Examining the Stability Versus Efficiency of Algorithms Related to the Stable Marriage Problem

By Maddison Barnwell, Cydney Miller, Esraa Marfing, David Oluwadara Owotomo

Overview

1: Problem and Algorithms Introduction

- 1.1 Stable Marriage Problem
- 1.2 Algorithms We Will Explore
- 1.3 Objective/Research Question

2: Description of Experiment

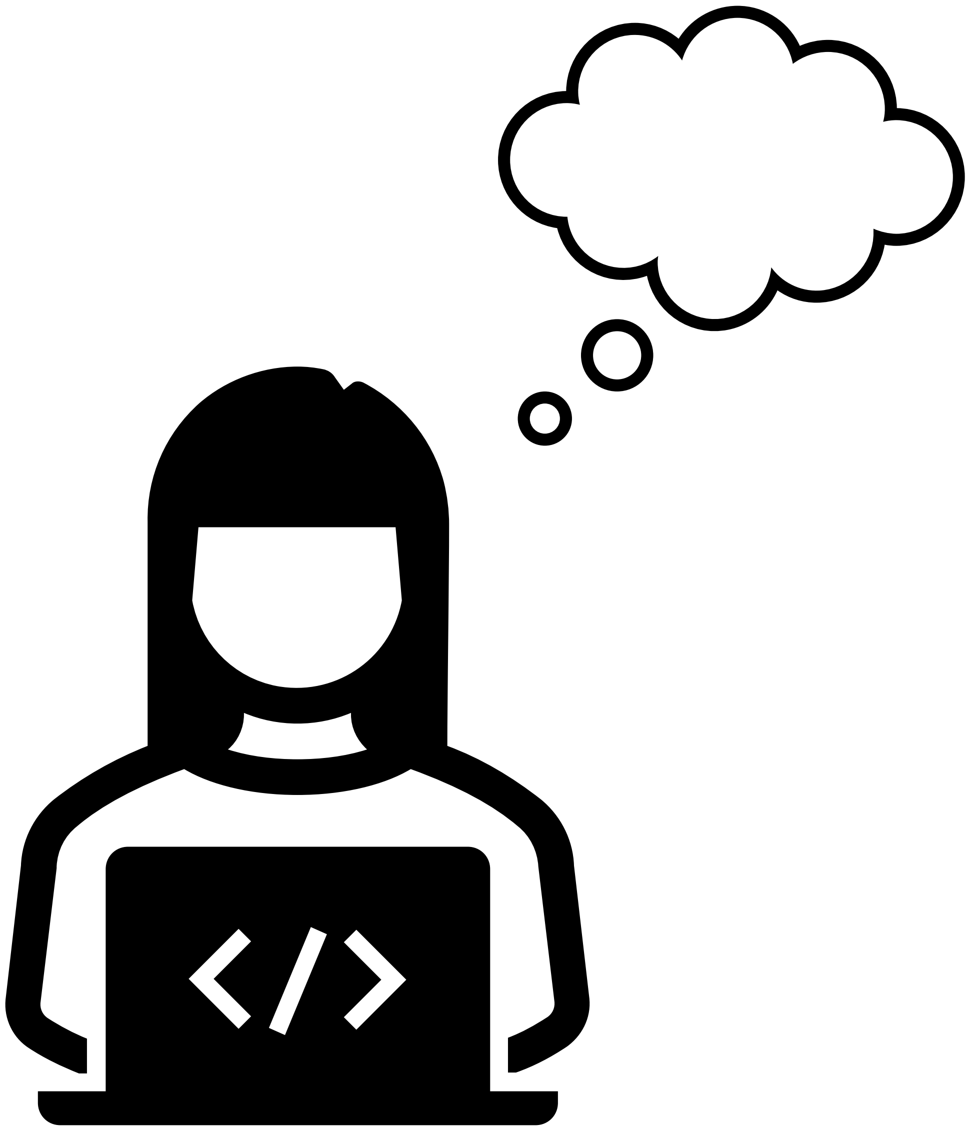
- 2.1 Implementation Details
- 2.2 Input Description
- 2.3 Experimentation Setup
- 2.4 Hardware Characteristics

3: Results and Interpretation

- 3.1 Results
- 3.2 Interpretation

4: Obstacles Encountered

5: Works Cited



1.1 Stable Marriage Problem

- The Stable Marriage Problem pairs N men and N women who have ranked all members of the opposite sex by preference.
- The aim is to create marriages where no two people prefer each other over their current partners.

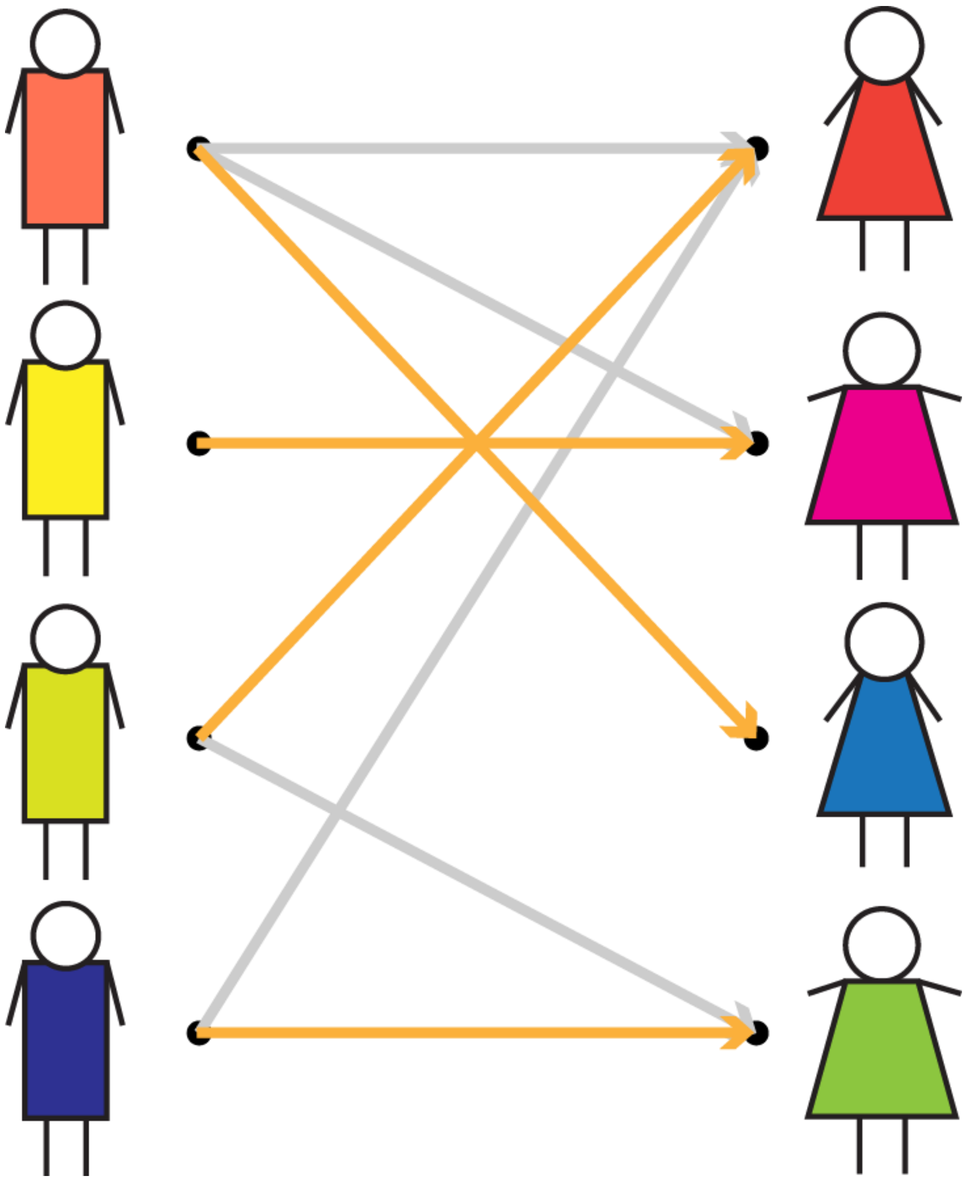


Image courtesy of [7]

1.2 Algorithms We Will Explore

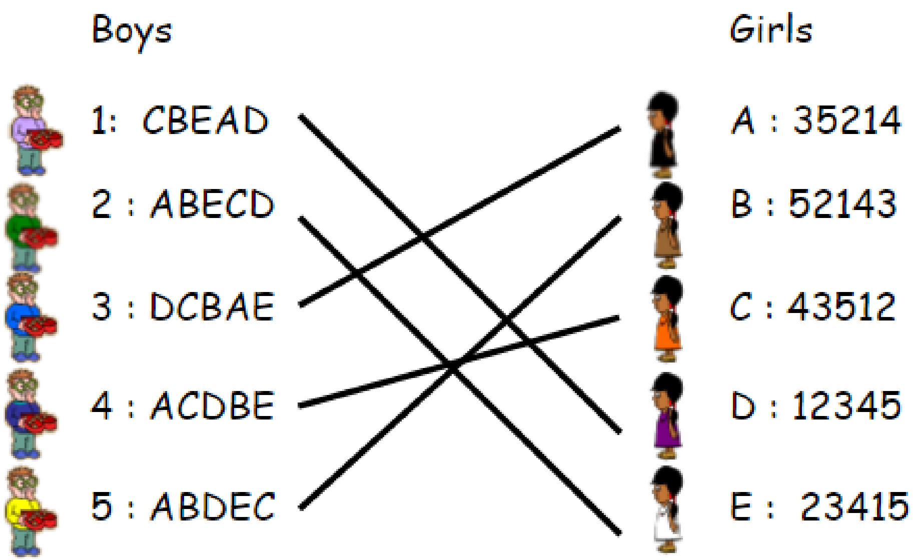
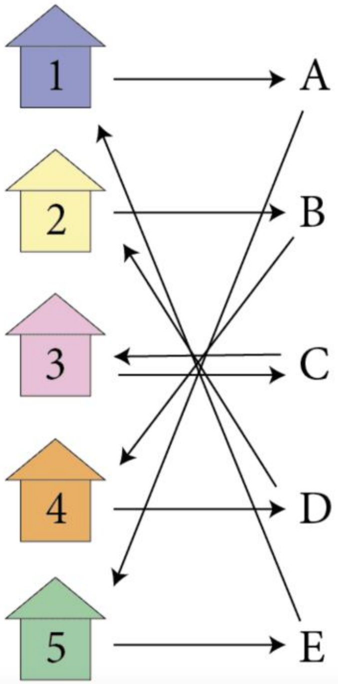


Image courtesy of [9]

- **The Gale–Shapley algorithm** is the standard method for producing a stable matching between two groups. *(Greedy, iterative improvement)*
- **Irving's Algorithm** solves the Stable Roommates Problem, a variant of the Stable Marriage Problem that works with a single group instead of two. *(Iterative reduction)*
- **Randomized Serial Dictatorship (RSD)** is a method for one-sided matching, often explained using the example of assigning agents to houses. *(Randomized greedy, no iterative refinement)*

Image courtesy of [4]



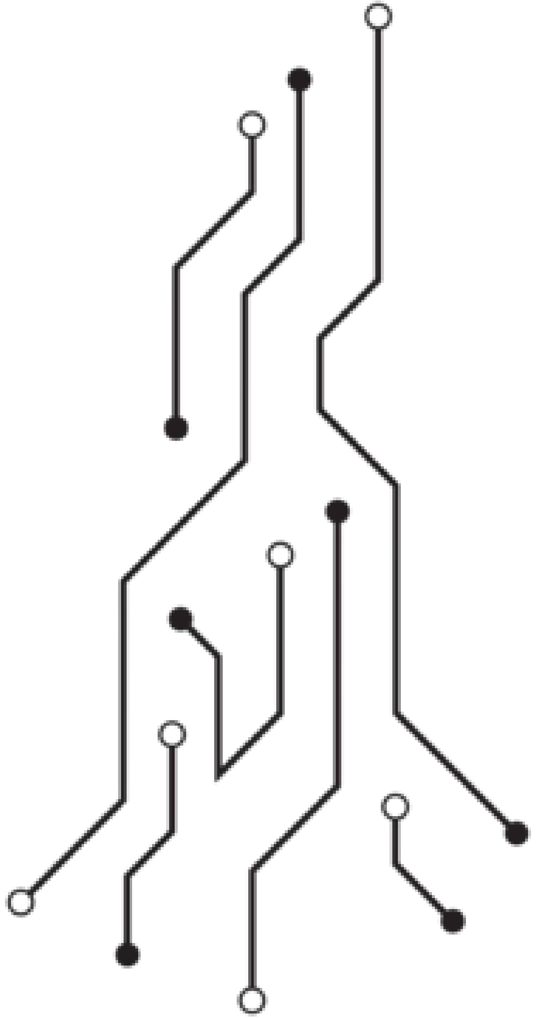
We acknowledge that Irving's Algorithm and RSD would not be used in practice to solve the Stable Marriage Problem, we are only using them for experimentational purposes.

1.3 Objective/Research Question

Compare the stability guarantees and asymptotic time complexities of Gale–Shapley, Irving's algorithm, and Randomized Serial Dictatorship, and explain the trade-offs between deterministic stability and computational efficiency.

2 Description of Experiment

- 2.1 Implementation Details
- 2.2 Input Description
- 2. 3 Experimentation Setup
- 2.4 Hardware Characteristics

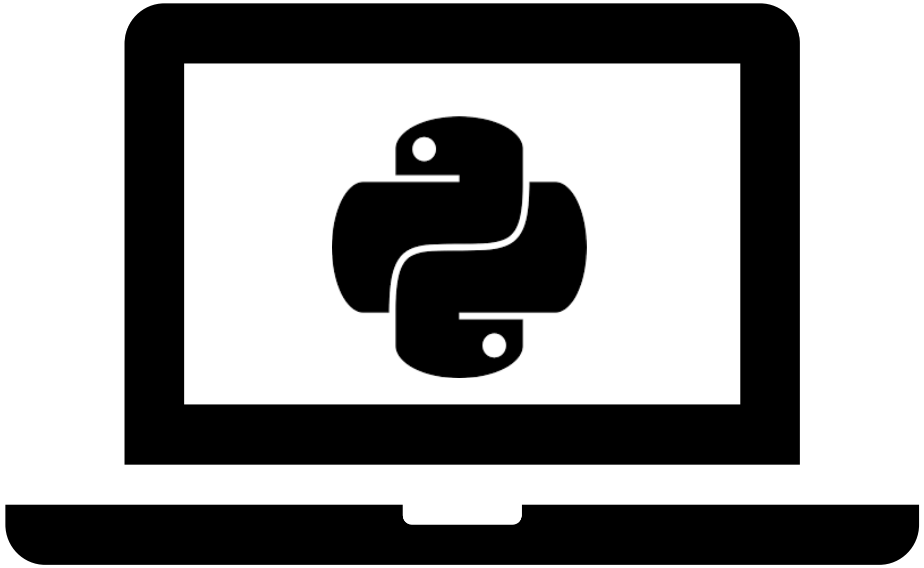


2.1 Implementation Details

- Environment: Designed and executed using the free version of Google Colab

- Language: Python 3.12.12 (Google Colab Standard)

- Libraries used throughout various functions are as follows:



2.2 Input Description

There were 6 distinct input types for this experiment:

Hand-Created:

- Preference Lists for GS and RSD-based
- Preference Lists for IA

Computer-Generated:

- Random: GS and RSD-based
- Random: IA
- Worst-Case: GS and RSD-based
- Worst-Case: IA

Hand Created Inputs:

Gale Shapley/RSD-based

```
men_prefs = {
    'm1': ['w1', 'w2', 'w3'],
    'm2': ['w1', 'w3', 'w2'],
    'm3': ['w2', 'w3', 'w1']
}
```

```
women_prefs = {
    'w1': ['m2', 'm1', 'm3'],
    'w2': ['m1', 'm3', 'm2'],
    'w3': ['m3', 'm2', 'm1']
}
```

Irving's Algorithm

```
stable_preferences = {
    'A': ['B', 'C', 'D'],
    'B': ['A', 'C', 'D'],
    'C': ['D', 'A', 'B'],
    'D': ['C', 'A', 'B']
}
```

2.2 Input Description Cont.

Why?

- **Gale–Shapley and RSD** use **two distinct groups**, each ranking members of the opposite group
- **Irving's Algorithm** works with **one group**, where each person ranks all others in the same set
- Because of these different input structures, we had to **generate custom inputs for Irving's Algorithm** during each testing step of our experiment

Hand Created Inputs:

Gale Shapley/RSD-based

```
men_prefs = {
    'm1': ['w1', 'w2', 'w3'],
    'm2': ['w1', 'w3', 'w2'],
    'm3': ['w2', 'w3', 'w1']
}
```

```
women_prefs = {
    'w1': ['m2', 'm1', 'm3'],
    'w2': ['m1', 'm3', 'm2'],
    'w3': ['m3', 'm2', 'm1']
}
```

Irving's Algorithm

```
stable_preferences = {
    'A': ['B', 'C', 'D'],
    'B': ['A', 'C', 'D'],
    'C': ['D', 'A', 'B'],
    'D': ['C', 'A', 'B']
}
```

2.3 Experimentation Setup

Data Collection Procedure

- Prompt user for max input
- Generate input sizes (10 steps up to max)
- For each input, collect time for each algorithm to perform their function recorded in microseconds

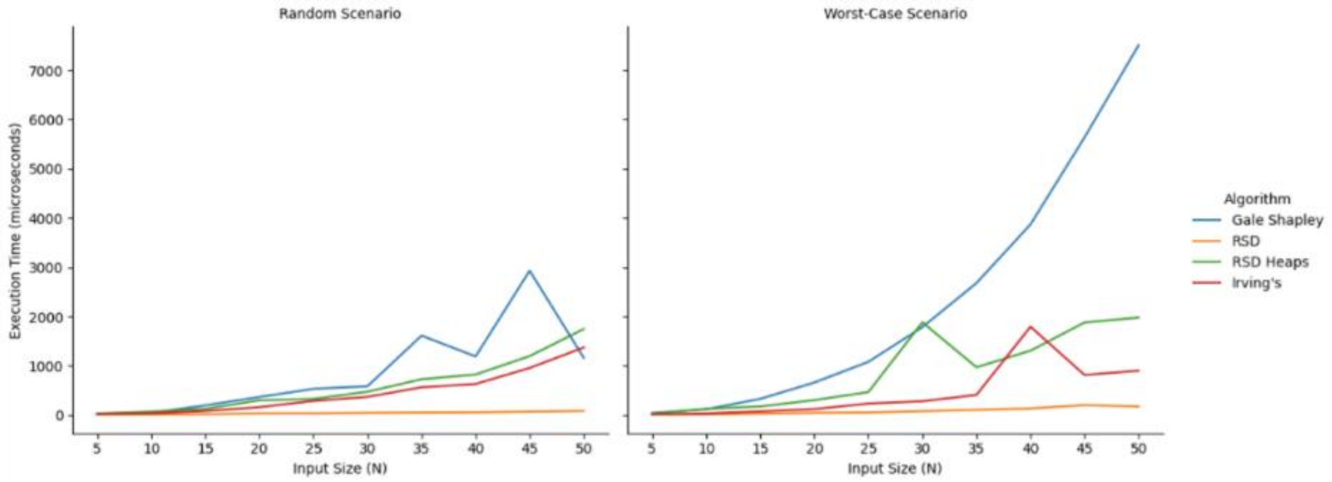
Data Structuring

- Collected data is compiled into a Pandas DataFrame which is used to graph results

Table 1
Trial One Input Size 50

N	Scenario	Algorithm	Time (Microseconds)
50	Random	Gale Shapley	1166.437
50	Random	RSD	89.504
50	Random	RSD Heaps	1751.769
50	Random	Irving's	1375.502
50	Worst-Case	Gale Shapley	7510.798
50	Worst-Case	RSD	179.319
50	Worst-Case	RSD Heaps	1982.512
50	Worst-Case	Irving's	904.758

Figure 1
Trial One Input Size 50

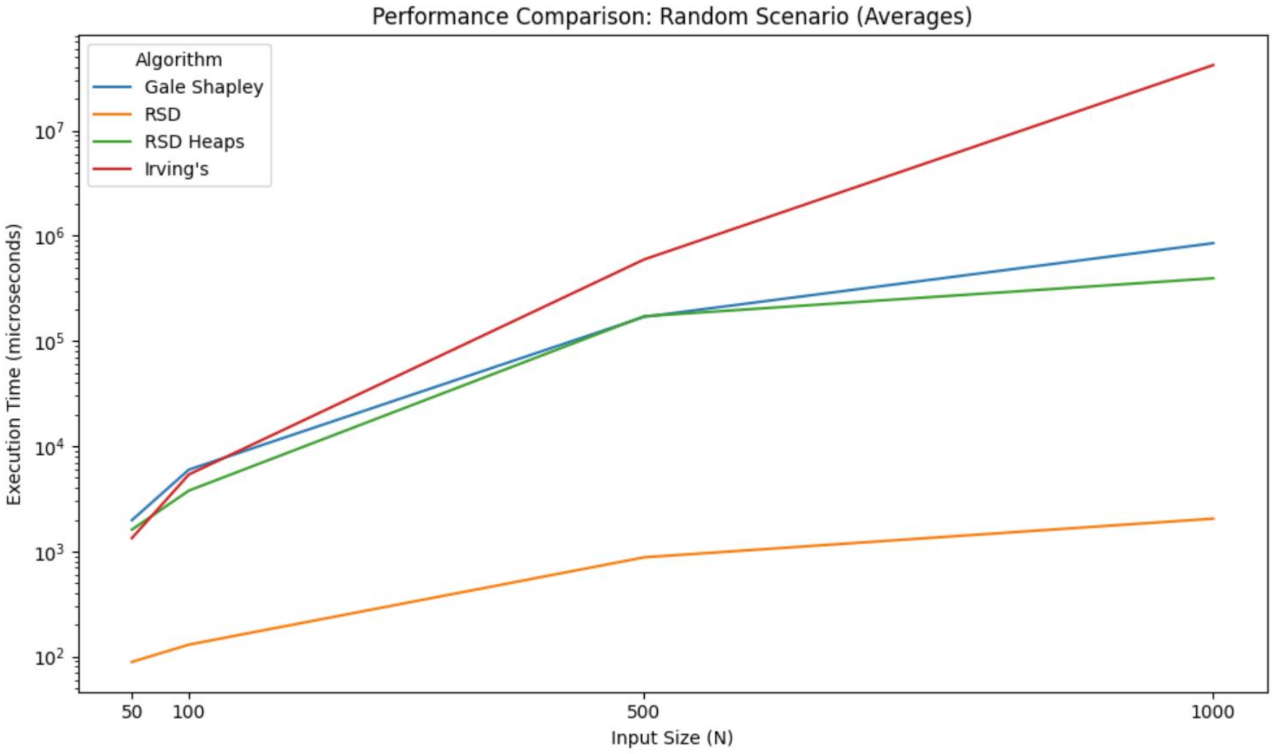


2.3 Experimentation Setup Cont.

- Hand-crafted input sizes were fixed
- For generated inputs, we tested four maximum sizes: **50, 100, 500, 1000**
- These sizes were chosen to measure algorithm efficiency as **n** increases while keeping experimentation time reasonable
- Each input size was run twice to compute an average since the data was randomly generated

Figure 9

Average of All Trials for Random Data



2.4 Hardware Characteristics

- All parts of this experiment were executed in Google Colab, which runs code on a remote virtual machine.

- The environment provided the following hardware configuration when we ran our experiment on November 18th, 2025, at 7:30 P.M. CST:

```
Python version: 3.12.12
model name      : Intel(R) Xeon(R) CPU @ 2.20GHz
MemTotal:       13286956 kB
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 22.04.4 LTS
Release:        22.04
Codename:       jammy
```

3.1 Results

Hand-Created Inputs

Trial One:
- Gale Shapley: 15.3 ms
- Random Serial Dictatorship: 26.1 ms
- Random Serial Dictatorship with Heaps: 17.5 ms
- Irving’s Algorithm: 18.8 ms

Trial Two:
- Gale Shapley: 14.8 ms
- Random Serial Dictatorship: 22.4 ms
- Random Serial Dictatorship with Heaps: 15.5 ms
- Irving’s Algorithm: 17.5 ms

Figure 9

Average of All Trials for Random Data

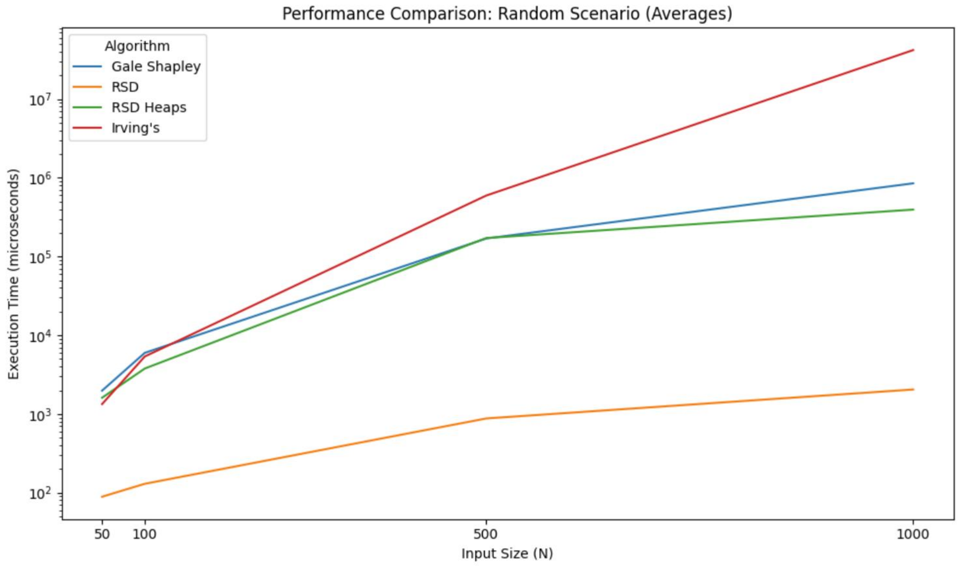
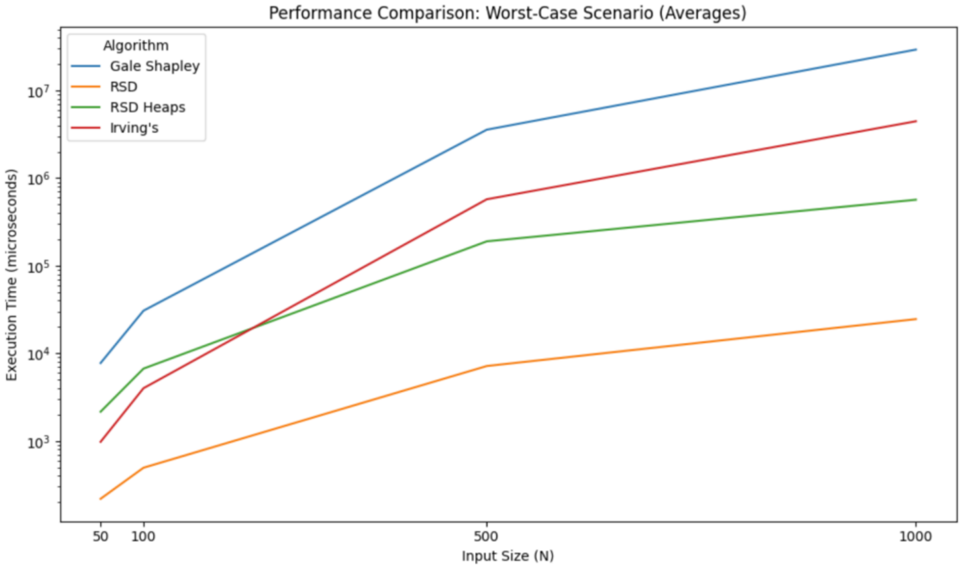


Figure 10

Average of All Trials for Worst-Case Data



3.2 Interpretation

•Stable algorithms (GS and IA) showed the highest runtimes
•Unstable algorithms (RSD with and without heaps) ran the fastest
•This aligned with our expectations
•Despite having the same theoretical time complexities, GS, IA, and RSD showed different practical behavior, with **Gale–Shapley consistently producing the longest runtimes on worst-case inputs**
•We also expected heaps to speed up RSD, but in some cases **RSD with heaps ran slower than Irving's Algorithm**

Figure 9

Average of All Trials for Random Data

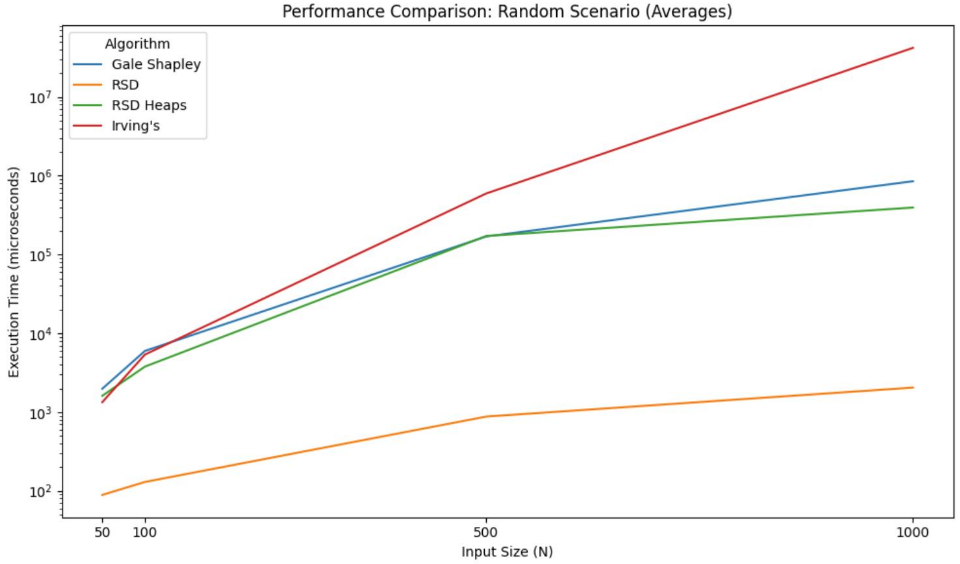
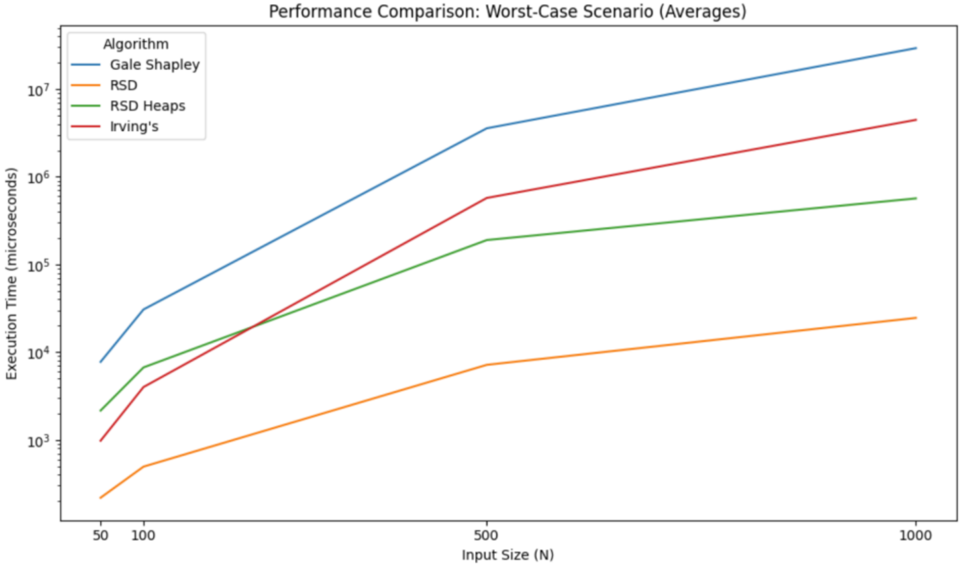


Figure 10

Average of All Trials for Worst-Case Data



3.2 Interpretation Cont.

Limitations
- Algorithms were not designed to solve the same problem
- Data cannot be applied universally, requiring accommodations for Irving's Algorithm
- The experiment was run entirely in Google Colab, which introduced its own computational constraints

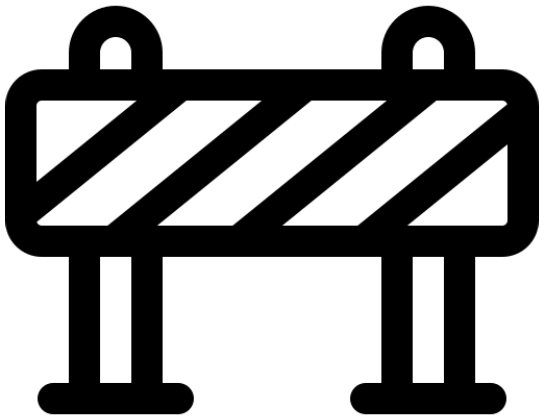
Things We Would Change
- Be more selective about which algorithms to include in the experiment
- Record runtimes in seconds instead of microseconds

Further Research
- Analyze normal distribution and standard deviation of runs
- Explore practical uses of these algorithms
- Modify Gale–Shapley for ties or incomplete lists and assess impact

Final Thoughts
- Experiment answered our research question
- Stable algorithms show clear time trade-offs on worst-case inputs

4 Obstacles Encountered

- **Coordinating meeting times** was difficult due to conflicting schedules
- **Google Colab could not export graphs**, so we had to manually screenshot them and transfer the data into custom tables
- **The algorithms required different input formats**, so we had to adjust the inputs for each run and compare results with those differences in mind



5 Works Cited

1. Anthropic. Claude, Sonnet 4.5, Anthropic, 2025. Accessed 17 Nov. 2025; Used to write sample data generation functions.
2. Austin, David. "The Stable Marriage Problem and School Choice." *American Mathematical Society*, March 2015, ams.org. Accessed 13 Nov. 2025.
3. Google. Google Gemini, Gemini 2.5 Flash, Google, 2025. Accessed 17 Nov. 2025; Used for debugging and chart generation.
4. Hidakatsu, Joe. Structure of the Stable Marriage and Stable Roommate Problems and Applications. Master's thesis, University of South Carolina, 2016. Scholar Commons, https://scholarcommons.sc.edu/etd/3756
5. Irving, Robert W. "An Efficient Algorithm for the 'Stable Roommates' Problem." *Journal of Algorithms*, Volume 6, Issue 4, December 1985, Pages 577-595, sciencedirect.com. Accessed 12 Nov. 2025.
6. Kun, Jeremy. "Serial Dictatorships and House Allocation." *Math ∩ Programming*, October 2015, jeremykun.com. Accessed 13 Nov. 2025.
7. Lo, Raymond. "A Nobel-Winning Solution to Stable Marriages" *Project Delta,* July 2014, pjdelta. Accessed 24 Nov. 2025.
8. OpenAI. ChatGPT, version 5.1, OpenAI, 2025. Accessed 12 Nov. 2025; Used to generate algorithms and help with condensing presentation phrasing.
9. Shi, Hua. "Game Theory and Data Science." *Medium,* Feb 2020, https://melaniesoek0120.medium.com/game-theory-for-data-scientists-81c5756fdf83. Accessed 24 Nov. 2025.