

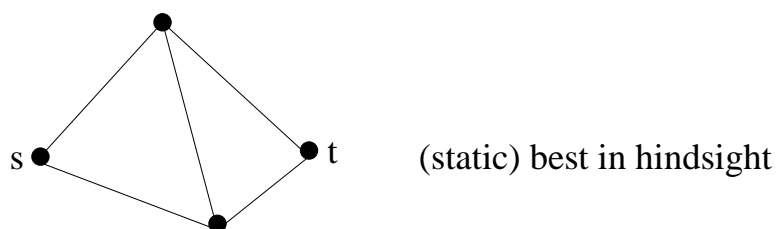
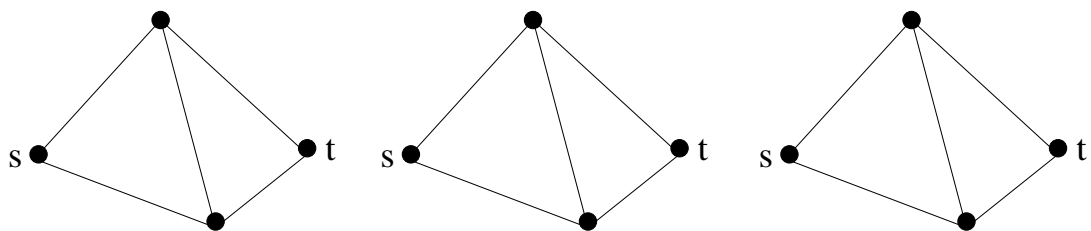
Algorithms for Online Optimization Problems

Adam Kalai
Santosh Vempala

IMA
May 2003

(Static) Online Optimization Example

Online shortest paths [TW02]:



For any sequence of bounded times,

$$E[\text{time}] \leq (1 + \epsilon) \times (\text{best in hindsight}) + \frac{C}{\epsilon}$$

Summary of results

Similar problems

Splay trees, list update [ST85]

Adaptive Huffman coding [F73,G78,K85]

Predicting from experts [LW87,...]

Online decision tree pruning [HS95]

Online shortest paths [TW02]

Online linear optimization [KV02]

...

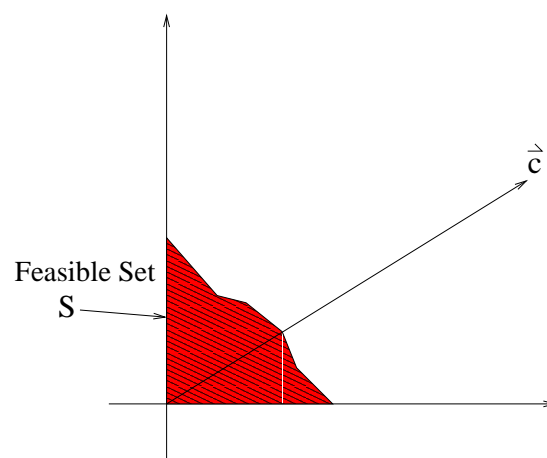
Simple algorithm, single analysis gives: [KV02]

$$E[\text{cost}] \leq (1 + \epsilon) \times (\text{best in hindsight}) + \frac{c}{\epsilon}$$

Efficient alg, ϵ probability of recomputing.

Previous work: not $(1 + \epsilon)$ -optimal, or inefficient, or problem specific.

Online Linear Optimization



$$\max_{\vec{x} \in S} \vec{x} \cdot \vec{c}$$

state $\vec{x} = (\# \text{ chairs}, \# \text{ tables})$

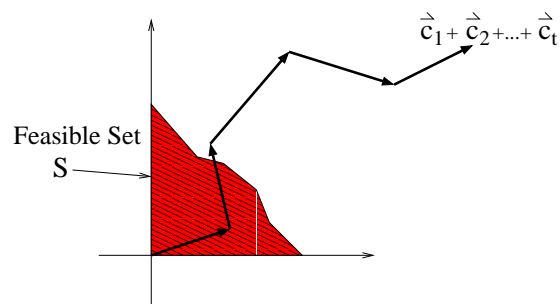
objective $\vec{c} = (\text{chair profit}, \text{table profit})$

Day	(chair profit, table profit)
1	(\$40, \$60)
2	(\$30, \$70)
3	(\$35, \$66)
⋮	⋮

Online Optimization Model

- S = feasible state set
- $\vec{c}_1, \vec{c}_2, \dots$, daily objective vectors, $|\vec{c}_j|_1 \leq 1$
- On day j , we choose state $\vec{x}_j \in S$ knowing only S and $\vec{c}_1, \vec{c}_2, \dots, \vec{c}_{j-1}$
- Get score = $\vec{x}_j \cdot \vec{c}_j$
- Maximize total score = $\vec{x}_1 \cdot \vec{c}_1 + \dots + \vec{x}_t \cdot \vec{c}_t$

Static Offline/Online Problems



$$\text{best state score} = \max_{\vec{x} \in S} \vec{x} \cdot (\vec{c}_1 + \vec{c}_2 + \dots + \vec{c}_t)$$

$t = \#$ days, $n = \#$ dimensions

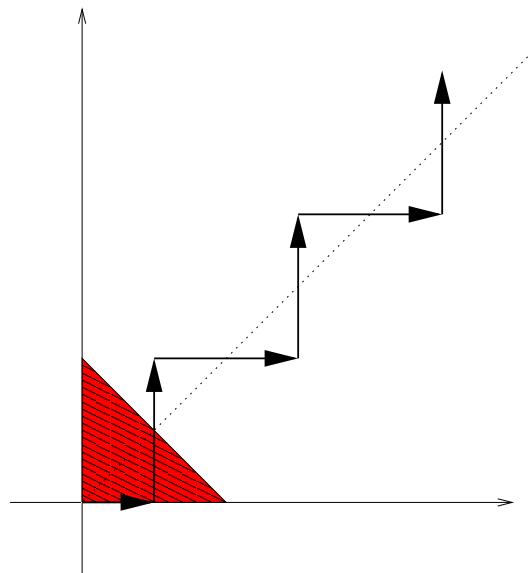
$d = L_1$ diameter of feasible state set S

$$\text{our score} \geq \text{best state score} - \epsilon t - \frac{d^2}{\epsilon}$$

For non-negative scores $\vec{x} \cdot \vec{c}_j \geq 0, \forall \vec{x} \in S$,

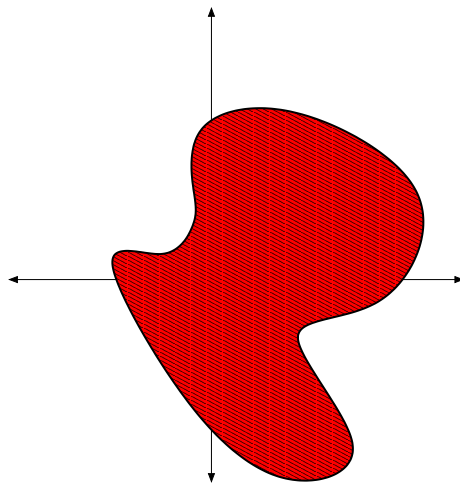
$$\text{our score} \geq (1 - \epsilon)(\text{best state score}) - \frac{d \log n}{\epsilon}$$

Follow the Leader



\vec{c}_j	best so far (leader)	Follow's total score	best score so far
(0.5, 0)	(1, 0)	0?	0.5
(0, 1)	(0, 1)	0	1.0
(1, 0)	(1, 0)	0	1.5
(0, 1)	(0, 1)	0	2.0
(1, 0)	(1, 0)	0	2.5
⋮	⋮	⋮	⋮

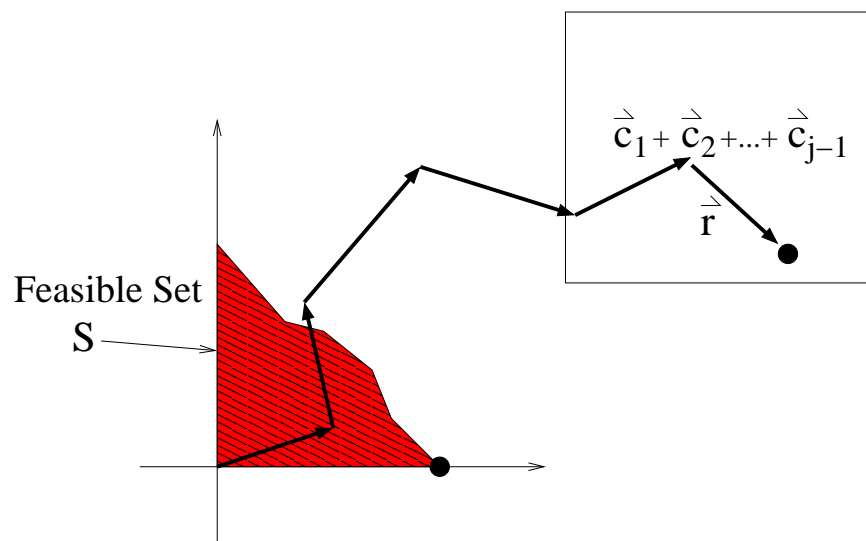
Maximization oracle



- Oracle $M : \mathcal{R}^n \rightarrow \mathcal{R}^n$
- $M(\vec{c})$ is any $\vec{x} \in S$ that maximizes $\vec{x} \cdot \vec{c}$
- best state score is

$$M(\vec{c}_1 + \cdots + \vec{c}_t) \cdot (\vec{c}_1 + \cdots + \vec{c}_t)$$

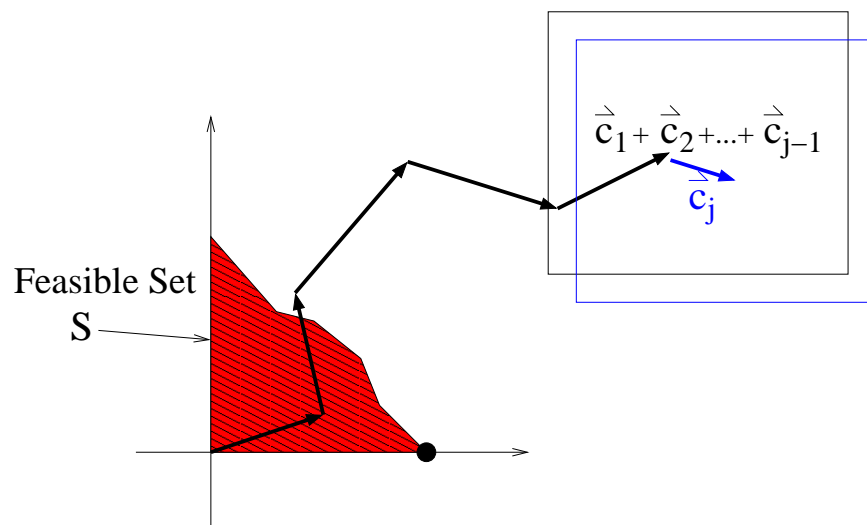
Follow the Leader



On day j use $M(\vec{c}_1 + \dots + \vec{c}_{j-1} + \vec{r})$, where \vec{r} is random from cube of side L .

Similar to Hannan's 1951 idea in game theory.

$\widetilde{\text{Be the Leader}}$

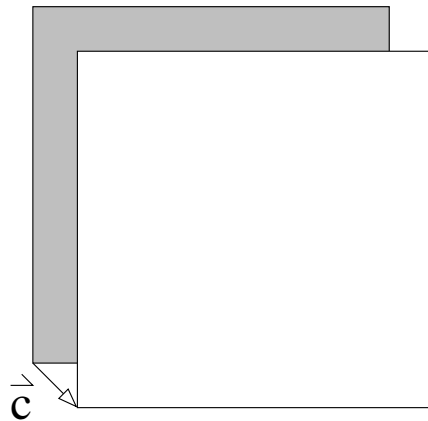


On day j use $M(\vec{c}_1 + \dots + \vec{c}_j + \vec{r})$, where \vec{r} is random from cube of side L .

Intuition: added randomness makes

$\widetilde{\text{follow the leader}} \approx \widetilde{\text{be the leader}}$

Nonoverlap



$$Pr(\text{nonoverlap}) \leq \frac{|\vec{c}|_1}{L} \leq \frac{1}{L}$$

$$\begin{aligned} E[M(\vec{c}_1 + \dots + \vec{c}_j + \vec{r}) \cdot \vec{c}_j - M(\vec{c}_1 + \dots + \vec{c}_{j-1} + \vec{r}) \cdot \vec{c}_j] \\ \leq (1 - \frac{1}{L})0 + \frac{1}{L}(\vec{v} \cdot \vec{c}_j - \vec{w} \cdot \vec{c}_j) \leq \frac{d}{L} \end{aligned}$$

Follow's Analysis

follow leader's score \geq be leader's score $- \frac{d}{L}t$
(from previous slide)

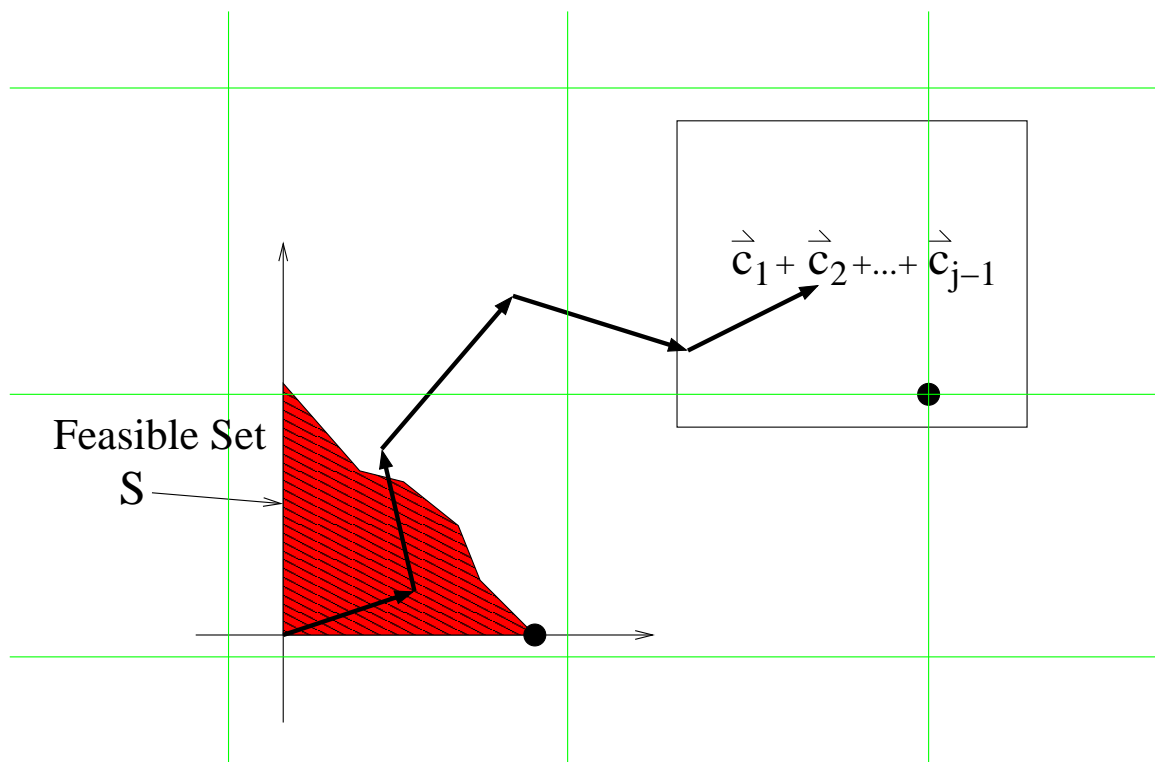
be leader's score \geq best score in hindsight $- dL$

Proof: With pretend initial perturbation $\vec{c}_0 = r$, be the leader is as good as best in hindsight. Difference is at most $(\vec{v} - \vec{w}) \cdot \vec{r} \leq dL$.

With $L = \frac{d}{\epsilon}$,

$$\begin{aligned} \text{follow's score} &\geq \text{best score} - \frac{d}{L}t - dL \\ &= \text{best score} - \epsilon t - \frac{d^2}{\epsilon} \end{aligned}$$

Lazy algorithm

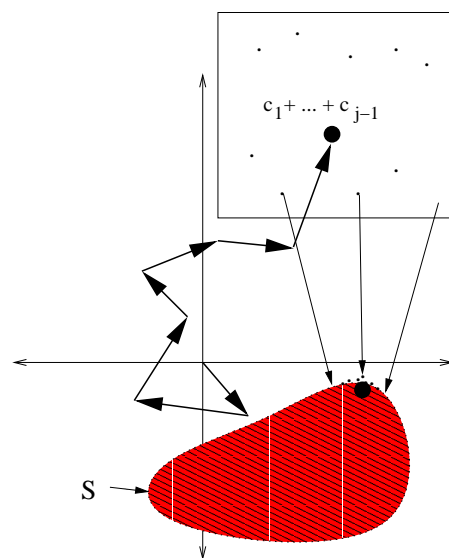


$$E[\text{lazy's score}] = E[\text{follow's score}]$$

Lazy recomputes with probability $\leq 1/L = \epsilon/d$

Cost of switching is no problem.

High probability bounds



1. Choose $\vec{r}_1, \dots, \vec{r}_k \in \mathbb{R}^n$ from $\text{cube}_n(d/\epsilon)$
2. On day j use average

$$\left(\frac{1}{k} \sum_{i=1}^k M(\vec{r}_i + \vec{c}_1 + \dots + \vec{c}_{j-1}) \right) \in S$$

For $k \geq \frac{d^2}{\epsilon^2 \delta t}$, with prob. at least $1 - \delta$,

our score \geq best state score $- 2\epsilon t - \frac{d^2}{\epsilon}$

$(1 - \epsilon)$ algorithm

1. Choose \vec{r} with density $\propto (1 - \epsilon)^{|\vec{r}|_1}$
2. On day j , use state $M(\vec{c}_1 + \dots + \vec{c}_{j-1} + \vec{c}_r)$

If $\vec{c}_j \cdot \vec{x} \geq 0 \quad \forall \vec{x} \in S, j \geq 1,$

our score $\geq (1 - \epsilon)(\text{best state score}) - \frac{d \log n}{\epsilon}$

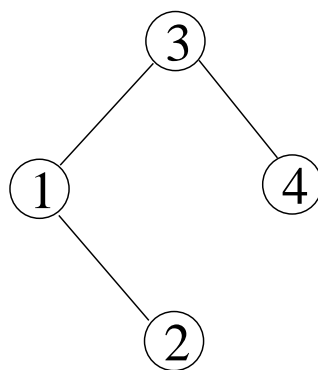
Shortest path application

- A dimension for each edge
- Objective vector = time on each edge
- Path = 0-1 vector with 1's on path edges
- Feasible set = set of paths
- Optimization oracle M = shortest path
- Lazy algorithm recomputes only with probability ϵ/d , and finds the shortest path given totals on all edges + randomness.

Conclusions

- Optimization \Rightarrow $(1 + \epsilon)$ -online optimization
- Extensions
 1. High probability bounds
 2. Lazy algorithm
 3. Partial information (bandits version)
 4. Tracking (dynamic bounds)
- Approximate optimization \Rightarrow approximate online optimization?
- Online nonlinear optimization? [Z03]

Online Binary Search Trees



Access	Cost	
1	2	$= (1,0,0,0) \cdot (2,3,1,2)$
4	2	$= (0,0,0,1) \cdot (2,3,1,2)$
4	2	$= (0,0,0,1) \cdot (2,3,1,2)$
1	2	$= (1,0,0,0) \cdot (2,3,1,2)$
2	3	$= (0,1,0,0) \cdot (2,3,1,2)$
3	1	$= (0,0,1,0) \cdot (2,3,1,2)$
	12	$= (2,1,1,2) \cdot (2,3,1,2)$

“splaying” cost $\leq 4.75(\text{best tree cost}) + O(\log n)$
 [Sleator, Tarjan 85]

Air Conditioned Trees

1. Initially, for $1 \leq i \leq n$, choose $a_i \in \{0, 1, \dots, N\}$
2. After access to element i ,
if ($\#$ accesses to i) $\geq a_i + N$ then
 - (a) $a_i := a_i + N$
 - (b) Change trees to $M(a_1, a_2, \dots, a_n)$

$N = n^2/\epsilon$ (L_1 diameter of trees $\leq n^2$)

$Pr(\text{changing trees}) \leq \epsilon/n^2$

Best tree $M(\cdot)$ can be computed in time $O(n^2)$

$$\begin{aligned} E[\text{our cost}] &\leq (\text{best tree cost}) + \epsilon t + \frac{n^4}{\epsilon} \\ &\leq (1 + \epsilon)(\text{best tree cost}) + \frac{n^4}{\epsilon} \end{aligned}$$

Online k -median problem

