# The Stable Marriage Problem

William Hunt
Lane Department of Computer Science and Electrical Engineering,
West Virginia University,
Morgantown, WV
William.Hunt@mail.wvu.edu

## 1 Introduction

Imagine you are a matchmaker, with one hundred female clients, and one hundred male clients. Each of the women has given you a complete list of the hundred men, ordered by her preference: her first choice, second choice, and so on. Each of the men has given you a list of the women, ranked similarly. It is your job to arrange one hundred happy marriages.

In this problem, we have a set of $n$ men and $n$ women. Each person has their own preference list of the persons they want to marry. Our job is to determine an assignment where each man is married to one and only one woman (monogamous and heterosexual).

Each man, denoted by the list (**A,B,C,...**), has a list of women (**a,b,c,...**) ordered by his preference as in figure 1. Each woman has a similarly ranked list. Every man is on every woman's list and every woman is on every man's list.

The goal is to have a set of stable marriages, **M**, between the men and the women.

When given a married pair, **X-a** and **Y-b**, if man **X** prefers another woman **b** more than his current wife **a** and woman **b** prefers **X** more than her current man **Y**, then **X-b** is called a dissatisfied pair.

| Men's List | Women's List |
|---|---|
| A: a,b,c,d | a: A,B,C,D |
| B: b,a,c,d | b: D,C,B,A |
| C: a,d,c,b | c: A,B,C,D |
| D: d,c,a,b | d: C,D,A,B |

Figure 1: Sample Preference List for Men and Women

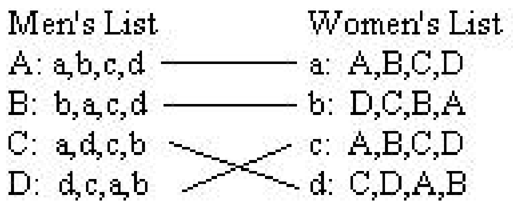


Figure 2: Sample Stable Marriage for Men and Women

The marriage **M** is said to be a stable marriage if there are no dissatisfied pairs. The figure 2 shows a stable marriage for the preference lists given in figure 1.

The simplest approach to solving this problem is the following:

**Function** SIMPLE-PROPOSAL-BUT-INVALID

```
Start with some assignment between the men and women
loop
   if assignment is stable then
      stop
   else
      find a dissatisfied pair and swap mates to satisfied the pair
   end if
end loop
```

**Algorithm 1.1:** An Invalid Simple Algorithm for Proposal

This will NOT work since a loop can occur. Swaps can be made that might continually result in dissatisfied pairs. We can come up with an equally simple, deterministic algorithm.

## 1.1 Proposal Algorithm

**Function** Proposal-Algorithm

```
while there is an unpaired man do
  pick an unpaired man X and the first woman w on his list
  remove w from his list so it won't be picked again
  if w is engaged then
    if w prefers X more than her current partner Y then
      set X-w as married
      set Y-w as unmarried so now Y is unpaired
    else
      X is still unpaired since w is happier with Y
    end if
  else
    the woman was not previously paired so accept immediately, X-w, as married
  end if
end while
```

**Algorithm 1.2:** Proposal Algorithm

At each iteration, the men will eliminate one woman from their list. Since each list has n elements, there are at most $n^2$ proposals.

Now, we have a few questions to ask regarding this algorithm.

1. Does the algorithm terminate?
   Once a woman becomes attached, she remains married, but can change a partner for a better mate that proposes to her. That makes this algorithm a greedy algorithm for the women. A man will eliminate a choice from his list during each iteration, thus if the rounds continue long enough, he will get rid of his entire preference list entries and there will be no one left to propose too. Therefore all women and men are married and the algorithm terminates.

2. Is the resulting marriage a "stable marriage"?
   To show that it is a stable marriage, let's assume we have a dissatisfied pair, **X-b**, where in the marriage they are paired as **X-a** and **Y-b**. Since **X** prefers woman **b** over his current partner **a**, then he must have proposed to **b** before **a**. Woman **b** either rejected him or accepted him, but dropped him for another better man than **X**. Thus, **b** must prefer **Y** to **X**, contradicting our assumption that **b** is dissatisfied, so it is a stable marriage.

### 1.1.1 Probabilistic Analysis

The following is an average-case analysis of the Proposal Algorithm.

Let $T_P$ = number of proposals made
Since $T_P$ has a lot of dependencies during each step, it is difficult to analyze it.

In this analysis, we are going to assume that the men's lists are chosen uniformly, independently, and at random over all input. The women's lists are arbitrary, but fixed in advance.

Since there are $n!$ different lists, the probability that a man will get a particular sequence is $\frac{1}{n!}$.

We are going to argue that the expected value of the number of proposals is roughly $O(n \ln n)$.

### 1.1.2 Principle of Deferred Decisions

To argue about the expected value, we are going to use the technique of the **Principle of Deferred Decisions**. This principle uses an idea that random choices are not all made in advance but the algorithm makes random choices as it needs them.

An illustration of this technique is the Clock Solitaire Game. In this game, you have a shuffled deck of 52 cards. Four cards are dealt into 13 piles. Each pile is named with a distinct member of *A,1,2,3,...,J,Q,K*. On the first move, draw a card from the $K$ pile. The following draws come from the pile named by the face value of the card from the previous draw. The game ends when you try to draw from an empty pile. If all cards are drawn, then you win.

Will this game end? Yes. It will always end with a king in your hand. There are 4 different cards for each suit in each pile except for the king pile because you started with that particular pile. Therefore, there is possibility of ending the game by drawing all cards from the piles with the last card drawn being a king.

To determine the probability of winning, we need to consider that every time a card is drawn, a new dependency occurs. To calculate this is tough. Another way of determining the probability of winning is to think of the game as drawing cards, one after another without replacement, at random from the deck of cards. To win the game, we need the probability that the 52nd card drawn is a king. Thus, the winning probability is $\frac{\text{4 kings}}{\text{52 cards in deck}} = \frac{1}{13}$.

## 1.2 Amnesiac Algorithm

In the analysis of the Proposal Algorithm, we can simplify by assuming that men generate their lists by generating one woman at a time out of the women that haven't rejected him. A problem that arises is that a woman's choice depends on the man that proposes to her.

To resolve the woman's dependency problem, we can modify the behavior of our algorithm. We can have the man generate his list by selecting a woman uniformly at random from all $n$ women, including those that have rejected him. He has forgotten the fact that women have already rejected him, thus the Amnesiac Algorithm.

This is easy to analyze because we are dealing with the total number of proposals only because each proposal is independently made to one of the $n$ women chosen at random.

We can let $T_A$ be the number of proposals made by the Amnesiac Algorithm.

$$\text{for all m, } \mathbf{Pr}[T_A > m] \geq \mathbf{Pr}[T_P > m]$$

From above we can see that $T_A$ stochastically dominates $T_P$. Therefore, we do not need to find an upper bound on $T_P$ (which is hard to do). Instead, we can use the upper bound on $T_A$ (which is easy to do).

***Theorem: 1.1***

$$\lim_{n \to \infty} \mathbf{Pr}[T_A > m] = 1 - e^{-e^{-c}}, \ \textit{for } m = n \ln n + cn, \ c \in \Re^{+}$$

This theorem result can be derived using the Coupon Collector's Problem.

### 1.2.1 Coupon Collector's Problem

To analyze how long the algorithm takes, we need to find out how many random proposals to $n$ females need to occur before there are no longer anyone left to propose to.

This is the same as an occupancy problem where there are $m$ balls randomly put into $n$ bins.

That occupancy problem can be translated into the "supermarket" realm. In this problem, there are $n$ types of coupons and $m$ visits to the store. At each visit, you randomly and uniformly get a coupon. The question is: How many visits, $m$, do I have to do to make sure that I have one coupon of each type?

Analysis

Let $X$ be a random variable defined to be the number of trials required to collect at least 1 of each type of coupon.

Let $C_1, C_2, \cdots, C_X$ denote the sequence of trials, where $C_i \in \{1, 2, \cdots n\}$ is the type of the coupon drawn in the $i^{th}$ trial.

$C_i$ is considered a success, if the coupon type $C_i$ was NOT drawn in the $1^{st}$ $i-1$ selections. By this definition, $C_1$ and $C_X$ will always be successes.

Divide the sequence into epochs where epoch $i$ begins with the trial following the $i^{st}$ success and ends with the trail on which we obtain the $(i+1)^{st}$ success. So, we can define a random variable, $X_i$, with $0 \leq i \leq n-1$, as the number of trials in the $i^{st}$ epoch.

We can then express $X$ as a function of $X_i$ by the following since we are dividing $X$ into different portions:

$$X = \sum_{i=0}^{n-1} X_i$$

Now we need to answer the following question: What about the distribution of each $X_i$?

Let $P_i$ be the probability of success of any trial of the $i^{st}$ epoch. $P_i$, in the occupancy problem viewpoint, is the probability of getting a ball that hasn't been drawn before. Since there have already been $i$ balls drawn, the probability of success is:

$$P_i = \frac{n-i}{n}$$

$X_i$ is geometrically distributed, therefore the following are true by definition of the distribution:

$$\mathbf{E}[X_i] = \tfrac{1}{P_i}$$

$$\sigma^2_{X_i} = \tfrac{1-P_i}{P_i^2}$$

By linearity of expectations and summation of variances of independent random variables, we can calculate $\mathbf{E}[X]$ and $\sigma^2_X$ as follows:

$$\mathbf{E}[X] = \sum_{i=0}^{n-1} \mathbf{E}[X_i] = \sum_{i=0}^{n-1} \frac{1}{P_i} = \sum_{i=0}^{n-1} \frac{n}{n-i} = n\sum_{i=1}^{n} \frac{1}{i} = nH_n$$

$$H_n = \ln n + \Theta(1), therefore, \mathbf{E}[X] = n\ln n + O(n).$$

$$\sigma^2_X = \sum_{i=0}^{n-1} \sigma^2_{X_i} = \sum_{i=0}^{n-1} \frac{1-P_i}{P_i^2} = \sum_{i=0}^{n-1} \frac{ni}{(n-i)^2} = \sum_{i=1}^{n} \frac{n(n-i)}{i^2} = n^2 \sum_{i=1}^{n} \frac{1}{i^2} - nH_n$$

Since $\sum_{i=1}^{n} \frac{1}{i^2}$ converges to $\frac{\pi^2}{6}$ as $n \to \infty$, we have the following limit:

$$\lim_{n\to\infty} \frac{\sigma^2_X}{n^2} = \frac{\pi^2}{6}.$$

Our next goal is to show that $X$ will not deviate far from its expected value.

Let $\mathbf{E}_i^r$ denote the event that coupon $i$ is NOT collected in the $1^{st}$ $r$ trials. These trials are done independently and with replacement.

$$\mathbf{Pr}[\mathbf{E}_i^r] = \left(1 - \frac{1}{n}\right)^r \leq e^{\frac{-r}{n}}.$$

If we let $r = \beta n \ln n$, then $\mathbf{Pr}[\mathbf{E}_i^r] = n^{-\beta}$.

Since the probability of a union of events is always less than the sum of the probabilities of those events, we can calculate $\mathbf{Pr}[X > r]$, for $r = \beta n \ln n$ as:

$$\mathbf{Pr}[X > r] = \mathbf{Pr}\left[\bigcup_{i=1}^{n} \mathbf{E}_i^r\right] \leq \sum_{i=1}^{n} \mathbf{Pr}[\mathbf{E}_i^r] \leq \sum_{i=1}^{n} n^{-\beta} = n^{-(\beta-1)}$$

### 1.2.2 Poisson Heuristic

To help us show that $X$ will not deviate far from its expected value, we can utilize the Poisson distribution as an approximation of the binomial distribution.

Let $N_i^r$ be the number of time coupon $i$ is chosen during the $1^{st}$ $r$ trials. These trials follow the binomial distribution with parameters $r$ and $p = \frac{1}{n}$.

$$\mathbf{Pr}[N_i^r = x] = \binom{r}{x} p^x (1-p)^{r-x}, \text{ with } 0 \leq x \leq r$$

$$\mathbf{Pr}[\mathbf{E}_i^r] = \mathbf{Pr}[N_i^r] = 0$$

Let $\lambda \in \Re^+$. A random variable $Y$ follows the Poisson distribution with parameter $\lambda$ if for any positive integer $y$,

$$\mathbf{Pr}[Y = y] = \frac{\lambda^y e^{-\lambda}}{y!}$$

Assuming that $\lambda$ is small and $r \to \infty$, then the Poisson distribution is a good approximation for the binomial distribution. When we use the Poisson distribution, we can show that the $\mathbf{E}_i^r$ events are independent.

Using this approximation, with $\lambda = \frac{r}{n}$, the probability of the event $\mathbf{E}_i^r$ is:

$$\mathbf{Pr}[\mathbf{E}_i^r] = \mathbf{Pr}[N_i^r = 0] \approx \frac{\lambda^0 e^{-\lambda}}{0!} = e^{\frac{-r}{n}}$$

A benefit of using the Poisson distribution is that we can now say that the events $\mathbf{E}_i^r$, for $1 \leq i \leq n$, are "almost independent".

***Claim: 1.1*** *For $1 \leq i \leq n$, and for any set of indices $\{j_1, \cdots, j_k\}$ not containing $i$, we want to show:*

$$\mathbf{Pr}\left[\mathbf{E}_i^r | \bigcap_{l=1}^{k} \mathbf{E}_{j_l}^r\right] \approx \mathbf{Pr}[\mathbf{E}_i^r] = e^{\frac{-r}{n}}$$

<u>Proof</u>:
*Working with the left hand side:*

$$\mathbf{Pr}\left[\mathbf{E}_i^r | \bigcap_{l=1}^{k} \mathbf{E}_{j_l}^r\right] = \frac{\mathbf{Pr}\left[\mathbf{E}_i^r \cap \left(\bigcap_{l=1}^{k} \mathbf{E}_{j_l}^r\right)\right]}{\mathbf{Pr}\left[\bigcap_{l=1}^{k} \mathbf{E}_{j_l}^r\right]} \Leftarrow \textit{by the definition of conditional probabilities}$$

*The numerator is the same as saying we want $k+1$ coupons not selected in $r$ trials, and the denominator is when we want $k$ coupons not selected in $r$ trials, giving us:*

$$= \frac{\left(1 - \frac{k+1}{n}\right)^r}{\left(1 - \frac{k}{n}\right)^r}$$

*Using the identity:* $\lim_{\alpha\to\infty}(1+\alpha)^{\frac{1}{\alpha}} = e$, *we can rewrite the left hand side as:*

$\approx \frac{e^{\frac{-r(k+1)}{n}}}{e^{\frac{-rk}{n}}} = e^{\frac{-r}{n}}$, *thus the desired result of declaring the events independent is shown.*

□

A new question: What is the probability that <u>all</u> coupons are collected in the first $m$ trials?

$$\mathbf{Pr}\left[\neg\left(\bigcup_{i=1}^{n}\mathbf{E}_i^m\right)\right] = \mathbf{Pr}\left[\bigcap_{i=1}^{n}(\neg\mathbf{E}_i^m)\right] \quad \Leftarrow \text{ by DeMorgan's Law}$$

$$= \left(1 - e^{\frac{m}{n}}\right)^n \approx e^{-ne^{\frac{-m}{n}}} \quad \Leftarrow \text{ since they are independent events.}$$

Let $m = n(\ln n + c)$, for any $c \in \Re$, then by the preceding argument,

$$\mathbf{Pr}[X > m = n(\ln n + c)] = \mathbf{Pr}\left[\left(\bigcup_{i=1}^{n}\mathbf{E}_i^m\right)\right] \approx \mathbf{Pr}\left[\bigcap_{i=1}^{n}(\neg\mathbf{E}_i^m)\right] = 1 - e^{-e^{-c}}$$

This shows that the probability that all coupons collected within $m$ trials is very high. There is also not a lot of deviation from $n\ln n$ since for a large positive $c$, the probably of $e^{-e^{-c}}$ is close to 1 and is negligibly small for a large negative $c$.

***Result: 1.1*** *Therefore, we can conclude our analysis of the Stable Married Problem by summarizing the following points:*

1. *The worst case of the algorithm is* $n^2$.
2. *The expected (average) case is* $n\ln n$.
3. *Deviation is small from the expected value.*