Beating Simplex for fractional packing and covering linear programs

Christos Koufogiannakis and Neal E. Young
University of California, Riverside

March 9, 2009

G&K's sublinear-time algorithm for zero-sum games

Theorem (Grigoriadis and Khachiyan, 1995)

Given a two-player zero-sum $m \times n$ matrix game A with payoffs in $[-1, 1]$, near-optimal mixed strategies can be computed in time

$$O((m + n)\log(mn)/\varepsilon^2).$$

Each strategy gives expected payoff within additive ε of optimal.

Matrix has size $m \times n$, so for fixed ε this is **sublinear** time.

The algorithm can be viewed as fictitious play,
where each player plays randomly from a distribution.
The distribution gives more weight to pure strategies
that are good responses to opponent's historical average play.

Takes $O(\log(mn)/\varepsilon^2)$ rounds, each round takes $O(m + n)$ time.

G&K's sublinear-time algorithm for zero-sum games

Theorem (Grigoriadis and Khachiyan, 1995)

Given a two-player zero-sum $m \times n$ matrix game A with payoffs in $[-1, 1]$, near-optimal mixed strategies can be computed in time

$$O((m + n) \log(mn)/\varepsilon^2).$$

Each strategy gives expected payoff within **additive** *ε of optimal.*

Matrix has size $m \times n$, so for fixed ε this is **sublinear** time.

The algorithm can be viewed as fictitious play,
where each player plays randomly from a distribution.
The distribution gives more weight to pure strategies
that are good responses to opponent's historical average play.

Takes $O(\log(mn)/\varepsilon^2)$ rounds, each round takes $O(m + n)$ time.

How do LP algorithms do in practice?

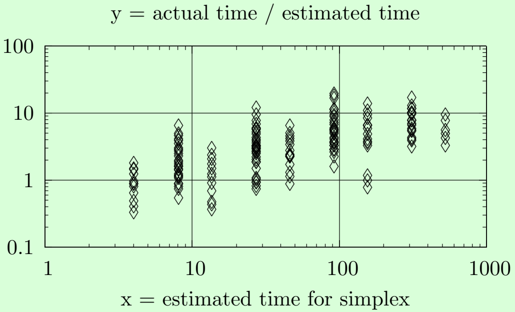
Simplex, interior-point methods, ellipsoid method

optimistic estimate of Simplex run time (# basic operations):

$$(\#\text{ pivots}) \times (\text{time per pivot}) \quad \approx \quad 5\min(m,n) \times mn$$

m rows, n columns

Empirically, ratio (observed time / this estimate) is in [0.3,20]:



How do LP algorithms do in practice?

Simplex, interior-point methods, ellipsoid method

optimistic estimate of Simplex run time (# basic operations):

$$(\#\ \text{pivots}) \times (\text{time per pivot}) \quad \approx \quad 5 \min(m, n) \times mn$$

m rows, n columns

in terms of number of non-zeroes, N: $(m + n \leq N \leq m\, n)$

- if constraint matrix is dense: time $\Theta(N^{1.5})$
- if constraint matrix is sparse: time $\Theta(N^3)$

This is optimistic — can be slower if numerical issues arise.
Time to find, say, .95-*approximate* solution is comparable.
Time for interior-point seems similar (within constant factors).

We will extend G&K to LPs with non-negative coefficients:

packing: maximize $c \cdot x$ such that $Ax \leq b$; $x \geq 0$

covering: minimize $b \cdot y$ such that $A^{\mathsf{T}} y \geq c$; $y \geq 0$

... solutions with *relative* error ε (harder to compute):

- a feasible x with cost $\geq (1-\varepsilon)OPT$,
- a feasible y with cost $\leq (1+\varepsilon)OPT$, or
- a primal-dual pair (x, y) with $c \cdot x \geq b \cdot y/(1+\varepsilon)$.

But... isn't LP equivalent to solving a zero-sum game?

canonical packing LP		**equivalent game**
maximize $\lvert x\rvert_1$ $$\begin{aligned} Ax &\leq 1 \\ x &\geq 0 \end{aligned}$$	$\iff$	minimize λ $$\begin{aligned} Az &\leq \lambda \\ z &\geq 0 \\ \lvert z\rvert_1 &= 1 \end{aligned}$$
solution x^* (can be large)	$\iff$	solution $\begin{aligned} z^* &= x^*/\lvert x^*\rvert \\ \lambda^* &= 1/\lvert x^*\rvert \end{aligned}$
relative error ε	$\iff$	*additive* error $\varepsilon/\lvert x^*\rvert$

- Straight G&K algorithm (given $A_{ij} \in [0,1]$) requires time

$$|x^*|^2(m+n)\log(m+n)/\varepsilon^2$$

to achieve relative error ε.

Run time it will take us to get relative error ε

Worst-case time: *n = rows, m = columns, N = non-zeros*

$$n + m \leq N \leq nm$$

$$O(N + (n+m)\log(nm)/\varepsilon^2)$$

- This is $O(N)$ (linear) for fixed ε and slightly dense matrices.
- Really? In practice $1/\varepsilon^2$ is a "constant" that matters...

  ... for $\varepsilon \approx 1\%$ down to 0.1%,

  "constant" $1/\varepsilon^2$ is 10^4 to 10^6.

Run time it will take us to get relative error ε

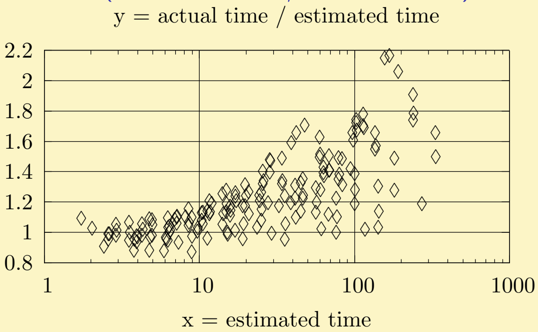
Worst-case time: *n = rows, m = columns, N = non-zeros*

$n + m \leq N \leq nm$

$$O(N + (n+m)\log(nm)/\varepsilon^2)$$

Empirically: about $40N + 12(n+m)\log(nm)/\varepsilon^2$ basic ops

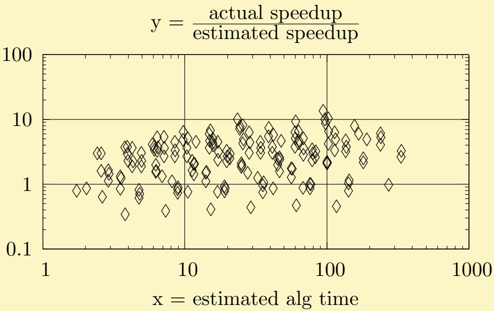
Empirically, ratio of (observed time / this estimate) is in [1,2]:



Estimated speedup versus Simplex ($n \times n$ matrix)

$$\text{estimated speedup} \approx \frac{\text{est. Simplex run time}}{\text{est. algorithm run time}} \approx \frac{\varepsilon^2 n^2}{12 \ln n}$$

Empirically, ratio (observed speedup/this estimate) is in [0.4,10]:



Slower than Simplex for small n, faster than Simplex for large n.

"Hours instead of days, days instead of years."

Estimated speedup versus Simplex ($n \times n$ matrix)

$$\text{estimated speedup} \approx \frac{\text{est. Simplex run time}}{\text{est. algorithm run time}} \approx \frac{\varepsilon^2 n^2}{12 \ln n}$$

- Slower than Simplex for small n, faster for large n.
- Break even at about 900 rows and columns (for $\varepsilon = 1\%$).
- For larger problems, speedup grows proportionally to $n^2 / \ln n$.

"Hours instead of days, days instead of years."

(with $\varepsilon = 1\%$ and 1GHz CPU)

Next (sketch of algorithm):

- canonical forms for packing and covering
- some smooth penalty functions
- simple gradient-based basic packing and covering algorithms
- coupling two algorithms (Grigoriadis & Khachiyan)
- non-uniform increments (Garg & Konemann)
- combining coupling and non-uniform increments (new)
- a random-sampling trick (new) — won't present today

packing and covering, canonical form

$$\text{maximize}_x \frac{|x|_1}{\max_i A_i x} = \text{OPT} = \text{minimize}_y \frac{|y|_1}{\min_j A_j^{\mathsf{T}} y}.$$

A $(1+\varepsilon)$-*approximate primal-dual pair*: $x \geq 0$, $y \geq 0$ with

$$\frac{|x|_1}{\max_i A_i x} \geq (1 - O(\varepsilon)) \times \frac{|y|_1}{\min_j A_j^{\mathsf{T}} y}.$$

A – constraint matrix (rows $i = 1..m$, columns $j = 1..n$)

$|x|$ – size (1-norm), $\sum_j x_j$

$A_i x$ – left-hand side of ith packing constraint

$A_j^{\mathsf{T}} y$ – left-hand side of jth covering constraint

smooth estimates of max and min

Define $\mathrm{smax}(z_1, z_2, \ldots, z_m) = \ln \sum_i e^{z_i}$.

1. smax approximates max within an additive $\ln m$:

$$|\,\mathrm{smax}(z_1, z_2, \ldots, z_m) - \max_i z_i| \;\leq\; \ln m.$$

2. smax is $(1+\varepsilon)$-smooth within an ε-neighborhood:

   If each $d_i \leq \varepsilon$, then

$$\mathrm{smax}(z + d) \leq \mathrm{smax}(z) + (1+\varepsilon)\, d \cdot \nabla \mathrm{smax}(z)$$

analogous estimate of min:

$$\mathrm{smin}(z_1, z_2, \ldots, z_n) = -\ln \sum_i e^{-z_i} \quad \ldots \geq \min_j z_j - \ln n$$

Packing algorithm, assuming each $A_{ij} \in [0,1]$

1. $x \leftarrow 0$
2. while $\max_i A_i x \leq \ln(m)/\varepsilon$ do:
3. Let vector $p = \nabla \mathrm{smax}(Ax)$.
4. Choose j minimizing $A_j^\top p$. *(=derivative of* $\mathrm{smax}\, Ax$ *w.r.t.* x_j*)*
5. Increase x_j by ε.
6. return x (appropriately scaled).

Theorem (e.g. GK,PST,Y,GK,...(??), 1990's)

Alg. returns $(1+O(\varepsilon))$*-approximate packing solution.*

Proof.

In each iteration, since $A_{ij} \in [0,1]$, each $A_i x$ increases by $\leq \varepsilon$.
Using smoothness of smax, show invariant

$$\mathrm{smax}\, Ax \;\leq\; \ln m + (1+O(\varepsilon))\frac{|x|}{\mathrm{OPT}} \cdots \qquad \square$$

Covering algorithm, assuming each $A_{ij} \in [0,1]$

1. $y \leftarrow 0$
2. while $\min_j A_j^\top y \le \ln(n)/\varepsilon$ do:
3. Let vector $q = \nabla \mathrm{smin}(A^\top y)$.
4. Choose i maximizing $A_i q$. *(= derivative of* smin $A^\top y$ *w.r.t.* y_i*)*
5. Increase y_i by ε.
6. return y (appropriately scaled).

Theorem (e.g. GK,PST,Y,GK,...(??), 1990's)

Alg. returns $(1 - O(\varepsilon))$-approximate covering solution.

Proof.

Similar invariant:

$$\mathrm{smin}\, A^\top y \;\ge\; -\ln m + (1 - O(\varepsilon))\frac{|y|}{\mathrm{OPT}} \quad \ldots$$

□

The two algorithms ...

packing

1. $x \leftarrow 0$
2. while $\max_i A_i x \leq \ln(m)/\varepsilon$ do:
3. Let vector $p = \nabla \mathrm{smax}(Ax)$.
4. Choose j minimizing $A_j^{\mathsf{T}} p$.
5. Increase x_j by ε.

covering

1. $y \leftarrow 0$
2. while $\min_j A_j^{\mathsf{T}} y \leq \ln(n)/\varepsilon$ do:
3. Let vector $q = \nabla \mathrm{smin}(A^{\mathsf{T}} y)$.
4. Choose i maximizing $A_i q$.
5. Increase y_i by ε.

The two algorithms ... coupled.

packing	**covering** *(coupled)*
1. $x \leftarrow 0$	1. $y \leftarrow 0$
2. while $\max_i A_i x \le \ln(mn)/\varepsilon$ do:	2. while $\min_j A_j^{\mathsf{T}} y \le \ln(nm)/\varepsilon$ do:
3. Let vector $p = \nabla \text{smax}(Ax)$.	3. Let vector $q = \nabla \text{smin}(A^{\mathsf{T}} y)$.
4. Choose j ~~minimizing~~ $\cancel{A_j^{\mathsf{T}} p}$ randomly from distribution $q/\lvert q \rvert$.	4. Choose i ~~maximizing~~ $\cancel{A_i q}$. randomly from distribution $p/\lvert p \rvert$.
5. Increase x_j by ε.	5. Increase y_i by ε.

The two algorithms ... coupled.

packing	**covering** *(coupled)*
1. $x \leftarrow 0$	1. $y \leftarrow 0$
2. while $\max_i A_i x \le \ln(mn)/\varepsilon$ do:	2. while $\min_j A_j^\mathsf{T} y \le \ln(nm)/\varepsilon$ do:
3. Let vector $p = \nabla \mathrm{smax}(Ax)$.	3. Let vector $q = \nabla \mathrm{smin}(A^\mathsf{T} y)$.
4. Choose j ~~minimizing~~ $\cancel{A_j^\mathsf{T} p}$ randomly from distribution $q/\lvert q\rvert$.	4. Choose i ~~maximizing~~ $\cancel{A_i q}$. randomly from distribution $p/\lvert p\rvert$.
5. Increase x_j by ε.	5. Increase y_i by ε.

Theorem ($\approx$ Grigoriadis & Khachiyan, 1995)

W.h.p., alg. returns $(1+O(\varepsilon))$-approximate primal-dual pair (x, y).

Proof.

Invariants:

$$|x| = |y|$$

in expectation: $$\mathrm{smax}\, Ax \le \ln n + \ln m + (1+O(\varepsilon))\, \mathrm{smin}\, A^\mathsf{T} y$$

$\square$

Why couple?

packing

1. $x \leftarrow 0$
2. while $\max_i A_i x \leq \ln(m)/\varepsilon$ do:
3. Let vector $p = \nabla \text{smax}(Ax)$.
4. Choose j minimizing $A_j^{\mathsf{T}} p$.
5. Increase x_j by ε.

covering

1. $y \leftarrow 0$
2. while $\min_j A_j^{\mathsf{T}} y \leq \ln(n)/\varepsilon$ do:
3. Let vector $q = \nabla \text{smin}(A^{\mathsf{T}} y)$.
4. Choose i maximizing $A_i q$.
5. Increase y_i by ε.

Packing without coupling:

A

0	1	1	0	
0	1	0	1	
1	0	0	1	
1	0	1	0	

Why couple? Consider implementing each iteration...

packing
1. $x \leftarrow 0$
2. while $\max_i A_i x \le \ln(m)/\varepsilon$ do:
3.   Let vector $p = \nabla \mathrm{smax}(Ax)$.
4.   Choose j minimizing $A_j^{\mathsf{T}} p$.
5.   Increase x_j by ε.

covering
1. $y \leftarrow 0$
2. while $\min_j A_j^{\mathsf{T}} y \le \ln(n)/\varepsilon$ do:
3.   Let vector $q = \nabla \mathrm{smin}(A^{\mathsf{T}} y)$.
4.   Choose i maximizing $A_i q$.
5.   Increase y_i by ε.

Packing without coupling:

note: $p_i \propto e^{A_i x}$.

x_j increases

A

		x_3		
0	1	1	0	
0	1	0	1	
1	0	0	1	
1	0	1	0	

Why couple? Consider implementing each iteration...

packing
1. $x \leftarrow 0$
2. while $\max_i A_i x \leq \ln(m)/\varepsilon$ do:
3. Let vector $p = \nabla \text{smax}(Ax)$.
4. Choose j minimizing $A_j^\top p$.
5. Increase x_j by ε.

covering
1. $y \leftarrow 0$
2. while $\min_j A_j^\top y \leq \ln(n)/\varepsilon$ do:
3. Let vector $q = \nabla \text{smin}(A^\top y)$.
4. Choose i maximizing $A_i q$.
5. Increase y_i by ε.

Packing without coupling: *note:* $p_i \propto e^{A_i x}$.

x_j increases

$\implies p_i$ increases for i with $A_{ij} > 0$

A

		x_3		
0	1	1	0	$A_1 x$
0	1	0	1	
1	0	0	1	
1	0	1	0	$A_4 x$

Why couple? Consider implementing each iteration...

packing
1. $x \leftarrow 0$
2. while $\max_i A_i x \le \ln(m)/\varepsilon$ do:
3. Let vector $p = \nabla \mathrm{smax}(Ax)$.
4. Choose j minimizing $A_j^\top p$.
5. Increase x_j by ε.

covering
1. $y \leftarrow 0$
2. while $\min_j A_j^\top y \le \ln(n)/\varepsilon$ do:
3. Let vector $q = \nabla \mathrm{smin}(A^\top y)$.
4. Choose i maximizing $A_i q$.
5. Increase y_i by ε.

Packing without coupling: *note:* $p_i \propto e^{A_i x}$.

x_j increases

$\implies p_i$ increases for i with $A_{ij} > 0$

$\implies A_{j'}^\top p$ increases for many j'.

A

		x_3		
0	1	1	0	$A_1 x$
0	1	0	1	
1	0	0	1	
1	0	1	0	$A_4 x$
$A_1^\top p$	$A_2^\top p$	$A_3^\top p$		

Why couple? Consider implementing each iteration...

packing
1. $x \leftarrow 0$
2. while $\max_i A_i x \le \ln(m)/\varepsilon$ do:
3. Let vector $p = \nabla \mathrm{smax}(Ax)$.
4. Choose j minimizing $A_j^\mathsf{T} p$.
5. Increase x_j by ε.

covering
1. $y \leftarrow 0$
2. while $\min_j A_j^\mathsf{T} y \le \ln(n)/\varepsilon$ do:
3. Let vector $q = \nabla \mathrm{smin}(A^\mathsf{T} y)$.
4. Choose i maximizing $A_i q$.
5. Increase y_i by ε.

Packing without coupling: *note: $p_i \propto e^{A_i x}$.*

x_j increases

$\implies p_i$ increases for i with $A_{ij} > 0$

$\implies A_{j'}^\mathsf{T} p$ increases for many j'.

Update takes time $\Theta(N)$ (=#non-zeros).

A

		x_3		
0	1	1	0	$A_1 x$
0	1	0	1	
1	0	0	1	
1	0	1	0	$A_4 x$
$A_1^\mathsf{T} p$	$A_2^\mathsf{T} p$	$A_3^\mathsf{T} p$		

Why couple? Consider implementing each iteration...

packing	**covering** *(coupled)*
1. $x \leftarrow 0$	1. $y \leftarrow 0$
2. while $\max_i A_i x \le \ln(mn)/\varepsilon$ do:	2. while $\min_j A_j^\top y \le \ln(nm)/\varepsilon$ do:
3. Let vector $p = \nabla \text{smax}(Ax)$.	3. Let vector $q = \nabla \text{smin}(A^\top y)$.
4. Choose j ~~minimizing~~ $\cancel{A_j^\top p}$ randomly from distribution $q/\lvert q \rvert$.	4. Choose i ~~maximizing~~ $\cancel{A_i q}$. randomly from distribution $p/\lvert p \rvert$.
5. Increase x_j by ε.	5. Increase y_i by ε.

Packing without coupling: *note:* $p_i \propto e^{A_i x}$.

x_j increases

$\implies p_i$ increases for i with $A_{ij} > 0$

$\implies A_{j'}^\top p$ increases for many j'.

Update takes time $\Theta(N)$ *(=#non-zeros).*

Packing with coupling:

Maintain only p.

Update takes time $O(m)$ *(=#constraints).*

A

		x_3		
0	1	1	0	$A_1 x$
0	1	0	1	
1	0	0	1	
1	0	1	0	$A_4 x$
$A_1^\top p$	$A_2^\top p$	$A_3^\top p$		

Bounding the iterations ...

packing

1. $x \leftarrow 0$
2. while $\max_i A_i x \leq \ln(m)/\varepsilon$ do:
3. Let vector p by $p_i = e^{A_i x}$.
4. Choose j minimizing $A_j^\mathsf{T} p$
5. Increase x_j by ε.

Bounding the iterations using non-uniform increments

packing (general A)

1. $x \leftarrow 0$
2. while $\max_i A_i x \leq \ln(m)/\varepsilon$ do:
3. Let vector p by $p_i = e^{A_i x}$.
4. Choose j minimizing $A_j^{\mathsf{T}} p$
5. Increase x_j ~~by ε.~~

by δ_j such that max. increase in any $A_i x$ is ε.

Bounding the iterations using non-uniform increments

packing (general A)

1. $x \leftarrow 0$
2. while $\max_i A_i x \leq \ln(m)/\varepsilon$ do:
3. Let vector p by $p_i = e^{A_i x}$.
4. Choose j minimizing $A_j^{\mathsf{T}} p$
5. Increase x_j ~~by ε.~~
   by δ_j such that max. increase in any $A_i x$ is ε.

Theorem (Garg-Konemann, 1998)

Alg. returns $(1 + O(\varepsilon))$-approximate packing solution in at most $m \ln(m)/\varepsilon^2$ iterations. *($m = \#$ packing constraints)*

Bounding the iterations using non-uniform increments

packing (general A)

1. $x \leftarrow 0$
2. while $\max_i A_i x \leq \ln(m)/\varepsilon$ do:
3. Let vector p by $p_i = e^{A_i x}$.
4. Choose j minimizing $A_j^{\mathsf{T}} p$
5. Increase x_j ~~by ε.~~
   by δ_j such that max. increase in any $A_i x$ is ε.

Theorem (Garg-Konemann, 1998)

Alg. returns $(1 + O(\varepsilon))$-approximate packing solution in at most $m \ln(m)/\varepsilon^2$ iterations. *($m = \#$ packing constraints)*

Proof of iteration bound.

Charge each iteration to an increase in some $A_i x$. $\square$

Covering algorithm with non-uniform increments

covering (general A)

1. $y \leftarrow 0$
2. while $\min_j A_j^{\mathsf{T}} y \le \ln(n)/\varepsilon$ do:
3. Let vector q by $q_j = e^{-A_j^{\mathsf{T}} y}$.
4. Choose i maximizing $A_i q$.
5. Increase y_i by δ_i such that max. increase in any $A_j^{\mathsf{T}} y$ is ε.
6. Delete all covering constraints such that $A_j^{\mathsf{T}} y \ge \ln(n)/\varepsilon$.

Theorem (Konemann (?), 1998)

Alg. returns $(1 - O(\varepsilon))$-approximate covering solution in at most $n \ln(n)/\varepsilon^2$ iterations. *($n = \#$ covering constraints)*

Proof (of iteration bound).

Charge each iteration to an increase in some non-deleted $A_j^{\mathsf{T}} y$. $\square$

Coupled algorithm ...

coupled

1. $x \leftarrow 0$, $y \leftarrow 0$
2. while $\max_i A_i x \leq \ln(mn)/\varepsilon$ or $\min_j A_j^{\mathsf{T}} y \leq \ln(mn)/\varepsilon$ do:
3. Let vectors $p = \nabla \mathrm{smax}(Ax)$ and $q = \nabla \mathrm{smin}(A^{\mathsf{T}} y)$.
4. Choose i and j from distributions $p/|p|$ and $q/|q|$, resp.
5. Increase x_j and y_i by ε.

Coupled algorithm ... with non-uniform increments

coupled

1. $x \leftarrow 0$, $y \leftarrow 0$
2. while $\max_i A_i x \leq \ln(mn)/\varepsilon$ or $\min_j A_j^\mathsf{T} y \leq \ln(mn)/\varepsilon$ do:
3. Let vectors $p = \nabla\text{smax}(Ax)$ and $q = \nabla\text{smin}(A^\mathsf{T} y)$.
4. Choose i and j from distributions $p/|p|$ and $q/|q|$, resp.
5. Increase x_j and y_i ~~by ε.~~
   by δ_{ij}, so max increase in any $A_i x$ or $A_j^\mathsf{T} y$ is ε.
6. Delete all covering constraints such that $A_j^\mathsf{T} y \geq \ln(mn)/\varepsilon$.

Coupled algorithm ... with non-uniform increments

coupled

1. $x \leftarrow 0$, $y \leftarrow 0$
2. while $\max_i A_i x \leq \ln(mn)/\varepsilon$ or $\min_j A_j^{\mathsf{T}} y \leq \ln(mn)/\varepsilon$ do:
3. Let vectors $p = \nabla \text{smax}(Ax)$ and $q = \nabla \text{smin}(A^{\mathsf{T}} y)$.
4. Choose i and j from ~~distributions $p/|p|$ and $q/|q|$, resp.~~
   joint distribution $\propto$ $p_i q_j / \delta_{ij}$
5. Increase x_j and y_i ~~by ε.~~
   by δ_{ij}, so max increase in any $A_i x$ or $A_j^{\mathsf{T}} y$ is ε.
6. Delete all covering constraints such that $A_j^{\mathsf{T}} y \geq \ln(mn)/\varepsilon$.

Coupled algorithm ... with non-uniform increments

coupled

1. $x \leftarrow 0,\ y \leftarrow 0$
2. while $\max_i A_i x \le \ln(mn)/\varepsilon$ or $\min_j A_j^{\mathsf{T}} y \le \ln(mn)/\varepsilon$ do:
3. Let vectors $p = \nabla \mathrm{smax}(Ax)$ and $q = \nabla \mathrm{smin}(A^{\mathsf{T}} y)$.
4. Choose i and j from ~~distributions $p/|p|$ and $q/|q|$, resp.~~
   joint distribution $\propto\ p_i q_j/\delta_{ij}$
5. Increase x_j and y_i ~~by ε.~~
   by δ_{ij}, so max increase in any $A_i x$ or $A_j^{\mathsf{T}} y$ is ε.
6. Delete all covering constraints such that $A_j^{\mathsf{T}} y \ge \ln(mn)/\varepsilon$.

Theorem (KY, 2007)

W.h.p., alg. returns $(1+O(\varepsilon))$-approximate primal-dual pair (x, y) in time $O(N + (m+n)\log(mn)/\varepsilon^2)$.

(Iterations: $(m+n)\log(mn)/\varepsilon^2$.)

Summary

Grigoriadis and Khachiyan's sublinear-time algorithm for games

+ Garg/Konemann's non-uniform increments

+ a random-sampling trick

Theorem (KY, 2007)

For fractional packing and covering,
solutions with relative error ε
can be computed in time proportional to

$$\#\textit{non-zeros} \; + \; \frac{(\#\textit{rows} + \textit{cols}) \log(\#\textit{non-zeros})}{\varepsilon^2}.$$

"Hours instead of days, days instead of years."

(w/ $\varepsilon = 0.01$ and 1GHz CPU)

Possible directions

- positive LPs with both packing *and* covering constraints?
- improve Luby/Nisan's parallel algorithm (1993) to $1/\varepsilon^3$?
- extend to implicitly defined problems,
  e.g. multicommodity flow?

Comments? Questions?