Symmetric Datalog $\neq$ Linear Datalog

László Egri[1],

joint work with Benoît Larose[2] and Pascal Tesson[3]

[1]McGill University

[2]Concordia University

[3]Université Laval

UA and CSP, Nashville June 2007

Introduction

- Symmetric Datalog (LE, Larose, Tesson, 2007)

Introduction

- Symmetric Datalog (LE, Larose, Tesson, 2007)
  - In logarithmic space (using Reingold, 2005)

Introduction

- Symmetric Datalog (LE, Larose, Tesson, 2007)
  - In logarithmic space (using Reingold, 2005)
  - Boolean domains + standard complexity assumptions $\longrightarrow$ all CSPs in L are in symmetric Datalog

Introduction

- Symmetric Datalog (LE, Larose, Tesson, 2007)
  - In logarithmic space (using Reingold, 2005)
  - Boolean domains + standard complexity assumptions $\longrightarrow$ all CSPs in L are in symmetric Datalog
  - Conjecture: all CSPs in L are in symmetric Datalog

Introduction

- Symmetric Datalog (LE, Larose, Tesson, 2007)
    - In logarithmic space (using Reingold, 2005)
    - Boolean domains + standard complexity assumptions $\longrightarrow$ all CSPs in L are in symmetric Datalog
    - Conjecture: all CSPs in L are in symmetric Datalog
- Undirected st-connectivity <u>is</u> definable in symmetric Datalog

Introduction

- Symmetric Datalog (LE, Larose, Tesson, 2007)
  - In logarithmic space (using Reingold, 2005)
  - Boolean domains + standard complexity assumptions $\longrightarrow$ all CSPs in L are in symmetric Datalog
  - Conjecture: all CSPs in L are in symmetric Datalog
- Undirected st-connectivity <u>is</u> definable in symmetric Datalog
- Directed st-connectivity <u>is not</u> definable in symmetric Datalog

Introduction

- Symmetric Datalog (LE, Larose, Tesson, 2007)
  - In logarithmic space (using Reingold, 2005)
  - Boolean domains + standard complexity assumptions $\longrightarrow$ all CSPs in L are in symmetric Datalog
  - Conjecture: all CSPs in L are in symmetric Datalog
- Undirected st-connectivity <u>is</u> definable in symmetric Datalog
- Directed st-connectivity <u>is not</u> definable in symmetric Datalog
  - Reflexive transitive closure of a binary relation <u>is not</u> definable in symmetric Datalog

Introduction

- Symmetric Datalog (LE, Larose, Tesson, 2007)
  - In logarithmic space (using Reingold, 2005)
  - Boolean domains + standard complexity assumptions $\longrightarrow$ all CSPs in L are in symmetric Datalog
  - Conjecture: all CSPs in L are in symmetric Datalog
- Undirected st-connectivity <u>is</u> definable in symmetric Datalog
- Directed st-connectivity <u>is not</u> definable in symmetric Datalog
  - Reflexive transitive closure of a binary relation <u>is not</u> definable in symmetric Datalog
  - $\neg\mathrm{CSP}(\langle\{0,1\};\leq,\{0\},\{1\}\rangle)$ <u>is not</u> definable in symmetric Datalog

Outline

- Recap symmetric Datalog through an example

Outline

- Recap symmetric Datalog through an example
- Definitions: **free derivation path**, **the free structure**

Outline

- Recap symmetric Datalog through an example
- Definitions: **free derivation path**, **the free structure**
- Overview of the general proof

Outline

- Recap symmetric Datalog through an example
- Definitions: **free derivation path**, **the free structure**
- Overview of the general proof
- The main idea through an example

Datalog and Derivation Path Example

Input Vocabulary:

$$S^1, T^1, E^2$$

Linear (Symmetric) Program:

EDB: Extensional Database Predicate

IDB: Intensional Database Predicate

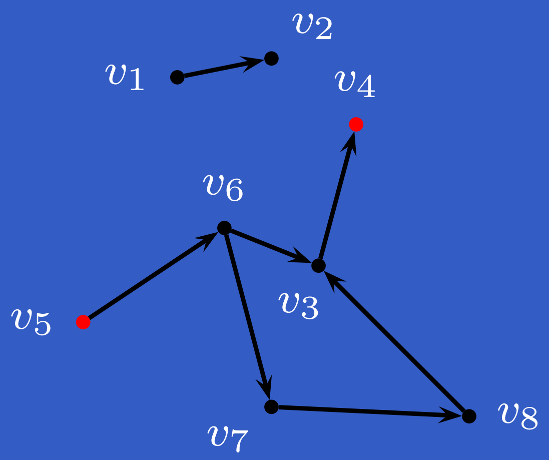
$$I(y) \leftarrow S(y)$$

$$I(y) \leftarrow I(x); E(x, y)$$

$$(I(x) \leftarrow I(y); E(x, y))$$

$$G \leftarrow I(y); T(y)$$

Input Structure:



$$S = \{v_5\}, T = \{v_4\}$$

Derivation Path:

$$\begin{tikzcd}
 & & & & G \arrow[dl, dash] \arrow[dr, dash] & \\
 & & & I(v_4) \arrow[dl, dash] \arrow[dr, dash] & & T(v_4) \\
 & & I(v_3) \arrow[dl, dash] \arrow[dr, dash] & & E(v_3, v_4) & \\
 & I(v_6) \arrow[dl, dash] \arrow[dr, dash] & & E(v_6, v_3) & & \\
 I(v_5) \arrow[d, dash] & & E(v_5, v_6) & & & \\
 S(v_5) & & & & &
\end{tikzcd}$$

The Free Derivation Path

Symmetric Program $\mathfrak{D}$:

$$I(y) \leftarrow S(y)$$

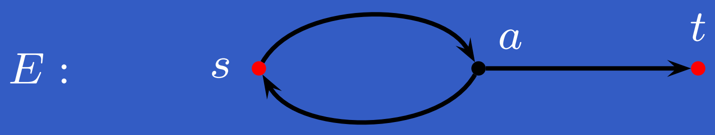
$$I(y) \leftarrow I(x); E(x,y)$$

$$I(x) \leftarrow I(y); E(x,y)$$

(Rename the vars: $I(y) \leftarrow I(x); E(y,x)$)

$$G \leftarrow I(y); T(y)$$

Input Structure:



$$S = \{s\}, T = \{t\}$$

Derivation Path:

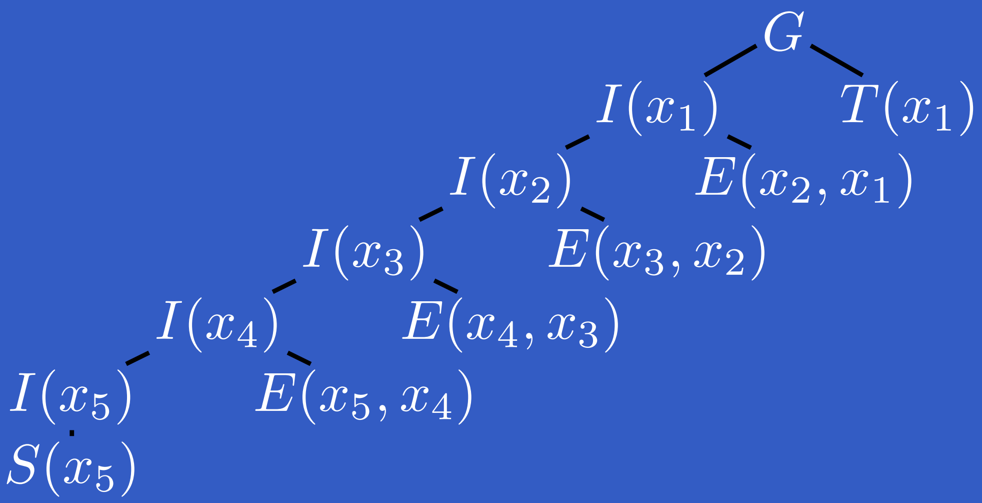
$$\begin{tikzcd}
& & & & & G \arrow[dl, no head] \arrow[dr, no head] & \\
& & & & I(t) \arrow[dl, no head] \arrow[dr, no head] & & T(t) \\
& & & I(a) \arrow[dl, no head] \arrow[dr, no head] & & E(a,t) & \\
& & I(s) \arrow[dl, no head] \arrow[dr, no head] & & E(s,a) & & \\
& I(a) \arrow[dl, no head] \arrow[dr, no head] & & E(a,s) & & & \\
I(s) \arrow[d, no head] & & E(s,a) & & & & \\
S(s) & & & & & &
\end{tikzcd}$$

Free Derivation Path:

$$\begin{tikzcd}
& & & & & G \arrow[dl, no head] \arrow[dr, no head] & \\
& & & & I(x_1) \arrow[dl, no head] \arrow[dr, no head] & & T(x_1) \\
& & & I(x_2) \arrow[dl, no head] \arrow[dr, no head] & & E(x_2,x_1) & \\
& & I(x_3) \arrow[dl, no head] \arrow[dr, no head] & & E(x_3,x_2) & & \\
& I(x_4) \arrow[dl, no head] \arrow[dr, no head] & & E(x_4,x_3) & & & \\
I(x_5) \arrow[d, no head] & & E(x_5,x_4) & & & & \\
S(x_5) & & & & & &
\end{tikzcd}$$

The Free Structure

Free Derivation Path $\mathcal{F}$:



Free Structure $\mathbf{F}$:

Domain: $F = \{x_1, x_2, x_3, x_4, x_5\}$

$$E^{\mathbf{F}} : \begin{tikzcd} \overset{x_5}{\bullet} \arrow[r] & \overset{x_4}{\bullet} \arrow[r] & \overset{x_3}{\bullet} \arrow[r] & \overset{x_2}{\bullet} \arrow[r] & \overset{x_1}{\bullet} \end{tikzcd}$$

$$S^{\mathbf{F}} = \{x_5\}, T^{\mathbf{F}} = \{x_1\}$$

- The free structure $\mathbf{F}$ is accepted by $\mathfrak{D}$

*

Proof Strategy

- Assume $\mathfrak{D}$ works

Proof Strategy

- Assume $\mathfrak{D}$ works
- Input: long enough path

Proof Strategy

- Assume $\mathfrak{D}$ works
- Input: long enough path
- Abstract away, i.e. take the free derivation path $\mathcal{F}$

Proof Strategy

- Assume $\mathfrak{D}$ works
- Input: long enough path
- Abstract away, i.e. take the free derivation path $\mathcal{F}$
- Using the symmetricity of $\mathfrak{D}$, “zig-zag” on $\mathcal{F}$ to create a new free derivation path $\mathcal{F}'$ such that:

Proof Strategy

- Assume $\mathfrak{D}$ works
- Input: long enough path
- Abstract away, i.e. take the free derivation path $\mathcal{F}$
- Using the symmetricity of $\mathfrak{D}$, “zig-zag” on $\mathcal{F}$ to create a new free derivation path $\mathcal{F}'$ such that:
  - In $\mathcal{F}'$, there is no path from the vertex in S to the vertex in T

Proof Strategy

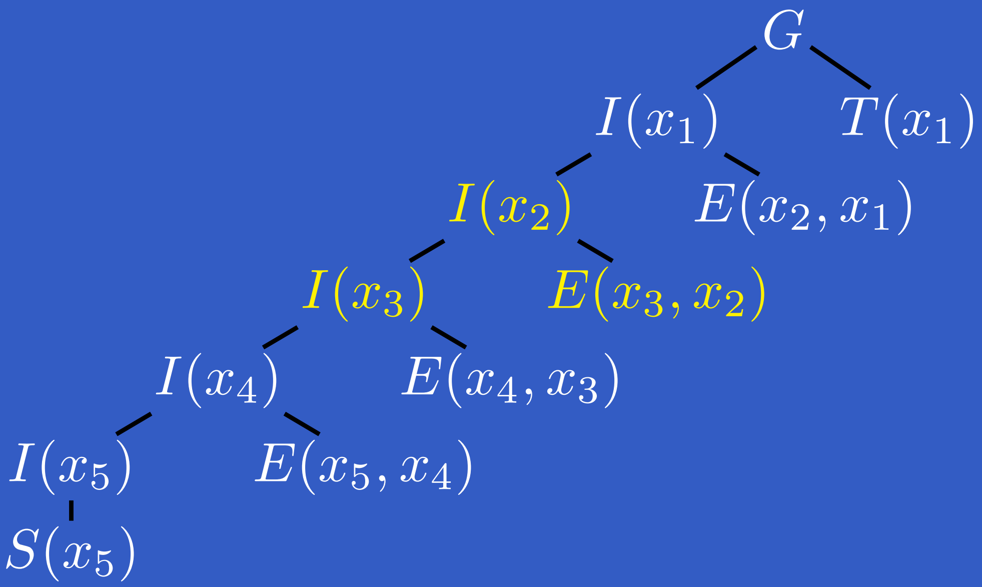
- Assume $\mathfrak{D}$ works
- Input: long enough path
- Abstract away, i.e. take the free derivation path $\mathcal{F}$
- Using the symmetricity of $\mathfrak{D}$, “zig-zag” on $\mathcal{F}$ to create a new free derivation path $\mathcal{F}'$ such that:
  - In $\mathcal{F}'$, there is no path from the vertex in S to the vertex in T
  - $\mathcal{F}'$ is a valid derivation path for $\mathfrak{D}$ over the free structure of $\mathcal{F}'$

Proof Strategy

- Assume $\mathfrak{D}$ works
- Input: long enough path
- Abstract away, i.e. take the free derivation path $\mathcal{F}$
- Using the symmetricity of $\mathfrak{D}$, “zig-zag” on $\mathcal{F}$ to create a new free derivation path $\mathcal{F}'$ such that:
  - In $\mathcal{F}'$, there is no path from the vertex in S to the vertex in T
  - $\mathcal{F}'$ is a valid derivation path for $\mathfrak{D}$ over the free structure of $\mathcal{F}'$
- Contradiction

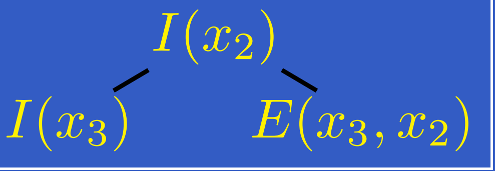
Zig-Zag (Simple Example)

Free Derivation Path $\mathcal{F}$:

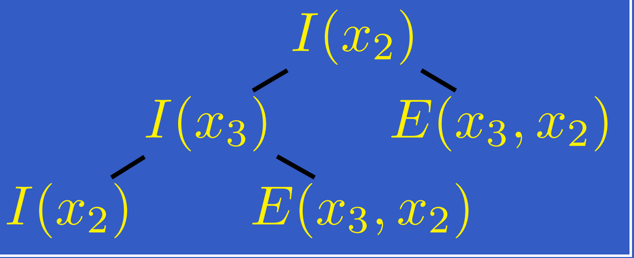


Zig-zag (mirror) the yellow segment:

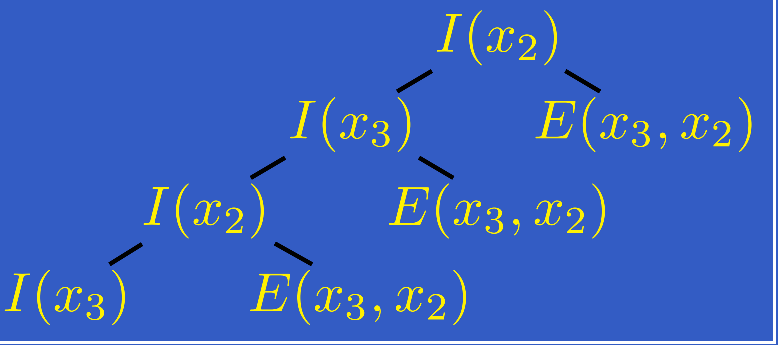
1.



2.



3.



$$I(y) \leftarrow S(y)$$

$$I(y) \leftarrow I(x); E(x, y)$$

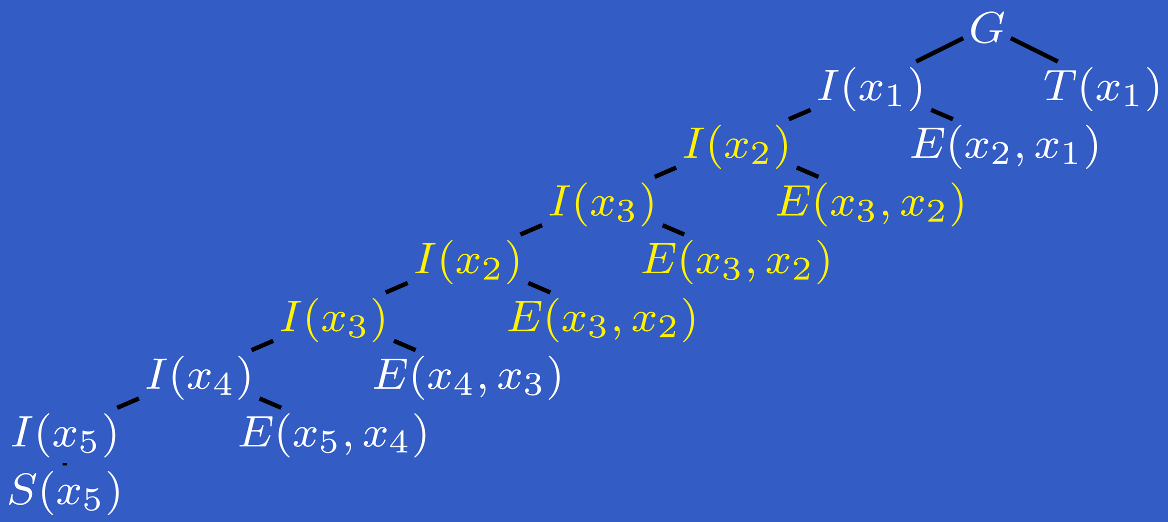
$$I(x) \leftarrow I(y); E(x, y)$$

$$G \leftarrow I(y); T(y)$$

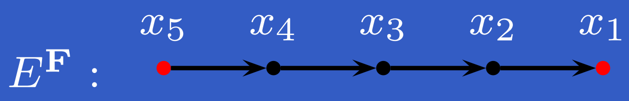
*

Zig-Zag Continued (Simple Example)

Before renaming the variables in mirrored $\mathcal{F}$

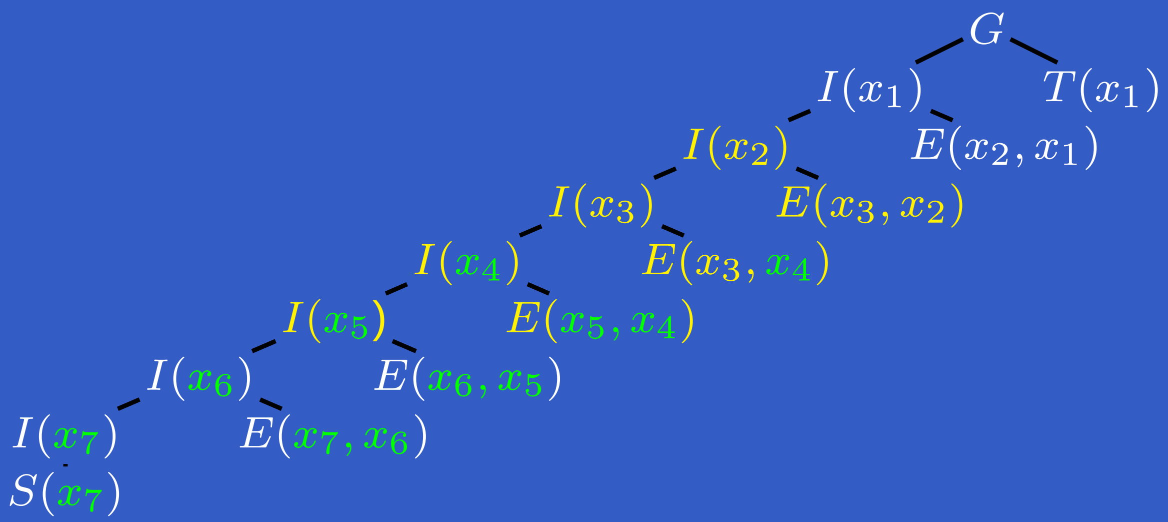


Structure $\mathbf{F}$

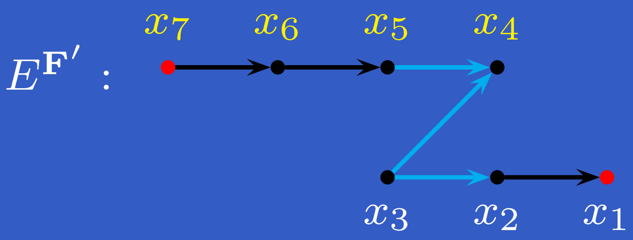


$$S^{\mathbf{F}} = \{x_5\}, T^{\mathbf{F}} = \{x_1\}$$

$\mathcal{F}'$ (variables are renamed):



Structure $\mathbf{F}'$:



$$S^{\mathbf{F}'} = \{x_7\}, T^{\mathbf{F}'} = \{x_1\}$$

About The General Proof

- Two main complications:

About The General Proof

- Two main complications:
  - There could be more than one path from the vertex in S to the vertex in T in a free derivation path. See free structure.

About The General Proof

- Two main complications:
  - There could be more than one path from the vertex in S to the vertex in T in a free derivation path. See free structure.
    - We disconnect each

About The General Proof

- Two main complications:
  - There could be more than one path from the vertex in S to the vertex in T in a free derivation path. See free structure.
    - We disconnect each
    - Careful, we do not want to create new paths when we disconnect a path

About The General Proof

- Two main complications:
  - There could be more than one path from the vertex in S to the vertex in T in a free derivation path. See free structure.
    - We disconnect each
    - Careful, we do not want to create new paths when we disconnect a path
    - A bit technical

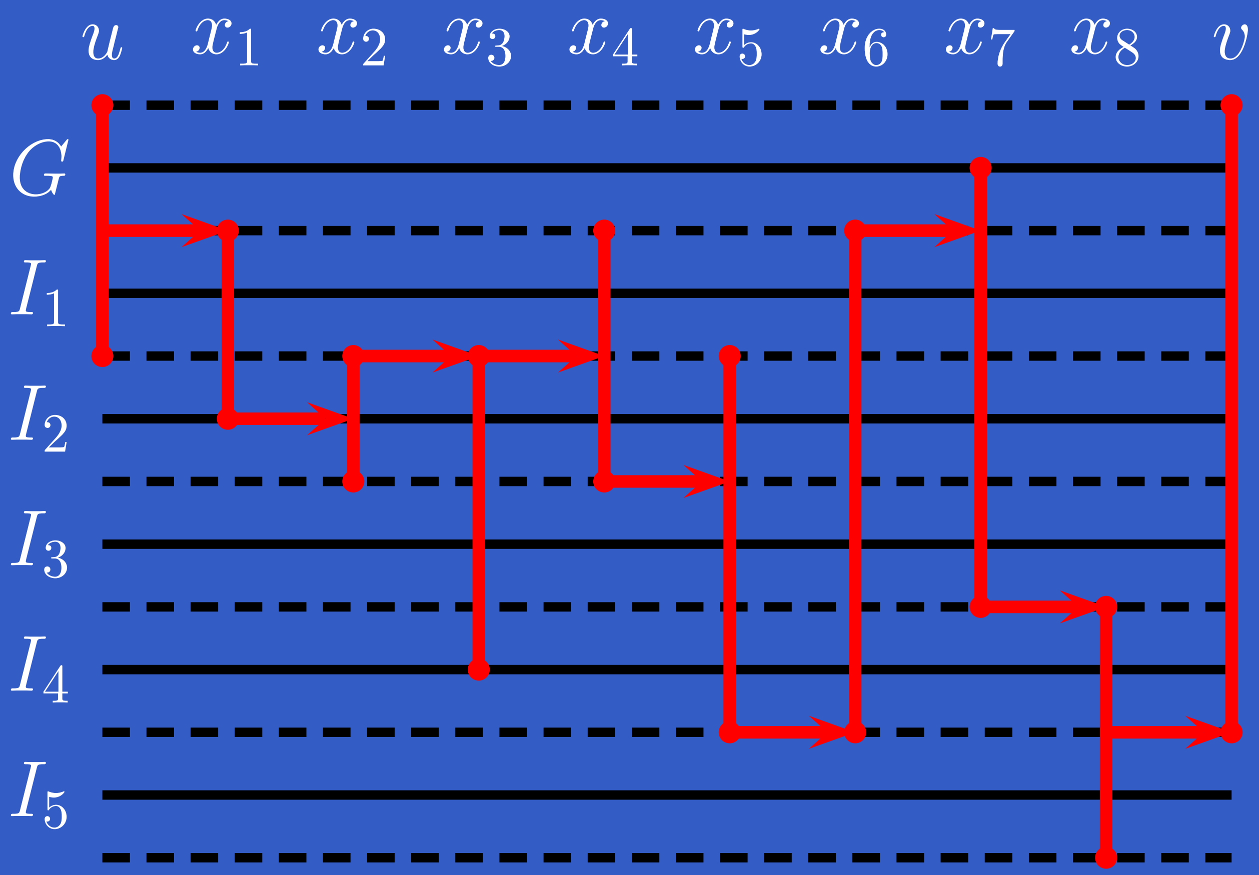
About The General Proof

- Two main complications:
  - There could be more than one path from the vertex in S to the vertex in T in a free derivation path. See free structure.
    - We disconnect each
    - Careful, we do not want to create new paths when we disconnect a path
    - A bit technical
  - Arity of the IDBs can be arbitrary (but fixed). See our example program.

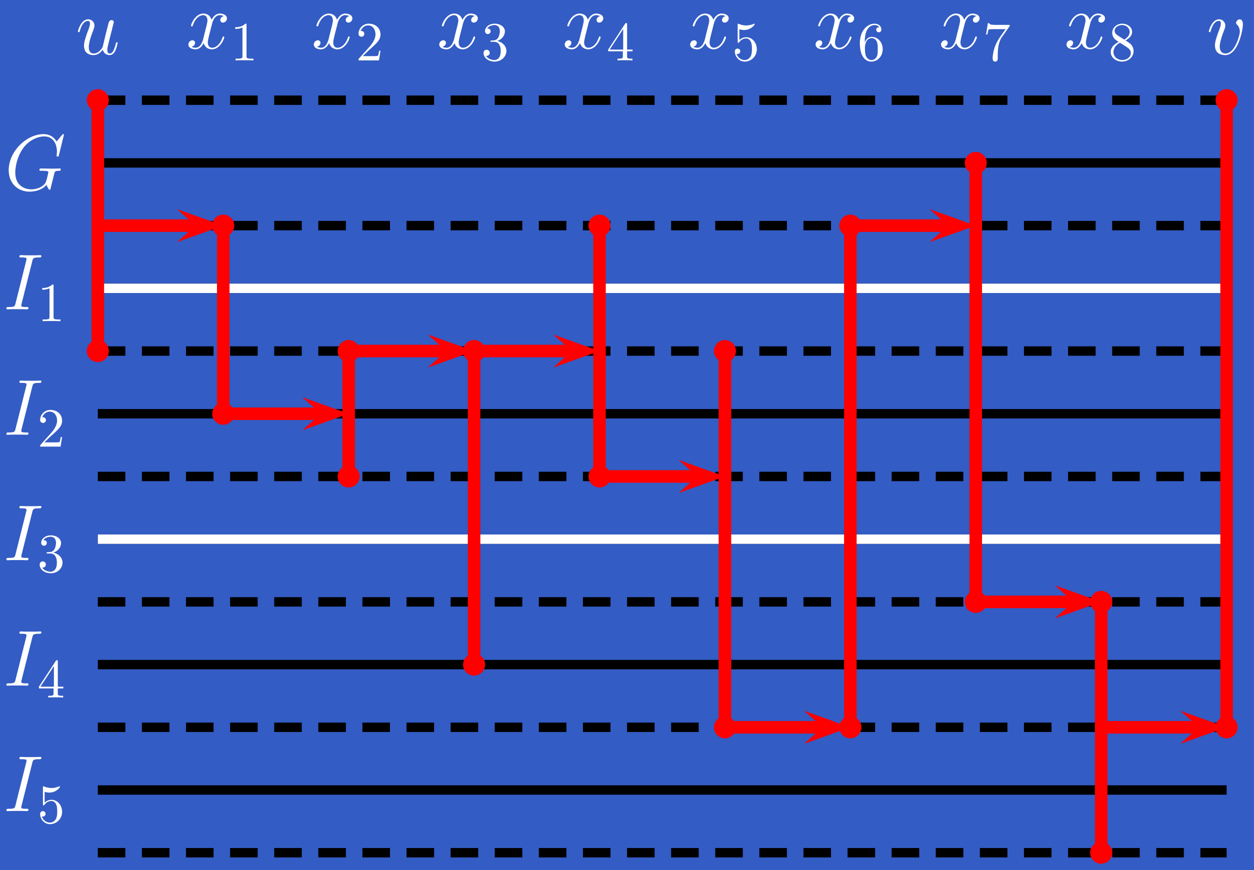
About The General Proof

- Two main complications:
  - There could be more than one path from the vertex in S to the vertex in T in a free derivation path. See free structure.
    - We disconnect each
    - Careful, we do not want to create new paths when we disconnect a path
    - A bit technical
  - Arity of the IDBs can be arbitrary (but fixed). See our example program.
    - We give an intuition how to handle higher arities.

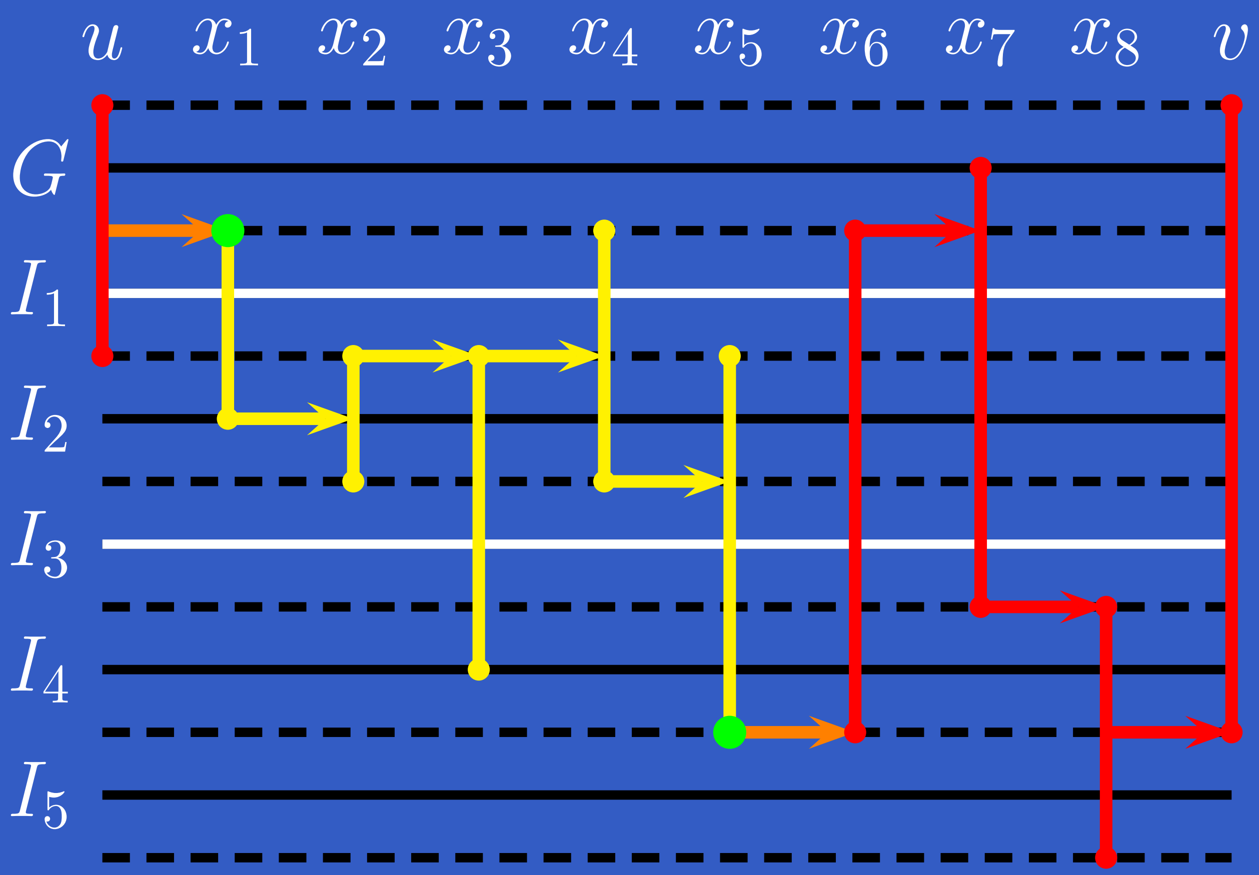
The UV-Path Following Diagram



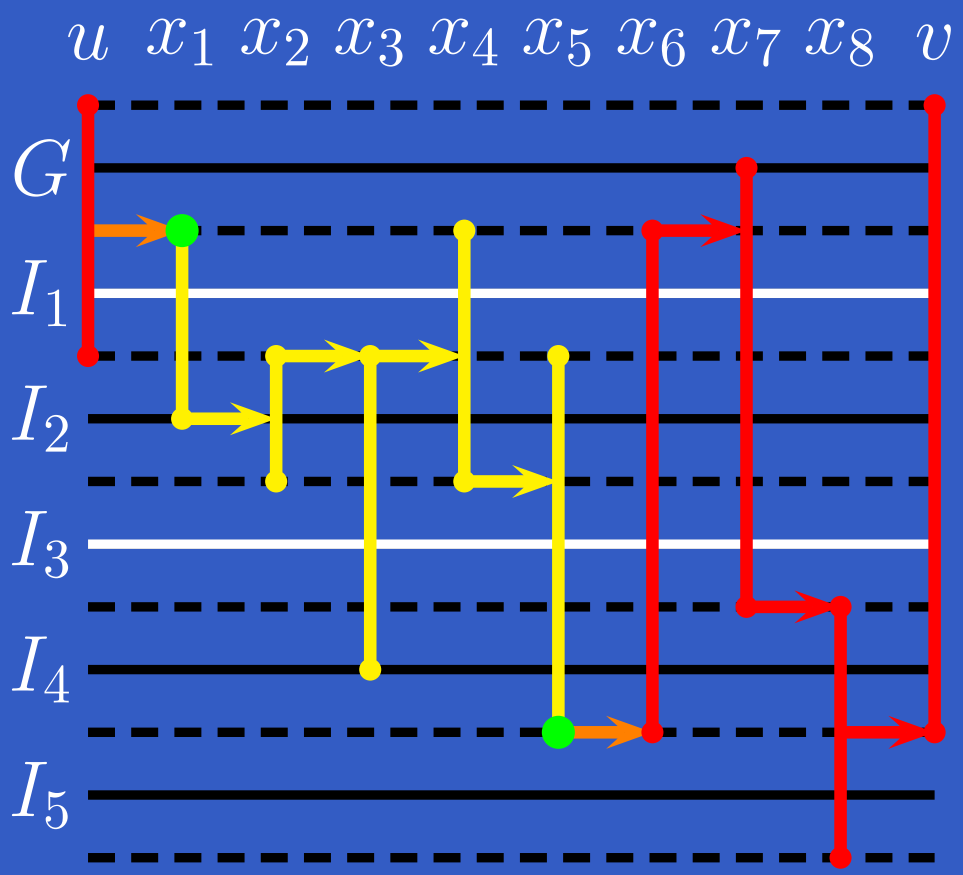
The UV-Path Following Diagram

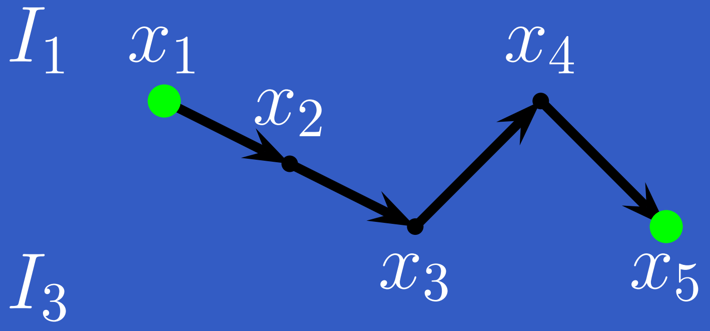


The UV-Path Following Diagram

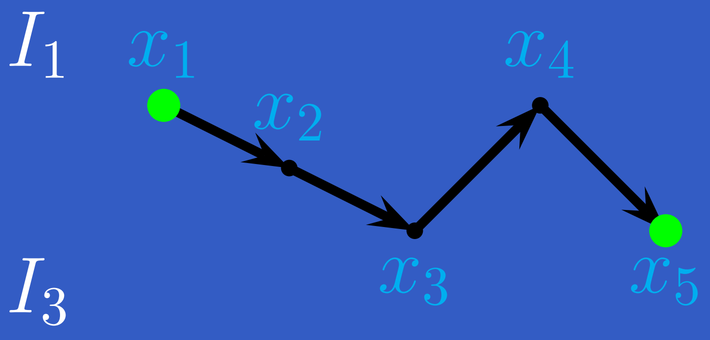


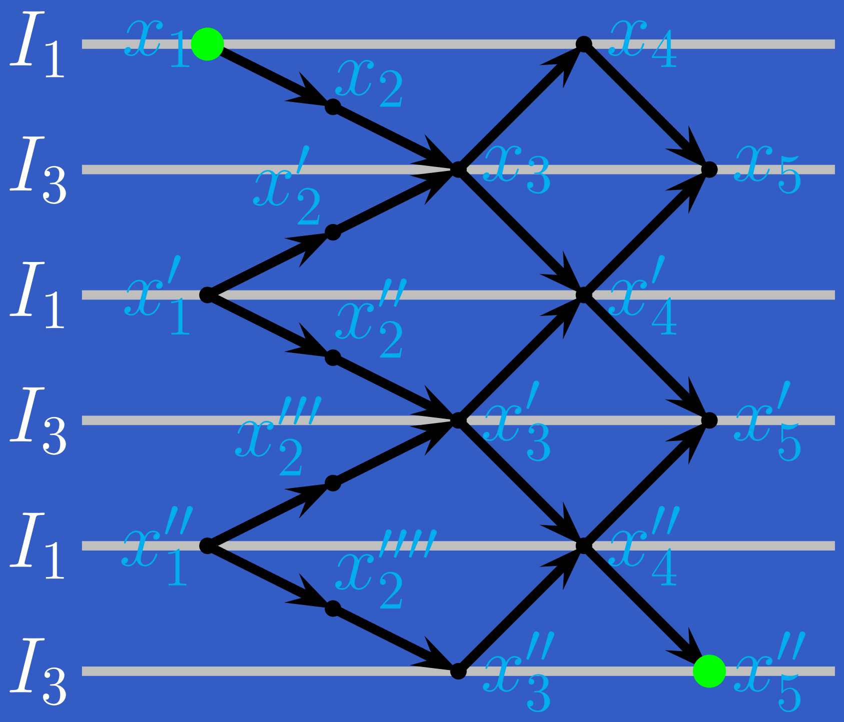
The UV-Path Following Diagram





Zig-Zag


I_1
I_3
x_1
x_2
x_3
x_4
x_5


I_1
I_3
I_1
I_3
I_1
I_3
x_1
x_2
x_4
x_2'
x_3
x_5
x_1'
x_2''
x_4'
x_2'''
x_3'
x_5'
x_1''
x_2''''
x_4''
x_3''
x_5''

Questions

?

Questions

?

Questions

?

Questions

?

Questions

?

Questions

?

Questions

?

Questions

Questions

¿

Questions

¿

Questions

¿

Questions

Questions

Questions

?

Questions

Questions

?

Questions

?

Questions

?

Questions

?