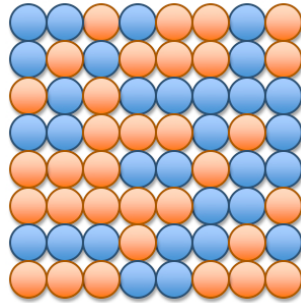




UNIT 14A

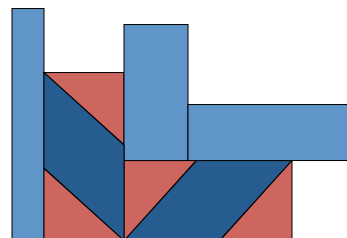
The Limits of Computing: Intractability

15110 Principles of Computing, Carnegie
Mellon University - CORTINA

1

Decision Problems

- A specific set of computations are classified as decision problems.
- An algorithm describes a **decision problem** if its output is simply YES or NO, depending on whether a certain property holds for its input.
- Example:
Given a set of N shapes,
can these shapes be
arranged into a rectangle?

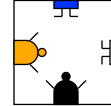


15110 Principles of Computing, Carnegie
Mellon University - CORTINA

2

The Monkey Puzzle

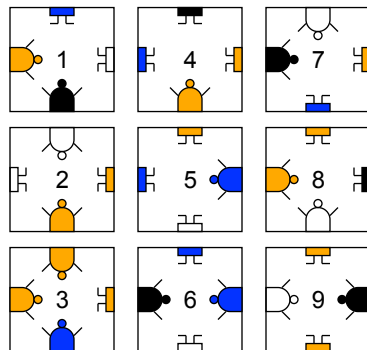
- **Given:**
 - A set of N square cards whose sides are imprinted with the upper and lower halves of colored monkeys.
 - N is a square number, such that $N = M^2$.
 - Cards cannot be rotated.
- **Problem:**
 - Determine if an arrangement of the N cards in an $M \times M$ grid exists such that each adjacent pair of cards display the upper and lower half of a monkey of the same color.



15110 Principles of Computing, Carnegie Mellon University - CORTINA

3

Example



15110 Principles of Computing, Carnegie Mellon University - CORTINA

4

Algorithm

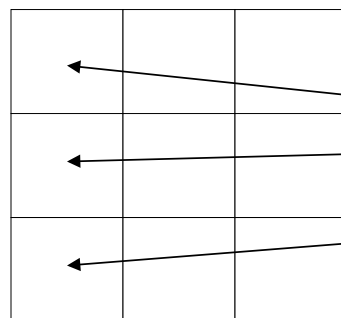
Simple brute-force algorithm:

- Pick one card for each cell of $M \times M$ grid.
- Verify if each pair of touching edges make a full monkey of the same color.
- If not, try another arrangement until a solution is found or all possible arrangements are checked.
- Answer "YES" if a solution is found. Otherwise, answer "NO" if all arrangements are analyzed and no solution is found.

15110 Principles of Computing, Carnegie Mellon University - CORTINA

5

Analysis



If there are $N = 9$ cards ($M = 3$):

To fill the first cell, we have 9 card choices.

To fill the second cell, we have 8 card choices left.

To fill the third cell, we have 7 card choices remaining.

etc.

The total number of unique arrangements for $N = 9$ cards (without rotation) is $9!$

15110 Principles of Computing, Carnegie Mellon University - CORTINA

6

Analysis (cont' d)

For N cards, the number of arrangements to examine is $N!$ (N factorial)

If we can analyze one arrangement in a microsecond:

<u>N</u>	<u>Time to analyze all arrangements</u>
9	$362,880 \mu s = 0.37 \text{ seconds}$
16	$20,922,789,888,000 \mu s = 242 \text{ days}$
25	$15,511,210,043,330,985,984,000,000 \mu s$ $= 492 \text{ billion years}$

Classifications

- Algorithms that are $O(N^k)$ for some fixed k are **polynomial-time** algorithms.
 - $O(1)$, $O(\log N)$, $O(N)$, $O(N \log N)$, $O(N^2)$
 - reasonable, **tractable**
- All other algorithms are **super-polynomial-time** algorithms.
 - $O(2^N)$, $O(3^N)$, $O(N!)$
 - unreasonable, **intractable**

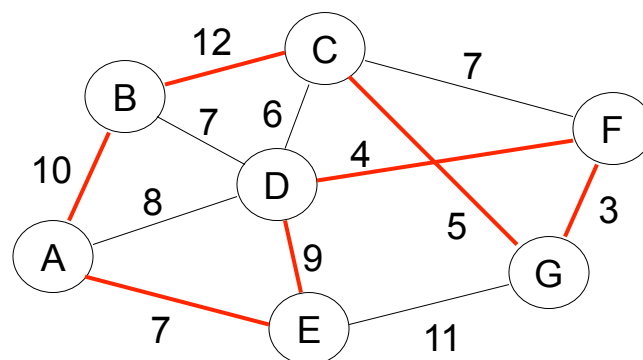
Traveling Salesperson

- Given: a weighted graph of nodes representing cities and edges representing flight paths (weights represent cost)
- Is there a route that takes the salesperson through every city and back to the starting city with cost no more than K?
 - The salesperson can visit a city only once (except for the start and end of the trip).

15110 Principles of Computing, Carnegie Mellon University - CORTINA

9

Traveling Salesperson



Is there a route with cost at most 52?
Is there a route with cost at most 48?

YES (Route above costs 50.)
YES? NO?

15110 Principles of Computing, Carnegie Mellon University - CORTINA

10

Analysis

- If there are N cities, what is the maximum number of routes that we might need to compute?
- Worst-case: There is a flight available between every pair of cities. (In our example, every city would have 6 flights available.)
- Compute cost of every possible route.
 - Pick a starting city (any city will do)
 - Pick the next city from $N-1$ choices remaining
 - Pick the next city from $N-2$ choices remaining
 - ...
- Maximum number of routes: $(N-1)! = O(N!)$

15110 Principles of Computing, Carnegie
Mellon University - CORTINA

11

Map Coloring

- Given a map of N territories, can the map be colored using K colors such that no two adjacent territories are colored with the same color?
- $K=4$: Answer is always yes.
- $K=2$: Only if the map contains no point that is the junction of an odd number of territories.

15110 Principles of Computing, Carnegie
Mellon University - CORTINA

12

Map Coloring

- Given a map of N territories, can the map be colored using **3** colors such that no two adjacent territories are colored with the same color?



15110 Principles of Computing, Carnegie
Mellon University - CORTINA

13

Analysis

- Given a map of N territories, can the map be colored using **3** colors such that no two adjacent territories are colored with the same color?
 - Pick a color for territory 1 (3 choices)
 - Pick a color for territory 2 (3 choices)
 - ...
- There are 3^{48} possible colorings to examine.
- In general, for N territories, the number of colorings to check is $3^N = O(3^N)$

15110 Principles of Computing, Carnegie
Mellon University - CORTINA

14

Satisfiability

- Given a Boolean formula with N variables using the operators AND, OR and NOT:
 - Is there an assignment of boolean values for the variables so that the formula is true (satisfied)?
Example: $(A \text{ AND } B) \text{ OR } (\text{NOT } C \text{ AND } A)$
 - Truth assignment: $A = \text{True}, B = \text{True}, C = \text{False}$.
- How many assignments do we need to check for N variables? (Think: how big is the truth table?)
 - Each symbol has 2 possibilities ...
There are $2^N = O(2^N)$ possible assignments.

15110 Principles of Computing, Carnegie Mellon University - CORTINA

15

The Big Picture

- Intractable problems are solvable if the amount of data (N) that we're processing is small.
- But if N is not small, then the amount of computation grows exponentially and the solutions quickly become intractable (i.e. out of our reach).
- Computers can solve these problems if N is not small, but it will take far too long for the result to be generated.
 - We would be long dead before the result is computed.



15110 Principles of Computing, Carnegie Mellon University - CORTINA

16

What's Next

- For a specific decision problem, is there single tractable (polynomial-time) algorithm to solve any instance of this problem?
- If one existed, can we use it to solve other decision problems?
- What is one of the big computational questions to be answered in the 21st century?