The Fine-Grained Complexity of Boolean Conjunctive Queries and Sum-Product Problems

Austen Z. Fan   Paraschos Koutris   Hangdong Zhao

University of Wisconsin-Madison

ICALP 2023

Introduction

Venue

Conf.	Type
ICALP	Theory
SODA	Theory
PODS	Theory
SIGMOD	Database
...	...

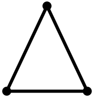
Field

Name	Type
Paris	Theory
Handong	Theory
Austen	Theory
AnHai	Database
...	...

Talk

Name	Conf.
Manuel	ICALP
Moritz	ICALP
Austen	ICALP
Hangdong	PODS
...	...

$$q() :- Venue(Conf., Type), Field(Name, Type), Talk(Name, Conf.)$$



Introduction

BCQ is tightly connected to CSP, FO Logic and many fields in computer science.

Introduction

BCQ is tightly connected to CSP, FO Logic and many fields in computer science.

In this paper, we study the fine-grained complexity of BCQ.

Introduction

BCQ is tightly connected to CSP, FO Logic and many fields in computer science.

In this paper, we study the fine-grained complexity of BCQ.

We introduce the *Clique Embedding Power*, which provides the conditional lower bound $O(\|I\|^{\mathrm{emb}(H)})$.

Prior Work

Prior Work

Upper Bound

Prior Work

Upper Bound

[Yan81] observed that acyclic queries can be answered in linear time.

Prior Work

Upper Bound

[Yan81] observed that acyclic queries can be answered in linear time.

After a long line of research [GM06, Mat07, GLVV12, NPRR12, AGM13, NRR13, Vel14, GM14, KNS16]...

Prior Work

Upper Bound

[Yan81] observed that acyclic queries can be answered in linear time.

After a long line of research [GM06, Mat07, GLVV12, NPRR12, AGM13, NRR13, Vel14, GM14, KNS16]...

[KNS17] showed that a query can be answered in time $\tilde{O}(\|I\|^{\mathrm{subw}(H)})$.

Prior Work

Upper Bound

[Yan81] observed that acyclic queries can be answered in linear time.

After a long line of research [GM06, Mat07, GLVV12, NPRR12, AGM13, NRR13, Vel14, GM14, KNS16]...

[KNS17] showed that a query can be answered in time $\tilde{O}(\|I\|^{\mathrm{subw}(H)})$.

Our clique embedding power is provably always smaller or equal to the submodular width.

Prior Work

Prior Work

Lower Bound

Prior Work

Lower Bound

[Mar13] showed that, assuming ETH, CSP($\mathcal{H}$) is fixed-parameter tractable if and only if $\mathcal{H}$ has bounded submodular width.

Prior Work

Lower Bound

[Mar13] showed that, assuming ETH, CSP($\mathcal{H}$) is fixed-parameter tractable if and only if $\mathcal{H}$ has bounded submodular width.

Our method gives a fine-grained lower bound for every query.

Our Contributions

Our Contributions

- Introduce the notion of the *clique embedding power* $\text{emb}(H)$ and explore its properties; most importantly, we prove that $\text{emb}(H) \leq \text{subw}(H)$.

Our Contributions

- Introduce the notion of the *clique embedding power* $\text{emb}(H)$ and explore its properties; most importantly, we prove that $\text{emb}(H) \leq \text{subw}(H)$.
- Show that computing $\text{emb}(H)$ is decidable and give a Mixed Integer Linear Program formulation.

Our Contributions

- Introduce the notion of the *clique embedding power* $\mathrm{emb}(H)$ and explore its properties; most importantly, we prove that $\mathrm{emb}(H) \leq \mathrm{subw}(H)$.
- Show that computing $\mathrm{emb}(H)$ is decidable and give a Mixed Integer Linear Program formulation.
- Construct a semiring-oblivious reduction from the k-clique problem to any query and derive conditional lower bounds for its running time.

Our Contributions

- Introduce the notion of the *clique embedding power* $\text{emb}(H)$ and explore its properties; most importantly, we prove that $\text{emb}(H) \leq \text{subw}(H)$.
- Show that computing $\text{emb}(H)$ is decidable and give a Mixed Integer Linear Program formulation.
- Construct a semiring-oblivious reduction from the k-clique problem to any query and derive conditional lower bounds for its running time.
- Identify several classes of hypergraphs for which $\text{emb}(H) = \text{subw}(H)$, and a hypergraph with six vertices for which there is a gap between these two quantities.

Sum-of-Product Over Semirings

Sum-of-Product Over Semirings

[GKT07] observed that different database semantics can be unified at the level of Sum-of-Product computation over various semirings.

Sum-of-Product Over Semirings

[GKT07] observed that different database semantics can be unified at the level of Sum-of-Product computation over various semirings.

$$q() : -R_1(\vec{x}_1), R_2(\vec{x}_2), \ldots, R_n(\vec{x}_n)$$

Sum-of-Product Over Semirings

[GKT07] observed that different database semantics can be unified at the level of Sum-of-Product computation over various semirings.

$$q\left(\right) : -R_1\left(\vec{x}_1\right), R_2\left(\vec{x}_2\right), \ldots, R_n\left(\vec{x}_n\right)$$

$$q(I) := \bigvee_{v:\text{valuation}} \bigwedge_{i=1}^{n} R_i(v\left(\vec{x}_i\right))$$

Sum-of-Product Over Semirings

[GKT07] observed that different database semantics can be unified at the level of Sum-of-Product computation over various semirings.

$$q() :- R_1(\vec{x}_1), R_2(\vec{x}_2), \ldots, R_n(\vec{x}_n)$$

$$q(I) := \bigvee_{v:\text{valuation}} \bigwedge_{i=1}^{n} R_i(v(\vec{x}_i))$$

$$q(I) := \bigoplus_{v:\text{valuation}} \bigotimes_{i=1}^{n} R_i(v(\vec{x}_i))$$

Sum-of-Product Over Semirings

[GKT07] observed that different database semantics can be unified at the level of Sum-of-Product computation over various semirings.

$$q() : - R_1(\vec{x}_1), R_2(\vec{x}_2), \ldots, R_n(\vec{x}_n)$$

$$q(I) := \bigvee_{v:\text{valuation}} \bigwedge_{i=1}^{n} R_i(v(\vec{x}_i))$$

$$q(I) := \bigoplus_{v:\text{valuation}} \bigotimes_{i=1}^{n} R_i(v(\vec{x}_i))$$

Example

Sum-of-Product Over Semirings

[GKT07] observed that different database semantics can be unified at the level of Sum-of-Product computation over various semirings.

$$q\left(\right) :- R_1\left(\vec{x}_1\right), R_2\left(\vec{x}_2\right), \ldots, R_n\left(\vec{x}_n\right)$$

$$q(I) := \bigvee_{v:\text{valuation}} \bigwedge_{i=1}^{n} R_i(v\left(\vec{x}_i\right))$$

$$q(I) := \bigoplus_{v:\text{valuation}} \bigotimes_{i=1}^{n} R_i(v\left(\vec{x}_i\right))$$

Example

Set semantics $\leftrightarrow (\{\textsc{True}, \textsc{False}\}, \vee, \wedge)$

Sum-of-Product Over Semirings

[GKT07] observed that different database semantics can be unified at the level of Sum-of-Product computation over various semirings.

$$q\,() : -R_1\,(\vec{x}_1)\,, R_2\,(\vec{x}_2)\,, \ldots, R_n\,(\vec{x}_n)$$

$$q(I) := \bigvee_{v:\text{valuation}} \bigwedge_{i=1}^{n} R_i(v\,(\vec{x}_i))$$

$$q(I) := \bigoplus_{v:\text{valuation}} \bigotimes_{i=1}^{n} R_i(v\,(\vec{x}_i))$$

Example

Set semantics $\leftrightarrow (\{\textsc{True}, \textsc{False}\}, \vee, \wedge)$
Bag semantics $\leftrightarrow (\mathbb{N}, +, *)$

Sum-of-Product Over Semirings

Example

Sum-of-Product Over Semirings

Example

Given an edge-weighted graph $G = (V, \text{weight})$

Sum-of-Product Over Semirings

Example

Given an edge-weighted graph $G = (V, \text{weight})$

Compute $\bigvee_{V' \subseteq V, |V'|=k} \bigwedge_{\{v,w\} \in V'} \text{weight}(\{v, w\}) \leftrightarrow$ Boolean k-clique

Sum-of-Product Over Semirings

Example

Given an edge-weighted graph $G = (V, \text{weight})$

Compute $\bigvee_{V' \subseteq V, |V'|=k} \bigwedge_{\{v,w\} \in V'} \text{weight}(\{v, w\}) \leftrightarrow$ Boolean k-clique

Compute $\min_{V' \subseteq V, |V'|=k} \sum_{\{v,w\} \in V'} \text{weight}(\{v, w\}) \leftrightarrow$ Minimum k-clique

Sum-of-Product Over Semirings

Example

Given an edge-weighted graph $G = (V, \text{weight})$

Compute $\bigvee_{V' \subseteq V, |V'|=k} \bigwedge_{\{v,w\} \in V'} \text{weight}(\{v, w\}) \leftrightarrow$ Boolean k-clique

Compute $\min_{V' \subseteq V, |V'|=k} \sum_{\{v,w\} \in V'} \text{weight}(\{v, w\}) \leftrightarrow$ Minimum k-clique

Compute $\sum_{V' \subseteq V, |V'|=k} \prod_{\{v,w\} \in V'} \text{weight}(\{v, w\}) \leftrightarrow$ Counting k-clique

Sum-of-Product Over Semirings

Example

Given an edge-weighted graph $G = (V, \text{weight})$

Compute $\bigvee_{V' \subseteq V, |V'|=k} \bigwedge_{\{v,w\} \in V'} \text{weight}(\{v, w\}) \leftrightarrow$ Boolean k-clique

Compute $\min_{V' \subseteq V, |V'|=k} \sum_{\{v,w\} \in V'} \text{weight}(\{v, w\}) \leftrightarrow$ Minimum k-clique

Compute $\sum_{V' \subseteq V, |V'|=k} \prod_{\{v,w\} \in V'} \text{weight}(\{v, w\}) \leftrightarrow$ Counting k-clique

Our hardness reduction from k-clique is semiring-oblivious.

Outline

Clique Embedding Power, I

Clique Embedding Power, I

Definition (Touch)

We say $X, Y \subseteq \mathcal{V}$ *touch* in $\mathcal{H}$ if either $X \cap Y \neq \emptyset$ or $\exists e \in \mathcal{E}$ such that $e \cap X \neq \emptyset$ and $e \cap Y \neq \emptyset$.

Clique Embedding Power, I

Definition (Touch)

We say $X, Y \subseteq \mathcal{V}$ *touch* in $\mathcal{H}$ if either $X \cap Y \neq \emptyset$ or $\exists e \in \mathcal{E}$ such that $e \cap X \neq \emptyset$ and $e \cap Y \neq \emptyset$.

Definition (K-Clique Embedding)

A *k-clique embedding* from C_k to $\mathcal{H}$ is a mapping ψ from $[k]$ to $\mathcal{P}(\mathcal{V}) \setminus \emptyset$ such that (1) $\forall v \in [k], \psi(v)$ induces a connected subhypergraph, and (2) $\forall v, u \in [k]$, $\psi(v), \psi(u)$ touch in $\mathcal{H}$.

Clique Embedding Power, I

Definition (Touch)

We say $X, Y \subseteq \mathcal{V}$ *touch* in $\mathcal{H}$ if either $X \cap Y \neq \emptyset$ or $\exists e \in \mathcal{E}$ such that $e \cap X \neq \emptyset$ and $e \cap Y \neq \emptyset$.

Definition (K-Clique Embedding)

A *k-clique embedding* from C_k to $\mathcal{H}$ is a mapping ψ from $[k]$ to $\mathcal{P}(\mathcal{V}) \setminus \emptyset$ such that (1) $\forall v \in [k], \psi(v)$ induces a connected subhypergraph, and (2) $\forall v, u \in [k]$, $\psi(v), \psi(u)$ touch in $\mathcal{H}$.

Example

Clique Embedding Power, I

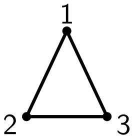
Definition (Touch)

We say $X, Y \subseteq \mathcal{V}$ *touch* in $\mathcal{H}$ if either $X \cap Y \neq \emptyset$ or $\exists e \in \mathcal{E}$ such that $e \cap X \neq \emptyset$ and $e \cap Y \neq \emptyset$.

Definition (K-Clique Embedding)

A *k-clique embedding* from C_k to $\mathcal{H}$ is a mapping ψ from $[k]$ to $\mathcal{P}(\mathcal{V}) \setminus \emptyset$ such that (1) $\forall v \in [k], \psi(v)$ induces a connected subhypergraph, and (2) $\forall v, u \in [k]$, $\psi(v), \psi(u)$ touch in $\mathcal{H}$.

Example



Clique Embedding Power, I

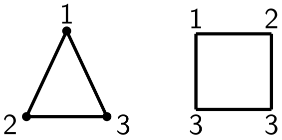
Definition (Touch)

We say $X, Y \subseteq \mathcal{V}$ *touch* in $\mathcal{H}$ if either $X \cap Y \neq \emptyset$ or $\exists e \in \mathcal{E}$ such that $e \cap X \neq \emptyset$ and $e \cap Y \neq \emptyset$.

Definition (K-Clique Embedding)

A *k-clique embedding* from C_k to $\mathcal{H}$ is a mapping ψ from $[k]$ to $\mathcal{P}(\mathcal{V}) \setminus \emptyset$ such that (1) $\forall v \in [k], \psi(v)$ induces a connected subhypergraph, and (2) $\forall v, u \in [k]$, $\psi(v), \psi(u)$ touch in $\mathcal{H}$.

Example



Clique Embedding Power, I

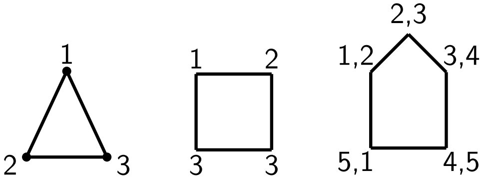
Definition (Touch)

We say $X, Y \subseteq \mathcal{V}$ *touch* in $\mathcal{H}$ if either $X \cap Y \neq \emptyset$ or $\exists e \in \mathcal{E}$ such that $e \cap X \neq \emptyset$ and $e \cap Y \neq \emptyset$.

Definition (K-Clique Embedding)

A *k-clique embedding* from C_k to $\mathcal{H}$ is a mapping ψ from $[k]$ to $\mathcal{P}(\mathcal{V}) \setminus \emptyset$ such that (1) $\forall v \in [k], \psi(v)$ induces a connected subhypergraph, and (2) $\forall v, u \in [k]$, $\psi(v), \psi(u)$ touch in $\mathcal{H}$.

Example



Clique Embedding Power, I

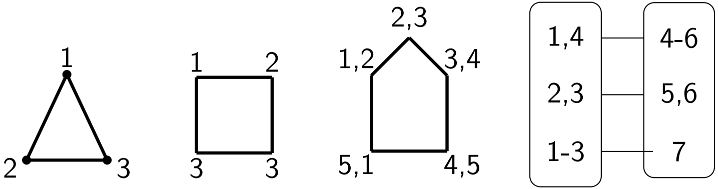
Definition (Touch)

We say $X, Y \subseteq \mathcal{V}$ *touch* in $\mathcal{H}$ if either $X \cap Y \neq \emptyset$ or $\exists e \in \mathcal{E}$ such that $e \cap X \neq \emptyset$ and $e \cap Y \neq \emptyset$.

Definition (K-Clique Embedding)

A *k-clique embedding* from C_k to $\mathcal{H}$ is a mapping ψ from $[k]$ to $\mathcal{P}(\mathcal{V}) \setminus \emptyset$ such that (1) $\forall v \in [k], \psi(v)$ induces a connected subhypergraph, and (2) $\forall v, u \in [k]$, $\psi(v), \psi(u)$ touch in $\mathcal{H}$.

Example



Clique Embedding Power, II

Clique Embedding Power, II

Definition (Weak Edge Depth)

$\forall e$ the *weak edge depth of* e is $d_\psi(e) := |\{v \in [k] \mid \psi(v) \cap e \neq \emptyset\}|$.
The *weak edge depth of* ψ $\mathrm{wed}(\psi) := \max_e d_\psi(e)$.

Clique Embedding Power, II

Definition (Weak Edge Depth)

$\forall e$ the *weak edge depth of* e is $d_\psi(e) := |\{v \in [k] \mid \psi(v) \cap e \neq \emptyset\}|$.
The *weak edge depth of* ψ $\mathrm{wed}(\psi) := \max_e d_\psi(e)$.

Definition (Clique Embedding Power)

The *k-clique embedding power* is $\mathrm{emb}_k(\mathcal{H}) := \max_\psi \frac{k}{\mathrm{wed}(\psi)}$. The *clique embedding power* is $\mathrm{emb}(\mathcal{H}) := \sup_{k \geq 3} \mathrm{emb}_k(\mathcal{H})$.

Clique Embedding Power, II

Definition (Weak Edge Depth)

$\forall e$ the *weak edge depth of* e is $d_\psi(e) := |\{v \in [k] \mid \psi(v) \cap e \neq \emptyset\}|$.
The *weak edge depth of* ψ $\mathrm{wed}(\psi) := \max_e d_\psi(e)$.

Definition (Clique Embedding Power)

The *k-clique embedding power* is $\mathrm{emb}_k(\mathcal{H}) := \max_\psi \frac{k}{\mathrm{wed}(\psi)}$. The *clique embedding power* is $\mathrm{emb}(\mathcal{H}) := \sup_{k \geq 3} \mathrm{emb}_k(\mathcal{H})$.

Example

Clique Embedding Power, II

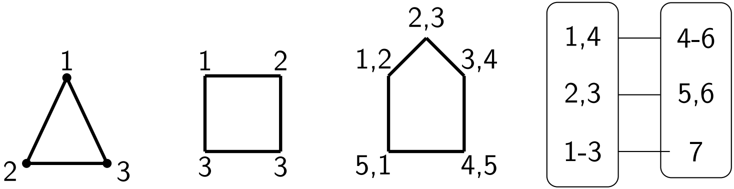
Definition (Weak Edge Depth)

$\forall e$ the *weak edge depth of* e is $d_\psi(e) := |\{v \in [k] \mid \psi(v) \cap e \neq \emptyset\}|$.
The *weak edge depth of* ψ $\mathrm{wed}(\psi) := \max_e d_\psi(e)$.

Definition (Clique Embedding Power)

The *k-clique embedding power* is $\mathrm{emb}_k(\mathcal{H}) := \max_\psi \frac{k}{\mathrm{wed}(\psi)}$. The *clique embedding power* is $\mathrm{emb}(\mathcal{H}) := \sup_{k \geq 3} \mathrm{emb}_k(\mathcal{H})$.

Example



emb $= \frac{3}{2}$ emb $= \frac{3}{2}$ emb $= \frac{5}{3}$ emb $= \frac{7}{4}$

Decidability and MIP formulation, I

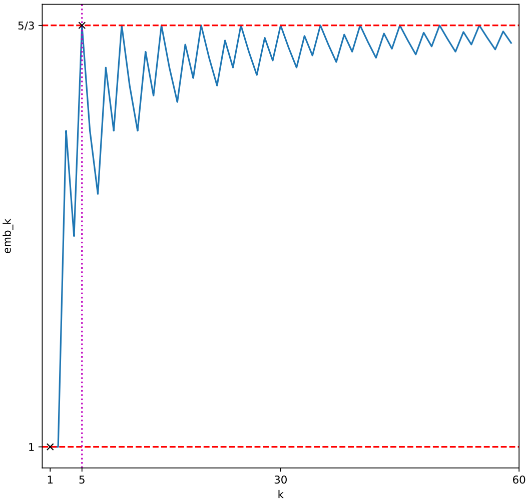


Figure: $\text{emb}_k(\mathcal{H})$ for the 6-cycle

Decidability and MIP formulation, III

$$\min \quad w$$

Decidability and MIP formulation, III

$$\begin{aligned} \min \quad & w \\ \text{s.t.} \quad & \sum_{S \subseteq \mathcal{V}} x_S = k \end{aligned}$$

Decidability and MIP formulation, III

$$\begin{array}{llll}
\min & w & & \\
\text{s.t.} & \displaystyle\sum_{S \subseteq \mathcal{V}} x_S & = & k \\
& x_S & = & 0 \qquad \forall S \subseteq \mathcal{V} \\
& & & \quad \text{where } S \text{ is not connected}
\end{array}$$

Decidability and MIP formulation, III

$$
\begin{array}{llcll}
\min & w & & & \\
\text{s.t.} & \displaystyle\sum_{S \subseteq \mathcal{V}} x_S & = & k & \\
& x_S & = & 0 & \forall S \subseteq \mathcal{V} \\
& & & & \text{ where } S \text{ is not connected} \\
& \min\{x_S, x_T\} & = & 0 & \forall S, T \subseteq \mathcal{V} \\
& & & & \text{ where } S, T \text{ do not touch}
\end{array}
$$

Decidability and MIP formulation, III

$$
\begin{array}{llll}
\min & w & & \\
\text{s.t.} & \displaystyle\sum_{S \subseteq \mathcal{V}} x_S & = & k \\
& x_S & = & 0 \qquad \forall S \subseteq \mathcal{V} \\
& & & \quad \text{where } S \text{ is not connected} \\
& \min\{x_S, x_T\} & = & 0 \qquad \forall S, T \subseteq \mathcal{V} \\
& & & \quad \text{where } S, T \text{ do not touch} \\
& \displaystyle\sum_{S \subseteq V : e \cap S \neq \emptyset} x_S & \leq & w \qquad \forall e \in \mathcal{E}
\end{array}
$$

Decidability and MIP formulation, III

$$
\begin{array}{llcll}
\min & w & & & \\
\text{s.t.} & \displaystyle\sum_{S \subseteq \mathcal{V}} x_S & = & k & \\
& x_S & = & 0 & \forall S \subseteq \mathcal{V} \\
& & & & \text{where } S \text{ is not connected} \\
& \min\{x_S, x_T\} & = & 0 & \forall S, T \subseteq \mathcal{V} \\
& & & & \text{where } S, T \text{ do not touch} \\
& \displaystyle\sum_{S \subseteq V : e \cap S \neq \emptyset} x_S & \leq & w & \forall e \in \mathcal{E} \\
& x_S & \in & \mathbb{Z}_{\geq 0} & \forall S \subseteq V
\end{array}
$$

Decidability and MIP formulation, III

Decidability and MIP formulation, III

$$
\begin{aligned}
\min \quad & w \\
\text{s.t.} \quad & \sum_{S \subseteq \mathcal{V}} x_S = 1 \\
& x_S = 0 \quad \forall S \subseteq \mathcal{V} \text{ where } S \text{ is not connected} \\
& \min\{x_S, x_T\} = 0 \quad \forall S, T \subseteq \mathcal{V} \text{ where } S, T \text{ do not touch} \\
& \sum_{S \subseteq V : e \cap S \neq \emptyset} x_S \leq w \quad \forall e \in \mathcal{E} \\
& x_S \in \mathbb{R}_{\geq 0} \quad \forall S \subseteq V
\end{aligned}
$$

Decidability and MIP formulation, III

$$
\begin{array}{llll}
\min & w & & \\
\text{s.t.} & \displaystyle\sum_{S \subseteq \mathcal{V}} x_S & = & 1 \\
& x_S & = & 0 \quad \forall S \subseteq \mathcal{V} \text{ where } S \text{ is not connected} \\
& \min\{x_S, x_T\} & = & 0 \quad \forall S, T \subseteq \mathcal{V} \text{ where } S, T \text{ do not touch} \\
& \displaystyle\sum_{S \subseteq V : e \cap S \neq \emptyset} x_S & \leq & w \quad \forall e \in \mathcal{E} \\
& x_S & \in & \mathbb{R}_{\geq 0} \quad \forall S \subseteq V
\end{array}
$$

Theorem

Let w^* be the optimal solution of MIP. Then, $\mathrm{emb}(\mathcal{H}) = 1/w^*$*. Additionally, there exists an integer* $K \geq 3$ *such that* $\mathrm{emb}(\mathcal{H}) = \mathrm{emb}_K(\mathcal{H})$*.*

Conditional Lower Bound, I

Conjectures related to k-Clique

Conditional Lower Bound, I

Conjectures related to k-Clique

Conjecture (Combinatorial k-Clique; Lincoln, Vassilevska-Williams & Williams, 17')

Any combinatorial algorithm to detect a k-clique in a graph with n nodes requires $n^{k-o(1)}$ time on a Word RAM model.

Conditional Lower Bound, I

Conjectures related to k-Clique

Conjecture (Combinatorial k-Clique; Lincoln, Vassilevska-Williams & Williams, 17')

Any combinatorial algorithm to detect a k-clique in a graph with n nodes requires $n^{k-o(1)}$ time on a Word RAM model.

Conjecture (Min Weight k-Clique; Lincoln, Vassilevska-Williams & Williams, 17')

Any randomized algorithm to find a k-clique of minimum total edge weight requires $n^{k-o(1)}$ time on a Word RAM model.

Conditional Lower Bound, II

Conditional Lower Bound, II

Theorem

For any $\mathcal{H}$, $\mathrm{CSP}(\mathcal{H})$ cannot be computed via a combinatorial algorithm in time $O(|I|^{\mathrm{emb}(\mathcal{H})-\epsilon})$ unless the Combinatorial k-Clique Conjecture is false.

Conditional Lower Bound, II

Theorem

For any $\mathcal{H}$, $\mathrm{CSP}(\mathcal{H})$ cannot be computed via a combinatorial algorithm in time $O(|I|^{\mathrm{emb}(\mathcal{H})-\epsilon})$ unless the Combinatorial k-Clique Conjecture is false.

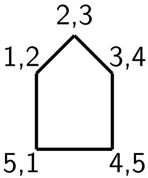
Proof.

Conditional Lower Bound, II

Theorem

For any $\mathcal{H}$, $\mathrm{CSP}(\mathcal{H})$ *cannot be computed via a combinatorial algorithm in time* $O(|I|^{\mathrm{emb}(\mathcal{H})-\epsilon})$ *unless the Combinatorial k-Clique Conjecture is false.*

Proof.

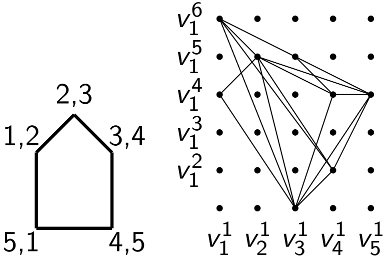


Conditional Lower Bound, II

Theorem

For any $\mathcal{H}$, CSP($\mathcal{H}$) cannot be computed via a combinatorial algorithm in time $O(|I|^{\mathrm{emb}(\mathcal{H})-\epsilon})$ unless the Combinatorial k-Clique Conjecture is false.

Proof.



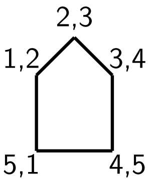
□

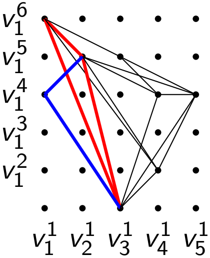
Conditional Lower Bound, II

Theorem

For any $\mathcal{H}$, CSP($\mathcal{H}$) cannot be computed via a combinatorial algorithm in time $O(|I|^{\text{emb}(\mathcal{H})-\epsilon})$ unless the Combinatorial k-Clique Conjecture is false.

Proof.





x_1	x_2
$\langle v_1^6, v_2^5 \rangle$	$\langle v_2^5, v_3^1 \rangle$
$\langle v_1^4, v_2^5 \rangle$	$\langle v_2^5, v_3^1 \rangle$

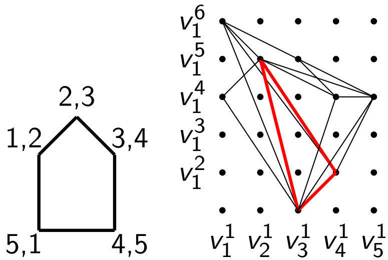
□

Conditional Lower Bound, II

Theorem

For any $\mathcal{H}$, CSP($\mathcal{H}$) cannot be computed via a combinatorial algorithm in time $O(|I|^{\text{emb}(\mathcal{H})-\epsilon})$ unless the Combinatorial k-Clique Conjecture is false.

Proof.



x_1	x_2
$\langle v_1^6, v_2^5\rangle$	$\langle v_2^5, v_3^1\rangle$
$\langle v_1^4, v_2^5\rangle$	$\langle v_2^5, v_3^1\rangle$

x_2	x_3
$\langle v_2^5, v_3^1\rangle$	$\langle v_3^1, v_4^2\rangle$

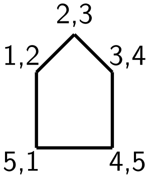
□

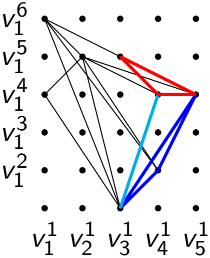
Conditional Lower Bound, II

Theorem

For any $\mathcal{H}$, CSP($\mathcal{H}$) cannot be computed via a combinatorial algorithm in time $O(|I|^{\text{emb}(\mathcal{H})-\epsilon})$ unless the Combinatorial k-Clique Conjecture is false.

Proof.





x_1	x_2
$\langle v_1^6, v_2^5\rangle$	$\langle v_2^5, v_3^1\rangle$
$\langle v_1^4, v_2^5\rangle$	$\langle v_2^5, v_3^1\rangle$

x_2	x_3
$\langle v_2^5, v_3^1\rangle$	$\langle v_3^1, v_4^2\rangle$

x_3	x_4
$\langle v_3^5, v_4^4\rangle$	$\langle v_4^4, v_5^4\rangle$
$\langle v_3^1, v_4^2\rangle$	$\langle v_4^2, v_5^4\rangle$
$\langle v_3^1, v_4^4\rangle$	$\langle v_4^4, v_5^4\rangle$

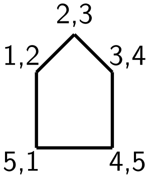
□

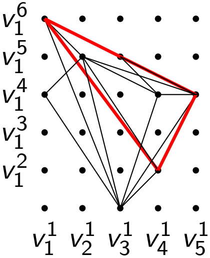
Conditional Lower Bound, II

Theorem

For any $\mathcal{H}$, CSP($\mathcal{H}$) cannot be computed via a combinatorial algorithm in time $O(|I|^{\mathrm{emb}(\mathcal{H})-\epsilon})$ unless the Combinatorial k-Clique Conjecture is false.

Proof.





x_1	x_2
$\langle v_1^6, v_2^5 \rangle$	$\langle v_2^5, v_3^1 \rangle$
$\langle v_1^4, v_2^5 \rangle$	$\langle v_2^5, v_3^1 \rangle$

x_2	x_3
$\langle v_2^5, v_3^1 \rangle$	$\langle v_3^1, v_4^2 \rangle$

x_3	x_4
$\langle v_3^5, v_4^4 \rangle$	$\langle v_4^4, v_5^4 \rangle$
$\langle v_3^1, v_4^2 \rangle$	$\langle v_4^2, v_5^4 \rangle$
$\langle v_3^1, v_4^4 \rangle$	$\langle v_4^4, v_5^4 \rangle$

x_4	x_5
$\langle v_4^2, v_5^4 \rangle$	$\langle v_5^4, v_1^6 \rangle$

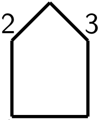
□

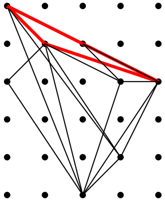
Conditional Lower Bound, II

Theorem

For any $\mathcal{H}$, CSP($\mathcal{H}$) cannot be computed via a combinatorial algorithm in time $O(|I|^{\text{emb}(\mathcal{H})-\epsilon})$ unless the Combinatorial k-Clique Conjecture is false.

Proof.





x_1	x_2
$\langle v_1^6, v_2^5 \rangle$	$\langle v_2^5, v_3^1 \rangle$
$\langle v_1^4, v_2^5 \rangle$	$\langle v_2^5, v_3^1 \rangle$

x_2	x_3
$\langle v_2^5, v_3^1 \rangle$	$\langle v_3^1, v_4^2 \rangle$

x_3	x_4
$\langle v_3^5, v_4^4 \rangle$	$\langle v_4^4, v_5^4 \rangle$
$\langle v_3^1, v_4^2 \rangle$	$\langle v_4^2, v_5^4 \rangle$
$\langle v_3^1, v_4^4 \rangle$	$\langle v_4^4, v_5^4 \rangle$

x_4	x_5
$\langle v_4^2, v_5^4 \rangle$	$\langle v_5^4, v_1^6 \rangle$

x_5	x_1
$\langle v_5^4, v_1^6 \rangle$	$\langle v_1^6, v_2^5 \rangle$

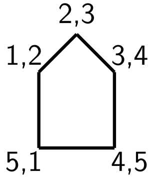
□

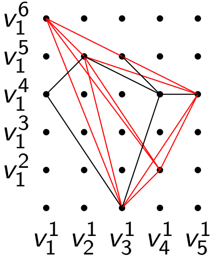
Conditional Lower Bound, II

Theorem

For any $\mathcal{H}$, CSP($\mathcal{H}$) cannot be computed via a combinatorial algorithm in time $O(|I|^{\mathrm{emb}(\mathcal{H})-\epsilon})$ unless the Combinatorial k-Clique Conjecture is false.

Proof.





x_1	x_2
$\langle v_1^6, v_2^5 \rangle$	$\langle v_2^5, v_3^1 \rangle$
$\langle v_1^4, v_2^5 \rangle$	$\langle v_2^5, v_3^1 \rangle$

x_2	x_3
$\langle v_2^5, v_3^1 \rangle$	$\langle v_3^1, v_4^2 \rangle$

x_3	x_4
$\langle v_3^5, v_4^4 \rangle$	$\langle v_4^4, v_5^4 \rangle$
$\langle v_3^1, v_4^2 \rangle$	$\langle v_4^2, v_5^4 \rangle$
$\langle v_3^1, v_4^4 \rangle$	$\langle v_4^4, v_5^4 \rangle$

x_4	x_5
$\langle v_4^2, v_5^4 \rangle$	$\langle v_5^4, v_1^6 \rangle$

x_5	x_1
$\langle v_5^4, v_1^6 \rangle$	$\langle v_1^6, v_2^5 \rangle$

□

Conditional Lower Bound, III

Conditional Lower Bound, III

The proof can be adapted to tropical semiring (min k-clique) by assigning each pair $\{u, v\} \subseteq [k]$ to a unique hyperedge according to ψ.

Conditional Lower Bound, III

The proof can be adapted to tropical semiring (min k-clique) by assigning each pair $\{u, v\} \subseteq [k]$ to a unique hyperedge according to ψ.

Theorem

For any $\mathcal{H}$, $\mathrm{CSP}(\mathcal{H})$ over tropical semiring cannot be computed via any randomized algorithm in time $O(|I|^{\mathrm{emb}(\mathcal{H})-\epsilon})$ unless the Min Weight k-Clique Conjecture is false.

Tightness & Gap

	emb	subw
Acyclic	1	1
Chordal	=	=
ℓ-cycle	$2 - 1/\lceil \ell/2 \rceil$	$2 - 1/\lceil \ell/2 \rceil$
$K_{2,\ell}$	$2 - 1/\ell$	$2 - 1/\ell$
$K_{3,3}$	2	2
A_ℓ	$(\ell - 1)/2$	$(\ell - 1)/2$
$\mathcal{H}_{\ell,k}$	ℓ/k	ℓ/k
Q_b	$7/4$	2

Table: Clique embedding power and submodular width for query classes

Future Work

Future Work

- Understand the gap in the boat query...

Future Work

- Understand the gap in the boat query...
- Fast matrix multiplication ([NP85] solves k-clique in $O(n^{\frac{k}{3}\cdot\omega}))$...

Future Work

- Understand the gap in the boat query...
- Fast matrix multiplication ([NP85] solves k-clique in $O(n^{\frac{k}{3}\cdot\omega})$)...
- ...

Thank You!

References I

Albert Atserias, Martin Grohe, and Dániel Marx.
Size bounds and query plans for relational joins.
SIAM J. Comput., 42(4):1737–1767, 2013.

Todd J. Green, Gregory Karvounarakis, and Val Tannen.
Provenance semirings.
In *PODS*, pages 31–40. ACM, 2007.

Georg Gottlob, Stephanie Tien Lee, Gregory Valiant, and Paul Valiant.
Size and treewidth bounds for conjunctive queries.
J. ACM, 59(3):16:1–16:35, 2012.

Martin Grohe and Dániel Marx.
Constraint solving via fractional edge covers.
In *SODA*, pages 289–298. ACM Press, 2006.

References II

Martin Grohe and Dániel Marx.
Constraint solving via fractional edge covers.
ACM Trans. Algorithms, 11(1):4:1–4:20, 2014.

Martin Grohe.
The complexity of homomorphism and constraint satisfaction problems seen from the other side.
J. ACM, 54(1):1:1–1:24, 2007.

Mahmoud Abo Khamis, Hung Q. Ngo, and Dan Suciu.
Computing join queries with functional dependencies.
In *PODS*, pages 327–342. ACM, 2016.

Mahmoud Abo Khamis, Hung Q. Ngo, and Dan Suciu.
What do shannon-type inequalities, submodular width, and disjunctive datalog have to do with one another?
In *PODS*, pages 429–444. ACM, 2017.

References III

Dániel Marx.
Can you beat treewidth?
Theory Comput., 6(1):85–112, 2010.

Dániel Marx.
Tractable hypergraph properties for constraint satisfaction and conjunctive queries.
J. ACM, 60(6):42:1–42:51, 2013.

Frantisek Matús.
Infinitely many information inequalities.
In *ISIT*, pages 41–44. IEEE, 2007.

Jaroslav Nešetřil and Svatopluk Poljak.
On the complexity of the subgraph problem.
Commentationes Mathematicae Universitatis Carolinae, 26(2):415–419, 1985.

References IV

Hung Q. Ngo, Ely Porat, Christopher Ré, and Atri Rudra.
Worst-case optimal join algorithms: [extended abstract].
In *PODS*, pages 37–48. ACM, 2012.

Hung Q. Ngo, Christopher Ré, and Atri Rudra.
Skew strikes back: new developments in the theory of join algorithms.
SIGMOD Rec., 42(4):5–16, 2013.

Todd L. Veldhuizen.
Triejoin: A simple, worst-case optimal join algorithm.
In *ICDT*, pages 96–106. OpenProceedings.org, 2014.

Mihalis Yannakakis.
Algorithms for acyclic database schemes.
In *VLDB*, pages 82–94. IEEE Computer Society, 1981.