Knapsack Problem Results

Group Members

Andreas, Andrew, Jaden

Problem Description

The knapsack problem is a classic optimization problem where you have a knapsack with a limited weight capacity, and a set of items, each with a weight and a value. The goal is to select the items to put in the knapsack to maximize the total value without exceeding the weight limit.

Knobs to Twiddle

In this experiment, we focused on adjusting the maximum number of generations to determine when we would start to experience diminishing returns in our scores.

Unchanging System Parts

We kept the following aspects of the system constant:

- Scoring system: We used the cliff-scorer system provided in knapsack-ga.
- Mutation rate: We used the default mutation rate provided in knapsack-ga.

Framework

We used the knapsack-ga framework for our experiment.

Experimental Design

We performed a series of runs across several different scopes to identify the point at which

Values for Knobs

We used the following sets of values for the maximum number of generations:

- 100 to 1000 with an increment interval of 100
- 10 to 250 with an increment interval of 20
- 5 to 100 with an increment interval of 5

Number of Runs

We performed 50 runs for each set of parameter values.

Data Collection

For each run, we collected the following data:

- Run number
- Generation count
- Generation found
- Best score
- Run time
- Incremental

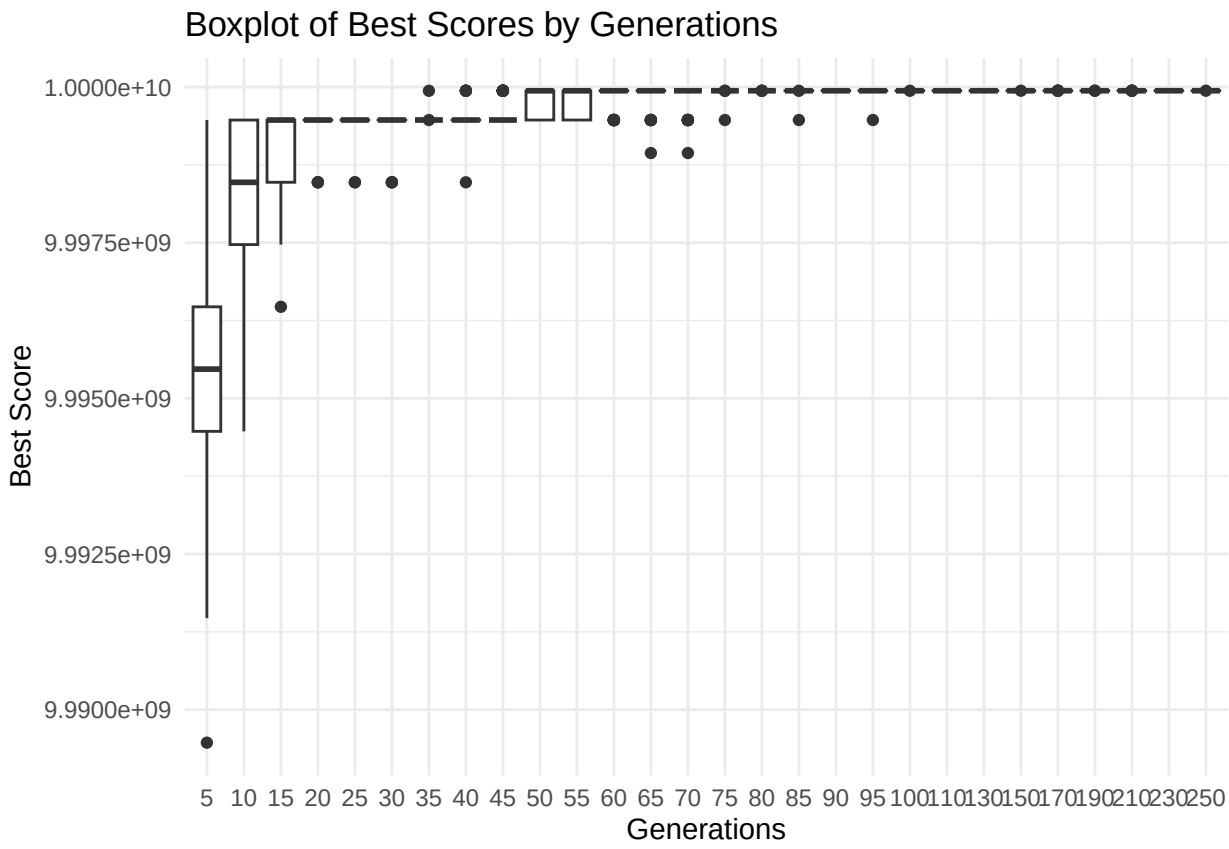
Results

Summary

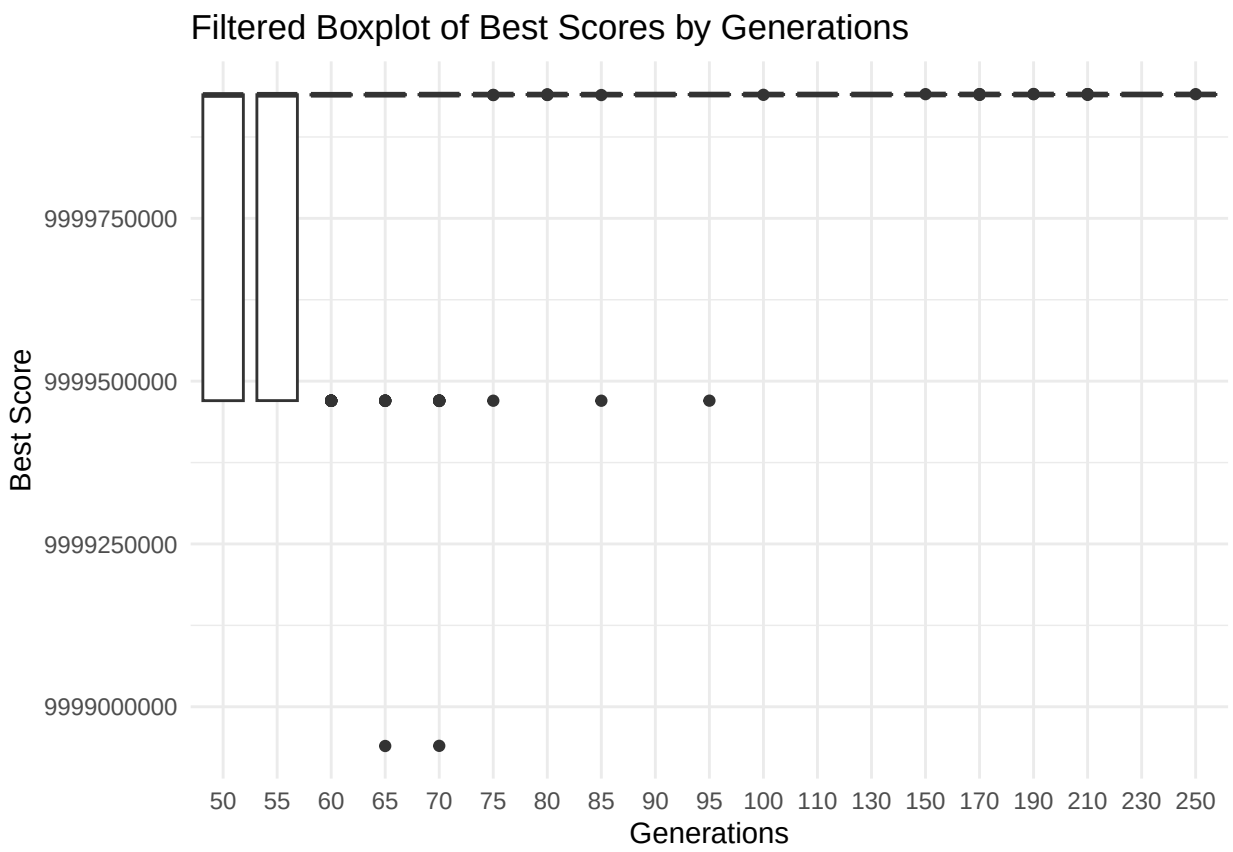
Our results suggest that increasing the maximum number of generations initially leads to improved scores, but beyond a certain point, the gains become statistically insignificant.

Tables and Graphs

```
library(ggplot2)
library(reshape2)

data <- read.csv("knapsack_results_250base_inc20.csv")
filtered_data <- subset(data, !(generations %in% c(5, 10, 15, 20, 25, 30, 35, 40, 45)))

# boxplot
ggplot(data, aes(x = as.factor(generations), y = best_score)) +
  geom_boxplot() +
  labs(title = "Boxplot of Best Scores by Generations", x = "Generations", y = "Best Score") +
  theme_minimal()
```




```
ggplot(filtered_data, aes(x = as.factor(generations), y = best_score)) +
  geom_boxplot() +
  labs(title = "Filtered Boxplot of Best Scores by Generations", x = "Generations", y = "Best Score") +
  theme_minimal()
```

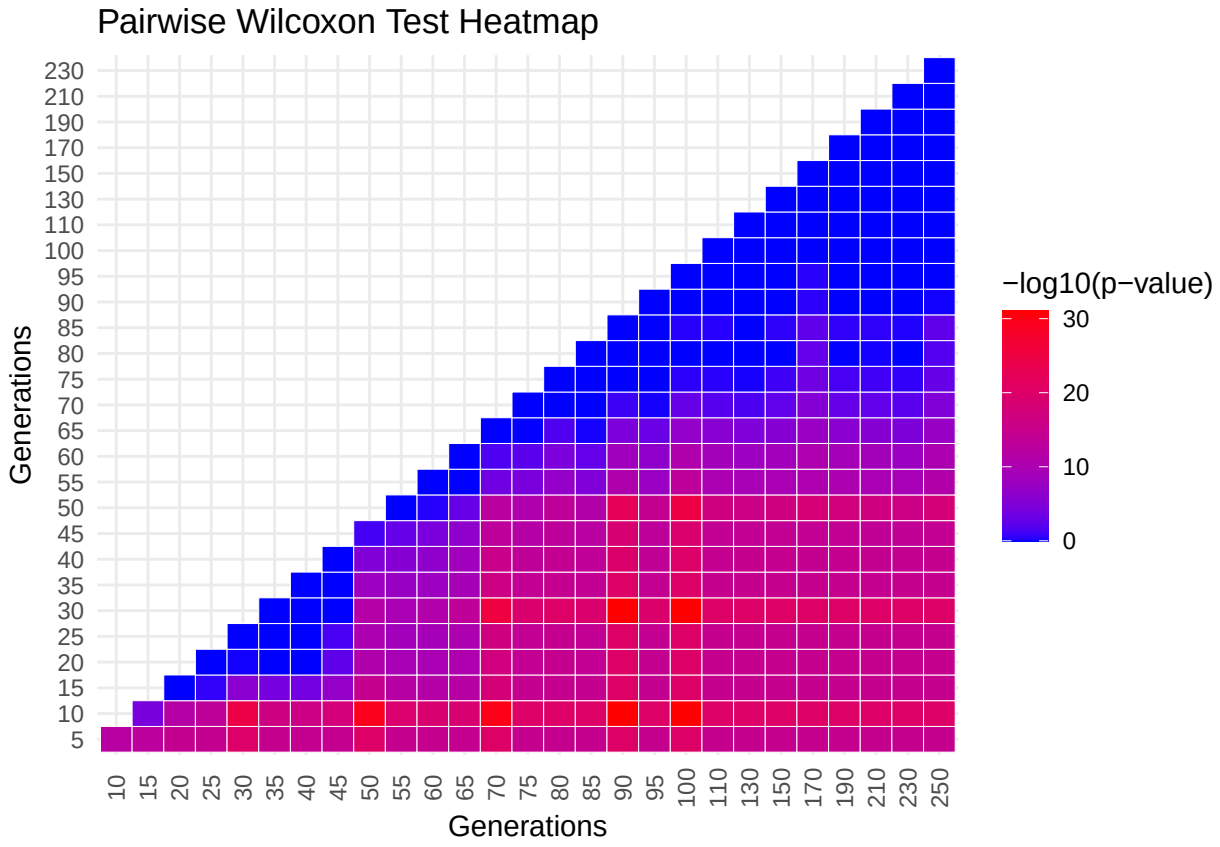


Statistical Tests

We used the Pairwise Wilcoxon test to determine the point at which the number of max generations stops returning scores with a statistically significant difference.

```r
# p values
library(knitr)
library(kableExtra)

wilcox_results <- pairwise.wilcox.test(data$best_score,
                                       as.factor(data$generations),
                                       p.adjust.method = "bonferroni")

# Extract Wilcoxon test results as a matrix
matrix_results <- wilcox_results$p.value

# Round the matrix_results to the nearest hundredth
rounded_matrix <- round(matrix_results, 2)

# Replace NA values with "-" for better readability
matrix_results[is.na(matrix_results)] <- "-"

# Convert Wilcoxon test results into a data frame
wilcox_matrix <- as.data.frame(as.table(wilcox_results$p.value))

# Rename columns for clarity
colnames(wilcox_matrix) <- c("Gen1", "Gen2", "P_Value")
```

```r
# Remove NA values
wilcox_matrix <- na.omit(wilcox_matrix)

# Convert generations to numeric
wilcox_matrix$Gen1 <- as.numeric(as.character(wilcox_matrix$Gen1))
wilcox_matrix$Gen2 <- as.numeric(as.character(wilcox_matrix$Gen2))

# Create the heatmap
ggplot(wilcox_matrix, aes(
  x = factor(Gen1, levels = unique(wilcox_matrix$Gen1)),
  y = factor(Gen2, levels = unique(wilcox_matrix$Gen2)),
  fill = -log10(P_Value)
)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "blue",
                      high = "red",
                      name = "-log10(p-value)") +
  labs(title = "Pairwise Wilcoxon Test Heatmap", x = "Generations", y = "Generations") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```



Discussion

Based on the boxplots, it initially appeared that 45 generations would be the optimal number due to its median score aligning with higher generation counts, but It had a much lower range compared to higher values. We see this range fall in line with higher numbers of generations around 60, but with drastically lower

outliers. We don't see the outliers fall in line until 100 max generations.

We verified our findings using a Pairwise Wilcoxen Test, and we discovered that 100 is indeed the point at which any further numbers of max generations provide no statistically significant difference, with any minor differences coming down to the random nature of our tests.

Conclusion

In this experiment, we explored the relationship between the maximum number of generations and the best score achieved in the knapsack problem. We learned that increasing the maximum number of generations can improve the solution, but there is a point of diminishing returns. Based on our results, using a maximum of 100 generations appears to be sufficient for achieving near-optimal solutions.

Apendix

```
print(wilcox_results)
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum test with continuity correction
##
## data:  data$best_score and as.factor(data$generations)
##
##     5       10      15      20      25      30      35      40      45
## 10  7.8e-13 -       -       -       -       -       -       -       -
## 15  1.7e-13 5.3e-05 -       -       -       -       -       -       -
## 20  4.9e-15 2.9e-12 1.00000 -       -       -       -       -       -
## 25  3.4e-15 5.2e-14 0.18610 1.00000 -       -       -       -       -
## 30  < 2e-16 < 2e-16 1.5e-06 0.74207 1.00000 -       -       -       -
## 35  2.7e-15 < 2e-16 5.9e-05 1.00000 1.00000 1.00000 -       -       -
## 40  3.0e-15 < 2e-16 0.00012 1.00000 1.00000 1.00000 1.00000 -       -
## 45  3.0e-15 < 2e-16 8.0e-08 0.00295 0.03038 1.00000 1.00000 1.00000 -
## 50  < 2e-16 < 2e-16 1.6e-15 1.6e-11 7.2e-11 3.2e-12 7.5e-09 9.0e-06 0.06135
## 55  2.7e-15 < 2e-16 1.4e-12 3.2e-10 1.9e-09 1.4e-10 2.9e-08 2.5e-06 0.00121
## 60  2.7e-15 < 2e-16 3.9e-12 2.0e-10 7.5e-10 8.8e-12 7.3e-09 3.9e-07 4.3e-05
## 65  3.0e-15 < 2e-16 1.7e-12 3.7e-11 4.9e-11 4.7e-14 4.6e-10 2.1e-09 3.9e-07
## 70  < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 5.4e-16 7.3e-14
## 75  2.7e-15 < 2e-16 3.8e-15 7.4e-15 9.3e-15 < 2e-16 1.3e-14 4.5e-14 2.1e-12
## 80  2.7e-15 < 2e-16 2.7e-15 2.7e-15 2.7e-15 < 2e-16 2.8e-15 5.5e-15 7.9e-14
## 85  2.7e-15 < 2e-16 3.8e-15 7.8e-15 9.3e-15 < 2e-16 1.2e-14 2.9e-14 5.5e-13
## 90  < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16
## 95  2.7e-15 < 2e-16 3.4e-15 4.1e-15 5.5e-15 < 2e-16 5.8e-15 1.5e-14 8.3e-14
## 100 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16 < 2e-16
## 110 2.7e-15 < 2e-16 2.7e-15 2.7e-15 2.7e-15 < 2e-16 2.7e-15 3.0e-15 1.2e-14
## 130 2.7e-15 < 2e-16 2.7e-15 2.7e-15 2.7e-15 < 2e-16 2.7e-15 2.7e-15 1.2e-14
## 150 2.7e-15 < 2e-16 2.7e-15 2.7e-15 2.7e-15 < 2e-16 2.7e-15 3.0e-15 1.0e-14
## 170 2.7e-15 < 2e-16 2.7e-15 2.7e-15 2.7e-15 < 2e-16 2.7e-15 3.6e-15 7.8e-15
## 190 2.7e-15 < 2e-16 2.7e-15 2.7e-15 2.7e-15 < 2e-16 2.7e-15 2.8e-15 9.6e-15
## 210 2.7e-15 < 2e-16 2.7e-15 2.7e-15 2.7e-15 < 2e-16 2.8e-15 3.4e-15 1.5e-14
## 230 2.7e-15 < 2e-16 2.7e-15 2.7e-15 2.7e-15 < 2e-16 2.7e-15 3.4e-15 1.4e-14
## 250 2.7e-15 < 2e-16 2.7e-15 2.7e-15 2.7e-15 < 2e-16 2.7e-15 2.7e-15 6.6e-15
##     50      55      60      65      70      75      80      85      90
## 10  -       -       -       -       -       -       -       -       -
## 15  -       -       -       -       -       -       -       -       -
## 20  -       -       -       -       -       -       -       -       -
```

```
## 25  -       -       -       -       -       -       -       -       -
## 30  -       -       -       -       -       -       -       -       -
## 35  -       -       -       -       -       -       -       -       -
## 40  -       -       -       -       -       -       -       -       -
## 45  -       -       -       -       -       -       -       -       -
## 50  -       -       -       -       -       -       -       -       -
## 55  1.00000 -       -       -       -       -       -       -       -
## 60  0.40205 1.00000 -       -       -       -       -       -       -
## 65  0.00081 1.00000 1.00000 -       -       -       -       -       -
## 70  3.6e-13 0.00022 0.02157 1.00000 -       -       -       -       -
## 75  2.8e-11 4.1e-05 0.00548 0.95686 1.00000 -       -       -       -
## 80  5.2e-14 1.1e-07 2.3e-05 0.01878 1.00000 1.00000 -       -       -
## 85  6.5e-12 1.3e-05 0.00169 0.67674 1.00000 1.00000 1.00000 -       -
## 90  < 2e-16 1.7e-11 3.3e-09 1.8e-05 0.09525 1.00000 1.00000 1.00000 -
## 95  4.6e-15 2.0e-08 3.4e-07 0.00071 0.68908 1.00000 1.00000 1.00000 1.00000
## 100 < 2e-16 7.0e-14 1.6e-11 1.3e-07 0.00105 0.31562 1.00000 0.37992 1.00000
## 110 < 2e-16 1.2e-10 1.3e-09 1.9e-06 0.00915 0.33676 1.00000 0.38079 1.00000
## 130 < 2e-16 2.7e-10 8.4e-09 1.2e-05 0.02511 0.66085 1.00000 1.00000 1.00000
## 150 < 2e-16 1.4e-10 1.7e-09 3.2e-06 0.00210 0.07974 1.00000 0.26911 1.00000
## 170 < 2e-16 3.0e-11 3.3e-11 2.7e-08 4.0e-06 0.00021 0.00146 0.00247 0.31112
## 190 < 2e-16 4.9e-11 6.5e-10 1.4e-06 0.00128 0.03764 0.95692 0.21704 1.00000
## 210 < 2e-16 1.8e-10 1.2e-09 3.0e-06 0.00191 0.06863 0.70086 0.25589 1.00000
## 230 < 2e-16 4.2e-10 1.2e-08 1.5e-05 0.00535 0.22837 1.00000 0.50321 1.00000
## 250 < 2e-16 6.5e-12 5.7e-11 3.1e-08 1.1e-05 0.00103 0.01130 0.00239 0.76229
##     95      100     110     130     150     170     190     210     230
## 10  -       -       -       -       -       -       -       -       -
## 15  -       -       -       -       -       -       -       -       -
## 20  -       -       -       -       -       -       -       -       -
## 25  -       -       -       -       -       -       -       -       -
## 30  -       -       -       -       -       -       -       -       -
## 35  -       -       -       -       -       -       -       -       -
## 40  -       -       -       -       -       -       -       -       -
## 45  -       -       -       -       -       -       -       -       -
## 50  -       -       -       -       -       -       -       -       -
## 55  -       -       -       -       -       -       -       -       -
## 60  -       -       -       -       -       -       -       -       -
## 65  -       -       -       -       -       -       -       -       -
## 70  -       -       -       -       -       -       -       -       -
## 75  -       -       -       -       -       -       -       -       -
## 80  -       -       -       -       -       -       -       -       -
## 85  -       -       -       -       -       -       -       -       -
## 90  -       -       -       -       -       -       -       -       -
## 95  -       -       -       -       -       -       -       -       -
## 100 1.00000 -       -       -       -       -       -       -       -
## 110 1.00000 1.00000 -       -       -       -       -       -       -
## 130 1.00000 1.00000 1.00000 -       -       -       -       -       -
## 150 1.00000 1.00000 1.00000 1.00000 -       -       -       -       -
## 170 0.34096 1.00000 1.00000 1.00000 1.00000 -       -       -       -
## 190 1.00000 1.00000 1.00000 1.00000 1.00000 1.00000 -       -       -
## 210 1.00000 1.00000 1.00000 1.00000 1.00000 1.00000 1.00000 -       -
## 230 1.00000 1.00000 1.00000 1.00000 1.00000 1.00000 1.00000 1.00000 -
## 250 1.00000 1.00000 1.00000 1.00000 1.00000 1.00000 1.00000 1.00000 1.00000
##
## P value adjustment method: bonferroni
```