Getting negative approximabiliy results for your favorite problem: a tutorial

Marin Bougeret

Outline

Introduction

Context/Notations

- *NPO*: "standard" opt problems (VC, TSP, MAX SAT..). In particular:
  - given input I of $\Pi \in NPO$, poly to decide if a string s is a solution and to compute its value $m(I,s)$ (denoted $m(s)$)
  - can be max or min problem
  - $opt(I)$ denote the optimal value
- given min problem Π, a poly algorithm A has ratio $\rho \geq 1$ iff $\forall I,\ A(I) \leq \rho(I)opt(I)$ ($A(I) \geq \frac{opt(I)}{\rho(I)}$ for max problem)
- basic classes of problems:
  PTAS (for any $\epsilon > 0$ ratio $(1+\epsilon)$) $\subseteq$ *APX* (ratio c where c is a constant) $\subseteq$ *NPO*

Situation of interest here

- given $\Pi \in NPO$, how getting negative approximability results for Π ? (no ratio ρ (in poly time) unless ..)
- ~~structural theory of approximability~~
- ~~approximability preserving reduction: a tutorial~~

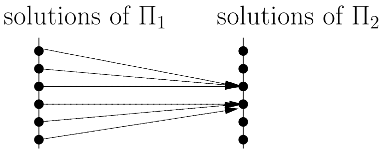
Answer

As expected: by providing reductions:

- chose a Π' hard to approximate (no ρ' for Π' unless ..)
- find a reduction $\Pi' \leq_R \Pi$ that "preserves value of solutions"
- deduce ρ for $\Pi \Rightarrow \rho'$ for Π', and thus no ρ for Π unless ..

Introduction

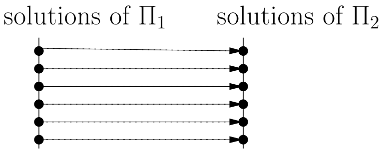
- what does "preserves value of solutions" mean ?
- different scenarios are possible



- which condition $\mathcal{C}$ the reduction should satisfy to transmit a given ratio ?
- let's check exisiting tools

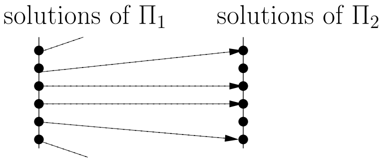
Introduction

- what does "preserves value of solutions" mean ?
- different scenarios are possible



- which condition $\mathcal{C}$ the reduction should satisfy to transmit a given ratio ?
- let's check exisiting tools

- what does "preserves value of solutions" mean ?
- different scenarios are possible



- which condition $\mathcal{C}$ the reduction should satisfy to transmit a given ratio ?
- let's check exisiting tools

Tools: gap reduction Vs approx. preserving reduction

Tool 1: Gap reduction

Tool 1: Gap reduction

Tools: gap reduction Vs approx. preserving reduction

Gap reduction

- extremely natural ($\mathcal{C}$ is natural), powerful (derive no $PTAS$, no APX..), widely used tool

$\Rightarrow$ no need to do a tutorial :)

ATTENTION CORRIGER LA DEF For the sake of completeness: given input (I, k)

- classical Π_{dec}: decide if $Opt(I) \geq k$ or $Opt(I) < k$
- $\Pi_{\rho-gap}$: decide if $Opt(I) \geq k$ or $Opt(I) \leq \frac{k}{\rho(I)}$
- the classical karp reduction between Π'_{dec} and Π_{dec} is replaced by a karp reduction between $\Pi'_{\rho'-gap}$ and $\Pi_{\rho-gap}$
- thus, proving an innapproimability result = proving that $\Pi_{\rho-gap}$ is hard (and thus no ratio $\rho - \epsilon$)

Moreover, thanks to PCP theory, there is a lot of candidate source problems whose hardness is known for a large gap.

Tools: gap reduction Vs approx. preserving reduction

Tool 2: Approximation preserving reduction (short guide in [Cre97])

Tools: gap reduction Vs approx. preserving reduction

Tool 2: Approximation preserving reduction (short guide in [Cre97])

Red.	Ref.	Additional parameters	Constraints to be satisfied	Membership preserved
$\leq_{\text{strict}}$	[34]		$R_A(x, g(x,y)) \leq R_B(f(x), y)$	all
$\leq_{\text{A}}$	[34]	function c	$R_B(f(x),y) \leq r \Rightarrow R_A(x, g(x,y)) \leq c(r)$	APX
$\leq_{\text{P}}$	[34]	function c	$R_B(f(x),y) \leq c(r) \Rightarrow R_A(x, g(x,y)) \leq r$	PTAS
$\leq_{\text{C}}$	[41]	constant α	$R_A(x, g(x,y)) \leq \alpha R_B(f(x), y)$	APX
$\leq_{\text{L}}$	[36]	constants α, β	$\mathsf{opt}_B(f(x)) \leq \alpha \mathsf{opt}_A(x)$ $E_A(x, g(x,y)) \leq \beta E_B(f(x), y)$	PTAS APX if $\text{type}_A = \min$
$\leq_{\text{S}}$	[13]		$\mathsf{opt}_B(f(x)) = \mathsf{opt}_A(x)$ $\mathsf{m}_A(x, g(x,y)) = \mathsf{m}_B(f(x), y)$	all
$\leq_{\text{E}}$	[29]	polynomial p constant β	$\mathsf{opt}_B(f(x)) \leq p(\lvert x \rvert)\, \mathsf{opt}_A(x)$ $R_A(x, g(x,y)) \leq 1 + \beta(R_B(f(x), y) - 1)$	all
$\leq_{\text{PTAS}}$	[18]	ratio r	$R_B(f(x,r),y) \leq c(r) \Rightarrow R_A(x, g(x,y,r)) \leq r$	PTAS
$\leq_{\text{AP}}$	[15]	constant α	$R_B(f(x,r),y) \leq r \Rightarrow R_A(x, g(x,y,r)) \leq 1 + \alpha(r-1)$	all

Tools: gap reduction Vs approx. preserving reduction

Tool 2: Approximation preserving reduction (short guide in [Cre97])

Red.	Ref.	Additional parameters	Constraints to be satisfied	Membership preserved
$\leq_{\text{strict}}$	[34]		$R_A(x, g(x,y)) \leq R_B(f(x), y)$	all
$\leq_{\text{A}}$	[34]	function c	$R_B(f(x), y) \leq r \Rightarrow R_A(x, g(x,y)) \leq c(r)$	APX
$\leq_{\text{P}}$	[34]	function c	$R_B(f(x), y) \leq c(r) \Rightarrow R_A(x, g(x,y)) \leq r$	PTAS
$\leq_{\text{C}}$	[41]	constant α	$R_A(x, g(x,y)) \leq \alpha R_B(f(x), y)$	APX
$\leq_{\text{L}}$	[36]	constants α, β	$\mathsf{opt}_B(f(x)) \leq \alpha \mathsf{opt}_A(x)$ $E_A(x, g(x,y)) \leq \beta E_B(f(x), y)$	PTAS APX if $\text{type}_A = \min$
$\leq_{\text{S}}$	[13]		$\mathsf{opt}_B(f(x)) = \mathsf{opt}_A(x)$ $\mathsf{m}_A(x, g(x,y)) = \mathsf{m}_B(f(x), y)$	all
$\leq_{\text{E}}$	[29]	polynomial p constant β	$\mathsf{opt}_B(f(x)) \leq p(\lvert x \rvert)\, \mathsf{opt}_A(x)$ $R_A(x, g(x,y)) \leq 1 + \beta\,(R_B(f(x), y) - 1)$	all
$\leq_{\text{PTAS}}$	[18]	ratio r	$R_B(f(x,r), y) \leq c(r) \Rightarrow R_A(x, g(x,y,r)) \leq r$	PTAS
$\leq_{\text{AP}}$	[15]	constant α	$R_B(f(x,r), y) \leq r \Rightarrow R_A(x, g(x,y,r)) \leq 1 + \alpha(r-1)$	all

And this is why we will talk about it!

Tools: gap reduction Vs approx. preserving reduction

- in all reduction we must provide a pair (f, g) where f maps instances, g backward maps solutions, both polynomial
- then, depending on the reduction (previous slide) (f, g) must satisfy additional properties.. which are not very "natural"

Unlike Karp or param. reduction, f only depends on I (~~$f(I, k)$~~).

Example: L reduction (Given Π_1 and Π_2 in NPO, max or min)

$\Pi_1 \leq_L \Pi_2$ iff $\exists$ poly (f, g) and $\alpha_1, \alpha_2 > 0 \mid \forall I_1$, $\forall s$ solution of $f(I_1)$:

- $opt_{\Pi_2}(f(I_1)) \leq \alpha_1 opt_{\pi_1}(I_1)$
- $|m_1(g(s)) - opt_{\pi_1}(I_1)| \leq \alpha_2 |m_2(s) - opt_{\pi_2}(f(I_1))|$

Conclusion

- previous reductions have interest for structural theory
- but given Π, and a target class (no PTAS) painful to try each of these reductions

Condition $\mathcal{C}$

- In practice, what do we (I? :) do once our reduction f from Π_1 to Π_2 is defined (even before knowing if we look for gap, or $\leq_*$):
  - given a "good" solution s_1 of I_1 we show that a "good" solution s_2 exists for $f(I_1)$
  - given a "good" solution s_2 of $f(I_1)$ we show that a "good" solution s_1 exists for I_1

Definition of $\mathcal{C}$ for two min problems

f verifies $\mathcal{C}$ for function c_1 and c_2 iff $(I_2 = f(I_1))$:
$\forall t, \exists s_1$ sol of $I_1 \mid m_1(s_1) \leq c_1(t) \Leftrightarrow \exists s_2$ sol of $I_2 \mid m_2(s_2) \leq c_2(t)$

Definition of $\mathcal{C}$ is adapted for any combination of min/max problem by replacing $\leq$ by $\geq$

If even have a poly function that computes s_1 from s_2 (fixme other idrectio important ?) case 2 will imply L reduction Otherwise, the statement is equivalent with "$OPT_1 <= \ldots <=> OPT_2 <= \ldots$",

Condition $\mathcal{C}$: some common cases

Case 1 (Karp reduction)

~~$\forall t$~~ $\exists s_1$ for I_1 st. $m_1(s_1) \leq c_1 \Leftrightarrow \exists s_2$ for I_2 st. $m_2(s_2) \leq c_2$

Case 2

$\forall t$ $\exists s_1$ for I_1 st. $m_1(s_1) \leq p + \alpha t \Leftrightarrow \exists s_2$ for I_2 st. $m_2(s_2) \leq t$
(with possibly $\exists c$ st. $p \leq c \times opt_1(I_1)$)

Case 3

$\forall t$ $\exists s_1$ for I_1 st. $m_1(s_1) \leq t \Leftrightarrow \exists s_2$ for I_2 st. $m_2(s_2) \leq p + \alpha t$
(with possibly $\exists c$ st. $p \leq c \times opt_1(I_1)$)

Condition $\mathcal{C}$: some common cases

- Why these particular functions c_i?: these cases occur in a lot of reductions
- In particular, many L reductions (to show no PTAS) are implicitely proved by using Case 3
- Do not list all the implications for all cases (like "with these values of α, p, min/max problems, case * implies a * reduction") but:
  1. try to prove the equivalence for a pair $c_1(t)$ and $c_2(t)$
  2. then check: if I have ρ_2 for Π_2, then I have $\rho_1 = ..$ for Π_1

Example: consequences of Case 3

Case 3

$\forall t\ \exists s_1$ for I_1 st. $m_1(s_1) \leq t \Leftrightarrow \exists s_2$ for I_2 st. $m_2(s_2) \leq p + \alpha t$
(with possibly $\exists c$ st. $p \leq c\,opt_1(I_1)$)

- Suppose I have a ρ_2 approximate solution algorithm A_2.
- Given input I_1, let $I_2 = f(I_1)$ and $s_2 = A_2(I_2)$.

$$\begin{array}{rcccl} & & s_2 & \leq & \rho_2 OPT(I_2) \\ s_1 & \leq & \frac{s_2 - p}{\alpha} & \leq & \frac{\rho_2(p + \alpha OPT(I_1)) - p}{\alpha} \\ & & & \leq & \rho_2 OPT(I_1) + p\frac{\rho_2 - 1}{\alpha} \end{array}$$

Thus, if $\exists c$ such that $p \leq cOPT(I_1)$ (which is standard):

- PTAS for Π_2 implies PTAS for Π_1
- APX Π_2 implies APX Π_1 (with a different ratio)

If we even want to benefit from structural theory, we can even observe that Case 3 implies a L-reduction. Thus if Π_1 is complete for L-reduction, so is Π_2

Gap vs reduction verfying $\mathcal{C}$

Suppose that we reduce from VC to our min problem Π, and that we have the two following reductions.

Reduction f_1 (gap)

f_1 maps any input (I, k) of Dec_{VC} to an input I' of Π such that

- $VC(I) \leq k \Rightarrow opt(I') \leq n + k$
- $VC(I) \geq k + 1 \Rightarrow opt(I') \geq n + k + 1$

(to be more formal we could say that f_1 maps to an input (I', k) of $gap_{a,b}\Pi$ with $a(I', k) = n + k + 1$ and $b(I', k) = n + k$)

Reduction f_2 (satisfying $\mathcal{C}$)

f_2 maps any input I of VC to an input I' of Π such that
$\forall k, VC(I) \leq k \Leftrightarrow opt(I') \leq n + k$
which is equivalent to: for any k,

- $VC(I) \leq k \Rightarrow opt(I') \leq n + k$
- $VC(I) \geq k + 1 \Rightarrow opt(I') \geq n + k + 1$

Gap vs reduction verfying $\mathcal{C}$

Reduction f_1 (gap)

f_1 maps any input (I, k) of Dec_{VC} to an input I' of Π such that

- $VC(I) \leq k \Rightarrow opt(I') \leq n + k$
- $VC(I) \geq k + 1 \Rightarrow opt(I') \geq n + k + 1$

Reduction f_2 (satisfying $\mathcal{C}$)

f_2 maps any input I of VC to an input I' of Π such that for any k,

- $VC(I) \leq k \Rightarrow opt(I') \leq n + k$
- $VC(I) \geq k + 1 \Rightarrow opt(I') \geq n + k + 1$

Looks the same .. but

- f_1 implies: for any $\epsilon > 0$, no algo that for any I', k has $\frac{a(I',k)}{b(I',k)} - \epsilon$ ratio .. which here give no $(1 + \frac{1}{n+k}) - \epsilon$ (which only tells us no FPTAS)
- f_2 implies no PTAS

Gap vs reduction verfying $\mathcal{C}$

So .. why is f_2 more powerfull ?

Because f_1 depends on k and f_2 does not:

- given I, for each k, f_1 produces an input I'_k (gadgets may depend on k) such that previous equations are satisfied
- given I, f_2 produces an input I' such that previous equations are satisfied for any k

Outline

Vertex Cover in cubic graphs

$VC(\Delta)$: vertex cover pb in graphs of maximum degree Δ.

Known

$VC(4)$ does not admit a PTAS unless P=NP

Theorem [AK97]

$VC(3)$ does not admit a PTAS unless P=NP.
$\rightarrow$ We will prove this using case 3

We could also say (as case 3 $\Rightarrow \leq_L \Rightarrow \leq_{PTAS}$):

Known

$VC(4)$ is APX-complete (for PTAS red)

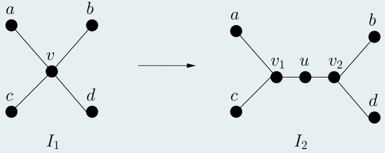
Theorem

$VC(3)$ is APX-complete

Vertex Cover in cubic graphs

Proof: reduction from $VC(4)$

- let I_1 be an instance of $VC(4)$
- we construct I_2 as follows:



- let s be number of deg 4 vertices in I_1
- $\forall t, \exists S_1$ st $|S_1| \leq t \Leftrightarrow \exists S_2$ st $|S_2| \leq t + s$

  $\Rightarrow$ if $d(v) \leq 3$ take v in S_2 iff $v \in S_1$
  if $d(v) = 4$ and $v \in S_1$ take $\{v_1, v_2\} \in S_2$
  if $d(v) = 4$ and $v \notin S_1$ take $\{u\} \in S_2$
- $\exists c$ st $s \leq cOPT(I_1)$ as $OPT(I_1) \geq \frac{n_1-1}{4} \geq \frac{s-1}{4}$

Max Cut

Theorem

Max Cut does not admit a PTAS unless P=NP.
$\rightarrow$ We will prove this using case 3

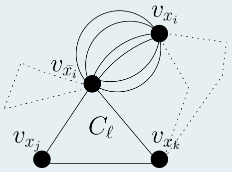
Proof: reduction from MAX NAE 3SAT from [PY88]

MAX NAE 3SAT:

- input: n variables and m clauses on 3 variables (ex $C_\ell = \bar{x}_i \vee x_j \vee x_k$)
- a clause is satisfied iff it has at least one true literal and at least one false literal (ex $x_i = f$, $x_j = t$, $x_k = t$ does not satisfy C_ℓ, but with $x_k = f$ it does)

Proof: from MAX NAE 3SAT to MAX CUT in multigraphs

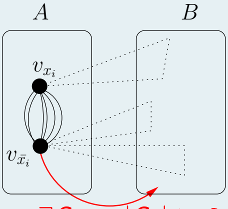
- let I_1 be an instance of MAX NAE 3SAT
- we construct I_2 as follows (we first define a multigraph):



  - for each variable x_i: create two vertices $v_{x_i}, v_{\bar{x}_i}$ with $2k_i$ parallel edges (k_i is the total number of occurences of x_i and $\bar{x}_i$)
  - for each clause C_ℓ: add edges to create a triangle (ex for $C_\ell = \bar{x}_i \vee x_j \vee x_k$, add $\{v_{\bar{x}_i}, v_{x_j}\}, \{v_{x_j}, v_{x_k}\}, \{v_{x_k}, v_{\bar{x}_i}\}$)
- $\forall t$, $\exists S_1$ st $|S_1| \geq t \Leftrightarrow \exists S_2$ st $|S_2| \geq 2t + 2k$ (where $k = \sum_{i=1}^{n} k_i$)

  $\Rightarrow$ each variables adds $2k_i$ edges, each satisfied clause adds 2 edges

Proof: from MAX NAE 3SAT to MAX CUT in multigraphs



- $\forall t,\ \exists S_1$ st $|S_1| \geq t \Leftrightarrow \exists S_2$ st $|S_2| \geq 2t + 2k$ (where $k = \sum_{i=1}^{n} k_i$)
  - $\Leftarrow$ Let A, B be a partition of V.
    - for every i, it is always better to have v_{x_i} and $v_{\bar{x}_i}$ in differents parts: we get $2k$ edges
    - then, each triangle either contributes to 0 or 2 edges
- $\exists c$ st $2k \leq cOPT(I_1)$ as $k = \sum_{i=1}^{n} k_i \leq 3m$ and $OPT(I_1) \geq \frac{3m}{4}$ (fron random assignement)

Thus, MAX CUT in multigraphs does not admit a PTAS unless P=NP.

Max Cut

Proof: from MAX CUT in multigraphs to MAX CUT

- let I_1 be an instance of MAX CUT in multigraphs with m_1 edges
- we construct I_2 of MAX CUT by replacing each edge $e = \{u, v\}$ by a path $P_e = \{u, a_e, b_e, v\}$
- $\forall t$, $\exists S_1$ st $|S_1| \geq t \Leftrightarrow \exists S_2$ st $|S_2| \geq t + 2m_1$ (where m_1 is the number of edges of the mutligraph)
  - $\Rightarrow$ For each edge in the cut in S_1 we get 3 edges in S_2, and for the other edges we get 2 edges. Thus, $|S_2| \geq 3t + 2(m_1 - t)$
  - $\Leftarrow$ Same argument
- $\exists c$ st $2m_1 \leq cOPT(I_1)$ as $OPT(I_1) \geq \frac{m_1}{2}$ (fron random assignement)

Max 3 SAT(B): using expander

Theorem

Max 3SAT(B) (where each literal appears in at most B clauses) does not admit a PTAS unless P=NP.
$\rightarrow$ We will prove this using case 3

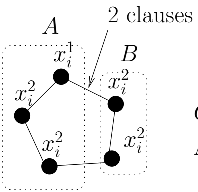
Proof: from MAX 3SAT to MAX 3SAT(B) (from [PY88])

- let I_1 be an instance of MAX 3SAT with n variables and m clauses. Wlog let us suppose that each literal appears (total number of positive and negative apparaitions) c times.
- let us recall the classical Karp reduction:
  - for each variable x_i introduce c variables $x_i^1, \dots, x_i^c$, and add $2c$ clauses $x_i^1 \Leftrightarrow x_i^2, \dots, x_i^c \Leftrightarrow x_i^1$
  - use now copies in the original clauses ($x_i \vee \bar{x}_j \vee x_k$ becomes $x_i^{u_1} \vee x_k^{\bar{u}_2} \vee x_l^{u_3}$)

Max 3 SAT(B): using expander

Proof of the classical Karp reduction

- let G_c be the corresponding graph with c vertices $\{x_i^1, \dots, x_i^c\}$ and $m_{G_c} = c$ following edges: add $\{x_i^u, x_i^v\}$ iff there is a clause with $x_i^u \Leftrightarrow x_i^v$ (G_c is a cycle)
- if all the x_i^ℓ have the same truth value, we get $2m_{G_c}$ satisfied clauses from the variable gadget
- thus: $\exists S_1$ st $|S_1| = m \Leftrightarrow \exists S_2$ st $|S_2| = m + 2nm_{G_c}$



G_c with $c = 5$

A cut of size $x = 2$

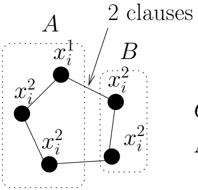
Max 3 SAT(B): using expander

Why does it fail for case 3

- $\forall t$, $\exists S_1$ st $|S_1| \geq t \Leftarrow \exists S_2$ st $|S_2| \geq t + 2nm_{G_c}$ is wrong.

$\Leftarrow$ Tentative proof. Suppose in a sol S_2 that a variable i has n_1 copies set to true and n_2 to false, with $n_1 + n_2 = c$ and $n_1 \leq n_2$.

- The truth values of x_i^ℓ defines a partition X_1, X_2 and a cut of size x
- if we set the n_1 copies to false we get $val(S_2') \geq val(S_2) - |X_1| + x$, and thus we need $x \geq |X_1|$.. not true when G_c is a cycle



G_c with $c = 5$

A cut of size $x = 2$

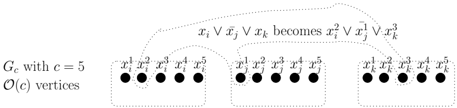
Max 3 SAT(B): using expander

What do we need for G_c

- $\mathcal{O}(c)$ vertices are allowed, with c distinguished vertices (that will appear in the original clauses of MAX 3SAT)
- $\forall$ partition X_1, X_2, at least $\min(s_1, s_2)$ edges in the cut where X_i contains s_i distinguished vertices
- maximum degree B (and thus we can't use a clique)

If we have such a G_c, we get our result for Max 3SAT(B):

- for each variable x_i introduce n_{G_c} variables
- add equivalences between these variables according to G_c
- use the c distinguished copies in the original clauses (we get $m + 2nm_{G_c}$ clauses in the instance of Max 3SAT(B))

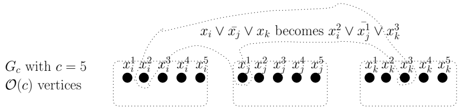


What do we need for G_c

- $\mathcal{O}(c)$ vertices are allowed, with c distinguished vertices (that will appear in the original clauses of MAX 3SAT)
- $\forall$ partition X_1, X_2, at least $\min(s_1, s_2)$ edges in the cut where X_i contains s_i distinguished vertices
- maximum degree B

If we have such a G_c, we get our result for Max 3SAT(B):

- we get $\forall t$, $\exists S_1$ st $|S_1| \geq t \Leftrightarrow \exists S_2$ st $|S_2| \geq t + 2nm_{G_c}$ as it is always better to assign the same values to the n_{G_c} copies of every variable
- $\exists c'$ st $2nm_{G_c} \leq c' OPT(I_1)$ as $nm_{G_c} \leq n\mathcal{O}(c)B$, $nc = 3m$, and $OPT(I_1) \geq \frac{7m}{8}$ (fron random assignement)



Definition

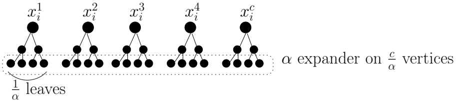
A n vertices graph is a α-expander if every subset S of at most $\frac{n}{2}$ vertices is adj. to $\geq \alpha|S|$ vertices outside S ($cut(S, V \setminus S) \geq \alpha|S|$)

Theorem

There exists a constant $\alpha > 0$ such that for any n there is a α-expander on n vertices with maximum degree 3.

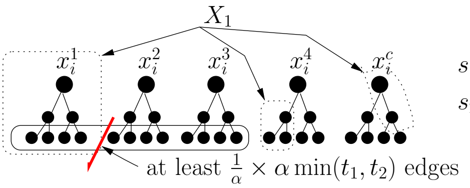
Constructing G_c

- take c disjoint full binary trees with at least $\frac{1}{\alpha}$ leaves each
- connect their leaves in a cubic α expander
- mark the c roots as distinguished nodes



Constructing G_c

- G has $\mathcal{O}(c)$ vertices
- G has constant degree
- let X_1, X_2 a partition and $e = cut(X_1, X_2)$ where X_i contains s_i distinguished nodes
  - let $s_i = t_i + t'_i$ with t_i the number of trees included in X_i
  - $e \geq \frac{1}{\alpha}(\alpha \min(t_1, t_2)) + t'_1 + t'_2 \geq \min(t_1 + t'_1, t_2 + t'_2)$



$s_1 = 2$ with $t_1 = 1$ and $t'_1 = 1$

$s_2 = 3$ with $t_2 = 2$ and $t'_2 = 1$

Outline

A word on structural approximation theory

Example of results in structural theory

- Given a class $\mathcal{C}$, a problem Π (not necessarily in $\mathcal{C}$) and a reduction $\leq_R$, prove that Π is $\mathcal{C}$-complete for $\leq_R$.
  One consequence: Π becomes a candidate to separate classes: if $\mathcal{C}' \subseteq \mathcal{C}$ and $\leq_R$ preserves $\mathcal{C}'$, either $\Pi \notin \mathcal{C}'$, either $\mathcal{C}' = \mathcal{C}$.
- Or $\bar{\mathcal{C}}' = \mathcal{C}$ where $\bar{\mathcal{C}}' = \{\Pi \mid \exists \Pi' \in \mathcal{C}' \mid \Pi' \leq_R \Pi\}$

A bit of history (from [AP05])

$(\leq_R, \mathcal{C}', \mathcal{C}, \Pi)$ means Π is $\mathcal{C}$-complete for $\leq_R$ and $\leq_R$ preserves $\mathcal{C}'$

- $(\leq_S, , min - NPO, minWSAT)$
- $(\leq_S, , max - NPO, maxWSAT)$
- $(\leq_A, APX, NPO, \Pi_1)$
- $(\leq_P, PTAS, APX, \Pi_2)$
- $(\leq_F, FPTAS, PTAS, \Pi_3)$

However, Π_i are artificial problems. Are they classes where complete problems are natural ? Yes: MAX SNP

Max SNP

Definition [KMSV98]

MAX SNP is the class of NPO problems expressible as finding a S which maximizes the objective function

$$f(I, S) = |\{x \mid \phi(I, S, x)\}|$$

where $I = (U, P)$ denotes the input (consisting of a finite universe U and a finite set of bounded arity predicates P), and ϕ is a quantifier-free first order formula.

Example: MAX CUT $\in$ MAX SNP

$f(I, S) = |\{\{u, v\} \mid u \in S \wedge v \notin S \wedge \{u, v\} \in E\}|$ where $I = G$ with $G = (V, E)$

Max SNP

Definition [KMSV98]

MAX SNP is the class of NPO problems expressible as finding a S which maximizes the objective function

$$f(I, S) = |\{x \mid \phi(I, S, x)\}|$$

where $I = (U, P)$ denotes the input (consisting of a finite universe U and a finite set of bounded arity predicates P), and ϕ is a quantifier-free first order formula.

Example: MAX 2 SAT $\in$ MAX SNP

formulation not in MAX SNP:
$f(I, S) = |\{c \mid \exists x((Pos(c, x) \wedge x \in S) \vee (Neg(c, x) \wedge x \notin S))\}|$
where $I = (U, P)$ with $P = \{Pos, Neg\}$

Max SNP

Definition [KMSV98]

MAX SNP is the class of NPO problems expressible as finding a S which maximizes the objective function

$$f(I, S) = |\{x \mid \phi(I, S, x)\}|$$

where $I = (U, P)$ denotes the input (consisting of a finite universe U and a finite set of bounded arity predicates P), and ϕ is a quantifier-free first order formula.

Example: MAX 2 SAT $\in$ MAX SNP

formulation in MAX SNP: $f(I, S) = |\{((x_1, x_2) \mid$
$((x_1, x_2) \in C_0 \Rightarrow (x_1 \in S \vee s_2 \in S))\wedge$
$((x_1, x_2) \in C_1 \Rightarrow (x_1 \notin S \vee s_2 \in S))\wedge$
$((x_1, x_2) \in C_2 \Rightarrow (x_1 \notin S \vee s_2 \notin S))\}|$ where C_i is the set of predicates where the first i variables appear negatively and the $2 - i$ others positively

Max SNP

Nice facts about Max SNP [PY88]

- MAX SNP $\subseteq$ APX (and "easy" proof)
- MAX SNP has several natural complete problems (for $\leq_L$): MAX 3 SAT(B), MAX IS(B), ... (and "easy" proof of first problem hard, MAX 3SAT)

More: see for example [KMSV98].

Conclusion

- a personal roadmap given your favorite problem Π:
  - if you want big inapproximability results, try gap reductions.

    Candidates: IS, VC, Kdm, *SAT, ...
  - if you want no PTAS, try to prove condition of case 3 (even if it could be used for other inapproximaility results).

    Candidates : all problems on cubic graphs, **SAT, ...
    Condition "extra add. factor $\leq cOpt_1(I)$" often easy to get.
- approximation preserving reduction can be used for positive and negative results, but breaks the gap
- please help me finding $L/PTAS$ reduction not using case 3

Bibliography

[AK97] Paola Alimonti and Viggo Kann.
Hardness of approximating problems on cubic graphs.
In Italian Conference on Algorithms and Complexity, pages 288–298. Springer, 1997.

[AP05] Giorgio Ausiello and Vangelis Paschos.
Approximability preserving reduction.
2005.

[Cre97] Pierluigi Crescenzi.
A short guide to approximation preserving reductions.
In Computational Complexity, 1997. Proceedings., Twelfth Annual IEEE Conference on (Formerly: Structure in Complexity Theory Conference), pages 262–273. IEEE, 1997.

[KMSV98] Sanjeev Khanna, Rajeev Motwani, Madhu Sudan, and Umesh Vazirani.
On syntactic versus computational views of approximability.
SIAM Journal on Computing, 28(1):164–191, 1998.

[PY88] Christos Papadimitriou and Mihalis Yannakakis.
Optimization, approximation, and complexity classes.
In Proceedings of the twentieth annual ACM symposium on Theory of computing, pages 229–234. ACM, 1988.