Review
Offline vs. Online Algorithms

- Traditional algorithms are ‘offline’ in the sense that they have the whole input in their hand.
- Online algorithms, in contrast, do not have/need the whole input in order to solve a problem
  - The online algorithms often take *irrevocable decisions* to process the input.
In a Glance . . .

- Online algorithms are
  - Practical
  - Diverse
  - Fun (really!)
- Let’s ‘play’ with online algorithms and enjoy
Logistics

- Office hours: 11:30 am-12:30 pm, Tuesdays and Thursdays, in E2586 or by appointment
Cow-Path Problem
Problem Definition

- A cow faces a fence, infinite in both directions
- She wants to find a hole in order to get to the green pasture on the other side
- The cows **online strategy** specifies the path traveled in search of the hole.
- The goal is to minimize the distance traveled.
Offline Strategy

Let $u$ an integer indicating the distance between the initial location of the cow and the location of the hole.

- $u$ is unknown to the cow!
Offline Strategy

- Let $u$ an integer indicating the distance between the initial location of the cow and the location of the hole.
  - $u$ is unknown to the cow!
- An optimal offline algorithm $Opt$ (i.e., a cow which knows the location of the hole), incurs a cost of $u$
**Smart-Cow Algorithm (SCA)**

- Gradually extend the explored interval of the fence
- Alternate between left and right!
  - Go right for distance $d_0$
  - Go back to the origin, left for distance $d_1$
  - Go back to the origin, right for distance $d_2$
  - Continue accordingly for $d_3, \ldots, d_k$ until the hole is found.
Competitive Ratio of SCA

- Recall that the competitive ratio of an online algorithm is the maximum ratio between the cost of that algorithm and an optimal offline \( \text{OPT} \) algorithm \( \text{OPT} \)
- The cost of \( \text{OPT} \) is \( u \)
- The cost of SCA is \( 2d_0 + 2d_1 + \ldots + 2d_{k-2} + 2d_{k-1} + u \)
  - \( d_{k-2} < u \leq d_k \)
Competitive Ratio of SCA (cntd.)

- The competitive ratio would be

\[
\frac{2d_0 + 2d_1 + \ldots + 2d_{k-2} + 2d_{k-1} + u}{u} = 1 + 2 \frac{d_0 + d_1 + \ldots + d_{k-1}}{u}
\]

- what is the value of \(u\) in the worst case?
  - If you are an adversary and want to fail the algorithm, where you place the hole?
The competitive ratio would be

\[ \frac{2d_0 + 2d_1 + \ldots + 2d_{k-2} + 2d_{k-1} + u}{u} = 1 + 2\frac{d_0 + d_1 + \ldots + d_{k-1}}{u} \]

In the worst case, \( u = d_{k-2} + \epsilon \).

- Just a bit more than the previous probe!
- So, the competitive ratio of a Smart-Cow algorithm is

\[ 1 + 2\frac{d_0 + d_1 + \ldots + d_{k-1}}{d_{k-2} + \epsilon} \]
The Doubling Technique

- Assume $d_i = 2^i$, i.e., first go one unit to the right, go back to the origin, go two units to the left, back to origin, four units to the right, etc.

- We will have
  
  $$d_0 + d_1 + \ldots + d_{k-1} = 1 + 2 + 4 + \ldots + 2^{k-1} = 2^k - 1 = 4 \cdot 2^{k-2}.$$  

- The competitive ratio would be
  
  $$1 + 2 \frac{d_0 + d_1 + \ldots + d_{k-1}}{d_{k-2} + \epsilon} = 1 + 2 \frac{4 \cdot 2^{k-2}}{2^{k-2} + \epsilon} \approx 9.$$
Theorem

The smart-cow algorithm with steps that double (i.e., \( d_i = 2^i \)) has a competitive ratio of at most 9.

- It turns out that no deterministic algorithm can achieve a ratio better than 9.
  - The proof is a bit involved and we skip it here.
- So, the doubling technique results an optimal algorithm in this case.
Search Problems under Uncertainty

- A cow can be a robot (or the other way around)!
- In practice, robots often do not have full information about their environment.
- Cow-path problem and its variant are a way to model many types of search problems.
Variants of Search Problems

- Path-cow problem is an online search problem on a path.
- Consider a star, where \( w \) paths have one common endpoint.
- Assume a robot is initially located at the common point, and needs to find a target located in an unknown position.
- What is a good algorithm?
Variants of Search Problems

- The best strategy is to have 
  \[ d_i = \left(\frac{w}{w-1}\right)^i. \]
  - For \( w = 2 \), it requires doubling.
  - For \( w = 3 \), we jump by a factor of \( \frac{3}{2} \), and so on.

- The competitive ratio will be at most 
  \[ 1 + 2 \frac{w^w}{(w-1)^{w-1}} \approx 1 + 2e(w - 1) \] (when \( w \) is sufficiently large).
  - \( e \approx 2.71 \) is the Euler’s constant

- Note that doubling is not optimal here.
  - But it is still competitive, i.e., it has a constant competitive ratio.
Randomized Online Algorithms
Randomized Online Algorithms

- Randomization often helps online algorithms to achieve better competitive ratios.

- In competitive analysis of online algorithms, we consider worst-case inputs generated by an adversary.

- For randomized algorithms, we compare online algorithms against an oblivious adversary which is unaware of random choices made by the algorithms.

  - The adversary knows the code of the algorithm but does not know the run-time random bits used by the algorithm.
Randomized Smart-Cow Algorithms

- There are two mirroring algorithms for the deterministic algorithm.
- The worst-case position of the hole is not the same for these two algorithms.

- Randomized-Smart-Cow: chose one of the mirroring algorithm uniformly at random.
Analysis of Randomized Smart-Cow

- In order to find the expected cost of the algorithm, we find the summation of costs of the two mirroring algorithms.
  - w.l.o.g. assume the hole is located on the right
  - The total sum on the left of the origin is
    \[ 2(1 + 2 + \ldots + 2^{k-1}) = 2(2^k - 1). \]
  - The total sum on the right of the origin is
    \[ 2(1 + 2 + \ldots + 2^{k-2}) + 2u = 2(2^{k-1} - 1 + u). \]
  - The expected cost of the algorithm is \( 2^k - 1 + 2^{k-1} - 1 + u. \)

- The competitive ratio is at most \( \frac{2^k + 2^{k-1} + u - 2}{u} = 1 + \frac{2^k + 2^{k-1} - 2}{u}. \)

- In the worst case, \( u = 2^{k-2} \) which gives a competitive ratio of at most 7.

![Graph showing the cost distribution for the algorithm.](image)
Review of Randomized Smart-Cow

- With only one random bit, we achieve a competitive ratio of 7, which is better than any what any deterministic algorithm can achieve.

- There is a more complicated randomized algorithm achieves a competitive ratio of 4.591.
  - Smart-Cow with different jumps.
Online Algorithms with Advice
Assume an online algorithm receives some \textit{bits of advice} from a benevolent oracle.

The advice can be anything that can help the algorithm its competitive ratio

What is a good advice for cow-path problem?
Relevant Questions with Respect to Advice

- How many bits of advice are sufficient to achieve an optimal solution?
- What is the best competitive ratio that one can achieve with $c$ bits of advice?
Search Problem on a Star with Advice

- How many bits are sufficient to achieve an optimal solution?
How many bits are sufficient to achieve an optimal solution?

- With $\Theta(\lceil \log w \rceil)$ bits, indicate the row at which the target is located.
- Indeed, at least $\lceil \log w \rceil$ bits are also required to be optimal!
Search Problem on a Star with Advice

- What can we do with \( c < \log w \) bits of advice?
  - Recall that the competitive ratio of smart-cow with \( d_i = (w/(w-1))^i \) is roughly \( 1 + 2e(w-1) \).

- Assume we can have 1 bit of advice!
  - Partition the \( w \) paths into two groups of size \( w/2 \).
  - The advice to specifies in which group the target is located.
  - The competitive ratio becomes \( 1 + 2e(w/2 - 1) \).

- With \( c \) bits, the competitive ratio will be \( 1 + 2e(w/2c - 1) \).
Variants of Online Search Problems

Potential Topics for Final Projects
Online Search on Trees and Tree-like Graphs

- Assume a robot needs to search on a tree instead of a path or a star.

- What is the competitive ratio of smart-cow with doubling jumps?
  - The tree might or might not be regular, i.e., all internal nodes have an equal $k$ neighbors.

- What about graphs which are almost trees?
  - e.g., Cactus graphs?

- How randomization and advice can help tackling these problems?
Two-dimensional Cow-Path Problem

- Assume the cow is in a 2d plane and wants to find a hole in an (unseen) fence.
Two-dimensional Cow-Path Problem

- Assume the cow is in a 2d plane and wants to find a hole in an (unseen) fence.
- Maybe we should make jumps in different directions?
- There are many variants of this problem!
Online Search with Moving Objects

- Assume the ‘cow’ is a cat and the ‘hole’ is a mouse.
  - Cat moves faster than the mouse!
  - What is the strategy for the cat to catch the mouse while moving a minimum distance?

- What if they are located on a tree or a cactus?
- What if they are limited to a fence?
- How advice/randomization helps?
Conclusions
Concluding Remarks

- Online Search problems have many practical applications.
- They can be modeled with variants of Cow-path problem.
- Doubling technique and its variants often result in competitive algorithms.
- Randomization and advice are tools that can help improve competitive ratio of online algorithms.