

CS294-248 Special Topics in Database Theory
Unit 6: Constraints, Incomplete and Probabilistic
Databases

Dan Suciu

University of Washington

Outline

- Today: Generalized Constraints, Semantics Optimization.
- Thursday: Repairs, Incomplete Databases

Constraints

A **constraint** is an assertion on the database D that must always hold.

How does this differ from invariants in programs?

Constraints

A **constraint** is an assertion on the database D that must always hold.

How does this differ from invariants in programs?

Constraints: we check them at runtime (this may be costly)

Invariants: we prove them offline, do not check at runtime.

Applications of Constraints

- Enforce database consistency.
Most common constraint in practice:
“Please type in your phone number using $XXX - XXX - XXXX$ ”;
- Database normalization.
- Semantic optimization: given query Q find a “better” query Q' s.t.
 $Q \equiv Q'$ on databases satisfying the constraints.
- Database repair: if $D \not\models \Sigma$, delete/insert tuples s.t. $D' \models \Sigma$.
- Consistent query answering: given query Q return only those answers that are present in $Q(D')$ for all repairs D' .

Classical Database Constraints

Classical Database Constraints

- Functional Dependencies (FD).
- Multivalued Dependencies (MVD).
- Join Dependencies (JD).
- Inclusion Dependencies (IND).

Functional Dependency

Notation:

$$\boxed{U \rightarrow V}$$

Semantics: $R^D \models U \rightarrow V$ if:

$$\forall u, v_1, w_1, v_2, w_2 (R(u, v_1, w_1) \wedge R(u, v_2, w_2) \Rightarrow v_1 = v_2)$$

Consequence

Lossless decomposition: $R(\mathbf{U}, \mathbf{V}, \mathbf{W}) = R_1(\mathbf{U}, \mathbf{V}) \bowtie R_2(\mathbf{U}, \mathbf{W})$.

The implication problem: axiomatizable (Armstrong), decidable in PTIME.

Multivalued Dependency

Notation: given a partition all attribute $\mathbf{X} = \mathbf{U} \cup \mathbf{V} \cup \mathbf{W}$:

$$\boxed{\mathbf{U} \twoheadrightarrow \mathbf{V}; \mathbf{W}}$$

Semantics: $R^D \models \mathbf{U} \twoheadrightarrow \mathbf{V}; \mathbf{W}$ if:

$$\forall \mathbf{u}, \mathbf{v}_1, \mathbf{w}_1, \mathbf{v}_2, \mathbf{w}_2 (R(\mathbf{u}, \mathbf{v}_1, \mathbf{w}_1) \wedge R(\mathbf{u}, \mathbf{v}_2, \mathbf{w}_2) \Rightarrow R(\mathbf{u}, \mathbf{v}_1, \mathbf{w}_2))$$

Equivalent Definition

Lossless decomposition: $R(\mathbf{U}, \mathbf{V}, \mathbf{W}) = R_1(\mathbf{U}, \mathbf{V}) \bowtie R_2(\mathbf{U}, \mathbf{W})$.

The implication problem for FD+MVD: axiomatizable, decidable.

Join Dependencies

Notation: given a cover of all attributes $\mathbf{X} = \mathbf{U}_1 \cup \dots \cup \mathbf{U}_k$:

$$\bowtie (\mathbf{U}_1, \mathbf{U}_2, \dots, \mathbf{U}_k)$$

Join Dependencies

Notation: given a cover of all attributes $\mathbf{X} = \mathbf{U}_1 \cup \dots \cup \mathbf{U}_k$:

$$\bowtie (\mathbf{U}_1, \mathbf{U}_2, \dots, \mathbf{U}_k)$$

Semantics by example. $R^D(X, Y, Z) \models \bowtie (XY, YZ, XZ)$ if R^D satisfies

$$\forall x, x', y, y, z, z' (R(x, y, z') \wedge R(x', y, z) \wedge R(x, y', z)) \Rightarrow R(x, y, z)$$

Join Dependencies

Notation: given a cover of all attributes $\mathbf{X} = \mathbf{U}_1 \cup \dots \cup \mathbf{U}_k$:

$$\bowtie (\mathbf{U}_1, \mathbf{U}_2, \dots, \mathbf{U}_k)$$

Semantics by example. $R^D(X, Y, Z) \models_{\bowtie} (XY, YZ, XZ)$ if R^D satisfies

$$\forall x, x', y, y, z, z' (R(x, y, z') \wedge R(x', y, z) \wedge R(x, y', z)) \Rightarrow R(x, y, z)$$

Equivalently: $R^D \models_{\bowtie} (\mathbf{U}_1, \dots, \mathbf{U}_k)$ if:

Definition of JD

Lossless decomposition: $R(\mathbf{X}) = R_1(\mathbf{U}_1) \bowtie \dots \bowtie R_k(\mathbf{U}_k)$

Join Dependencies

Notation: given a cover of all attributes $\mathbf{X} = \mathbf{U}_1 \cup \dots \cup \mathbf{U}_k$:

$$\bowtie (\mathbf{U}_1, \mathbf{U}_2, \dots, \mathbf{U}_k)$$

Semantics by example. $R^D(X, Y, Z) \models \bowtie (XY, YZ, XZ)$ if R^D satisfies

$$\forall x, x', y, y, z, z' (R(x, y, z') \wedge R(x', y, z) \wedge R(x, y', z)) \Rightarrow R(x, y, z)$$

Equivalently: $R^D \models \bowtie (\mathbf{U}_1, \dots, \mathbf{U}_k)$ if:

Definition of JD

Lossless decomposition: $R(\mathbf{X}) = R_1(\mathbf{U}_1) \bowtie \dots \bowtie R_k(\mathbf{U}_k)$

JD implication problem not axiomatizable [Abiteboul et al., 1995, pp.171].

FD+JD implication problem is decidable (later).

Inclusion Dependencies

Notation: relation schemas $R(\mathbf{X}), S(\mathbf{Y}), \mathbf{U} \subseteq \mathbf{X}, \mathbf{V} \subseteq \mathbf{Y}, |\mathbf{U}| = |\mathbf{V}|$:

$$R[\mathbf{U}] \subseteq S[\mathbf{V}]$$

Inclusion Dependencies

Notation: relation schemas $R(\mathbf{X}), S(\mathbf{Y}), \mathbf{U} \subseteq \mathbf{X}, \mathbf{V} \subseteq \mathbf{Y}, |\mathbf{U}| = |\mathbf{V}|$:

$$R[\mathbf{U}] \subseteq S[\mathbf{V}]$$

Semantics: (what you expect, but watch the FO sentence):

$$\forall \mathbf{u} \forall \mathbf{r} (R(\mathbf{u}, \mathbf{r}) \Rightarrow \exists \mathbf{s} S(\mathbf{u}, \mathbf{s}))$$

Inclusion Dependencies

Notation: relation schemas $R(\mathbf{X}), S(\mathbf{Y}), \mathbf{U} \subseteq \mathbf{X}, \mathbf{V} \subseteq \mathbf{Y}, |\mathbf{U}| = |\mathbf{V}|$:

$$R[\mathbf{U}] \subseteq S[\mathbf{V}]$$

Semantics: (what you expect, but watch the FO sentence):

$$\forall \mathbf{u} \forall \mathbf{r} (R(\mathbf{u}, \mathbf{r}) \Rightarrow \exists \mathbf{s} S(\mathbf{u}, \mathbf{s}))$$

[Abiteboul et al., 1995, pp.171-202]:

- IND is axiomatizable (3 simple axioms).
- The implication problem for IND is PSPACE complete.
- The implication problem for FD+IND is undecidable.

Discussion

FDs, MVDs, JDs, INDs, . . . , why so many kinds?

It turns out that all can be captured by a single formalism:

Generalized Dependencies

Generalized Dependencies

Generalized Dependency

Relational schema: R_1, R_2, \dots

A **Generalized Dependency** is a statement of one of these two forms:

Tuple-Generating Dependency (TGD):

$$\boxed{\forall \mathbf{x} (A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y} (B_1 \wedge \dots \wedge B_k))}$$

Generalized Dependency

Relational schema: R_1, R_2, \dots

A **Generalized Dependency** is a statement of one of these two forms:

Tuple-Generating Dependency (TGD):

$$\boxed{\forall \mathbf{x} (A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y} (B_1 \wedge \dots \wedge B_k))}$$

The TGD is **full** if there is no $\exists \mathbf{y}$

Generalized Dependency

Relational schema: R_1, R_2, \dots

A **Generalized Dependency** is a statement of one of these two forms:

Tuple-Generating Dependency (TGD):

$$\boxed{\forall \mathbf{x} (A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y} (B_1 \wedge \dots \wedge B_k))}$$

The TGD is **full** if there is no $\exists \mathbf{y}$

Equality-Generating Dependency (EGD):

$$\boxed{\forall \mathbf{x} (A_1 \wedge \dots \wedge A_m \Rightarrow x_i = x_j)}$$

Generalized Dependency

Relational schema: R_1, R_2, \dots

A **Generalized Dependency** is a statement of one of these two forms:

Tuple-Generating Dependency (TGD):

$$\forall \mathbf{x} (A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y} (B_1 \wedge \dots \wedge B_k))$$

The TGD is **full** if there is no $\exists \mathbf{y}$

Equality-Generating Dependency (EGD):

$$\forall \mathbf{x} (A_1 \wedge \dots \wedge A_m \Rightarrow x_i = x_j)$$

Examples

FD: $\forall u \forall x_1 \forall x_2 (R(u, x_1) \wedge R(u, x_2) \Rightarrow x_1 = x_2)$.

Generalized Dependency

Relational schema: R_1, R_2, \dots

A **Generalized Dependency** is a statement of one of these two forms:

Tuple-Generating Dependency (TGD):

$$\boxed{\forall \mathbf{x} (A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y} (B_1 \wedge \dots \wedge B_k))}$$

The TGD is **full** if there is no $\exists \mathbf{y}$

Equality-Generating Dependency (EGD):

$$\boxed{\forall \mathbf{x} (A_1 \wedge \dots \wedge A_m \Rightarrow x_i = x_j)}$$

Examples

FD: $\forall u \forall x_1 \forall x_2 (R(u, x_1) \wedge R(u, x_2) \Rightarrow x_1 = x_2)$.

MVD: $\forall u, v_1, w_1, v_2, w_2 (R(u, v_1, w_1) \wedge R(u, v_2, w_2) \Rightarrow R(u, v_1, w_2))$.

Generalized Dependency

Relational schema: R_1, R_2, \dots

A **Generalized Dependency** is a statement of one of these two forms:

Tuple-Generating Dependency (TGD):

$$\boxed{\forall \mathbf{x} (A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y} (B_1 \wedge \dots \wedge B_k))}$$

The TGD is **full** if there is no $\exists \mathbf{y}$

Equality-Generating Dependency (EGD):

$$\boxed{\forall \mathbf{x} (A_1 \wedge \dots \wedge A_m \Rightarrow x_i = x_j)}$$

Examples

FD: $\forall u \forall x_1 \forall x_2 (R(u, x_1) \wedge R(u, x_2) \Rightarrow x_1 = x_2)$.

MVD: $\forall u, v_1, w_1, v_2, w_2 (R(u, v_1, w_1) \wedge R(u, v_2, w_2) \Rightarrow R(u, v_1, w_2))$.

IND: $\forall x \forall x' (R(x, x') \Rightarrow \exists y S(x, y'))$.

Dissecting Generalized Dependencies

$$\forall \mathbf{x}(A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y}(B_1 \wedge \dots \wedge B_k))$$

- Need \exists on the right, but not on the left:

$$\forall x(\exists y R(x, y) \Rightarrow \exists z S(x, z))$$

Dissecting Generalized Dependencies

$$\forall \mathbf{x}(A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y}(B_1 \wedge \dots \wedge B_k))$$

- Need \exists on the right, but not on the left:

$$\forall x(\exists y R(x, y) \Rightarrow \exists z S(x, z)) \text{ equivalent to } \forall x \forall y (R(x, y) \Rightarrow \exists z S(x, z))$$

Dissecting Generalized Dependencies

$$\forall \mathbf{x}(A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y}(B_1 \wedge \dots \wedge B_k))$$

- Need \exists on the right, but not on the left:

$$\forall x(\exists y R(x, y) \Rightarrow \exists z S(x, z)) \text{ equivalent to } \forall x \forall y (R(x, y) \Rightarrow \exists z S(x, z))$$

- When \exists is missing, then we can split the RHS:

$$\forall x \forall y (R(x, y) \Rightarrow S(x) \wedge (x = y))$$

Dissecting Generalized Dependencies

$$\forall \mathbf{x}(A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y}(B_1 \wedge \dots \wedge B_k))$$

- Need \exists on the right, but not on the left:

$$\forall x(\exists y R(x, y) \Rightarrow \exists z S(x, z)) \text{ equivalent to } \forall x \forall y (R(x, y) \Rightarrow \exists z S(x, z))$$

- When \exists is missing, then we can split the RHS:

$$\begin{array}{ll} \forall x \forall y (R(x, y) \Rightarrow S(x) \wedge (x = y)) & \text{equivalent to two GDs:} \\ \forall x \forall y (R(x, y) \Rightarrow S(x)) & \forall x \forall y (R(x, y) \Rightarrow (x = y)) \end{array}$$

Dissecting Generalized Dependencies

$$\forall \mathbf{x}(A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y}(B_1 \wedge \dots \wedge B_k))$$

- Need \exists on the right, but not on the left:

$$\forall x(\exists y R(x, y) \Rightarrow \exists z S(x, z)) \text{ equivalent to } \forall x \forall y (R(x, y) \Rightarrow \exists z S(x, z))$$

- When \exists is missing, then we can split the RHS:

$$\begin{array}{ll} \forall x \forall y (R(x, y) \Rightarrow S(x) \wedge (x = y)) & \text{equivalent to two GDs:} \\ \forall x \forall y (R(x, y) \Rightarrow S(x)) & \forall x \forall y (R(x, y) \Rightarrow (x = y)) \end{array}$$

- A GD is equivalent to a query containment assertion:

$$\begin{array}{ll} \forall x (\exists y R(x, y) \Rightarrow \exists z S(x, z)) & \text{is equivalent to:} \\ Q_1 \subseteq Q_2 & \text{where } Q_1(x) \stackrel{\text{def}}{=} \exists y R(x, y), Q_2(x) \stackrel{\text{def}}{=} \exists z S(x, z). \end{array}$$

Dissecting Generalized Dependencies

$$\forall \mathbf{x}(A_1 \wedge \dots \wedge A_m \Rightarrow \exists \mathbf{y}(B_1 \wedge \dots \wedge B_k))$$

- Need \exists on the right, but not on the left:

$$\forall x(\exists y R(x, y) \Rightarrow \exists z S(x, z)) \text{ equivalent to } \forall x \forall y (R(x, y) \Rightarrow \exists z S(x, z))$$

- When \exists is missing, then we can split the RHS:

$$\begin{array}{ll} \forall x \forall y (R(x, y) \Rightarrow S(x) \wedge (x = y)) & \text{equivalent to two GDs:} \\ \forall x \forall y (R(x, y) \Rightarrow S(x)) & \forall x \forall y (R(x, y) \Rightarrow (x = y)) \end{array}$$

- A GD is equivalent to a query containment assertion:

$$\begin{array}{ll} \forall x(\exists y R(x, y) \Rightarrow \exists z S(x, z)) & \text{is equivalent to:} \\ Q_1 \subseteq Q_2 & \text{where } Q_1(x) \stackrel{\text{def}}{=} \exists y R(x, y), Q_2(x) \stackrel{\text{def}}{=} \exists z S(x, z). \end{array}$$

- To check $\mathbf{D} \models \sigma$, compute $Q_1(\mathbf{D}), Q_2(\mathbf{D})$; in PTIME.

Discussion

- GDs are a fragment of FO, powerful enough to capture classical constraints, yet weak enough to be useful.
- Next: we show their utility in [semantics optimization](#).

Semantic Query Optimization

Overview

Semantics Query Optimization means query optimization that uses the database constraints Σ

Replace a query Q by Q' such that $Q(\mathbf{D}) = Q'(\mathbf{D})$ for every database instance \mathbf{D} that satisfies the constraints.

We write $\Sigma \models Q \equiv Q'$

Note that, in general, $Q \not\equiv Q'$.

Semantic optimization is an old idea
[King, 1981, Chakravarthy et al., 1990].

Example

$$Q_1(z) = R(x, 55) \wedge R(x, y) \wedge S(y, z)$$

$$Q_2(z) = S(55, z)$$

Which of the following hold?

$$Q_1 \subseteq Q_2?$$

$$Q_2 \subseteq Q_1?$$

Example

$$Q_1(z) = R(x, 55) \wedge R(x, y) \wedge S(y, z)$$

$$Q_2(z) = S(55, z)$$

Which of the following hold?

$Q_1 \subseteq Q_2$? NO

$Q_2 \subseteq Q_1$?

Example

$$Q_1(z) = R(x, 55) \wedge R(x, y) \wedge S(y, z)$$

$$Q_2(z) = S(55, z)$$

Which of the following hold?

$Q_1 \subseteq Q_2$? NO

$Q_2 \subseteq Q_1$? NO

(In class: show canonical database refuting these containments)

Example

$$Q_1(z) = R(x, 55) \wedge R(x, y) \wedge S(y, z)$$

$$Q_2(z) = S(55, z)$$

Which of the following hold?

$Q_1 \subseteq Q_2$? NO

$Q_2 \subseteq Q_1$? NO

(In class: show canonical database refuting these containments)

What constraint implies $Q_1 \subseteq Q_2$?

Example

$$Q_1(z) = R(x, 55) \wedge R(x, y) \wedge S(y, z)$$

$$Q_2(z) = S(55, z)$$

Which of the following hold?

$Q_1 \subseteq Q_2$? NO

$Q_2 \subseteq Q_1$? NO

(In class: show canonical database refuting these containments)

What constraint implies $Q_1 \subseteq Q_2$?

$R.x$ is a key:

$$\sigma_1 : \forall x, y, w (R(x, w) \wedge R(x, y) \Rightarrow (w = y))$$

Example

$$Q_1(z) = R(x, 55) \wedge R(x, y) \wedge S(y, z)$$

$$Q_2(z) = S(55, z)$$

Which of the following hold? $Q_1 \subseteq Q_2$? NO $Q_2 \subseteq Q_1$? NO

(In class: show canonical database refuting these containments)

What constraint implies $Q_1 \subseteq Q_2$?

$R.x$ is a key:

$$\sigma_1 : \forall x, y, w (R(x, w) \wedge R(x, y) \Rightarrow (w = y))$$

Then $Q_1(z) \equiv R(x, 55) \wedge R(x, 55) \wedge S(55, z) \equiv R(x, 55) \wedge S(55, z)$

Example

$$Q_1(z) = R(x, 55) \wedge R(x, y) \wedge S(y, z)$$

$$Q_2(z) = S(55, z)$$

Which of the following hold? $Q_1 \subseteq Q_2$? NO $Q_2 \subseteq Q_1$? NO

(In class: show canonical database refuting these containments)

What constraint implies $Q_1 \subseteq Q_2$?

$R.x$ is a key:

$$\sigma_1 : \forall x, y, w (R(x, w) \wedge R(x, y) \Rightarrow (w = y))$$

Then $Q_1(z) \equiv R(x, 55) \wedge R(x, 55) \wedge S(55, z) \equiv R(x, 55) \wedge S(55, z)$

What constraint implies $Q_2 \subseteq Q_1$?

Example

$$Q_1(z) = R(x, 55) \wedge R(x, y) \wedge S(y, z)$$

$$Q_2(z) = S(55, z)$$

Which of the following hold? $Q_1 \subseteq Q_2$? NO $Q_2 \subseteq Q_1$? NO

(In class: show canonical database refuting these containments)

What constraint implies $Q_1 \subseteq Q_2$?

$R.x$ is a key:

$$\sigma_1 : \forall x, y, w (R(x, w) \wedge R(x, y) \Rightarrow (w = y))$$

Then $Q_1(z) \equiv R(x, 55) \wedge R(x, 55) \wedge S(55, z) \equiv R(x, 55) \wedge S(55, z)$

What constraint implies $Q_2 \subseteq Q_1$?

IND:

$$\sigma_2 : \forall y, z (S(y, z) \Rightarrow \exists x R(x, y)) .$$

Example

$$Q_1(z) = R(x, 55) \wedge R(x, y) \wedge S(y, z)$$

$$Q_2(z) = S(55, z)$$

Which of the following hold?

$Q_1 \subseteq Q_2$? NO

$Q_2 \subseteq Q_1$? NO

(In class: show canonical database refuting these containments)

What constraint implies $Q_1 \subseteq Q_2$?

$R.x$ is a key:

$$\sigma_1 : \forall x, y, w (R(x, w) \wedge R(x, y) \Rightarrow (w = y))$$

Then $Q_1(z) \equiv R(x, 55) \wedge R(x, 55) \wedge S(55, z) \equiv R(x, 55) \wedge S(55, z)$

What constraint implies $Q_2 \subseteq Q_1$?

IND:

$$\sigma_2 : \forall y, z (S(y, z) \Rightarrow \exists x R(x, y))$$

Then:

$Q_2(z) \equiv S(y, z) \wedge y = 55 \equiv R(x, y) \wedge S(y, z) \wedge y = 55 \equiv R(x, 55) \wedge S(55, z)$

Example

$$Q_1(z) = R(x, 55) \wedge R(x, y) \wedge S(y, z)$$

$$Q_2(z) = S(55, z)$$

Which of the following hold? $Q_1 \subseteq Q_2$? NO $Q_2 \subseteq Q_1$? NO

(In class: show canonical database refuting these containments)

What constraint implies $Q_1 \subseteq Q_2$?

$R.x$ is a key:

$$\sigma_1 : \forall x, y, w (R(x, w) \wedge R(x, y) \Rightarrow (w = y))$$

Then $Q_1(z) \equiv R(x, 55) \wedge R(x, 55) \wedge S(55, z) \equiv R(x, 55) \wedge S(55, z)$

What constraint implies $Q_2 \subseteq Q_1$?

IND:

$$\sigma_2 : \forall y, z (S(y, z) \Rightarrow \exists x R(x, y))$$

Then:

$Q_2(z) \equiv S(y, z) \wedge y = 55 \equiv R(x, y) \wedge S(y, z) \wedge y = 55 \equiv R(x, 55) \wedge S(55, z)$

Assume the database satisfies σ_1, σ_2 . Then we can optimize Q_1 to Q_2

The Chase: Overview

- The Chase takes a query Q and a GD σ and creates a new query Q_1 by “applying” σ to Q .
- The important semantics property of the chase is: $\sigma \models Q \equiv Q_1$.
- By repeatedly applying the chase we obtain a sequence Q, Q_1, Q_2, \dots
- To check $\Sigma \models Q \equiv Q'$ it suffices to find a chase sequence from Q to Q_m , and one from Q' to Q'_n , then prove $Q_m \equiv Q'_n$ (unconditioned).

Definition of the Chase

Let σ be $\forall \mathbf{x}(A \Rightarrow C)$ where A is a conjunction of atoms, Q be a CQ.

Definition (The Chase)

For a homomorphism $\theta : A \rightarrow Q$, we write $Q \xrightarrow{\sigma, \theta} Q'$ where Q' is:

- If σ is a TGD with $C \equiv \exists \mathbf{y} B$, then $Q' \stackrel{\text{def}}{=} Q \wedge \theta(B)$.
- If σ is an EGD with $C \equiv (x_i = x_j)$, then $Q' \stackrel{\text{def}}{=} Q[x_j/x_i]$.

Definition of the Chase

Let σ be $\forall \mathbf{x}(A \Rightarrow C)$ where A is a conjunction of atoms, Q be a CQ.

Definition (The Chase)

For a homomorphism $\theta : A \rightarrow Q$, we write $Q \xrightarrow{\sigma, \theta} Q'$ where Q' is:

- If σ is a TGD with $C \equiv \exists \mathbf{y} B$, then $Q' \stackrel{\text{def}}{=} Q \wedge \theta(B)$.
- If σ is an EGD with $C \equiv (x_i = x_j)$, then $Q' \stackrel{\text{def}}{=} Q[x_j/x_i]$.

Example $Q(x) = R(x, y) \wedge A(y) \wedge R(x, z) \wedge B(z)$

Definition of the Chase

Let σ be $\forall \mathbf{x}(A \Rightarrow C)$ where A is a conjunction of atoms, Q be a CQ.

Definition (The Chase)

For a homomorphism $\theta : A \rightarrow Q$, we write $Q \xrightarrow{\sigma, \theta} Q'$ where Q' is:

- If σ is a TGD with $C \equiv \exists \mathbf{y} B$, then $Q' \stackrel{\text{def}}{=} Q \wedge \theta(B)$.
- If σ is an EGD with $C \equiv (x_i = x_j)$, then $Q' \stackrel{\text{def}}{=} Q[x_j/x_i]$.

Example

$$Q(x) = R(x, y) \wedge A(y) \wedge R(x, z) \wedge B(z)$$

$$\sigma_1 = \forall u \forall v \forall w (R(u, v) \wedge R(u, w) \Rightarrow (v = w))$$

$$\sigma_2 = \forall u \forall v (R(u, v) \Rightarrow \exists w S(u, w))$$

Definition of the Chase

Let σ be $\forall \mathbf{x}(A \Rightarrow C)$ where A is a conjunction of atoms, Q be a CQ.

Definition (The Chase)

For a homomorphism $\theta : A \rightarrow Q$, we write $Q \xrightarrow{\sigma, \theta} Q'$ where Q' is:

- If σ is a TGD with $C \equiv \exists \mathbf{y} B$, then $Q' \stackrel{\text{def}}{=} Q \wedge \theta(B)$.
- If σ is an EGD with $C \equiv (x_i = x_j)$, then $Q' \stackrel{\text{def}}{=} Q[x_j/x_i]$.

Example

$$Q(x) = R(x, y) \wedge A(y) \wedge R(x, z) \wedge B(z)$$

$$\sigma_1 = \forall u \forall v \forall w (R(u, v) \wedge R(u, w) \Rightarrow (v = w))$$

$$\sigma_2 = \forall u \forall v (R(u, v) \Rightarrow \exists w S(u, w))$$

Chase Q with σ_1 , $\theta_1 : (u, v, w) \mapsto (x, y, z)$.

$$Q \xrightarrow{\sigma_1, \theta_1} ?$$

Definition of the Chase

Let σ be $\forall \mathbf{x}(A \Rightarrow C)$ where A is a conjunction of atoms, Q be a CQ.

Definition (The Chase)

For a homomorphism $\theta : A \rightarrow Q$, we write $Q \xrightarrow{\sigma, \theta} Q'$ where Q' is:

- If σ is a TGD with $C \equiv \exists \mathbf{y} B$, then $Q' \stackrel{\text{def}}{=} Q \wedge \theta(B)$.
- If σ is an EGD with $C \equiv (x_i = x_j)$, then $Q' \stackrel{\text{def}}{=} Q[x_j/x_i]$.

Example

$$Q(x) = R(x, y) \wedge A(y) \wedge R(x, z) \wedge B(z)$$

$$\sigma_1 = \forall u \forall v \forall w (R(u, v) \wedge R(u, w) \Rightarrow (v = w))$$

$$\sigma_2 = \forall u \forall v (R(u, v) \Rightarrow \exists w S(u, w))$$

Chase Q with $\sigma_1, \theta_1 : (u, v, w) \mapsto (x, y, z)$.

$$Q \xrightarrow{\sigma_1, \theta_1} R(x, y) \wedge A(y) \wedge B(y)$$

Definition of the Chase

Let σ be $\forall \mathbf{x}(A \Rightarrow C)$ where A is a conjunction of atoms, Q be a CQ.

Definition (The Chase)

For a homomorphism $\theta : A \rightarrow Q$, we write $Q \xrightarrow{\sigma, \theta} Q'$ where Q' is:

- If σ is a TGD with $C \equiv \exists \mathbf{y} B$, then $Q' \stackrel{\text{def}}{=} Q \wedge \theta(B)$.
- If σ is an EGD with $C \equiv (x_i = x_j)$, then $Q' \stackrel{\text{def}}{=} Q[x_j/x_i]$.

Example

$$Q(x) = R(x, y) \wedge A(y) \wedge R(x, z) \wedge B(z)$$

$$\sigma_1 = \forall u \forall v \forall w (R(u, v) \wedge R(u, w) \Rightarrow (v = w))$$

$$\sigma_2 = \forall u \forall v (R(u, v) \Rightarrow \exists w S(u, w))$$

Chase Q with σ_1 , $\theta_1 : (u, v, w) \mapsto (x, y, z)$.

$$Q \xrightarrow{\sigma_1, \theta_1}$$

$$R(x, y) \wedge A(y) \wedge B(y)$$

Chase the result with σ_2 , $\theta_2 : (u, v) \mapsto (x, y)$.

$$Q' \xrightarrow{\sigma_2, \theta_2} ?$$

Definition of the Chase

Let σ be $\forall \mathbf{x}(A \Rightarrow C)$ where A is a conjunction of atoms, Q be a CQ.

Definition (The Chase)

For a homomorphism $\theta : A \rightarrow Q$, we write $Q \xrightarrow{\sigma, \theta} Q'$ where Q' is:

- If σ is a TGD with $C \equiv \exists \mathbf{y} B$, then $Q' \stackrel{\text{def}}{=} Q \wedge \theta(B)$.
- If σ is an EGD with $C \equiv (x_i = x_j)$, then $Q' \stackrel{\text{def}}{=} Q[x_j/x_i]$.

Example

$$Q(x) = R(x, y) \wedge A(y) \wedge R(x, z) \wedge B(z)$$

$$\sigma_1 = \forall u \forall v \forall w (R(u, v) \wedge R(u, w) \Rightarrow (v = w))$$

$$\sigma_2 = \forall u \forall v (R(u, v) \Rightarrow \exists w S(u, w))$$

Chase Q with σ_1 , $\theta_1 : (u, v, w) \mapsto (x, y, z)$.

$$Q \xrightarrow{\sigma_1, \theta_1} R(x, y) \wedge A(y) \wedge B(y)$$

Chase the result with σ_2 , $\theta_2 : (u, v) \mapsto (x, y)$.

$$Q' \xrightarrow{\sigma_2, \theta_2} R(x, y) \wedge A(y) \wedge B(y) \wedge S(x, w)$$

When the Chase Goes Wrong

- Given a set Σ of GD, we can repeatedly apply the chase:

$$Q \xrightarrow{\sigma_1, \theta_1} Q_1 \xrightarrow{\sigma_2, \theta_2} Q_2 \dots$$

¹The book doesn't consider constants; need to add this to allow constants.

When the Chase Goes Wrong

- Given a set Σ of GD, we can repeatedly apply the chase:

$$Q \xrightarrow{\sigma_1, \theta_1} Q_1 \xrightarrow{\sigma_2, \theta_2} Q_2 \dots$$

- In general, this may not terminate:

$$\sigma = \forall x \forall y (R(x, y) \rightarrow \exists z R(y, z)) \qquad Q() = R(u_0, u_1)$$

$$R(u_0, u_1) \rightarrow (R(u_0, u_1) \wedge R(u_1, u_2)) \rightarrow (R(u_0, u_1) \wedge R(u_1, u_2) \wedge R(u_2, u_3)) \rightarrow \dots$$

¹The book doesn't consider constants; need to add this to allow constants.

When the Chase Goes Wrong

- Given a set Σ of GD, we can repeatedly apply the chase:

$$Q \xrightarrow{\sigma_1, \theta_1} Q_1 \xrightarrow{\sigma_2, \theta_2} Q_2 \dots$$

- In general, this may not terminate:

$$\sigma = \forall x \forall y (R(x, y) \rightarrow \exists z R(y, z)) \qquad Q() = R(u_0, u_1)$$

$$R(u_0, u_1) \rightarrow (R(u_0, u_1) \wedge R(u_1, u_2)) \rightarrow (R(u_0, u_1) \wedge R(u_1, u_2) \wedge R(u_2, u_3)) \rightarrow \dots$$

- Fact:** if all TGDs are *full* (i.e. no \exists) then any chase terminates.

¹The book doesn't consider constants; need to add this to allow constants.

When the Chase Goes Wrong

- Given a set Σ of GD, we can repeatedly apply the chase:

$$Q \xrightarrow{\sigma_1, \theta_1} Q_1 \xrightarrow{\sigma_2, \theta_2} Q_2 \dots$$

- In general, this may not terminate:

$$\begin{aligned} \sigma &= \forall x \forall y (R(x, y) \rightarrow \exists z R(y, z)) & Q() &= R(u_0, u_1) \\ R(u_0, u_1) &\rightarrow (R(u_0, u_1) \wedge R(u_1, u_2)) \rightarrow (R(u_0, u_1) \wedge R(u_1, u_2) \wedge R(u_2, u_3)) \rightarrow \dots \end{aligned}$$

- Fact:** if all TGDs are *full* (i.e. no \exists) then any chase terminates.
- In general, the chase may also fail:
 $\sigma = \forall x \forall y \forall z (R(x, y) \wedge R(x, z) \Rightarrow (y = z)) \quad Q() = R(x, 33) \wedge R(x, 55)$
 $Q \rightarrow \text{fail}.$

¹The book doesn't consider constants; need to add this to allow constants.

When the Chase Goes Wrong

- Given a set Σ of GD, we can repeatedly apply the chase:

$$Q \xrightarrow{\sigma_1, \theta_1} Q_1 \xrightarrow{\sigma_2, \theta_2} Q_2 \dots$$

- In general, this may not terminate:

$$\begin{aligned} \sigma &= \forall x \forall y (R(x, y) \rightarrow \exists z R(y, z)) & Q() &= R(u_0, u_1) \\ R(u_0, u_1) &\rightarrow (R(u_0, u_1) \wedge R(u_1, u_2)) \rightarrow (R(u_0, u_1) \wedge R(u_1, u_2) \wedge R(u_2, u_3)) \rightarrow \dots \end{aligned}$$

- Fact:** if all TGDs are *full* (i.e. no \exists) then any chase terminates.
- In general, the chase may also fail:
 $\sigma = \forall x \forall y \forall z (R(x, y) \wedge R(x, z) \Rightarrow (y = z)) \quad Q() = R(x, 33) \wedge R(x, 55)$
 $Q \rightarrow \text{fail}.$
- Fact:** if Σ does not contain EGDs, then the chase never fails.

¹The book doesn't consider constants; need to add this to allow constants.

When the Chase Goes Wrong

- Given a set Σ of GD, we can repeatedly apply the chase:

$$Q \xrightarrow{\sigma_1, \theta_1} Q_1 \xrightarrow{\sigma_2, \theta_2} Q_2 \dots$$

- In general, this may not terminate:

$$\begin{aligned} \sigma &= \forall x \forall y (R(x, y) \rightarrow \exists z R(y, z)) & Q() &= R(u_0, u_1) \\ R(u_0, u_1) &\rightarrow (R(u_0, u_1) \wedge R(u_1, u_2)) \rightarrow (R(u_0, u_1) \wedge R(u_1, u_2) \wedge R(u_2, u_3)) \rightarrow \dots \end{aligned}$$

- Fact:** if all TGDs are *full* (i.e. no \exists) then any chase terminates.
- In general, the chase may also fail:
 $\sigma = \forall x \forall y \forall z (R(x, y) \wedge R(x, z) \Rightarrow (y = z)) \quad Q() = R(x, 33) \wedge R(x, 55)$
 $Q \rightarrow \text{fail}.$
- Fact:** if Σ does not contain EGDs, then the chase never fails.
- Theorem** [Abiteboul et al., 1995, Theorem 8.4.18]: if Σ consists of full TGDs and EDGs (i.e. no \exists) and the chase succeeds¹ then all terminating chases end in the same query, denoted $\text{Chase}(Q)$.

¹The book doesn't consider constants; need to add this to allow constants.

Soundness

Theorem (Soundness Theorem)

Let $\sigma = \boxed{\forall \mathbf{x}(A \Rightarrow C)}$ be a GD. If $Q \xrightarrow{\sigma, \theta} Q_1$ then $\sigma \models Q \subseteq Q_1$.

Soundness

Theorem (Soundness Theorem)

Let $\sigma = \boxed{\forall \mathbf{x}(A \Rightarrow C)}$ be a GD. If $Q \xrightarrow{\sigma, \theta} Q_1$ then $\sigma \models Q \subseteq Q_1$.

Proof Assume Q is a Boolean query. Let \mathbf{D} be s.t. $\mathbf{D} \models \sigma$, $Q(\mathbf{D}) = \text{true}$.

Soundness

Theorem (Soundness Theorem)

Let $\sigma = \boxed{\forall \mathbf{x}(A \Rightarrow C)}$ be a GD. If $Q \xrightarrow{\sigma, \theta} Q_1$ then $\sigma \models Q \subseteq Q_1$.

Proof Assume Q is a Boolean query. Let \mathbf{D} be s.t. $\mathbf{D} \models \sigma$, $Q(\mathbf{D}) = \text{true}$.

Then $\exists \varphi : Q \rightarrow \mathbf{D}$. Compose it with θ : $\varphi \circ \theta = A \xrightarrow{\theta} Q \xrightarrow{\varphi} \mathbf{D}$

Soundness

Theorem (Soundness Theorem)

Let $\sigma = \boxed{\forall \mathbf{x}(A \Rightarrow C)}$ be a GD. If $Q \xrightarrow{\sigma, \theta} Q_1$ then $\sigma \models Q \subseteq Q_1$.

Proof Assume Q is a Boolean query. Let \mathbf{D} be s.t. $\mathbf{D} \models \sigma$, $Q(\mathbf{D}) = \text{true}$.

Then $\exists \varphi : Q \rightarrow \mathbf{D}$. Compose it with θ : $\varphi \circ \theta = A \xrightarrow{\theta} Q \xrightarrow{\varphi} \mathbf{D}$

Case 1: σ is a TGD $\boxed{\forall \mathbf{x}(A \Rightarrow \exists \mathbf{y}B)}$ Then $Q_1 \stackrel{\text{def}}{=} Q \wedge \theta(B)$.

Soundness

Theorem (Soundness Theorem)

Let $\sigma = \boxed{\forall \mathbf{x}(A \Rightarrow C)}$ be a GD. If $Q \xrightarrow{\sigma, \theta} Q_1$ then $\sigma \models Q \subseteq Q_1$.

Proof Assume Q is a Boolean query. Let \mathbf{D} be s.t. $\mathbf{D} \models \sigma$, $Q(\mathbf{D}) = \text{true}$.

Then $\exists \varphi : Q \rightarrow \mathbf{D}$. Compose it with θ : $\varphi \circ \theta = A \xrightarrow{\theta} Q \xrightarrow{\varphi} \mathbf{D}$

Case 1: σ is a TGD $\boxed{\forall \mathbf{x}(A \Rightarrow \exists \mathbf{y}B)}$ Then $Q_1 \stackrel{\text{def}}{=} Q \wedge \theta(B)$.

$\varphi \circ \theta(A) \subseteq \mathbf{D}$ and $\mathbf{D} \models \sigma$ implies $\varphi \circ \theta$ extends to a homomorphism $B \rightarrow \mathbf{D}$ that factors as $B \xrightarrow{\theta} Q_1 \rightarrow \mathbf{D}$, thus $Q_1(\mathbf{D}) = \text{true}$.

Soundness

Theorem (Soundness Theorem)

Let $\sigma = \boxed{\forall \mathbf{x}(A \Rightarrow C)}$ be a GD. If $Q \xrightarrow{\sigma, \theta} Q_1$ then $\sigma \models Q \subseteq Q_1$.

Proof Assume Q is a Boolean query. Let \mathbf{D} be s.t. $\mathbf{D} \models \sigma$, $Q(\mathbf{D}) = \text{true}$.

Then $\exists \varphi : Q \rightarrow \mathbf{D}$. Compose it with θ : $\varphi \circ \theta = A \xrightarrow{\theta} Q \xrightarrow{\varphi} \mathbf{D}$

Case 1: σ is a TGD $\boxed{\forall \mathbf{x}(A \Rightarrow \exists \mathbf{y}B)}$ Then $Q_1 \stackrel{\text{def}}{=} Q \wedge \theta(B)$.

$\varphi \circ \theta(A) \subseteq \mathbf{D}$ and $\mathbf{D} \models \sigma$ implies $\varphi \circ \theta$ extends to a homomorphism $B \rightarrow \mathbf{D}$ that factors as $B \xrightarrow{\theta} Q_1 \rightarrow \mathbf{D}$, thus $Q_1(\mathbf{D}) = \text{true}$.

Case 2: is an EGD $\boxed{\forall \mathbf{x}(A \Rightarrow (x_i = x_j))}$ In class.

Chase for Query Containment

We want to check $\Sigma \models Q \subseteq Q'$

- Simple (but important) observation. If $Q \rightarrow Q_1$ then $Q_1 \subseteq Q$ (unconditioned). **Why?**
- The Soundness Theorem proves $\Sigma \models Q \subseteq Q_1$.
- To check $\Sigma \models Q \subseteq Q'$, repeatedly chase Q :
$$Q \rightarrow Q_1 \rightarrow Q_2 \rightarrow \dots$$
$$\dots \subseteq Q_2 \subseteq Q_1 \subseteq Q \text{ (unconditioned)}$$
- If $Q_m \subseteq Q'$ (unconditioned) for some $m \geq 0$, then $\Sigma \models Q \subseteq Q'$ **why?**

Chase for Query Equivalence

To check equivalence $\Sigma \models Q \equiv Q'$, we need to chase both Q and Q' :

$$Q \rightarrow Q_1 \rightarrow Q_2 \rightarrow \dots$$

$$Q' \rightarrow Q'_1 \rightarrow Q'_2 \rightarrow \dots$$

If $Q_m \equiv Q'_n$ for some m, n , then $\Sigma \models Q \equiv Q'$

Chase and Backchase

[Popa et al., 2000]

Semantics optimization of Q under constraints Σ .

Assume Σ has only full TGDs and EGDs.

Chase Chase Q to completion: $Q \xrightarrow{*} \text{Chase}(Q)$.

Backchase Go in reverse $\text{Chase}(Q) \leftarrow Q'_1 \leftarrow Q'_2 \leftarrow \dots$

There are multiple choices for the backchase: this is an optimization problem.

Example: Using Physical Access Structures

Relation $R(k, x, y)$, key k , index $I(k, x)$ on $R.x$

Want to optimize $Q(y) = R(k, 55, y)$ to $Q'(y) = R(k, x, y) \wedge I(k, 55)$

Example: Using Physical Access Structures

Relation $R(k, x, y)$, key k , index $I(k, x)$ on $R.x$

Want to optimize $Q(y) = R(k, 55, y)$ to $Q'(y) = R(k, x, y) \wedge I(k, 55)$

FD σ_0 : $\forall k, x_1, x_2, y_1, y_2 (R(k, x_1, y_1) \wedge R(k, x_2, y_2) \Rightarrow (x_1 = x_2))$

IND1: σ_1 : $\forall k, x, y (R(k, x, y) \Rightarrow I(k, x))$

IND2: σ_2 : $\forall k I(k, x) \rightarrow \exists y R(k, x, y)$

Example: Using Physical Access Structures

Relation $R(k, x, y)$, key k , index $I(k, x)$ on $R.x$

Want to optimize $Q(y) = R(k, 55, y)$ to $Q'(y) = R(k, x, y) \wedge I(k, 55)$

FD σ_0 : $\forall k, x_1, x_2, y_1, y_2 (R(k, x_1, y_1) \wedge R(k, x_2, y_2) \Rightarrow (x_1 = x_2))$

IND1: σ_1 : $\forall k, x, y (R(k, x, y) \Rightarrow I(k, x))$

IND2: σ_2 : $\forall k I(k, x) \rightarrow \exists y R(k, x, y)$

$$\begin{aligned}
 Q &\equiv R(k, 55, y) \xrightarrow{\sigma_1} R(k, 55, y) \wedge I(k, 55) \equiv \text{Chase}(Q). \\
 Q' &\equiv R(k, x, y) \wedge I(k, 55) \xrightarrow{\sigma_2} R(k, x, y) \wedge R(k, 55, y') \wedge I(k, 55) \\
 &\quad \xrightarrow{\sigma_0} R(k, 55, y) \wedge I(k, 55) \equiv \text{Chase}(Q')
 \end{aligned}$$

$\text{Chase}(Q) = \text{Chase}(Q')$, implies $\Sigma \models Q \equiv Q'$.

Example: Using Physical Access Structures

Relation $R(k, x, y)$, key k , index $I(k, x)$ on $R.x$

Want to optimize $Q(y) = R(k, 55, y)$ to $Q'(y) = R(k, x, y) \wedge I(k, 55)$

FD $\sigma_0: \forall k, x_1, x_2, y_1, y_2 (R(k, x_1, y_1) \wedge R(k, x_2, y_2) \Rightarrow (x_1 = x_2))$

IND1: $\sigma_1: \forall k, x, y (R(k, x, y) \Rightarrow I(k, x))$

IND2: $\sigma_2: \forall k I(k, x) \rightarrow \exists y R(k, x, y)$

$$\begin{aligned} Q &\equiv R(k, 55, y) \xrightarrow{\sigma_1} R(k, 55, y) \wedge I(k, 55) \equiv \text{Chase}(Q). \\ Q' &\equiv R(k, x, y) \wedge I(k, 55) \xrightarrow{\sigma_2} R(k, x, y) \wedge R(k, 55, y') \wedge I(k, 55) \\ &\xrightarrow{\sigma_0} R(k, 55, y) \wedge I(k, 55) \equiv \text{Chase}(Q') \end{aligned}$$

$\text{Chase}(Q) = \text{Chase}(Q')$, implies $\Sigma \models Q \equiv Q'$.

Given Q , chase/Backchase computes $\text{Chase}(Q)$ the searches for Q' :

$$Q \xrightarrow{\sigma_1} \xrightarrow{\sigma_0} \xrightarrow{\sigma_2} Q'$$

Summary

- Constraints are restricted sentences in FO.
- The implication problem: elegant theory because it's a special case of logical implication.
- Chase: simple, yet fundamental technique. Egg and Egglog use chase. Termination is undecidable in general. Understanding termination is a major open problem.



Abiteboul, S., Hull, R., and Vianu, V. (1995).

Foundations of Databases.

Addison-Wesley.



Chakravarthy, U. S., Grant, J., and Minker, J. (1990).

Logic-based approach to semantic query optimization.

ACM Trans. Database Syst., 15(2):162–207.



King, J. J. (1981).

QUIST: A system for semantic query optimization in relational databases.

In *Very Large Data Bases, 7th International Conference, September 9-11, 1981, Cannes, France, Proceedings*, pages 510–517. IEEE Computer Society.



Popa, L., Deutsch, A., Sahuguet, A., and Tannen, V. (2000).

A chase too far?

In Chen, W., Naughton, J. F., and Bernstein, P. A., editors, *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*, pages 273–284. ACM.