Probabilistic All-or-Nothing Subsets and Extensions

Noam Goldberg
Department of Management
Bar-Ilan University

Based on collaborations with :
Michael Poss, CNRS Montpellier, France
Gabor Rudolf, Koç University, Turkey

April 26, 2018

Outline

- Given n items or activities, each item $i \in [n]$ with integer profit $c_i \geq 1$ and success probability p_i, our “generic” *probabilistic all-or-nothing* problem is

$$\max_{S \in \mathcal{S}} \sum_{j \in S} c_j \prod_{i \in S} p_i$$

where $\mathcal{S} \subseteq 2^{[n]}$ defines the sets that are feasible (not given as input).

- Given n items or activities, each item $i \in [n]$ with integer profit $c_i \geq 1$ and success probability p_i, our "generic" *probabilistic all-or-nothing* problem is

$$\max_{S \in \mathcal{S}} \sum_{j \in S} c_j \prod_{i \in S} p_i$$

  where $\mathcal{S} \subseteq 2^{[n]}$ defines the sets that are feasible (not given as input).
- Probabilistic All-or-Nothing Subset: $\mathcal{S}$ is the power set of $[n]$.

- Given n items or activities, each item $i \in [n]$ with integer profit $c_i \geq 1$ and success probability p_i, our "generic" *probabilistic all-or-nothing* problem is

$$\max_{S \in \mathcal{S}} \sum_{j \in S} c_j \prod_{i \in S} p_i$$

  where $\mathcal{S} \subseteq 2^{[n]}$ defines the sets that are feasible (not given as input).
- Probabilistic All-or-Nothing Subset: $\mathcal{S}$ is the power set of $[n]$.
- Constrained problems:
  - Extension to $\mathcal{S}$ that is a downward closed set system, e.g., $\mathcal{S}$ is the set of all matchings in a graph.
  - $\mathcal{S}$ is the set of all paths in a DAG.

Probabilistic All-or-Nothing Subset - Complexity

Theorem

This problem is NP-hard.

The proof is by a (nontrivial) reduction from subset sum: Is there a set S such that $\sum_{i \in S} c_i = M$?

Useful is the observation that

$$\max_{S \subseteq [n]} \sum_{j=1}^{n} c_j \prod_{i=1}^{n} p_i = \max_{x \in \{0,1\}^n} \sum_{j=1}^{n} c_j x_j \prod_{i=1}^{n} p_i^{x_i}$$

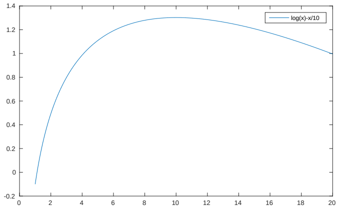
$$\max_x \sum_{j=1}^{n} c_j x_j \prod_{i=1}^{n} p_i^{x_i} \Leftrightarrow \max_x \sum_{i=1}^{n} \log(p_i) x_i + \log \left(\sum_{j=1}^{n} c_j x_j \right)$$

Complexity - Cont'd

Consider a reduction of subset sum with $\boxed{\text{same } c_i\text{'s and } p_i = e^{-\frac{c_i}{M}}.}$

Lemma

1. *There exists an $x \in \{0,1\}^n$ such that $\sum_{i=1}^n c_i x_i = M$ if and only if* $\max_{\mathbf{x}\in\{0,1\}^n} \log\left(\sum_{i=1}^n c_i x_i\right) - \frac{1}{M}\sum_{i=1}^n c_i x_i = \log M - 1.$
2. *For integer $M > 1$ and $y > 0$, $f(y) = \log(y) - \frac{y}{M}$ is concave, with a unique maximum at $f(M) = \log(M) - 1$. Further, for any positive integer $N \neq M$, $f(M) - f(N) \geq \frac{1}{5M^2}$.*



Max Probabilistic All-or-Nothing Subset - Pseudo-Polynomial Algorithm

For $i = 1, \dots, n$ let $P(i, C)$ denote the maximum probability of a subset of $\{1, \dots, i\}$ with a profit of exactly C. Consider the DP given by

$$P(i,C) = \begin{cases} \max\{P(i-1,C), p_i \cdot P(i-1, C-c_i)\} & i \geq 1 \text{ and } c_i \leq C \\ P(i-1,C) & i \geq 1 \text{ and } c_i > C \\ 1 & i \geq 0, C = 0 \\ -\infty & \text{otherwise.} \end{cases}$$

Using a straight-forward upper bound $\bar{C} = \sum_{i=1}^{n} c_i$,

$$max_{x \in \{0,1\}^n} \prod p_i^{x_i} \sum_{j=1}^{n} c_j x_j = \max_C \left\{ C \cdot P(n,C) \;\middle|\; C = \min_{i=1,\dots,n}\{c_i\}, \dots, \bar{C} \right\}.$$

The running time of this algorithm is $O(n\bar{C})$.

All-or-Nothing Subset Lemma

Lemma

Suppose S^* is (the support of a solution that is) optimal for all-or-nothing subset with $|S^*| \geq 2$. Then, for all $Q \subset S^*$, with $\prod_{i \in Q} p_i < \frac{1}{2}$, $\prod_{i \in S^* \setminus Q} p_i > \frac{1}{2}$.

Corollary

Suppose S^* is (the support of a solution that is) optimal. Then

$$\left| \left\{ i \in S^* \;\middle|\; p_i < \frac{1}{2} \right\} \right| \leq 1.$$

Omitting Small Probability Items?

Consider the instance $p = (0.99, 0.8, 0.18)$ and $c = (110, 120, 1000)$.

Here, $S^* = \{1, 3\}$ with value
$z^* = 0.99 \times 0.18 \times (110 + 1000) \approx 197.80$ is optimal.

Restricted to large probability items then $S = \{1, 2\}$ would be optimal with value 182.16.

$S = \{1, 2, 3\}$ has a value ≈ 47.05

Maximum Probabilistic All-or-Nothing Subset - FPTAS

Let
$N_{1/2} = \left\{ i \;\middle|\; p_i \geq \frac{1}{2} \right\} = \{1, \ldots, h\}$,

$$\hat{z}(i,j) = \max_{C=\min_{k\in[i]}\{\hat{c}_k\},\ldots,\hat{C}} (C+\hat{c}_j)\cdot\hat{P}(i,C)\cdot p_j.$$

```
Input: $\epsilon, c, p$
1: $z_{max} \leftarrow -\infty$
2: $\kappa \leftarrow \frac{\epsilon \max_{i\in N_{1/2}}\{p_i c_i\}}{n}$
3: for $j = h+1, \ldots, n+1$ do
4:     $z \leftarrow \kappa \cdot \hat{z}(h,j)$
5:     if $z_{\max} < z$ then
6:         $z_{\max} \leftarrow z$
7:     end if
8: end for
Output: $z_{\max}$
```

Proposition

This algorithm is an FPTAS with a runtime complexity bound of $O(n^4/\epsilon)$.

Extension to Downward Closed Set Systems

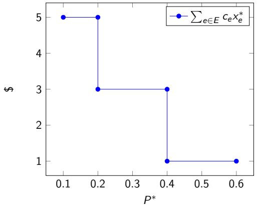
Letting $\mathcal{F} \subseteq \{0,1\}^n$ denote the set of indicator vectors corresponding to $\mathcal{S}$, consider the constrained linear problem:

$$\begin{aligned} \max \quad & \sum_e c_e x_e \\ \text{st} \quad & -\sum_e \ln(p_e) x_e \leq -\ln(P) \\ & x \in \mathcal{F} \end{aligned}$$

The previous all-or-nothing lemma also implies lower bounds on P^* – the following corollary.

Corollary

Let x^* be an optimal solution. Then $P^* \equiv \prod_{e \in E} p_e^{x_e^*} \geq \min\{1/8, p_{\min}/2\}$.

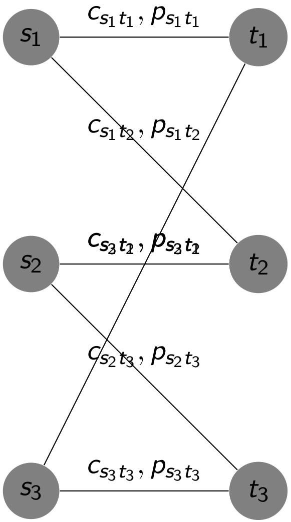


```
Input: $c, p, 0 < \epsilon < 1$
 1: $z_{\max} \leftarrow -\infty$
 2: for $j = h+1, \dots, n+1$ do
 3:     $x_{e_j} \leftarrow 1$
 4:     for $P \in \left\{ (1-\epsilon/2)^k \mid k = 1, \dots, \lceil 1 - \ln(8)/\ln(1-\epsilon/2) \rceil \right\}$ do
 5:         Compute $\epsilon/2$-approximate solution $x$ with value $z$ for budgeted problem on ground set $N_{\frac{1}{2}} \cup \{e_j\}$ (having fixed $x_{e_j} = 1$).
 6:         if $z_{\max} < z$ then
 7:             $z_{\max} \leftarrow z$
 8:         end if
 9:     end for
10: end for
Output: $z_{\max}$
```

Example: Probabilistic All-or-Nothing Matching

Now let E be edges of a graph.

$$\begin{aligned} \max \quad & \prod_{e \in E} p_e^{x_e} \left(\sum_{f \in E} c_f x_f \right) & \\ \text{subject to} \quad & \sum_{j \in \delta(i)} x_{ij} \leq 1 & i \in V \\ & x_e \in \{0,1\} & e \in E. \end{aligned}$$



s_1
t_1
$c_{s_1 t_1}, p_{s_1 t_1}$
$c_{s_1 t_2}, p_{s_1 t_2}$
s_2
t_2
$c_{s_3 t_1}, p_{s_3 t_1}$
$c_{s_2 t_2}, p_{s_2 t_2}$
$c_{s_2 t_3}, p_{s_2 t_3}$
s_3
t_3
$c_{s_3 t_3}, p_{s_3 t_3}$

Probabilistic All-or-Nothing Matching - cont'd

Proposition

Suppose $\mathcal{F}$ that is downward closed, a given $\epsilon > 0$, and that t is a running time complexity bound of a $(1 - \epsilon/2)$-approximation algorithm for the budgeted problem. Then, our outputs a $(1 - \epsilon)$-approximation with an $O\left(\frac{n}{\epsilon} t\right)$ complexity bound.

Lemma (Berger, Bonifaci, Grandoni, Schäfer, 2009)

For each $\epsilon > 0$ budgeted maximum matching can be solved to within $(1 - \epsilon)$ of the optimum with a runtime complexity bound of $O(m^{2/\epsilon + O(1)})$.

These results imply a PTAS for *maximum probabilistic all-or-nothing graph matching*.

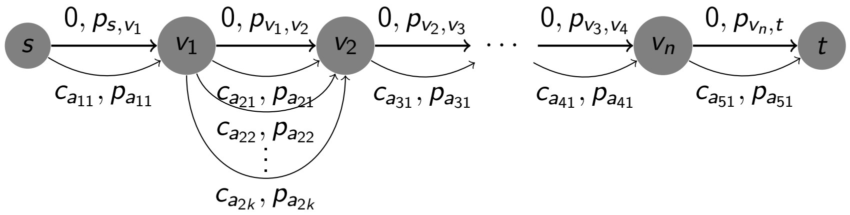
Probabilistic All-or-Nothing Network Path

Let $G = (V, E)$ be a directed graph, $s, t \in V$, and let $\delta^+(i) = \{j \in V : (i, j) \in E\}$ and $\delta^-(i) = \{j \in V : (j, i) \in E\}$. For each edge $e = (i, j) \in E$, we are given a positive (integer) profit $c_e = c_{ij} \geq 0$ and a probability of success $p_e = p_{ij} \in [0, 1]$.

The probabilistic all-or-nothing shortest path problem is to find a path π from s to t maximizing a similar objective

$$z(\pi) = \sum_{e \in \pi} c_e \prod_{f \in \pi} p_f .$$

Example Application – Project Critical Path Procurement



Pseudo-polynomial DP Method for All-or-Nothing Path

$$Z(C,i) = \begin{cases} \max\limits_{j \in \delta^-(i)} \left\{ C p_{ji} \frac{Z(C-c_{ji}, j)}{C-c_{ji}} \right\} & \substack{i \in V \setminus \delta^+(s) \text{ or} \\ c_{si} < C} \\ \max \left\{ c_{si} p_{si}, \max\limits_{j \in \delta^-(i) \setminus \{s\}} \left\{ C p_{ji} \frac{Z(C-c_{ji}, j)}{C-c_{ji}} \right\} \right\} & i \in \delta^+(s), C = c_{si} \\ -\infty & C < 0. \end{cases}$$

To determine z^* using the DP method given an upper bound $\bar{C} \geq \sum_{a \in \pi*} c_a$, Z must be evaluated for $C = 1, \ldots, \bar{C}$. So,

$$z^* = \max_{C=1,\ldots,\bar{C}} Z(C,t).$$

FPTAS I: Profit Scaling and Rounding

Profit Rounding and Scaling Approximation Schemes

```
Input: $G = (V, E), p, c, \epsilon, \ell, u$
1: $a^* \leftarrow \text{argmax}_{a \in E} R(a)$
2: while $q = 0, \dots, \lceil \log u - \log \ell \rceil - 1$ do
3:     $K \leftarrow \frac{\epsilon 2^q \ell}{n-1}$
4:     $\bar{c}_e \leftarrow \begin{cases} -\infty & c_e > 2^{q+1}\ell \\ \lfloor c_e / K \rfloor & \text{otherwise} \end{cases}$
5:     $\bar{z}_q \leftarrow K \max_{C=0,\dots,\lfloor \frac{2(n-1)}{\epsilon} \rfloor} Z(C, t, \bar{c})$, let $\pi_q = \pi(C, t)$ be the
       corresponding path.
6: end while
7: Select $q^* \in \text{argmax}\{z_q\}$
Output: $z(\pi_q)$
```

Profit Scaling and Rounding Result

Proposition

The profit scaling and rounding algorithm outputs a solution with objective $z \geq (1-\epsilon)z^*$. The complexity of the algorithm is $O(\frac{1}{\epsilon} mn \log(u/\ell))$

Profit Scaling and Rounding Result

Proposition

The profit scaling and rounding algorithm outputs a solution with objective $z \geq (1-\epsilon)z^*$*. The complexity of the algorithm is* $O(\frac{1}{\epsilon} mn \log(u/\ell))$

Corollary

The profit rounding and scaling algorithm invoked with $\ell = c_{\min}, u = nc_{\max}$ *outputs a solution with objective* $z \geq (1-\epsilon)z^*$*. The complexity of the algorithm is* $O(\frac{1}{\epsilon} mn \log(nc_{\max}/c_{\min}))$

Profit Scaling and Rounding Result

Proposition

The profit scaling and rounding algorithm outputs a solution with objective $z \geq (1-\epsilon)z^*$. *The complexity of the algorithm is* $O(\frac{1}{\epsilon} mn \log(u/\ell))$

Corollary

The profit rounding and scaling algorithm invoked with $\ell = c_{\min}, u = nc_{\max}$ *outputs a solution with objective* $z \geq (1-\epsilon)z^*$. *The complexity of the algorithm is* $O(\frac{1}{\epsilon} mn \log(nc_{\max}/c_{\min}))$

Proposition

The profit rounding and scaling algorithm can be invoked repeatedly to output a solution with objective $z \geq (1-\epsilon)z^*$ *with a complexity bound of* $O(\frac{1}{\epsilon} m^2 n \log n)$.

FPTAS II: Probability Rounding

Considering an alternative DP:

$$Z_p(P,i) = \begin{cases} \max_{j \in \delta^-(i)} \{p_{ji} Z_p(P/p_{ji}) + P \cdot c_{ji}\} & i \neq s, P \leq 1 \\ 0 & i = s, P = 1 \\ -\infty & i = s, P \neq 1. \end{cases}$$

Probability Rounding Scheme

Input: $G = (V, E), p, c, \epsilon$

```
1: For each $e \in E$ let $p'_e = (1-\epsilon)^{\lceil \log_{(1-\epsilon)^{1/n}} p_e \rceil / n}$.
2: $z_{p'} = \max_{k=1,\dots,n\lceil \log_{(1-\epsilon)^{1/n}} p_{\min} \rceil} Z_{p'}\left((1-\epsilon)^{k/n}, t\right)$. Let $\pi$ be the
   corresponding path.
```

Output: $z(\pi)$

Probability Rounding Result

Proposition

If there exist $a > 0$ and $k_e \in \mathbb{Z}$, for each $e \in E$, such that $p_e = a^{k_e}$, then the probability-based DP solves the probabilistic all-or-nothing path problem in polynomial time.

Proposition

Our probability rounding algorithm computes a solution with objective at least $(1-\epsilon)z^*$ in time $O(-\frac{1}{\epsilon}mn^2\log(p_{\min}))$.

MINLP Method

- The log-transformed objective is used to formulate a convex MINLP.
- It is linearized using gradient based cuts.

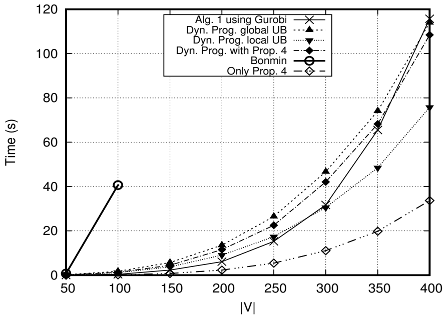
The linearized subproblem given $C \subset \Pi \cup \{\mathbb{1}\}$.

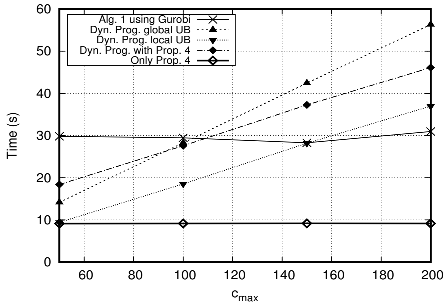
$$
\begin{aligned}
\max_{x,u} \quad & u + \sum_{e\in E} \ln(p_e) x_e && \text{(1a)}\\
\text{subject to} \quad & u - \frac{\sum_{e\in E} c_e x_e}{\sum_{e\in E} c_e \bar{x}_e} - \ln\left(\sum_{e\in E} c_e \bar{x}_e\right) + 1 \leq 0 \quad \bar{x} \in C && \text{(1b)}\\
& x \in \Pi && \text{(1c)}\\
& x \in \{0,1\}^m, && \text{(1d)}
\end{aligned}
$$

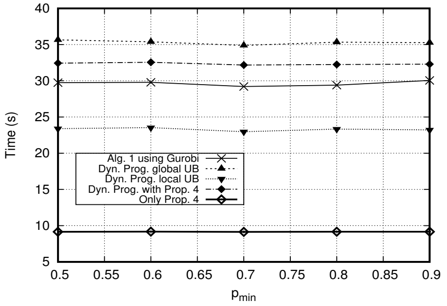
where

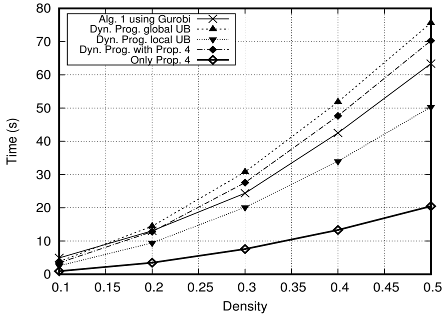
$$
\Pi \equiv \left\{ x \in \mathbb{R}^m_+ \;\middle|\; \begin{matrix} \sum_{(j,i)\in E} x_{ji} = \sum_{(ij)\in E} x_{ij}, & i \in V \setminus \{s,t\} \\ \sum_{(s,i)\in E} x_{si} = 1, & \sum_{(j,t)\in E} x_{jt} = 1 \end{matrix} \right\}.
$$

Instances with “Non-correlated Probabilities”









Partially Successful Path Model

Now suppose that the value of a partially executed project depends on the value of tasks performed up to the point where the first failure occures. The problem becomes:

$$\max_{\pi \in \mathcal{P}(t)} \sum_{(i,j) \in \pi} c_{ij} \prod_{f \in \pi(j)} p_f$$

where $\pi(i)$ is the subpath from s to i.

A similar objective applies to chains of kidney transplant donations with an altruistic donor. For example Dickerson, Procaccia and Sandholm (2013).

FPTAS based on probability rounding can be extended to apply for this problem (in DAGs).

Conclusions

- Maximum probabilistic all-or-nothing problems are NP-hard.
- FPTAS for subsets with running time $O(n^4/\epsilon)$.
- FPTAS or PTAS for probabilistic all-or-nothing with downward closed systems depending on the complexity of the constrained linear problem (PTAS for probabilistic all-or-nothing graph matchings).
- FPTAS methods for probabilistic all-or-nothing paths with running times: $O(\frac{1}{\epsilon} m^2 n \log n)$.
- Partial success problem: FPTAS. Ongoing and future work: complexity of this problem and exact solution techniques.

Thank you!

Questions?

For more information you can also reach me at:
noam.goldberg@biu.ac.il

Online:
N. Goldberg and G. Rudolf. *On the complexity and approximation of the maximum expected value all-or-nothing subset.* arXiv preprint arXiv:1706.07406, 2017.

Submitted:
N. Goldberg and M. Poss. *FPTAS for Acyclic Discounted Utility and Probabilistic Critical Path Problems*. Submitted 2018.

Network Path Interdiction

- A two player game between an interdictor and a smuggler.

Network Path Interdiction

- A two player game between an interdictor and a smuggler.
- The smuggler is a network user who attempts to traverse it from designated source(s) to destination(s) with highest reliability or profit.

Network Path Interdiction

- A two player game between an interdictor and a smuggler.
- The smuggler is a network user who attempts to traverse it from designated source(s) to destination(s) with highest reliability or profit.
- The interdictor is typically an enforcement agency who would like to prevent (interdict) the use of the network by the smuggler.

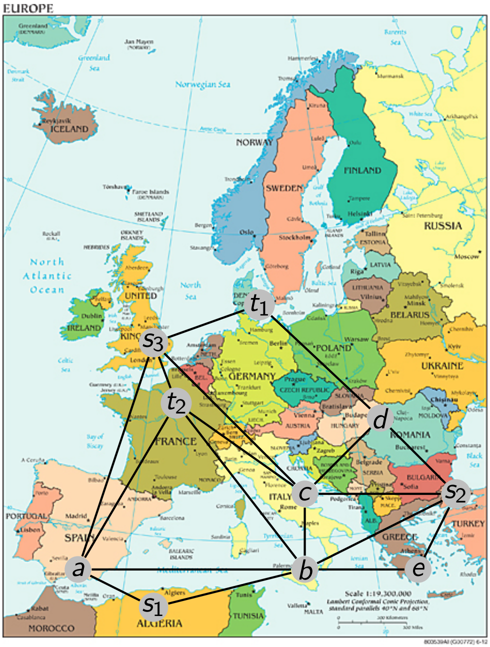
Network Path Interdiction

- A two player game between an interdictor and a smuggler.
- The smuggler is a network user who attempts to traverse it from designated source(s) to destination(s) with highest reliability or profit.
- The interdictor is typically an enforcement agency who would like to prevent (interdict) the use of the network by the smuggler.
- Interdiction efforts are subject to a budget constraint.

Network Path Interdiction

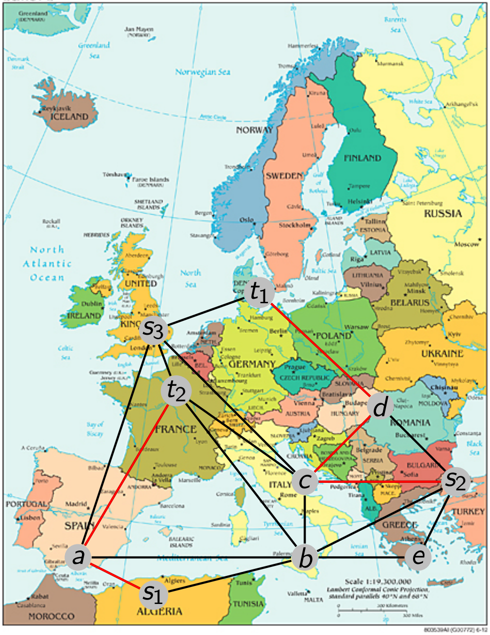
- A two player game between an interdictor and a smuggler.
- The smuggler is a network user who attempts to traverse it from designated source(s) to destination(s) with highest reliability or profit.
- The interdictor is typically an enforcement agency who would like to prevent (interdict) the use of the network by the smuggler.
- Interdiction efforts are subject to a budget constraint.
- Simultaneous moves or Stackelberg.

Terrorist Funding Example



Terrorist Funding Example

EUROPE



The Game Setting and Actions Sets

- Given a graph $G = (V, A)$ with n vertices and m arcs.
- Sources $S \subseteq V$ and destinations $T \subseteq V$.
- $\mathcal{P}$ is an index set of all paths from nodes S to nodes T.
- The interdictor continuously allocates a resource subject to a budget constraint (single resource case) to the arcs

$$X = \left\{ x \in \mathbb{R}_+^m \;\middle|\; \sum_{a \in A} x_a \leq B \right\}.$$

- The smuggler selects a probability distribution on paths

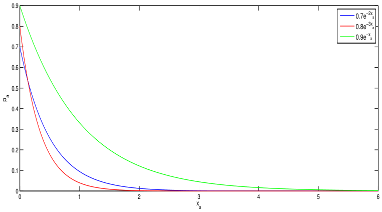
$$Y = \left\{ y \in \mathbb{R}_+^{|\mathcal{P}|} \;\middle|\; \sum_{i \in \mathcal{P}} y_i = 1 \right\}.$$

Nonlinear Arc Evasion Probability

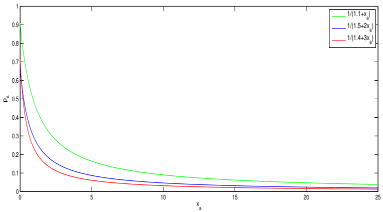
For each arc a the probability of evasion $p_a(x_a)$ depends on the inspection resource x_a:

- $p_a : [0, B] \to [0, 1]$.
- We need to assume that it is logarithmically convex.
- We will also assume $p_a' < 0$.

Examples of log-convex evasion probability functions:



(a) $p_a(z) = \alpha e^{-\beta z}, \quad \alpha, \beta > 0$



(b) $p_a(z) = \frac{1}{\alpha+\beta z}, \quad \alpha \geq 1, \beta > 0$

The Nonzero-Sum Game

- $c(i)$ and $d(i)$ are path i's payoffs, to the smuggler and interdictor, respectively, and that depend on i's source $s(i)$ and destination $t(i)$.
- The utility of the smuggler for $(x, y) \in X \times Y$ is:

$$u_S(x, y) = \sum_{i \in \mathcal{P}} y_i c(i) \prod_{a \in \mathcal{P}_i} p_a(x_a)$$

- The interdictor's utility is given by

$$u_I(x, y) = -\sum_{i \in \mathcal{P}} y_i d(i) \prod_{a \in \mathcal{P}_i} p_a(x_a).$$

Best Responses

- Given x^* the best response of the smuggler is

$$\max_{y \in Y} \left\{ \sum_{i \in \mathcal{P}} y_i c(i) \prod_{a \in \mathcal{P}_i} p_a(x_a^*) \right\} = \max_{i \in \mathcal{P}} \left\{ c(i) \prod_{a \in \mathcal{P}_i} p_a(x_a^*) \right\}.$$

  This is a (multiple-pair) most reliable path problem.

- Given y^* the interdictor's best response is

$$\begin{aligned} \min \quad & \sum_{i \in \mathcal{P}} y_i^* d(i) \prod_{a \in \mathcal{P}_i} p_a(x_a) \\ \text{subject to} \quad & \sum_{a \in A} x_a \leq B \\ & x \geq 0. \end{aligned}$$

Nash Equilibria

We say that $(x^*, y^*) \in X \times Y$ is a Nash equilibrium if

$$x^* \in \text{argmax}_{x \in X}\, u_X(x, y^*) \qquad y^* \in \text{argmax}_{y \in Y}\, u_W(x^*, y)$$

By an early result for convex-concave games (Rosen 1965), since

$u_X(x, y*)$ and $u_Y(x^*, y)$ are concave in x and y, respectively, and X and Y are convex, a Nash equilibrium exists.

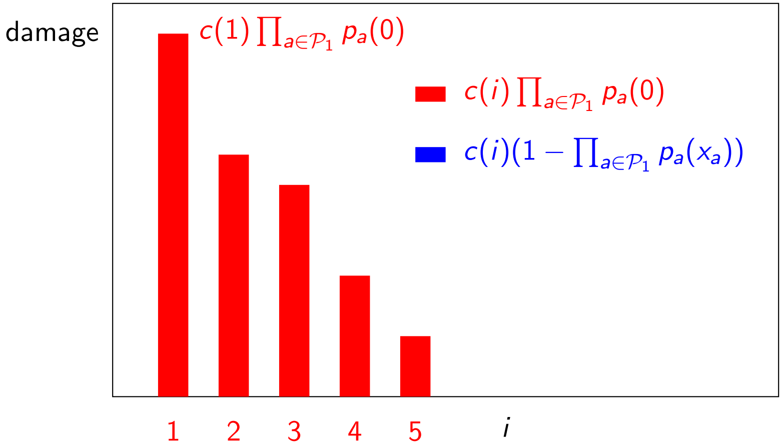
Computing Nash Equilbiria Inspection Allocations

- The smuggler selecting the most profitable paths in expectation, and the interdictor's payoff increasing in the resource allocated to this paths, suggests that the minmax allocation yields a NE (formally proved in paper).

$$\begin{aligned} \min \quad & \theta \\ \text{subject to} \quad & c(i) \prod_{a \in \mathcal{P}_i} p_a(x_a) \le \theta && i \in \mathcal{P} \\ & \sum_{a \in A} x_a \le B \\ & x \ge 0 \end{aligned}$$

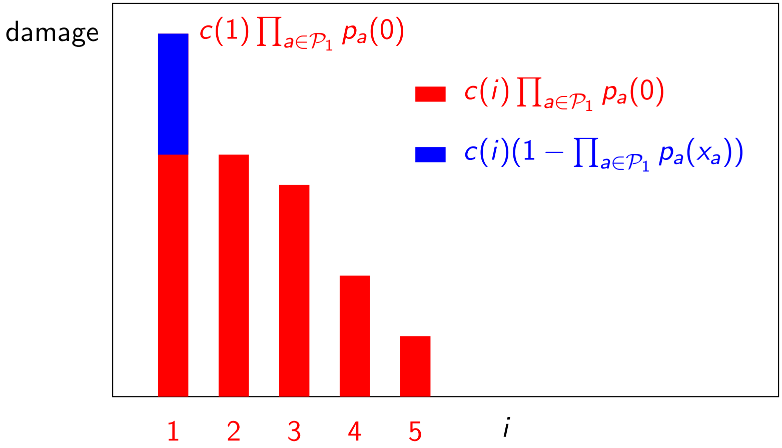
Nash Equilibrium Allocation Intuition

- increase x_a for $a \in \mathcal{P}_1$,



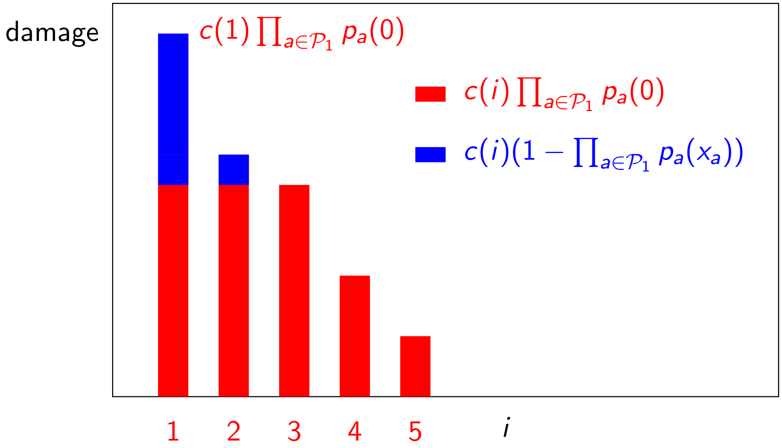
Nash Equilibrium Allocation Intuition

- increase x_a for $a \in \mathcal{P}_1$,



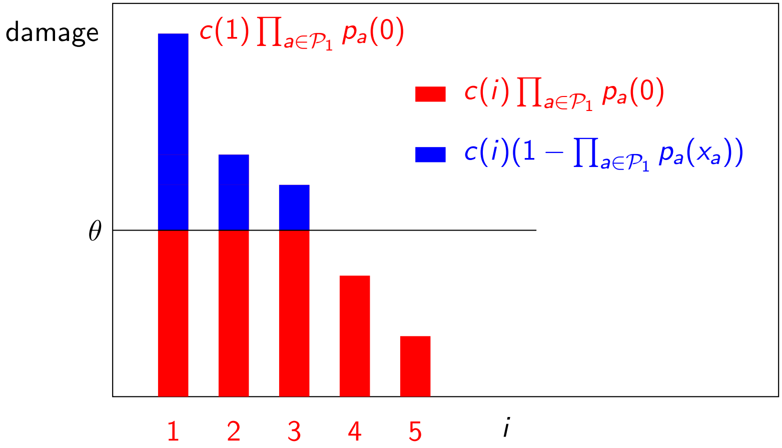
Nash Equilibrium Allocation Intuition

- increase x_a for $a \in \mathcal{P}_1$,
- increase x_a for $a \in \mathcal{P}_1 \cup \mathcal{P}_2$,



Nash Equilibrium Allocation Intuition

- increase x_a for $a \in \mathcal{P}_1$,
- increase x_a for $a \in \mathcal{P}_1 \cup \mathcal{P}_2$,
- increase x_a for $a \in \mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{P}_3$ until the budget is exhausted.



Solving for y^* given x^*

$$\begin{array}{lcll}
\theta^* \displaystyle\sum_{i \in \mathcal{P}^* : a \in \mathcal{P}_i} \frac{d(i)}{c(i)} y_i & = & \dfrac{-p_a(x_a^*)\lambda}{p_a'(x_a^*)}, & a \in A : x_a^* > 0 \\
\theta^* \displaystyle\sum_{i \in \mathcal{P}^* : a \in \mathcal{P}_i} \frac{d(i)}{c(i)} y_i & \leq & \dfrac{-p_a(x_a^*)\lambda}{p_a'(x_a^*)}, & a \in A : x_a^* = 0 \\
\displaystyle\sum_{i \in \mathcal{P}^*} y_i & = & 1 & \\
y_i & = & 0, & c(i) \displaystyle\prod_{a \in \rangle} p_a(x_a^*) < \theta^* \\
y, \lambda \geq 0. & & &
\end{array}$$

- Inspect a only if $\exists i$: $\mathcal{P}_i \ni a$ and $c(i) \prod_{a \in \rangle} p_a(x_a^*) = \theta^*$.
- For an inspected arc $a \in A$ used by a single path i:

$$y_i = -\frac{c(i)}{d(i)} \frac{p_a(x_a^*)\lambda}{p_a'(x_a^*)\theta^*}.$$

Solving for $y*$ given x^*

Proposition

Suppose that (x^*, θ^*) is optimal for the minmax formulation, and $\widehat{P} \subseteq \mathcal{P}^*$ is a given support and values of Lagrange multipliers $\nu^{|\mathcal{P}|}$ of the path constraints. Then y^* given by

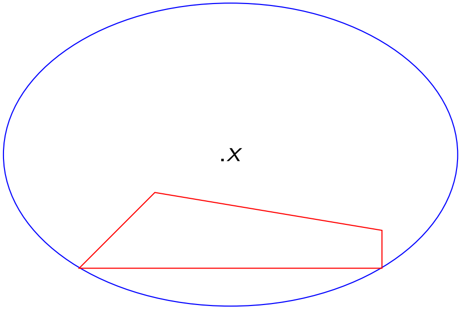
$$y_i^* = \begin{cases} \dfrac{\frac{c(i)}{d(i)}\nu_i}{\sum_{k \in \widehat{P}} \nu_k \frac{c(k)}{d(k)}} & i \in \widehat{\mathcal{P}}. \\ 0 & i \in \mathcal{P} \setminus \widehat{P} \end{cases}$$

together with x^* forms a Nash equilibrium (x^*, y^*).

Efficient Solution for x^*

The minmax problem has an exponential number of constraints.

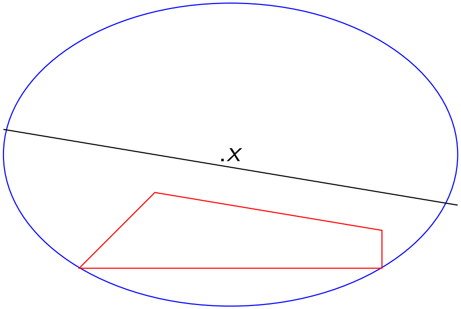
- The Ellipsoid method (Shor; Nemirovski & Yudin; Kachiyan) solves convex optimization/feasibility problems.
- Does not require writing down all the constraints.
- Polynomial time as long as the separation problem is solved in polynomial time.



Efficient Solution for x^*

The minmax problem has an exponential number of constraints.

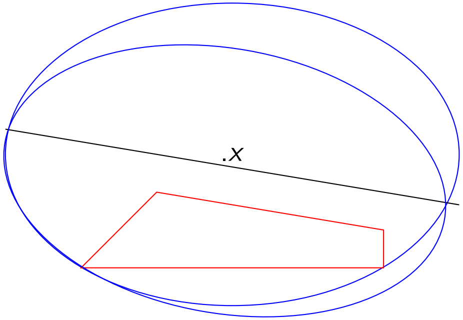
- The Ellipsoid method (Shor; Nemirovski & Yudin; Kachiyan) solves convex optimization/feasibility problems.
- Does not require writing down all the constraints.
- Polynomial time as long as the separation problem is solved in polynomial time.



Efficient Solution for x^*

The minmax problem has an exponential number of constraints.

- The Ellipsoid method (Shor; Nemirovski & Yudin; Kachiyan) solves convex optimization/feasibility problems.
- Does not require writing down all the constraints.
- Polynomial time as long as the separation problem is solved in polynomial time.



The Separation, (Multi-Pair) Most Reliable Path Problem

```
Input: Graph $G = (V, A)$, arc probabilities $p_a(x_a^*)$ for $a \in A$
1: if $|T| < |S|$ then
2:     Reorient the graph from $T$ to $S$ and swap the two.
3: end if
4: Compute an MRPP tree from each $s \in S$ in $G$ with arc
   probabilities $p_a(x_a^*)$.
   Let $\bar{\mathcal{P}}$ denote the union of these paths over all $s \in S'$ and $t \in T$.
5: $\{i^*\} \leftarrow \min \left\{ \operatorname{argmax}_{i \in \bar{\mathcal{P}}} \left\{ c(i) \prod_{a \in \mathcal{P}_i} p_a(x_a^*) \right\} \right\}$
Output: Path $i^* \in \mathcal{P}$
```

The overall running time is determined by the invocation of $\min\{|S|, |T|\}$ runs of a variant of Dijkstra's algorithm, which overall is $O(\min\{|S|, |T|\}(m + n \log n)) \subseteq O(n(m + n \log n))$.

Complexity of determining x^* with general log-convex p_a

Proposition

The minmax problem can be solved to within an additive error of $\epsilon \in (0,1)$ in polynomial time in m, $\log B$, $|\log \epsilon|$. In particular, the ϵ-solution can be determined in

$$N(\epsilon) \leq \left\lceil 2m^2 \left[2 + \log(m+1) + \log\left(\frac{\epsilon + B}{\epsilon}\right)\right]\right\rceil + 1$$

iterations of the Ellipsoid algorithm with a total runtime complexity bound of $O\left(m^4(\log n + \log\left(\frac{B}{\epsilon}\right))\right)$.

Nonlinear Matching Interdiction

- In a matching setting a smuggler would like to match each $s \in S$ with a $t \in T$.
- Under separate pair interdiction the smuggler's utility becomes

$$u_S(x,y) = \sum_{i \in \mathcal{M}} y_i \sum_{(s,t) \in \mathcal{M}_i} c(s,t) \max_{P \in \mathcal{P}(s,t)} \prod_{a \in P} p_a(x),$$

  where $\mathcal{M}$ is the index set of all matchings and $\mathcal{M}_i$ is the set of pairs associated with matching i.
- In an "all-or-nothing" setting, the smuggler's utility function is

$$u_S(x,y) = \sum_{i \in \mathcal{M}} y_i \left(\sum_{(s,t) \in \mathcal{M}_i} c(s,t)) \prod_{(s,t) \in \mathcal{M}_i} \max_{P \in \mathcal{P}(s,t)} \prod_{a \in P} p_a(x) \right).$$

Nonlinear Matching Interdiction

- In a matching setting a smuggler would like to match each $s \in S$ with a $t \in T$.
- Under separate pair interdiction the smuggler's utility becomes

$$u_S(x,y) = \sum_{i \in \mathcal{M}} y_i \sum_{(s,t) \in \mathcal{M}_i} c(s,t) \max_{P \in \mathcal{P}(s,t)} \prod_{a \in P} p_a(x),$$

  where $\mathcal{M}$ is the index set of all matchings and $\mathcal{M}_i$ is the set of pairs associated with matching i.
- In an "all-or-nothing" setting, the smuggler's utility function is

$$u_S(x,y) = \sum_{i \in \mathcal{M}} y_i \left(\sum_{(s,t) \in \mathcal{M}_i} c(s,t)) \prod_{(s,t) \in \mathcal{M}_i} \max_{P \in \mathcal{P}(s,t)} \prod_{a \in P} p_a(x) \right).$$