Analysing Roommate Matchmaking Algorithms

Sore Adedotun, Sarah Mohamed, and Sunny Parmar

COMP 4602

Dr. Koon-Ho Alan Tsang

Abstract

This paper explores approaches to roommate matchmaking, drawing on the principles of the *Stable Roommate* and *Stable Marriage Problems*. Unlike traditional methods that require individuals to rank each other, this paper discusses algorithms that match roommates based on how closely their lifestyle preferences align. The categories considered are Cleanliness, Noise Sensitivity, Hospitality, Preferred Work Time, Social Orientation, and Smoking Habits. When looking at a permutation, recursion and a greedy combination algorithm, we compare everyone's preferences within the categories against those of every other participant. The algorithms assign a score reflecting the compatibility of preferences between individuals. These scores are then used to pair individuals whose preferences are most aligned. The outcomes of these algorithmic pairings are presented as a weighted graph, where nodes represent individuals and the edges between nodes are weighted to indicate the degree of preference compatibility. This visualization shows which individuals are matched and highlights which traits are most influential in promoting successful matches.

Introduction

The transition to university life or a move to a new city often entails significant lifestyle adjustments, particularly regarding shared living. Finding a compatible roommate is crucial in such scenarios as compatibility can greatly impact personal satisfaction and academic or professional productivity. This need for effective roommate pairing presents a well-recognized challenge that various fields, including mathematics and computer science, have been addressing by studying stable matching problems. Historically, the stable matching problem has been central to economic theory and algorithmic research. It gained prominence with the seminal work of Gale and Shapley in 1962, who introduced an algorithm to solve the stable marriage problem [2]. This algorithm ensures that in a group of men and women, where each person ranks their potential partners, pairs are formed such that no two individuals would prefer each other over their current partners. An extension of this problem to non-binary groupings, such as same-sex roommates, is known as the stable roommate problem. Unlike the stable marriage problem, a stable solution is not guaranteed in the roommate scenario. The first algorithmic approach to find a solution, or confirm its absence, was developed by Robert Irving [4]. Psychological research supports the hypothesis that individuals sharing similar lifestyles and personality traits are more likely to maintain harmonious living conditions. This concept, known as homophily, suggests that similarity fosters compatibility and reduces conflict [1]. Building on this foundation, our research aims to delve into the algorithms of roommate matchmaking by focusing on lifestyle and personality attributes. Previous studies have employed various algorithms to address this issue, each offering different levels of efficacy and efficiency. In this study, we explore strategies such as utilizing a permutation, recursion and greedy combination algorithm to assess their performance in generating compatible roommate pairings based on real questionnaire data gathered from COMP4602 students. This approach allows us to directly test the impact of different matching attributes on living harmony and overall satisfaction, shedding light on the practical applications of theoretical models in real-world settings.

Related Work

Almost Stable Matchings by Truncating the Gale–Shapley Algorithm

This study investigates the application of a truncated Gale-Shapley algorithm under the constraint of limited proposal rounds, which is crucial for distributed systems where complete global information and unlimited interactions are impractical [3]. The findings provide a unique insight into achieving near-stable matchings in environments where each participant has limited local knowledge about potential matches. This work is particularly relevant for large-scale distributed systems, such as online roommate matching platforms, where participants must make decisions based on partial information. The paper's model of synchronized communication across a network, where nodes only interact with their neighbours, provides a practical framework for developing efficient matchmaking systems in decentralized settings [3]. The approach suggests that even with a finite number of communication rounds, systems can achieve a stable configuration with a reduced number of unstable matches, which is vital for applications in social networks and dynamic matching problems.

Homophily and Individual Performance

This research investigates the nuanced impact of homophily—individuals' tendency to associate with others similar to themselves—on employee performance within the context of a knowledge-intensive sector, specifically an investment bank [1]. The findings highlight how homophily can facilitate easier access to information and resources by fostering closer relationships among similar colleagues, although it may also reduce the diversity of their networks. This is crucial in environments where a broad range of knowledge is essential for effective performance, illustrating how personal choices in relationship formation can significantly affect organizational dynamics. The paper presents a differential impact of homophily based on the hierarchical position within the organization, showing that while beneficial for those in lower positions, it can negatively impact those higher up due to reduced network diversity [1]. This research integrates social network analysis and organizational behaviour theories, offering significant insights into navigating professional settings where the balance of homophily and diversity is crucial to individual and organizational success.

The Stable Roommates Problem with Ties

This article explores a variant of the well-known stable roommates problem where participants can express ties in their preference lists, introducing additional complexity in defining and achieving stability [4]. The authors discuss two specific stability criteria: super-stability and weak stability, presenting a linear-time algorithm capable of finding a super-stable matching, if one exists, within this framework. This is significant as it addresses the computational challenges posed by preferences that include ties, which are known to complicate the solution process. The study also extends its algorithm to handle cases where preference lists are incomplete or partially ordered, further highlighting the real-world applicability of these theoretical constructs [4]. In scenarios where preference lists include ties and are incomplete, the authors note that weakly stable matchings can vary significantly in size, demonstrating that finding a maximum cardinality weakly stable matching is NP-hard. This exploration not only advances the theoretical understanding of the stable roommates problem but also has practical implications for designing algorithms in systems where participant preferences are not strictly ordered.

The Stable Marriage Problem: An Interdisciplinary Review from the Physicist's Perspective

This paper provides an extensive interdisciplinary review of the stable marriage problem, examining it from a physicist's perspective to explore the broader implications of matching theory beyond traditional social contexts [f]. The review encompasses various applications and adaptations of the original Gale-Shapley algorithm, highlighting its robustness and the vast range of scenarios where it can be effectively implemented. The paper underscores the algorithm's influence on the development of complex networks and systems, thereby illustrating its versatility and impact across different fields. Further discussions in the paper detail how the stable marriage problem has been adapted to address challenges in domains such as labour markets, school placements, and other areas where stable matching is crucial [f]. The review serves as a bridge connecting mathematical theories with practical applications, showing how theoretical insights can lead to innovative solutions to real-world problems. This comprehensive examination helps to contextualize the significance of stable matching algorithms in modern computational and social systems.

Background

The challenge of effectively matching individuals based on preferences and characteristics has long been a focal point in both theoretical research and practical applications, encompassing areas ranging from marriage markets to roommate assignments. The underlying complexity of these problems has spurred a wealth of studies that seek to address various nuances of human interactions and preferences, employing sophisticated algorithms to ensure stable and satisfactory outcomes. One of the cornerstone models in this field is the Gale-Shapley algorithm, initially designed to solve the stable marriage problem. This algorithm ensures that all participants are matched in such a way that no two individuals would prefer each other over their current partners, creating a stable matching [2]. Over time, the application of this algorithm has expanded beyond the domain of marriage, addressing different types of matching markets and introducing new variations to accommodate more complex and realistic scenarios, such as preferences with ties and partial information [4]. However, traditional applications of the Gale-Shapley algorithm and its variations often assume full knowledge of all participants' preferences, which is not always feasible in large, distributed systems. Recognizing this limitation, Floreen et al. explored a truncated version of the Gale-Shapley algorithm, designed to achieve nearly stable matchings by conducting only a limited number of proposal-acceptance rounds. This adaptation is particularly pertinent for settings where participants have incomplete knowledge about potential matches, reflecting the real-world constraints of large-scale and distributed environments [3]. Further enhancing our understanding of matching dynamics, Matthews and Garenne introduced a dynamic model to study the marriage market, incorporating variables like age preference and availability, which adapt over time. Their work utilizes iterative proportional fitting to continuously adjust to changing demographic data, offering a model that not only reflects but also predicts market trends, thereby aiding in policy formulation and demographic studies [6]. In parallel, studies on personal relationships and workplace dynamics have shown that homophily, or the tendency of individuals to connect with similar others, plays a crucial role in the stability and satisfaction of matches. Ertug et al.'s investigation into how homophily affects performance in organizational settings underscores the importance of considering personal compatibility in matching algorithms, as it influences information access and resource sharing within networks [1]. These studies collectively provide a robust theoretical foundation for developing advanced matching algorithms. They highlight the necessity of integrating dynamic, real-time data and accommodating diverse and complex human preferences. This background serves as the basis for our research, which aims to refine and apply these principles to the practical challenges of roommate matchmaking, striving to enhance compatibility and satisfaction among matched individuals in shared living arrangements.

Model

In this study, we use a graph-theoretical model to simulate and analyze individual compatibility to optimize flatmate pairings in university or metropolitan living circumstances. Each participant in the simulation is represented as a node, and potential roommate pairings are denoted as edges, with weights on these edges quantifying compatibility based on a comprehensive set of criteria obtained from a standardized questionnaire. These attributes include cleanliness level, noise preference, hospitality, sleep schedule, personality type, and smoking habits, ach crucial for determining lifestyle and personality compatibility. The simulation seeks to maximize the total compatibility score across the network, ensuring that each individual is only paired once, resulting in a perfect matching scenario. By applying three distinct algorithmic strategies—permutation, recursion, and greedy combination—our study rigorously evaluates which method achieves the highest compatibility score among roommates. The permutation method employs a brute force approach, examining every possible configuration and serving as a baseline for comparison. The recursive method reduces computational redundancy, designed to handle mid-sized datasets efficiently. The greedy combination algorithm focuses on rapid processing capabilities, suitable for scenarios requiring quick decision-making, although it presents some limitations in smaller sample sizes. We will examine the runtime and accuracy of these algorithms against a manually derived solution to assess their practical efficacy. By comparing algorithmic outcomes to these manual pairings, we can measure the precision and effectiveness of each method in real-world settings. This comprehensive approach not only aims to yield highly compatible roommate pairings but also enhances our understanding of the underlying social network phenomena that contribute to successful shared living arrangements.

Methodology

Male Participants Data	Cleanliness	Noise Sensitivity	Social Frequency	Preferred Work Time	Social Orientation	Smoking Habits
P_0	3	3	2	2	3	5
P_1	3	4	2	3	3	5
P_2	2	3	2	3	4	4
P_3	2	3	1	2	3	4
P_4	3	3	2	2	1	5
P_5	2	5	2	2	1	5
P_6	2	4	2	2	2	3
P_7	2	4	2	3	3	5

Female Participants Data	Cleanliness	Noise Sensitivity	Social Frequency	Preferred Work Time	Social Orientation	Smoking Habits
P_0	5	5	3	2	4	5
P_1	3	4	4	3	3	2
P_2	4	3	2	4	5	5
P_3	3	4	2	2	3	5

The questionnaire options were displayed to the survey participants (potential roommates) on a Likert scale, where we quantified this categorical data by representing each option as a number from 1 to 5. The responses were then reviewed for errors before extracting the data from participants of 4 women and 12 men. As previously mentioned the goal is to attempt to create 6 pairs of compatible roommate matchings. The matchings would ideally consist of 4 pairs of men and 2 pairs of women, out of these 12 participants. To do this, we designed pairing algorithms in the Python programming language and compared the results for performance and accuracy.

A manual test was carried out using the 12 men's scores and 4 women's scores in every category. Each category's score for a participant was compared to every other participant's, repeated for every category and for each category where two participants have the same score, a similarity score of 1 was given. Applying this across all men, the resulting pairs were (P_0, P_4), (P_1, P_7), (P_2, P_3), (P_5, P_6), (P_8, P_9) and (P_{10}, P_{11}). Applying the same approach to the women, the result was (P_0, P_2) and (P_1, P_3).

<u>Method 1: Permutation</u>

This first method employs a brute force algorithm which checks every possible permutation of the participants. It begins by creating an array for every set of response values for each participant where the response value for a question is at the same index for all participants. For each permutation of participants that exists, the similarity score is calculated by finding the inverse of the absolute difference between the sum of two participants' response values for all categories. A higher similarity score represents a more suitable roommate pairing and an overall optimal matching. The generated graphs of the relationships are then generated with the participants represented as nodes. The edges connecting each participant are weighted with the number of equal category scores.

<u>Method 2: Recursion</u>

The second method also employs a brute force algorithm but aims to improve on the first with a recursive approach. The same approach is used to calculate the overall similarity score between participants. Rather than comparing all possible permutations of the participants and their pairs, a depth-first search explores all possible combinations of pairings and selects the one that maximizes the total similarity across all pairs. The recursive_pairing function takes three parameters, people: the list of people to pair, ratings: the list of all sub-lists containing the collection of scores of a person in all categories, and current_pairs: a list to track the current pairing during the recursive process.

The base case is an empty list showing all individuals have been paired, and if it is not empty, the first person in the list is selected and the attempt is made to pair this person with everyone else in the list. For each potential pair (first, second), a new pair (new_pairs) is generated by adding the current pair to the existing list of current pairs. The list of remaining people (new_people) is then created by removing the paired people from the original list. The function then recursively calls itself with the new list of people and the updated pair list. Each recursive call returns a total similarity score and a configuration of pairs. The function compares the returned score with the best score tracked so far (best_score). If the returned score is higher than the best_score, it updates the best_score and saves the associated pair configuration as best_configuration. When the list is empty, the process terminates.

Method 3: Greedy algorithm and Combination

The final method is a greedy algorithm that employs combinations from the itertools module. The similarity scores are calculated as discussed in the previous methods. The function iterates over all possible combinations of pairs, comparing each person with every other person combination exactly once. For each pair, the calculate_similarity function is called, and the score is stored in the similarities dictionary with the tuple as the key. The pairs are sorted based on their similarity scores in descending order. This is to ensure that the pairs with the highest similarity are considered first, adhering to the greedy nature of the algorithm. The sorted list is iterated and each element per pair is checked if it has already been matched with someone else. If neither has been matched, they are added to the pairs list and marked as matched by adding them to the matched set. This repeats until all people are paired or there are no more pairs.

Results and Analysis

Method 1		Method 2		Method 3	
Number of Participants	Runtime (ms)	Number of Participants	Runtime (ms)	Number of Participants	Runtime (ms)
16	-	16	20086.3812	16	0.3061
14	-	14	1199.3599	14	0.1628
12	-	12	81.8411	12	0.134
10	22006.882	10	6.655	10	0.1068
8	202.5161	8	0.7172	8	0.0823
4	0.1421	4	0.061	4	0.0451

Performance

The tables depict the resulting runtime of each approach for different numbers of people. To test the performance, the lists of men and women were combined to generate pairs from all 16 participants before being split. According to the data gathered, method 1 is the most expensive operation with the longest runtime and is unable to compute pairs for numbers of more than 10 people within an acceptable time. This result is expected because the permutations unnecessarily repeat calculations as the same pairs are revisited in different orders e.g. (1, 2) and (2, 1) despite having the same score. As a result of these operations, the complexity of the process is abysmal at $O((n-1)!)$. Compared to method 1, the runtime of method 2 is faster and more feasible for as many as 16 participants. The recursive approach cuts down on the repeated permutations done in method 1 leading to a time complexity of $O((n-1)!)$).

Method 3 improves on the runtime of the other techniques considerably because its algorithm implements combinations and does not check as many pairing possibilities as the previous two. The estimated complexity is O(n^(2*k)) due to the pairwise calculations still being the dominant process.

<u>Accuracy:</u>

The manual pairing resulted in the following configuration for men: (P_0, P_4), (P_1, P_7), (P_2, P_3), (P_5, P_6), (P_8, P_9) and (P_{10}, P_{11}). The results for women were (P_0, P_2) and (P_1, P_3). All 3 methods had the same pairing result (see Figure 1 below) as the correct manual pairing for the men. For women, the results of methods 1 and 2 (See Figure 2 below) were the same as the manual approach but method 3 had a different result: (P_0, P_3) and (P_1, P_2).

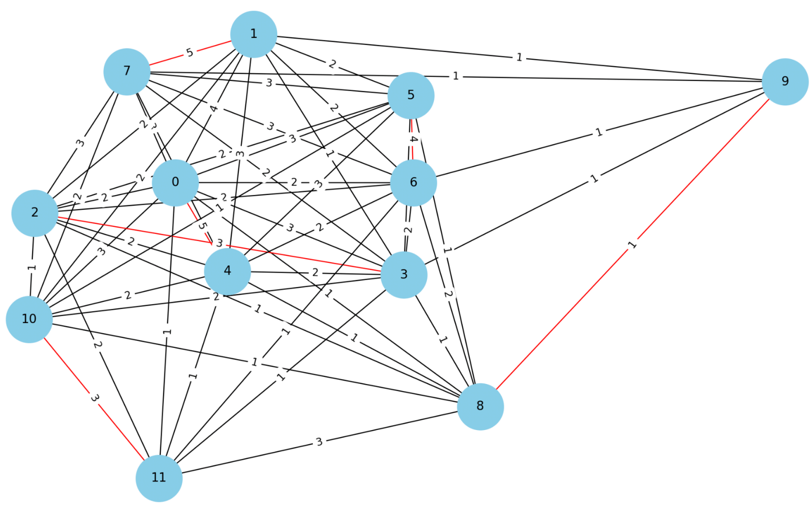

Figure 1. Network graph showing the optimal pairings of men using methods 1, 2 and 3

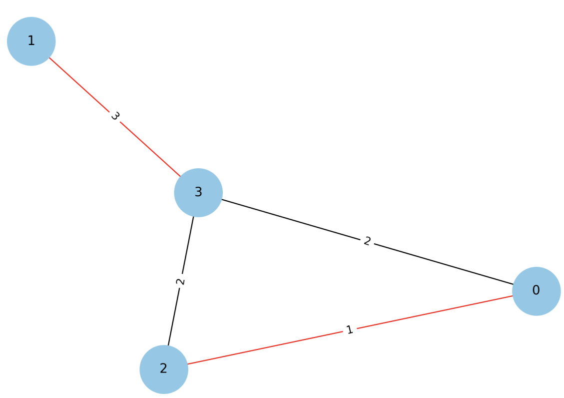

Figure 2. Network graph showing the result of methods 1 and 2 for women

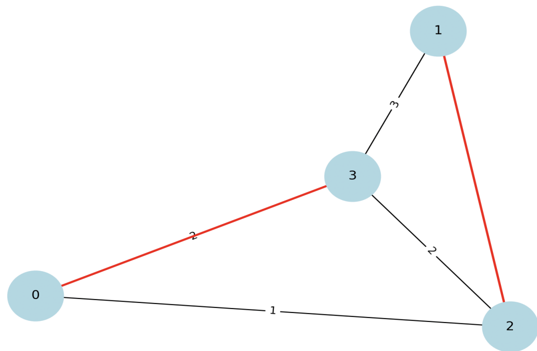

Figure 3. Network graph showing the result of method 3 for women

This shows that the three methods are effective for 4 to 10 people but method 1 is infeasible above 10 and method 3 is infeasible below 6. Method 2 is optimal for the sample of our data from class as it pairs the 12 men and 4 women effectively. Method 3 fails for the 4 participants which may be a niche case but for our study cannot be preferred to method 2.

Conclusion

This paper explored the application of various algorithmic strategies—permutation, recursion, and greedy combination algorithms—in the context of roommate matchmaking, focusing on the alignment of lifestyle and personality attributes for optimal pairing. By leveraging real questionnaire data from COMP4602 students, the practical viability and efficiency of these methods are evident for specific real-world settings. The permutation method, while comprehensive, proved to be too computationally intensive for larger datasets. The recursive approach, in particular, stood out for its effectiveness and feasibility, making it especially suitable for our specific problem involving real questionnaire data gathered from COMP4602 students. This method reduced unnecessary computational redundancies and adeptly handled the mid-sized dataset provided by the survey. The greedy combination algorithm excelled in scenarios requiring rapid decision-making, albeit with some limitations in smaller groups. In conclusion, our study outlines algorithms to be utilized in specific scenarios and presents a positive direction, for continual advancements in the study of roommate matchmaking.

References

[1] Ertug, G., Gargiulo, M., Galunic, C., & Zou, T. (2018). Homophily and Individual Performance. *Organization Science (Providence, R.I.)*, *29*(5), 912–930. https://doi.org/10.1287/orsc.2018.1208

[2] Fenoaltea, E. M., Baybusinov, I. B., Zhao, J., Zhou, L., & Zhang, Y.-C. (2021). The Stable Marriage Problem: An interdisciplinary review from the physicist's perspective. *Physics Reports*, *917*, 1–79. https://doi.org/10.1016/j.physrep.2021.03.001

[3] FLOREEN, P., KASKI, P., POLISHCHUK, V., & SUOMELA, J. (2010). Almost Stable Matchings by Truncating the Gale-Shapley Algorithm: Matching Under Preferences. Algorithmica, 58(1), 102–118.

[4] Irving, R. W., & Manlove, D. F. (2002). The Stable Roommates Problem with Ties. Journal of Algorithms, 43(1), 85–105. https://doi.org/10.1006/jagm.2002.1219

[5] Jung, J. J., You, E., & Park, S.-B. (2013). Emotion-based character clustering for managing story-based contents: a cinemetric analysis. Multimedia Tools and Applications, 65(1), 29–45. https://doi.org/10.1007/s11042-012-1133-x

[6] Matthews, A. P., & Garenne, M. L. (2013). A dynamic model of the marriage market—Part 1: Matching algorithm based on age preference and availability. *Theoretical Population Biology*, *88*, 78–85. https://doi.org/10.1016/j.tpb.2013.01.006

[7] Park, S.-B., Oh, K.-J., & Jo, G.-S. (2012). Social network analysis in a movie using character-net. Multimedia Tools and Applications, 59(2), 601–627. https://doi.org/10.1007/s11042-011-0725-1