letter Recognition dataset

Presented By: Ajay Kulkarni (G01024139)

Introduction

- Letter Recognition dataset consist of 20,000 unique letter images of 26 uppercase letters from 20 different commercial fonts.

- The features of each of the 20,000 characters were summaries in terms of 16 primitive numerical attributes.

```
> head(mydata)
  Letter X1 X2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13 X14 X15 X16
1      T  2  8  3  5  1  8 13  0  6   6  10   8   0   8   0   8
2      I  5 12  3  7  2 10  5  5  4  13   3   9   2   8   4  10
3      D  4 11  6  8  6 10  6  2  6  10   3   7   3   7   3   9
4      N  7 11  6  6  3  5  9  4  6   4   4  10   6  10   2   8
5      G  2  1  3  1  1  8  6  6  6   6   5   9   1   7   5  10
6      S  4 11  5  8  3  8  8  6  9   5   6   6   0   8   9   7
```

Source - Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Random Forest

- Random Forest is an ensemble learning method for classification and regression that operates by constructing a lot of decision trees.

- Each tree is trained on roughly 2/3rd of total training data. Some predictor variables are selected at random out of all the predictor variables.

- Each tree gives a classification and we set the tree "votes" for that class. The forest chooses the classification having the most votes over all the trees in the forest.

Source - http://www.listendata.com/2014/11/random-forest-with-r.html

Ranger

- Ranger is a fast implementation of Random Forest for high dimensional data.

- Optimisation of the Ranger is done by the extensive runtime and memory profiling.

- Two different splitting algorithms were used.

- Major improvements were achieved by using the algorithm by Knuth (1985) for sampling without replacement.

Source - ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R by Marvin N. Wright and Andreas Ziegler

doMC and foreach

```
library(randomForest)
library(foreach)
library(doMC)

newdata <- read.csv('letter-recognition.data')

registerDoMC(2)

rf <- foreach(ntree=rep(500,2), .combine = combine, .packages = 'randomForest') %dopar%
randomForest(Letter~X1+X2+X3+X4+X5+X6+X7+X8+X9+X10+X11+X12+X13+X14+X
15+X16,data = newdata,ntree=ntree,confusion=TRUE,importance=TRUE)
```

Performance Matrices

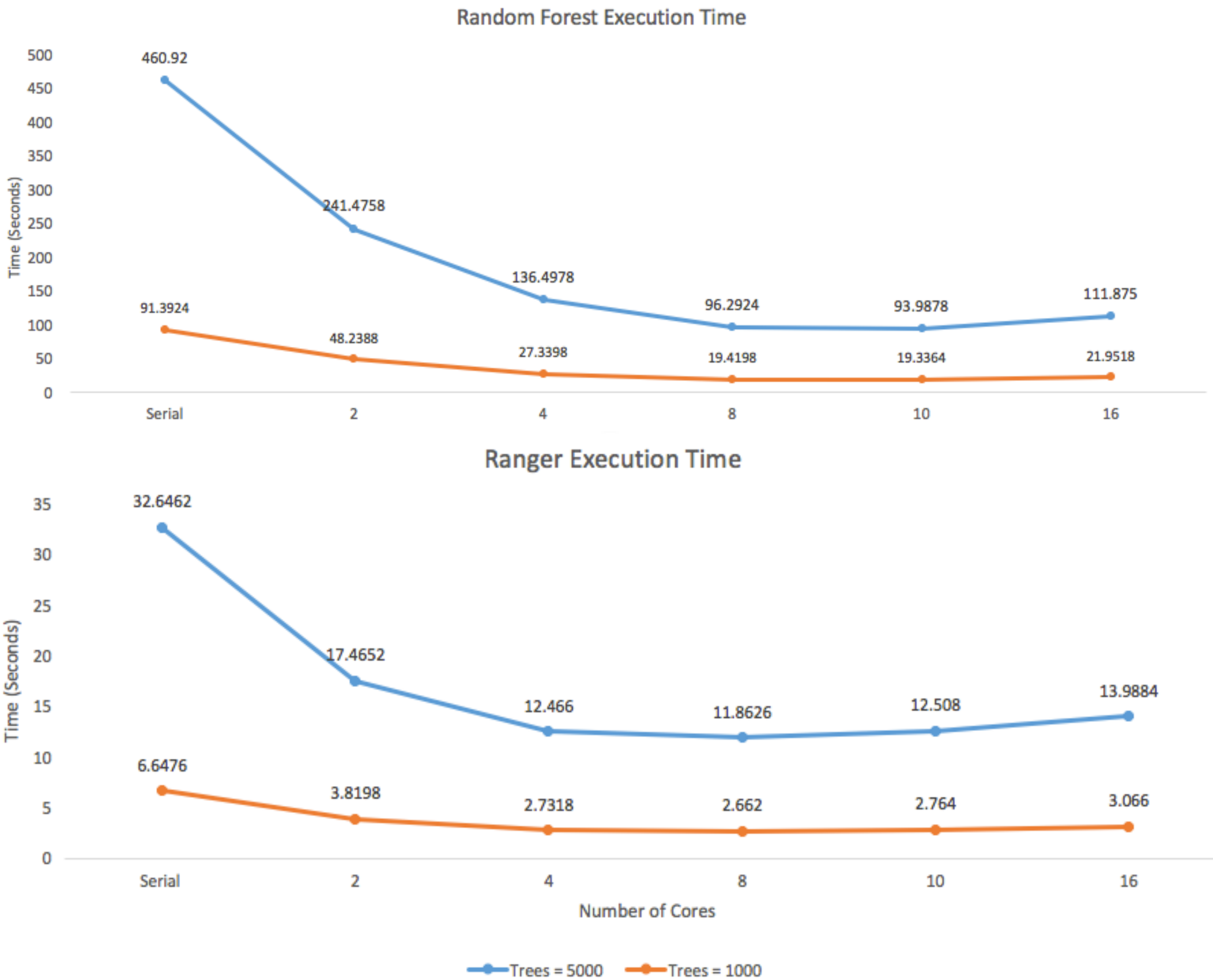
- Four performance matrices were used for the analysis.

  - ✤ Execution Time
  - ✤ Speedup $(S = \frac{T_s}{T_p})$
  - ✤ Efficiency $(E = \frac{S}{P_i})$
  - ✤ Cost $(C = T(P_i) * P_i)$

Source - https://users.info.uvt.ro/~petcu/calcul/PC-2.pdf

Results

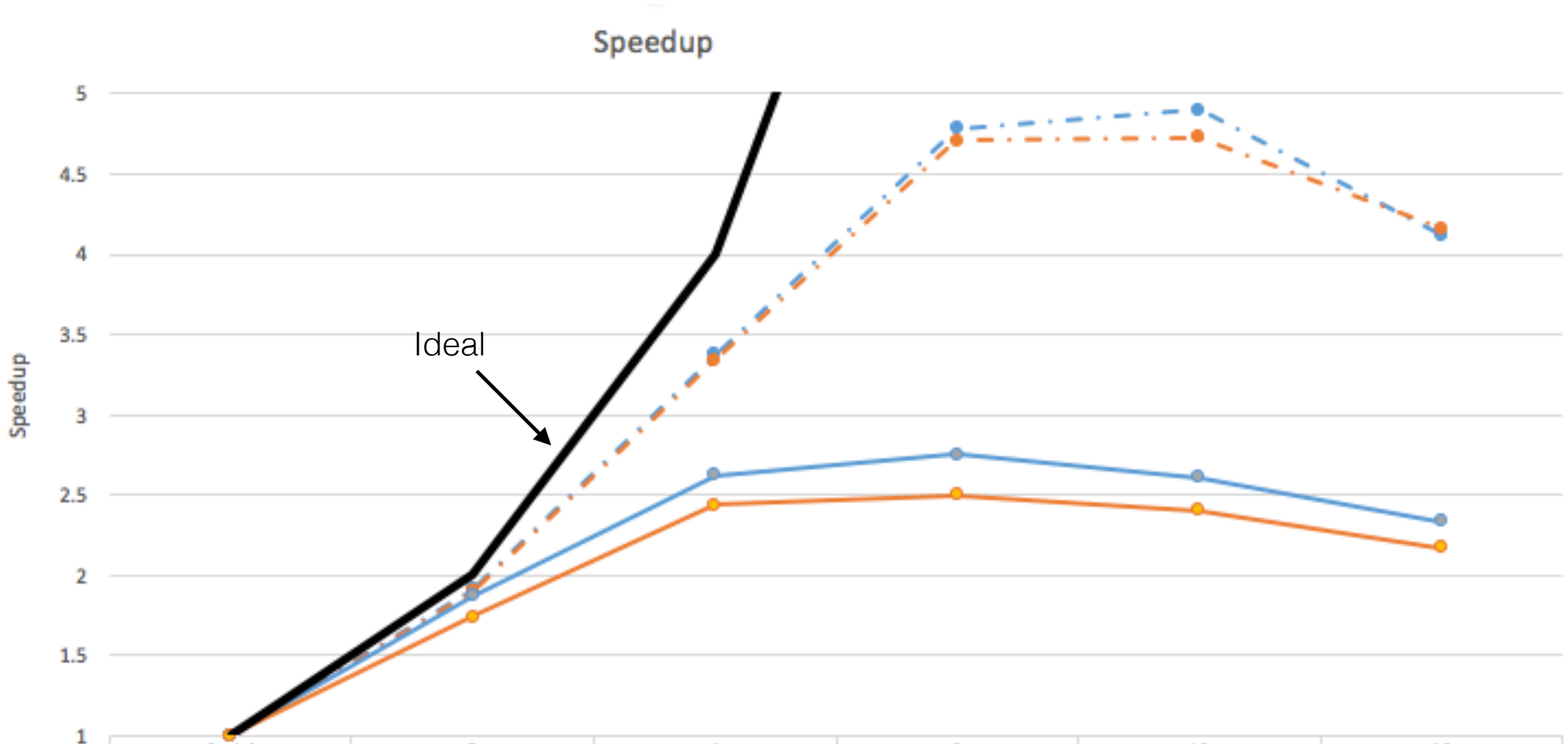
Execution Time



Execution Time

	Random Forest			Ranger		
Number of cores	Time for Trees = 5000 (Sec)	Time for Tress = 1000 (Sec)	Percentage increase	Time for Trees = 5000 (Sec)	Time for Tress = 1000 (Sec)	Percentage increase
1	460.900	91.392	404.31%	32.6462	6.6476	391.1%
2	241.758	48.239	401.17%	17.4652	3.8198	357.23%
4	136.498	27.340	399.26%	12.4660	2.7318	356.33%
8	96.292	19.420	395.84%	11.8626	2.6620	345.63%
10	93.988	19.336	386.08%	12.5080	2.7640	352.53%
16	111.575	21.952	408.27%	13.9884	3.0660	356.24%

Speedup

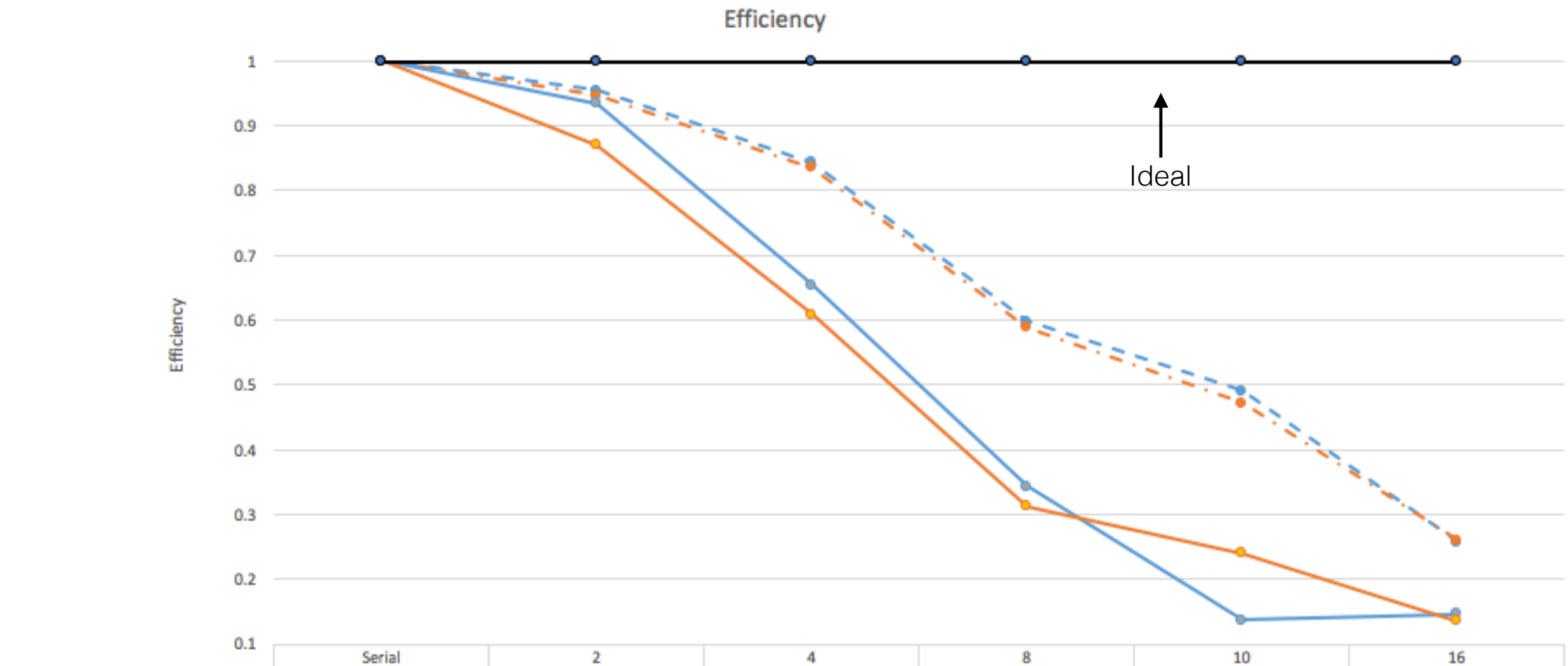


	Serial	2	4	8	10	16
randomForest (ntree = 5000)	1	1.9087627	3.376757721	4.786670599	4.904040737	4.119955307
randomForest (ntree = 1000)	1	1.894582784	3.342833525	4.706145274	4.726443392	4.163321459
ranger (ntree = 5000)	1	1.869214209	2.618819188	2.75202738	2.610025584	2.333805153
ranger (ntree = 1000)	1	1.740300539	2.433413866	2.497220135	2.405065123	2.168166993

Speedup

	Random Forest				Ranger			
Number of cores	Speedup for Trees = 5000	Percentage change in speedup	Speedup for Trees = 1000	Percentage change in speedup	Speedup for Trees = 5000	Percentage change in speedup	Speedup for Trees = 1000	Percentage change in speedup
1	1	-	1	-	1	-	1	-
2	1.90876	90.88%	1.89458	89.46%	1.86921	86.92%	1.74030	74.03%
4	3.37676	237.68%	3.34283	234.28%	2.61882	161.88%	2.43341	143.34%
8	4.78667	378.67%	4.70615	370.62%	*2.75203*	*175.2%*	*2.49722*	*149.72%*
10	*4.90404*	*390.4%*	*4.72644*	*372.64%*	2.61003	161%	2.40507	140.51%
16	4.11996	312%	4.16332	316.33%	2.33381	133.38%	2.16817	116.82%

Efficiency

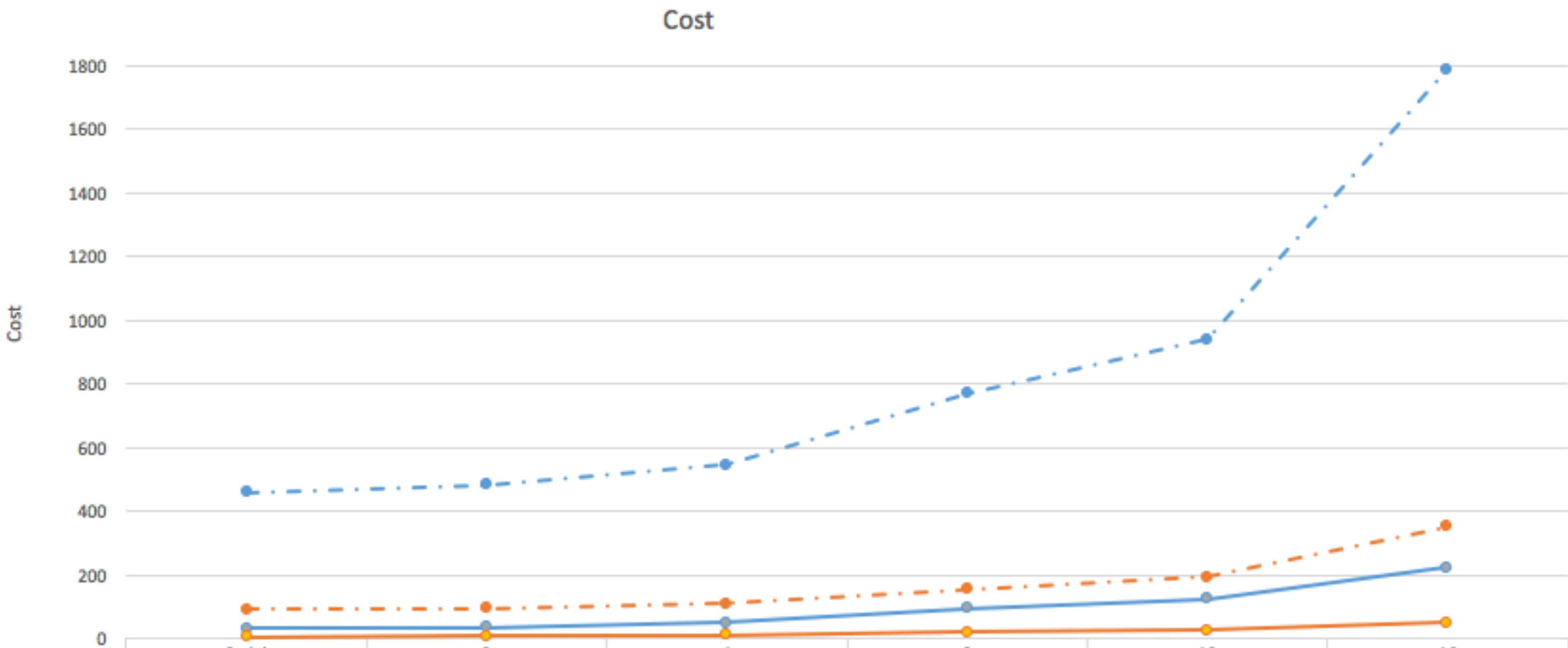


	Serial	2	4	8	10	16
randomForest (ntree = 5000)	1	0.95438135	0.84418943	0.598333825	0.490404074	0.257497207
randomForest (ntree = 1000)	1	0.947291392	0.835708381	0.588268159	0.472644339	0.260207591
ranger (ntree = 5000)	1	0.934607104	0.654704797	0.344003423	0.137369768	0.145862822
ranger (ntree = 1000)	1	0.87015027	0.608353467	0.312152517	0.240506512	0.135510437

Efficiency

	Random Forest				Ranger			
Number of cores	Efficiency for Trees = 5000	Percentage decrease in efficiency	Efficiency for Trees = 1000	Percentage decrease in efficiency	Efficiency for Trees = 5000	Percentage decrease in efficiency	Efficiency for Trees = 1000	Percentage decrease in efficiency
1	1	-	1	-	1	-	1	-
2	0.95438	4.56%	0.94729	5.27%	0.93461	6.54%	0.87015	12.99%
4	0.84419	15.58%	0.83571	16.43%	0.65470	34.53%	0.60835	39.17%
8	0.59833	40.17%	0.58827	41.17%	0.34400	65.6%	0.31215	68.79%
10	0.49040	50.96%	0.47264	52.74%	*0.13737*	*86.26%*	0.24050	75.95%
16	*0.25750*	*74.25%*	*0.26021*	*73.98%*	0.14586	85.41%	*0.13551*	*86.45%*

Cost



	Serial	2	4	8	10	16
randomForest (ntree = 5000)	460.92	482.9516	545.9912	770.3392	939.878	1790
randomForest (ntree = 1000)	91.3924	96.4776	109.3592	155.3584	193.364	351.2288
ranger (ntree = 5000)	32.6462	34.9304	49.864	94.9008	125.08	223.8144
ranger (ntree = 1000)	6.6476	7.6396	10.9272	21.296	27.64	49.056

Random Forest Profiling

```
> newdata <- read.csv('letter-recognition.data')
> Rprof()
> rf <- randomForest(Letter~X1+X2+X3+X4+X5+X6+X7+X8+X9+X10+X11+X12+X13+X14+X15+X16,data = newdata,ntree=1000,importance=TRUE)
> Rprof(NULL)
> summaryRprof(rf)
Error in file(filename, "rt") : invalid 'description' argument
> summaryRprof()
$by.self
                      self.time self.pct total.time total.pct
".C"                      87.90    94.46      87.90     94.46
"matrix"                   1.86     2.00       1.86      2.00
"array"                    0.88     0.95       0.88      0.95
"randomForest.default"     0.74     0.80      93.04     99.98
"integer"                  0.66     0.71       0.66      0.71
"aperm.default"            0.62     0.67       0.62      0.67
"double"                   0.26     0.28       0.26      0.28
"apply"                    0.04     0.04       0.08      0.09
"/"                        0.02     0.02       0.02      0.02
"cbind"                    0.02     0.02       0.02      0.02
"data.matrix"              0.02     0.02       0.02      0.02
"na.omit.data.frame"       0.02     0.02       0.02      0.02
"t.default"                0.02     0.02       0.02      0.02

$by.total
                      total.time total.pct self.time self.pct
"randomForest"             93.06    100.00      0.00     0.00
"randomForest.formula"     93.06    100.00      0.00     0.00
"randomForest.default"     93.04     99.98      0.74     0.80
".C"                       87.90     94.46     87.90    94.46
"matrix"                    1.86      2.00      1.86     2.00
"aperm"                     1.50      1.61      0.00     0.00
"array"                     0.88      0.95      0.88     0.95
"integer"                   0.66      0.71      0.66     0.71
"aperm.default"             0.62      0.67      0.62     0.67
"double"                    0.26      0.28      0.26     0.28
"t"                         0.10      0.11      0.00     0.00
"apply"                     0.08      0.09      0.04     0.04
"/"                         0.02      0.02      0.02     0.02
"cbind"                     0.02      0.02      0.02     0.02
"data.matrix"               0.02      0.02      0.02     0.02
"na.omit.data.frame"        0.02      0.02      0.02     0.02
"t.default"                 0.02      0.02      0.02     0.02
"do.call"                   0.02      0.02      0.00     0.00
".External2"                0.02      0.02      0.00     0.00
"FUN"                       0.02      0.02      0.00     0.00
"is.na"                     0.02      0.02      0.00     0.00
"is.na.data.frame"          0.02      0.02      0.00     0.00
"model.frame"               0.02      0.02      0.00     0.00
"model.frame.default"       0.02      0.02      0.00     0.00
"na.omit"                   0.02      0.02      0.00     0.00

$sample.interval
```

Ranger Profiling

```
> library(ranger)

> library(tictoc)
> newdata <- read.csv('letter-recognition.data')
> Rprof()
> rf <- ranger(Letter~X1+X2+X3+X4+X5+X6+X7+X8+X9+X10+X11+X12+X13+X14+X15+X16,data = newdata,num.tree=1000)
> Rprof(NULL)
> summaryRprof()
$by.self
               self.time self.pct total.time total.pct
".Call"            59.26    99.97      59.26     99.97
"as.character"      0.02     0.03       0.02      0.03

$by.total
                   total.time total.pct self.time self.pct
"ranger"                59.28    100.00      0.00     0.00
".Call"                 59.26     99.97     59.26    99.97
"rangerCpp"             59.26     99.97      0.00     0.00
"as.character"           0.02      0.03      0.02     0.03
"factor"                 0.02      0.03      0.00     0.00
"integer.to.factor"      0.02      0.03      0.00     0.00

$sample.interval
[1] 0.02

$sampling.time
[1] 59.28
```

Conclusion

- Ranger is fast as compared to Random Forest in serial as well as in parallel execution.

- Random Forest has a better accuracy and efficiency as compared to the Ranger in serial as well as in parallel execution.

- Random Forest is very costly as compared to the Ranger.

- In practice, Ranger is better as compared to the Random Forest.

References

- P. W. Frey and D. J. Slate. "Letter Recognition Using Holland-style Adaptive Classifiers". (Machine Learning Vol 6 #2 March 91)
- http://www.glennklockwood.com/data-intensive/r/on-hpc.html
- http://www.glennklockwood.com/data-intensive/r/parallel-options.html
- http://www.bu.edu/tech/support/research/training-consulting/online-tutorials/matlab-pct/scalability/
- https://users.info.uvt.ro/~petcu/calcul/PC-2.pdf
- https://cran.r-project.org/web/packages/tictoc/tictoc.pdf
- https://cran.r-project.org/web/packages/doMC/vignettes/gettingstartedMC.pdf
- https://www.r-bloggers.com/profiling-r-code/

Thank You