Path Querying on Graph Databases

Jelle Hellings
Hasselt University and transnational University of Limburg
Overview

Graph Databases

Motivation

Walk Logic

Relations with FO and MSO

Relations with CTL* and Hybrid CTL*

Conjunctive Regular Path Queries

Open Problems and Conclusion
Overview

Graph Databases

Motivation

Walk Logic

Relations with FO and MSO

Relations with CTL* and Hybrid CTL*

Conjunctive Regular Path Queries

Open Problems and Conclusion
Graphs

- Pieces of data (nodes)
- Relations between the pieces of data (edges)

Example (Social networks)

- **Jan** (worksAt **UHasselt**)
- **Jelle** (worksAt **UHasselt**)
- **Jelle** (worksWith **Jan**)
- **Jan** (worksAt **UHasselt**)

Diagram shows Jelle working at UHasselt, Jelle working with Jan, and Jan working at UHasselt.
Applications

- XML and RDF,
- Social networks,
- Transportation networks,
- The World Wide Web,
- ...
Graph Database: Google Maps

- Nodes: points of interest, addresses, ...
- Edges: road network
- Queries:

**Example (Distance based query)**

university close to <my address>
(answer: Universiteit Hasselt; 5.2 km)

**Example (Route-planning query)**

From: <my address>, to: <university>
(answer: options for university; followed by route)
Challenges

- Engineering: big data
  *Storage, distributed processing, hardware failures, ...*

- Conceptual: semantics and consistency
  *Structured data (facebook) versus structured? data (the web)*

- Conceptual: data querying
  *Local/navigational based versus graph-wide path based*
    - No widely used general purpose languages
    - Current practice: application specific languages
Challenges

- Engineering: big data
  *Storage, distributed processing, hardware failures, ...*

- Conceptual: semantics and consistency
  *Structured data (facebook) versus structured? data (the web)*

- Conceptual: data querying
  *Local/navigational based versus graph-wide path based*
    - No widely used general purpose languages
    - Current practice: application specific languages

Our focus

Path-based graph querying
Overview

Graph Databases

Motivation

Walk Logic

Relations with FO and MSO

Relations with CTL* and Hybrid CTL*

Conjunctive Regular Path Queries

Open Problems and Conclusion
Motivation

- Expressing graph-queries
- Properties of paths, walks, ...

Route planning

We want to travel from our office to a cafeteria and from this cafeteria get back to the office using a different route.
Graph querying

Graphs as traditional relations

\[
\text{worksAt}(\text{person}, \text{company})
\]

- Already deep knowledge of these systems
- First-order based query languages:

\[
Q(n) := \exists m \ \text{worksWith}(n, m) \land \text{worksAt}(m, \text{UHasselt})
\]

- Largely restricted to ‘local’ reasoning
  
  *no paths, no or only limited reachability, ...*
Higher-order logics

Monadic second-order logic

Extend first-order logic with quantification over sets

- Strong theoretical background
  - Sets with only nodes versus sets with nodes and edges
- Some graph problems are naturally expressible with sets:
  - Graph coloring, bipartite graph, ...

\[
\exists S \exists T (\forall x (x \in S \lor x \in T) \land \\
( x \in S \implies x \notin T) \land ( x \in T \implies x \notin S) \land \\
\forall y \text{ edge}(x, y) \implies ((x \in S \land y \in T) \lor (y \in S \land x \in T)))
\]

- Paths non-straightforward: *y is reachable from x*

\[
\forall S [(x \in S) \land \forall u \forall v (u \in S \land \text{edge}(u, v) \implies v \in S) \implies y \in S]
\]
Conjunctive Regular Path Queries

Idea

- Query nodes based on labelling of paths between nodes
- Express labelling by a *regular expression*

Example

\[ Q(a, b) := a\pi b, (\alpha\beta + \gamma\delta)^+(\pi) \]

\[
\begin{array}{ccccccc}
  n_1 & \xrightarrow{\alpha} & n_2 & \xrightarrow{\beta} & n_3 & \xrightarrow{\gamma} & n_4 \\
  \delta & \downarrow & \delta & \downarrow & \delta & \\
  n_5 & \xrightarrow{\gamma} & n_6 & \xrightarrow{\alpha} & n_7 & \xrightarrow{\beta} & n_8 \\
\end{array}
\]
Conjunctive Regular Path Queries

Idea

- Query nodes based on labelling of paths between nodes
- Express labelling by a *regular expression*

Example

\[
Q(a, b) := a\pi b, (\alpha\beta + \gamma\delta)^+(\pi)
\]

Diagram:

- Nodes: \(n_1, n_2, n_3, n_4, n_5, n_6, n_7, n_8\)
- Edges:
  - \(n_1 \xrightarrow{\alpha} n_2 \xrightarrow{\beta} n_3 \xrightarrow{\gamma} n_4\)
  - \(n_5 \xrightarrow{\delta} n_6 \xrightarrow{\gamma} n_7 \xrightarrow{\alpha} n_8 \xrightarrow{\beta}\)
Conjunctive Regular Path Queries

Idea

- Query nodes based on labelling of paths between nodes
- Express labelling by a regular expression

Example

\[ Q(a, b) := a \pi b, (\alpha \beta + \gamma \delta)^+(\pi) \]

\[
\begin{array}{c}
n_1 \xrightarrow{\alpha} n_2 \xrightarrow{\beta} n_3 \xrightarrow{\gamma} n_4 \\
\delta \downarrow \delta \\
n_5 \xrightarrow{\gamma} n_6 \xrightarrow{\alpha} n_7 \xrightarrow{\beta} n_8
\end{array}
\]
Extended Conjunctive Regular Path Queries

Idea

Comparing labelling of paths

- Regular expressions over \( n \)-tuples
- Use special symbol \( \bot \) to specify end-of-path

Example

\[
Q(a, b) := a\pi_1 b, a\pi_2 b, ([\alpha \beta]^+ [\alpha \bot])(\pi_1, \pi_2)
\]

Graph:

- \( n_1 \xrightarrow{\alpha} n_2 \xrightarrow{\alpha} n_3 \xrightarrow{\alpha} n_4 \)
- \( n_6 \xrightarrow{\beta} n_7 \xrightarrow{\beta} n_8 \)
Computation tree logic

Usage: verification of formal models

- Describe behaviour by a transition system (graph)
- Write propositions that should hold

Example

```
→ buy  → start  → idle  → work
     ↘   ↘     ↗     ↗
     ◀   ◀     ◆     ◆
        ↗   ↗
        ◀   ◀
  shutdown  crash  giveUp
```
Computation tree logic*

Usage: verification of formal models

- Describe behaviour by a *transition system* (graph)
- Write propositions that should hold

Example

![Diagram of a state transition system with states buy, start, idle, work, shutdown, crash, and giveUp.]

Machine never crashes: $\mathbf{A} \mathbf{G} \neg \text{crash}$
Computation tree logic

Usage: verification of formal models

- Describe behaviour by a *transition system* (graph)
- Write propositions that should hold

Example

Machine can work without crashing: $\mathbf{E} \mathbf{G} \neg \text{crash}$
Hybrid CTL*

Idea

- CTL* has only implicit paths and nodes
- Add ability to *name* nodes in our formulae

Example

We can get from *a* to *b* in two different ways:

\[
\mathbf{E} \downarrow v_1 \quad \mathbf{E} \mathbf{F}(\downarrow v_2 \quad \mathbf{E} \mathbf{X} \mathbf{F}(b \land \downarrow v_3 \quad \mathbf{C}_{v_1} \quad \mathbf{E}(\neg v_2 \mathbf{U} \ v_3)))
\]
Hybrid CTL*

Idea

- CTL* has only implicit paths and nodes
- Add ability to name nodes in our formulae

Example

We can get from $a$ to $b$ in two different ways:

$$E \downarrow_{v_1} EF(\downarrow_{v_2} EXF(b \land \downarrow_{v_3} \circ_{v_1} E(\neg v_2 U v_3)))$$
Hybrid CTL*

Idea

- CTL* has only implicit paths and nodes
- Add ability to *name* nodes in our formulae

Example

We can get from $a$ to $b$ in two different ways:

$$E_{\downarrow v_1} E F(\downarrow v_2 E X F(b \land \downarrow v_3 \ @_{v_1} E(\neg v_2 U v_3)))$$
Hybrid CTL*

Idea

- CTL* has only implicit paths and nodes
- Add ability to name nodes in our formulae

Example

We can get from $a$ to $b$ in two different ways:

$$
\text{E} \downarrow_{v_1} \text{E} \text{F}(\downarrow_{v_2} \text{E X F}(b \land \downarrow_{v_3} \odot_{v_1} \text{E}(\neg v_2 \cup v_3)))
$$
Hybrid CTL*

Idea

- CTL* has only implicit paths and nodes
- Add ability to \textit{name} nodes in our formulae

Example

We can get from \(a\) to \(b\) in two different ways:

\[
E\downarrow_{v_1} E\mathbf{F}(\downarrow_{v_2} E\mathbf{X}\mathbf{F}(b \land \downarrow_{v_3} \mathbf{G}_{v_1} E(\neg v_2 U v_3)))
\]
Overview

Graph Databases

Motivation

Walk Logic

Relations with FO and MSO

Relations with CTL* and Hybrid CTL*

 Conjunctive Regular Path Queries

Open Problems and Conclusion
Walk Logic

Idea: extend first-order logic

- Add walks
- Add positions on walks
- Necessary operators to compare positions

Route planning

We want to travel from our office to a cafeteria (R) and from this cafeteria get back to the office using a different route (S)

$$\exists R \exists S \exists t_1^R \exists t_2^R \exists u_1^S \exists u_2^S \exists u_3^S \left( \text{office}(t_1) \land t_1 < t_2 \land \text{cafeteria}(t_2) \land u_1 < u_3 < u_2 \land u_1 \sim t_2 \land u_2 \sim t_1 \land \forall t_3^R \left( t_1 < t_3 < t_2 \implies t_3 \nsim u_3 \right) \right)$$
Definitions

**Definition (Directed node-labeled graph)**

A directed node-labeled graph is a triple $G = (N, E, l)$:

- $N$ is a finite set of nodes
- $E \subseteq N \times N$ is the set of edges
- $l : N \rightarrow 2^{AP}$ is a node-label function
Walk Logic

- Walk variables
- Position variables per walk variable

### Atomic Formulae

<table>
<thead>
<tr>
<th>Form</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a(t))</td>
<td>Node referred to by position variable (t) has labelling (a)</td>
</tr>
<tr>
<td>(t_1 \sim t_2)</td>
<td>Position variables (t_1, t_2) refer to the same node</td>
</tr>
<tr>
<td>(t_1 &lt; t_2)</td>
<td>Position variable (t_1) comes before (t_2) in walk (W)</td>
</tr>
<tr>
<td></td>
<td>Position variables (t_1) and (t_2) must be of the same sort</td>
</tr>
</tbody>
</table>

\(\varphi, \psi\) are formulae

<table>
<thead>
<tr>
<th>Form</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\neg \varphi, \varphi \lor \psi)</td>
<td>Negation and disjunction</td>
</tr>
<tr>
<td>(\exists W \varphi)</td>
<td>Quantification over walks</td>
</tr>
<tr>
<td>(\exists t^W \varphi)</td>
<td>Quantification over positions on walks</td>
</tr>
</tbody>
</table>
## Definition

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Infinite walk</strong></td>
<td>A finite or infinite sequence $v_1 \ldots$ of nodes such that $(v_i, v_{i+1}) \in E$ for each $1 \leq i \leq</td>
</tr>
<tr>
<td><strong>Walk</strong></td>
<td>A nonempty finite sequence $v_1 \ldots v_n$ of nodes such that $(v_i, v_{i+1}) \in E$ for each $1 \leq i \leq n$</td>
</tr>
<tr>
<td><strong>Trail</strong></td>
<td>A <em>walk</em> without edge repetition</td>
</tr>
<tr>
<td><strong>Path</strong></td>
<td>A <em>walk</em> without node repetition</td>
</tr>
</tbody>
</table>
Semantics: ‘walks’?

<table>
<thead>
<tr>
<th>Definition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Infinite walk</td>
<td>A finite or infinite sequence $v_1 \ldots$ of nodes such that $(v_i, v_{i+1}) \in E$ for each $1 \leq i \leq</td>
</tr>
<tr>
<td>Walk</td>
<td>A nonempty finite sequence $v_1 \ldots v_n$ of nodes such that $(v_i, v_{i+1}) \in E$ for each $1 \leq i \leq n$</td>
</tr>
<tr>
<td>Trail</td>
<td>A walk without edge repetition</td>
</tr>
<tr>
<td>Path</td>
<td>A walk without node repetition</td>
</tr>
</tbody>
</table>

- CTL* and Hybrid CTL*: primarily infinite walks
- CRPQs: primarily walks
Semantics: expressive power

Hierarchy of expressive power

**Infinite walk**  A walk $W$ is finite:

$$\exists t^W \neg \exists u^W \ t < u$$

**Walk**  A *walk* is a *trail* (informally):

$$\forall t^W \forall u^W \ (t \sim u \land t+1 \sim u+1) \implies t = u$$

**Trail**  A *trail* $W$ is a *path* (informally):

$$\forall t^W \forall u^W \ t \sim u \implies t = u$$

Path Logic $\preceq$ Trail Logic $\preceq$ Walk Logic $\preceq$ Infinite Walk Logic
Walk-based Graph Properties

Example (Hamiltonian Path (in Path Logic))

\[ \exists P \forall Q \forall t^Q \exists u^P (t \sim u) \]

Example (Eulerian Trail (in Trail Logic))

\[ \exists T \forall Q \forall t^Q \exists u^T (t \sim u) \land (t_{+1} \sim u_{+1}) \]

Example (Strongly Connected)

\[ \forall P \forall Q \forall t^P \forall u^Q \exists R \exists v^R \exists w^R (v < w \land t \sim v \land u \sim w) \]
Properties on undirected graphs

**Theorem**

*Weakly Connected is not expressible on directed graphs*

**Proof.**

\[ n_1 \leftrightarrow n_2 \rightarrow n_3 \leftrightarrow n_4 \rightarrow n_5 \leftrightarrow n_6 \]

All walks contain at most 2 nodes: *reduce to first-order logic*

- Direction matters!
- On undirected graphs:
  - *Weakly Connected* same way as strongly connected
  - *Planar Graph* using Kuratowski’s Theorem
Overview

Graph Databases

Motivation

Walk Logic

Relations with FO and MSO

Relations with CTL* and Hybrid CTL*

Conjunctive Regular Path Queries

Open Problems and Conclusion
MSO(nodes, edges) and paths

Observations

- Path: sequence of connected edges
- No node repetition: nodes and positions coincide
- Node \(a\) before node \(b\) on path \(P\) if and only if Node \(b\) is reachable from \(a\) using the edges in \(P\)

Theorem

*Path Logic *\( \preceq \) *MSO(nodes, edges)*
Set-based Graph Properties

Theorem

Bipartite graph is not expressible on directed graphs

Lemma (Dénes Kőnig)

A graph is bipartite iff it does not contain an odd cycle

Proof.

\[
\begin{align*}
n_2 & \rightarrow n_3 \\
n_1 & \quad \downarrow \quad \downarrow \\
m_2 & \rightarrow m_3 \rightarrow m_4 \\
m_1 & \leftarrow m_6 \leftarrow m_5
\end{align*}
\]

All walks contain at most 3 nodes: reduce to first-order logic

- MSO(nodes) *can* express bipartite graph
- Is Walk Logic strictly subsumed by MSO?
**Eulerian Trail**

**Theorem**

*MSO*(nodes, edges) cannot express Eulerian Trail

**Lemma (well known result)**

*MSO* cannot distinguish sets with *i* from sets with *j* elements

**Proof.**

For MSO: existence of Eulerian Trail in the graph

\[
\begin{align*}
  a_n & \quad v_2 \quad b_m \\
  \vdots & \quad v_1 \quad \vdots \\
  a_1 & \quad b_1
\end{align*}
\]

\[
\begin{align*}
  a_n & \quad b_m \\
  \vdots & \quad \vdots \\
  a_1 & \quad b_1
\end{align*}
\]

Reduces to sets *A* and *B* having the equal number of elements
Relations with FO and MSO

- We have $\text{FO} \prec \text{Path Logic} \prec \text{MSO(nodes, edges)}$
- Trail Logic, Walk Logic, and Infinite Walk Logic are incomparable with $\text{MSO(nodes)}$ and $\text{MSO(nodes, edges)}$

**Lemma (Courcelle and Engelfriet)**

$\text{MSO(nodes)}$ cannot express Hamiltonian Path

- Path Logic and $\text{MSO(nodes)}$ are incomparable

$\text{Path Logic} \prec \text{Trail Logic} \prec ^1 \text{Walk Logic} \prec \text{Infinite Walk Logic}$

---

^1The proof for $\text{Trail Logic} \prec \text{Walk Logic}$ is omitted but is similar to the proof of $\text{Path Logic} \prec \text{Trail Logic}$
Overview

Graph Databases

Motivation

Walk Logic

Relations with FO and MSO

Relations with CTL* and Hybrid CTL*

Conjunctive Regular Path Queries

Open Problems and Conclusion
Definition

Let $a$ be an atomic proposition, $x$ a node variable, $\varphi_1$ and $\varphi_2$ node formulas, $\psi_1$ and $\psi_2$ path formulas.

Node formulas

$$\varphi ::= a \mid x \mid \neg \varphi_1 \mid \varphi_1 \lor \varphi_2 \mid \downarrow x \varphi_1 \mid \emptyset x \varphi_1 \mid E \psi$$

Path formulas

$$\psi ::= \varphi \mid \neg \psi_1 \mid \psi_1 \lor \psi_1 \mid X \psi_1 \mid \psi_1 U \psi_2$$
# Translating Hybrid CTL* to Walk Logic

## Idea

<table>
<thead>
<tr>
<th><strong>Node formulas</strong></th>
<th>Properties of single node:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>translate to properties of single position</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Hybrid extensions</strong></th>
<th>Named nodes:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>translate to named position variables</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Path formulas</strong></th>
<th>Properties on a single path with forward navigation:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>translate to walk variable; keep track of current position using position variables and ≤</td>
</tr>
</tbody>
</table>
## Translating Hybrid CTL* to Walk Logic

### Idea

- **Node formulas**  Properties of single node:  
  
  \( \text{translate to properties of single position} \)

- **Hybrid extensions**  Named nodes:  
  
  \( \text{translate to named position variables} \)

- **Path formulas**  Properties on a single path with forward navigation:  
  
  \( \text{translate to walk variable; keep track of current position using position variables and <} \)

\[ \text{CTL}^* \preceq \text{Hybrid CTL}^* \preceq \text{Infinite Walk Logic} \]
# Hybrid $\text{CTL}^* \prec \text{Infinite Walk Logic}$?

## Theorem

$\text{Hybrid } \text{CTL}^* \prec \text{Infinite Walk Logic}$

## Proof.

- $\text{CTL}^* \prec \text{Infinite Walk Logic}$ as $\text{CTL}^*$ is invariant under bisimulation

- $\text{Hybrid } \text{CTL}^* \prec \text{Infinite Walk Logic}$ as Hybrid $\text{CTL}^*$ is invariant under generated submodels
Theorem

Hybrid CTL* ⪯ Infinite Walk Logic

Proof.

- CTL* ⪯ Infinite Walk Logic as CTL* is invariant under bisimulation
- Hybrid CTL* ⪯ Infinite Walk Logic as Hybrid CTL* is invariant under generated submodels

CTL* ⪯ Hybrid CTL* ⪯ Infinite Walk Logic
Hybrid $\text{CTL}^* \prec \text{Infinite Walk Logic}$?

**Theorem**

$\text{Hybrid CTL}^* \prec \text{Infinite Walk Logic}$

**Proof.**

- $\text{CTL}^* \prec \text{Infinite Walk Logic}$ as $\text{CTL}^*$ is invariant under bisimulation
- $\text{Hybrid CTL}^* \prec \text{Infinite Walk Logic}$ as Hybrid $\text{CTL}^*$ is invariant under generated submodels

$\text{CTL}^* \prec \text{Hybrid CTL}^* \prec \text{Infinite Walk Logic}$

Walk Logic $\preceq$ Infinite Walk Logic
Overview

Graph Databases

Motivation

Walk Logic

Relations with FO and MSO

Relations with CTL* and Hybrid CTL*

Conjunctive Regular Path Queries

Open Problems and Conclusion
### CRPQs versus Walk Logics

#### Different languages!

<table>
<thead>
<tr>
<th>Focus on path labelling versus focus on path-structure of graphs</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶ All CRPQs are incomparable with all Walk Logics.</td>
</tr>
</tbody>
</table>
# CRPQs versus Walk Logics

**Different languages!**

Focus on path labelling versus focus on path-structure of graphs

- All CRPQs are incomparable with all Walk Logics.

- Similar semantically questions
- Similar proof techniques
Some results

- Hamiltonian path cannot be expressed
- Eulerian trail cannot be expressed
Some results

- Hamiltonian path cannot be expressed
- Eulerian trail cannot be expressed

Paths versus Walks

CRPQ with paths can express queries not expressible in the strongest language with Walks!
Overview

Graph Databases

Motivation

Walk Logic

Relations with FO and MSO

Relations with CTL* and Hybrid CTL*

Conjunctive Regular Path Queries

Open Problems and Conclusion
Open Problems

- **Walk Logic versus Infinite Walk Logic:**
  - Infinite walks are the standard in verification logics
  - Can we express the verification logics in Walk Logic?
  - Also interesting: finite CTL* versus infinite CTL*

- **Complexity bounds on model checking for WL:**
  - WL model checking is decidable
  - Current approach has horrible complexity
Conclusion

- General walk-based reasoning on graphs
- Relates to practical graph languages
- Framework for studying expressivity