

## Problem 1 (10 pts)

For this problem, we explore the issue of *truthfulness* in the Gale-Shapley algorithm for Stable Matching. Show that a participant can improve its outcome by lying about its preferences. Consider  $r \in R$ . Suppose  $r$  prefers  $p$  to  $p'$ , but  $p$  and  $p'$  are low on  $r$ 's preference list. Show that it is possible that by switching the order of  $p$  and  $p'$  on  $r$ 's preference list,  $r$  achieves a better outcome, e.g., is matched with a  $p''$  higher on the preference list than the one if the actual order was used.

*Hint:* Prove the claim by finding one specific instance of stable matching problem and comparing the stable matching before and after the switching.

## Solution

We can show that it is possible for a receiver  $r$  to obtain a more-preferred match by being untruthful, by switching the order of two options that  $r$  actually likes less (even though in his true ranking one is better than the other).

Let the two sides be:

- **Residents (receivers)  $R$ :**  $r, s, t$
- **Programs (proposers)  $P$ :**  $p, p', q$

Assume that the matching is produced by a **program-proposing Gale-Shapley algorithm**.

### True Preference Lists

- **Resident  $r$ 's true preferences:**

$$q > p > p'$$

So  $r$  likes  $q$  best; although he prefers  $p$  to  $p'$ , both  $p$  and  $p'$  are lower than  $q$ .

- **Resident  $s$ 's true preferences:**

$$p > q > p'$$

- **Resident  $t$ 's true preferences:**

$$p > p' > q$$

- **Programs' preferences:**

- $p$ :  $r > s > t$
- $p'$ :  $r > t > s$
- $q$ :  $s > r > t$

## Outcome When $r$ Reports Truthfully

We run the **program-proposing Gale-Shapley algorithm** using the true lists.

### 1. Round 1:

- $p$  proposes to its top choice:  $r$ .
- $p'$  also proposes to its top choice:  $r$ .
- $q$  proposes to its top choice:  $s$ .

### 2. Resident Decisions:

- $r$  receives proposals from  $p$  and  $p'$ .
- Using his true order  $q > p > p'$ , he compares these two. Since  $p > p'$  in his true list, he tentatively holds  $p$  and rejects  $p'$ .
- $s$  gets  $q$ .
- $t$  gets no proposal in this round.

### 3. Round 2:

- $p'$ , having been rejected by  $r$ , now moves to its next choice and proposes to  $t$ .

### 4. Final Matching:

- $r$  is matched with  $p$ .
- $s$  is matched with  $q$ .
- $t$  is matched with  $p'$ .

Outcome for  $r$ : He gets  $p$ , but he would prefer  $q$  over  $p$ . Since  $q$  proposed only to  $s$ ,  $q$  is already taken.

## Outcome When $r$ Misreports

Now suppose that  $r$  lies about his 2nd and 3rd rankings by switching  $p$  and  $p'$ . In other words, he submits as his preference list:

$$q > p' > p$$

Even though he actually prefers  $p$  to  $p'$ , he reverses these when reporting.

Now run the algorithm again:

### 1. Round 1:

- $p$  proposes to its top choice:  $r$ .
- $p'$  proposes to its top choice:  $r$ .
- $q$  proposes to its top choice:  $s$ .

## 2. Resident Decisions:

- $r$  receives proposals from  $p$  and  $p'$ .
- Using his reported order  $q > p' > p$ , he compares the two proposals and holds  $p'$  (since  $p' > p$  in his reported list), rejecting  $p$ .
- $s$  gets  $q$ .
- $t$  gets nothing yet.

## 3. Round 2:

- $p$ , having been rejected by  $r$ , moves to its next choice and proposes to  $s$ .
- $s$ 's true order is  $p > q > p'$ , so she prefers  $p$  over  $q$ .
- $s$  drops  $q$  and holds  $p$ .
- $q$ , now rejected by  $s$ , proposes to its next choice, which is  $r$ .

## 4. Round 3:

- $r$  is currently holding  $p'$  from round 1.
- Now  $r$  receives a proposal from  $q$ .
- Since  $r$ 's true order is  $q > p > p'$ , he prefers  $q$  over his current match  $p'$ .
- So  $r$  drops  $p'$  and accepts  $q$ .

## 5. Round 4:

- $p'$ , now rejected by  $r$ , goes to its next choice and proposes to  $t$ .
- $t$  (true order:  $p > p' > q$ ) accepts  $p'$ .

## 6. Final Matching:

- $r$  is now matched with  $q$ .
- $s$  is matched with  $p$ .
- $t$  is matched with  $p'$ .

## Conclusion

By simply swapping the order of  $p$  and  $p'$  in his reported preference list (even though he truly prefers  $p$  to  $p'$ ), resident  $r$  can trigger a different chain reaction of proposals.

In the truthful run, he ended up with  $p$  (his second choice). However, by lying, he eventually receives a proposal from  $q$ , his true first choice.

This proves that in the program-proposing Gale-Shapley algorithm, truth-telling is not necessarily an optimal strategy. A participant can benefit by misreporting their preferences.

## Problem 2 (10 pts)

Arrange in increasing order of asymptotic growth. All logs are in base 2.

1.  $n^{\frac{5}{3}} \log^2 n$
2.  $2^{\sqrt{\log n}}$
3.  $\sqrt{n^n}$
4.  $\frac{n^2}{\log n}$
5.  $2^n$

## Solution

### Step 1: Rewrite Functions

1.  $f_1(n) = n^{\frac{5}{3}} (\log n)^2$
2.  $f_3(n) = n^{\frac{n}{2}}$

### Step 2: Comparing $f_2(n)$ with Polynomials

First, compare  $f_2(n) = 2^{\sqrt{\log n}}$  with polynomial functions.

Taking base-2 logarithms:

$$\log_2 (2^{\sqrt{\log_2 n}}) = \sqrt{\log_2 n}, \quad \log_2(n^x) = x \log_2 n.$$

As  $n$  grows to infinity, the polynomial function  $c \log_2 n$  dominates  $\sqrt{\log_2 n}$ , meaning:

$$2^{\sqrt{\log n}} \ll n^x.$$

Since both  $f_1(n) = n^{5/3}(\log n)^2$  and  $f_4(n) = \frac{n^2}{\log n}$  are polynomial up to logarithmic factors, we conclude:

$$f_2(n) < f_1(n), \quad f_2(n) < f_4(n).$$

### Step 3: Comparing $f_1(n)$ and $f_4(n)$

Now, compare:

$$f_1(n) = n^{\frac{5}{3}} (\log n)^2, \quad f_4(n) = \frac{n^2}{\log n}.$$

We examine their ratio:

$$\frac{f_1(n)}{f_4(n)} = \frac{n^{\frac{5}{3}} (\log n)^2}{\frac{n^2}{\log n}} = \frac{n^{\frac{5}{3}} (\log n)^3}{n^2} = \frac{(\log n)^3}{n^{\frac{1}{3}}}.$$

Since  $n^{1/3}$  grows faster than  $(\log n)^3$  as  $n \rightarrow \infty$ , we get:

$$\frac{(\log n)^3}{n^{1/3}} \rightarrow 0.$$

Thus,  $f_1(n) \ll f_4(n)$ , meaning:

$$n^{\frac{5}{3}}(\log n)^2 \ll \frac{n^2}{\log n}.$$

So far, the order is:

$$f_2(n) < f_1(n) < f_4(n).$$

#### Step 4: Comparing the Exponentials $f_5(n)$ vs. $f_3(n)$

We compare:

$$f_3(n) = \sqrt{n^n} = n^{\frac{n}{2}}, \quad f_5(n) = 2^n.$$

Rewriting them in exponential form:

$$f_3(n) = \exp\left(\frac{n}{2} \ln n\right), \quad f_5(n) = \exp(n \ln 2).$$

We compare the growth rates of their exponents:

$$\frac{n}{2} \ln n \quad \text{vs.} \quad n \ln 2.$$

Since  $\ln n$  grows without bound, it eventually surpasses the constant  $2 \ln 2$ . Therefore,

$$\frac{n}{2} \ln n \gg n \ln 2.$$

which implies:

$$n^{n/2} \gg 2^n.$$

Thus,  $f_3(n) \gg f_5(n)$ , meaning  $f_3(n)$  grows the fastest.

#### Step 5: Final Ordering

Combining all comparisons, the increasing order of asymptotic growth is:

$$\boxed{2^{\sqrt{\log n}} < n^{\frac{5}{3}}(\log n)^2 < \frac{n^2}{\log n} < 2^n < \sqrt{n^n}.$$

### Problem 3 (10 pts)

We say that  $T(n)$  is  $O(f(n))$  if there exist  $c$  and  $n_0$  such that for all  $n > n_0$ ,  $T(n) \leq cf(n)$ . Use this definition for parts *a* and *b*.

1. Prove that  $4n^2 + 3n\log n + 6n + 20\log^2 n + 11$  is  $O(n^2)$ . (You may use, without proof, the fact that  $\log n < n$  for  $n \geq 1$ .)
2. Suppose that  $f(n)$  is  $O(r(n))$  and  $g(n)$  is  $O(s(n))$ . Let  $h(n) = f(n)g(n)$  and  $t(n) = r(n)s(n)$ . Prove that  $h(n)$  is  $O(t(n))$ .

### Problem 4 (10 pts)

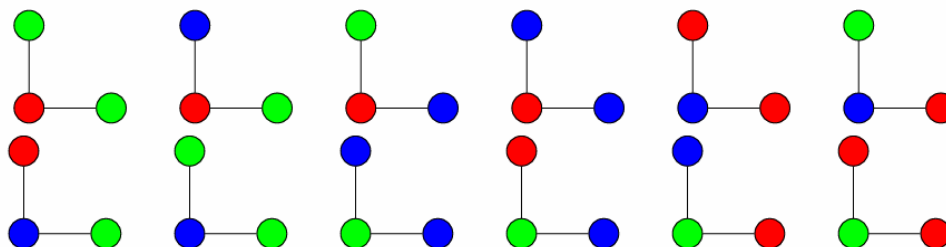
The *diameter* of an undirected graph is the maximum distance between any pair of vertices. If a graph is not connected, its diameter is infinite. Let  $G$  be an  $n$  node undirected graph, where  $n$  is even. Suppose that every vertex has degree at least  $n/2$ . Prove that  $G$  has diameter at most 2.

*Hint:* Proof by contradiction

### Problem 5 (10 pts)

Show that there are at least  $3 \cdot 2^{n-1}$  ways to properly color vertices of a tree  $T$  with  $n$  vertices using 3 colors, i.e., to color vertices with three colors such that any two adjacent vertices have distinct colors. Note that it can be shown that there are exactly  $3 \cdot 2^{n-1}$  ways to properly color vertices of  $T$  with 3 colors but in this problem, to receive full credit, it is enough prove the “at least” part.

For example, there are (at least)  $3 \cdot 2^2 = 12$  ways to color a tree with 3 vertices as show below:



### Problem 6 (Extra Credit: 10 pts)

Given a directed graph  $G$  with  $n$  vertices  $V = \{1, 2, \dots, n\}$  and  $m$  edges. We say that a vertex  $j$  is reachable from  $i$  if there is a directed path from  $i$  to  $j$ . Design an  $O(m + n)$ -time algorithm (show the pseudo-code) that for any vertex  $i$  outputs the smallest label reachable from  $i$ . For example, given the following graph you should output 1,2,2,2,1 corresponding to the smallest indices reachable from vertices 1,2,3,4,5 respectively.

