Maximizing Revenue for the Online Dial-A-Ride Problem

Ananya Das Christman
Middlebury College

1

Outline

- Background: What is an Online Problem?
- Analyzing Online Algorithms: *Competitive Analysis*
- Online Dial a Ride Problem
- Related Open Problems

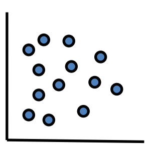
2

Classical Computer Science Problems

- Sorting

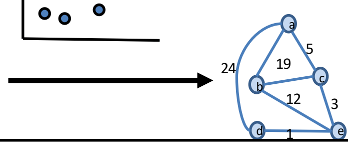
24	**19**	**5**	**12**	**3**	**1**	**14**	**82**	**80**

- Closest Points



All input known in advance – **Offline** Problems

- Shortest Paths



3

Classical Computer Science Problems

- Sorting

24								

- Closest Points
- Shortest Paths

4

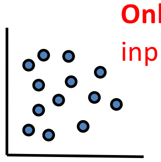
Classical Computer Science Problems

- Sorting

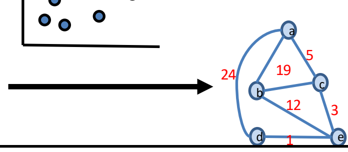
24	**19**	**5**	**12**	**3**	**1**	**14**	**82**	**80**

Online problem - input arrives over time

- Closest Points

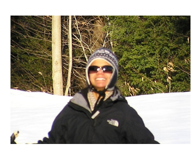


- Shortest Paths



5

Ski Rental Problem



6

Ski Rental Problem

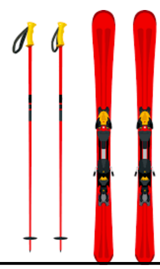
7

Ski Rental Problem

8

Ski Rental Problem

Rent or Buy?



9

Offline Ski Rental Problem

- Ski resort – cheap but owner often shuts down in the middle of the season to go to Florida…
- Cost to Rent: **$1** /day, Cost to Buy: **$10**
- Input: *d* = number of days resort will stay open
- Goal: Decide whether to Rent or Buy to achieve cheapest cost

```
Alg-Check_d(input: d):
   if (d < 10)
     Rent (Cost = d)
   else
     Buy  (Cost = 10)
```

Optimal Algorithm

10

Online Ski Rental Problem

Everyday for a ski season, you decide:

Rent – or – Buy?

Input (# days open) arrives over time

Need a “good” **Online** Algorithm

11

Outline

- Background: What is an Online Problem?
- Analyzing Online Algorithms: *Competitive Analysis*
- Online Dial a Ride Problem
- Related Open Problems

12

Optimistic Algorithm for Ski Rental

```
Optimist-Alg:
  If open on first
  day: BUY
```

Worst input?
- Resort closes on Day 2

	Day 1	Day 2	...	Day 90
Optimist-Alg (ON)	$10	X	X	X

13

Pessimistic Algorithm for Ski Rental

```
Pessimist-Alg:
  while (open):
     RENT
```

Worst input?
- Resort stays open all season

	Day 1	Day 2	...	Day 90
Pessimist-Alg (ON)	$1	$1	$1	$1

14

Cautious-Optimist for Ski Rental

```
Cautious-Optimist:

  For the first 3
  days: RENT

  If still open on
  day 4, BUY
```

Worst input?
- Resort closes on Day 5

	Day 1	Day 2	Day 3	Day 4	Day 5	...	Day 90
Cautious-Alg (ON)	$1	$1	$1	$10	X	X	X

15

How to Analyze an Online Algorithm?

- Compare its cost to the optimal offline cost, given the worst possible input.

16

How to Analyze an Online Algorithm?

- Compare its cost to the optimal offline cost, given the worst possible input.
- Compare using ratio: $\frac{Cost(ON)}{Cost(OPT)}$ competitive ratio
- If $\frac{Cost(ON)}{Cost(OPT)} \leq c$ then *ON is c-competitive*

Competitive Analysis

17

Optimist Competitive Ratio

```
Optimist-Alg:
  If open on first
  day: BUY
```

Worst input?
- Resort closes on Day 2

	Day 1	Day 2	...	Day 90	Total Cost
Optimist-Alg (ON)	$10	X	X	X	**$10**
OPT	$1	X	X	X	**$1**

$$\frac{Cost(ON)}{Cost(OPT)} = \frac{10}{1}$$

18

Pessimist Competitive Ratio

```
Pessimist-Alg:
  while (open):
    RENT
```

Worst input?
- Resort stays open all season

	Day 1	Day 2	…	Day 90	Total Cost
Pessimist-Alg (ON)	$1	$1	$1	$1	**$90**
OPT	$10	-	-	-	**$10**

$$\frac{Cost(ON)}{Cost(OPT)} = \frac{90}{10}$$

19

Cautious-Optimist Competitive Ratio

```
Cautious-Optimist:

  For the first 3
  days: RENT

  If still open on
  day 4, BUY
```

Worst input?
- Resort closes on Day 5

$$\frac{Cost(ON)}{Cost(OPT)} = \frac{13}{4} = 3.25$$

	Day 1	Day 2	Day 3	Day 4	Day 5	…	Day 90	Total Cost
Cautious-Alg (ON)	$1	$1	$1	$10	X	X	X	**$13**
OPT	$1	$1	$1	$1	X	X	X	**$4**

20

New-Cautious-Optimist Competitive Ratio

```
New-Cautious-Optimist:

  For the first 9
  days, RENT

  If open on the 10th
  day, BUY
```

Worst input?
- Resort closes on Day 11

$$\frac{Cost(ON)}{Cost(OPT)} = \frac{19}{10} = 1.9$$

	Day 1	Day 2	…	Day 9	Day 10	Day 11	…	Day 90
New-Cautious (ON)	$1	$1	$1	$1	$10	X	X	-

21

Generalized Competitive Ratio

```
Generalized-Algorithm:
  Let b = cost to buy skis
  For the first b-1 days,
     RENT

  If open on the bth day,
     BUY
```

For any b: $\frac{Cost(ON)}{Cost(OPT)} < 2$

Worst input?
- Resort closes on Day b+1

	Day 1	Day 2	…	Day b-1	Day b	Day b+1	…	Day 90	Total Cost
Generalized (ON)	$1	$1	$1	$1	$b	X	X	X	**$(b-1) + b**
OPT	$b	-	-	-	-	X	X	X	**$b**

22

Generalized Competitive Ratio

	Day 1	Day 2	…	Day b-1	Day b	Day b+1	…	Day 90	Total Cost
Generalized (ON)	$1	$1	$1	$1	$b	X	X	X	**$(b-1) + b**
OPT	$b	-	-	-	-	X	X	X	**$b**

$$Cost(ON) = (b - 1) + b = 2b - 1$$
$$Cost(OPT) = b$$

$$\frac{Cost(ON)}{Cost(OPT)} < 2 - \frac{1}{b}$$

Best ratio for any online algorithm! (no other number of rental days yields a better ratio)

ON is: $\left(2 - \frac{1}{b}\right)$ – competitive

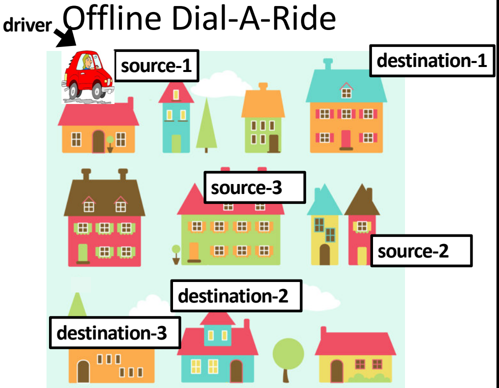
23

Outline

- Background: What is an Online Problem?
- Analyzing Online Algorithms: *Competitive Analysis*
- Online Dial a Ride Problem
- Related Open Problems

25

Offline Dial-A-Ride



[N. Wilson; Joint Automatic Control Conference, 1971]

26

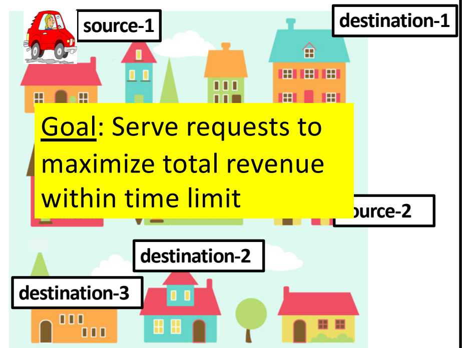
Online Dial-A-Ride

revenue1 = $10 ✓

revenue2 = $30 ✓

revenue3 = $50

Time Limit = 24



Goal: Serve requests to maximize total revenue within time limit

[N. Ascheuer, S. Krumke, and J. Rambau; STACS 2000]

27

Applications

29

OFFline vs ONline

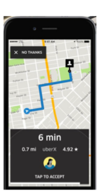
OFFline ~ "**Reservation** Service": Receives all requests in advance

Source	Destination	Time	Revenue
Oak Street	Airport	9am	$15
Maple Ave	Grocery	3pm	$10
Pine Street	Airport	12pm	$5
Elm Rd.	2nd Street	2:30pm	$50
1st Street	Main Street	6:30pm	$12

30

OFFline vs ONline

ONline ~ Uber Receives each request only when it is released

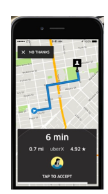


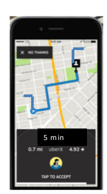
Revenue = $15

32

OFFline vs ONline

ONline ~ Uber Receives each request only when it is released



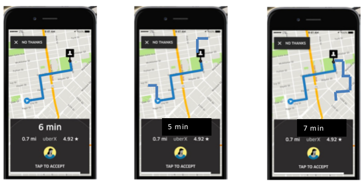


Revenue = $15 $12

33

OFFline vs ONline

ONline ~ Uber Receives each request only when it is released



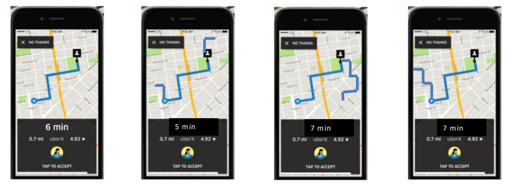
Revenue = $15 | $12 | $17

✔ Accept

34

OFFline vs ONline

ONline ~ Uber Receives each request only when it is released



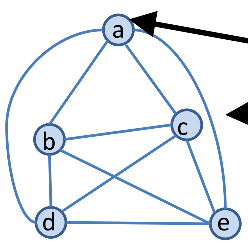
Revenue = $15 | $12 | $17 | $100

✔ Accept

35

Online-Dial-A-Ride

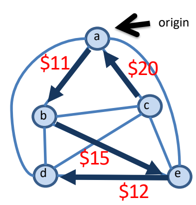
Model with a complete weighted **Graph**



36

Online-Dial-A-Ride with Revenue

Input:
- Complete Weighted Graph
- Initial Location of Server (origin)
- Requests: (source, destination, release time, revenue)
- Time Limit (T)



Time Limit T = 24

Goal:
Serve requests to **Maximize Total Revenue** within T

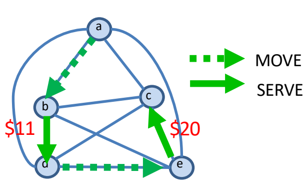
Offline version is NP-hard[1]

1. Christman, Chung, Jaczko, Westvold [ATMOS 2017]

37

Worst Input?

- Every request takes ~~a long~~ time to serve
  V
  max (where *max* is the maximum edge weight in the graph)
- After each request we serve, we have to move for ~~a long~~ time
  V
  max



Alternate between moving and serving, each for *max* time

38

Segmented-Best-Path (SBP) Algorithm[1]

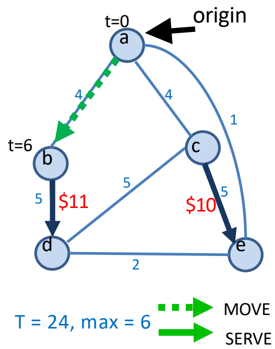
```
1. Let max be the maximum edge weight in the graph.

2. Split the total time T into T/max “segments”,
   each of length max.

3. While there is still time remaining:
   Alternate between:
      a) Find the request set with the highest
         revenue that can be served within time max;
         let S denote this set. Move to the source of
         the first request in S.
         (Wait until the end of the time segment)

      b) Serve the requests in set S.
         (Wait until the end of the time segment)
```

takes time ≤ max

1. Christman, Chung, Jaczko, Li, Westvold, Xu, Yuen [IWOCA 2020]

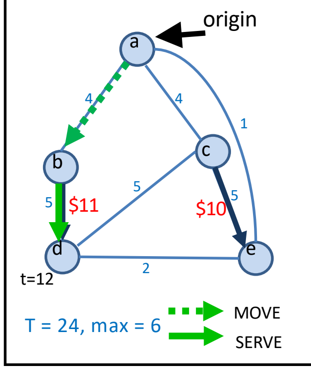
39

SBP Example



40

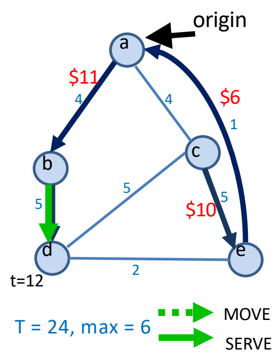
SBP Example



Revenue earned = $11

41

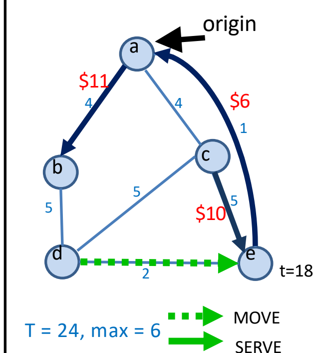
SBP Example



Revenue earned = $11

42

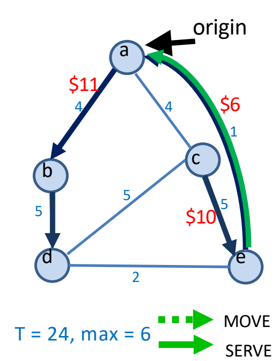
SBP Example



Revenue earned = $11

43

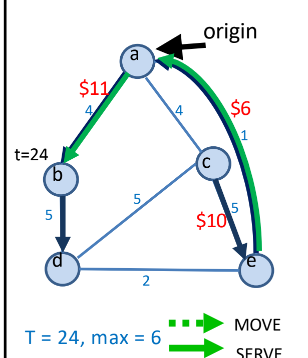
SBP Example



Revenue earned = $11 + $6

44

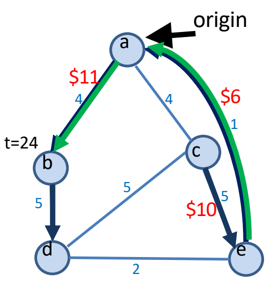
SBP Example



Revenue earned = $11 + $6 + $11
= $28

45

SBP Example

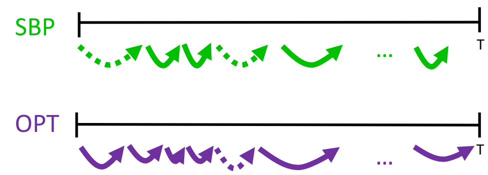


Revenue earned = $11 + $6 + $11
= $28

OPT = $11 + $6 + $11 + $10
= $38

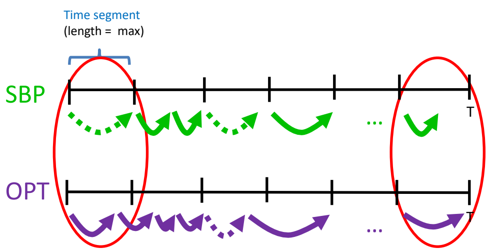
T = 24, max = 6

SBP is 1/5-Competitive Proof



SBP is 1/5-Competitive Proof

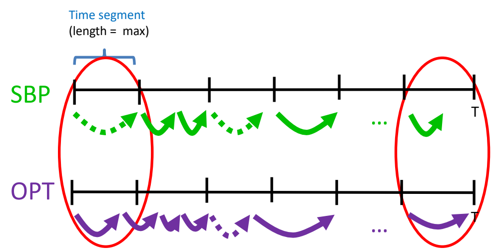
- Compare **schedules** of SBP and OPT, one segment at a time:



- Consider the segments where SBP earns less than OPT

SBP is 1/5-Competitive Proof

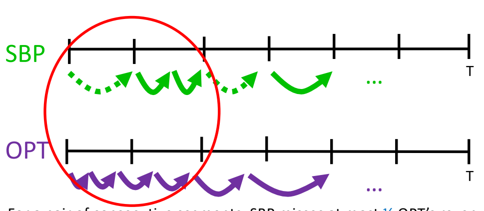
- Compare schedules of SBP and OPT, one segment at a time:



- Consider the segments where SBP earns less than OPT
- Create a modified schedule $\overline{\text{SBP}}$ (initially $\overline{\text{SBP}}$ = SBP) that will contain additional copies of requests
- For now: assume all requests served by OPT during a segment are available to SBP

SBP is 1/5-competitive

- SBP serves in only every other time segment, OPT may serve the entire time

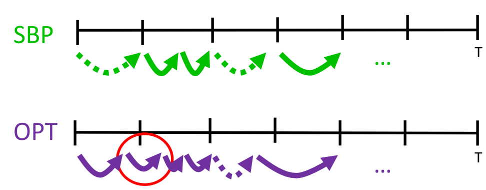


- For a pair of consecutive segments, SBP misses at most ½ OPT's revenue
- Initially: revenue($\overline{\text{SBP}}$) ≤ revenue(SBP) · 2

≤ 2 revenue(SBP)

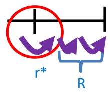
SBP is 1/5-Competitive Proof

- SBP does not serve any requests that overlap time segments



Overlapping Request

OPT



- Both r* and R can separately be served within a time segment
- So SBP will serve a set with revenue at least max(r*, R)

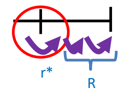
  → revenue(SBP) ≥ max(r*, R)

- So SBP misses at most ½ OPT

52

Overlapping Request

OPT



- Both r* and R can separately be served within a time segment
- So SBP will serve a set with revenue at least max(r*, R)

  → revenue(SBP) ≥ max(r*, R) ≥ ½ (r* + R) ≥ ½ revenue(OPT)

- So SBP misses at most ½ OPT

- So far: revenue($\overline{\text{SBP}}$) ≤ 2 revenue(SBP) · 2

  ≤ 4 revenue(SBP)

53

SBP is 1/5-competitive

- Recall: we assumed all requests served by OPT during a segment are available to SBP
- Need another factor of revenue(SBP) to compensate when this assumption is removed

revenue($\overline{\text{SBP}}$) ≤ 4 revenue(SBP) + revenue(SBP)

≤ 5 revenue(SBP)

54

SBP is 1/5-competitive

- Recall: we assumed all requests served by OPT during a segment are available to SBP
- Need another factor of revenue(SBP) to compensate when this assumption is removed

revenue($\overline{\text{SBP}}$) ≤ 4 revenue(SBP) + revenue(SBP)

≤ 5 revenue(SBP)

And:

revenue($\overline{\text{SBP}}$) ≥ revenue(OPT)

So:

revenue(SBP) ≥ 1/5 revenue(OPT)

55

Uniform Revenue

- Useful when all requests have equal priority

- SBP is 1/4-competitive

56

Bipartite Graphs

sources

destinations

SBP is *k* –competitive, where *k* is the ratio between the minimum and maximum edge weights in the input graph

57

Outline

- Background: What is an Online Problem?
- Analyzing Online Algorithms: *Competitive Analysis*
- Online Dial a Ride Problem
- Related Open Problems

59

Summary of Results + Open Problems

Competitive ratio of SBP for OLDARP

	Uniform Revenue	Nonuniform Revenue
Weighted General Graphs	1/4	1/5
Weighted Bipartite Graphs	⌈ *k* ⌉*	⌈ *k* ⌉*

Can we achieve better ratios?

**k* is a fraction used to bound the minimum edge weight in the input graph

60

Segmented-Best-Path (SBP) Algorithm[1]

```
1. Let max be the maximum edge weight in the graph.

2. Split
   each o

3. While there is still time remaining:
  Alternate between:
    a) Find the request set with the highest
       revenue that can be served within time max;
       let S denote this set. Move to the source of
       the first request in S.
       (Wait                                  egment)

    b) Serve the requests in set S.
       (Wait until the end of the time segment)
```

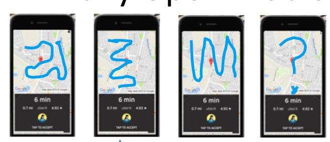
Running time is exponential in the number of available requests

Tradeoff? Efficiency vs. performance

1. Christman, Chung, Jaczko, Li, Westvold, Xu, Yuen [IWOCA 2020]

61

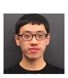
Many Open Problems…



Serve all requests while minimizing route complexity

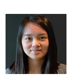
A fleet of Uber drivers

Ride-sharing



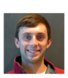
Lambus Li
Middlebury '20

Will Forcier
Lake Forest College '13

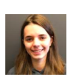


Annika Xu
Middlebury '20

Aayam Poudel '18



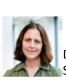
Nick Jaczko
Middlebury '19



Anna Vasilchenko '17

Scott Westvold

Thank You!



Dr. Barbara Anthony
Southwestern University

Dr. Christine Chung
Connecticut College

Dr. David Yuen

63