Online algorithm:
Efficient Optimization under Uncertainty

汇报人：肖霖畅

大纲

1. **在线问题和在线算法的简单介绍**
2. 在线算法的案例和证明方法
3. 从Worst case到ROM
4. 总结、未来研究方向和现有问题
5. 附录：Primal-Dual证明

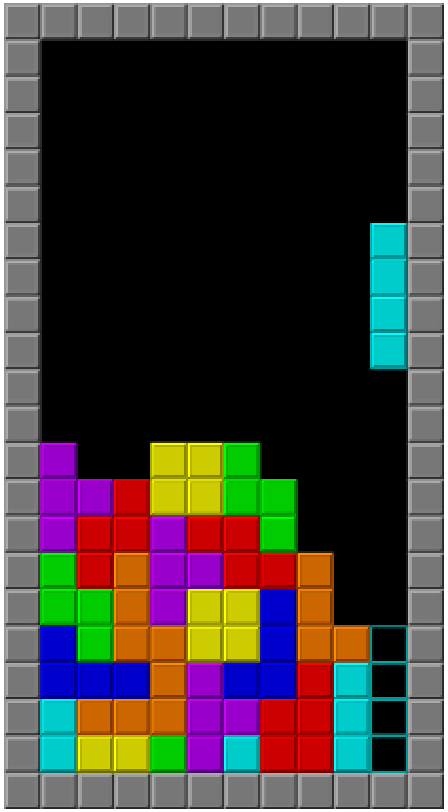
优化问题

- 优化（Optimization）：从一个可行解（feasible solutions）的集合中找到最优解（Optimal solution）
- 目标：最小化Cost 或 最大化Profit

- 现实场景中优化问题的难点：
  - Intractability：计算上的困难，比如NP问题
    - 任务调度的最优决策，用最少的颜色为图填色
  - Uncertainty：可能缺少足够的信息
    - 必须在不知道未来的输入信息条件下做决策
    - 例子：是否要读个博？是否和TA结婚
- 解决方案：在不确定性下提出一个高效的优化算法

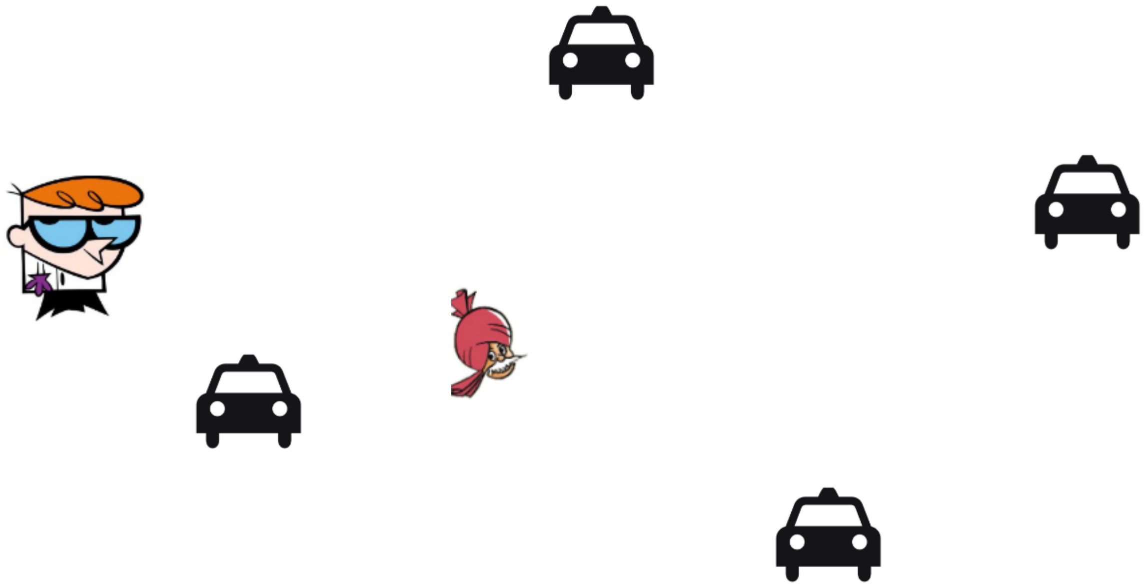
什么是在线问题

- 在线问题
  - 有一个输入序列，按时间切分成多个部分
  - 输入序列的每个部分按次序到达（即存在到达顺序）
  - 对输入序列的每部分，算法必须在缺乏未来信息的条件下做决策
  - 决策是及时的且不可撤回的



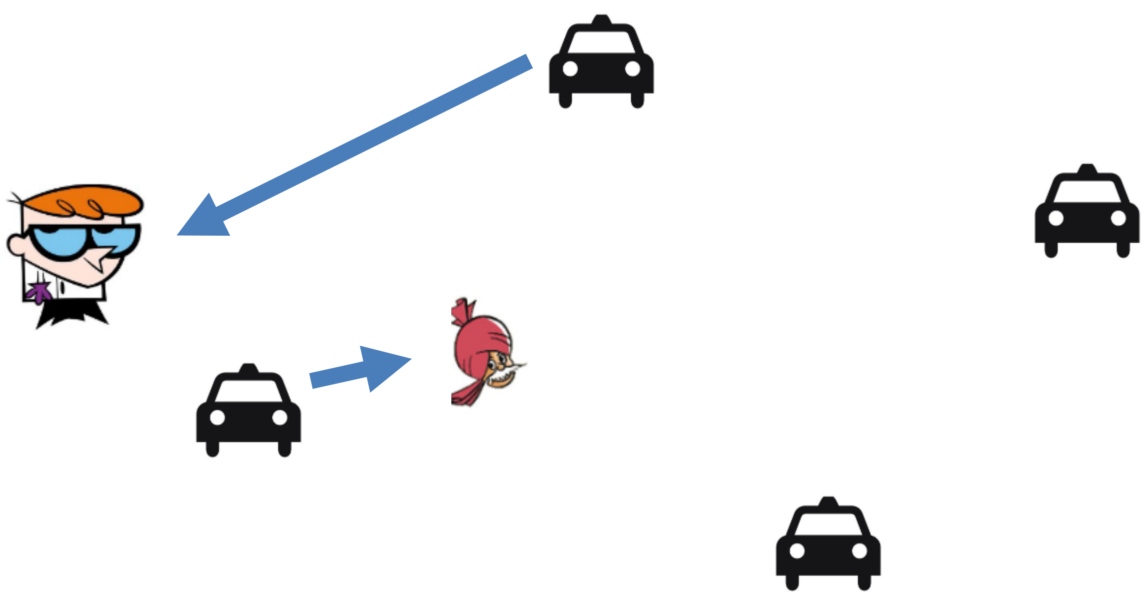
在线问题举例

- Problem 0：Taxi-dispatching (k-server)
  - 有k辆出租车
  - 每一个乘车请求在线到达
  - 目标：最小化出租车接到乘客的总行驶距离



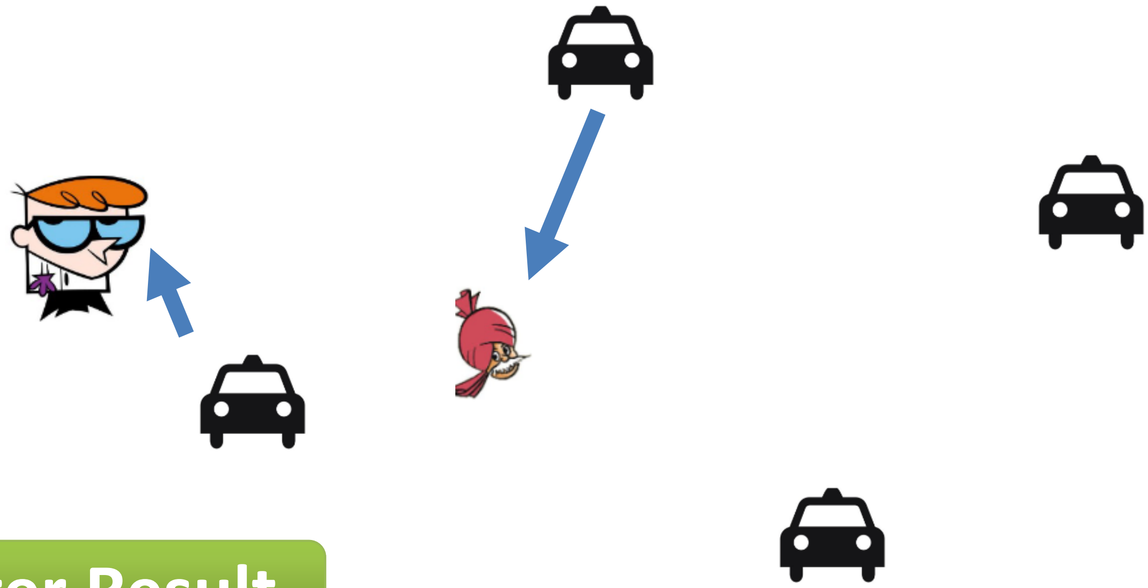
在线问题举例

- Problem 0：Taxi-dispatching (k-server)
  - 有k辆出租车
  - 每一个乘车请求在线到达
  - 目标：最小化出租车接到乘客的总行驶距离



在线问题举例

- Problem 0：Taxi-dispatching (k-server)
  - 有k辆出租车
  - 每一个乘车请求在线到达
  - 目标：最小化出租车接到乘客的总行驶距离
  - 直观的结论：处理输入序列的到达顺序是在线算法的关键



Better Result

为什么研究在线算法?

- 大多数实际问题本质是在线的，而且需要被快速解决
- 相比启发式算法，能够提供严谨的数学分析
- 可以与近似算法、凸优化、信息论、博弈论等领域结合获得更优的算法
- 为离散优化问题提供一个计算困难度的度量标准(竞争比)

如何评价在线算法A的性能

- 竞争比（Competitive Ratio）
  - I 代表任何可能的在线输入序列
  - OPT(I)：针对输入序列的离线最优解
  - 针对最小化优化问题，如果对任何的输入，都有

$$\mathrm{OPT}(I) \le \mathrm{A}(I) \le \alpha \cdot \mathrm{OPT}(I)$$

则算法A是一个α-competitive算法 (α>1) => $\alpha = \max_I \frac{A(I)}{OPT(I)}$

  - 针对最大化优化问题，如果对任何的输入，都有

$$\mathrm{A}(I) \le \mathrm{OPT}(I) \le \alpha \cdot \mathrm{A}(I)$$

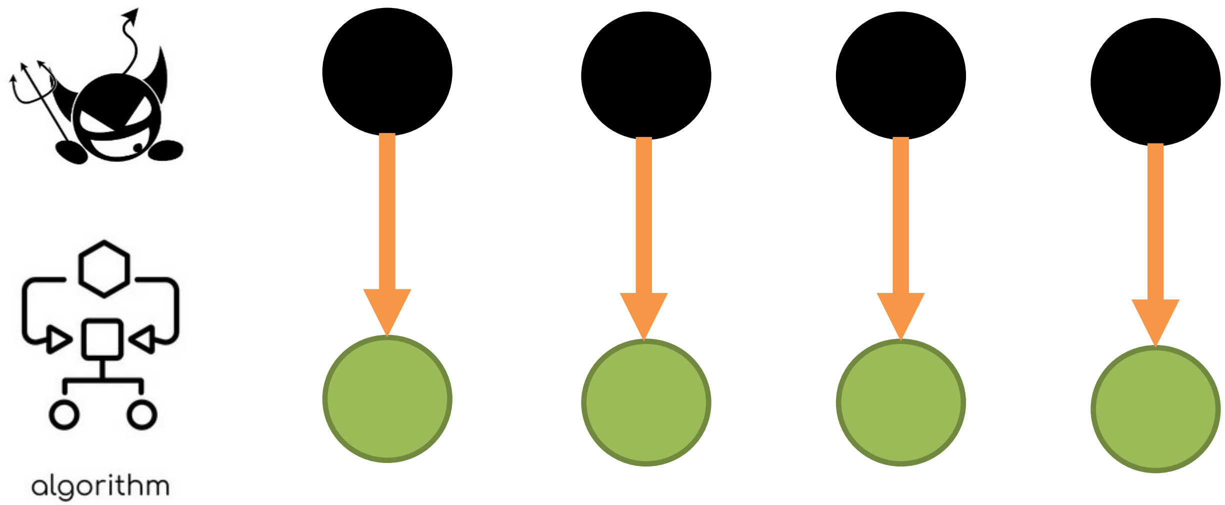
则算法A是一个α-competitive算法 (α>1) => $\alpha = \max_I \frac{A(I)}{OPT(I)}$

- 确定性算法 A(I) vs. 随机性算法 $Exp[\mathrm{A}(I)]$
- 证明困难：需要遍历所有的可能输入

Worst Case Model

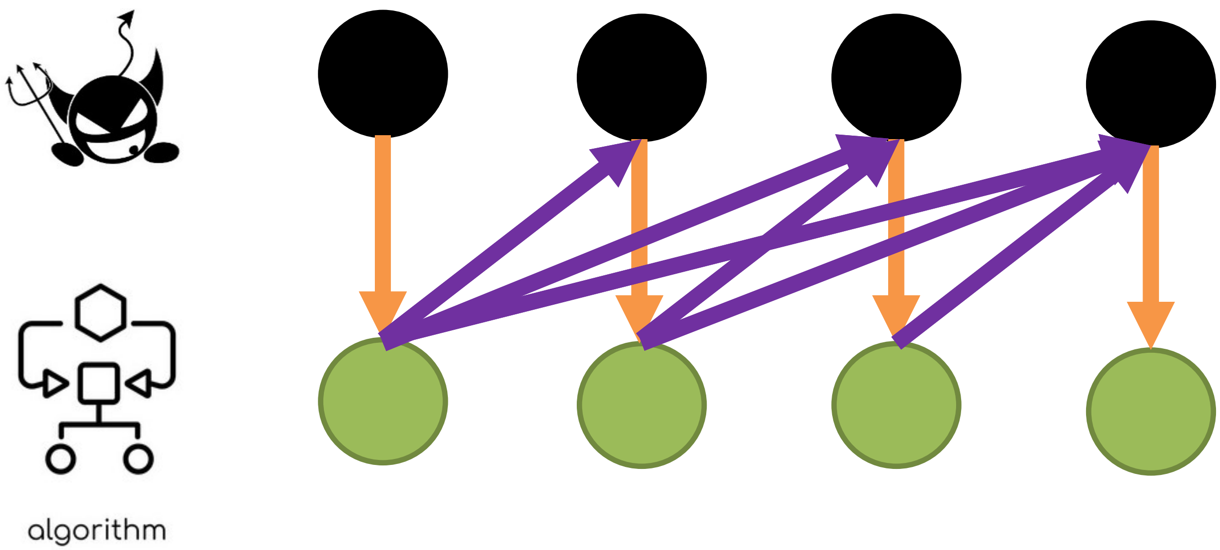
- 竞争比分析的困难
  - 需要遍历所有的可能输入： $\alpha = \max_{I} \frac{A(I)}{OPT(I)}$ 和 $\alpha = \max_{I} \frac{Exp[A(I)]}{OPT(I)}$
- Worst Case Model
  - 对手：根据决策者的在线算法确定最坏输入序列 I，并计算已知最坏输入序列下的离线最优解 OPT(I)
  - 决策者：根据在线算法在最坏输入序列中执行并得到算法解 A(I)
  - 通过算法解A(I) 和 离线最优解OPT(I) 分析竞争比
- 对手能力不同，最坏输入序列不同
  - Oblivious Adversary（短视对手）
  - Adaptive online Adversary （自适应在线对手）
  - Adaptive offline Adversary（自适应离线对手）

Worst Case Model中的对手(Adversary)



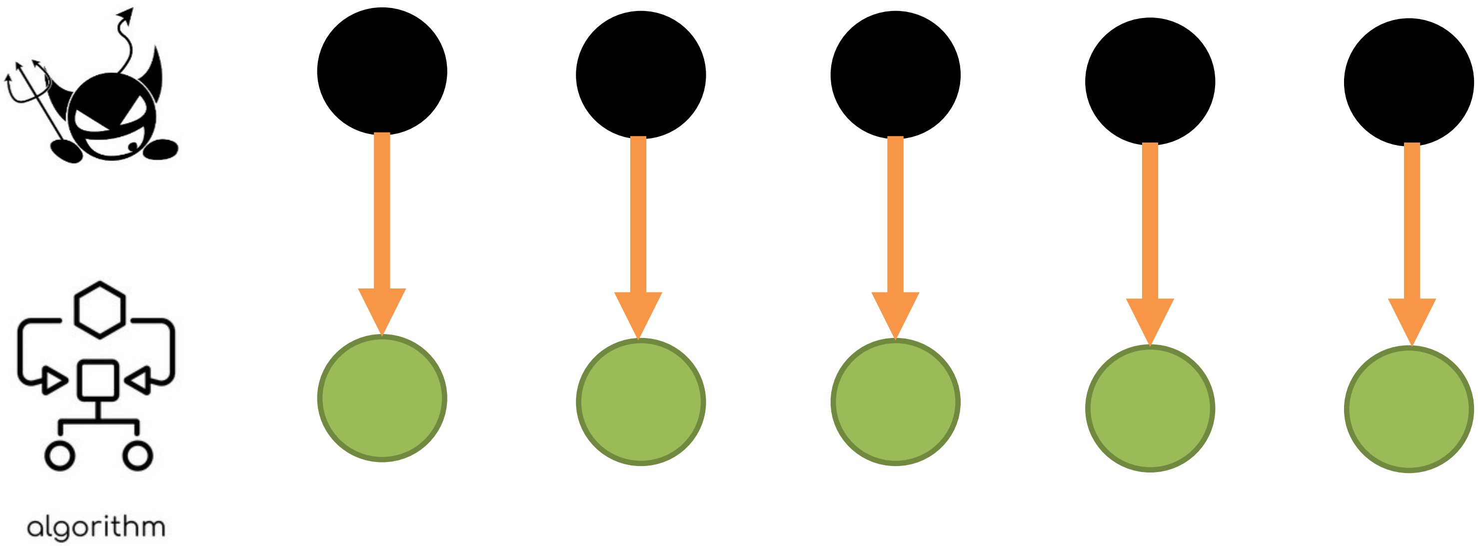
- 对手（Adversary）：知道你的算法!
  - Oblivious Adversary(短视对手):算法执行之前构建输入序列，获得离线最优解
  - Oblivious Adversary(短视对手)是一种weak adversary
  - 对于确定性在线算法， 可以构建出算法的最坏测试序列
  - 对于随机性在线算法：存在随机数生成因子，难以提前构建出最坏测试序列

Worst Case Model中的对手(Adversary)



- 对手（Adversary）：知道你的算法!
  - Adaptive online Adversary(自适应在线对手)：对手在算法执行时根据算法的action确定下一步的最坏测试序列输入项
  - Adaptive online Adversary(自适应在线对手)是一种medium adversary

Worst Case Model中的对手(Adversary)



- 对手（Adversary）：知道你的算法!
  - Adaptive offline Adversary(自适应离线对手)：对手全知全能，包括算法和算法随机数生成器的结果，算法执行前获得最坏测试序列
  - Adaptive offline Adversary(自适应离线对手)是一种strong adversary
  - 对于随机性在线算法，也可以构建出算法的最坏测试序列
  - 只是一种假设模型，实际场景中难以实现

Worst Case Model中的对手(Adversary)

- 难点：针对随机性在线算法，需要精心设计对手模型

- 方法1(姚氏定理)：要确定随机性在线算法的竞争比，只需找到输入的一个分布，并基于Oblivious Adversary(短视对手)模型证明没有确定性算法可以针对该分布表现良好。

Theorem 1.3. *(Yao's MINIMAX principle) For any randomized algorithm $\mathcal{A}$ and random instance $\mathcal{I}$ as detailed above we have*

$$\max_{I}\left[\frac{\mathbb{E}\mathcal{A}(I)}{OPT(I)}\right] \geq \min_{det\ A} \mathbb{E}_{\mathcal{I}}\left[\frac{A(\mathcal{I})}{OPT(\mathcal{I})}\right]$$

- 方法2：通过概率论、线性规划等工具，推出 $\alpha = \max_{I} \frac{Exp[A(I)]}{OPT(I)}$

大纲

1. 在线问题和在线算法的简单介绍
2. **在线算法的案例和证明方法**
3. 从Worst Case Model到ROM
4. 总结、未来研究方向和现有问题
5. 附录：Primal-Dual证明

Online Ski Rental问题

- 问题场景
  - 你想要在未来的 k 天去滑雪
  - 在线场景： k 未知 （可能取决于天气）
  - 每一天的决策选择方案：
    - 一次性购买雪橇（花费B元）
    - 租一天雪橇（花费1元/天）
- 目标：最小化 k 天内的花费
- 转化为Worst Case模型：
  - 对手：根据决策者的在线算法确定最坏测试序列 k ，并计算已知 k 下的离线最优解
  - 离线最优算法：当k>B, 则第一天买；否则每天租

Online Ski Rental问题

- 算法1：顾客第一天就购买雪橇
  - Worst Case模型对手： k = 1
  - C.R. = A(k) / OPT(k) = B / 1 = B
  - 当B趋近于无穷，C.R.结果差

- 算法2：顾客每天都租雪橇
  - Worst Case模型对手： $\mathrm{k} = n,\ n \to \infty$
  - C.R. = A(k) / OPT(k) = n / B
  - 当 n 趋近于无穷，C.R.结果差

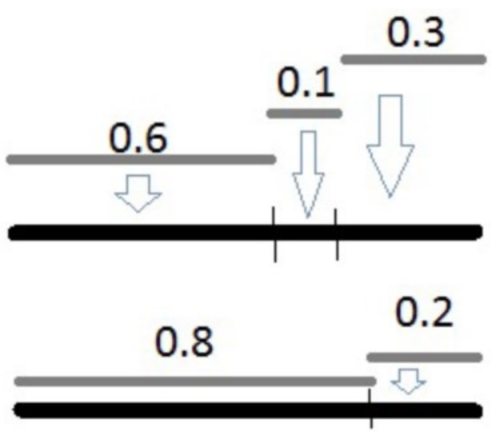
Online Ski Rental问题

- 算法3(break-even)：顾客前B-1天租，第B天买
  - Worst Case模型对手： k = B
  - C.R. = A(k) / OPT(k) = (B – 1) + B / B ≈ 2 [O(1)]
  - 没有一个确定性算法可以实现C.R. < 2
  - 但随机性算法可以提升!

- 算法4(optimal strategy)：顾客 p_i 的概率决定在第i天租并在第i+1天买，且 $(p_0 + p_1 + \ \ldots + p_{B-1}) = 1$
  - 当$\mathrm{p_i} = \dfrac{\left(1+\frac{1}{B}\right)^{i+1} - 1}{\left(1+\frac{1}{B}\right)^{B} - 1}, C.R. \leq \dfrac{e}{e-1} \approx 1.58$ [证明方法：见附录]
  - 没有一个随机性算法可以超过本算法

Online Bin Packing问题

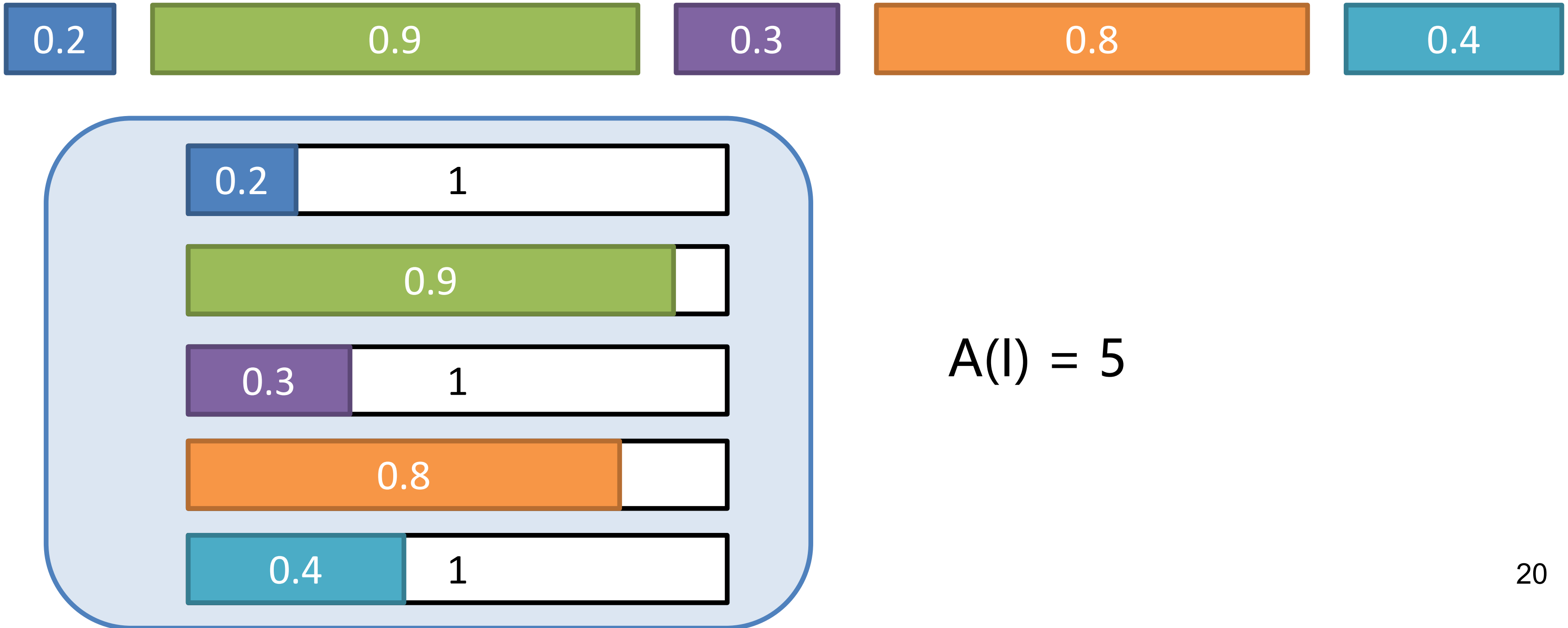
- 问题场景
  - 给定n个item，大小分别为$s_1, s_2, \ldots, s_n$，$s.t. s_i \in (0,1]$
  - 给定无限个bin，每个bin的大小为1
  - 在线场景：
    - item一个接一个在线到达，未来item信息未知
    - 每个item到达后必须快速且不可撤回地决策
    - 必须将所有的item装进bin中



- 目标：最小化装完所有item所需的bin数量

Online Bin Packing问题

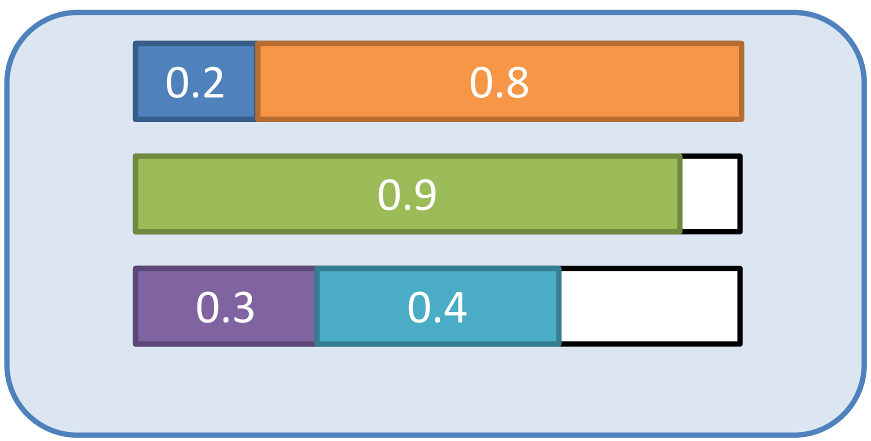
- 算法1： Next Fit
  - 每个时刻都维护一个open bin
  - 当一个新的item到达时，如果open bin满足需求，则装入；否则关闭并打开一个新的open bin
  - O(n) time，O(1) memory



A(I) = 5

Online Bin Packing问题

- 算法1： Next Fit
  - 每个时刻都维护一个open bin
  - 当一个新的item到达时，如果open bin满足需求，则装入；否则关闭并打开一个新的open bin
  - O(n) time，O(1) memory



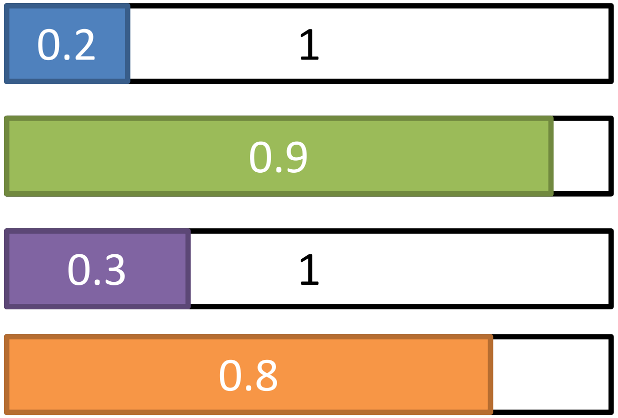
OPT(I) = 3

回顾：近似算法证明的流程

- 假设我们想证明一个算法 ALG 是一个对某些**最小化cost问题**的 α-近似 算法，通常的证明流程：
  - 对任何的输入实例I，找到OPT cost的下界(Lower Bound, LB)：$LB(I) \leq c\big(OPT(I)\big), \forall I$
  - 对任何的输入实例I， 都有：$c\big(ALG(I)\big) \leq \alpha LB(I), \forall I, \alpha \geq 1$
  - 推断出结论：$c(ALG) \leq \alpha LB \leq \alpha c(OPT)$

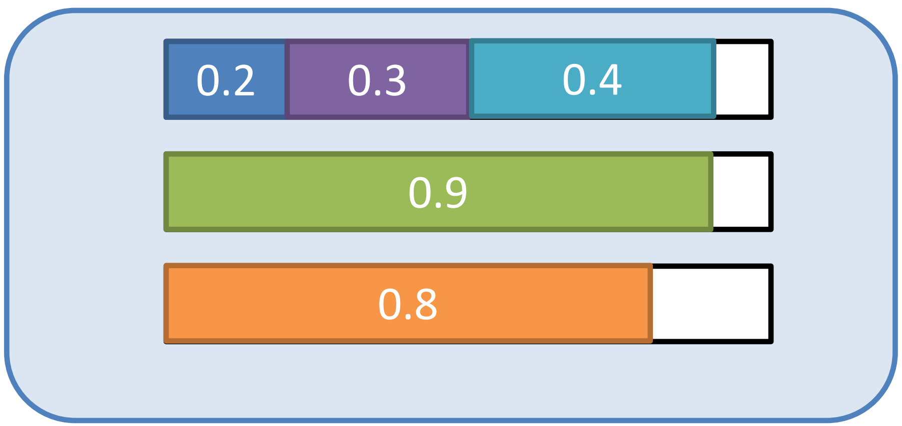
Online Bin Packing问题

- 算法1： Next Fit （竞争比证明）
  - 第一步[找OPT的下界LB]：$\mathrm{OPT}(I) \geq \sum_i s_i = LB(I)$
  - 第二步[证明算法ALG是LB的下界]： $\big(ALG(I)\big) \leq \alpha LB(I)$
    - 观察：{size of items in (2i-1)th bin and (2i)th bin} > 1
    - 推论：$\mathrm{LB}(I) = \sum_i s_i = \sum_{i=1}^{\frac{ALG(I)}{2}}$ {size of items in (2i−1)th bin and (2i)th bin}
    - $> \sum_{i=1}^{\frac{ALG(I)}{2}} \{1\} = \frac{ALG(I)}{2}$
  - 第三步[推断出结论]：$OPT(I) \geq LB(I) \geq \frac{ALG(I)}{2}$



Online Bin Packing问题

- 算法2： First Fit
  - 当一个新的item到达时，装入第一个满足需求的bin；否则关闭并打开一个新的open bin
  - O(n) time，O(n) memory
  - C.R. = 1.7 [证明略]



A(I) = 3

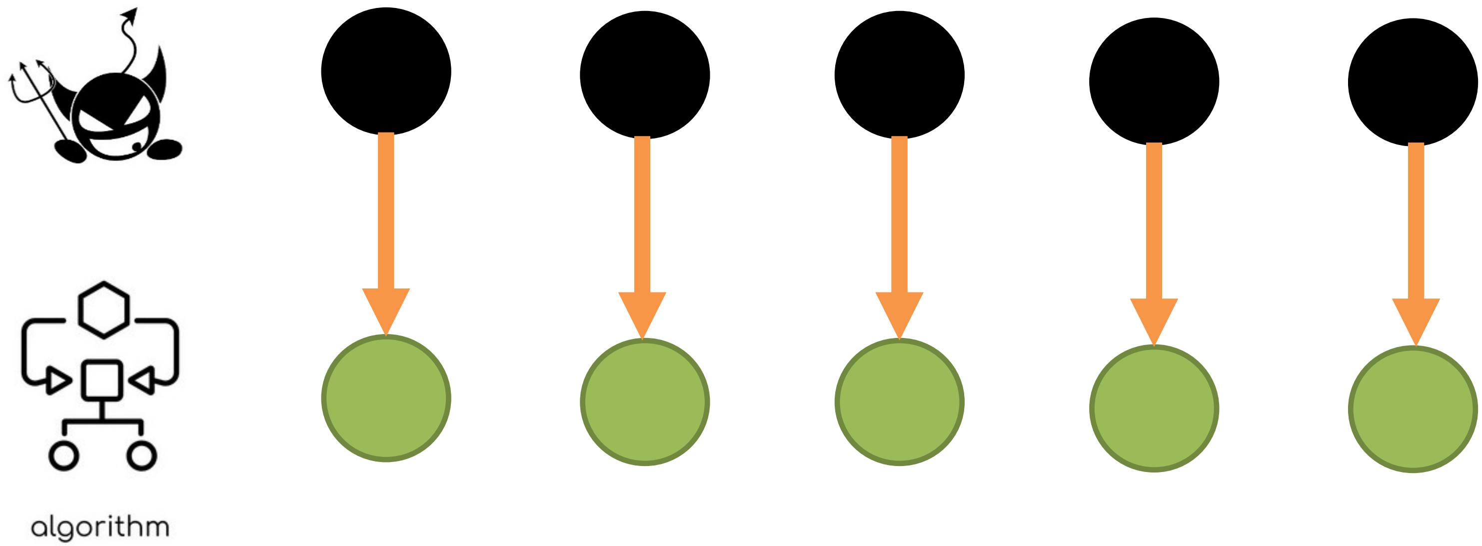
Online Bin Packing问题

- 算法3[Optimal]： Harmonic Algorithm [JACM’ 85]
  - 事先确定item的归属类:
    - item size $\in \left(\frac{1}{k+1}, \frac{1}{k}\right]$，则属于第k类，$1 \leq k < M$;
    - Item size $\in \left(0, \frac{1}{M}\right]$，则属于第M类
  - 每个时刻维护M个open bin，如果第k类item到达且对应bin满足需求则装入否则开启新bin
  - C.R. = 1.69103 [证明略]
  - 当只维护O(1)数量的open bin时，这个算法目前是最优的

大纲

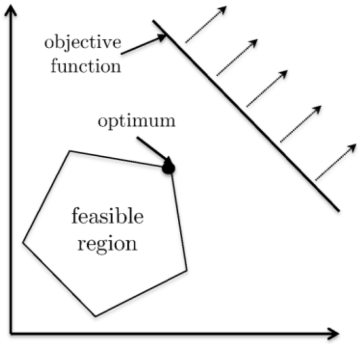
1. 在线问题和在线算法的简单介绍
2. 在线算法的案例和证明方法
3. **从Worst Case到ROM**
4. 总结、未来研究方向和现有问题
5. 附录：Primal-Dual证明

回顾：Worst Case Model



- Worst Case Model中的对手需要：
  - Step 1: 选择一个最坏输入集合
  - Step 2: 确定一个最坏的输入到达顺序

回顾：Worst Case Model



- Worst Case Model的缺点
  - 过于悲观，很多算法难以获得形式上的竞争比
  - 评价指标可能不符合实际场景
  - 缺少数据模型 （or, Worst Case Model本质是一个“墨菲定律”模型）
- 典型的难以用Worst Case Model分析的有效算法：Deep Learning
  - 无法在Worst Case Model分析出竞争比
  - 过度参数化的神经网络有优秀的泛化能力？ => 原因：现实输入序列通常不是最坏Case，评价指标不适合
- 研究方向：找到更贴合现实世界场景的对手模型

Secretary问题

7分

1分

5分

8分

- 招聘一个秘书：n个价值不同的候选人，次序到达
- 当候选人到达时，可以获得候选人的价值
- HR只能选择 接受 或 拒绝，无法撤回操作
- 当HR接受了候选人，则招聘结束
- 当HR拒绝了候选人，则等待下一个候选人出现
- 目标：接受价值最高的候选人

Worst-Case Model:

Random Order Model:

针对最坏输入序列，
[illegible]CR.较
好的算法

Secretary问题 - ROM

7分

1分

5分

8分

- Random Order Model (ROM) 的对手模型
  - 有n个价值不同的候选人
  - 从(n!)个候选人排列中选择一个随机排列作为输入序列
  - 获得该输入序列的离线最优值
- ROM的对手是Worst Case Model的弱化版对手
  - 不知道决策者的算法
  - 不再决定最坏输入序列

Secretary问题 - ROM

7分

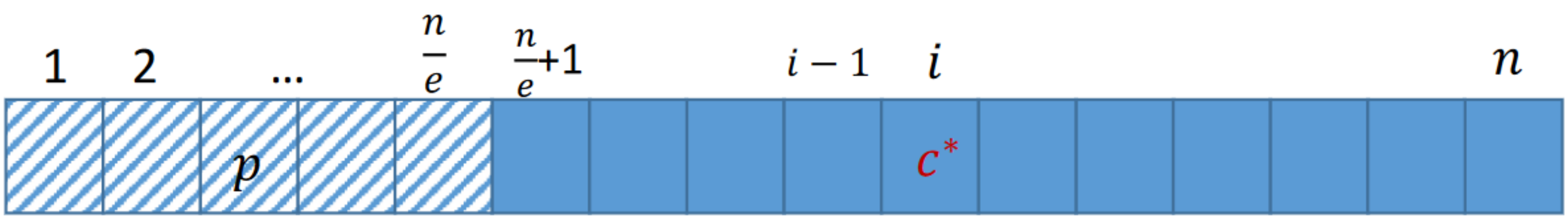
1分

5分

8分

- 算法(when n -> ∞)
  - 拒绝前n/e个候选人（构成集合S），以采样足够的信息
  - 计算集合S中候选人的最大价值p
  - 从剩余的候选人中选择第一个价值大于p的候选者
- 竞争比：1/e

Secretary问题 – 竞争比分析



算法 C.R. 证明

- $E\big(ALG(I)\big) = \Pr[c^*\ is\ selected\ ] = \sum_{i=\frac{n}{e}+1,\ldots,n} \Pr[c^*\ comes\ in\ i'th] * \Pr[\mathrm{c}^*\ \text{is}\ accepted\ in\ i'th\ round]$
- $\Pr[c^*\ comes\ in\ i'th] = \frac{1}{n}$
- $\Pr[\mathrm{c}^*\ \text{is}\ accepted\ in\ i'th\ round] = \Pr\left[\mathrm{i} > \frac{n}{e}\right] * \Pr\left[item\ j < p,\ \forall j \in \left[\frac{n}{e}+1, i-1\right]\right]$

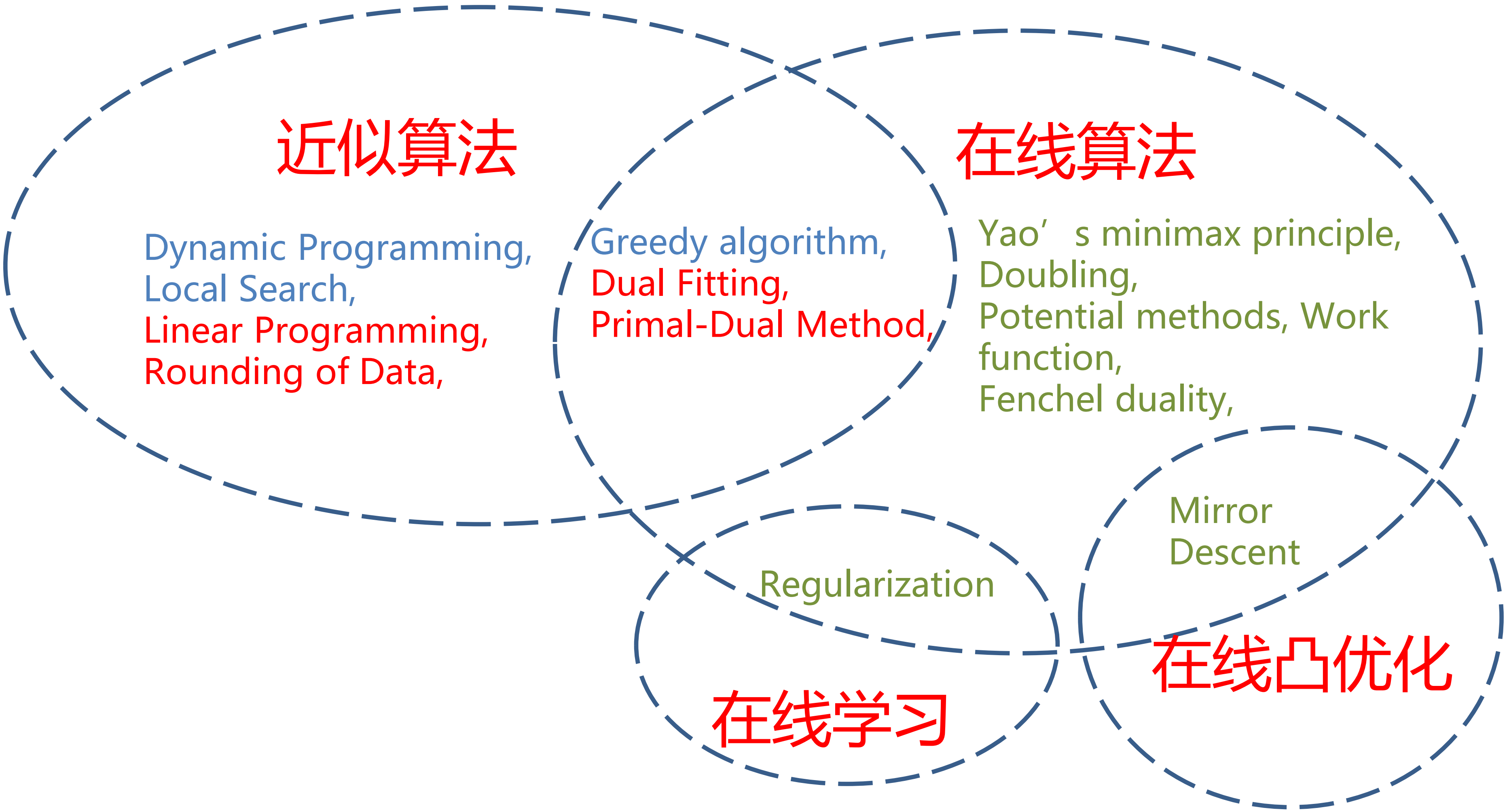
$$= \frac{n}{\mathrm{e}} * \frac{1}{\mathrm{i}-1}$$

- $\Pr[c^*\ is\ selected\ ] = \sum_{i=\frac{n}{e}+1,\ldots,n} \left(\frac{1}{n} * \frac{n}{e} * \frac{1}{i-1}\right) \geq \frac{1}{e}\ln\left(n * \frac{e}{n}\right) \approx \frac{1}{e}$

大纲

1. 在线问题和在线算法的简单介绍
2. 在线算法的案例和证明方法
3. 从Worst Case到ROM
4. **总结、未来研究方向和现有问题**
5. 附录：Primal-Dual证明

技术和工具



- 需要掌握的工具：概率论、线性规划、凸优化、博弈论、图论…

一些相关的在线问题

- Online load balancing and scheduling
- Online network routing
- Online graph coloring
- Paging, metrical task systems, and k-server
- Online TSP and Steiner tree
- Online Facility Location problem
- Online Set Cover problem

现有问题

- 在线算法只能保证部分场景下的性能下界
  - Worst Case Model：保证在最坏输入下算法的性能，其他输入下性能可能不如其他的算法

- 考验建模能力、需要对问题进行简化
  - 近似算法严重依赖场景的设计和问题的构造

- 大量的问题没有可行的在线算法
  - 现实数据集下性能指标不如启发式算法
  - 问题本身无法证明竞争比

(适合我们的)未来研究方向

- 现有在线优化算法在新场景中的应用
  - 现实世界中大量问题都可以被（简化）建模成已知的在线问题
  - Scheduling、Resource Management、Robot Motion Planning
- 对目前现有的近似算法/在线算法进行详细整理
  - 这个工作很耗时但对实验室理论研究比较重要
  - 整理成可用的工具集，增强论文的理论性保证

学习资料

- Approximation Algorithms
  - CMU 15-854: Approximation Algorithms
  - CMU 15-854(B): Advanced Approximation Algorithms
  - Umich IOE 2713: Approximation & Online Algorithms
  - Williamson D P, Shmoys D B. The design of approximation algorithms[M]. Cambridge university press, 2011.
- Online Algorithms
  - UofT CSC2421: Topics in Algorithms: Online and other Myopic Algorithms
  - Nguyen T K. Primal-Dual Approaches in Online Algorithms, Algorithmic Game Theory and Online Learning[D]. Université Paris Sorbonne, 2019.

学习资料

Optimization Online

- Categories – Optimization Online (optimization-online.org)

- Applications – OR and Management Sciences (1,556)
  - Airline Optimization (31)
  - Finance and Economics (185)
  - Marketing (14)
  - Production and Logistics (151)
  - Scheduling (215)
  - Supply Chain Management (82)
  - Telecommunications (109)
  - Transportation (295)
  - Yield Management (16)
- Applications – Science and Engineering (1,209)
  - Basic Sciences Applications (79)
  - Biomedical Applications (99)
  - Chemical Engineering (29)
  - Civil and Environmental Engineering (28)
  - Control Applications (142)
  - Data-Mining (155)
  - Facility Planning and Design (78)
  - Mechanical Engineering (43)
  - Multidisciplinary Design Optimization (33)
  - Optimization of Systems modeled by PDEs (61)
  - Smart Grids (42)
  - Statistics (179)
  - VLSI layout (10)
- Integer Programming (1,692)
  - (Mixed) Integer Linear Programming (591)
  - (Mixed) Integer Nonlinear Programming (483)
  - 0-1 Programming (265)
  - Cutting Plane Approaches (280)
- Linear, Cone and Semidefinite Programming (1,409)
  - Cone Programming (5)
  - Linear Programming (290)
  - Second-Order Cone Programming (107)
  - Semi-definite Programming (554)
- Network Optimization (271)
- Nonlinear Optimization (2,089)
  - Bound-constrained Optimization (78)
  - Constrained Nonlinear Optimization (658)
  - Nonlinear Systems and Least-Squares (104)
  - Quadratic Programming (261)
  - Systems governed by Differential Equations Optimization (121)
  - Unconstrained Optimization (329)
- Optimization in Data Science (14)
  - Data Science Algorithms (1)
  - Data Science Applications (1)
  - Data Science Theory (3)

学习资料

Scheduling Zoo

- The scheduling zoo (lip6.fr)

The Scheduling Zoo

interface : simple | advanced

Machine environment α

type : 1 | P | Q | R | O | F | J

Constraints β

number of jobs : $\varnothing$ | $n=2$ | $n=3$ | $n=k$

precedence relation : $\varnothing$ | prec | chains | tree | intree | outtree | opposing forest | sp-graph | bounded height | level order | interval order | quasi-interval order | over-interval order | Am-order | DC-graph | 2-dim partial order | k-dim partial order

release time : $\varnothing$ | r_j | online-r_j

preemption : $\varnothing$ | pmtn

due date : $\varnothing$ | $d_j = d$

processing times : $\varnothing$ | $p_j = 1$ | $p_j = p$

batching : $\varnothing$ | s-batch | batch(∞) | batch(b)

Objective function γ

Objective function : $C_{\max}$ | $C_{\min}$ | $\sum C_j$ | $\sum w_j C_j$ | $L_{\max}$ | $\sum U_j$ | $\sum w_j U_j$ | $\sum T_j$ | $\sum w_j T_j$

谢谢各位老师指正

中山大学计算机学院

汇报人：肖霖畅

大纲

1. 在线问题和在线算法的简单介绍
2. 在线算法的案例和证明方法
3. 从Worst Case到ROM
4. 总结、未来研究方向和现有问题
5. **附录：Primal-Dual证明**

Online Ski Rental问题: Primal-Dual

- 算法4(optimal strategy)：顾客 p_i 的概率决定在第i天租并在第i+1天买，且 $(p_0 + p_1 + \ ... + p_{B-1}) = 1$

- 当$p_i = \frac{\left(1+\frac{1}{B}\right)^{i+1} - 1}{\left(1+\frac{1}{B}\right)^{B} - 1}$, $C.R. \leq \frac{e}{e-1} \approx 1.58$

- Step 1：构建整数优化问题：

$$\min Bx + \sum_{i=1}^{k} z_i$$
$$\text{s.t.} \forall i \in [k], x + z_i \geq 1$$
$$x \geq 0, \forall i, z_i \geq 0$$

$$x = \begin{cases} 1 \text{ - Buy} \\ 0 \text{ - Don't Buy} \end{cases} \qquad z_i = \begin{cases} 1 \text{ - Rent on day i} \\ 0 \text{ - Don't rent on day i} \end{cases}$$

回顾：近似算法证明的流程

- 假设我们想要证明一个算法 ALG 是一个对某些**最小化cost问题**的 α-近似 算法，通常的证明流程：
  - 对任何的输入实例I，找到OPT cost的下界(Lower Bound，LB)：
    $LB(I) \le c\big(OPT(I)\big), \forall I$
  - 对任何的输入实例I， 都有：$c\big(ALG(I)\big) \le \alpha LB(I), \forall I, \alpha \ge 1$
  - 推断出结论：$c(AL$证明对任何的输入实例$G) \le \alpha LB \le \alpha c(OPT)$

- 挑战：
  - 挑战 1：难以找到下界LB以及最优解OPT和下界LB的关系
  - 挑战 2：找到了问题的下界LB，却找不到算法ALG可以与LB联系
- 一个非常有用的工具：线性规划！

回顾：线性规划-寻找问题下界LB的工具

- **重要性质**：任何一个整数规划(IP)的可行解也是它对应的放松线性规划问题(LP)的可行解， $\mathrm{OPT}_{\mathrm{LP}} = c(Z_{LP}^*) \le c(Z_{\mathrm{IP}}^*) = OPT$

- **一个简单的推论**：使用IP建模问题，其对应的放松LP问题的最优解就是OPT的一个下界，完成了近似算法构造的第一步：对任何的输入实例I，找到OPT的下界(Lower Bound, LB)：$LB(I) \le c\big(OPT(I)\big), \forall I$

回顾：Primal-Dual Method

- **用线性规划构造近似算法的难点：**
  - 证明对任何的输入实例I， 都有：$c(ALG(I)) \le \alpha * OPT_{LP}, \forall I, \alpha \ge 1$
- **LP对偶问题：为LP原始问题提供一个下界**
  - LP Monogamy: 原始问题和对偶问题相互对偶
  - Weak Duality: 对任何的原始对偶问题的解(x, y)，都有$c^T x \ge b^T y$ [下界]
    - $c^T x \ge OPT_P \ge OPT_D \ge b^T y$

$$\begin{aligned} &\text{minimize } c^T x \\ &\text{subject to } Ax \ge b \\ &\quad x \ge 0 \end{aligned} \qquad \begin{aligned} &\text{maximize } b^T y \\ &\text{subject to } A^T y \le c \\ &\quad y \ge 0 \end{aligned}$$

  - Strong Duality: 若原问题和对偶问题中有任意一个存在有界的最优解，则另外一个存在相同的最优解。
  - Complementary Slackness :若(x, y)是原始对偶问题的可行解，存在α，β使 $\begin{cases} \forall i,\ if\ x_i > 0,\ then\ c_i/\alpha \le (A^T)_i \cdot y \le c_i \\ \forall i,\ if\ y_i > 0,\ then\ b_i \le A_i \cdot x \le \beta b_i \end{cases}$ 成立，则 $c^T x \le \alpha \cdot \beta \cdot b^T y$
    - $\alpha \cdot \beta \cdot b^T y \ge c^T x \ge OPT_P \ge OPT_D \ge b^T y;\ \alpha \cdot \beta \cdot OPT_P \ge c^T x$

回顾：Primal-Dual Method

写LP对偶问题的一个简单例子

- 可能的复杂情况：各种求和，展开即可

①

$$
\begin{aligned}
\min\ & -50x_1+20x_2 \\
\text{s.t.}\ & 2x_1-x_2 \ge -5 && y_1 \\
& 3x_1+x_2 \ge 3 && y_2 \\
& 2x_1-3x_2 \le 12 && y_3 \\
& x_1, x_2 \ge 0
\end{aligned}
$$

②

$$
\begin{aligned}
2y_1x_1-y_1x_2 &\ge -5y_1 \\
3y_2x_1+y_2x_2 &\ge 3y_2 \\
-2y_3x_1+3y_3x_2 &\ge -12y_3
\end{aligned}
$$

$$\max\ -5y_1+3y_2-12y_3$$

③

$$
\begin{aligned}
2y_1x_1-y_1x_2 &\ge \cdots \\
3y_2x_1+y_2x_2 &\ge \cdots \\
-2y_3x_1+3y_3x_2 &\ge \cdots
\end{aligned}
$$

$$
\begin{aligned}
2y_1+3y_2-2y_3 &\le -50 \\
-y_1+y_2+3y_3 &\le 20
\end{aligned}
$$

④

$$
\begin{aligned}
\max\ & -5y_1+3y_2-12y_3 \\
\text{s.t.}\ & 2y_1+3y_2-2y_3 \le -50 \\
& -y_1+y_2+3y_3 \le 20 \\
& y_1, y_2, y_3 \ge 0
\end{aligned}
$$

Online Ski Rental问题: Primal-Dual

- Step 2：放松成LP问题后并获得对偶问题

$$\min Bx + \sum_{i=1}^{k} z_i$$
$$\text{s.t.} \forall i \in [k], x + z_i \geq 1$$
$$x \geq 0, \forall i, z_i \geq 0$$

$$\max \sum_{i=1}^{k} y_i$$
$$\text{s.t.} \sum_{i=1}^{k} y_i \leq B$$
$$\forall i, 0 \leq y_i \leq 1$$

- Step 3：针对放松的LP的Primal和Dual问题，提出一个合适的算法
  - 约束在线到达
  - Step 3.1：证明算法是Primal问题可行解
  - Step 3.2：证明算法是Dual问题可行解
  - Step 3.3：证明算法满足 ΔP < k ΔD

1. Initially x, all z_i and y_i are 0
2. When we see constraint i online
   - if $x = 1$, do nothing
   - else
     (a) $z_i \leftarrow 1 - x$
     (b) $x \leftarrow (1 + 1/B)x + 1/cB$
     (c) $y_i \leftarrow 1$

Online Ski Rental问题: Primal-Dual

- Step 3.1：证明算法是Primal问题可行解
  - 当 x < 1 时， $z_i = 1 - x$ 满足约束
  - 当 x >= 1 时， 满足约束
- Step 3.2：证明算法是Dual问题可行解
  - $y_i = 1$ 满足约束
  - 当 x < 1 时，最多只会执行B次： $y_i = 1$
- Step 3.3：证明算法满足 $\Delta P \le k\, \Delta D$
  - $\Delta D = 1$
  - $\Delta P = B \cdot \Delta x + z_i$
  - $= B \cdot \left(\left(1 + \frac{1}{B}\right)x + \frac{1}{cB} - x\right) + (1 - x)$
  - $= \left(x + \frac{1}{c}\right) + (1 - x) = 1 + \frac{1}{c}$
  - $\Delta P \le (1 + 1/c)\, \Delta D \Rightarrow cost_P = \sum \Delta P \le \left(1 + \frac{1}{c}\right) \sum \Delta D = \left(1 + \frac{1}{c}\right) \mathrm{cost_D}$

$$\min Bx + \sum_{i=1}^{k} z_i$$
$$\text{s.t.} \forall i \in [k], x + z_i \ge 1$$
$$x \ge 0, \forall i, z_i \ge 0$$

$$\max \sum_{i=1}^{k} y_i$$
$$\text{s.t.} \sum_{i=1}^{k} y_i \le B$$
$$\forall i, 0 \le y_i \le 1$$

1. Initially x, all z_i and y_i are 0
2. When we see constraint i online
   - if $x = 1$, do nothing
   - else
     (a) $z_i \leftarrow 1 - x$
     (b) $x \leftarrow (1 + 1/B)x + 1/cB$
     (c) $y_i \leftarrow 1$

Online Ski Rental问题: Primal-Dual

- Step 3.3：证明算法满足 $\Delta P \leq k\ \Delta D$
  - $cost_D \leq OPT_P \leq cost_P = \sum \Delta P \leq \left(1+\frac{1}{c}\right)\sum \Delta D = \left(1+\frac{1}{c}\right)\mathrm{cost}_{\mathrm{D}}$
  - $cost_P \leq \left(1+\frac{1}{c}\right) OPT_P$ => 获得竞争比
- Step 4：c是额外引入的变量，需要和问题中已有变量进行变换
  - 写入 x 的解析表达式：
  - $$x = \frac{1}{cB} \cdot \frac{\left(1+\frac{1}{B}\right)^B - 1}{\left(1+\frac{1}{B}\right) - 1} = \frac{\left(1+\frac{1}{B}\right)^B - 1}{c}$$
  - 当 $x \geq 1$时，$c \leq \left(1+\frac{1}{B}\right)^B - 1 \approx e - 1$
  - $1 + \frac{1}{c} \approx \frac{e}{e-1}$

1. Initially x, all z_i and y_i are 0
2. When we see constraint i online
   - if $x = 1$, do nothing
   - else
     (a) $z_i \leftarrow 1 - x$
     (b) $x \leftarrow (1 + 1/B)x + 1/cB$
     (c) $y_i \leftarrow 1$

谢谢各位老师指正

中山大学计算机学院

汇报人：肖霖畅