

Automatic Selection of Compiler Options Using Non-parametric Inferential Statistics

Masayo Haneda
Peter M.W. Knijnenburg
Harry A.G. Wijshoff
LIACS, Leiden University

Motivation

- An optimal compiler optimization setting can be obtained by considering the interaction between applications, architectures, and compiler optimizations.
- Profiling is the best way to understand this interaction.
- However a huge number of optimization settings exists.
- Find the optimal configuration with limited amount of profiling.

Outline of Our Approach

- Collect profiling data with an appropriate experimental setting
 - Orthogonal Arrays, which are well known in the Design of Experiments, are used for this purpose
- Apply Inferential Statistics to the profiling data to detect effective compiler options
 - Mann-Whitney test of non-parametric inferential statistics is employed

Orthogonal Arrays

- Orthogonal arrays (OAs) are well chosen fractional factorial designs.
- An OA is expressed as a $N \times k$ matrix of 0s and 1s.
- The columns are interpreted as factors (compiler options).
- Each row of an array defines a compiler setting.

Orthogonal Arrays (cont'd)

- An OA has a property that two arbitrary columns contain the patterns

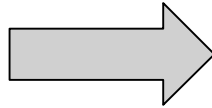
00 01 10 11

equally often.

- According to this property,
 - each compiler option is turned on and off equally often.
 - When we drop columns of an OA, the array is still an orthogonal array.
- An OA exists when the number of rows N is a multiple of 4

Example

0	0	0	0	0
1	0	0	1	1
0	1	0	1	0
0	0	1	0	1
1	1	0	0	1
1	0	1	1	0
0	1	1	1	1
1	1	1	0	0



Interpreted as
Compiler Settings

	O ₁	O ₂	O ₃	O ₄	O ₅
Run1	off	off	off	off	off
Run2	on	off	off	on	on
Run3	off	on	off	on	off
Run4	off	off	on	off	on
Run5	on	on	off	off	on
Run6	on	off	on	on	off
Run7	off	on	on	on	on
Run8	on	on	on	off	off

Inferential Statistics

- Inferential statistics is used to predict whether a factor of an experiment has a significant effect.
- Inferential statistics uses the logic of null hypothesis and a test statistic.

Null Hypothesis

- Null hypothesis denies the effect of a factor in an experiment.
- When a compiler option A is applied on an application B :
 - Compiler option A is not effective to optimize application B .
- When the null hypothesis is negated, we can say the compiler option A is effective to optimize application B
- Test statistic is computed from the experimental data to evaluate likelihood of the null hypothesis.

Mann-Whitney Test

- A well known test in non-parametric inferential statistics.
- Uses ranked order from experimental data to be applied the experimental data with a few assumption
- The experimental data is divided into 2 parts, experimental group and control group according to the null hypothesis.

Mann-Whitney Test (Cont'd)

- Null hypothesis:
 - Compiler option *A* is not effective to optimize application *B*.
- Experimental group contains the compiler settings using compiler option *A*
- Control group contains the compiler settings which do not use compiler option *A*

Test Statistic z

- Test statistic z represents how many units of standard deviation are in the distance between mean value and observed value.
- Z for observed value k is computed as follows.

$$z = \frac{k - \mu}{\sigma}$$

σ : Standard deviation of experimental data
 μ : Mean of experimental data

- Z is converted to the probability that the observed value occurs.

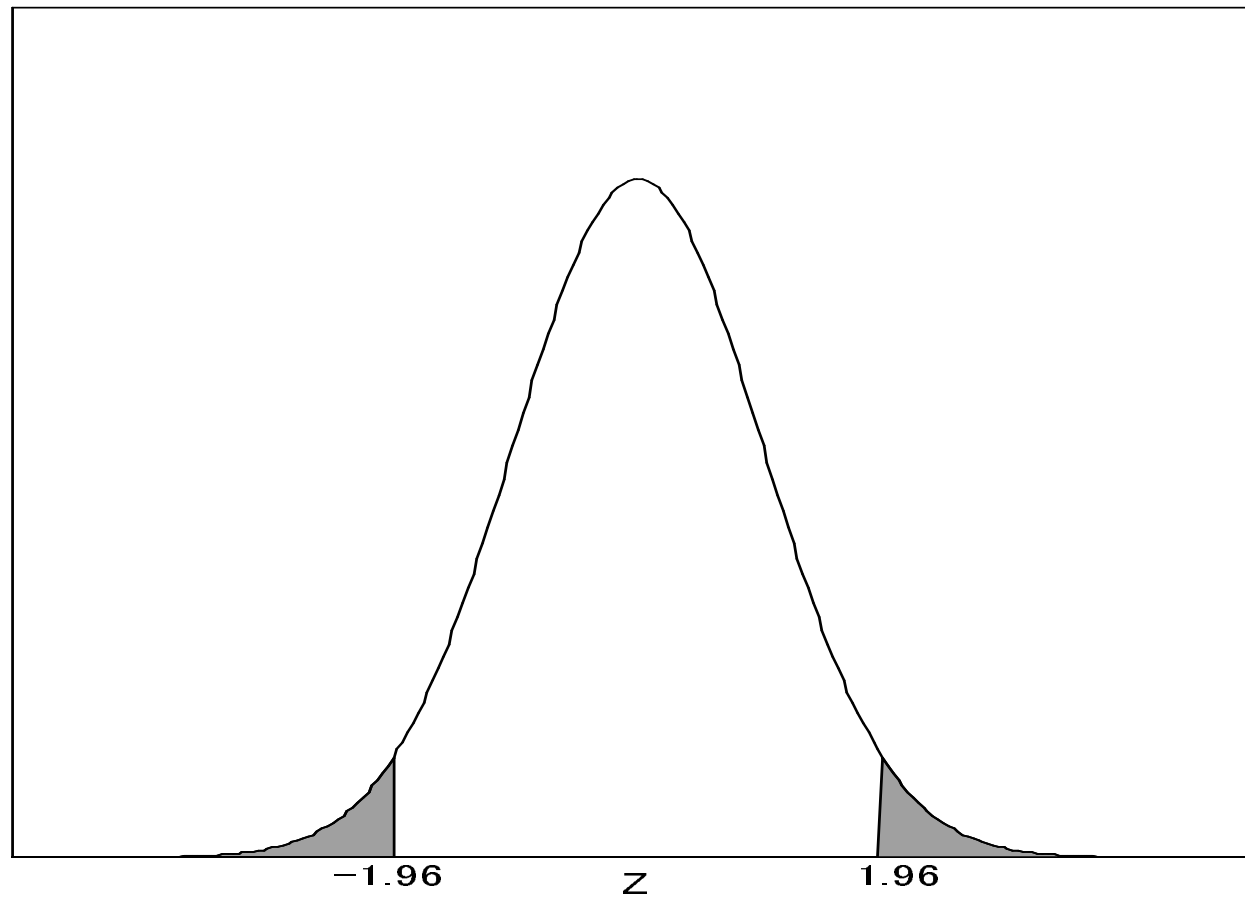
Z to Probability

- Ranked data has a normal distribution.
- When the distribution is normal, it is possible to compute the cumulative probability to observe the value within $|z|$.
- When $t=|z|$, $P(t)$ expresses the cumulative percentage of data values out of $|z|$.

- $$P(t) = (1 - 2 \cdot \int_0^t \frac{1}{\sigma \cdot \sqrt{2\pi}} e^{-\frac{1}{2}z^2} dz) \cdot 100\%$$

- In the theory of statistics, a z-value satisfying $P(t) < 5\%$ is considered significant.

Z to Probability



Example

- Find important options from the 10 compiler options

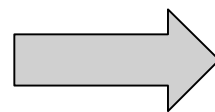
	O ₁	O ₂	O ₃	O ₄	O ₅	O ₆	O ₇	O ₈	O ₉	O ₁₀	Time	Rank
Run1	0	0	0	0	0	0	0	0	0	0	20	8
Run2	1	0	1	0	0	0	1	1	1	0	25	12
Run3	1	1	0	1	0	0	0	1	1	1	15	3
Run4	0	1	1	0	1	0	0	0	1	1	17	5
Run5	1	0	1	1	0	1	0	0	0	1	18	6
Run6	1	1	0	1	1	0	1	0	0	0	14	2
Run7	1	1	1	0	1	1	0	1	0	0	23	11
Run8	0	1	1	1	0	1	1	0	1	0	13	1
Run9	0	0	1	1	1	0	1	1	0	1	19	7
Run10	0	0	0	1	1	1	0	1	1	0	22	10
Run11	1	0	0	0	1	1	1	0	1	1	21	9
Run12	0	1	0	0	0	1	1	1	0	1	16	4

Example (Cont'd)

- Is O_2 an important option?

	- O_2 -	Time	Rank
Run1	0	20	8
Run2	0	25	12
Run3	1	15	3
Run4	1	17	5
Run5	0	18	6
Run6	1	14	2
Run7	1	23	11
Run8	1	13	1
Run9	0	19	7
Run10	0	22	10
Run11	0	21	9
Run12	1	16	4

Null hypothesis:
 O_2 is not an important compiler option



Divide into
2 groups

Experimental Group($O_2=1$)	Control Group($O_2=0$)
3 (Run3)	8 (Run1)
5 (Run4)	12 (Run2)
2 (Run6)	6 (Run5)
11 (Run7)	7 (Run9)
1 (Run8)	10 (Run10)
4 (Run12)	9 (Run11)

Compute test statistic z

Example

- Compute test statistic z for O_2

Experimental Group ($O_2=1$)	Control Group2($O_2=0$)
3 (Run3)	8 (Run1)
5 (Run4)	12 (Run2)
2 (Run6)	6 (Run5)
11 (Run7)	7 (Run9)
1 (Run8)	10 (Run10)
4 (Run12)	9 (Run11)

$$\sigma = \sqrt{\frac{6 \cdot 6 \cdot (6 + 6 + 1)}{12}} = \sqrt{39}$$

$$\mu = \frac{6(6 + 6 + 1)}{2} = 39$$

$$k = 3 + 5 + 2 + 11 + 1 + 4 = 26$$

$$z = \frac{k - \mu}{\sigma} = -2.08$$

$$P|-2.08| < 5\%$$

Iterative Algorithm

- Choose an orthogonal array A with as many columns as there are options.
- Repeat
 - Compile the application with each row from A as compiler setting and execute the optimized application.
 - Compute test statistic z for each compiler option
 - If the test statistic meets $P(|z|) < 5\%$
 - If z is negative then the option is turned on.
 - If z is positive then the option is turned off.
 - Remove the compiler options that have been selected from the factor list and drop the same number of columns from A .
- Until
 - All options are set, or
 - No option with a significant effect is detected anymore, or
 - The experimental data has not enough variation (low standard deviation) to apply the Mann-Whitney test meaningfully.
- Choose the compiler setting which has the best execution time in the last experiment.

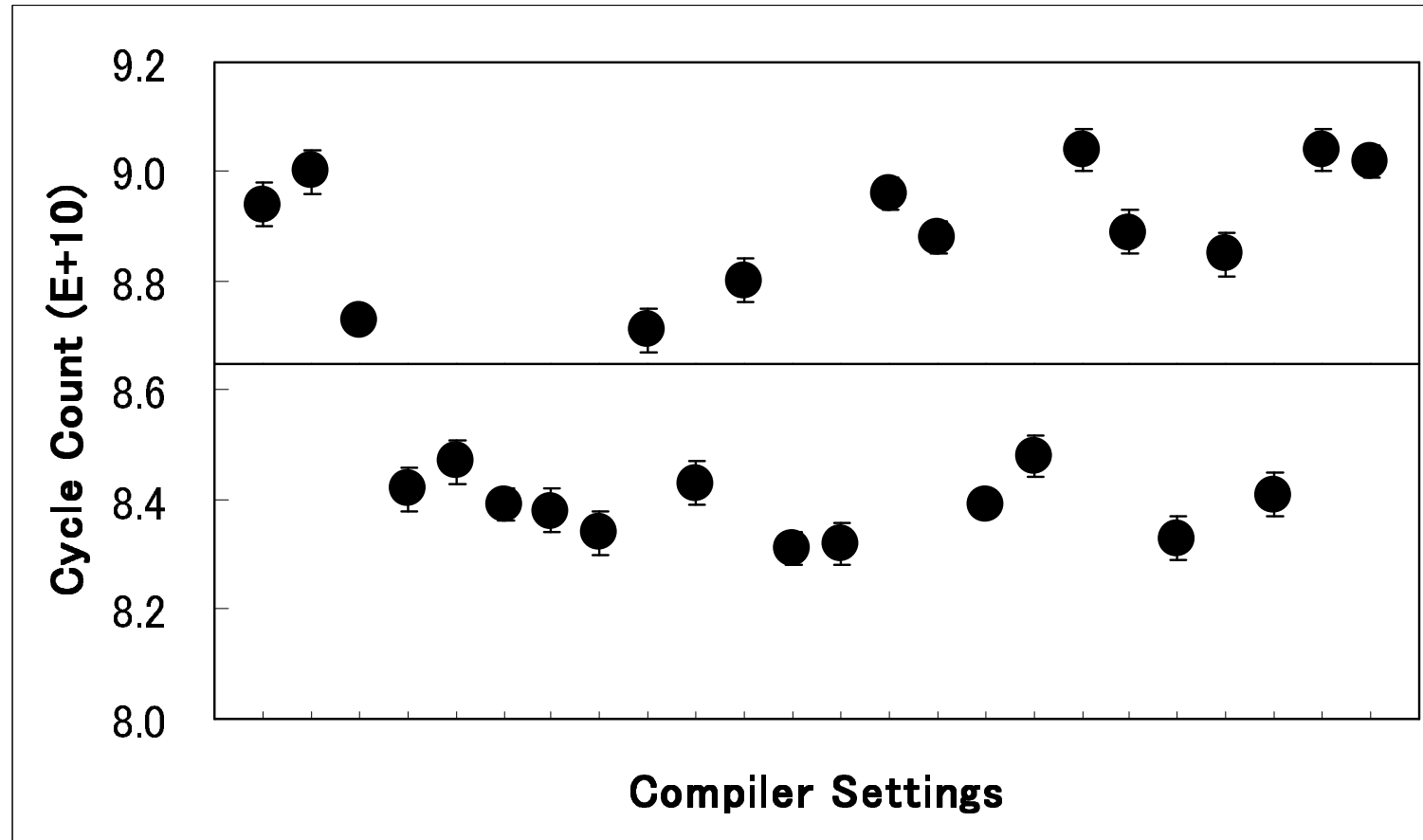
Application to GCC

- Compiler version: 3.3.1
- Number of options: 45 options
 - We arrange them into 23 factors according to their dependency.
- Architecture: Pentium 4 at 2.8GHz
- Applications: 10 programs from the SPEC 2000 benchmark suite with the train data set.
- Measurement: Clockticks observed by using VTune (a tool to analyze performance of applications on Intel architectures)

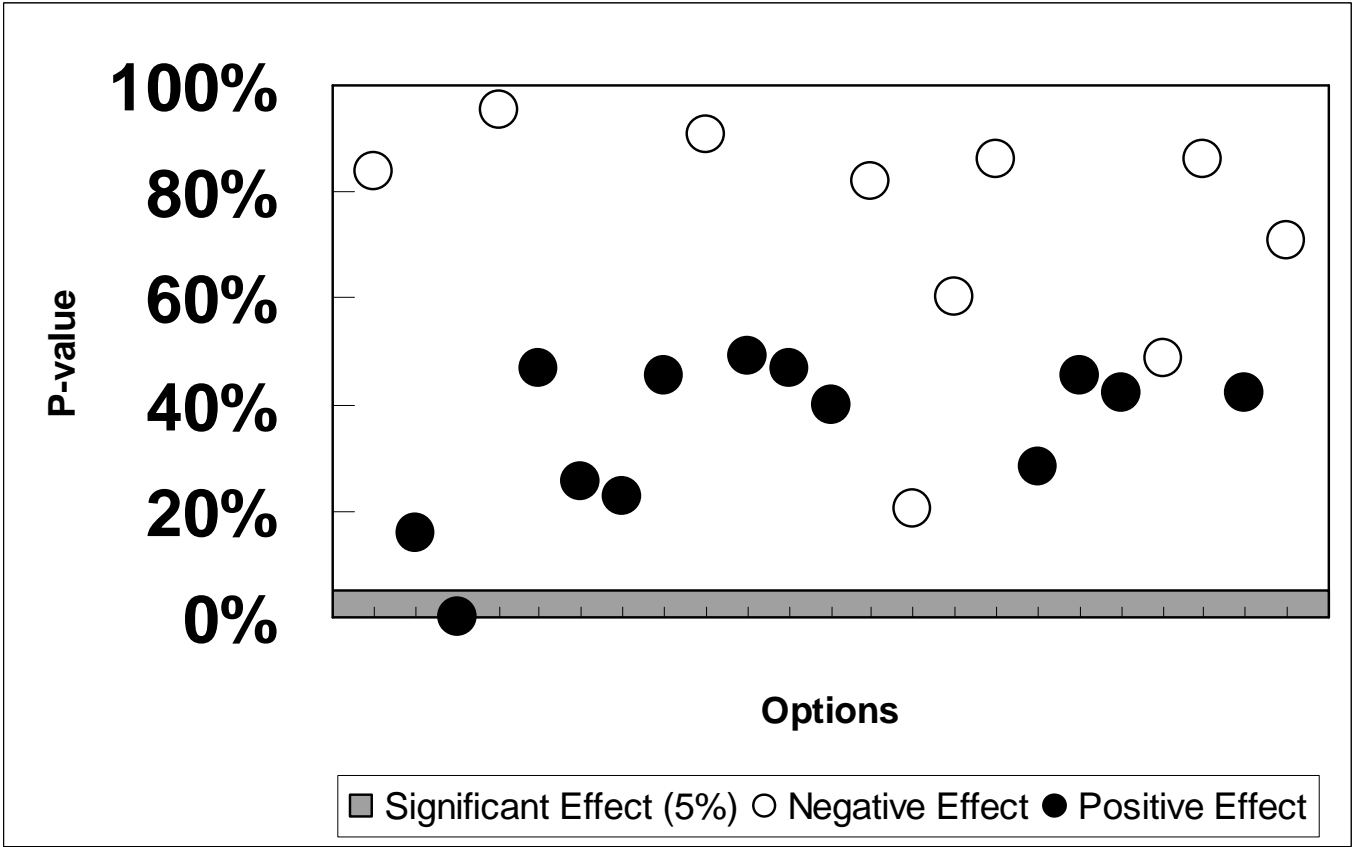
Case Study (181.mcf)

- We illustrate the performance of our algorithm with the results per iteration for 181.mcf.
- 23 options to be configured
- A 24x23 orthogonal array is chosen to design the experiment
- 181.mcf is compiled with 24 settings, and executed.

24 data points (1st Iteration)

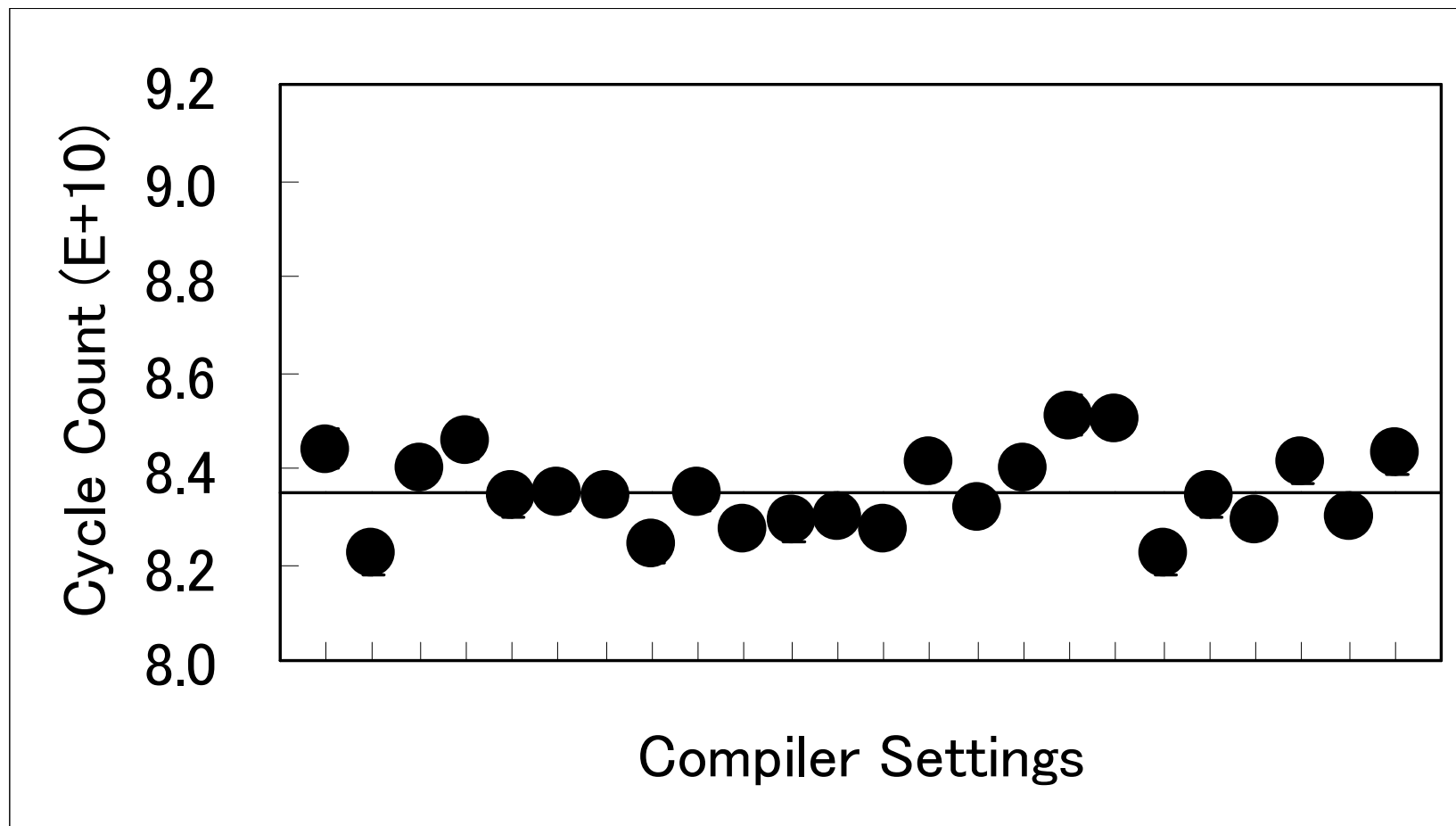


1st Iteration (P-value)

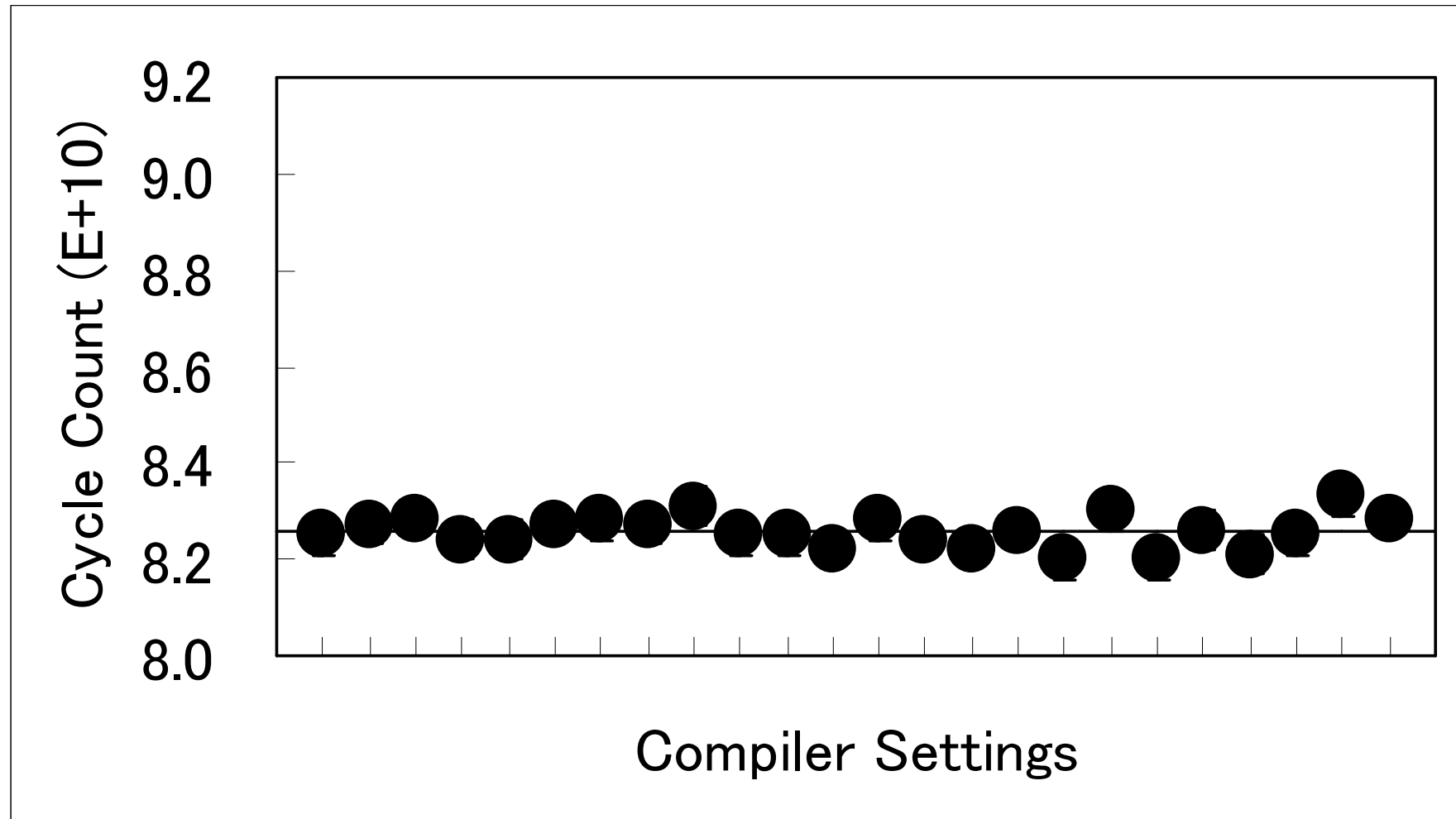


○ ₁	○ ₂	○ ₃	○ ₄	○ ₅	○ ₆	○ ₇	○ ₈	○ ₉	○ ₁₀	○ ₁₁	○ ₁₂	○ ₁₃	○ ₁₄	○ ₁₅	○ ₁₆	○ ₁₇	○ ₁₈	○ ₁₉	○ ₂₀	○ ₂₁	○ ₂₂	○ ₂₃	
								1															

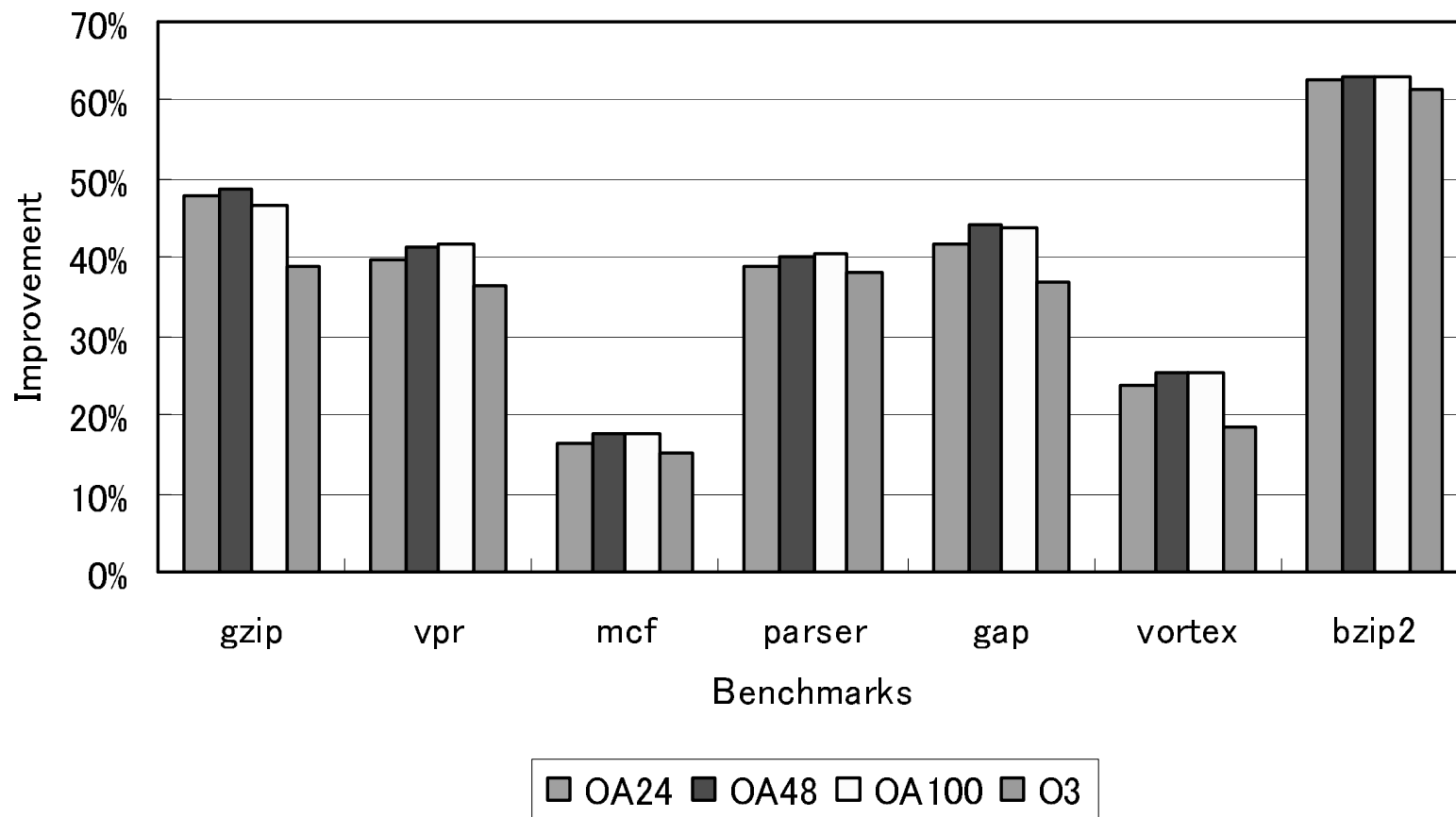
24 data points (2nd Iteration)



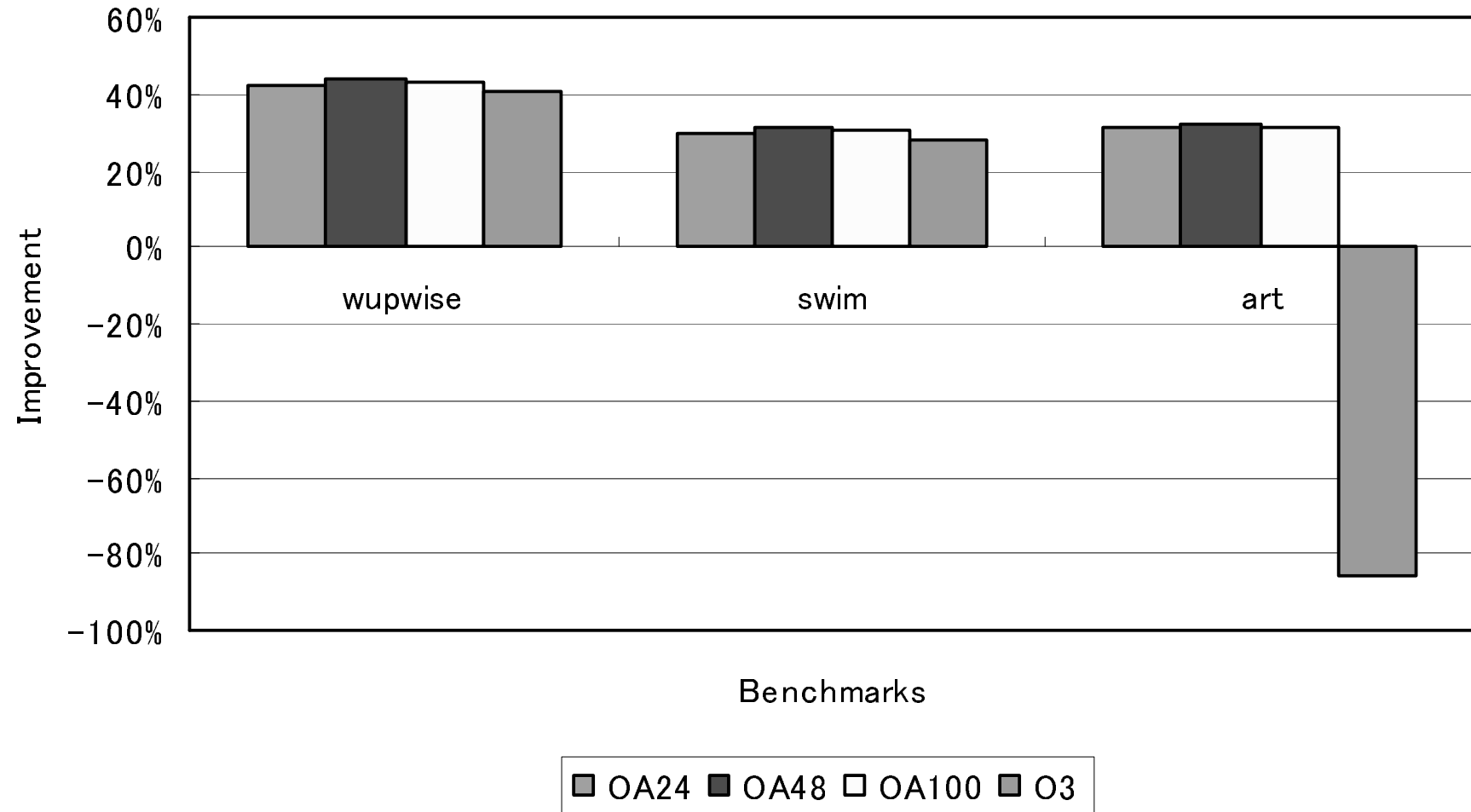
24 data points (3rd Iteration)



Results (SPEC2000 CINT)



Result (SPEC2000 CFP)



Summary

- We have introduced the concept of Design of Experiments to determine compiler settings.
- We have employed the Mann-Whitney test to examine the significance of the effect of compiler options.
- Select important compiler options iteratively and obtain one optimization setting for each application.
- All settings which we obtained for each benchmark outperform O3 which is the standard setting of gcc.

Sample Size

- Power analysis provides the adequate sample size according to the variance of sample data and strength of reliability.
- Sample size is given by using Z-values of the threshold for the Mann-Whitney test (Z_{α}) and the reliability of the test (Z_{β})

$$M = \left\lceil 5 \cdot (Z_{\alpha} + Z_{\beta})^2 \right\rceil$$

- α is 5%, and for β we use 60%, 85%, and 99.5% which are OA size 24, 48, and 100 respectively