Julian Garcia
Cse122

Blackbox Summary

Function 1:

Testing Details:

I began with n = 100 and added 100 ten times to it.

Big O:

$O(n^3)$

Based on the graph at the end of the document, and on the fact that the standard error was 0.2973%, I can conclude that this is the big o.

Gnuplot Details:

```
Final set of parameters            Asymptotic Standard Error
=======================            ==========================

c               = 0.00226147       +/- 6.724e-06    (0.2973%)
```

Function 2:

Testing Details:

Used n from 10 to 30, incrementing by one.

Big O:

$O(2^n)$

Based on the graph at the end of the document, and on the fact that the standard error was 0.212%, I can conclude that this is the big o.

Gnuplot Details:

```
Final set of parameters                Asymptotic Standard Error
=======================                ==========================

c                 = 0.00440304         +/- 9.334e-06(0.212%)
```

Julian Garcia
Cse122

Function 3:

Shuffled List:

Testing Details:

Started n at 100000 and added 100000 ten times.

Big O:

O(n)

Based on the graph at the end of the document, and on the fact that the Standard error was 1.294%, I can assume that this is the big o.

Gnuplot Details:

After 3 iterations the fit converged.
final sum of squares of residuals : 1.26246e+08
rel. change during last iteration : -4.23758e-06

degrees of freedom (FIT_NDF) : 9
rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 3745.3
variance of residuals (reduced chisquare) = WSSR/ndf : 1.40273e+07

Final set of parameters		Asymptotic Standard Error
c	= 0.147527	+/- 0.001909 (1.294%)

Reverse Sorted List:

Testing Details:

Started at n = 10 and multiplied by 10 seven times

Big O:

O(n)

Based on the graph at the end of the document, and on the fact that the Standard error was 0.57%, I can assume that this is the big o.

Gnuplot Details:

After 3 iterations the fit converged.
final sum of squares of residuals : 3.68106e+07
rel. change during last iteration : -3.70976e-06

degrees of freedom (FIT_NDF) : 6
rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 2476.91
variance of residuals (reduced chisquare) = WSSR/ndf : 6.13511e+06

Final set of parameters		Asymptotic Standard Error	
c	= 0.0429129	+/- 0.0002464	(0.5743%)

Sorted List:

Testing Details:

Started at n = 10 and multiplied by 10 seven times

Big O:

O(n)

Based on the graph at the end of the document, and on the fact that the Standard error was 0.6467%, I can assume that this is the big o.

Gnuplot Details:

After 4 iterations the fit converged.
final sum of squares of residuals : 8.84994e+08
rel. change during last iteration : -1.4413e-14

degrees of freedom (FIT_NDF) : 6
rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 12144.9
variance of residuals (reduced chisquare) = WSSR/ndf : 1.47499e+08

Final set of parameters		Asymptotic Standard Error	
c	= 0.186867	+/- 0.001208	(0.6467%)

Julian Garcia
Cse122

Analysis:

The reverse sorted list appears to be the best case. Using the same n and incrementation, the average time for the 7th run was higher on the sorted list than the reverse list. I ran them both up to the 7th run 5 times and came to an average runtime of 1828196 ticks for the sorted list and 428535 ticks for the reverse list.

Overall though, the big O of the algorithm is O(n) for all cases.

Function 4:

Testing Details:

Used n = 10 and multiplied by ten 8 times

Big O:

O(nlog(n))

Based on the graph at the end of the document, and on the fact that the standard error was 0.8643%, I can conclude that this is the big o.

Gnuplot Details:

After 4 iterations the fit converged.
final sum of squares of residuals : 2.35912e+11
rel. change during last iteration : -1.73872e-12

degrees of freedom (FIT_NDF) : 7
rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 183580
variance of residuals (reduced chisquare) = WSSR/ndf : 3.37018e+10

Final set of parameters		Asymptotic Standard Error
c	= 0.0114867	+/- 9.928e-05(0.8643%)

Function 5:

Testing Details:

Used n = 1000000 and increased by a factor of 10, ten times.

Big O:

O(log(n))

Based on the graph at the end of the document, and on the fact that the standard error was 0.689%, I can conclude that this is the big o.

Gnuplot Details:

```
Final set of parameters            Asymptotic Standard Error
======================             ==========================

c               = 2855.03          +/- 19.67       (0.689%)
```

Function 6:

Sorted List:

Testing Details:

Started at n = 10 and multiplied by 10 twenty times.

Big O:

$O(n^2 log(n))$

Based on the graph at the end of the document, and on the fact that the Standard error was 1.086%, I can assume that this is the big o.

Gnuplot Details:

```
After 3 iterations the fit converged.
final sum of squares of residuals : 5.02417e+09
rel. change during last iteration : -2.52745e-07

degrees of freedom (FIT_NDF)                        : 19
rms of residuals      (FIT_STDFIT) = sqrt(WSSR/ndf)  : 16261.3
variance of residuals (reduced chisquare) = WSSR/ndf  : 2.6443e+08

Final set of parameters            Asymptotic Standard Error
======================             ==========================
```

c = 0.000205977 +/- 2.237e-06(1.086%)

Reverse Sorted List:

Testing Details:

Started at n = 10 and multiplied by 10 one thousand times to see if the function was linear since smaller sample sizes made it seem like it was O(1).

Big O:

O(n)

Based on the graph at the end of the document, and on the fact that the Standard error was 1.315%, I can assume that this is the big o.

Gnuplot Details:

After 1 iterations the fit converged.
final sum of squares of residuals : 8.67934e+08
rel. change during last iteration : -1.37348e-16

degrees of freedom (FIT_NDF) : 999
rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 932.096
variance of residuals (reduced chisquare) = WSSR/ndf : 868803

Final set of parameters	Asymptotic Standard Error
c = 0.00388419	+/- 5.109e-05(1.315%)

Shuffle List:

Testing Details:

Started at n = 10 and multiplied by 10 ten times.

Big O:

$O(n^2 log(n))$

Based on the graph at the end of the document, and on the fact that the Standard error was 1.22%, I can assume that this is the big o.

Gnuplot Details:

After 1 iterations the fit converged.
final sum of squares of residuals : 2.40524e+08
rel. change during last iteration : -1.48687e-15

degrees of freedom (FIT_NDF) : 9
rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 5169.61
variance of residuals (reduced chisquare) = WSSR/ndf : 2.67248e+07

Final set of parameters		Asymptotic Standard Error
c	= 0.000381069	+/- 4.648e-06(1.22%)

Analysis:

The best case of this function would be when the data is reverse sorted, with a big o of O(n). The worst case is when its regularly sorted, with a big o of $O(n^2 log(n))$. The average case, with the shuffled list, also has a big o of $O(n^2 log(n))$.

Function 7:

Testing Details:

Used n = 1 to 10, incrementing by one.

Big O:

O(n!)

Based on the graph at the end of the document, and on the fact that the standard error was only 0.18%, I can conclude that this is the big o.

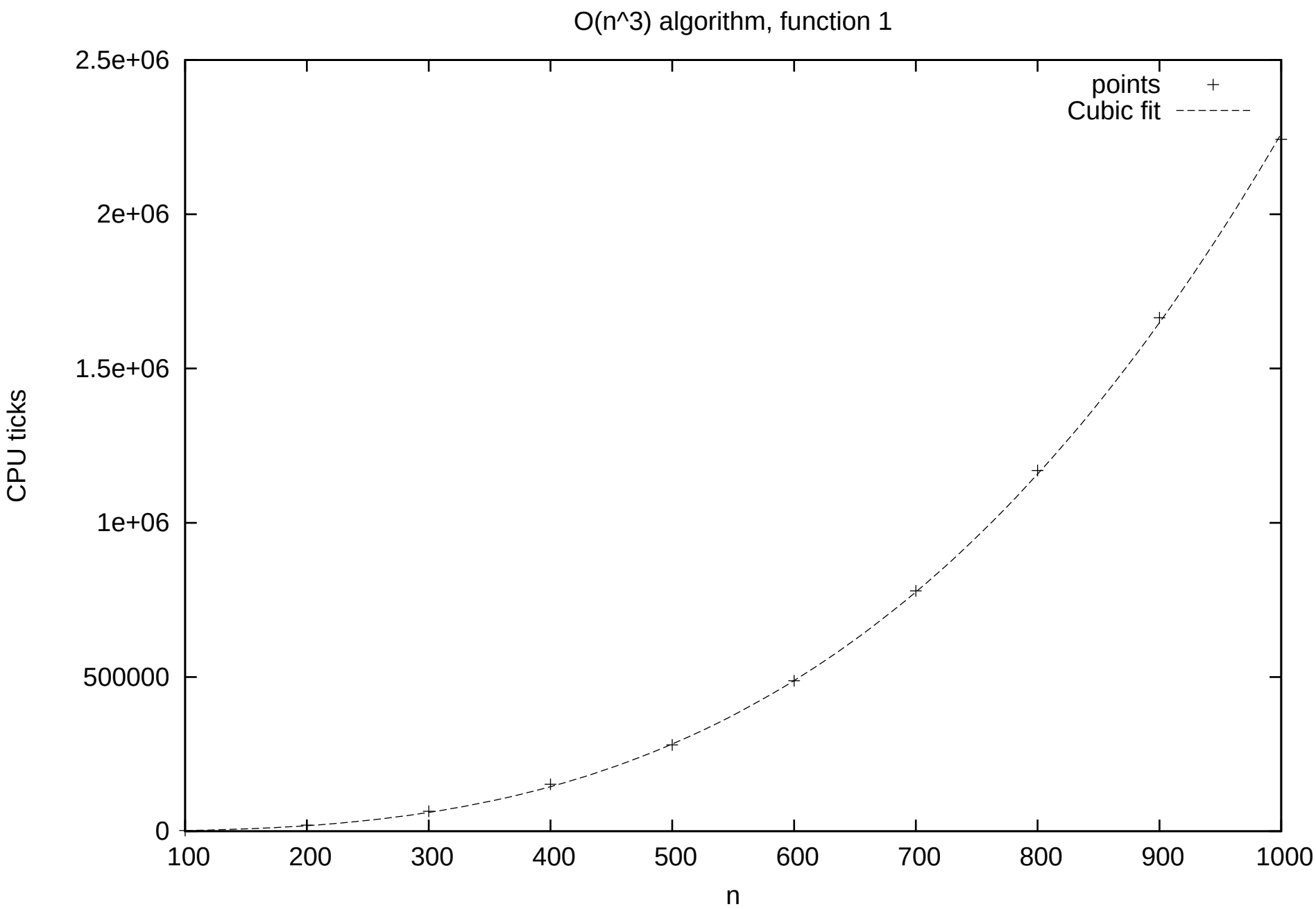
Gnuplot Details:

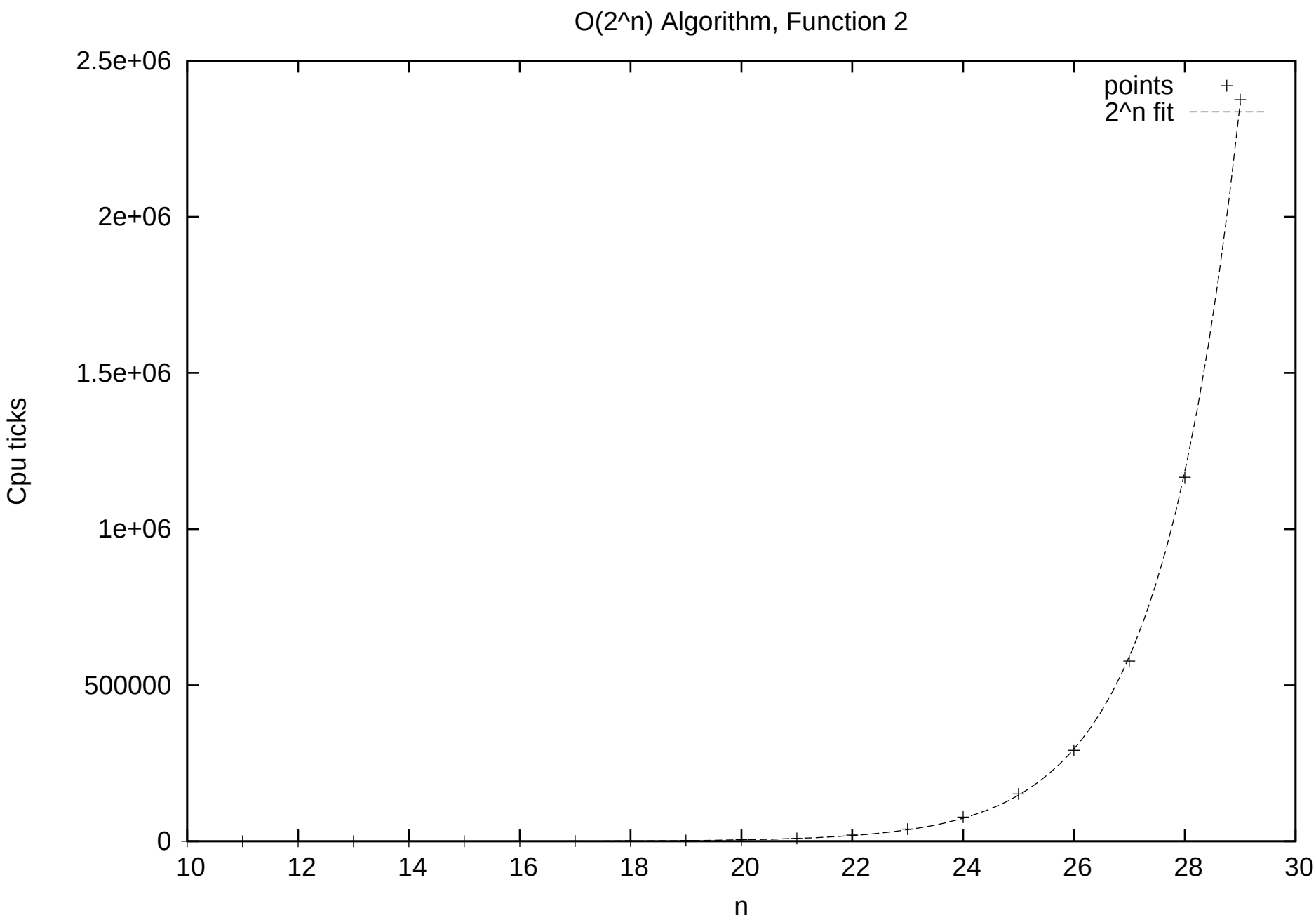
After 4 iterations the fit converged.
final sum of squares of residuals : 125907
rel. change during last iteration : -2.14628e-07

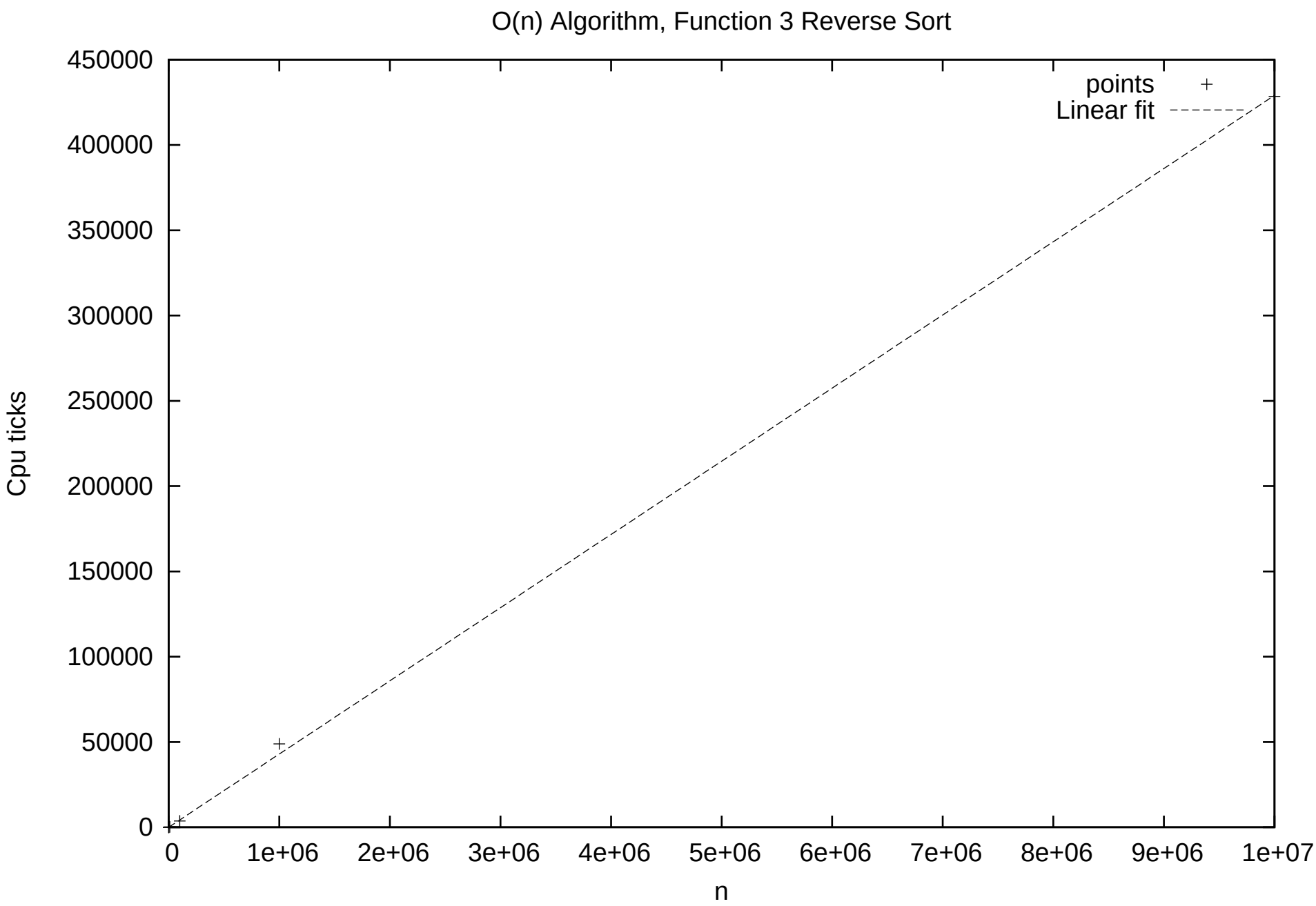
degrees of freedom (FIT_NDF) : 9

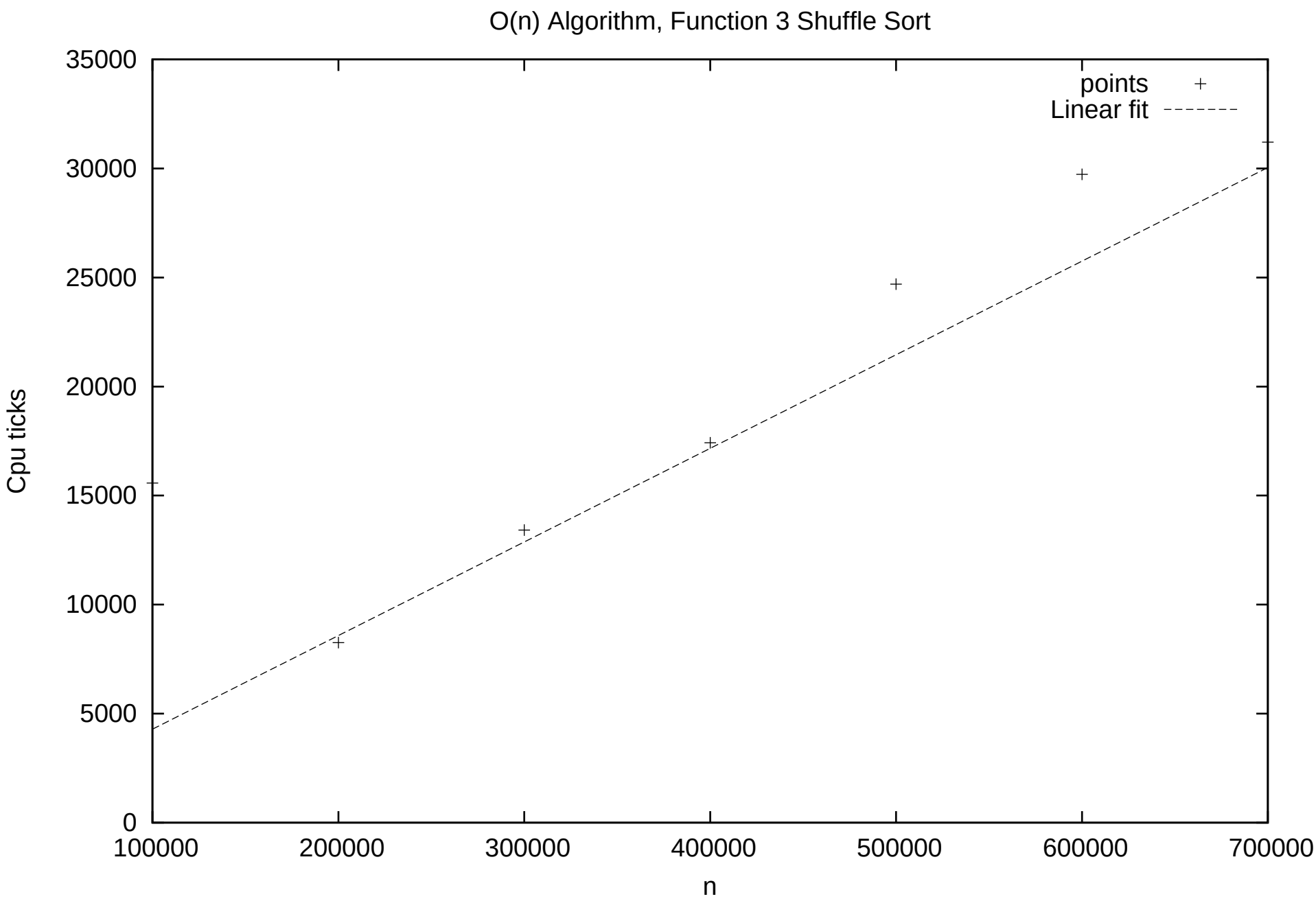
rms of residuals (FIT_STDFIT) = sqrt(WSSR/ndf) : 118.278
variance of residuals (reduced chisquare) = WSSR/ndf : 13989.7

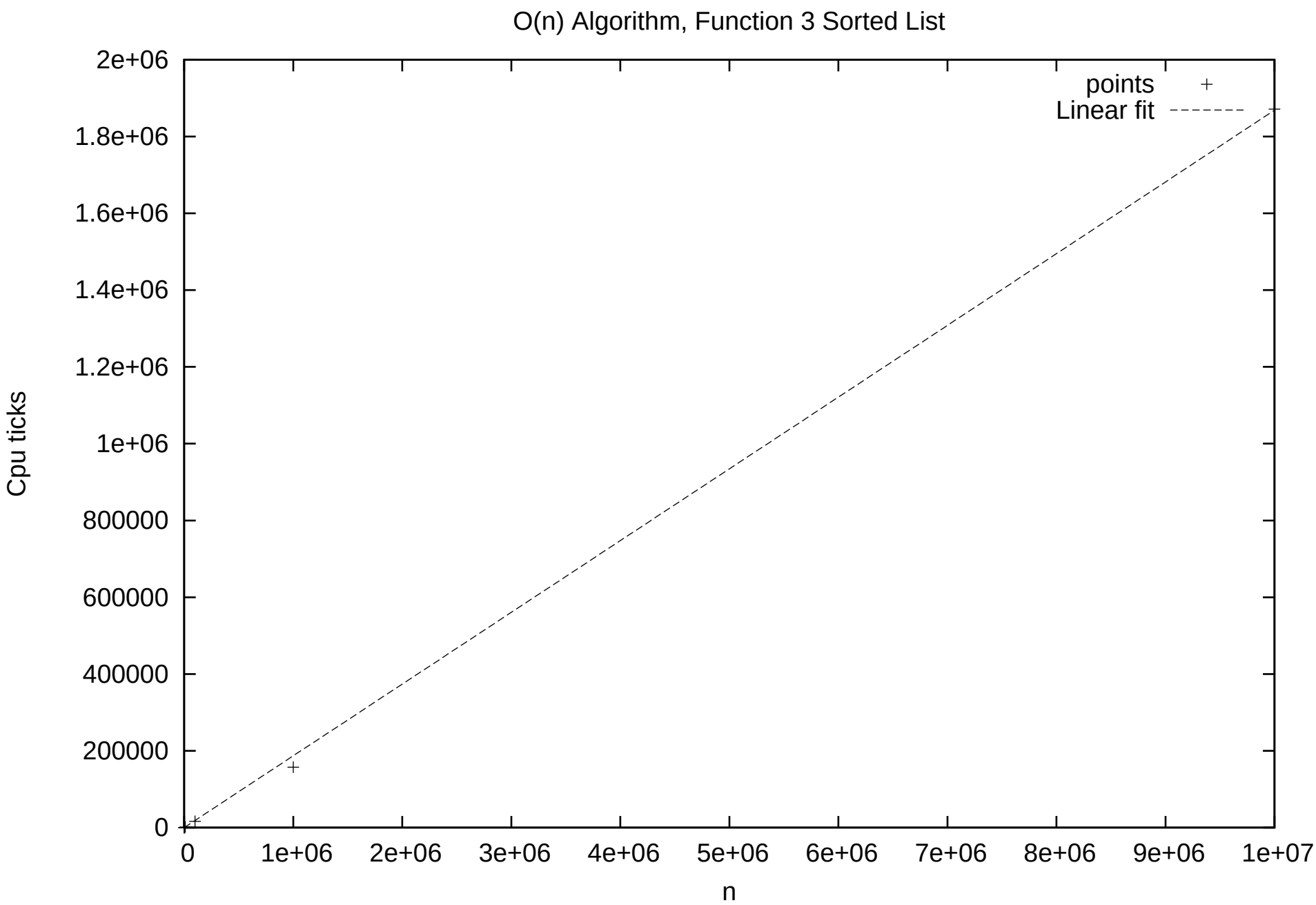
Final set of parameters		Asymptotic Standard Error
c	= 0.0177303	+/- 3.243e-05(0.1829%)

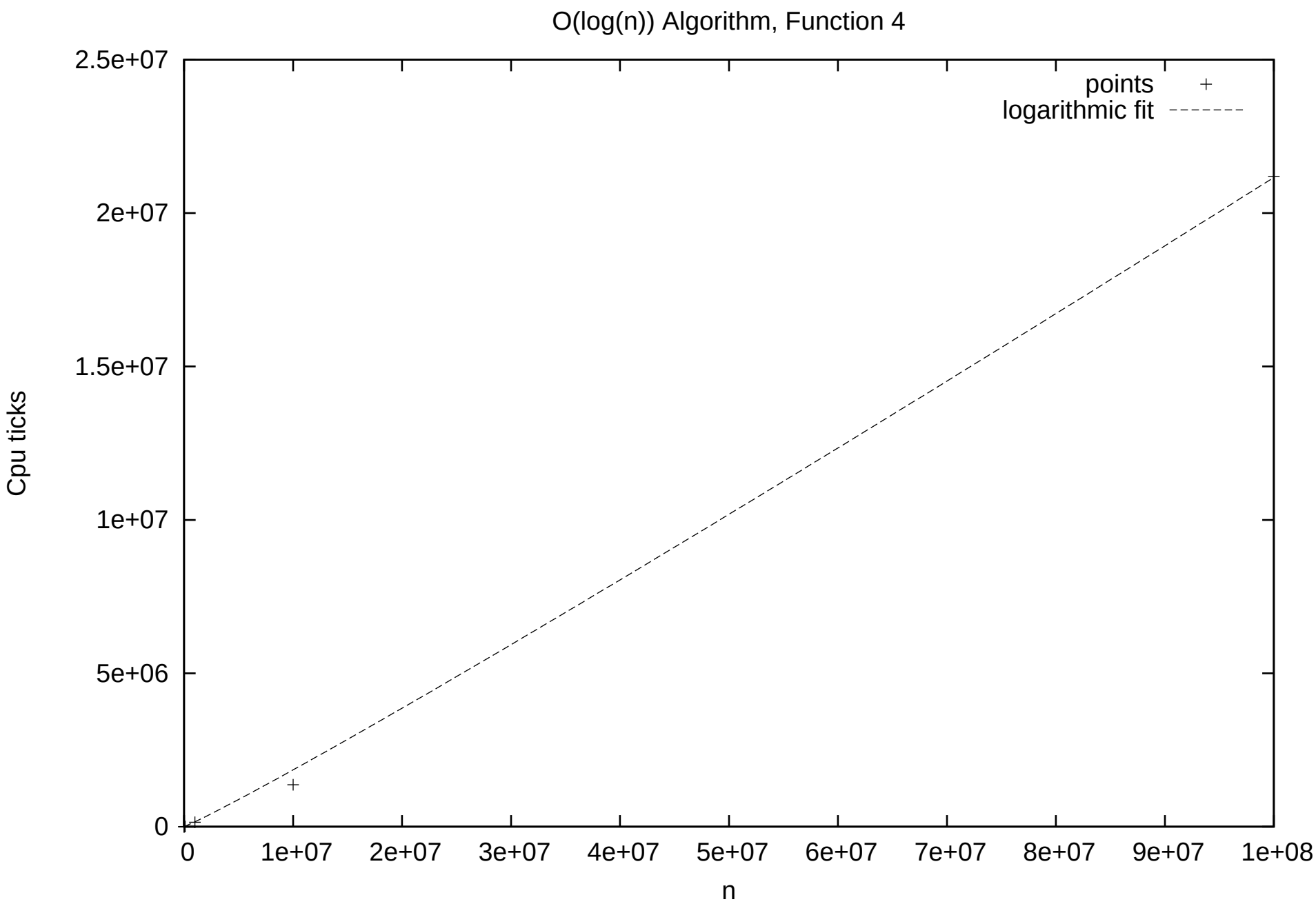

O(n^3) algorithm, function 1
points
Cubic fit
CPU ticks
n
2.5e+06
2e+06
1.5e+06
1e+06
500000
0
100
200
300
400
500
600
700
800
900
1000

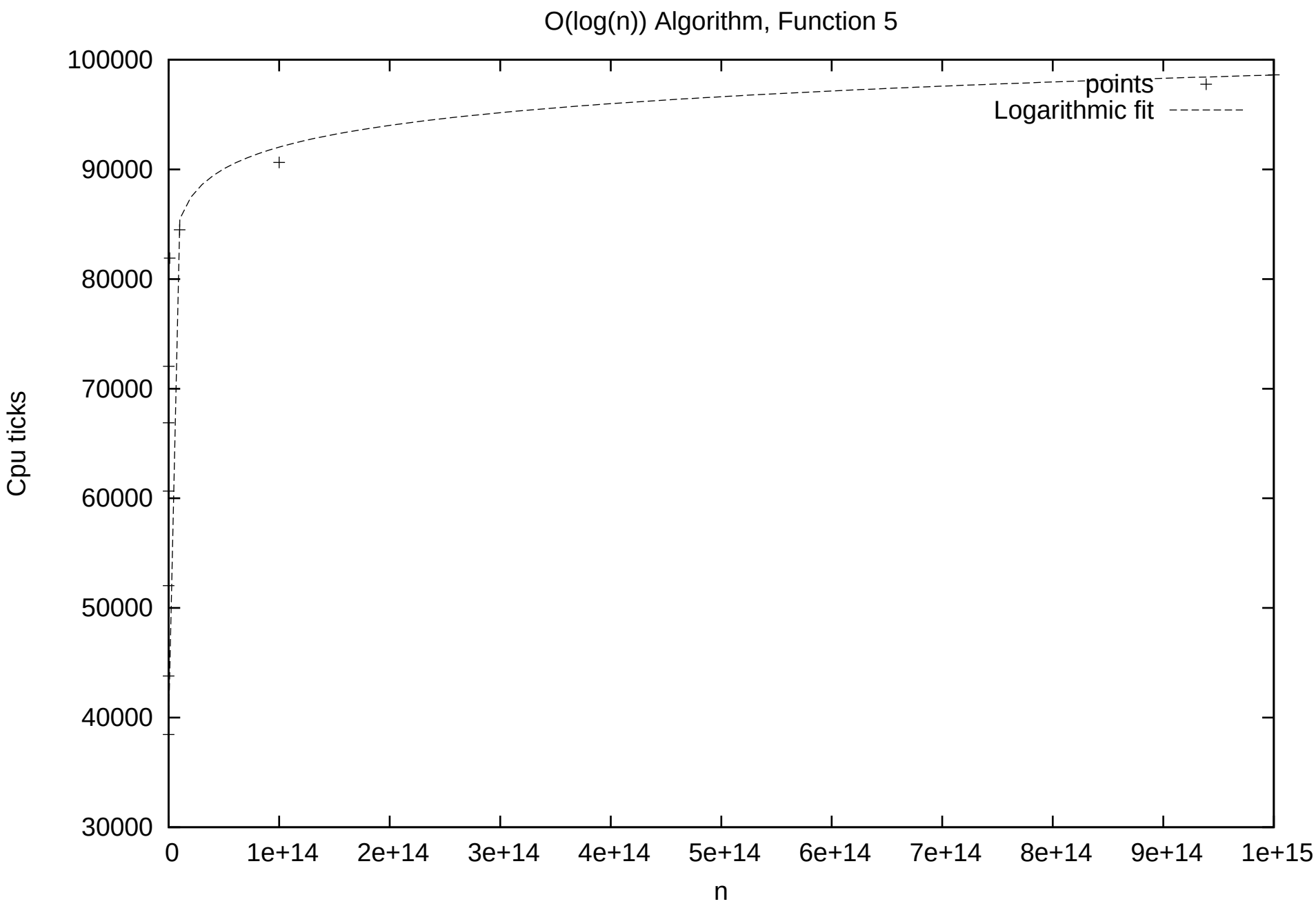

O(2^n) Algorithm, Function 2
points
2^n fit
Cpu ticks
n
2.5e+06
2e+06
1.5e+06
1e+06
500000
0
10
12
14
16
18
20
22
24
26
28
30

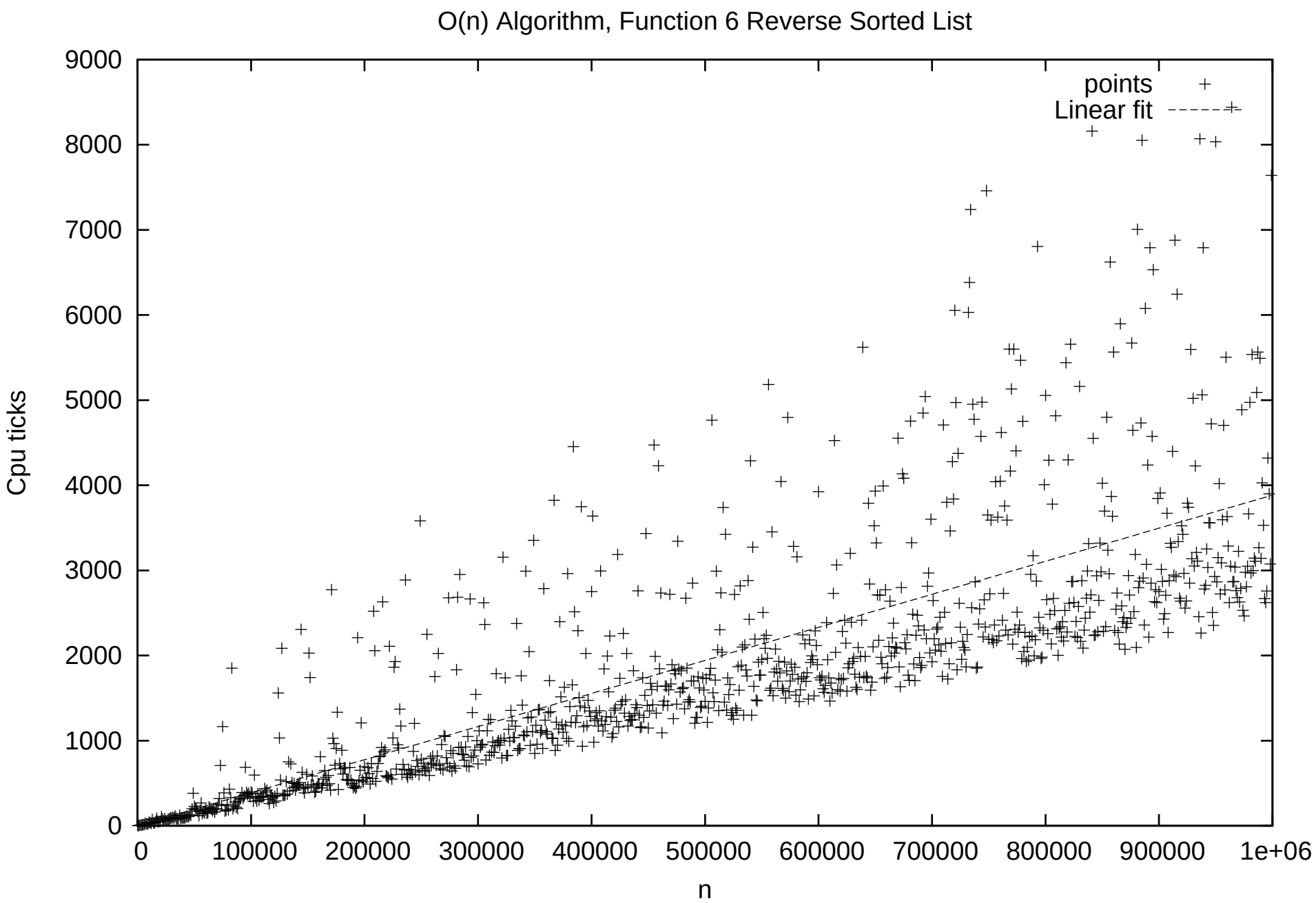

O(n) Algorithm, Function 3 Reverse Sort
Cpu ticks
n
points
Linear fit
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
0
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
7e+06
8e+06
9e+06
1e+07

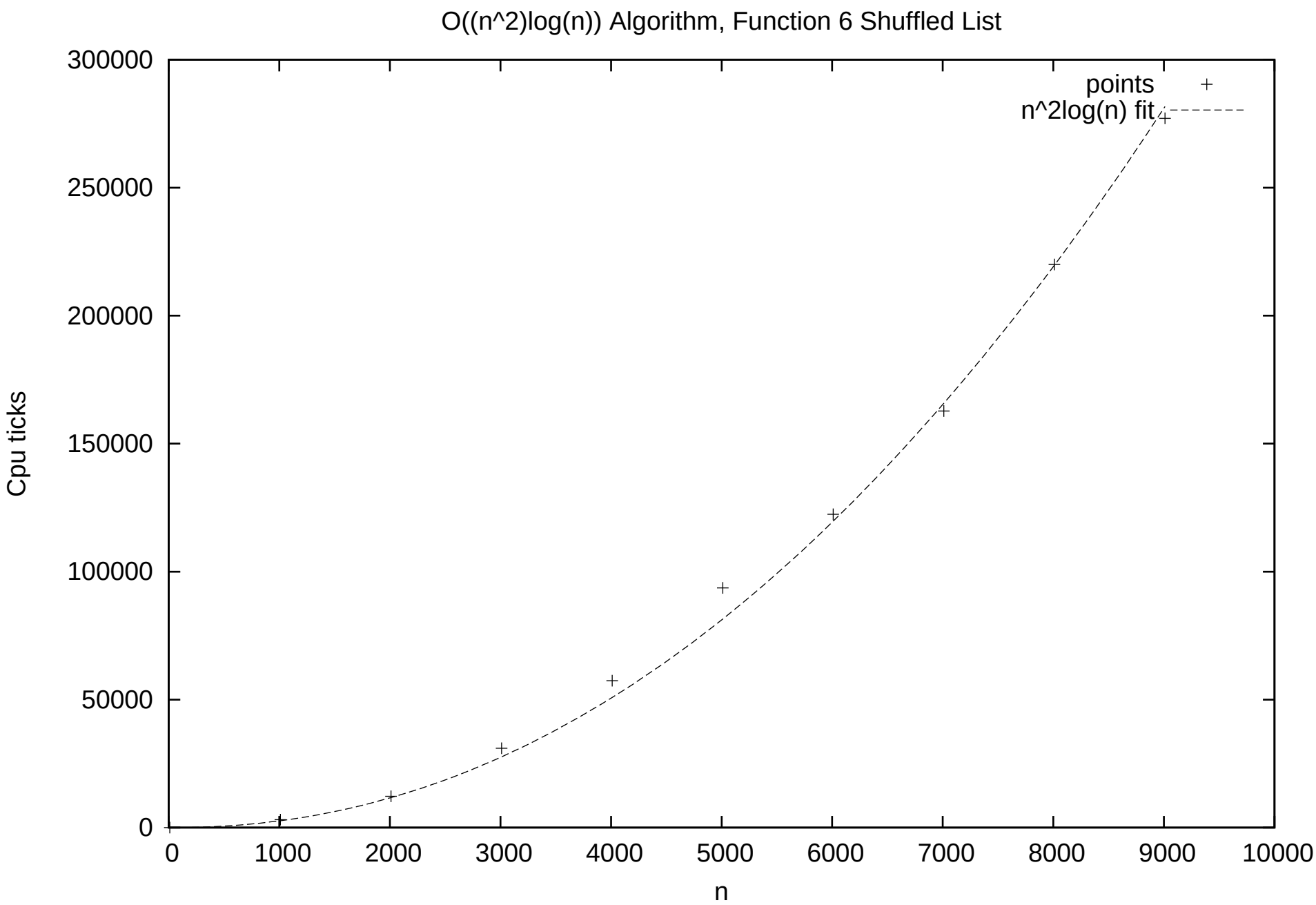

O(n) Algorithm, Function 3 Shuffle Sort
points
Linear fit
Cpu ticks
n
0
5000
10000
15000
20000
25000
30000
35000
100000
200000
300000
400000
500000
600000
700000

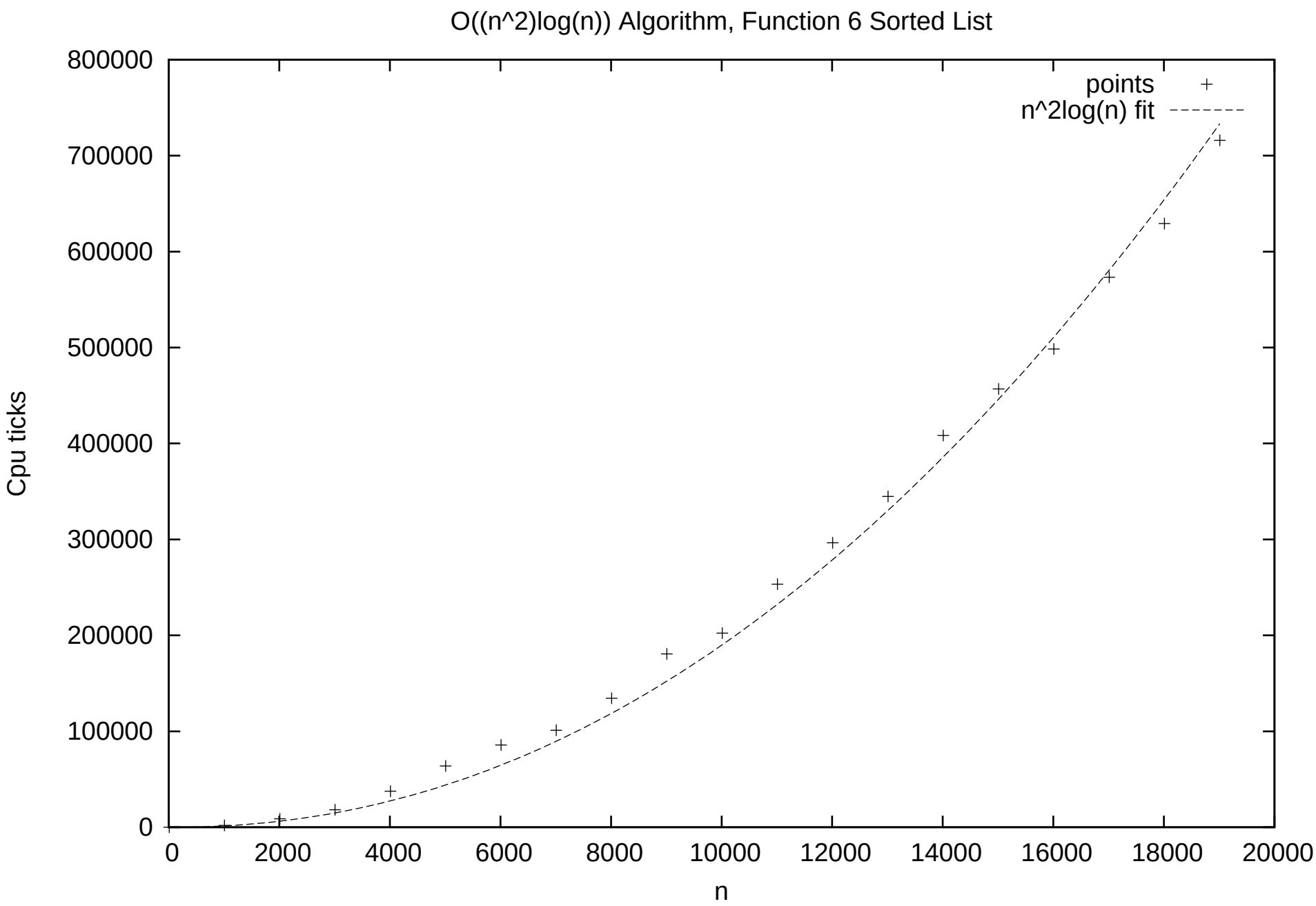

O(n) Algorithm, Function 3 Sorted List
points
Linear fit
Cpu ticks
n
0
200000
400000
600000
800000
1e+06
1.2e+06
1.4e+06
1.6e+06
1.8e+06
2e+06
0
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
7e+06
8e+06
9e+06
1e+07

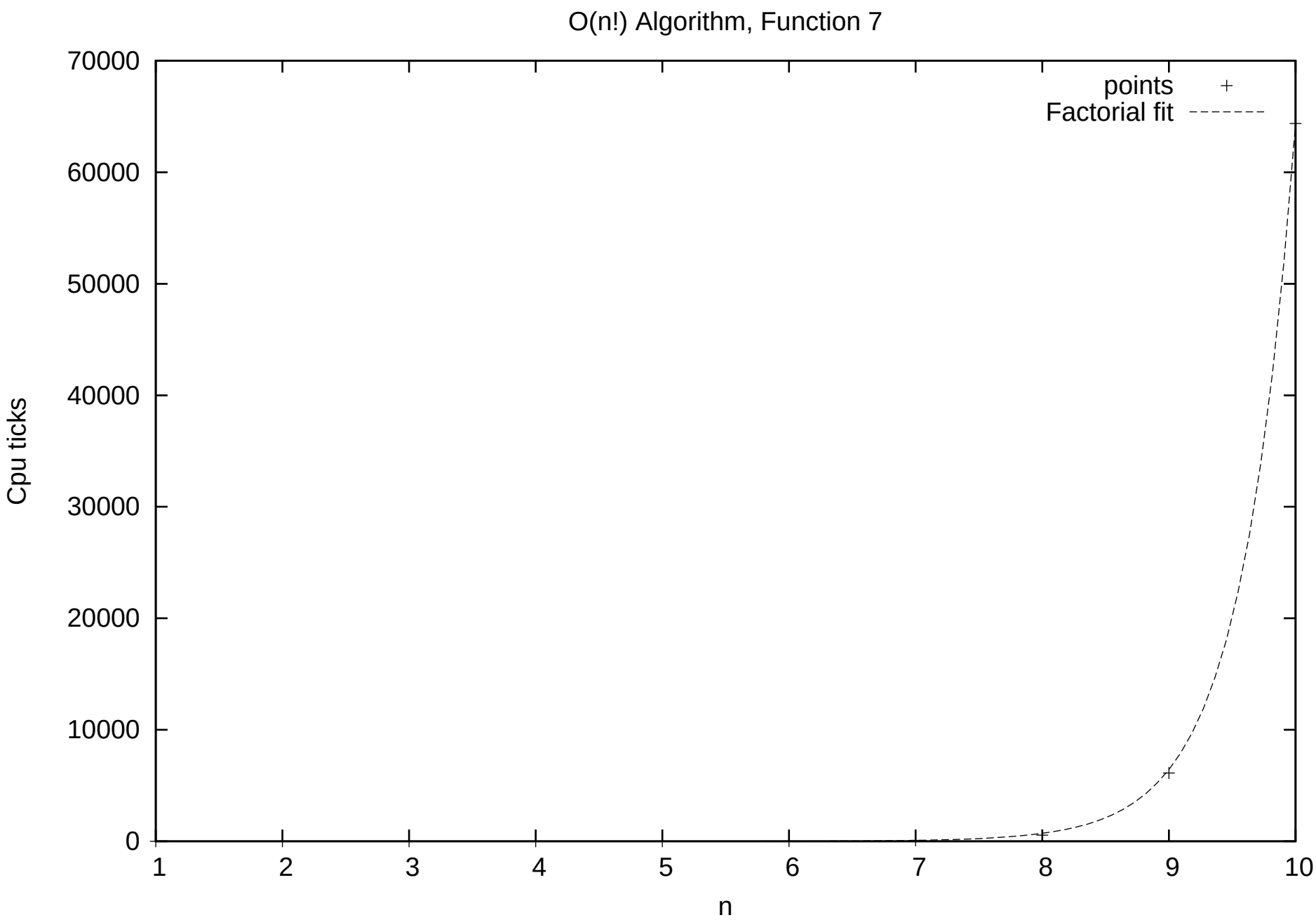

O(log(n)) Algorithm, Function 4
points
logarithmic fit
Cpu ticks
n
2.5e+07
2e+07
1.5e+07
1e+07
5e+06
0
0
1e+07
2e+07
3e+07
4e+07
5e+07
6e+07
7e+07
8e+07
9e+07
1e+08


O(log(n)) Algorithm, Function 5
Cpu ticks
n
points
Logarithmic fit
100000
90000
80000
70000
60000
50000
40000
30000
0
1e+14
2e+14
3e+14
4e+14
5e+14
6e+14
7e+14
8e+14
9e+14
1e+15


O(n) Algorithm, Function 6 Reverse Sorted List
points
Linear fit
Cpu ticks
n
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
0
100000
200000
300000
400000
500000
600000
700000
800000
900000
1e+06


O((n^2)log(n)) Algorithm, Function 6 Shuffled List
points
n^2log(n) fit
Cpu ticks
n
0
50000
100000
150000
200000
250000
300000
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
10000


O((n^2)log(n)) Algorithm, Function 6 Sorted List
points
n^2log(n) fit
Cpu ticks
n
0
100000
200000
300000
400000
500000
600000
700000
800000
0
2000
4000
6000
8000
10000
12000
14000
16000
18000
20000


O(n!) Algorithm, Function 7
Cpu ticks
n
points
Factorial fit
0
10000
20000
30000
40000
50000
60000
70000
1
2
3
4
5
6
7
8
9
10