Some Recent Results in Database Theory

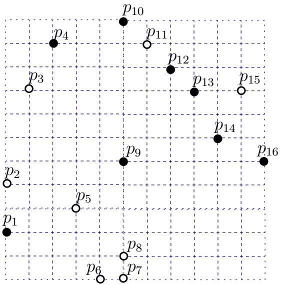
Yufei Tao

CSE Dept
Chinese University of Hong Kong

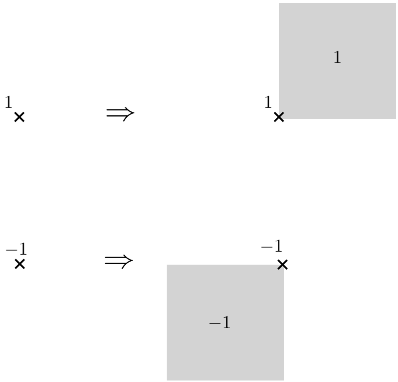
This talk aims to give you a flavor of fundamental database research.

I: Machine Learning

Classification

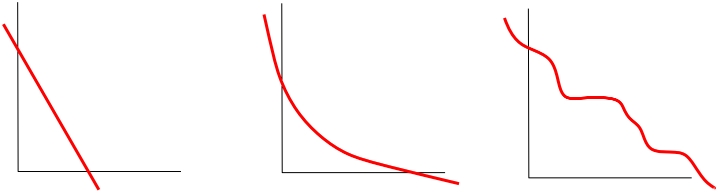


Monotone Classifier



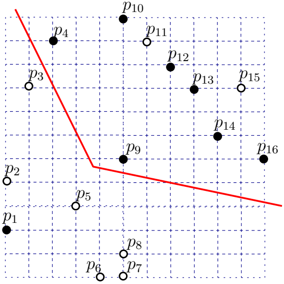
Monotone Classifier

Examples:



$k^* =$ minimum error of all monotone classifiers

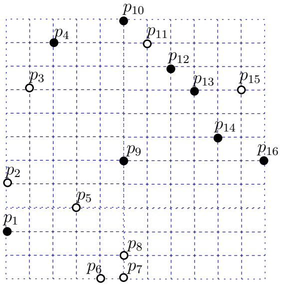
k^* = minimum error of all monotone classifiers



(black = 1, white = -1)

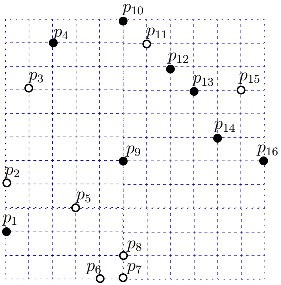
$k^* = 3$

Active Monotone Classification



Question: How many points must we probe to return a monotone classifier whose error is k^*?

Dominance width w of P



$$w = 6$$

Thm 1 [Tao and Wang'21]:
$\Omega(n)$ to ensure error k^*, even in 1D.

Thm 1 [Tao and Wang'21]:
$\Omega(n)$ to ensure error k^*, even in 1D.

Thm 2 [Tao'18]:
$\Omega(w \log \frac{n}{(1+k^*)w})$ to ensure error ck^*, regardless of constant $c > 0$.

Thm 1 [Tao and Wang'21]:
$\Omega(n)$ to ensure error k^*, even in 1D.

Thm 2 [Tao'18]:
$\Omega(w \log \frac{n}{(1+k^*)w})$ to ensure error ck^*, regardless of constant $c > 0$.

Thm 3 [Tao and Wang'21]:
$O(w \log \frac{n}{w} \cdot \log n)$ to ensure error $(1+\epsilon)k^*$, for any constant $\epsilon > 0$.

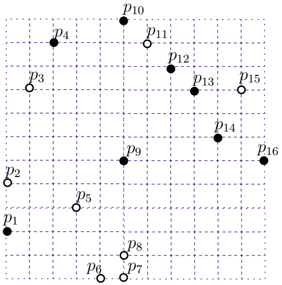
Thm 1 [Tao and Wang'21]:
$\Omega(n)$ to ensure error k^*, even in 1D.

Thm 2 [Tao'18]:
$\Omega(w \log \frac{n}{(1+k^*)w})$ to ensure error ck^*, regardless of constant $c > 0$.

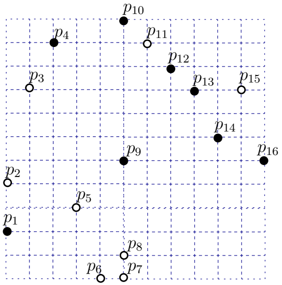
Thm 3 [Tao and Wang'21]:
$O(w \log \frac{n}{w} \cdot \log n)$ to ensure error $(1+\epsilon)k^*$, for any constant $\epsilon > 0$.

Thm 4 [Tao'18]:
$O(w \log \frac{n}{w})$ (expected) probes to ensure (expected) $2k^*$.

2-Approximate Algorithm

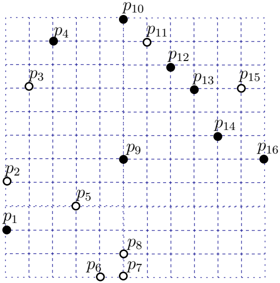


2-Approximate Algorithm



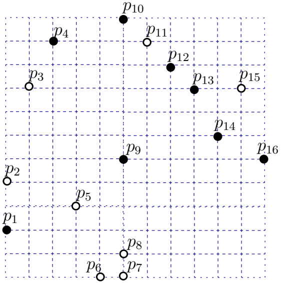
First probe $= p_1$

2-Approximate Algorithm



First probe $= p_1$
Second $= p_8$

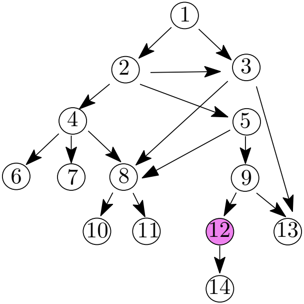
2-Approximate Algorithm



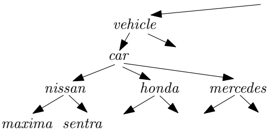
First probe $= p_1$
Second $= p_8$
Returns a classifier with error 5

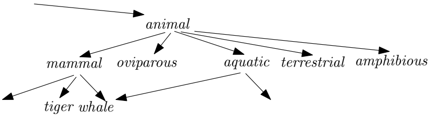
II: Crowdsourcing

Partial Order Search



Crowdsourcing



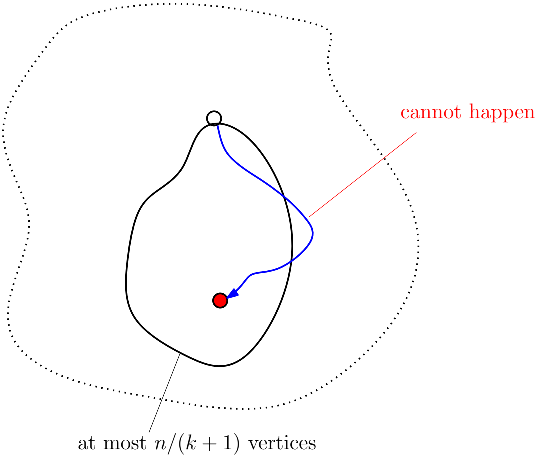


n = number of nodes
d = maximum out-degree
k = number of questions in each round

Thm 1 [Lu, Martens, Niewerth, Tao'21]
$\Omega(\log_{1+k} n + \frac{d}{k} \log_{1+d} n)$ probes necessary.

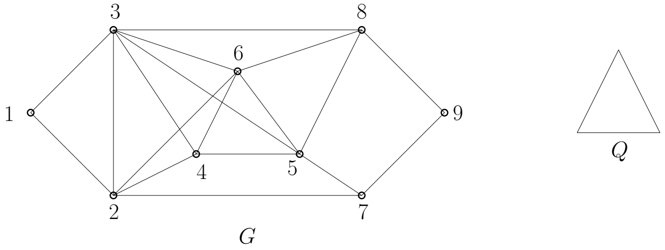
Thm 2 [Lu, Martens, Niewerth, Tao'21]
$O(\log_{1+k} n + \frac{d}{k} \log_{1+d} n)$ probes suffice.

With roughly $1 + d/k$ probes, we can achieve:

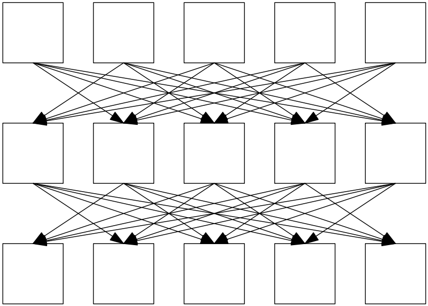


III: Massively Parallel Computation

Subgraph Enumeration

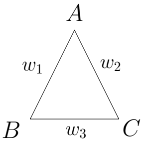


MPC



Goal: Minimize the amount of communication on every machine.

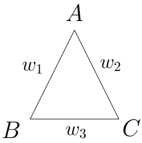
AGM Bound



Minimize $w_1 + w_2 + w_3$ subject to

$$\begin{aligned} w_1 + w_2 &\geq 1 \\ w_1 + w_3 &\geq 1 \\ w_2 + w_3 &\geq 1 \end{aligned}$$

AGM Bound

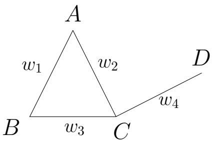


Minimize $w_1 + w_2 + w_3$ subject to

$$\begin{aligned} w_1 + w_2 &\geq 1 \\ w_1 + w_3 &\geq 1 \\ w_2 + w_3 &\geq 1 \end{aligned}$$

Answer: 1.5 — the **fractional edge covering number** ρ

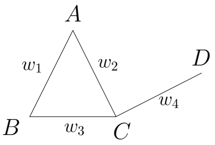
AGM Bound



Minimize $w_1 + w_2 + w_3 + w_4$ subject to

$$\begin{aligned} w_1 + w_2 &\geq 1 \\ w_1 + w_3 &\geq 1 \\ w_2 + w_3 + w_4 &\geq 1 \\ w_4 &\geq 1 \end{aligned}$$

AGM Bound



Minimize $w_1 + w_2 + w_3 + w_4$ subject to

$$
\begin{aligned}
w_1 + w_2 &\geq 1 \\
w_1 + w_3 &\geq 1 \\
w_2 + w_3 + w_4 &\geq 1 \\
w_4 &\geq 1
\end{aligned}
$$

Answer: 2 — the **fractional edge covering number** ρ

AGM Bound

The maximum number of occurrences of Q in a graph of m edges is $\Theta(m^\rho)$.

p = number of machines

Thm 1 At least one machine must communicate $\Omega(m/p^{1/\rho})$.

p = number of machines

Thm 1 At least one machine must communicate $\Omega(m/p^{1/\rho})$.

Every machine sees at most L edges

p = number of machines

Thm 1 At least one machine must communicate $\Omega(m/p^{1/\rho})$.

Every machine sees at most L edges
$\Rightarrow$ Each machine can generate $O(L^{\rho})$ occurrences of Q

p = number of machines

Thm 1 At least one machine must communicate $\Omega(m/p^{1/\rho})$.

Every machine sees at most L edges

$\Rightarrow$ Each machine can generate $O(L^\rho)$ occurrences of Q

$\Rightarrow$ In total, $O(p \cdot L^\rho)$ occurrences

p = number of machines

Thm 1 At least one machine must communicate $\Omega(m/p^{1/\rho})$.

Every machine sees at most L edges
$\Rightarrow$ Each machine can generate $O(L^\rho)$ occurrences of Q
$\Rightarrow$ In total, $O(p \cdot L^\rho)$ occurrences
But in the worst case G has $\Omega(m^\rho)$ occurrences of Q

p = number of machines

Thm 1 At least one machine must communicate $\Omega(m/p^{1/\rho})$.

Every machine sees at most L edges

$\Rightarrow$ Each machine can generate $O(L^\rho)$ occurrences of Q

$\Rightarrow$ In total, $O(p \cdot L^\rho)$ occurrences

But in the worst case G has $\Omega(m^\rho)$ occurrences of Q

$$\Rightarrow p \cdot L^\rho \quad = \quad \Omega(m^\rho)$$

p = number of machines

Thm 1 At least one machine must communicate $\Omega(m/p^{1/\rho})$.

Every machine sees at most L edges

$\Rightarrow$ Each machine can generate $O(L^\rho)$ occurrences of Q

$\Rightarrow$ In total, $O(p \cdot L^\rho)$ occurrences

But in the worst case G has $\Omega(m^\rho)$ occurrences of Q

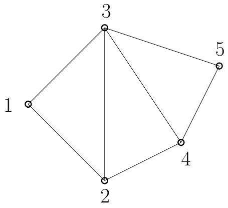
$$\Rightarrow p \cdot L^\rho \quad = \quad \Omega(m^\rho)$$

Thm 2 [Kestman, Suciu, Tao'20]

$\tilde{O}(m/p^{1/\rho})$ communication per machine suffices.

IV: Dynamic Graphs

Triangle Counting



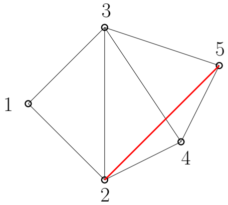
G

3 triangles

Dynamic Triangle Counting

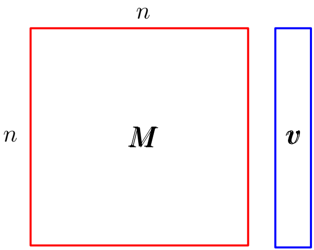
Design a structure to support

- **update**(e): insert/delete e
- **query**: count the number of triangles



G

OMv Conjecture



Need to calculate $\boldsymbol{M}\boldsymbol{v}_i$ for $i = 1, 2, ..., n$.
Can do $O(n^{3-\delta})$?

Dynamic Triangle Counting

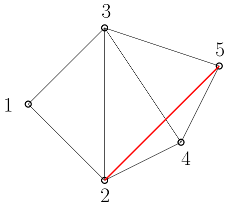
Thm 1 Subject to the OMv conjecture:

- either update needs $\Omega(\sqrt{m})$ time
- or a query needs $\Omega(m)$ time.

Approximate Dynamic Triangle Counting

Design a structure to support

- **update**(e): insert/delete e
- **query**: count the number of triangles up to relative error ϵ

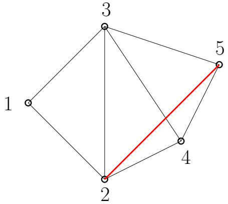


G

Approximate Dynamic Triangle Counting

Design a structure to support

- **update**(e): insert/delete e
- **query**: count the number of triangles up to relative error ϵ



G

Thm 2 Subject to the OMv conjecture, for $\epsilon = 0.49$,

- either update needs $\Omega(\sqrt{m})$ time
- or a query needs $\Omega(m)$ time.

Approximate Dynamic Triangle Counting

(ϵ, Γ)-**guarantee**:

- $\Gamma(m)$: a function of m
- If at least $\Gamma(m)$ triangles, **relative** error ϵ.
  Otherwise, **absolute** error $\epsilon \cdot \Gamma(m)$.

Approximate Dynamic Triangle Counting

(ϵ, Γ)-**guarantee**:

- $\Gamma(m)$: a function of m
- If at least $\Gamma(m)$ triangles, **relative** error ϵ.
  Otherwise, **absolute** error $\epsilon \cdot \Gamma(m)$.

Design a structure to support

- **update**(e)
- **query**: give an estimate with the (ϵ, Γ)-guarantee.

Approximate Dynamic Triangle Counting

Thm 3 [Lu and Tao'21]
Subject to the OMv conjecture, for $\epsilon = 0.49$,

- either update needs $\Omega(\frac{\sqrt{m}}{\Gamma(m)})$ time
- or a query needs $\Omega(m^{2/3})$ time.

Approximate Dynamic Triangle Counting

Thm 3 [Lu and Tao'21]
Subject to the OMv conjecture, for $\epsilon = 0.49$,

- either update needs $\Omega(\frac{\sqrt{m}}{\Gamma(m)})$ time
- or a query needs $\Omega(m^{2/3})$ time.

Thm 4 [Lu and Tao'21]
Can do $\tilde{O}(\frac{\sqrt{m}}{\Gamma(m)})$ per update and $O(1)$ per query for any constant $\epsilon > 0$.

THANK YOU