(1) FIND THE MAXIMUM VALUE IN ARRAY V
(V CONTAINS n INTEGERS)

```
DEF FIND_MAX (V)
    M = V[0]
    FOR i IN RANGE(1, LEN(V))
        IF V[i] > M:
            M = V[i]
    RETURN M
```

$O(n)$

(2) SEARCH FOR A VALUE X IN V

```
DEF FIND (V, x)
    FOR y IN V:
        IF y == x:
            RETURN TRUE
    RETURN FALSE
```

$O(n)$

(3) PROBLEM: I GIVE YOU AN ARRAY V OF SIZE n CONTAINING ALL THE INTEGERS FROM 0 TO n WITH THE EXCEPTION OF ONE. FIND THE EXCEPTIONAL INTEGER.

$V = [0, 2, 3]$ 0, 1, 2, 3

$V = [2, 1, 3]$

```
FOR i IN RANGE(n+1):
    IF NOT FIND(V, i):      O(n)
        RETURN i
```

$O(n^2)$

```
DEF FIND_MISSING(V):        // |V| = n
                            // LEN(V) = n
    S = 0
    FOR i IN RANGE(n+1):
        S = S + i
    // S = Σ_{i=0}^{n} i
    T = 0
    FOR x IN V:
        T = T + x
    // T = Σ_{x∈V} x
    RETURN S - T            // S-T = Σ_{i∈{0,1,...,n}} i - Σ_{x∈V} x
```

$O(n)$

- SORTING AN ARRAY
  INPUT: AN ARRAY OF LENGTH n
  SORT THE ARRAY (INCREASINGLY) $O(n \log n)$

- FINDING AN ELEMENT IN A SORTED ARRAY
  (BINARY SEARCH SOLVES THIS PROBLEM IN $O(\log n)$)

- I GIVE YOU AN ARRAY V OF n INTEGERS AND AN INTEGER x.
  DO THERE EXIST i, j SUCH THAT $V[i] + V[j] == x$?

```
FOR i IN RANGE(n):
    FOR j IN RANGE(n):
        IF V[i] + V[j] == x:
            RETURN TRUE
RETURN FALSE
```

$O(n^2)$

```
FOR i IN RANGE(n):
    // DOES THERE EXIST j S.T. V[j] == X - V[i]
    IF BS_FIND(V, x - V[i]):      O(log n)     n = |V|
        RETURN TRUE
RETURN FALSE
```

$O(n \log n)$

"ALGORITHM DESIGN" (KLEINBERG & TARDÖS)

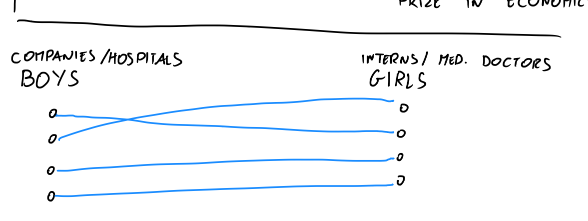
- MATHEMATICAL PROOFS
  - RUNTIME BOUNDS
  - CORRECTNESS
  - ...
- ALGORITHMIC TECHNIQUES
  - GREEDY
  - DYNAMIC PROGRAMMING
  - DIVIDE-ET-IMPERA
  - ...

STABLE MATCHINGS

- GALE-SHAPLEY WON THE 2012 NOBEL PRIZE IN ECONOMICS

COMPANIES/HOSPITALS
BOYS

INTERNS/MED. DOCTORS
GIRLS

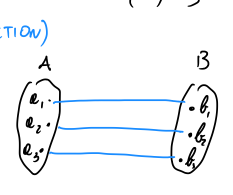


DEF: LET A AND B BE TWO DISJOINT SETS OF CARDINALITY n EACH ($|A| = |B| = n$).

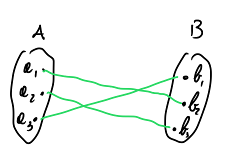
A PERFECT MATCHING BETWEEN A AND B IS A PAIRING OF THE ELEMENTS OF A WITH THE ELEMENTS OF B.

THAT IS, $M \subseteq \{\{a, b\} \mid a \in A \text{ AND } b \in B\}$ IS A PERFECT MATCHING IF AND ONLY IF $\forall a \in A$ THERE EXISTS A UNIQUE $b \in B$ SUCH THAT $\{a, b\} \in M$. (AND VICEVERSA).

(BIJECTION)



$M = \{\{a_1, b_1\}, \{a_2, b_2\}, \{a_3, b_3\}\}$

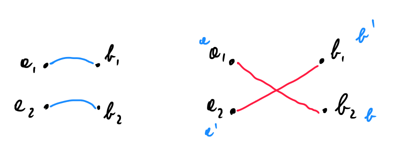


$M = \{\{a_1, b_2\}, \{a_2, b_3\}, \{a_3, b_1\}\}$

STABLE MATCHING

LET A, B BE TWO DISJOINT SETS, WITH $|A| = |B| = n$.

EACH $a \in A$ HAS A PREFERENCE LIST OVER ALL THE $b \in B$; LIKEWISE, EACH $b \in B$ HAS A PREFERENCE LIST OVER ALL THE $a \in A$.



$a_1 : b_1 > b_2$ $\quad$ $b_1 : a_1 > a_2$

$a_2 : b_1 > b_2$ $\quad$ $b_2 : a_2 > a_1$

LET M BE A PERFECT MATCHING BETWEEN A AND B.

TWO DISTINCT PAIRS $\{a, b\}, \{a', b'\} \in M$ EXHIBIT AN INSTABILITY IF

- a PREFERS b' TO b, AND
- b' PREFERS a TO a'.

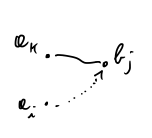
M IS A STABLE MATCHING IF IT CONTAINS NO INSTABILITIES.

QUESTIONS:

1. GIVEN THE PREFERENCE LISTS, DOES THERE EXIST A STABLE MATCHING?
2. IF SO, HOW CAN WE COMPUTE IT?

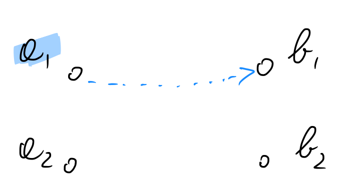
GALE-SHAPLEY()

- INITIALLY, EACH $a_i \in A$ AND EACH $b_j \in B$ IS FREE
- WHILE THERE EXISTS A FREE $a_i \in A$ THAT HAS NOT YET PROPOSED TO ALL $b_j \in B$:
  - LET a_i BE A FREE ITEM OF A.
  - LET $B' \subseteq B$ BE THE SET OF THE b_j'S THAT a_i HAS NOT YET PROPOSED TO.
  - LET $b_j \in B'$ BE THE ELEMENT OF B' THAT RANKS HIGHEST IN a_i'S PREFER. LIST
  - IF b_j IS FREE:
    - MATCH UP a_i AND b_j (WHO ARE THEN NOT FREE ANYMORE)
  - ELSE:
    - LET a_k BE THE CURRENT PARTNER OF b_j
    - IF b_j PREFERS a_k TO a_i:
      - NOTHING CHANGES (a_i REMAINS FREE, AND $\{b_j, a_k\}$ REMAIN TOGETHER)
    - ELSE:
      - "BREAK UP" $\{b_j, a_k\}$
      - "MATCH" $\{b_j, a_i\}$
      - a_k BECOMES FREE.



$a_1 : b_1 > b_2$ $\quad$ $b_1 : a_2 > a_1$

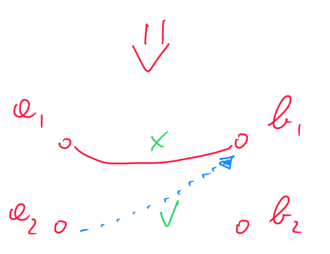
$a_2 : b_1 > b_2$ $\quad$ $b_2 : a_1 > a_2$



$a_i = a_1$ $\quad$ $B' = \{b_1, b_2\}$

$b_j = b_1$

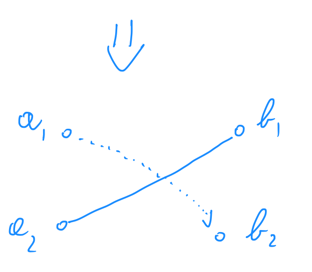
$\Downarrow$



$a_i = a_2$ $\quad$ $B' = \{b_1, b_2\}$

$b_j = b_1$

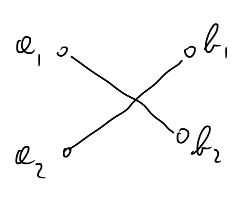
$\Downarrow$



$a_i = a_1$ $\quad$ $B' = \{b_2\}$

$b_j = b_2$

$\Downarrow$



$a_1 : b_1 > b_2$ $\quad$ $b_1 : a_2 > a_1$

$a_2 : b_1 > b_2$ $\quad$ $b_2 : a_1 > a_2$

$a_1 : b_1 > b_2$ $\quad$ $b_1 : a_2 > a_1$

$a_2 : b_2 > b_1$ $\quad$ $b_2 : a_1 > a_2$

TRY THIS INPUT BOTH FROM THE A'S, AND THE B'S, SIDES

L1: EACH $b \in B$ REMAINS MATCHED FROM THE 1ST TIME SHE GETS A PROPOSAL, UNTIL THE END OF THE EXECUTION.
ALSO, THE PARTNERS OF b GET BETTER (FROM b'S PERSPECTIVE) OVER TIME.

P: WHEN b GETS HER FIRST PROPOSAL, SHE ACCEPTS IT.
LATER, SHE MIGHT GET OTHER PROPOSALS:

- IF SHE ACCEPTS ONE SUCH PROPOSAL, SHE'LL REMAIN MATCHED AND SHE WILL LIKE HER NEW PARTNER BETTER THAN THE PREVIOUS ONE;
- IF SHE REJECTS A PROPOSAL, SHE KEEPS HER CURRENT PARTNER. □

L2: FOR EACH $a \in A$, THE SEQUENCE OF PROPOSALS OF a GET WORSE OVER TIME.

P: TRIVIAL (BY THE ALG.'S DEFINITION).

THM 1: THE ALGORITHM ENDS AFTER AT MOST n^2 ITERATIONS OF THE WHILE LOOP.

P: BY L2, IT IS IMPOSSIBLE FOR AN a_i TO PROPOSE MORE THAN ONCE TO THE SAME b_j.
THEREFORE $\forall a \in A$ $\forall b \in B$ a'S PROPOSAL TO b CAN BE MADE AT MOST ONCE.
THUS, THERE CAN BE AT MOST $|A| \cdot |B| = n^2$ PROPOSALS. GIVEN THAT EACH ITERATION RESULTS IN A PROPOSAL, THERE CAN BE AT MOST n^2 ITERATIONS.