Aspects of Dynamic Complexity

Thomas Schwentick

Highlights 2014
Paris, September 2014
Joint work with...

Thomas Zeume

(TZ) ≡ Slide borrowed from Thomas Zeume
The general setting (1/2)

Input

Reach(s,t)?

yes

no

yes

Aux. data
The general setting (2/2)

- In this talk, we only consider graph queries
  - ...graphs may have some distinguished vertices
- Graphs can be **changed** by insertions and deletions of edges — one at a time
- The auxiliary data usually consists of relations
- The auxiliary relations are **updated** after each change — by a logical formula
- There is a particular aux relation $Q$ (the “query relation”) yielding the current query result
- We will often consider the (Boolean) reachability query

**Definition: Reachability**

**Input:** Directed Graph $G$, vertices $s, t$

**Question:** Is there a path from $s$ to $t$ in $G$?
• Often when Thomas Zeume or I talk about dynamic complexity abroad, some train connections fail on our journey.

• Complexity workshop 2013 (Berlin → Hannover):

• WoLLIC 2013 (Dortmund → Darmstadt):

Train delays at the central station in Mainz have caused commuter chaos. The problem was caused by staff shortages at Deutsche Bahn, with too many rail dispatchers being sick or on holiday.

• Logic workshop 2013 (Dortmund → Bremen):
Dynamic complexity and travelling (2/2)

- Etienne Grandjean 60 Workshop 2013  
  (Dortmund → Caen):

- Essen → Paris:

  Deutsche Bahn to stop selling Thalys tickets
  Author - Alex McWhirter - 17 May 2013

  Deutsche Bahn has announced it will no longer sell tickets for the Thalys high-speed train service from June 5.

  The decision will mainly affect those passengers using the high-speed line linking Brussels with Cologne which is served by both Thalys and DB’s ICE trains.

  Passengers will lose out because they cannot book the full range of services and - if they need to use both companies’ services - they will have to purchase separate tickets with all the complications that will entail.

  The Brussels-Cologne route was recently upgraded to high-speed operation. It’s popular not just with travellers within mainland Europe but also with those originating in the UK and who are bound for Germany and beyond because Cologne is a major rail interchange.

- Dortmund → Essen:

- Detour:
Examples (1/2)

Example: Reachability under Insertion

- **Intention:** Binary auxiliary relation $T$ stores the transitive closure of the edge relation.

- Update formula $\phi_T^{INS}(a, b; x, y)$
  - should become true if $T$ contains $(x, y)$ after inserting $(a, b)$ to $E$,

- $\phi_T^{INS}(a, b; x, y) \overset{\text{def}}{=} T(x, y) \lor (T(x, a) \land T(b, y))$

- $T$ does not suffice if edges can also be deleted

[Dong, Libkin, Wong 95]
Examples (2/2)

Example: Reachability in acyclic graphs under Insertion

- On acyclic graphs, Reachability can be maintained with first-order updates [Dong, Su 93/95; Patnaik, Immerman 94/97]

- Challenge: how to know, whether the deleted edge \((a, b)\) was crucial or there is still a path \(p\) from \(x\) to \(y\) after deletion of an edge \((a, b)\)?

- There are two cases:
  1. \((a\) not reachable from \(x\)) or \((y\) not reachable from \(b\)); or
  2. otherwise \(p\) must have a last node \(u\) from which \(a\) can be reached

- Update formula \(\phi_T^{\text{DEL}}(a, b; x, y)\):
  \[
  (T(x, y) \land (\neg T(x, a) \lor \neg T(b, y))) \lor \\
  \exists u, v ((u \equiv a \lor v \equiv y) \land \\
  T(x, u) \land T(u, a) \land E(u, v) \land \neg T(v, a) \land T(v, y))
  \]

⚠️ Does not work for cyclic graphs
Motivation & related approaches

• Why (first-order) logic as update language?
  – Database theory view:
    ∗ SQL $\equiv$ FO
    · (ignoring counting, grouping, aggregation, arithmetics)
    · Which recursive queries can be maintained without recursion?
  – Complexity theoretic view:
    ∗ uniform $\text{AC}^0 \equiv \text{FO}(+, \times)$
    [Barrington, Immerman, Straubing 90]
    · $\text{AC}^0$ is the “weakest complexity class”
  – Logical view:
    ∗ Gain more insight into
    · the “dynamics” of logics
    · the power of FO on finite structures

• A close relative: First-order incremental evaluation systems (FOIES)
  – FOIES can add elements
  – set updates
  – queries

• History:
  – FOIES: Dong, Su 1993
  – DynFO: Patnaik, Immerman 1994

• A remote relative: dynamic algorithms
Research goals and questions

• Main Goal: Understand Dynamic Complexity
  – Which queries can be maintained with FO logic?
    ∗ e.g.: Can Reachability on general graphs be maintained with FO logic?
  – Which queries can not be maintained with FO logic?
    ∗ How to prove that some query can not be maintained with FO logic?
  – More generally:
    ∗ Which logics \( \mathcal{L}_1 \) can (or can not) maintain which problems?
    ∗ Which logics \( \mathcal{L}_1 \) capture which stronger logics \( \mathcal{L}_2 \) dynamically?
Short History of Dynamic Complexity

1990
- Introduction of the setting

1995
- Maintaining basic graph queries

2000
- Reachability in dynamic $\text{TC}^0$
- Lower bounds for restricted Arity

2005
- Connections to static complexity classes
- Lower bounds for small syntactic fragments

2010
- Reachability in dynamic non-uniform $\text{AC}^0[\oplus]$

2014
- Thomas Schwentick

Aspects of Dynamic Complexity
Contents

Introduction

Settings
  Positive results
  Lower bounds
  Conclusion
Towards an exact setting of dynamic complexity, there are many options to choose from:

- Is the universe fixed or can it change over time?
- What is the initial situation concerning the input graph and the auxiliary relations? * To be discussed soon
- What kinds of change operations do we allow?
  - Insertions and deletions of single tuples
  - Insertions and deletions of sets of tuples
  - Updates defined by formulas/queries with parameters
- What logics do we allow for update formulas?
  - Many choices, to be discussed throughout the talk
- What kind of auxiliary data do we allow?
  - Relations
  - Relations and functions
  - Other
- What kind of queries do we support?
  - Boolean
  - Arbitrary arity
- What is the semantics of update formulas?
  - They define the aux data explicitly
  - They only define the changes (aka $\delta$-semantics)
Initialization - Setting 1: empty input, empty aux aux

- Setting 1 starts from empty input and empty auxiliary data
  [Patnaik, Immerman 94/97]
- This setting is quite generous:
  - E.g.: a linear order can be defined incrementally
  - Arithmetic over the “activated elements” can be defined incrementally [Etessami 98]
- \text{DynFO} \overset{\text{def}}{=} \text{class of queries that can be maintained with first-order logic in Setting 1}
- \text{DynFO} contains, e.g.
  - Reachability on acyclic graphs
  - Reachability on undirected graphs
  [Patnaik, Immerman 94/97]
  - Reachability on embedded planar graphs
  [Datta, Hesse, Kulkarni 14]
  - Bipartiteness
  [Patnaik, Immerman 94/97]
  - Tree isomorphism [Etessami 98]
Initialization - Setting 2: empty input, precomputed aux

- Setting 2: process starts from an empty input but with precomputed auxiliary relations
  - (depending on the size of the universe)
- It relates to circuit complexity classes
- Arbitrary aux data:
  - Non-uniform $\text{AC}^0$
  - Arithmetic ($+, \times$) as aux data:
    - Uniform $\text{AC}^0$
- Various other subsettings depending on the power of the precomputation
Initialization - Setting 3: non-empty input, precomputed aux

- Setting 3 starts from non-empty input
  → precomputed aux data needed (depending on the actual input)
- Setting 3 emphasizes “maintain-ability”
- With \textbf{PTIME} precomputation, Settings 2 and 3 coincide
- With less precomputation power, Setting 3 can be
  - In particular, it might be impossible to build a linear order within the aux data
- E.g., with “logical initialization”, first-order logic
  - \textbf{can not maintain} Tree Isomorphism, but
  - can (still) maintain Reachability on undirected graphs and Bipartiteness

[Grädel, Siebertz 12]
## Overview of settings

<table>
<thead>
<tr>
<th>Aux-Initialization</th>
<th>Empty initial structure</th>
<th>Arbitrary initial structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arbitrary</td>
<td></td>
<td>Non-uniform Dyn $\text{AC}^0$</td>
</tr>
<tr>
<td>$+, \times$</td>
<td>Uniform Dyn $\text{AC}^0$</td>
<td></td>
</tr>
<tr>
<td>Empty</td>
<td>$\text{DynFO}$</td>
<td>[Patnaik/Immerman 94/97]</td>
</tr>
<tr>
<td>PTime</td>
<td>$\text{DynFO}^+$</td>
<td>[Patnaik/Immerman 94/97]</td>
</tr>
<tr>
<td>Logical</td>
<td></td>
<td>[Grädel/Siebertz 12]</td>
</tr>
</tbody>
</table>
## Definitions

### Dynamic schema:

\((\tau_{in}, \tau_{aux})\)

### Change:

\(\delta(\vec{a}), \delta \in \{\text{INS}_S, \text{DEL}_S\} (S \in \tau_{in}), \vec{a}\) from \(D\)

### Update program:

Update formulas \(\phi_\delta^R(\vec{u}; \vec{x})\), for every \(\delta \in \{\text{INS}_S, \text{DEL}_S\}, R \in \tau_{aux}\)

### Program state:

\((D, A)\) (input graph/database, auxiliary relations)

### Dynamic program:

\(P = (P, \text{INIT}, Q)\) (update program, initialization function, query relation)

### \(P\) maintains \(q\):

for every change sequence \(\alpha = \delta_1 \ldots \delta_m\) and every structure \(D\) it holds:

\[q(\alpha(D)) = \text{relation } Q \text{ in state } P_\alpha(\text{INIT}(D))\]

### DynC:

Queries that can be maintained by dynamic programs with formulas from \(C\)

### For the time being:

\(\tau_{in} = \{E\}\)

---

### Theorem

[Dong, Su 93/95, Patnaik, Immerman 94/97]

- ACYCLIC REACH \(\in\) DynFO
Contents

Introduction
Settings
Positive results
  Lower bounds
  Conclusion

Thomas Schwentick
Aspects of Dynamic Complexity
Undirected Reachability in DynFO (1/3)

- We already know: ACYCLIC \textsc{reach} \in \textbf{DynFO}
- As another restriction of \textsc{reach}, we now consider \textsc{sym-reach}:
  - Reachability for undirected graphs
- There are several proofs for \textsc{sym-reach} \in \textbf{DynFO}
  - We start with the simplest and first proof by
    [Patnaik, Immerman 94/97]
Undirected Reachability in DynFO (2/3)

Example: Insertion

Basic idea: maintain a spanning forest $F$ and its transitive closure $T$.

On arrival of a new edge, add it to $F$, if it connects two distinct components.

Example: Deletion

Deletion is, again, more tricky.

How to modify the spanning tree if an edge $(a, b)$ is deleted but its component remains connected?

- Determine nodes $u$ and $v$ in the subtrees of $a$ and $b$, respectively, such that $(u, v) \in E$, and add $(u, v)$ to $F$.

This can be done with

- a more sophisticated relation $T$ with all triples $(d, e, g)$ for which there is a path in $F$ from $d$ to $e$ through $g$.
- some order on the edges to choose $(u, v)$ uniquely.
Theorem [Patnaik, Immerman 94/97]

- **SYM-REACH ∈ DynFO**
- Is the ternary auxiliary relation $T$ necessary?
- $k$-ary DynFO: queries in DynFO that can be maintained with (at most) $k$-ary aux relations

Theorem [Dong, Su 95/98]

- **SYM-REACH ∈ binary DynFO**
- **SYM-REACH ∉ unary DynFO**

- The construction of [Patnaik, Immerman 94/97] uses a linear order on the vertices in a seemingly crucial way.
- However, [Grädel, Siebertz 12] show that such a linear order is not needed:
  - **SYM-REACH** can still be maintained in Setting 3 (with initialization of aux relations by fixed-point formulas)

- Thus: two important restrictions of **REACH** can be maintained in **DynFO**
- What about **REACH**?
- We see next that **REACH** can be maintained in some extension of **DynFO**
  - ... and some more recent results...
An extension of DynFO capturing Reach

By **FOC** we denote the extension of **FO** with number variables and counting quantifiers:

**Theorem** [Hesse 03]

**Reach ∈ DynFOC**

### Proof idea

- Let \( p_{u,v}(k) \) be the number of paths of length \( k \) from node \( u \) to node \( v \) (in a graph \( G' \)).

- Maintain generating functions:
  \[
  f_{u,v}(x) \overset{\text{def}}{=} \sum_{k=0}^{\infty} p_{u,v}(k) x^k
  \]

- Then for every \( x > 0 \):
  \[
  f_{s,t}(x) > 0 \iff \text{there is a path from } s \text{ to } t
  \]

- Updates can be expressed “easily” wrt generating functions, e.g., insertion of edge \((a, b)\) yields:
  \[
  f'_{u,v}(x) = f_{u,v}(x) +
  f_{u,a}(x) \cdot x \cdot f_{b,v}(x) \cdot \left( \sum_{k=0}^{\infty} f_{b,a}(x) x^k \right)
  \]

- How to deal with \( \infty \)? Truncate!

- Consider \[
  \sum_{k=0}^{n-1} p_{u,v}(k) (2^{n^2})^k
  \]
  and do computations modulo \( 2^{n^3} \)

\(\implies\) Functions from pairs of nodes to \( n^3 \)-digit numbers can be encoded by 5-ary relations
Theorem [Datta, Hesse, Kulkarni 14]

- Reachability in dynamic non-uniform $\text{AC}^0[\oplus]$.

- $\text{AC}^0[\oplus] \overset{\text{def}}{=} \text{families of constant-depth circuits with unbounded AND-, OR- and MOD 2 gates}$.

- Proof ingredients:
  - For each graph there is a non-uniform weighting scheme for its edges (with polynomially bounded weights) such that
    - any two vertices are connected by a unique minimal path
  - Non-uniformity
  - Maintain Hesse’s polynomials modulo 2
  - Actually, more involved polynomials are used to get the desired properties in the MOD 2 context

More recent developments:

- A submitted paper [Datta, Kulkarni 14] shows:
  - $\text{REACH} \in \text{Uniform AC}^0$ ✸ not yet published

- [Schwentick, Zeume 14] claim that for every domain independent query $Q$ it holds:
  - $Q \in \text{Uniform AC}^0 \iff Q \in \text{DynFO}$ ✸ not even submitted

- As $\text{REACH}$ is domain independent, these two statements (if correct!) yield:
  - $\text{REACH} \in \text{DynFO}$
Hesse initiated the study of dynamic programs with quantifier-free update formulas [Hesse 03]

**Definition**

- **DynProp:**
  - Queries that can be maintained in DynFO with quantifier-free formulas and aux \textbf{relations}

- **DynQF:**
  - Queries that can be maintained in DynFO with quantifier-free formulas and aux \textbf{functions} (and relations)

	extbf{DynQF} formulas can use “if-then-else”-terms

- Quantifier-free update formulas? Isn’t that extremely weak?
Dynamic programs with quantifier-free formulas (2/3)

**Theorem [Hesse 03]**

- **DET-Reach ∈ DynProp**
  (no quantifiers, aux relations)

**Proof idea**

- We first consider **acyclic** deterministic graphs
  - We already saw the update formula
    \[ \phi_T^{\text{INS}}(a, b; x, y) \] for the transitive closure
    under edge insertion:
    \[ T(x, y) \lor (T(x, a) \land T(b, y)) \]
  - For deterministic acyclic graphs the formula
    \[ \phi_T^{\text{DEL}}(a, b; x, y) \] for deletion is almost as simple:
    \[ T(x, y) \land \neg(T(x, a) \land E(a, b) \land T(b, y)) \]
- The construction for general deterministic graphs is more involved, based on “cut edges” of cycles

from [Hesse 03]
Dynamic programs with quantifier-free formulas (3/3)

Theorem [Hesse 03]

- **SYM-REACH** ∈ unary **DynQF**
  (no quantifiers, unary aux functions & relations)

- The involved construction maintains a Euler tour of a spanning tree of each connected component

from [Hesse 03]
Contents

Introduction
Settings
Positive results

Lower bounds
Conclusion
Lower bounds: a sad state

- Easy observation: \( q \in \text{DynFO} \Rightarrow q \in \text{PTIME} \)
  - Just insert the tuples of \( D \) into an empty database one by one, and compute all updates

- So far there are no other general lower bound results for \( \text{DynFO} \)

- Most existing lower bounds apply to
  - restricted logics or
  - aux relations of bounded arity
  - or both...
Reachability is not in unary DynFO (1/2)

Theorem [Dong Su 95/98]

- \( \text{Reach} \notin \text{unary DynFO} \)

Proof sketch

- Proof sketch for binary Reachability query
- Proof by contradiction with a locality argument
- Assume there is a unary dynamic program for \( \text{Reach} \) with \( m \) unary aux relations and a query formula \( \phi_Q^{\text{DEL}}(a, b; x, y) \) of quantifier-depth \( k \)
  - The aux relations induce, for each node, one of \( 2^m \) colors
- Consider a sufficiently long path
  with \( \geq 4(2 \cdot 4^k + 2)2^m(2 \cdot 4^k + 2) \) nodes
Reachability is not in unary DynFO (2/2)

Example

\[
\begin{align*}
&\quad p_1 & p_2 & p_3 & p_4 \\
& u_1v_1 & u_2v_2 & u_3v_3 & u_4v_4
\end{align*}
\]

Proof sketch (cont.)

- As the path is long enough, there must exist four disjoint subpaths of length \(2 \cdot 4^k + 2\) each with identical color (relations) sequence
  - Let \((u_1, v_1), \ldots, (u_4, v_4)\) be the innermost edges of these paths
  - After deletion of \((u_3, v_3)\),
    * \(u_2\) is still reachable from \(v_1\), but
    * \(u_4\) is no longer reachable from \(v_1\)
- The \(4^k\)-neighborhoods of \((v_1, u_3, v_3, u_2)\) and \((v_1, u_3, v_3, u_4)\) are isomorphic
  - \(\phi_{\text{DEL}}^Q(u_3, v_3; v_1, u_2) \equiv \phi_{\text{DEL}}^Q(u_3, v_3; v_1, u_4)\) by Gaifman’s Theorem
  - The program gives the same query answer to \((v_1, u_2)\) or \((v_1, u_4)\), after deletion of \((u_3, v_3)\)
  - The program is wrong with respect to either \((v_1, u_2)\) or \((v_1, u_4)\), the desired contradiction

\(\checkmark\) The proof already works for undirected graphs and deterministic graphs

Thomas Schwentick
Aspects of Dynamic Complexity
Towards more lower bounds

- To prove significant lower bounds for dynamic complexity we need to develop “dynamic methods”

- **Our approach:**
  - Start with small fragments of **DynFO**
  - **Starting point:** **DynProp** (no quantifiers, aux relations)

- **Two guiding questions:**
  - Are there generic proof methods for such small fragments?
  - Can lower bounds for fragments larger than **DynProp** be obtained?

- **Our research agenda:**
  - **Task 1:** Relationships between small fragments.
    - [Zeume, Schwentick 14]
  - **Task 2:** Lower bounds for **DynProp**
    - [Gelade, Marquardt, Schwentick 08/12]
      - [Zeume, Schwentick 13; Zeume 14]
  - **Task 3:** Lower bounds for larger fragments
    - [Zeume, Schwentick 13; Zeume 14]
  - **Task 4:** Extract generic lower proof methods
    - [Partially, not satisfactory]
How do small fragments of DynFO relate?

- Small fragments in the static world

  \[
  \text{FO} \\
  \text{UCQ} = \exists^* \text{FO} \quad \forall^* \text{FO} \\
  \text{Prop} \\
  \text{CQ}^\neg \\
  \text{PropUCQ} \\
  \text{CQ} \\
  \text{PropCQ}^\neg \\
  \text{PropCQ}
  \]

- Small fragments in the dynamic world

  \[
  \text{DynFO} \\
  \text{Dyn}\exists^* \text{FO} = \text{Dyn}\forall^* \text{FO} \\
  \text{DynCQ}^\neg = \text{DynUCQ}^\neg \\
  \text{DynCQ} = \text{DynUCQ} \\
  \text{DynQF} \\
  \text{DynProp} = \text{DynPropUCQ}^\neg \\
  \text{DynPropCQ}^\neg = \text{DynPropUCQ} \\
  \text{DynPropCQ}
  \]

  (non-empty input, PTIME aux) [Zeume, Schwentick 14]

- Many static classes coincide in the dynamic world
- Linear hierarchy of classes!
- Further: \( \text{FO} \subseteq \text{DynCQ}^\neg \)
Alternating Reachability is not in DynProp

Theorem [Gelade, Marquardt, Schwentick 08/12]
- Alternating Reachability \( \not\in \text{DynProp} \)

Proof idea

\[ \exists \] \[ \exists \] \[ \forall \] \[ \exists \] \[ \exists \]
\[ t \] \[ A \] \[ B \] \[ r' \] \[ r \] \[ s \]

- **A**: \( 2m \) existential nodes 
  \( v_1, \ldots, v_{2m} \)

- **B**: one universal node per size-\( m \)-subset of **A**

- **C**: one existential node per subset of **B**

Proof idea (cont.)

- Assume: \( P \) is a DynProp program for Alternating Reachability \( \exists \) (and let \( m \) be large enough)

  - There are \( \geq 2^{2m} \) nodes in **C**
  - There are \( < 2^{2m} \) types of tuples \((s, t, v_1, \ldots, v_{2m}, r)\) if \( m \) is sufficiently large with respect to \( P \)

  - There are \( r \equiv r' \) in **C** with the same tuple type (together with \( s, t, v_1, \ldots, v_{2m} \))

  - There is a set \( I \subseteq A \) such that insertion of all edges \((u, t), u \in I\), makes \( t \) (alternatingly) reachable from exactly one of \( r \) and \( r' \)

  - However, after adding either \((s, r)\) or \((s, r')\)
    the tuples \((s, t, v_1, \ldots, v_{2m}, r)\) and \((s, t, v_1, \ldots, v_{2m}, r')\) still have the same type

  - Contradiction

Theorem [Gelade, Marquardt, Schwentick 08/12]
- **FO \( \not\subseteq \text{DynProp} \)**
- On string structures: DynProp = Regular languages
A simple tool for DynProp lower bounds

- We used here a locality property of DynProp programs $\mathcal{P}$

- If
  - two induced substructures $S', T'$ of two structures $S, T$ are isomorphic (including the 0-ary query predicate), and
  - in both substructures the “same” sequence of changes is applied (modulo the isomorphism)
  then $\mathcal{P}$ gives the same answer on both resulting structures

Substructure Lemma (Variant of [Gelade et al. '09])

- For every DynProp-program $\mathcal{P}$:

\[
\begin{align*}
\alpha &= \delta(\vec{a}) \\
\beta &= \delta(\pi(\vec{a})) \\
\alpha(S) &\cong \beta(T)
\end{align*}
\]

- In particular:
  The query answer relations coincide in $\alpha(S)$ and $\beta(T)$
More lower bounds

- Even DynProp lower bounds do not come easily...
- For fixed, non-unary arity of the auxiliary data, matters can already become involved.
- We try to develop lower bound methods for higher arity relations.

**Theorem [Zeume, Schwentick 14]**

- \( \text{REACH} \notin \text{binary DynProp} \)

**Definition:** \( k\)-CLIQUE

<table>
<thead>
<tr>
<th>Input:</th>
<th>Undirected Graph ( G )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question:</td>
<td>Does ( G ) contain a ( k )-clique?</td>
</tr>
</tbody>
</table>

**Theorem [Zeume 14]**

- \( k\)-CLIQUE can be maintained in \((k-1)\)-ary DynProp, under edge insertions.
- \( k\)-CLIQUE \( \notin \) \((k-2)\)-ary DynProp.
\textbf{Theorem \cite{Zeume14}}

- \textbf{\(k\)-CLIQUE} \notin (\(k-2\))-ary DynProp

\section*{A very rough proof sketch}

- Assume that there is a (\(k-2\))-ary DynProp-program \(\mathcal{P}\) for \(k\)-CLIQUE.
- \textbf{Plan:} Force \(\mathcal{P}\) to fail on some graph & auxiliary data.
- We illustrate the case \(k = 4\): 4-CLIQUE is not in binary DynProp.
**Proof sketch (cont.)**

- Let $A$ be a large set ($[A]^3 \overset{\text{def}}{=} \text{set of } A\text{'s 3-element subsets}$)

**Ramsey lower bound:**
$[A]^3 = B \cup B'$ such that for each large $A' \subseteq A$ there are
- $b \subseteq A'$ with $b \in B$
- $b' \subseteq A'$ with $b' \in B'$

- Consider the graph with node set
  - $V \overset{\text{def}}{=} A \cup \{[b] \mid b \subseteq [A]^3\}$
  - for all $b \overset{\text{def}}{=} \{a_1, a_2, a_3\} \in B$

+ auxiliary relations

**Ramsey upper bound:** There is a large (auxiliary) clique $A' = \{a_1, \ldots, a_m\}$

→ Let $[b]$ and $[b']$ in $V$ such that $b, b' \subseteq A'$ and
  - $b \in B$
  - $b' \in B'$

**Example**

- Adding all internal edges for $b$ yields a 4-clique
- Adding all internal edges for $b'$ does not yield a 4-clique

**Contradiction!**

- Substructure Lemma

Thomas Schwentick

Aspects of Dynamic Complexity
Proof Ingredients

- We used:
  - Substructure Lemma
  - Discrepancy for Ramsey upper and lower bounds

Example

Theorem [Erdős-Hajnal-Rado]

For very large $A$:

- Every $c$-coloring of $[A]^k$ contains a large monochromatic clique.
- There is a 2-coloring of $[A]^{k+1}$ that contains no large clique.
Contents

- Introduction
- Settings
- Positive results
- Lower bounds

▶ Conclusion
Conclusion

- Dynamic complexity is an interesting field that is not yet well understood

- Many open problems:
  - Show that $\text{REACH} \not\in \text{DynFO}$
  - Show that $\text{REACH} \in \text{DynFO}$
  - Develop dynamic lower bound methods
  - Show that some (concrete) PTIME problem is not in $\text{DynFO}$
  - Show that some (concrete) PTIME problem is not in $\text{DynQF}$
  - Show that $\text{DynQF} \not\subseteq \text{DynFO}$
  - Show that $\text{REACH} \not\in \text{DynProp}$

- Which other “static logics” can be maintained dynamically by weaker logics?

Thank you!!!
Some references (1/2)


Some references (2/2)

- Thomas Zeume and Thomas Schwentick. Dynamic conjunctive queries. accepted for publication in *International Conference on Database Theory, ICDT 2014*