

View-based query processing

Maurizio Lenzerini

Dipartimento di Informatica e Sistemistica
Università di Roma “La Sapienza”

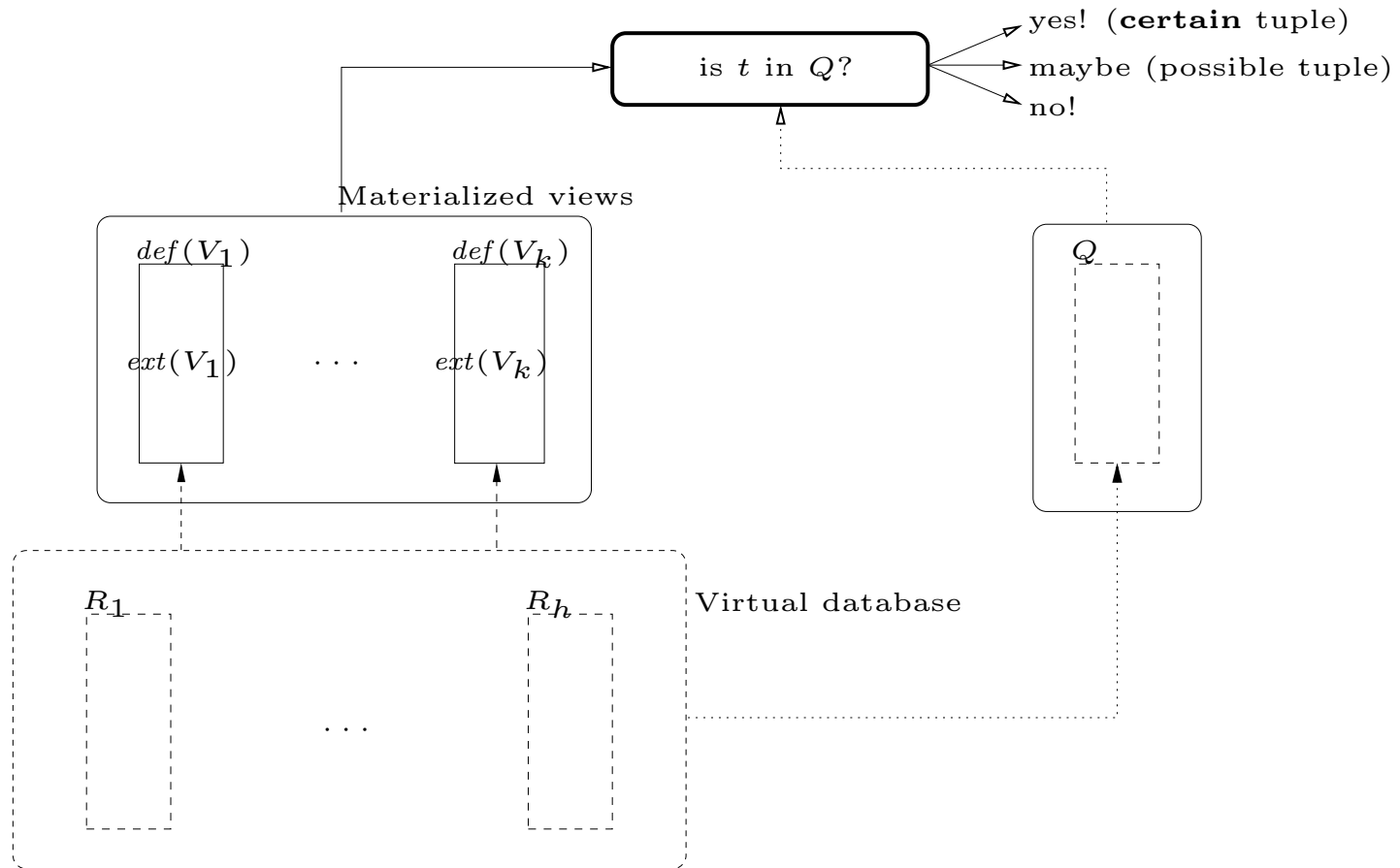
Symposium on the Effectiveness of Logic in Computer Science
in Honor of Moshe Y. Vardi

Saarbruecken, 4–6 March, 2002

Outline

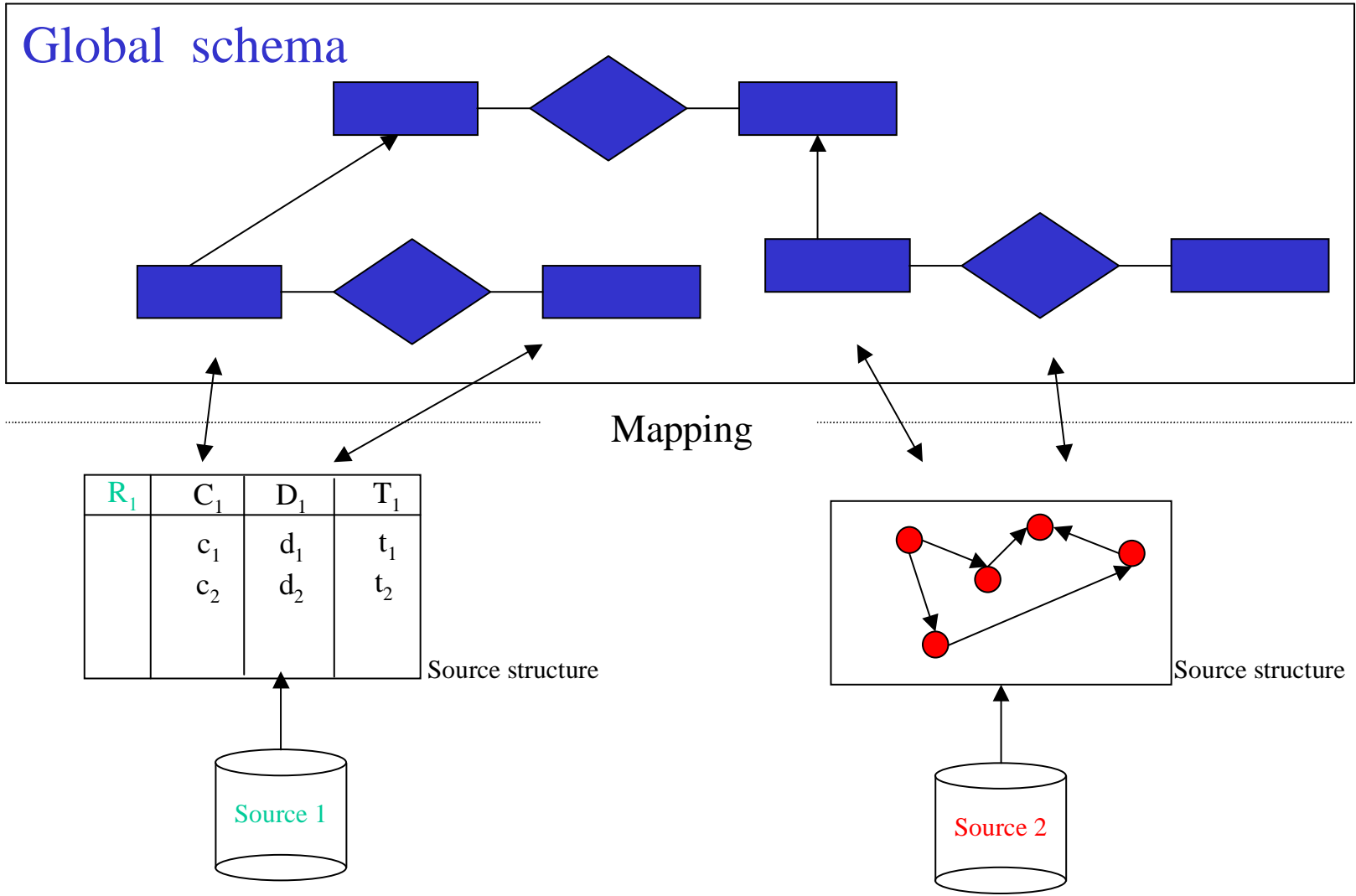
- View-based query processing (in data integration)
- Query answering in LAV
- Query answering in GAV
- Beyond FOL
- Conclusions

View-based query processing



Answer a query based on a set of views, rather than on the raw data in the database. Relevant problem in **data integration**.

Architecture for data integration



Framework for data integration

A data integration system \mathcal{I} is a triple $\langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, where

- \mathcal{G} is the global schema (structure and constraints),
- \mathcal{S} is the source schema (structures and constraints),
- \mathcal{M} is the mapping between \mathcal{G} and \mathcal{S} .

The semantics of \mathcal{I} determines which data satisfy \mathcal{G} . If \mathcal{C} is a source database (source data coherent with \mathcal{S}) over a fixed domain Γ :

$$\text{mod}^{\mathcal{C}}(\mathcal{I}) = \{ \mathcal{B} \mid \mathcal{B} \text{ is a database that is legal for } \mathcal{I} \text{ wrt } \mathcal{C}, \\ \text{i.e., that satisfies both } \mathcal{G} \text{ and } \mathcal{M} \text{ wrt } \mathcal{C} \}$$

A query q to \mathcal{I} is expressed over \mathcal{G} , and the set of certain answers to q wrt \mathcal{I} and \mathcal{C} is

$$q^{\mathcal{I}, \mathcal{C}} = \{ \vec{t} \mid \forall \mathcal{B} \in \text{mod}^{\mathcal{C}}(\mathcal{I}), \vec{t} \in q^{\mathcal{B}} \}$$

The truth, the whole truth, nothing but the truth!

Approaches to modeling data integration systems

- **Local-as-view (LAV)**: sources are defined as **views** over the global schema, i.e., the mapping \mathcal{M} associates to each element s in \mathcal{S} a query $\phi_{\mathcal{G}}$ over \mathcal{G} . Assertions in \mathcal{M} have the form

$$s \rightsquigarrow \phi_{\mathcal{G}} \quad \text{with the meaning} \quad \forall \vec{x} (s(\vec{x}) \rightarrow \phi_{\mathcal{G}}(\vec{x}))$$

- **Global-as-view (GAV)**: the global schema is defined in terms of the sources, i.e., the mapping \mathcal{M} associates to each element g in \mathcal{G} a **view** $\phi_{\mathcal{S}}$ over \mathcal{S} . Assertions in \mathcal{M} have the form

$$g \rightsquigarrow \phi_{\mathcal{S}} \quad \text{with the meaning} \quad \forall \vec{x} (\phi_{\mathcal{S}}(\vec{x}) \rightarrow g(\vec{x}))$$

We restrict our attention to **sound** sources. Other possibilities are

- **complete** (e.g., $\forall \vec{x} (s(\vec{x}) \leftarrow \phi_{\mathcal{G}}(\vec{x}))$)
- **exact** (e.g., $\forall \vec{x} (s(\vec{x}) \leftrightarrow \phi_{\mathcal{G}}(\vec{x}))$).

Global-as-view vs local-as-view – Example

Global schema: $\text{movie}(Title, Year, Director)$
 $\text{european}(Director)$
 $\text{review}(Title, Critique)$

Source 1: $r_1(Title, Year, Director)$ since 1960, european directors

Source 2: $r_2(Title, Critique)$ since 1990

Example of query: Title and critique of movies in 1998
 $\{ (T, R) \mid \text{movie}(T, 1998, D) \wedge \text{review}(T, R) \}$

Local-as-view – Example

Global schema: $\text{movie}(Title, Year, Director)$
 $\text{european}(Director)$
 $\text{review}(Title, Critique)$

Local-as-view: associated to relations at the sources we have **views** over the global schema

$$\begin{aligned} r_1(T, Y, D) &\rightsquigarrow \{ (T, Y, D) \mid \text{movie}(T, Y, D) \wedge \text{european}(D) \wedge Y \geq 1960 \} \\ r_2(T, R) &\rightsquigarrow \{ (T, R) \mid \text{movie}(T, Y, D) \wedge \text{review}(T, R) \wedge Y \geq 1990 \} \end{aligned}$$

Query $\{ (T, R) \mid \text{movie}(T, 1998, D) \wedge \text{review}(T, R) \}$ is processed by means of an inference mechanism that aims at re-expressing the atoms of the global schema in terms of atoms at the sources. In this case:

$$\{ (T, R) \mid r_2(T, R) \wedge r_1(T, 1998, D) \}$$

Global-as-view – Example

Global schema: $\text{movie}(Title, Year, Director)$
 $\text{european}(Director)$
 $\text{review}(Title, Critique)$

Global-as-view: associated to relations in the global schema we have **views** over the sources

$$\text{movie}(T, Y, D) \rightsquigarrow \{ (T, Y, D) \mid r_1(T, Y, D) \}$$

$$\text{european}(D) \rightsquigarrow \{ (D) \mid r_1(T, Y, D) \}$$

$$\text{review}(T, R) \rightsquigarrow \{ (T, R) \mid r_2(T, R) \}$$

Query $\{ (T, R) \mid \text{movie}(T, 1998, D) \wedge \text{review}(T, R) \}$ can be processed by means of unfolding, i.e., by expanding the atoms according to their definitions, so as to come up with source relations. In this case:

$$\{ (T, R) \mid r_2(T, R) \wedge r_1(T, 1998, D) \}$$

Global-as-view vs local-as-view

Local-as-view (Information Manifold, DWQ, Picstel, ...): The mapping \mathcal{M} does **not** provide direct information about which data satisfies the global schema. The only information I have on the global database is that certain tuples satisfy some views. Answering queries is a form of answering queries with **incomplete information**.

Global-as-view (Carnot, SIMS, Tsimmis, ...): \mathcal{M} provides direct information about which data satisfy the global schema. Thus, given a query q over \mathcal{G} , it **seems** that we can simply evaluate the query over these data (as if we had a single database at hand). However, when the sources are sound, and there are constraints in the global schema, we still have to deal with **incomplete information**.

For more details on the comparison, see [Ullman, TCS 2000], [Halevy, VLDBJ 2001].

Outline

- **Query answering in LAV**
- Query answering in GAV
- Beyond FOL
- Conclusions

View-based query processing in LAV

Two approaches:

- View-based query rewriting: query processing is divided in two steps, where the first one re-expresses the query in terms of a **given query language** over the alphabet of \mathcal{S} , and the second one evaluates the rewriting over the source database \mathcal{C} .
- **View-based query answering**: no limitation is posed on how queries are processed, and the only goal is to exploit all possible information, in particular the source database, to compute the certain answers to the query.

Query answering in LAV: some results

- Conjunctive queries using conjunctive views [Levy&al. PODS'95]
- Recursive queries (datalog programs) using conjunctive views [Duschka&Genesereth PODS'97]
- Assumptions on view extensions (sound, complete, exact) [Abiteboul&Duschka PODS'98] [Grahne&Mendelzon ICDT'99]
- With integrity constraints in the global schema [Calvanese&al. AAAI'00, Duschka'97, Gryz'98]
- **Variants of Regular Path Queries** [Calvanese&al. ICDE'00, PODS'00] [Deutsch&Tannen DBPL'01], [Calvanese&al. DBPL'01]

How incomplete information shows up

$\phi_{\mathcal{G}}$ conjunctive query:

$$\forall \vec{x} (s(\vec{x}) \rightarrow \exists \vec{y} g_1(\vec{x}, \vec{y}) \wedge \cdots \wedge g_n(\vec{x}, \vec{y}))$$

$\phi_{\mathcal{G}}$ union of conjunctive queries:

$$\forall \vec{x} (s(\vec{x}) \rightarrow \exists \vec{y} \alpha_1(\vec{x}, \vec{y}) \vee \cdots \vee \alpha_n(\vec{x}, \vec{y}))$$

Data complexity for different query languages (AD'98)

Sound Views	CQ	CQ \neq	PQ	datalog	FOL
CQ	<i>PTIME</i>	<i>coNP</i>	<i>PTIME</i>	<i>PTIME</i>	<i>undec.</i>
CQ \neq	<i>PTIME</i>	<i>coNP</i>	<i>PTIME</i>	<i>PTIME</i>	<i>undec.</i>
PQ	<i>coNP</i>	<i>coNP</i>	<i>coNP</i>	<i>coNP</i>	<i>undec.</i>
datalog	<i>coNP</i>	<i>undec.</i>	<i>coNP</i>	<i>undec.</i>	<i>undec.</i>
FOL	<i>undec.</i>	<i>undec.</i>	<i>undec.</i>	<i>undec.</i>	<i>undec.</i>
Exact Views	CQ	CQ \neq	PQ	datalog	FOL
CQ	<i>coNP</i>	<i>coNP</i>	<i>coNP</i>	<i>coNP</i>	<i>undec.</i>
CQ \neq	<i>coNP</i>	<i>coNP</i>	<i>coNP</i>	<i>coNP</i>	<i>undec.</i>
PQ	<i>coNP</i>	<i>coNP</i>	<i>coNP</i>	<i>coNP</i>	<i>undec.</i>
datalog	<i>undec.</i>	<i>undec.</i>	<i>undec.</i>	<i>undec.</i>	<i>undec.</i>
FOL	<i>undec.</i>	<i>undec.</i>	<i>undec.</i>	<i>undec.</i>	<i>undec.</i>

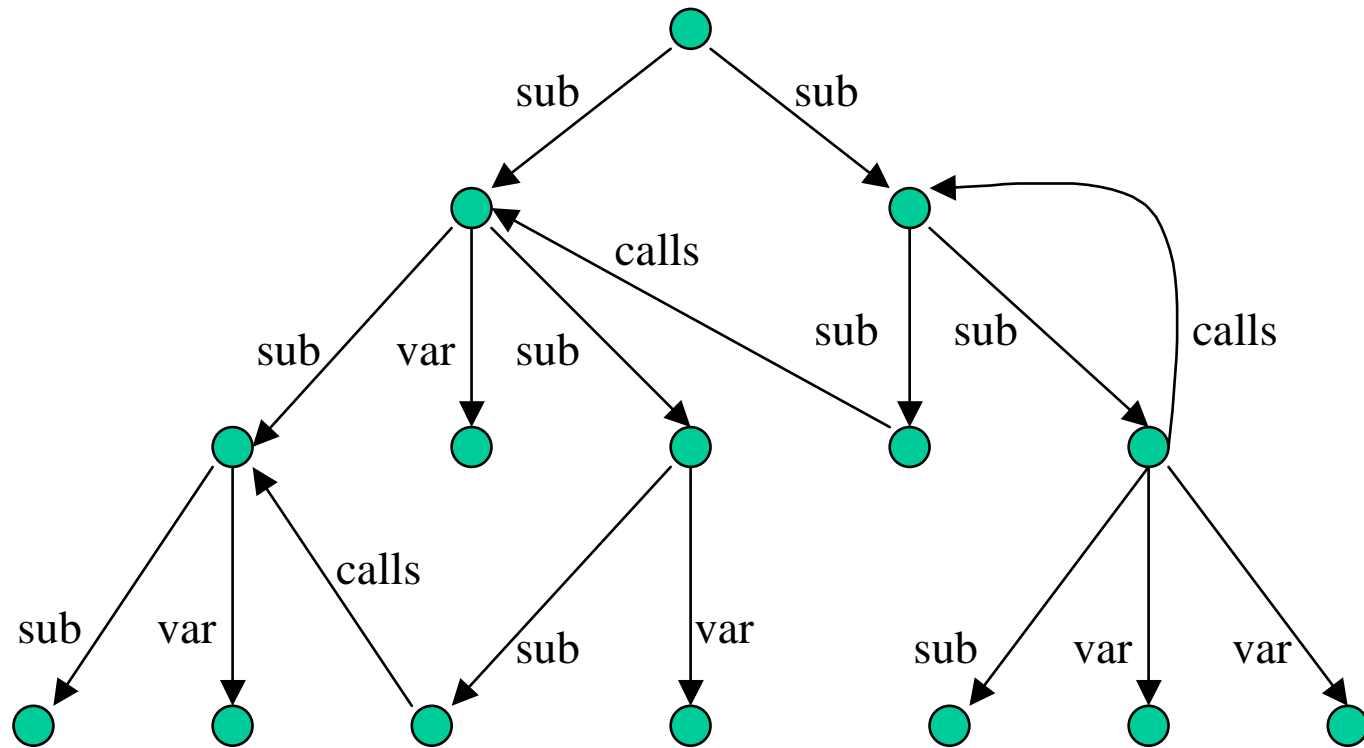
Query answering in LAV

We deal with the problem of answering queries to data integration systems of the form $\langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, where

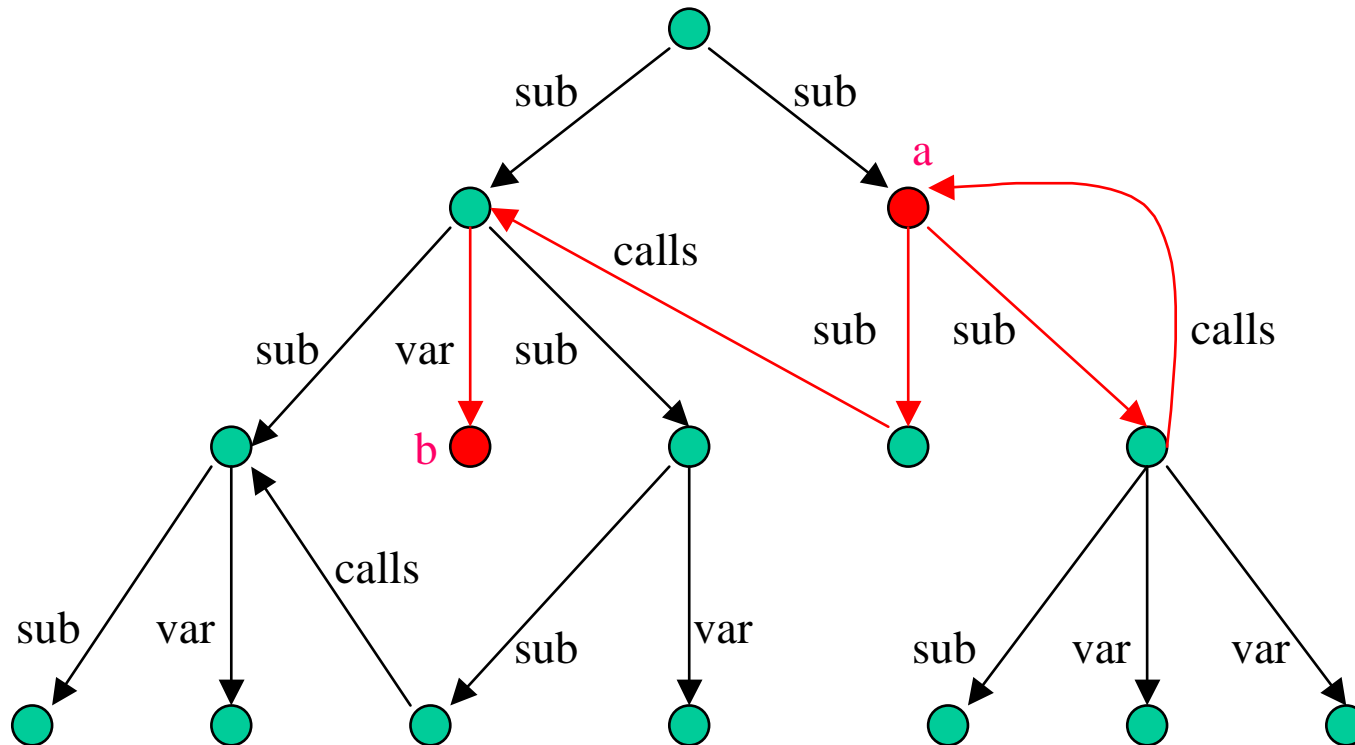
- \mathcal{G} fixes the labels of a semi-structured database
- the sources in \mathcal{S} are relational
- the mapping \mathcal{M} is of type LAV
- queries are typical of semi-structured data (variants of regular path queries)

*Joint work with Diego Calvanese, Giuseppe De Giacomo,
and Moshe Y. Vardi*

Global database

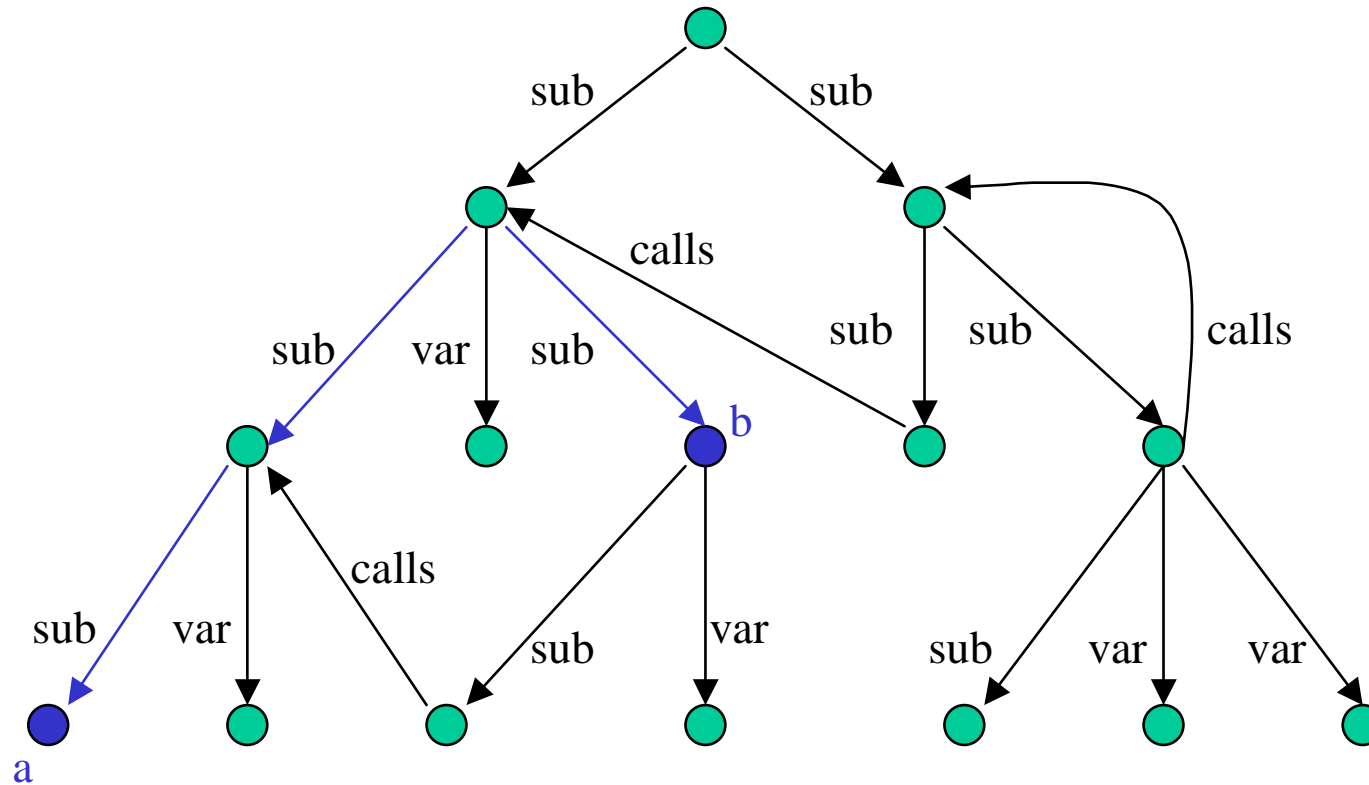


Global databases and queries



Regular Path Query (RPQ): $(sub)^* \cdot (sub \cdot (calls \cup sub))^* \cdot var$

Global databases and queries



2RPQ: $(sub^-)^* \cdot (var \cup sub)$

Query answering: Technique

Given $\mathcal{I} = \langle \Sigma, \mathcal{S}, \mathcal{M} \rangle$ (where \mathcal{M} maps each source to a 2RPQ), and source database \mathcal{C} , we want to determine whether $\vec{t} \in Q^{\mathcal{I}, \mathcal{C}}$, where Q is again a 2RPQ.

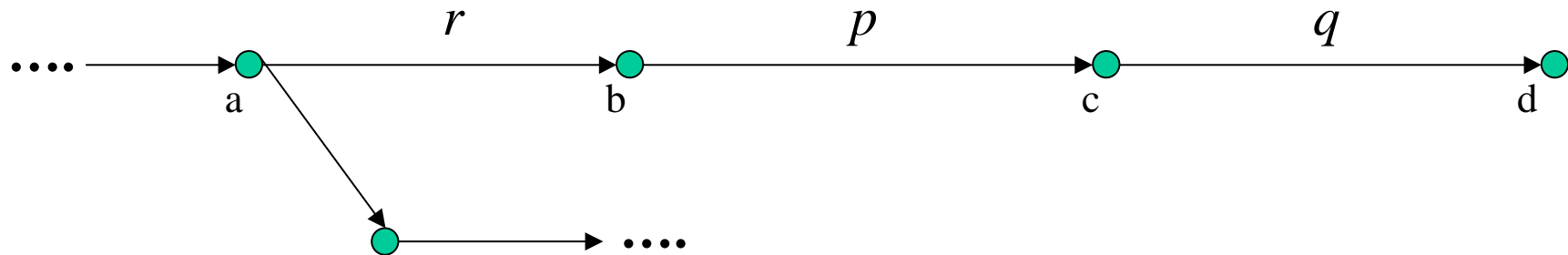
- We search for a **counterexample** to $\vec{t} \in Q^{\mathcal{I}, \mathcal{C}}$, i.e., a **database \mathcal{B}** legal for \mathcal{I} wrt \mathcal{C} such that $\vec{t} \notin Q^{\mathcal{B}}$
- **Crucial point:** it is sufficient to restrict our attention to **canonical** databases, i.e., databases \mathcal{B} that can be represented by a word $w_{\mathcal{B}}$

$$\$ d_1 w_1 d_2 \$ d_3 w_2 d_4 \$ \cdots \$ d_{2m-1} w_m d_{2m} \$$$

where d_1, \dots, d_{2m} are constants in \mathcal{C} , $w_i \in \Sigma^+$, and $\$$ acts as a separator

\Rightarrow *Use word-automata theoretic techniques!*

Finite state automata and RPQs



$$Q = r \cdot (p \cup q) \cdot q \cdot q^*$$

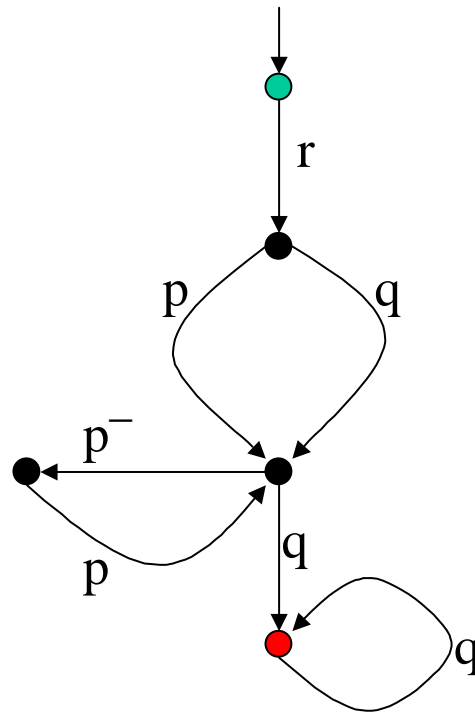
Automaton for Q $\left\{ \begin{array}{l} s_1 \in \delta(s_0, r), s_2 \in \delta(s_1, p), s_2 \in \delta(s_1, q), \\ s_3 \in \delta(s_2, q), s_3 \in \delta(s_3, q) \end{array} \right.$

The computation for RPQs is captured by finite state automata

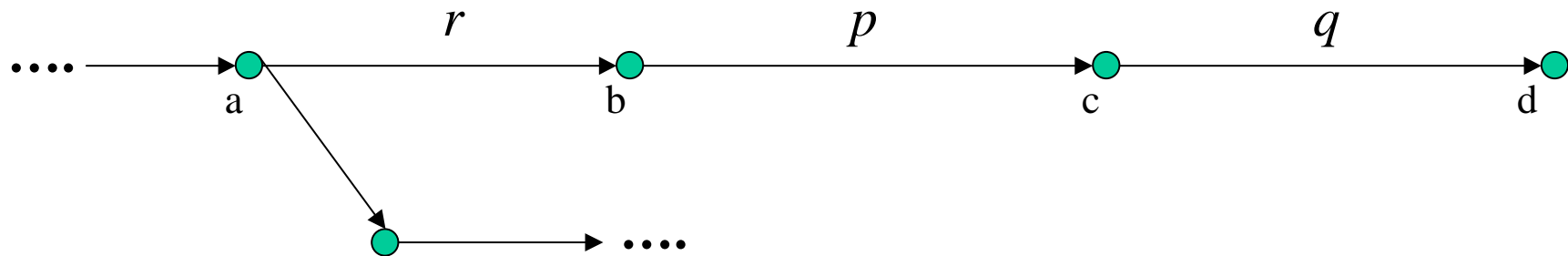
2way Regular Path Queries

2way Regular Path Queries (2RPQ) are expressed by means of finite-state automata over $\Sigma' \cup \{p^- \mid p \in \Sigma'\}$

$$r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$



Finite state automata and 2RPQs



Word:

$r p q$

Query:

$$Q = r \cdot (p \cup q) \cdot p^- \cdot p \cdot q \cdot q^*$$

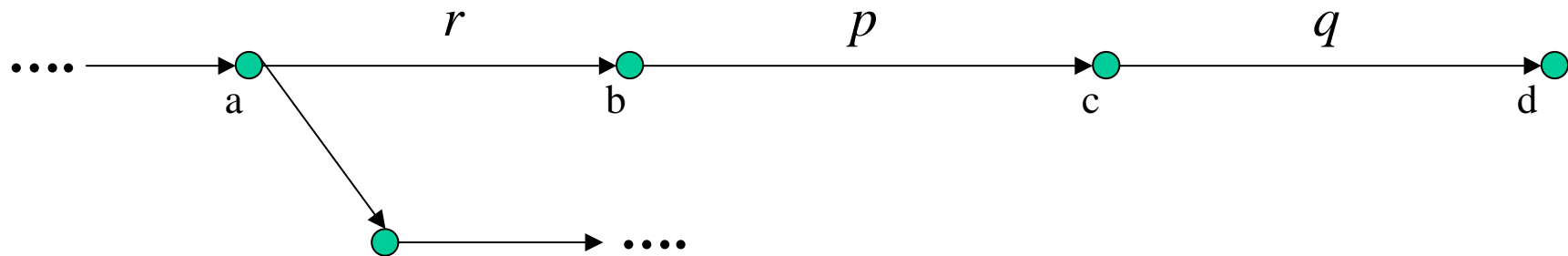
Automaton for Q

$$\left\{ \begin{array}{l} s_1 \in \delta(s_0, r), s_2 \in \delta(s_1, p), s_2 \in \delta(s_1, q), \\ s_3 \in \delta(s_2, p^-), s_4 \in \delta(s_3, p), s_5 \in \delta(s_4, q), s_5 \in \delta(s_5, q) \end{array} \right.$$

State: s_0

Transition: $s_1 \in \delta(s_0, r)$

Finite state automata and 2RPQs



Word:

$r p q$

Query:

$$Q = r \cdot (p \cup q) \cdot p^- \cdot p \cdot q \cdot q^*$$

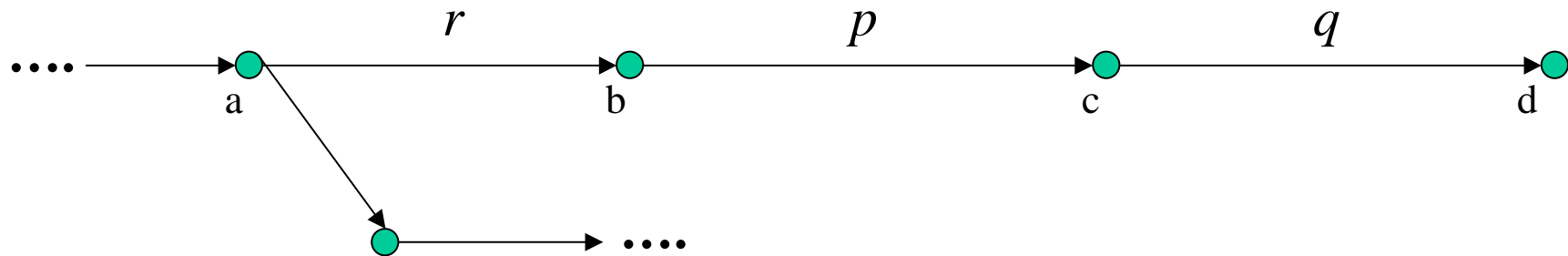
Automaton for Q

$$\left\{ \begin{array}{l} s_1 \in \delta(s_0, r), s_2 \in \delta(s_1, p), s_2 \in \delta(s_1, q), \\ s_3 \in \delta(s_2, p^-), s_4 \in \delta(s_3, p), s_5 \in \delta(s_4, q), s_5 \in \delta(s_5, q) \end{array} \right.$$

State: s_1

Transition: $s_2 \in \delta(s_1, p)$

Finite state automata and 2RPQs



Word: $r p q$

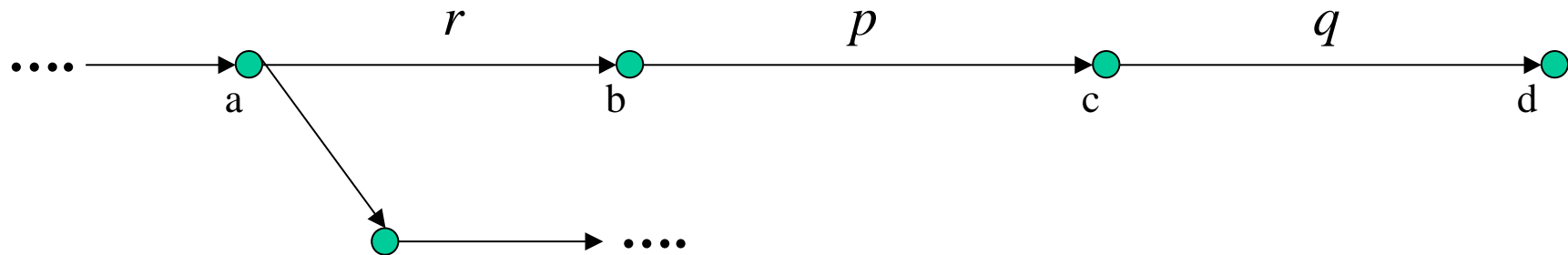
Query: $Q = r \cdot (p \cup q) \cdot p^- \cdot p \cdot q \cdot q^*$

Automaton for Q $\left\{ \begin{array}{l} s_1 \in \delta(s_0, r), s_2 \in \delta(s_1, p), s_2 \in \delta(s_1, q), \\ s_3 \in \delta(s_2, p^-), s_4 \in \delta(s_3, p), s_5 \in \delta(s_4, q), s_5 \in \delta(s_5, q) \end{array} \right.$

State: s_2

Transition: *none*

Finite state automata and 2RPQs



Word: $r p q$

Query: $Q = r \cdot (p \cup q) \cdot p^- \cdot p \cdot q \cdot q^*$

State: s_2

Transition: *none*

(a, d) satisfies query Q , but the path from a to d is not accepted by the 1NFA corresponding to Q : **the computation for 2RPQs is not captured by finite state automata**

2way automata (2NFA)

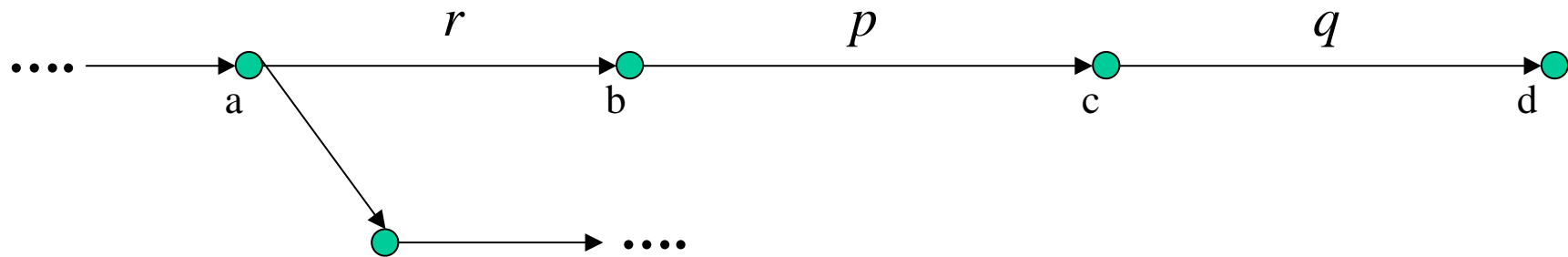
A *2way automaton* $A = (\Gamma, S, S_0, \rho, F)$ consists of an alphabet Γ , a finite set of states S , a set of initial states $S_0 \subseteq S$, a transition function

$$\rho : S \times \Sigma \rightarrow 2^S \times \{-1,0,1\}$$

and a set of accepting states $F \subseteq S$.

Given a 2way automaton A with n states, one can construct a one-way automaton B_1 with $O(2^{n \log n})$ states such that $L(B_1) = L(A)$, and a one-way automaton B_2 with $O(2^n)$ states such that $L(B_2) = \Gamma^* - L(A)$.

2way automata and 2RPQs

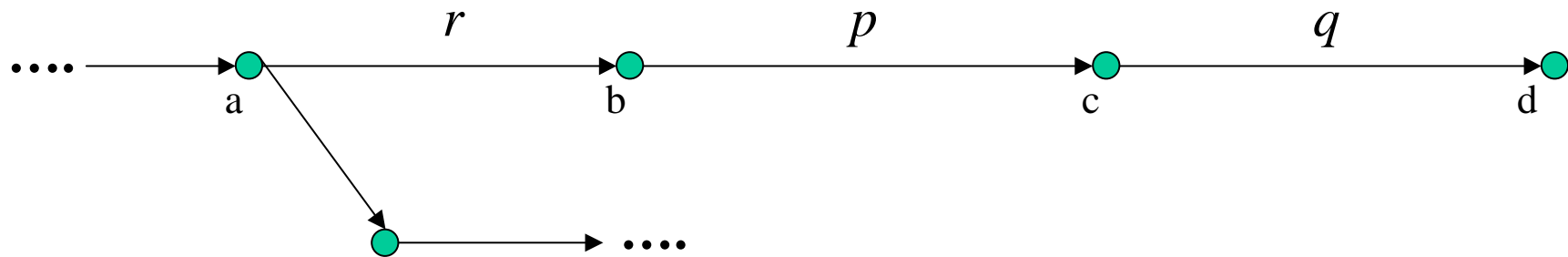


$$Q = r \cdot (p \cup q) \cdot p^{-} \cdot p \cdot q \cdot q^*$$

$$\text{Automaton for } Q \left\{ \begin{array}{l} s_1 \in \delta(s_0, r), \quad s_2 \in \delta(s_1, p), \quad s_2 \in \delta(s_1, q), \\ \mathbf{s_3} \in \delta(\mathbf{s_2}, \mathbf{p}^{-}), \quad s_4 \in \delta(s_3, p), \quad s_5 \in \delta(s_4, q), \quad s_5 \in \delta(s_5, q) \end{array} \right.$$

$$\text{2way automaton} \left\{ \begin{array}{l} (s_1, 1) \in \delta_A(s_0, r), \quad (s_2, 1) \in \delta_A(s_1, p), \quad (s_2, 1) \in \delta_A(s_1, q), \\ (\mathbf{s_2}^{\leftarrow}, -\mathbf{1}) \in \delta_{\mathbf{A}}(\mathbf{s_2}, \mathbf{q}), \quad (\mathbf{s_3}, \mathbf{0}) \in \delta_{\mathbf{A}}(\mathbf{s_2}^{\leftarrow}, \mathbf{p}), \\ (s_4, 1) \in \delta_A(s_3, p), \quad (s_5, 1) \in \delta_A(s_4, q), \quad (s_f, 1) \in \delta_A(s_5, \$) \end{array} \right.$$

2NFA and 2RPQs



Word:

$r \ p \ q \ \$$

Query:

$$Q = r \cdot (p \cup q) \cdot p^{-} \cdot p \cdot q \cdot q^{*}$$

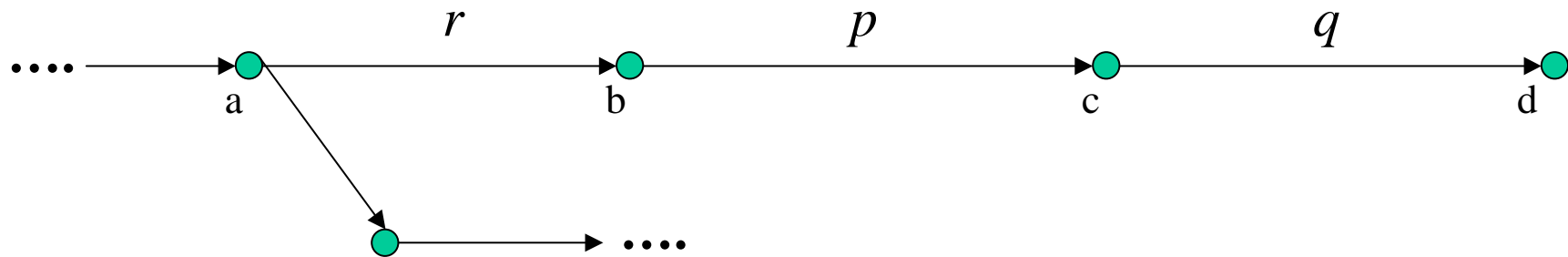
Automaton for Q

$$\left\{ \begin{array}{l} (s_1, 1) \in \delta_A(s_0, r), \quad (s_2, 1) \in \delta_A(s_1, p), \\ (s_2^{\leftarrow}, -1) \in \delta_A(s_2, q), \quad (s_3, 0) \in \delta_A(s_2^{\leftarrow}, p), \\ (s_4, 1) \in \delta_A(s_3, p), \quad (s_5, 1) \in \delta_A(s_4, q), \quad (s_f, 1) \in \delta_A(s_5, \$) \end{array} \right.$$

State: s_0

Transition: $(s_1, 1) \in \delta_A(s_0, r)$

2NFA and 2RPQs



Word:

$r \ p \ q \ \$$

Query:

$$Q = r \cdot (p \cup q) \cdot p^{-} \cdot p \cdot q \cdot q^*$$

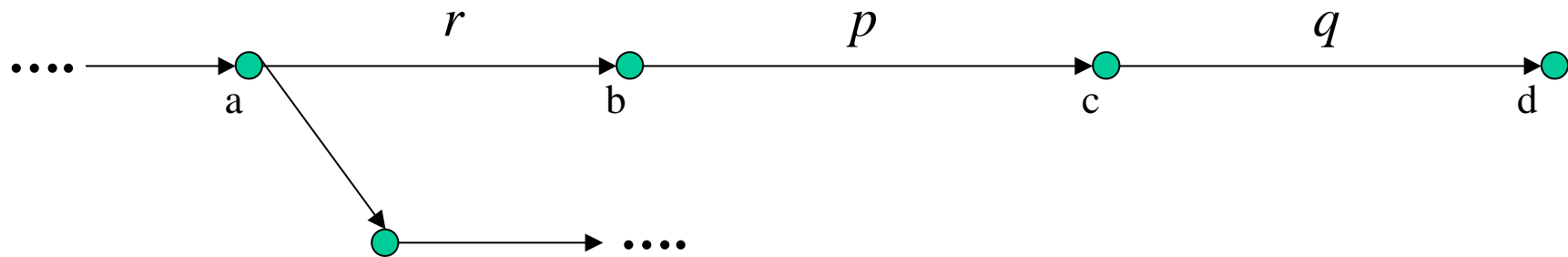
Automaton for Q

$$\left\{ \begin{array}{l} (s_1, 1) \in \delta_A(s_0, r), \quad (s_2, 1) \in \delta_A(s_1, p), \\ (s_2^{\leftarrow}, -1) \in \delta_A(s_2, q), \quad (s_3, 0) \in \delta_A(s_2^{\leftarrow}, p), \\ (s_4, 1) \in \delta_A(s_3, p), \quad (s_5, 1) \in \delta_A(s_4, q), \quad (s_f, 1) \in \delta_A(s_5, \$) \end{array} \right.$$

State: s_1

Transition: $(s_2, 1) \in \delta_A(s_1, p)$

2NFA and 2RPQs



Word:

$r \ p \ q \ \$$

Query:

$$Q = r \cdot (p \cup q) \cdot p^{-} \cdot p \cdot q \cdot q^*$$

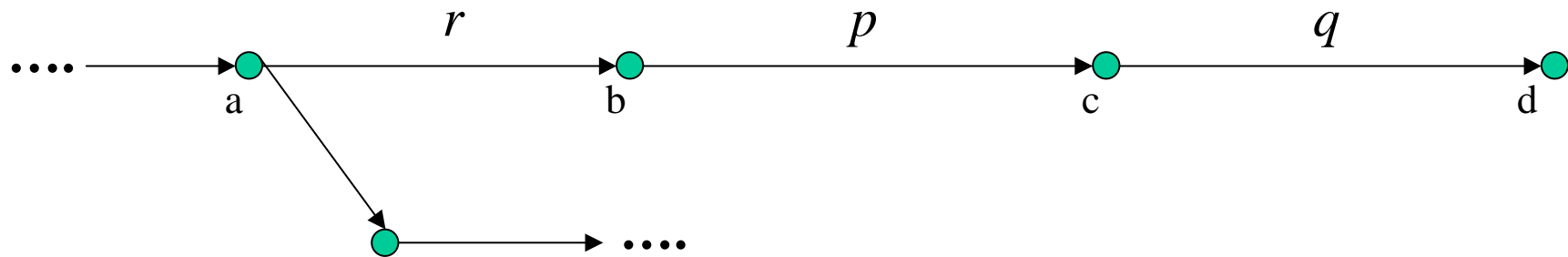
Automaton for Q

$$\left\{ \begin{array}{l} (s_1, 1) \in \delta_A(s_0, r), \quad (s_2, 1) \in \delta_A(s_1, p), \\ (s_2^{\leftarrow}, -1) \in \delta_A(s_2, q), \quad (s_3, 0) \in \delta_A(s_2^{\leftarrow}, p), \\ (s_4, 1) \in \delta_A(s_3, p), \quad (s_5, 1) \in \delta_A(s_4, q), \quad (s_f, 1) \in \delta_A(s_5, \$) \end{array} \right.$$

State: s_2

Transition: $(s_2^{\leftarrow}, -1) \in \delta_A(s_2, q)$

2NFA and 2RPQs



Word:

$r \ p \ q \ \$$

Query:

$$Q = r \cdot (p \cup q) \cdot p^- \cdot p \cdot q \cdot q^*$$

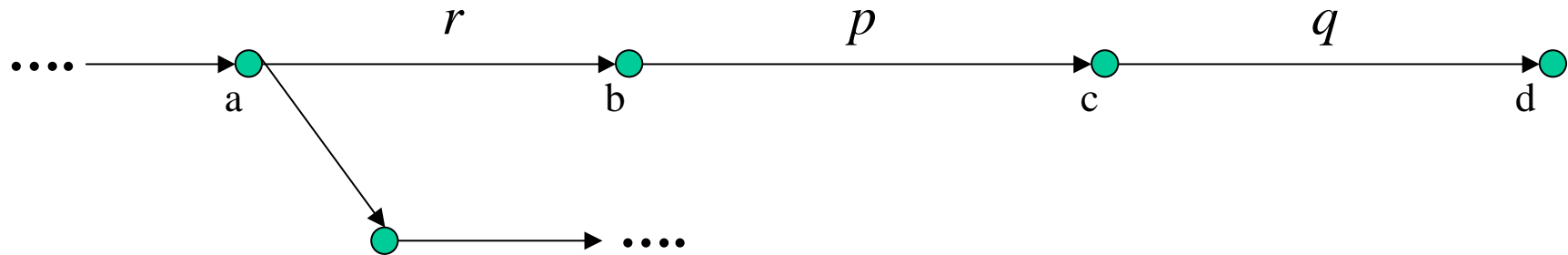
Automaton for Q

$$\left\{ \begin{array}{l} (s_1, 1) \in \delta_A(s_0, r), \quad (s_2, 1) \in \delta_A(s_1, p), \\ (s_2^{\leftarrow}, -1) \in \delta_A(s_2, q), \quad (s_3, 0) \in \delta_A(s_2^{\leftarrow}, p), \\ (s_4, 1) \in \delta_A(s_3, p), \quad (s_5, 1) \in \delta_A(s_4, q), \quad (s_f, 1) \in \delta_A(s_5, \$) \end{array} \right.$$

State: s_2^{\leftarrow}

Transition: $(s_3, 0) \in \delta_A(s_2^{\leftarrow}, p)$

2NFA and 2RPQs



Word:

$r \ p \ q \ \$$

Query:

$$Q = r \cdot (p \cup q) \cdot p^{-} \cdot p \cdot q \cdot q^*$$

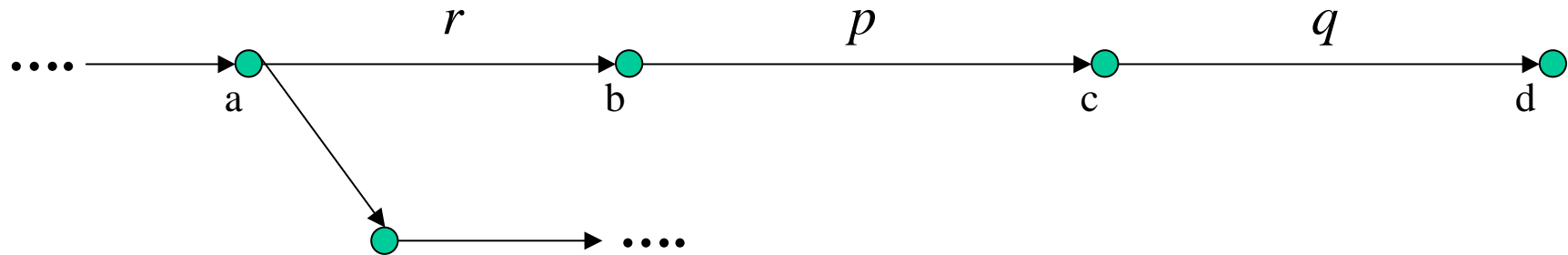
Automaton for Q

$$\left\{ \begin{array}{l} (s_1, 1) \in \delta_A(s_0, r), \quad (s_2, 1) \in \delta_A(s_1, p), \\ (s_2^{\leftarrow}, -1) \in \delta_A(s_2, q), \quad (s_3, 0) \in \delta_A(s_2^{\leftarrow}, p), \\ (s_4, 1) \in \delta_A(s_3, p), \quad (s_5, 1) \in \delta_A(s_4, q), \quad (s_f, 1) \in \delta_A(s_5, \$) \end{array} \right.$$

State: s_3

Transition: $(s_4, 1) \in \delta_A(s_3, p)$

2NFA and 2RPQs



Word:

$r \ p \ q \ \$$

Query:

$$Q = r \cdot (p \cup q) \cdot p^{-} \cdot p \cdot q \cdot q^{*}$$

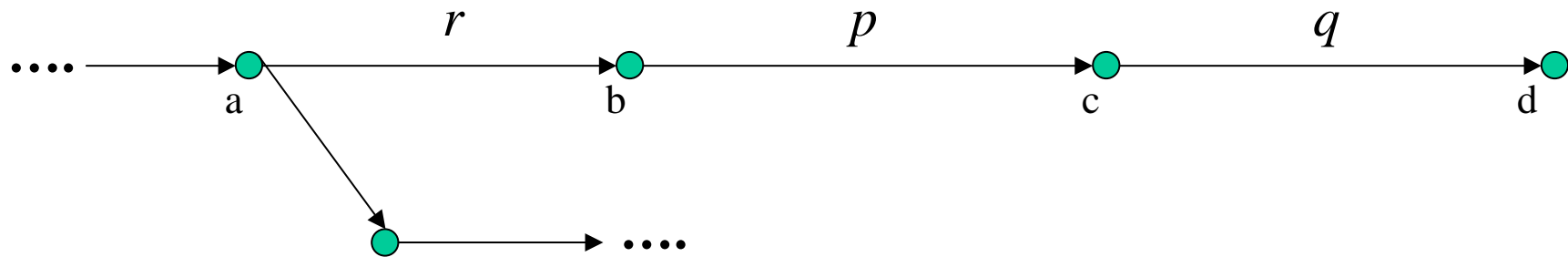
Automaton for Q

$$\left\{ \begin{array}{l} (s_1, 1) \in \delta_A(s_0, r), \quad (s_2, 1) \in \delta_A(s_1, p), \\ (s_2^{\leftarrow}, -1) \in \delta_A(s_2, q), \quad (s_3, 0) \in \delta_A(s_2^{\leftarrow}, p), \\ (s_4, 1) \in \delta_A(s_3, p), \quad (s_5, 1) \in \delta_A(s_4, q), \quad (s_f, 1) \in \delta_A(s_5, \$) \end{array} \right.$$

State: s_4

Transition: $(s_5, 1) \in \delta_A(s_4, q)$

2NFA and 2RPQs



Word:

$r p q \$$

Query:

$$Q = r \cdot (p \cup q) \cdot p^- \cdot p \cdot q \cdot q^*$$

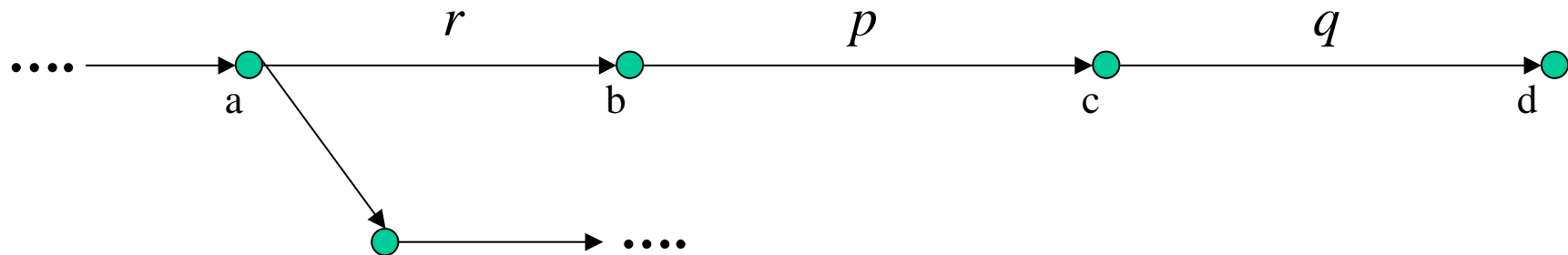
Automaton for Q

$$\left\{ \begin{array}{l} (s_1, 1) \in \delta_A(s_0, r), (s_2, 1) \in \delta_A(s_1, p), \\ (s_2^{\leftarrow}, -1) \in \delta_A(s_2, q), (s_3, 0) \in \delta_A(s_2^{\leftarrow}, p), \\ (s_4, 1) \in \delta_A(s_3, p), (s_5, 1) \in \delta_A(s_4, q), (s_f, 1) \in \delta_A(s_5, \$) \end{array} \right.$$

State: s_5

Transition: $(s_f, 1) \in \delta_A(s_5, \$)$

2NFA and 2RPQs



Word: $r p q \$$

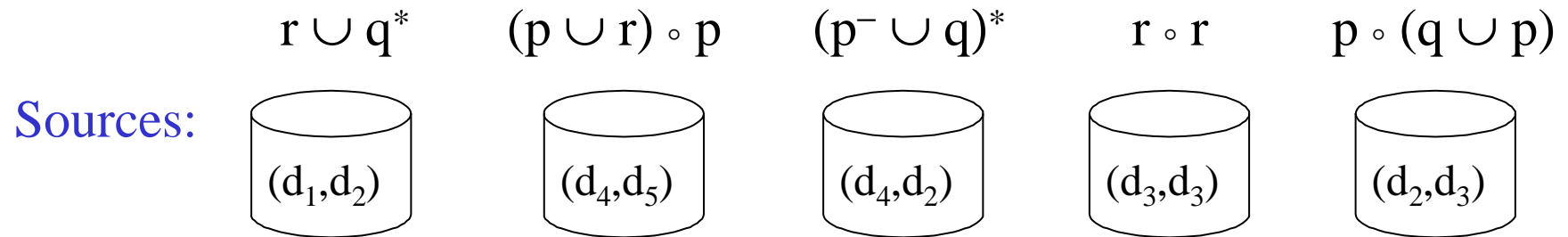
Query: $Q = r \cdot (p \cup q) \cdot p^- \cdot p \cdot q \cdot q^*$

State: s_f

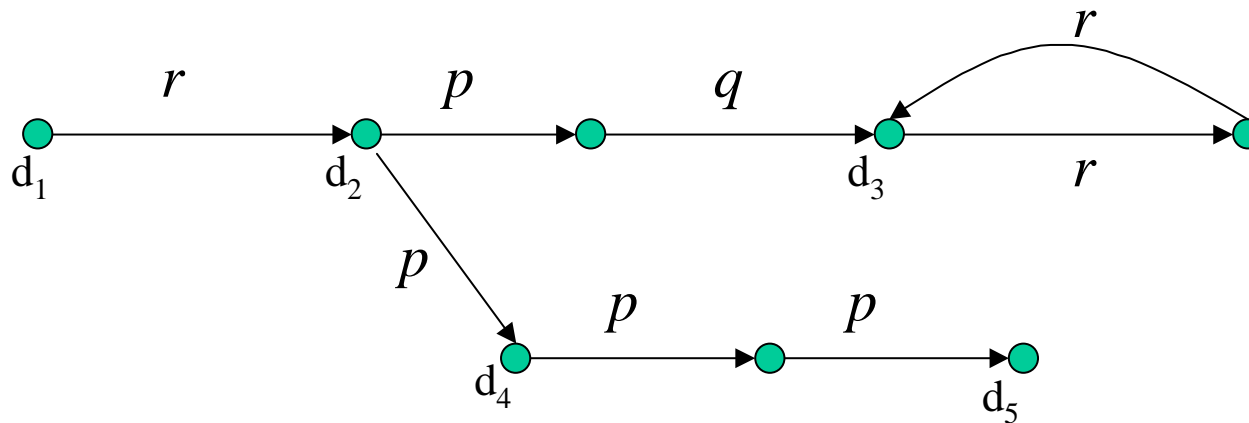
(a, d) satisfies query Q , and the path from a to d is accepted by the 2NFA corresponding to Q : **the computation for 2RPQs is captured by 2way automata**

2NFA and view extensions

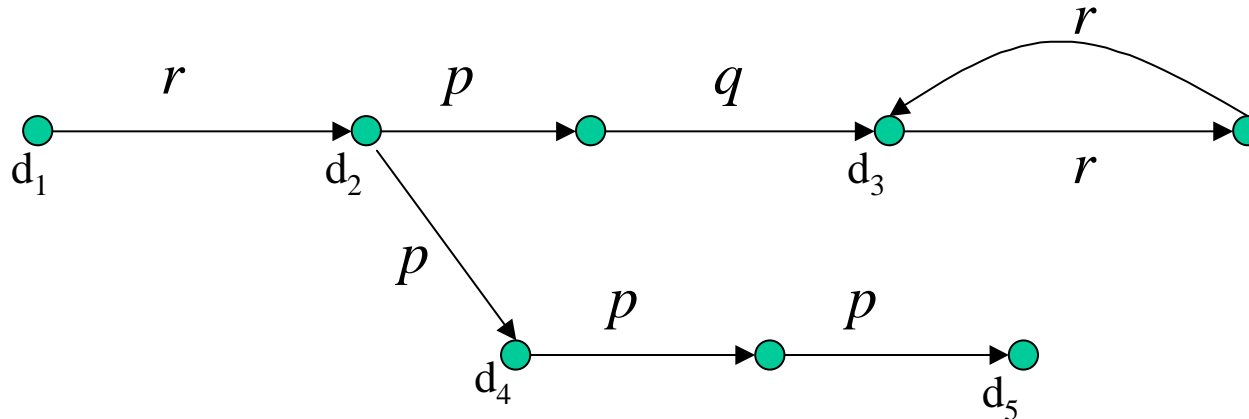
Global schema G : $(r \cup p \cup q \cup r^- \cup p^- \cup q^-)^*$



Database for G :



2NFA and view extensions

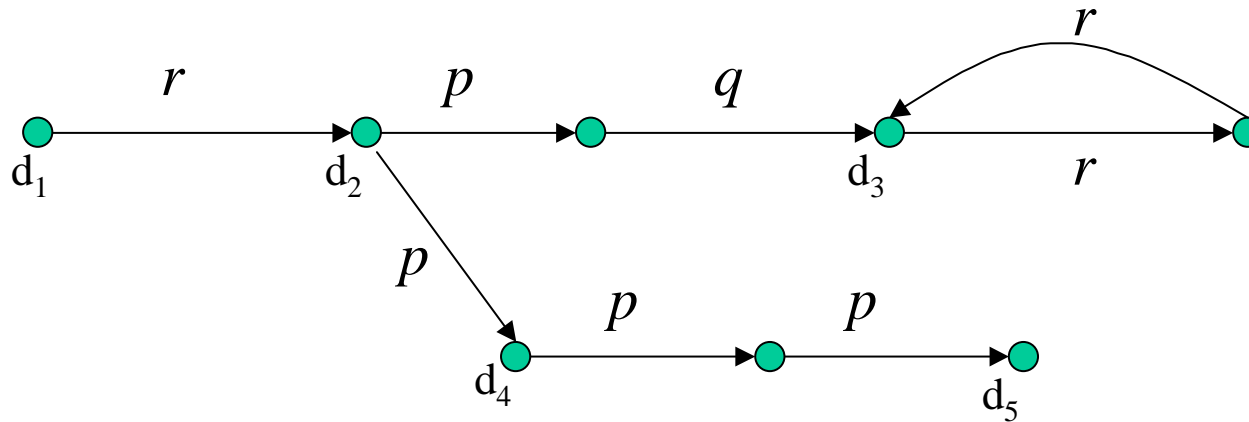


As a word: $\$d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

To verify that (d_1, d_3) satisfies Q in the above database \mathcal{B} , we build $A_{(Q, d_1, d_3)}$, by exploiting not only the ability of 2way automata to move on the word both **forward and backward**, but also the ability to **jump** from one position in the word representing a node to any other position (either preceding or succeeding) representing the same node.

A run of $A_{(Q,d_1,d_3)}$



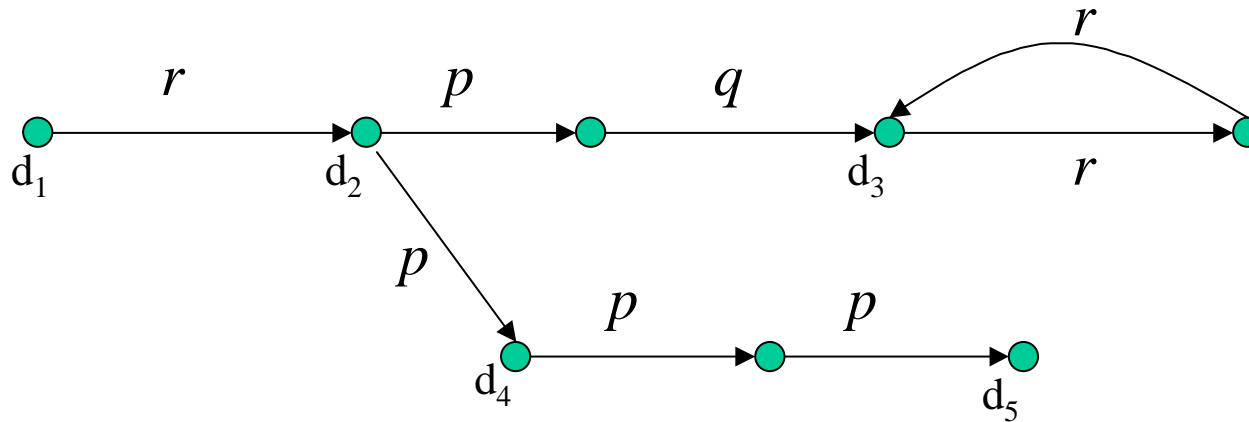
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_0

Transition: $(s_0, 1) \in \delta_A(s_0, \ell)$, for each $\ell \in \Sigma_A$

A run of $A_{(Q,d_1,d_3)}$



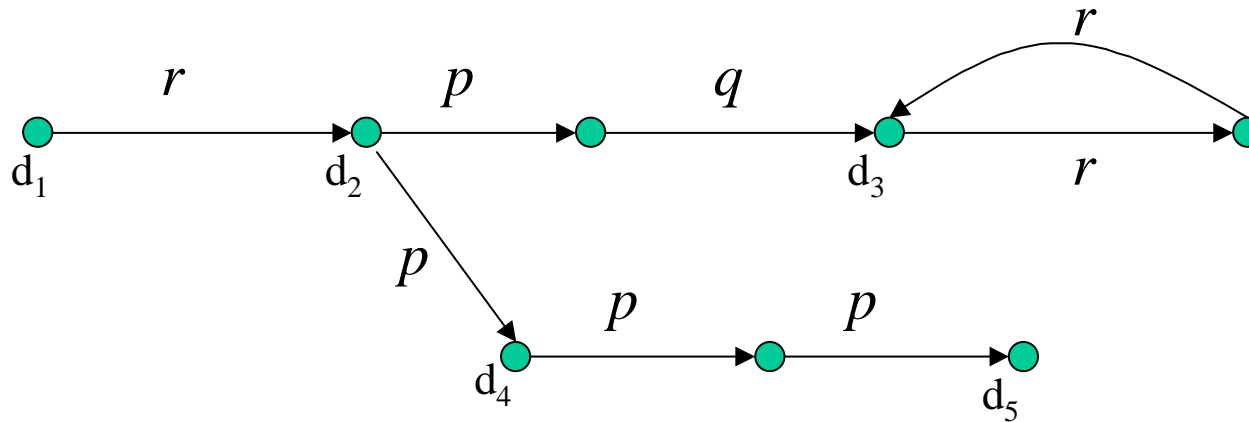
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_0

Transition: $(s_0, 1) \in \delta_A(s_0, \ell)$, for each $\ell \in \Sigma_A$

A run of $A_{(Q,d_1,d_3)}$



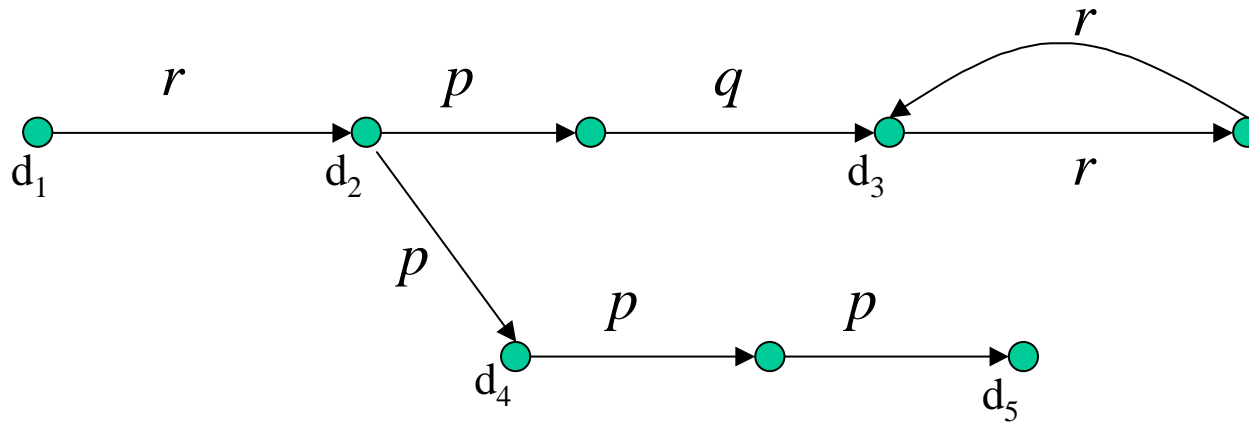
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_0

Transition: $(s_0, 1) \in \delta_A(s_0, \ell)$, for each $\ell \in \Sigma_A$

A run of $A_{(Q,d_1,d_3)}$



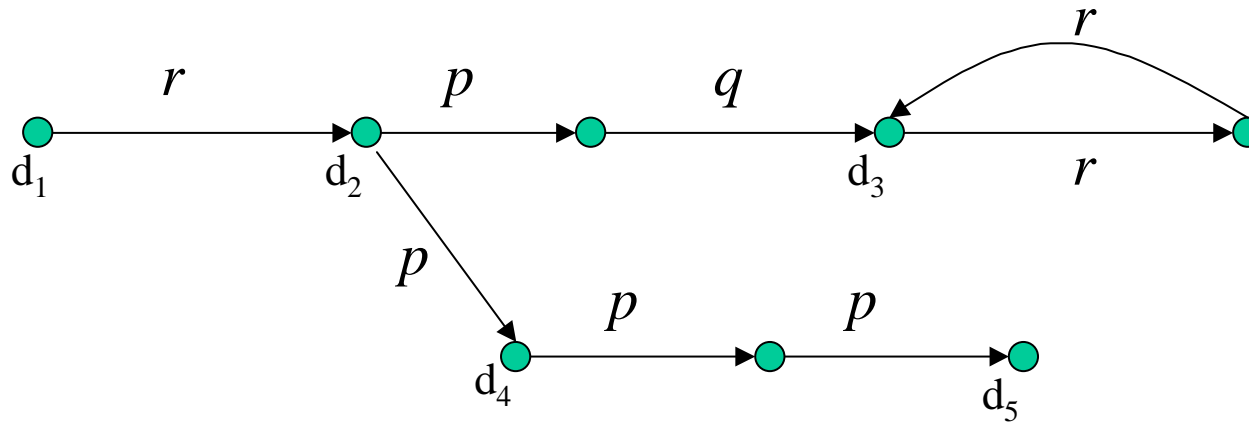
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_0

Transition: $(s_0, 1) \in \delta_A(s_0, \ell)$, for each $\ell \in \Sigma_A$

A run of $A_{(Q,d_1,d_3)}$



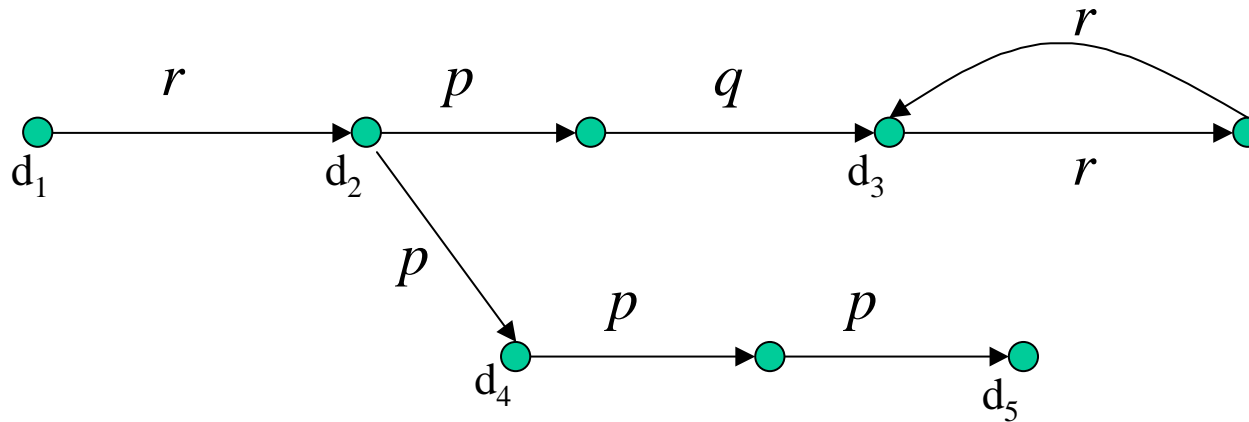
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_0

Transition: $(s_0, 1) \in \delta_A(s_0, \ell)$, for each $\ell \in \Sigma_A$

A run of $A_{(Q,d_1,d_3)}$



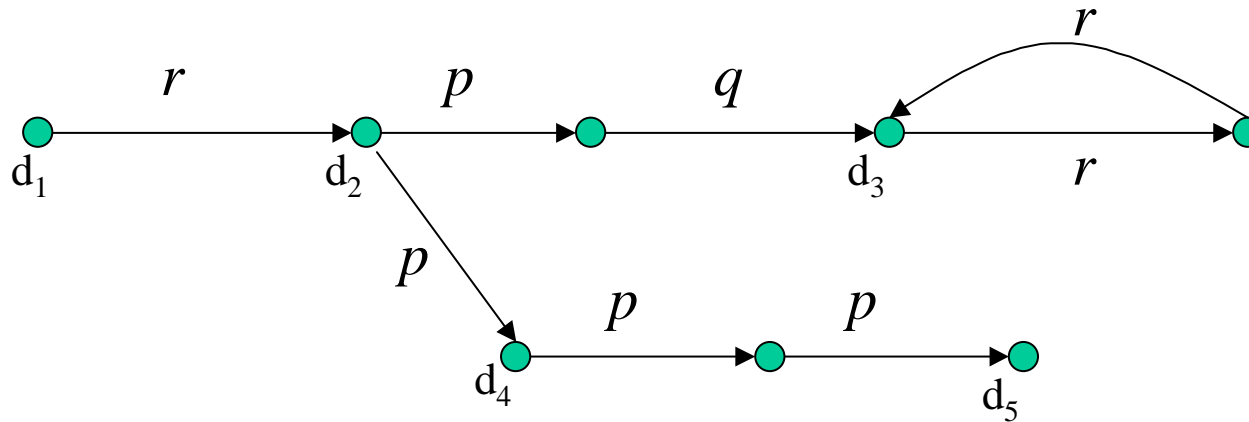
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_0

Transition: $(s_0, 1) \in \delta_A(s_0, \ell)$, for each $\ell \in \Sigma_A$

A run of $A_{(Q,d_1,d_3)}$



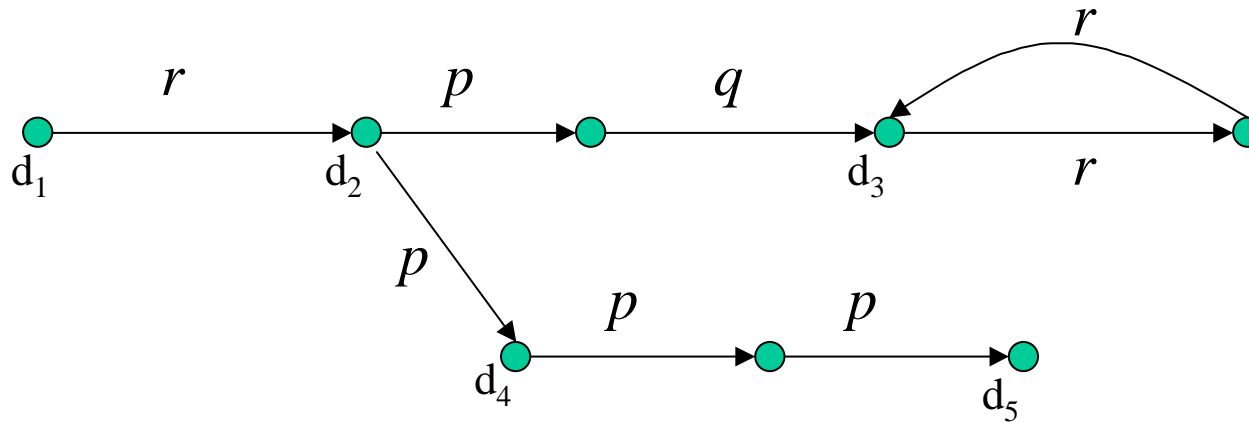
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_0

Transition: $(s_1, 0) \in \delta_A(s_0, d_1)$, s_1 initial state for Q

A run of $A_{(Q,d_1,d_3)}$



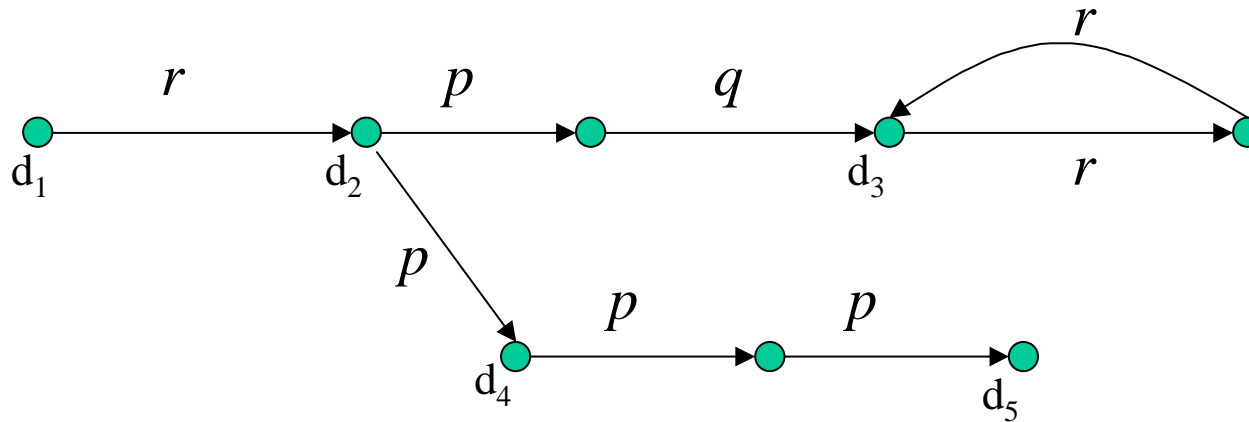
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_1

Transition: $(s_1, 1) \in \delta_A(s_1, d_1)$

A run of $A_{(Q,d_1,d_3)}$



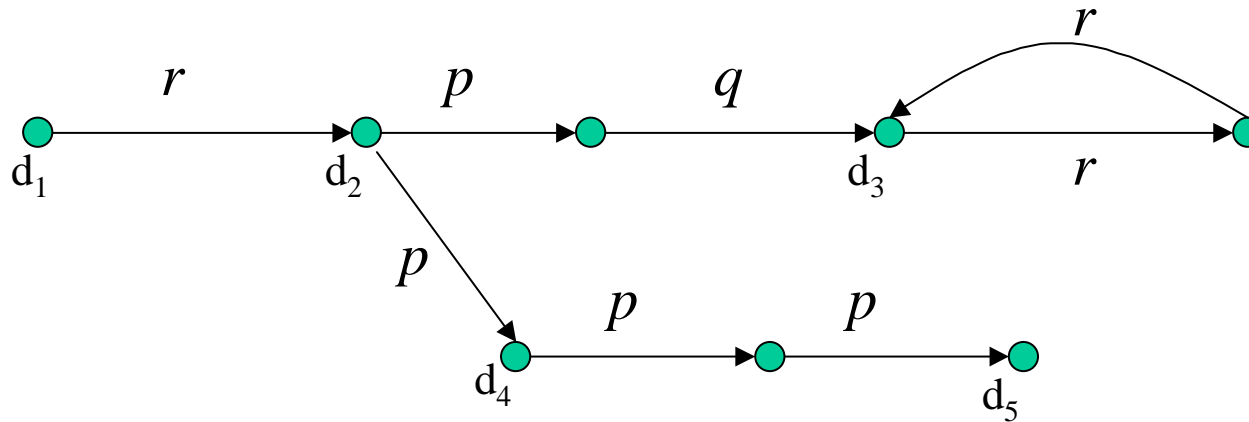
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_1

Transition: $(s_2, 1) \in \delta_A(s_1, r)$, transition coming from Q

A run of $A_{(Q,d_1,d_3)}$



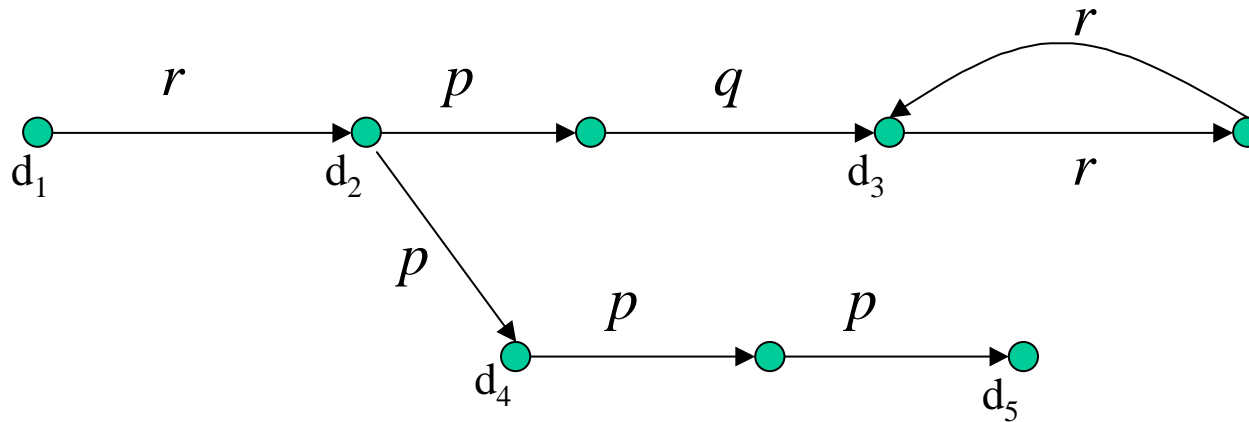
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_2

Transition: $((s_2, d_2), 1) \in \delta_A(s_2, d_2)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



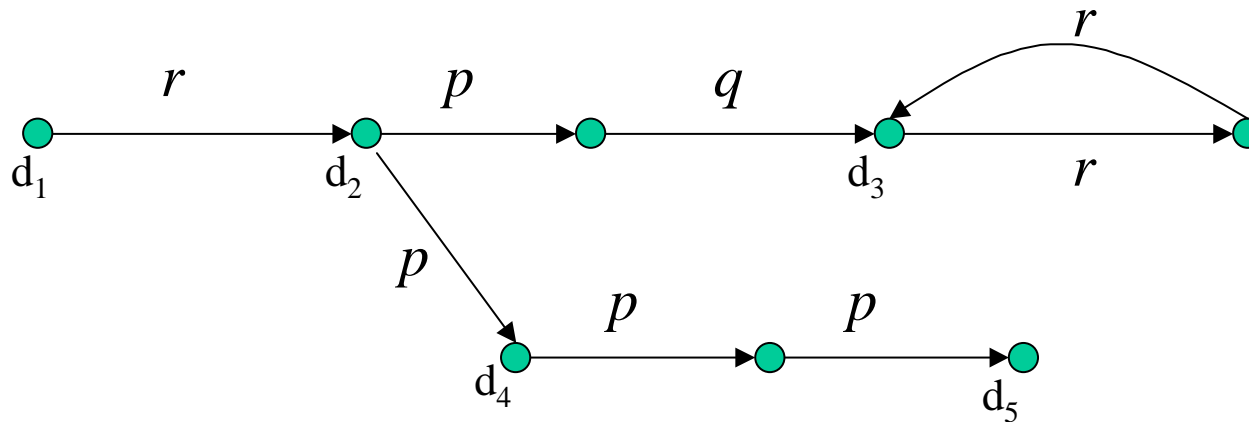
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_2, d_2)

Transition: $((s_2, d_2), 1) \in \delta_A((s_2, d_2), \$)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



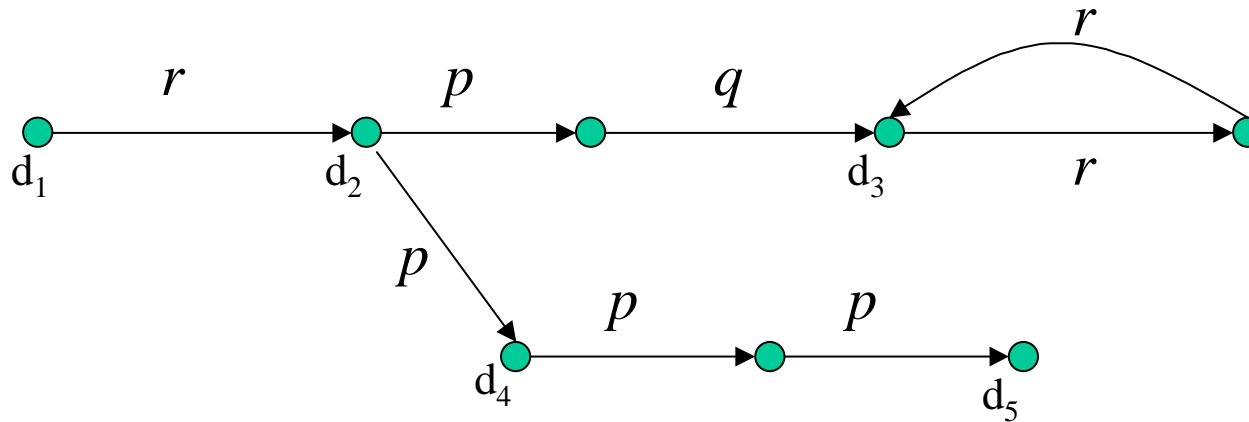
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_2, d_2)

Transition: $((s_2, d_2), 1) \in \delta_A((s_2, d_2), d_4)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



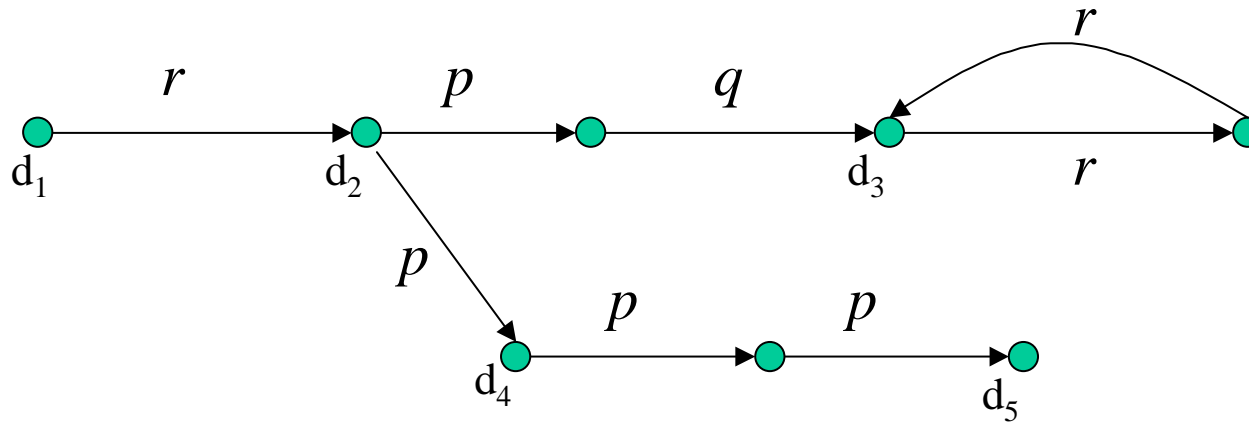
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_2, d_2)

Transition: $((s_2, d_2), 1) \in \delta_A((s_2, d_2), p^-)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



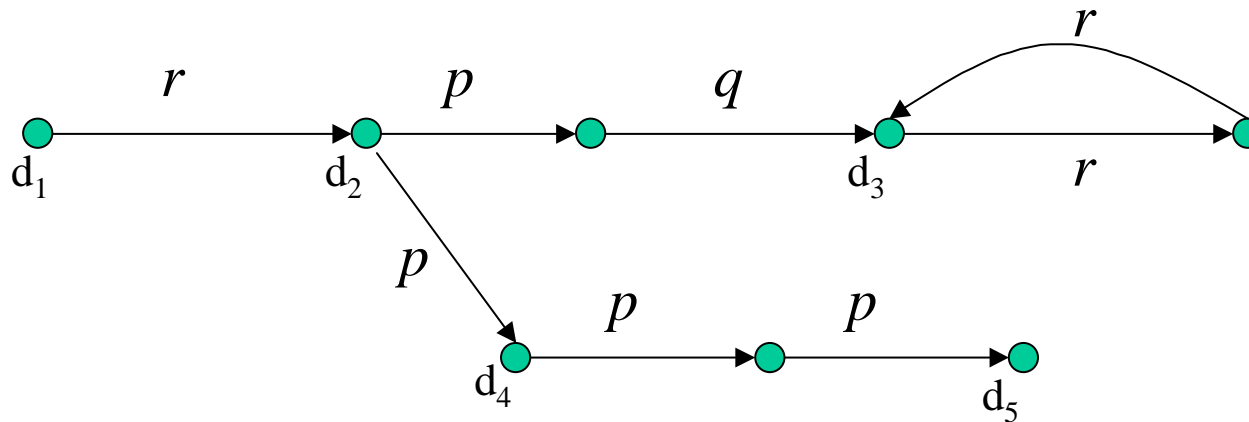
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_2, d_2)

Transition: $(s_2, 0) \in \delta_A((s_2, d_2), d_2)$, exit search mode

A run of $A_{(Q,d_1,d_3)}$



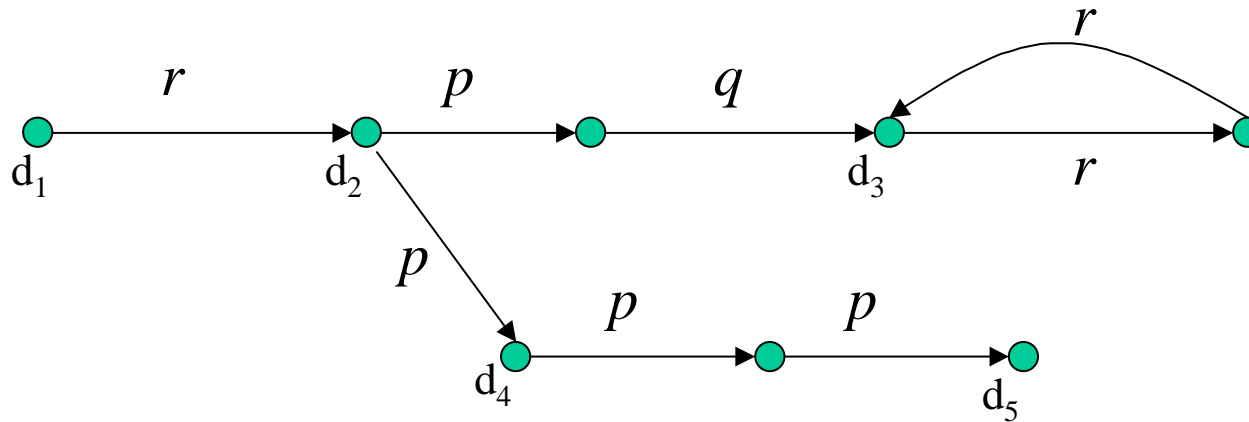
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_2

Transition: $(s_2^{\leftarrow}, -1) \in \delta_A(s_2, d_2)$, backward mode

A run of $A_{(Q,d_1,d_3)}$



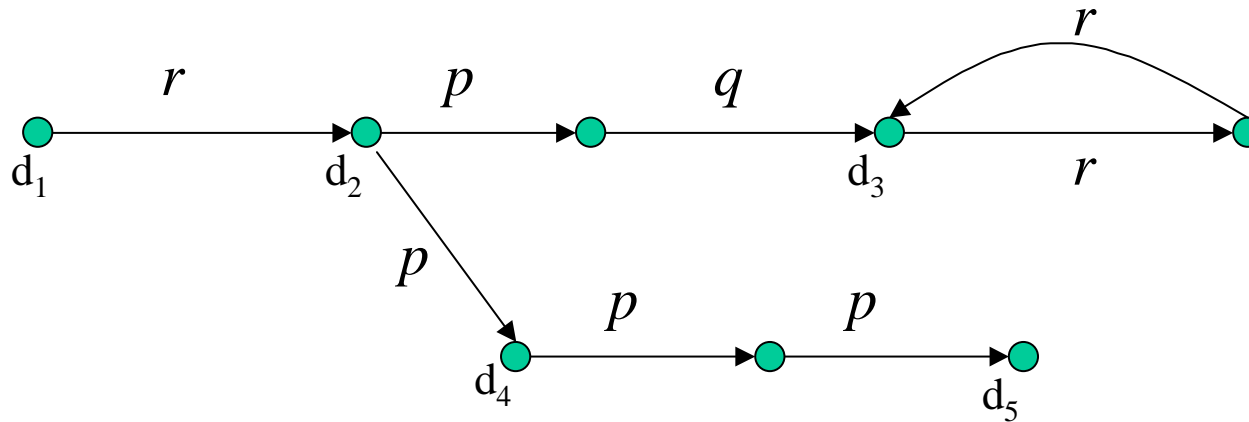
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_2^{\leftarrow}

Transition: $(s_3, 0) \in \delta_A(s_2^{\leftarrow}, p^-)$, transition coming from Q

A run of $A_{(Q,d_1,d_3)}$



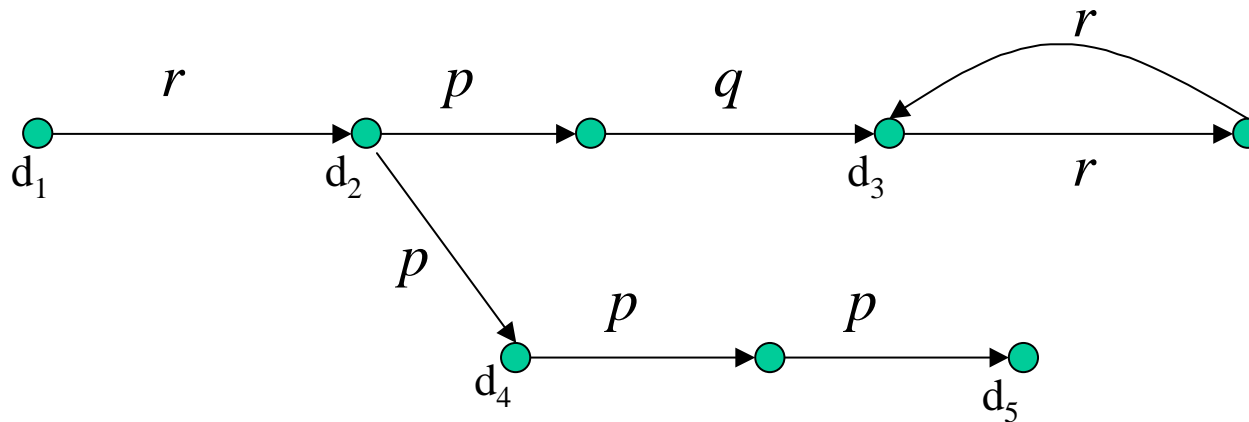
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_3

Transition: $(s_4, 1) \in \delta_A(s_3, p^-)$, transition coming from Q

A run of $A_{(Q,d_1,d_3)}$



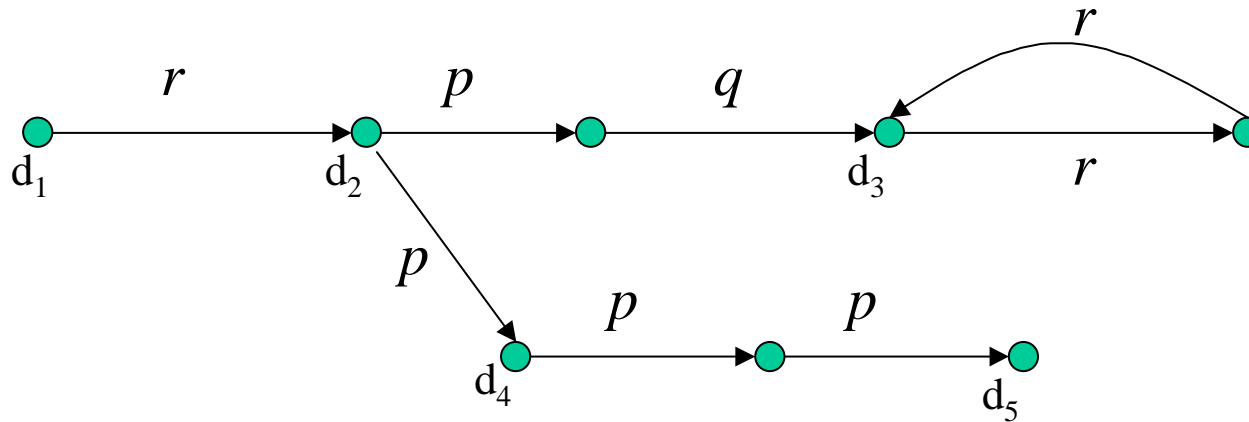
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_4

Transition: $((s_4, d_2), 1) \in \delta_A(s_4, d_2)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



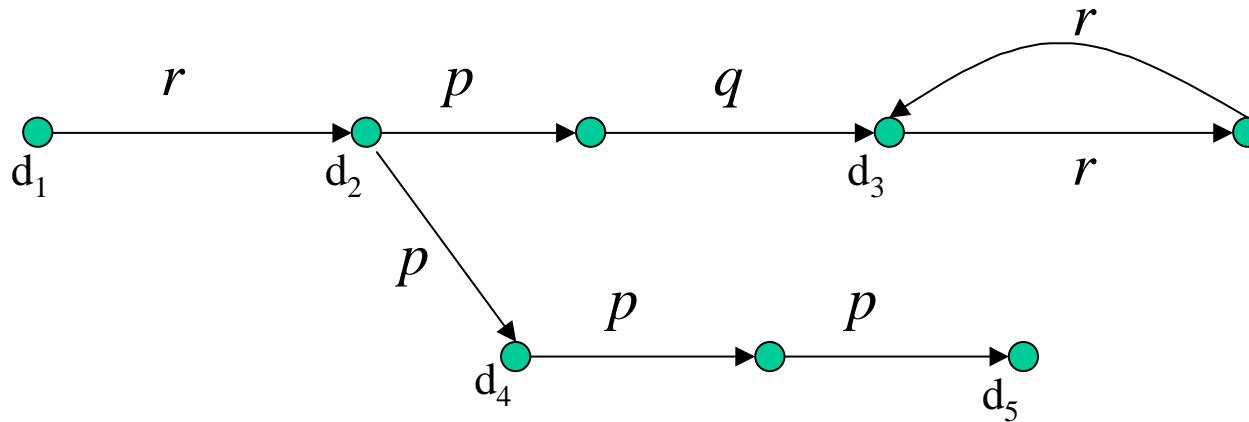
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_4, d_2)

Transition: $((s_4, d_2), 1) \in \delta_A((s_4, d_2), \$)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



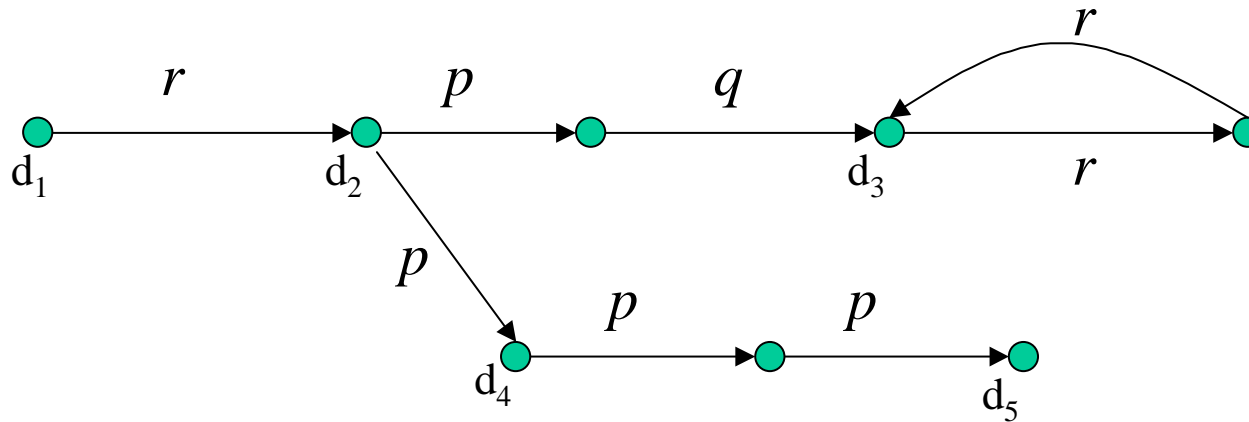
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_4, d_2)

Transition: $((s_4, d_2), 1) \in \delta_A((s_4, d_2), d_3)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



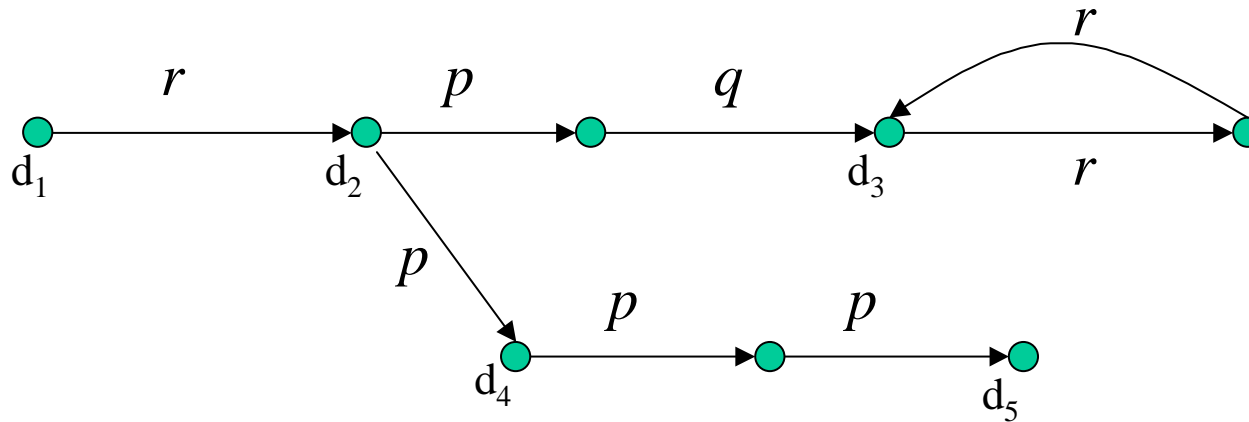
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_4, d_2)

Transition: $((s_4, d_2), 1) \in \delta_A((s_4, d_2), r)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



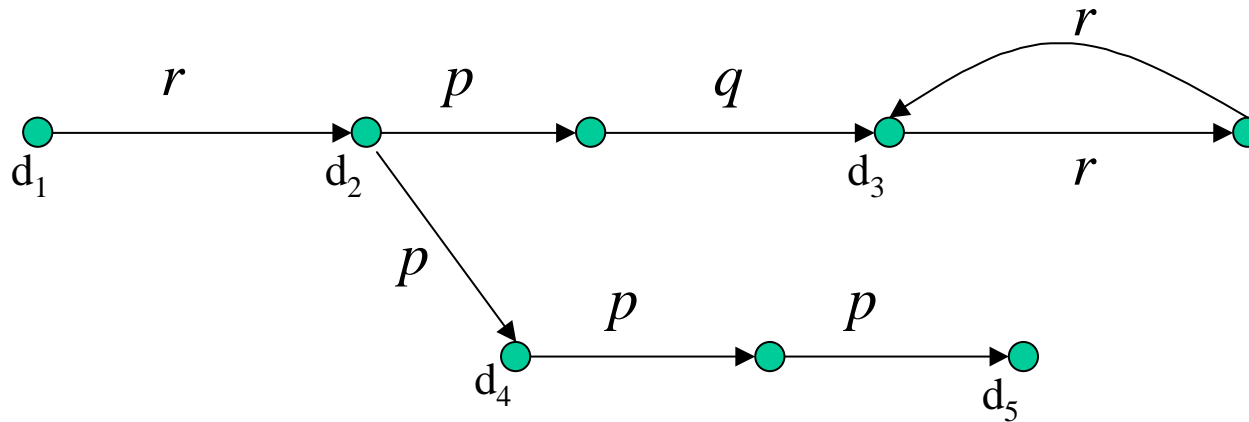
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_4, d_2)

Transition: $((s_4, d_2), 1) \in \delta_A((s_4, d_2), r)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



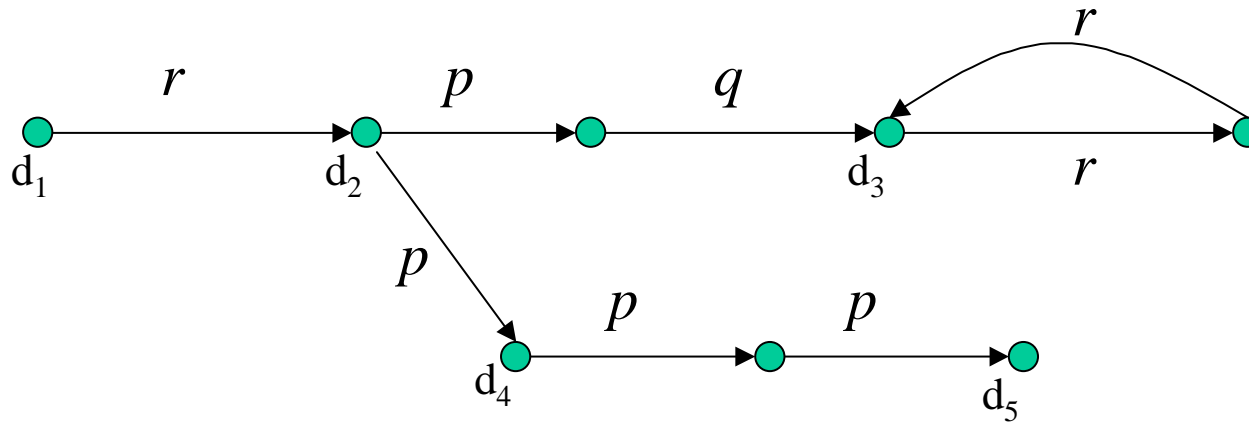
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_4, d_2)

Transition: $((s_4, d_2), 1) \in \delta_A((s_4, d_2), d_3)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



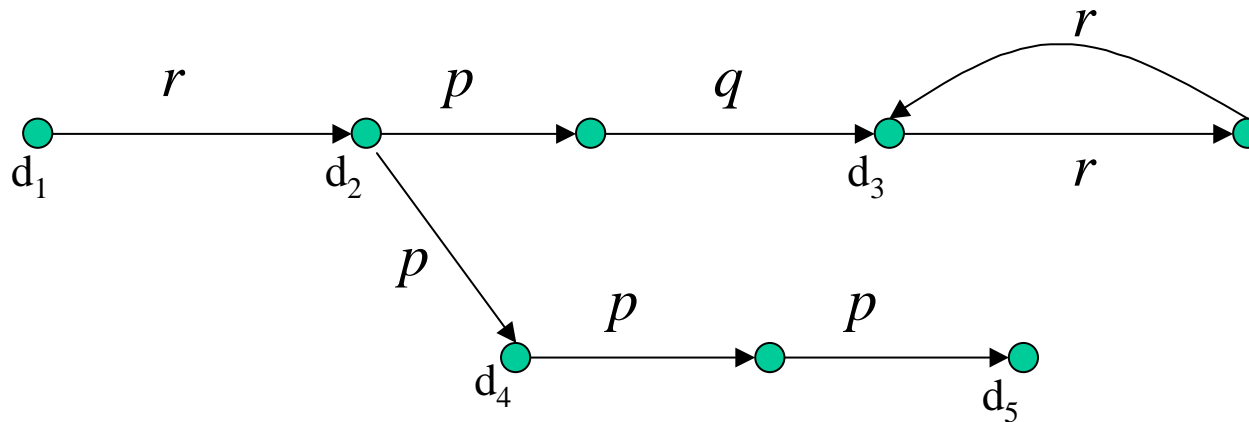
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_4, d_2)

Transition: $((s_4, d_2), 1) \in \delta_A((s_4, d_2), \$)$, search for d_2

A run of $A_{(Q,d_1,d_3)}$



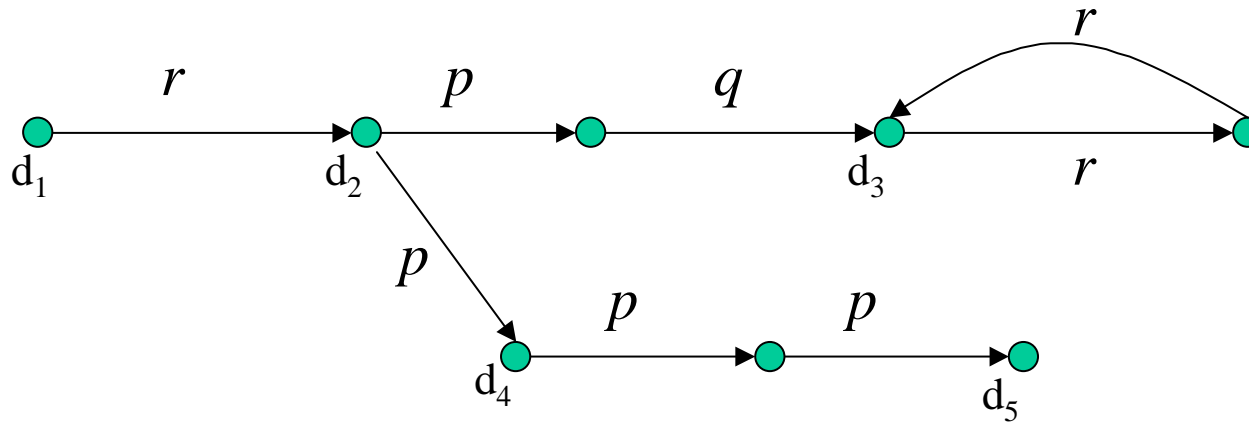
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: (s_4, d_2)

Transition: $(s_4, 0) \in \delta_A((s_4, d_2), d_2)$, exit search mode

A run of $A_{(Q,d_1,d_3)}$



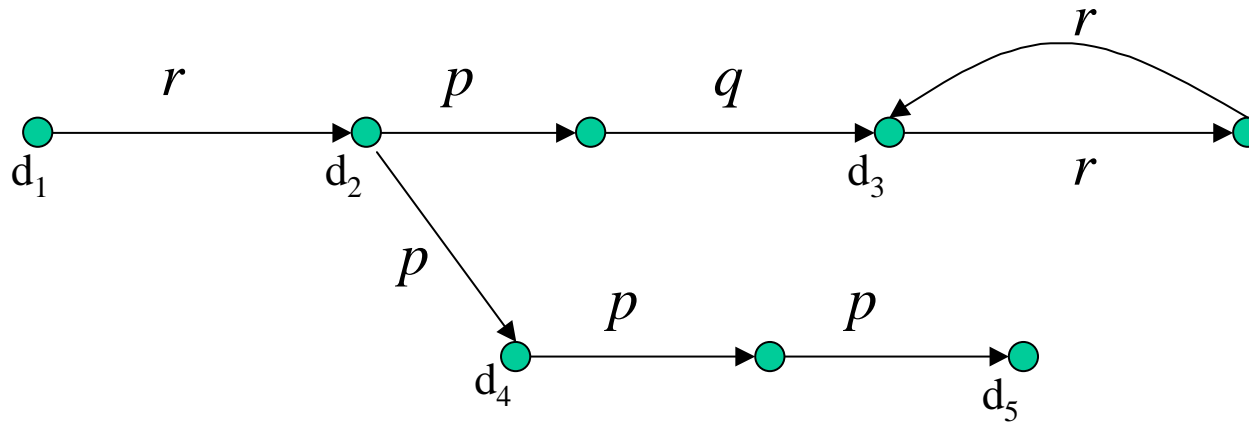
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_4

Transition: $(s_4, 1) \in \delta_A(s_4, d_2)$

A run of $A_{(Q,d_1,d_3)}$



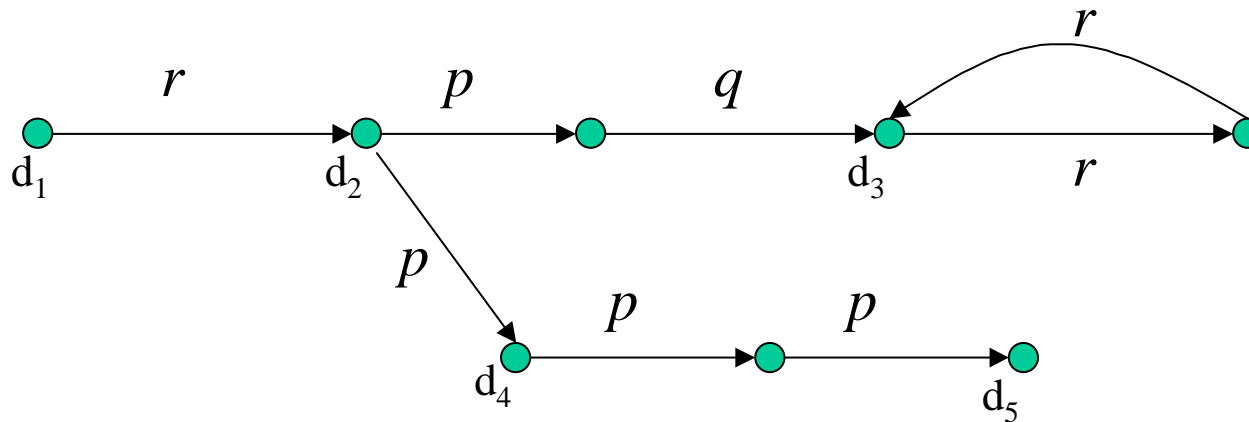
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_4

Transition: $(s_5, 1) \in \delta_A(s_4, p)$, transition coming from Q

A run of $A_{(Q,d_1,d_3)}$



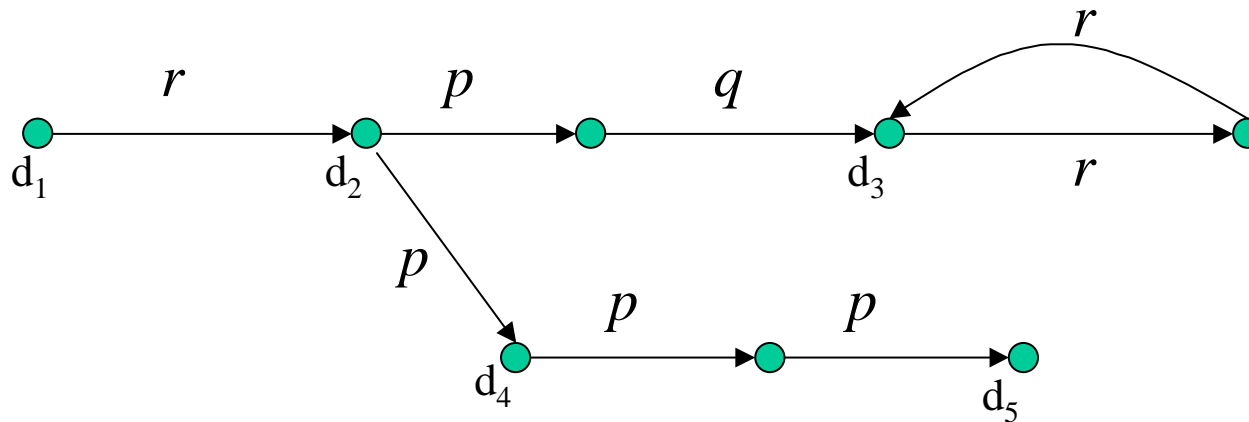
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_5

Transition: $(s_6, 1) \in \delta_A(s_5, q)$, transition coming from Q

A run of $A_{(Q,d_1,d_3)}$



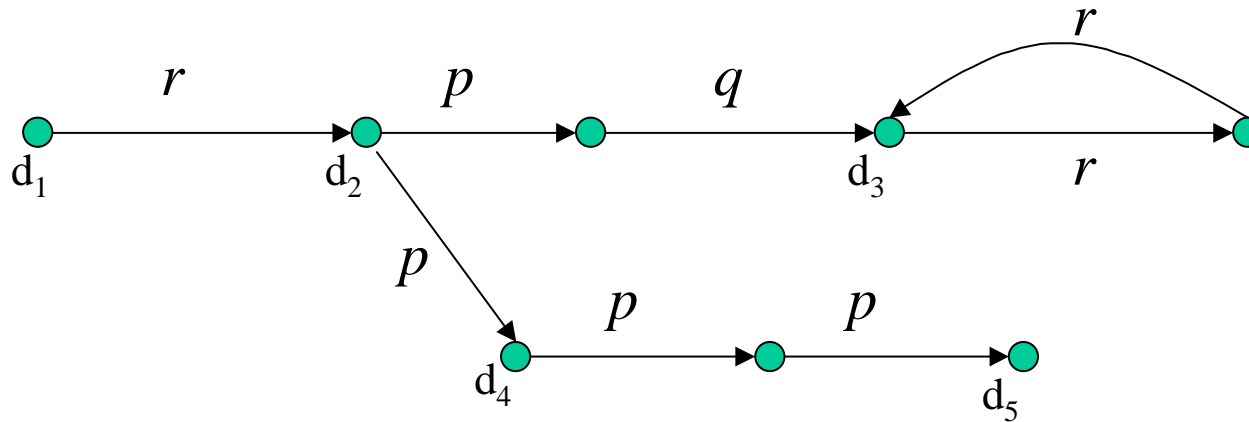
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_6

Transition: $(s_7, 0) \in \delta_A(s_6, d_3)$, s_7 final state

A run of $A_{(Q,d_1,d_3)}$



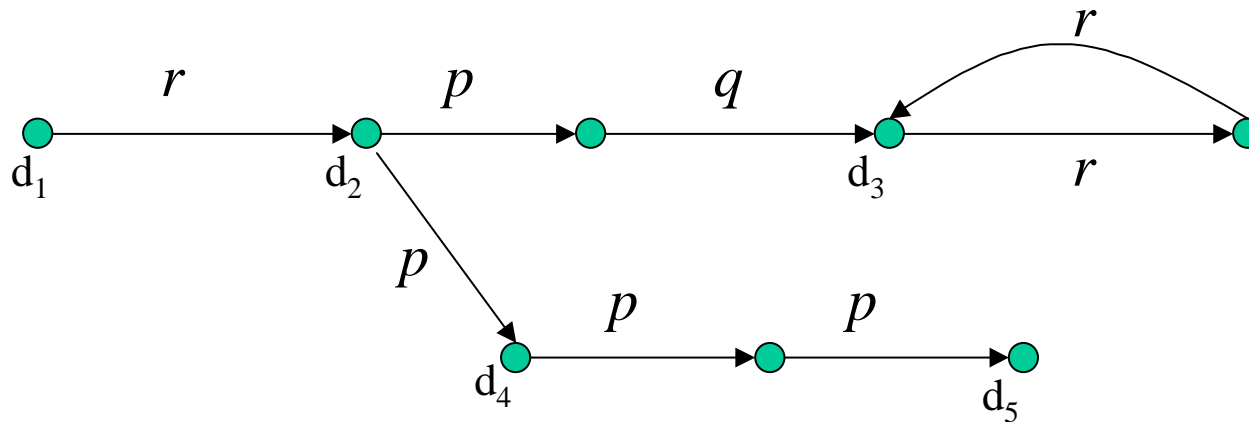
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_7

Transition: $(s_7, 1) \in \delta_A(s_7, d_3)$, s_7 final state

A run of $A_{(Q,d_1,d_3)}$



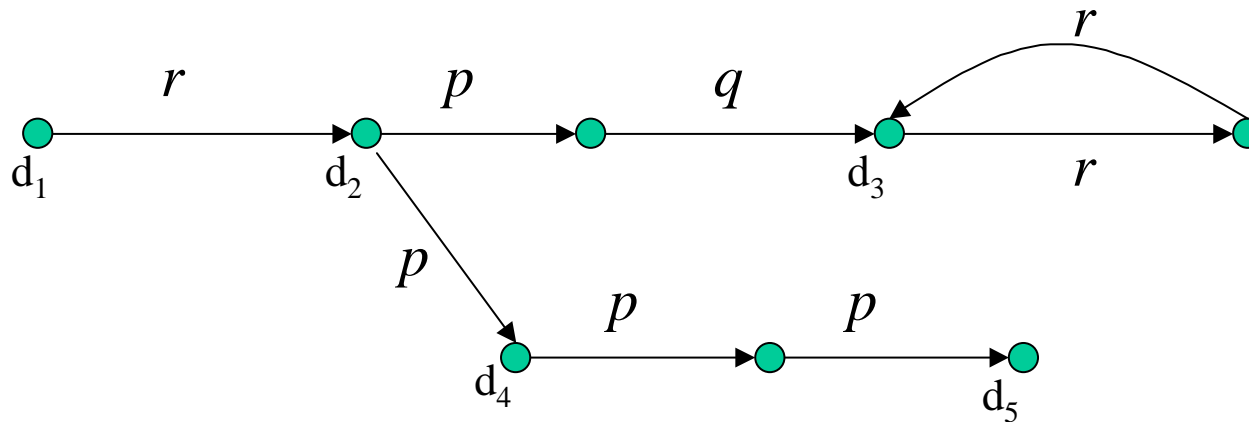
Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_7

Transition: $(s_7, 1) \in \delta_A(s_7, \$)$, s_7 final state

A run of $A_{(Q,d_1,d_3)}$



Word: $\$ d_4 p p d_5 \$ d_1 r d_2 \$ d_4 p^- d_2 \$ d_3 r r d_3 \$ d_2 p q d_3 \$$

$$Q = r \cdot (p \cup q) \cdot (p^- \cdot p)^* \cdot q \cdot q^*$$

State: s_7 final state

Word accepted by $A_{(Q,d_1,d_3)}$!

Query answering: Technique

To check whether $(c, d) \notin Q^{\mathcal{B}}$ for some $\mathcal{B} \in \text{mod}^{\mathcal{C}}(\mathcal{I})$, we check for nonemptiness of A , that is the **intersection** of

- the one-way automaton A_0 that accepts words that represent databases, i.e., words of the form $(\$ \cdot \mathcal{C} \cdot \Sigma^+ \cdot \mathcal{C})^* \cdot \$$
- the one-way automata corresponding to the various $A_{(S_i, a, b)}$ (for each source S_i and for each pair $(a, b) \in S_i^{\mathcal{C}}$)
- the one-way automaton corresponding to the complement of $A_{(Q, c, d)}$

Indeed, any word accepted by such intersection automaton represents a counterexample to $(c, d) \in Q^{\mathcal{I}, \mathcal{C}}$.

Query answering: Complexity

- All two-way automata constructed above are of linear size in the size of Q , $def(S_1), \dots, def(S_k)$, and S_1^C, \dots, S_k^C . Hence, the corresponding one-way automata would be exponential.
- However, we do not need to construct A explicitly. Instead, we can construct it **on the fly** while checking for nonemptiness.

Query answering for 2RPQs is PSPACE-complete in combined complexity (as for RPQs).

Outline

- Query answering in LAV
- **Query answering in GAV**
- Beyond FOL
- Conclusions

Coming back to GAV

In GAV, the mapping \mathcal{M} is constituted by a set of assertions:

$$g \rightsquigarrow \phi_{\mathcal{S}} \quad \text{with the meaning} \quad \forall \mathbf{x} (\phi_{\mathcal{S}}(\mathbf{x}) \rightarrow g(\mathbf{x}))$$

one for each element g in \mathcal{G} , where $\phi_{\mathcal{S}}$ is a query over \mathcal{S} .

If \mathcal{G} **does not have constraints**, we can simply limit our attention to **one** database of the data integration system, and answering queries reduces to

- using \mathcal{M} for computing from \mathcal{C} **the retrieved global database**, where each element g of \mathcal{G} satisfies exactly the tuples satisfying the $\phi_{\mathcal{S}}$ that \mathcal{M} associates to g ,
- evaluating the query q over the retrieved global database.

GAV with constraints in the global schema: example

Consider $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with

Global schema \mathcal{G} :

student($Scode, Sname, Scity$), $key\{Scode\}$

university($Ucode, Uname$), $key\{Ucode\}$

enrolled($Scode, Ucode$), $key\{Scode, Ucode\}$

enrolled[$Scode$] \subseteq student[$Scode$]

enrolled[$Ucode$] \subseteq university[$Ucode$]

Sources \mathcal{S} : s_1, s_2, s_3

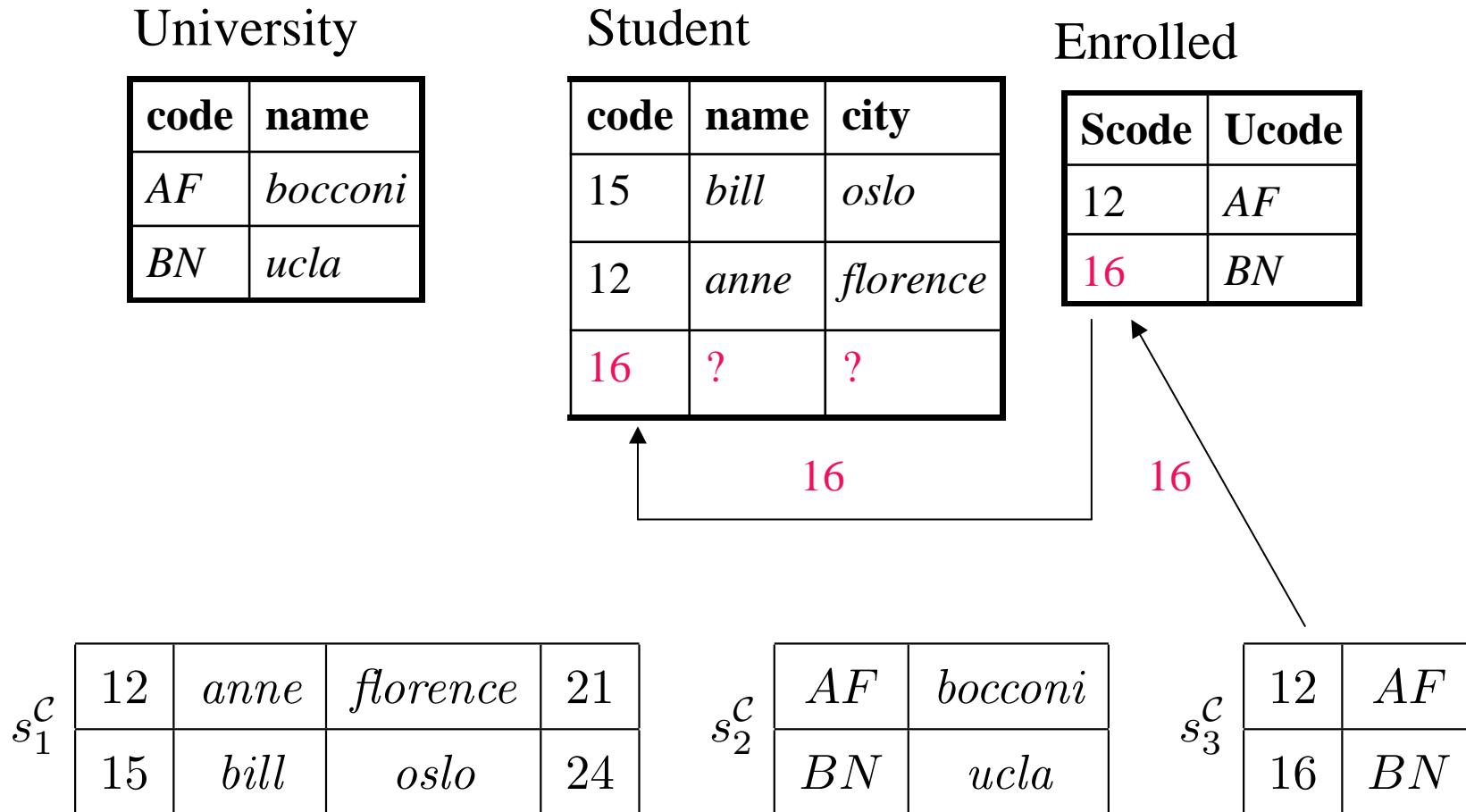
Mapping \mathcal{M} :

student \rightsquigarrow { (X, Y, Z) | $s_1(X, Y, Z, W)$ }

university \rightsquigarrow { (X, Y) | $s_2(X, Y)$ }

enrolled \rightsquigarrow { (X, W) | $s_3(X, W)$ }

Constraints in GAV: example



Constraints in GAV: example

Source database \mathcal{C} :

 $s_1^{\mathcal{C}}$

12	<i>anne</i>	<i>florence</i>	21
15	<i>bill</i>	<i>oslo</i>	24

 $s_2^{\mathcal{C}}$

<i>AF</i>	<i>bocconi</i>
<i>BN</i>	<i>ucla</i>

 $s_3^{\mathcal{C}}$

12	<i>AF</i>
16	<i>BN</i>

$s_3^{\mathcal{C}}(16, BN)$ implies $\text{enrolled}^{\mathcal{B}}(16, BN)$, for all $\mathcal{B} \in \text{mod}^{\mathcal{C}}(\mathcal{I})$.

Due to the integrity constraints in the global schema, **16 is the code of some student** in all $\mathcal{B} \in \text{mod}^{\mathcal{C}}(\mathcal{I})$.

Since \mathcal{C} says nothing about the name and the city of the student with code 16, we must accept as legal for \mathcal{I} wrt \mathcal{C} all virtual global databases that differ in such attributes.

GAV with constraints in the global schema

If \mathcal{G} does have constraints, then several situations become meaningful, given the source database \mathcal{C} :

- no database exists for the data integration system,
- **several databases** exist for the data integration system.

In GAV too, answering queries requires coping with [incomplete information](#).

Query answering in GAV

We deal with the problem of answering queries to data integration systems of the form $\langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, where

- the global schema \mathcal{G} is relational, with both key and foreign key constraints
- the sources in \mathcal{S} are relational
- the mapping \mathcal{M} is of type GAV
- views in \mathcal{M} are unions of conjunctive queries

Joint work with Domenico Lembo and Riccardo Rosati

Unfolding is not sufficient in our context

Mapping \mathcal{M} :

$$\begin{aligned} \text{student} &\rightsquigarrow \{ (X, Y, Z) \mid s_1(X, Y, Z, W) \} \\ \text{university} &\rightsquigarrow \{ (X, Y) \mid s_2(X, Y) \} \\ \text{enrolled} &\rightsquigarrow \{ (X, W) \mid s_3(X, W) \} \end{aligned}$$

 $s_1^{\mathcal{C}}$

12	<i>anne</i>	<i>florence</i>	21
15	<i>bill</i>	<i>oslo</i>	24

 $s_2^{\mathcal{C}}$

<i>AF</i>	<i>bocconi</i>
<i>BN</i>	<i>ucla</i>

 $s_3^{\mathcal{C}}$

12	<i>AF</i>
16	<i>BN</i>

Query: $\{ (X) \mid \text{student}(X, Y, Z), \text{enrolled}(X, W) \}$

Unfolding wrt \mathcal{M} : $\{ (X) \mid s_1(X, Y, Z, V), s_3(X, W) \}$

retrieves only the answer $\{12\}$ from \mathcal{C} , although $\{12, 16\}$ is the correct answer. The simple unfolding strategy is **not sufficient** in our context.

Logic program from a GAV system

We assume that \mathcal{M} does not violate any key constraint of \mathcal{G} (see later), and we associate to \mathcal{G} a logic program $\mathcal{P}_{\mathcal{G}}$, as follows.

- For each g in \mathcal{G} we have a rule in $\mathcal{P}_{\mathcal{G}}$ of the form:

$$g'(X_1, \dots, X_n) \leftarrow g(X_1, \dots, X_n)$$

- For each foreign key constraint

$$g_1[\mathbf{A}] \subseteq g_2[\mathbf{B}]$$

in \mathcal{G} where \mathbf{A} and \mathbf{B} are sets of attributes, we have a rule in $\mathcal{P}_{\mathcal{G}}$ of the form (the attributes in \mathbf{A} and \mathbf{B} are the first h in g_1 and g_2 , respectively; the f_i 's are fresh Skolem functions):

$$g_2'(X_1, \dots, X_h, f_1(X_1, \dots, X_h), \dots, f_{n-h}(X_1, \dots, X_h)) \leftarrow g_1'(X_1, \dots, X_h, \dots, X_m)$$

Processing queries in GAV: technique

Techniques for processing query q wrt $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$:

- We construct $\mathcal{P}_{\mathcal{G}}$ from \mathcal{G}
- We **partially evaluate** $\mathcal{P}_{\mathcal{G}}$ wrt q , and we obtain another query $exp_{\mathcal{G}}(q)$, called the expansion of q wrt the constraints of \mathcal{G}
- We unfold $exp_{\mathcal{G}}(q)$ wrt \mathcal{M} , and obtain a query $unf_{\mathcal{M}}(exp_{\mathcal{G}}(q))$ over the sources
- We evaluate $unf_{\mathcal{M}}(exp_{\mathcal{G}}(q))$ over the source database \mathcal{C}

$exp_{\mathcal{G}}(q)$ can be of exponential size wrt \mathcal{G} , but the whole process has polynomial time complexity wrt the size of \mathcal{C} .

Example

Suppose we have $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with \mathcal{G} :

person(*Pcode*, *Age*, *CityOfBirth*)

student(*Scode*, *University*)

city(*Name*, *Major*)

$key(\text{person}) = \{Pcode\}$

$key(\text{student}) = \{Scode\}$

$key(\text{city}) = \{Name\}$

$\text{person}[CityOfBirth] \subseteq \text{city}[Name]$

$\text{city}[Major] \subseteq \text{person}[PCode]$

$\text{student}[SCode] \subseteq \text{person}[PCode]$

Processing queries in GAV: technique

The logic program $\mathcal{P}_{\mathcal{G}}$ is

$$\begin{aligned} \text{person}'(X, Y, Z) &\leftarrow \text{person}(X, Y, Z) \\ \text{student}'(X, Y) &\leftarrow \text{student}(X, Y) \\ \text{city}'(X, Y) &\leftarrow \text{city}(X, Y) \\ \text{city}'(X, f_1(X)) &\leftarrow \text{person}'(Y, Z, X) \\ \text{person}'(Y, f_2(Y), f_3(Y)) &\leftarrow \text{city}'(X, Y) \\ \text{person}'(X, f_4(X), f_5(X)) &\leftarrow \text{student}'(X, Y) \end{aligned}$$

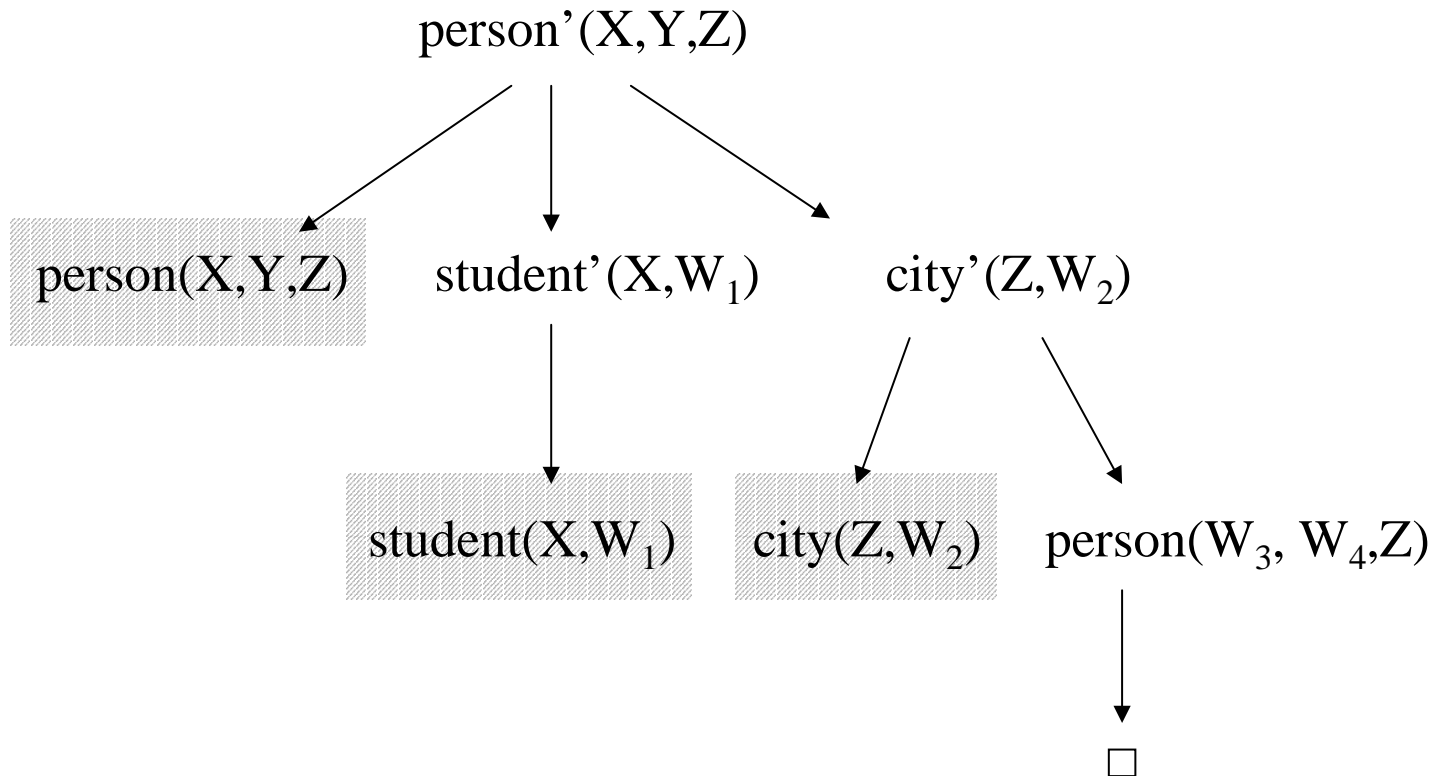
Consider the query

$$\{ (X) \mid \text{person}(X, Y, Z) \}$$

written as the rule

$$q(X) \leftarrow \text{person}'(X, Y, Z)$$

Example



$exp_G(q)$ is

$$\{ (X) \mid \text{person}(X, Y, Z) \vee \text{student}(X, W) \vee \text{city}(Z, X) \}$$

Outline

- Query answering in LAV
- Query answering in GAV
- **Beyond FOL**
- Conclusions

Beyond first-order logic

We now relax the assumption that \mathcal{M} does not violate the key constraints of \mathcal{G} . As we said before, in this general case, several situations for data integration system $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ and source database \mathcal{C} are possible:

- **no database** exists for the data integration system,
- the data integration system has one database,
- several databases exist for the data integration system.

When for data integration system $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ and source database \mathcal{C} , we have $mod^{\mathcal{C}}(\mathcal{I}) = \emptyset$, the first-order setting described above is not adequate.

Beyond first-order logic: related work

Several recent proposals address this problem:

- [Subrahmanian ACM-TODS'94]
- [Grant&al. IEEE-TKDE'95]
- [Dung CoopIS'96]
- [Lin&al. JICIS'98]
- [Yan&al. CoopIS'99]
- [Arenas&al. PODS'99]
- [Greco&al. LPAR'00]
- many approaches to KB revision and KB update

Beyond first-order logic: a proposal

Given $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ (with $\mathcal{M} = \{r_1 \rightsquigarrow V_1, \dots, r_n \rightsquigarrow V_n\}$), and source database \mathcal{C} for \mathcal{S} , we would like to focus our attention on those databases \mathcal{B} that

1. satisfy \mathcal{G} (constraints in \mathcal{G} are strong), and
2. **approximate as much as possible** the satisfaction relation for the mapping \mathcal{M} (assertions in \mathcal{M} are soft).

If \mathcal{B}_1 and \mathcal{B}_2 are two databases that satisfy \mathcal{G} , we say that \mathcal{B}_1 is **better** than \mathcal{B}_2 wrt \mathcal{I} and \mathcal{C} , denoted as $\mathcal{B}_1 \gg_{\mathcal{C}}^{\mathcal{I}} \mathcal{B}_2$, if there exists an assertion $r_i \rightsquigarrow V_i$ in \mathcal{M} such that

- $(r_i^{\mathcal{B}_1} \cap V_i^{\mathcal{C}}) \supset (r_i^{\mathcal{B}_2} \cap V_i^{\mathcal{C}})$, and
- $(r_j^{\mathcal{B}_1} \cap V_j^{\mathcal{C}}) \supseteq (r_j^{\mathcal{B}_2} \cap V_j^{\mathcal{C}})$ for all $r_i \rightsquigarrow V_i$ in \mathcal{M} with $j \neq i$.

Example

Consider $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with

- \mathcal{G} containing relation $r(x, y)$ with key x ,
- \mathcal{S} containing relations $s_1(x, y)$ and $s_2(x, y)$
- $\mathcal{M} = \{ r \rightsquigarrow \{ (x, y) \mid s_1(x, y) \vee s_2(x, y) \} \}$

and consider the source database $\mathcal{C} = \{ s_1(a, d), s_1(b, d), s_2(a, e) \}$, so that the retrieved global database is $\{ r(a, d), r(b, d), r(a, e) \}$

We have that

- $\{ r(a, d), r(b, d) \} \gg_{\mathcal{C}}^{\mathcal{I}} \{ r(a, d) \}$, $\{ r(a, e), r(b, d) \} \gg_{\mathcal{C}}^{\mathcal{I}} \{ r(a, e) \}$
- $\{ r(a, d), r(b, d) \}$ and $\{ r(a, e) \}$ are incomparable
- $\{ r(a, e), r(b, d), r(c, e) \}$ and $\{ r(a, e), r(b, d) \}$ are incomparable

Beyond first-order logic: a proposal

$\gg_{\mathcal{C}}^{\mathcal{I}}$ is a partial order.

A database \mathcal{B} that satisfy \mathcal{G} satisfies the mapping \mathcal{M} with respect to \mathcal{C} if \mathcal{B} is maximal wrt $\gg_{\mathcal{C}}^{\mathcal{I}}$, i.e., for no other global database \mathcal{B}' that satisfies \mathcal{G} , we have that $\mathcal{B}' \gg_{\mathcal{C}}^{\mathcal{I}} \mathcal{B}$:

$$\text{mod}^{\mathcal{C}}(\mathcal{I}) = \{ \mathcal{B} \mid \mathcal{B} \text{ is a database that satisfies } \mathcal{G}, \text{ and such that } \\ \neg \exists \mathcal{B}' \text{ such that } \mathcal{B}' \text{ satisfies } \mathcal{G} \text{ and } \mathcal{B}' \gg_{\mathcal{C}}^{\mathcal{I}} \mathcal{B} \}$$

The notion of legal database for \mathcal{I} with respect to \mathcal{C} , and the notion of certain answer remain the same, given the new definition of satisfaction of mapping.

Beyond first-order logic: a proposal

Given $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, we define the DATALOG[⊃] program $\mathcal{P}(\mathcal{I}, \mathcal{C})$ obtained by adding to the set of facts \mathcal{C} the following set of rules:

- for each $g \rightsquigarrow \{(\vec{x}) \mid body_1(\vec{x}, \vec{y}_1) \vee \dots \vee body_m(\vec{x}, \vec{y}_m)\}$ in \mathcal{M} , the rules:

$$g_{\mathcal{C}}(\vec{X}) \leftarrow body_1(\vec{X}, \vec{Y}_1) \quad \dots \quad g_{\mathcal{C}}(\vec{X}) \leftarrow body_m(\vec{X}, \vec{Y}_m)$$

- for each relation $g \in \mathcal{G}$, the rules

$$\begin{aligned} g(\vec{X}, \vec{Y}) &\leftarrow g_{\mathcal{C}}(\vec{X}, \vec{Y}), \text{ not } \bar{g}(\vec{X}, \vec{Y}) \\ \bar{g}(\vec{X}, \vec{Y}) &\leftarrow g(\vec{X}, \vec{Z}), \vec{Y} \neq \vec{Z} \end{aligned}$$

where

- in $g(\vec{X}, \vec{Y})$, \vec{X} is the key of g
- $\vec{Y} \neq \vec{Z}$ if there exists i such that $Y_i \neq Z_i$.

Beyond first-order logic: a proposal

The above rules force each stable model T of $\mathcal{P}(\mathcal{I}, \mathcal{C})$ to be such that, for each g in \mathcal{G} , g^T is a maximal subset of the tuples from the retrieved global database that are consistent with the key constraint for g .

- $t \in q^{\mathcal{I}, \mathcal{C}}$ if and only if $t \in q^T$ for each stable model T of the DATALOG[∇] program $\mathcal{P}(\mathcal{I}, \mathcal{C}) \cup \{exp_{\mathcal{G}}(q)\}$.
- The problem of deciding whether $t \in q^{\mathcal{I}, \mathcal{C}}$ is coNP-complete wrt data complexity.

Outline

- Query answering in LAV
- Query answering in GAV
- Beyond FOL
- **Conclusions**

Conclusions

- View-based query processing is strongly related to answering queries with incomplete information.
- This is true both in LAV and GAV data integration systems.
- Approaches going beyond first-order logics are needed to cope with inconsistencies between sources.