Exact Algorithms Via Monotone Local Search

$c^k \Rightarrow (2 - \frac{1}{c})^n$

Fedor Fomin, Serge Gaspers, Daniel Lokshtanov, Saket Saurabh

Institute for Computer Science and Control, Hungarian Academy of Sciences, Budapest, Hungary (MTA SZTAKI)

FPT seminar, July 4, 2016

Should someone mainly interested in exact algorithms care about parameterized complexity?

- Well yes, parameterized algorithms *are* exact algorithms...
- ... and a c^k FPT algorithm gives a c^n algorithm, if $k \leqslant n$

Should someone mainly interested in exact algorithms care about parameterized complexity?

- Well yes, parameterized algorithms *are* exact algorithms...
- ... and a c^k FPT algorithm gives a c^n algorithm, if $k \leqslant n$
- More interestingly, when $c < 4$,

$$\max_{0 \leqslant \alpha \leqslant 1} \{min\{c^\alpha, 2^{H(\alpha)}\}\} < 2$$

Say you want to solve a subset problem; i.e., select k elements from n to satisfy/optimize a property/objective value.

You may:

- use the best known FPT algorithm in c^k
- exhaustively try all $\binom{n}{k}$

Say you want to solve a subset problem; i.e., select k elements from n to satisfy/optimize a property/objective value.

You may:

- use the best known FPT algorithm in c^k
- exhaustively try all $\binom{n}{k}$

$$\max_{0 \leqslant k \leqslant n} \{min\{c^k, \binom{n}{k}\}\} = \max_{0 \leqslant \alpha \leqslant 1} \{min\{c^{\alpha n}, \binom{n}{\alpha n}\}\}$$

Say you want to solve a subset problem; i.e., select k elements from n to satisfy/optimize a property/objective value.

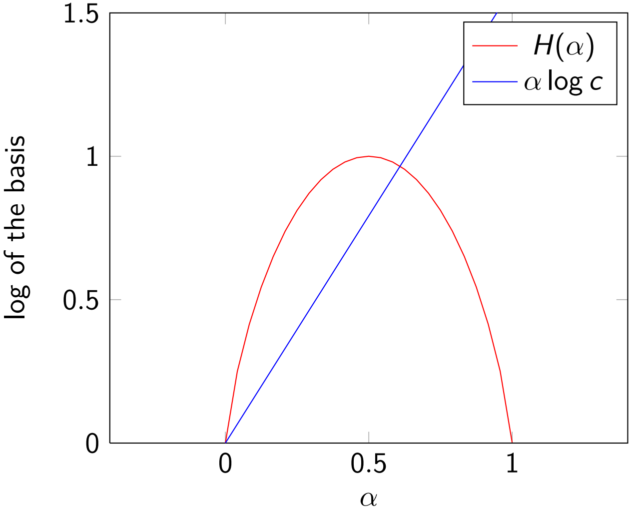
You may:

- use the best known FPT algorithm in c^k
- exhaustively try all $\binom{n}{k}$

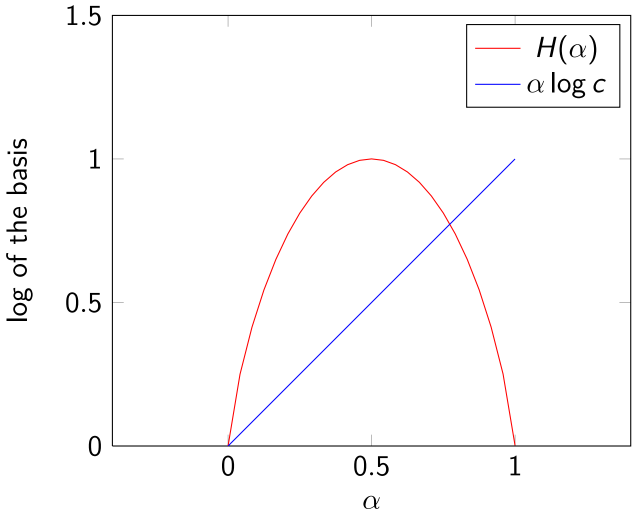
$$\max_{0 \leqslant k \leqslant n} \{min\{c^k, \binom{n}{k}\}\} = \max_{0 \leqslant \alpha \leqslant 1} \{min\{2^{\alpha \log c}, 2^{H(\alpha)}\}\}^n$$

where $H(x) = -x \log x - (1-x) \log(1-x)$

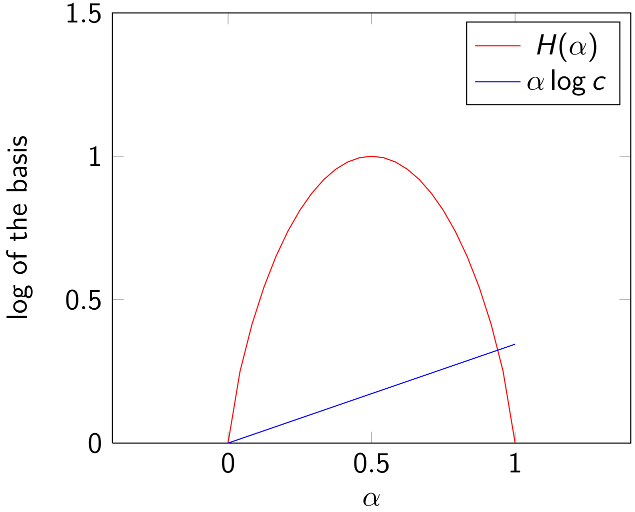
since $\binom{n}{\alpha n} \leqslant 2^{H(\alpha)n}$



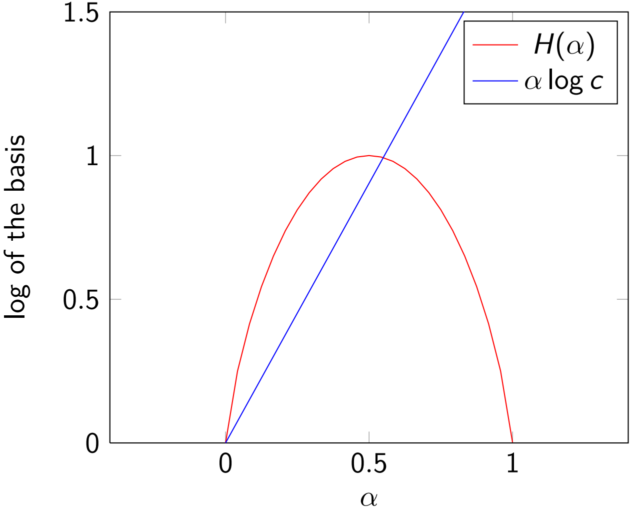
3^k implies 1.953^n



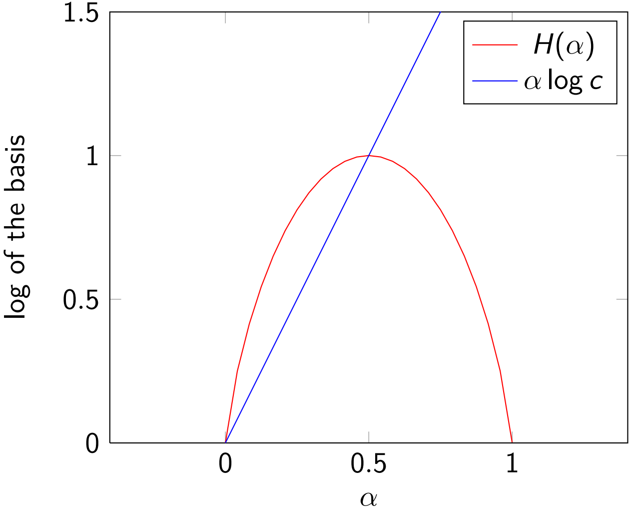
2^k implies 1.709^n



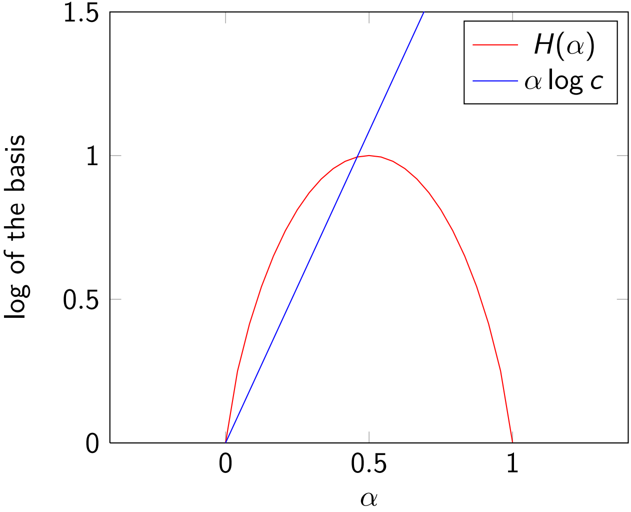
1.27^k implies 1.253^n



3.5^k implies 1.991^n



4^k implies 2^n



4.5^k implies nothing

A very simple algorithm inspired by local search

Let Π be a *subset* problem.

- Guess the size of the optimal solution k.
- Select $t \leqslant k$ elements uniformly at random.
- Complete the solution with $k - t$ elements in FPT time c^{k-t}.

A very simple algorithm inspired by local search

Let Π be a *subset* problem.

- Guess the size of the optimal solution k.
- Select $t \leqslant k$ elements uniformly at random.
- Complete the solution with $k - t$ elements in FPT time c^{k-t}.

Slight caveat: FPT algorithm for the *extension version* of Π.

Analysis

Probablity of guessing t *good* elements $\geqslant$

Analysis

Probablity of guessing t *good* elements $\geqslant \frac{\binom{k}{t}}{\binom{n}{t}}$

Analysis

Probablity of guessing t *good* elements $\geqslant \frac{\binom{k}{t}}{\binom{n}{t}}$

running time $\leqslant \max_{0 \leqslant k \leqslant n} \min_{0 \leqslant t \leqslant k} \frac{\binom{n}{t}}{\binom{k}{t}} c^{k-t}$

Analysis

Probablity of guessing t *good* elements $\geqslant \frac{\binom{k}{t}}{\binom{n}{t}}$

running time $\leqslant \max_{0 \leqslant k \leqslant n} \min_{0 \leqslant t \leqslant k} \frac{\binom{n}{t}}{\binom{k}{t}} c^{k-t}$

So?

Analysis

Probablity of guessing t *good* elements $\geqslant \frac{\binom{k}{t}}{\binom{n}{t}}$

running time $\leqslant \max\limits_{0 \leqslant k \leqslant n} \min\limits_{0 \leqslant t \leqslant k} \frac{\binom{n}{t}}{\binom{k}{t}} c^{k-t}$

So?

$\max\limits_{0 \leqslant k \leqslant n} \min\limits_{0 \leqslant t \leqslant k} \frac{\binom{n}{t}}{\binom{k}{t}} c^{k-t} \leqslant (2 - \frac{1}{c})^n$ (whiteboard)

Breakthrough!

Problem Name	Parameterized		New bound	Previous Bound	
Feedback Vertex Set	3^k (r)	[13]	1.6667^n (r)		
Feedback Vertex Set	3.592^k	[28]	1.7217^n	1.7347^n	[21]
Subset Feedback Vertex Set	4^k	[42]	1.7500^n	1.8638^n	[19]
Feedback Vertex Set in Tournaments	1.6181^k	[30]	1.3820^n	1.4656^n	[30]
Group Feedback Vertex Set	4^k	[42]	1.7500^n	NPR	
Node Unique Label Cover	$\|\Sigma\|^{2k}$	[42]	$(2 - \frac{1}{\|\Sigma\|^2})^n$	NPR	
Vertex (r, ℓ)-Partization $(r, \ell \leq 2)$	3.3146^k	[3, 29]	1.6984^n	NPR	
Interval Vertex Deletion	8^k	[8]	1.8750^n	$(2 - \varepsilon)^n$ for $\varepsilon < 10^{-20}$	[4]
Proper Interval Vertex Deletion	6^k	[40, 7]	1.8334^n	$(2 - \varepsilon)^n$ for $\varepsilon < 10^{-20}$	[4]
Block Graph Vertex Deletion	4^k	[1]	1.7500^n	$(2 - \varepsilon)^n$ for $\varepsilon < 10^{-20}$	[4]
Cluster Vertex Deletion	1.9102^k	[5]	1.4765^n	1.6181^n	[17]
Thread Graph Vertex Deletion	8^k	[27]	1.8750^n	NPR	
Multicut on Trees	1.5538^k	[26]	1.3565^n	NPR	
3-Hitting Set	2.0755^k	[41]	1.5182^n	1.6278^n	[41]
4-Hitting Set	3.0755^k	[17]	1.6750^n	1.8704^n	[17]
d-Hitting Set $(d \geq 3)$	$(d - 0.9245)^k$	[17]	$(2 - \frac{1}{(d-0.9245)})^n$		[11, 17]
Min-Ones 3-SAT	2.562^k	[31]	1.6097^n	NPR	
Min-Ones d-SAT $(d \geq 4)$	d^k		$(2 - \frac{1}{d})^n$	NPR	
Weighted d-SAT $(d \geq 3)$	d^k		$(2 - \frac{1}{d})^n$	NPR	
Weighted Feedback Vertex Set	3.6181^k	[1]	1.7237^n	1.8638^n	[18]
Weighted 3-Hitting Set	2.168^k	[39]	1.5388^n	1.6755^n	[11]
Weighted d-Hitting Set $(d \geq 4)$	$(d - 0.832)^k$	[17, 39]	$(2 - \frac{1}{d-0.932})^n$		[11]

What we did not talk about

- derandomization with so-called *set-inclusion-families*.
- extension to permissive FPT algorithms.
- can be used for enumeration.