Application Patterns of Projection/Forgetting

Christoph Wernhard

Technische Universität Dresden

Interpolation: From Proofs to Applications (iPRA 2014)

Vienna, 17 July 2014

Introduction

We assume a classical logic setting where projection and forgetting are available as **second-order operators that can be nested**

It allows to define concepts such as:

- Literal projection, literal forgetting
- Globally strongest necessary and weakest sufficient condition
- Definability and definientia

A variety of applications can be rendered with these:

- **View-based query processing**
  - Query rewriting
  - Characterizing definientia in formula classes
- **Knowledge base modularization**
  - Conservative theory extension
- **"Non-standard inferences"**
  - "Formula matching"
- **Non-monotonic reasoning and logic programming**
  - Stable and partial stable model semantics
  - Abduction w.r.t. these semantics

Classical Logic + Second-Order Operators

- We start with an **underlying classical logic**, e.g., first-order or propositional
- It is extended by **second-order operators**, e.g., predicate quantification or Boolean quantification

$$\exists q\,(p \rightarrow q) \land (q \rightarrow r)$$

- The associated computation is **second-order operator elimination**: computing an equivalent formula without second-order operators

$$\exists q\,(p \rightarrow q) \land (q \rightarrow r) \;\equiv\; p \rightarrow r.$$

Forgetting, Projection, Uniform Interpolants

- **Further second-order operators** can be defined in terms of predicate quantification
- An operator for **forgetting** can be seen as syntax for iterated existential predicate quantification:

$$\mathrm{forgetAboutPredicates}_{\{p,q\}}(F) \equiv \exists p\, \exists q\, F$$

- Elimination of $\mathrm{forgetAboutPredicates}$ is often called **computation of forgetting**
- Forgetting about all predicates **except** those explicitly specified is often called **projection** [Darwiche 01]

$$\mathrm{projectOntoPredicates}_{\{p,q\}}(F) \equiv \mathrm{forgetAboutPredicates}_{\textsc{AllPredicates} \setminus \{p,q\}}(F)$$

- Elimination of $\mathrm{projectOntoPredicates}$ is often called **computation of a uniform interpolant**
- Here we handle projection and forgetting **symmetrically as second-order operators**

Scopes as Parameters of Second-Order Operators

- The introduced second-order operators have a set of predicates as parameter

  We generalize this to a **set of ground literals**, called **scope**

- A scope can express different effects on **positive** and **negative** predicate occurrences

  Our basic second-order operators are now **literal projection** and **literal forgetting**:

Let $F = (p \rightarrow q) \land (q \rightarrow r)$

$$\mathrm{forget}_{\{\neg q\}}(F) \;\equiv\; \mathrm{project}_{\{p,q,r,\neg p,\neg r\}}(F) \;\equiv\; (p \rightarrow q) \land (p \rightarrow r)$$

[Lang* 03, W 08]

An interpretation is a set of ground literals, containing each ground atom either positively or negatively.

$I \models \mathrm{project}_S(F)$ iff$_{\text{def}}$ There exists a J s.t. $J \models F$ and $J \cap S \subseteq I$.

$$\mathrm{forget}_S(F) \stackrel{\text{def}}{=} \mathrm{project}_{\textsc{AllGroundLiterals} \setminus S}(F).$$

Notation for "in Scope"

- That F is **"in scope"** S is written as

$$F \Subset S$$

Let $F = p \lor \neg q \lor (r \land \neg r)$

$$\begin{array}{rcl} F & \Subset & \{p, \neg q\} \\ F & \Subset & \{p, q, r, s, \neg p, \neg q, \neg r, \neg s\} \\ F & \not\Subset & \{p\} \end{array}$$

$F \Subset S$ iff$_{\text{def}}$ $F \equiv \text{project}_S(F)$.

Globally Strongest Necessary and Weakest Sufficient Condition

- The **globally strongest necessary condition** of G on S within F is

  the strongest $X \Subset S$ s.th. $(F \land G) \models X$

  It can be expressed by a second-order operator

  $$\text{gsnc}_{\{p\}}((q \rightarrow p),\, q) \;\equiv\; p$$

- The **globally weakest sufficient condition** of G on S within F is

  the weakest $X \Subset S$ s.th. $(F \land X) \models G$

  It can be expressed by a second-order operator

  $$\text{gwsc}_{\{p\}}((p \rightarrow q),\, q) \;\equiv\; p$$

- The analog concepts in [Lin 01] are not unique modulo equivalence. See also [Doherty* 01, W 12]

Let $\overline{S}$ denote the set of the complements of the members of scope S.

$$\text{gsnc}_S(F, G) \quad \stackrel{\text{def}}{=} \quad \text{project}_S(F \land G).$$
$$\text{gwsc}_S(F, G) \quad \stackrel{\text{def}}{=} \quad \neg\text{project}_{\overline{S}}(F \land \neg G).$$

Definition, Definability

- A **definition of G in terms of S within F** is a formula $(G \leftrightarrow X)$ such that
  1. $X \Subset S$, and
  2. $F \models G \leftrightarrow X$

  G is the **definiendum**, X is the **definiens**

  Note: If F is a sentence, then $F \models G(\mathbf{x}) \leftrightarrow X(\mathbf{x})$ iff $F \models \forall \mathbf{x}(G(\mathbf{x}) \leftrightarrow X(\mathbf{x}))$

Let $F = (p \leftrightarrow q \wedge r) \wedge (q \to r)$

$(p \leftrightarrow q \wedge r)$	is a definition of p in terms of $\{q, r\}$ within F
$(p \leftrightarrow q)$	is a definition of p in terms of $\{q, r\}$ within F

- Existence of a definition is called **definability**

p is definable in terms of $\{q, r\}$ within F
p is definable in terms of $\{q\}$ within F
p is not definable in terms of $\{r\}$ within F

- This is a **semantic** characterization, aka implicit definability

Definition, Definability in Terms of Second-Order Operators

- **Definientia** are exactly those formulas in the scope that are **between the GSNC and the GWSC**

Let $F = (p \leftrightarrow q \land r) \land (q \rightarrow r)$

$$\begin{aligned} \mathrm{gsnc}_{\{q,r\}}(F,p) &\equiv \mathrm{project}_{\{q,r\}}(F \land p) &&\equiv q \land r \\ \mathrm{gwsc}_{\{q,r\}}(F,p) &\equiv \neg\mathrm{project}_{\{\neg q,\neg r\}}(F \land \neg p) &&\equiv q \end{aligned}$$

- **Definability** holds iff **the GSNC entails the GWSC**

$$\begin{array}{lcccccl} \mathrm{gsnc}_{\{q,r\}}(F,p) & \equiv & q \land r & \models & q & \equiv & \mathrm{gwsc}_{\{q,r\}}(F,p) \\ \mathrm{gsnc}_{\{q\}}(F,p) & \equiv & q & \models & q & \equiv & \mathrm{gwsc}_{\{q\}}(F,p) \\ \mathrm{gsnc}_{\{r\}}(F,p) & \equiv & r & \not\models & \bot & \equiv & \mathrm{gwsc}_{\{r\}}(F,p) \end{array}$$

- In case of definability, **the GSNC and GWSC provide the strongest and weakest definientia**

$\textsc{isDefinition}(X, G, S, F)$ iff$_{\mathrm{def}}$ $X \Subset S$ and $\mathrm{gsnc}_S(F, G) \models X \models \mathrm{gwsc}_S(F, G)$.

$\textsc{isDefinable}(G, S, F)$ iff$_{\mathrm{def}}$ $\mathrm{gsnc}_S(F, G) \models \mathrm{gwsc}_S(F, G)$.

View-Based Query Rewriting – Exact Views

[Halevy 01, Calvanese* 07, Marx 07, Nash* 10, Bárány* 13, W 14a]

- Given:

D	**"database scope"**	$\{a, \neg a\}$
U	**"view scope"**	$\{p, \neg p, q, \neg q\}$
$V \Subset D \cup U$	**"view specification"**	$(p \leftrightarrow a) \land (q \leftrightarrow a)$
$Q \Subset D$	**"query"**	a

- The **"view extension"** of V wrt. **"database"** $DB \Subset D$ is $\mathrm{project}_U(DB \land V)$

$$\mathrm{project}_U(a \land V) \;\equiv\; p \land q \qquad\qquad \mathrm{project}_U(\neg a \land V) \;\equiv\; \neg p \land \neg q$$

- "Queries to view extensions can be evaluated particularly well"

  The objective is to find an **"exact rewriting"** $R \Subset U$ s.t. for all $DB \Subset D$:

$$\mathrm{project}_U(DB \land V) \models R \text{ iff } DB \models Q$$

- Assume that all $R \Subset U$ are **uniquely definable** in terms of D within V

$$\mathrm{gsnc}_D(V, p) \equiv a \equiv \mathrm{gwsc}_D(V, p)$$

- **Then R is an exact rewriting iff R is a definiens of Q i.t.o. U within V**

$$\mathrm{gsnc}_U(V, Q) \;\equiv\; (p \land q) \begin{array}{c} \models\; p \;\models \\ \models\; q \;\models \end{array} (p \lor q) \;\equiv\; \mathrm{gwsc}_U(V, Q)$$

View-Based Query Rewriting – "Split Rewriting"

[W 14a], related to [Borgida* 10, Franconi* 13]

- Given: D **"database scope"**
  U **"view scope"**
  $V \Subset D \cup U$ **"view specification"**
  $Q \Subset D \cup U$ **"query"**

- The idea is to rewrite a $Q \Subset D \cup U$ to a $R \Subset D$ that can be evaluated by the "database system"

- The objective is to find a **"split rewriting"** $R \Subset D$ s.t. for all $DB \Subset D$:

$$DB \models R \text{ iff } DB \land V \models Q$$

- **R is a split rewriting iff** $R \equiv \mathbf{gwsc}_D(V, Q)$

View-Based Query Rewriting – Further Issues

- Investigation of **"determinacy" w.r.t. formula classes**
  [Segoufin and Vianu 05, Marx 07, Nash* 10, Bárány* 13]

  For Q, V in particular formula classes:

  - is the existence of an exact rewriting (definability) decidable?
  - what formula class contains all exact rewritings?

Definientia in Formula Classes

[W 14b]

- So far, we considered definientia in terms of a vocabulary

  Question: Can we apply second-order operators also to characterize definientia in efficiently processable formula classes?

- Yes, for the class of formulas that are equivalent to a **conjunction of atoms**

- This class excludes disjunction and negation and can thus be used to **encode other syntactic conditions on the meta level**

e.g., a Krom formula as a conjunction of atoms like $\mathrm{clause}(p, \neg q)$

$I \models \mathrm{project}_S(F)$ iff$_{\mathrm{def}}$ There exists a J s.t. $J \models F$ and $J \cap S \subseteq I$.

$I \models \mathrm{diff}_S(F)$ iff$_{\mathrm{def}}$ There exists a J s.t. $J \models F$ and $J \cap S \not\subseteq I$.

$\mathrm{glb}(F) \stackrel{\mathrm{def}}{=} \mathrm{circ}_{\mathrm{NEG}}(\neg \mathrm{diff}_{\mathrm{NEG}}(F))$.

$\mathrm{fhub}(F) \stackrel{\mathrm{def}}{=} \mathrm{project}_{\mathrm{POS}}(\mathrm{glb}(F)) \land \mathrm{project}_{\mathrm{NEG}}(F)$.

$\textsc{isCA-Definable}(G, S, F)$ iff $\mathrm{glb}(\mathrm{gsnc}_{S \cap \mathrm{POS}}(F, G)) \models \mathrm{gwsc}_{S \cap \mathrm{POS}}(F, G)$.

If $\textsc{isCA-Definable}(G, S, F)$, then

$\textsc{isCA-Definiens}(\mathrm{fhub}(\mathrm{gsnc}_{S \cap \mathrm{POS}}(F, G)), G, S, F)$.

Conservative Extensions Underlying Knowledge Base Modularization

[Ghilardi* 06, Cuenca Grau* 08]

	Adding G does not "damage my ontology" F	
iff	"All knowledge about the vocabulary of F that is expressed by $(F \wedge G)$ is expressed by F alone"	
iff	$(F \wedge G)$ is a **conservative extension** of F	
iff	G is **conservative** within F	[W 14a]
iff	G imports F in a **safe** way	[Cuenca Grau* 08]
iff	$F \models \text{project}_{\text{vocab}(F)}(F \wedge G)$	
iff	$F \equiv \text{project}_{\text{vocab}(F)}(F \wedge G)$	

"Formula Matching"

- **Concept matching modulo equivalence** is a non-standard inference in description logics [Borgida and McGuinness 96, Baader* 99],
- Here for arbitrary formulas but with single-variable patterns

Given:	F	**Background formula**	$\top$
	G	**Formula**	$p \leftrightarrow q$
	H	**Pattern: formula with special atom x**	$(p \land q) \lor x$

- Objective: Find a **"matching formula"** X such that

$$F \models G \leftrightarrow H[x \mapsto X]$$

$$\top \models (p \leftrightarrow q) \leftrightarrow ((p \land q) \lor x)$$
$$\top \models (p \leftrightarrow q) \leftrightarrow ((p \land q) \lor (\neg p \land \neg q))$$

- **There are two second-order formulas M_1 and M_2 such that solutions are exactly the X s.th. $M_1 \models X \models M_2$**

Basic characterization of X: $\models \forall x F \land (x \leftrightarrow X) \to (G \leftrightarrow H)$

This is equivalent to: $\exists x\, F \land \neg x \land \neg(G \leftrightarrow H) \models X$

and $X \models \forall x\, F \land x \to (G \leftrightarrow H)$

Stable Model Semantics for Logic Programming

Let $F = p \land (q \leftarrow p \land \neg r)$

It has three models: $\{p, q, r\}, \{p, q, \neg r\}, \{p, \neg q, r\}$

Considered as logic program it has a single **stable model**: $\{p, q\}$

- **Logic programs can be represented by classical formulas, where second-order operators associate logic programming semantics** [W 10]

$$\mathrm{stable}(p \land (q \leftarrow p \land \neg r^1)) \equiv (p \land q \land \neg r)$$

  A “replica” of the vocabulary, identified by the 1 superscript, is used for predicate occurrences under negation as failure

- $\mathrm{stable}(F) \stackrel{\text{def}}{=} \mathrm{rename}_{1 \mapsto 0}(\mathrm{circ}_{(0 \cap \mathrm{POS}) \cup 1}(F))$
  1. minimize undecorated predicates, while keeping 1 predicates fixed
  2. rename the 1 predicates to their undecorated correspondents

- The stable operator renders the characterization of the stable model semantics in terms of circumscription from [Lin 91]

- By combination with an encoding from [Janhunen* 06], a similar operator can render the 3-valued **partial stable model semantics**

Abduction with the Stable Model Semantics

[Kakas* 98, Lin and You 02, W 13a]

- Given: F **background** $(wet \leftarrow shower) \wedge (wet \leftarrow rain \wedge \neg umbrella^1) \wedge (umbrella \leftarrow forecastRain)$

  G **observation** wet

  S **abducibles** $\{shower, rain, forecastRain, \neg shower, \neg rain, \neg forecastRain\}$

- In classical logic, an **explanation** is an $X \Subset S$ s.th. $(F \wedge X) \models G$

  The weakest explanation is $\mathrm{gwsc}_S(F, G)$

  $\mathrm{gwsc}_S(F, G) \equiv shower$

- For the **stable model semantics**, a **"factual" explanation** is a conjunction of literals $X \Subset S$ s.th.

  $$\mathrm{stable}_S(F \wedge X) \models G$$

  stable_S effects that atoms occurring in S are subjected to the **open-world** assumption (passed as "fixed" to the circumscription)

The minimal factual explanations for the example are
$shower$ and $(rains \wedge \neg forecastRain)$

Abduction with the Stable Model Semantics (2)

[W 13a]

For the stable model semantics, a "factual" explanation is a conjunction of literals $X \Subset S$ s.th.

$$\mathrm{stable}_S(F \land X) \models G$$

- The **minimal factual explanations** are the **prime implicants** of

$$\mathrm{gwsc}_{S\cap 0}(\mathrm{stable}_S(F), G)$$

  - $S \cap 0$ specifies the undecorated literals in S
  - The underlying justification is that for $H \Subset S \cup \overline{S}$ it holds that

$$\mathrm{stable}_S(F \land H) \equiv \mathrm{stable}_S(F) \land H$$

$$\mathrm{gwsc}_{S\cap 0}(\mathrm{stable}_S(F), G) \equiv \neg\mathrm{project}_{\overline{S}\cap 0}(\mathrm{stable}_S(F) \land \neg G)$$

Abduction with 3-Valued Logic Programming Semantics

[W 13a]

- Abduction can be analogously characterized with the **GWSC** for
  - the **well founded semantics**
  - the **partial stable model semantics**
- For the partial stable model semantics, this seems so far the only thorough formalization of abduction
- Unlike the well-founded semantics, the partial stable model semantics allows to obtain **explanations for the undefinedness of observations**

Background: The barber shaves all
    males who do not shave themselves
  The barber shaves the barber
    if the barber has been sentenced to shave himself

Observation: "The barber shaves the barber" is undefined

Explanation: The barber is male and
  has not been sentenced to shave himself

Conclusion – Towards Practice

- ToyElim [W 13b] is a Prolog-based **prototype system** which supports to define second-order operators as outlined and is useful for small experiments

- Relevant general processing techniques include:
  - **second-order quantifier elimination methods** based on first-order logic [Gabbay and Ohlbach 92, Doherty* 97]
  - recent advances in **uniform interpolation for description logics** [Ghilardi* 06, Konev* 09, Koopmann and Schmidt 13]
  - progress in **SAT pre- and inprocessing** [Eén and Biere 05, Heule* 10, Manthey* 13]

- General agenda: Investigate processing of the particular **formula patterns** in which combinations of second-order operators are used in applications

  Consider these patterns also for **restricted argument formulas**

Conclusion – Classical Logic + Second-Order Operators

- Provides an **integrating view on a variety of applications** in areas such as
  - view-based query processing
  - knowledge base modularization
  - many “non-standard” inferences
  - non-monotonic reasoning and logic programming
  - abductive reasoning
- **Operators can be nested and combined**
- **New operators can be defined in terms of other ones**
- **Operators let instructive relationships become evident**
- **Operators seems useful for mechanization**
- **Second-order operators shift techniques from a theoretical background to a mechanizable and user accessible formalization**

References

[Baader and Küsters 98] Baader, F. and Küsters, R. (1998).
Computing the least common subsumer and the most specific concept in the presence of cyclic $\mathcal{ALN}$-concept descriptions.
In *KI-98*, volume 1504 of *LNCS*, pages 129–140. Springer.

[Baader* 99] Baader, F., Küsters, R., Borgida, A., and McGuinness, D. (1999).
Matching in description logics.
JLC, 9(3):411–447.

[Bárány* 13] Bárány, V., Benedikt, M., and ten Cate, B. (2013).
Rewriting guarded negation queries.
In *Mathematical Foundations of Computer Science 2013*, volume 8087 of *LNCS*, pages 98–110. Springer.

[Borgida* 10] Borgida, A., de Bruijn, J., Franconi, E., Seylan, I., Straccia, U., Toman, D., and Weddell, G. (2010).
On finding query rewritings under expressive constraints.
In *Proc. 18th Italian Symp. on Advanced Database Systems, SEBD 2010*.

[Borgida and McGuinness 96] Borgida, A. and McGuinness, D. L. (1996).
Asking queries about frames.
In *Proc. 5th Int. Conf. on Knowledge Rep. and Reasoning, KR'96*, pages 340–349. Morgan Kaufmann.

[Calvanese* 07] Calvanese, D., Giacomo, G. D., Lenzerini, M., and Vardi, M. Y. (2007).
View-based query processing: On the relationship between rewriting, answering and losslessness.
TCS, 371(3):169–182.

[Cuenca Grau* 08] Cuenca Grau, B., Horrocks, I., Kazakov, Y., and Sattler, U. (2008).
Modular reuse of ontologies: Theory and practice.
JAIR, 31:273–318.

[Darwiche 01] Darwiche, A. (2001).
Decomposable negation normal form.
JACM, 48(4):608–647.

[Dechter and Pearl, 1992] Dechter, R. and Pearl, J. (1992).
Structure identification in relational data.
AI, 58:237–270.

[Doherty* 97] Doherty, P., Łukaszewicz, W., and Szałas, A. (1997).
Computing circumscription revisited: A reduction algorithm.
JAR, 18(3):297–338.

[Doherty* 01] Doherty, P., Łukaszewicz, W., and Szałas, A. (2001).
Computing strongest necessary and weakest sufficient conditions of first-order formulas.
In *IJCAI-01*, pages 145–151. Morgan Kaufmann.

[Eén and Biere 05] Eén, N. and Biere, A. (2005).
Effective preprocessing in SAT through variable and clause elimination.
In *SAT 2005*, volume 3569 of *LNCS*, pages 61–75.

[Ferraris* 11] Ferraris, P., Lee, J., and Lifschitz, V. (2011).
Stable models and circumscription.
AI, 175(1):236–263.

[Franconi* 13] Franconi, E., Kerhet, V., and Ngo, N. (2013).
Exact query reformulation over databases with first-order and description logics ontologies.
JAIR, 48:885–922.

[Gabbay and Ohlbach 92] Gabbay, D. and Ohlbach, H. J. (1992).
Quantifier elimination in second-order predicate logic.
In *KR'92*, pages 425–435. Morgan Kaufmann.

[Gabbay* 08] Gabbay, D. M., A., R., Schmidt, and Szałas, A. (2008).
Second-Order Quantifier Elimination: Foundations, Computational Aspects and Applications.
College Publications, London.

[Ghilardi* 06] Ghilardi, S., Lutz, C., and Wolter, F. (2006).
Did I damage my ontology? A case for conservative extensions in description logics.
In *KR 2006*, pages 187–197. AAAI Press.

[Halevy 01] Halevy, A. Y. (2001).
Answering queries using views: a survey.
The VLDB Journal, 10(4):270–294.

[Heule* 10] Heule, M., Järvisalo, M., and Biere, A. (2010).
Clause elimination procedures for CNF formulas.
In *LPAR-17*, volume 6397 of *LNCS*, pages 357–371. Springer.

[Janhunen* 06] Janhunen, T., Niemelä, I., Seipel, D., Simons, P., and You, J.-H. (2006).
Unfolding partiality and disjunctions in stable model semantics.
ACM Trans. Comput. Log., 7(1):1–37.

[Kakas* 98] Kakas, A. C., Kowalski, R. A., and Toni, F. (1998).
The role of abduction in logic programming.
In D. M. Gabbay et al., editor, *Handbook of Logic in Artif. Int.*, volume 5, pages 235–324. Oxford University Press.

[Konev* 09] Konev, B., Walther, D., and Wolter, F. (2009).
Forgetting and uniform interpolation in large-scale description logic terminologies.
In *IJCAI-09*, pages 830–835. AAAI Press.

[Koopmann and Schmidt 13] Koopmann, P. and Schmidt, R. A. (2013).
Uniform interpolation of $\mathcal{ALC}$-ontologies using fixpoints.
In *FroCoS 2013*, volume 8152 of *LNCS (LNAI)*, pages 87–102. Springer.

[Lang* 03] Lang, J., Liberatore, P., and Marquis, P. (2003).
Propositional independence – formula-variable independence and forgetting.
JAIR, 18:391–443.

[Lifschitz 94] Lifschitz, V. (1994).
Circumscription.
In Gabbay, D. M., Hogger, C. J., and Robinson, J. A., editors, *Handbook of Logic in Artif. Int. and Logic Prog.*, volume 3, pages 298–352. Oxford University Press.

[Lin 91] Lin, F. (1991).
A Study of Nonmonotonic Reasoning.
PhD thesis, Stanford Univ.

[Lin 01] Lin, F. (2001).
On strongest necessary and weakest sufficient conditions.
AI, 128(1–2):143–159.

[Lin and You 02] Lin, F. and You, J.-H. (2002).
Abduction in logic programming: A new definition and an abductive procedure based on rewriting.
AI, 140(1/2):175–205.

[Manthey* 13] Manthey, N., Philipp, T., and Wernhard, C. (2013).
Soundness of inprocessing in clause sharing SAT solvers.
In *SAT 2013*, volume 7962 of *LNCS*, pages 22–39. Springer.

[Marx 07] Marx, M. (2007).
Queries determined by views: pack your views.
In *PODS '07*, pages 23–30. ACM.

[McCarthy 80] McCarthy, J. (1980).
Circumscription – a form of non-monotonic reasoning.
AI, 13:27–39.

[Nash* 10] Nash, A., Segoufin, L., and Vianu, V. (2010).
Views and queries: Determinacy and rewriting.
TODS, 35(3).

[Przymusinski 90] Przymusinski, T. (1990).
Well-founded semantics coincides with three-valued stable semantics.
Fundamenta Informaticae, 13(4):445–464.

[Segoufin and Vianu 05] Segoufin, L. and Vianu, V. (2005).
Views and queries: Determinacy and rewriting.
In *PODS 2005*, pages 49–60.

[Selman and Kautz, 1991] Selman, B. and Kautz, H. A. (1991).
Knowledge compilation using Horn approximations.
In *AAAI-91*, pages 904–909. AAAI Press.

[Tarski 35] Tarski, A. (1935).
Einige methologische Untersuchungen zur Definierbarkeit der Begriffe.
Erkenntnis, 5:80–100.

[W 08] Wernhard, C. (2008).
Literal projection for first-order logic.
In *JELIA 08*, volume 5293 of *LNCS (LNAI)*, pages 389–402. Springer.

[W 10] Wernhard, C. (2010).
Circumscription and projection as primitives of logic programming.
In *Tech. Comm. ICLP'10*, volume 7 of *LIPIcs*, pages 202–211.

[W 12] Wernhard, C. (2012).
Projection and scope-determined circumscription.
JSC, 47(9):1089–1108.

[W 13a] Wernhard, C. (2013a).
Abduction in logic programming as second-order quantifier elimination.
In *FroCoS 2013*, volume 8152 of *LNCS (LNAI)*, pages 103–119. Springer.

[W 13b] Wernhard, C. (2013b).
Computing with logic as operator elimination: The ToyElim system.
In *INAP 2011/WLP 2011*, volume 7773 of *LNCS (LNAI)*. Springer.

[W 14a] Wernhard, C. (2014a).
Expressing view-based query processing and related approaches with second-order operators.
Technical Report KRR 14–02, TU Dresden.
`http://www.wv.inf.tu-dresden.de/Publications/2014/report-2014-02.pdf`.

[W 14b] Wernhard, C. (2014b).
Second-order characterizations of definientia in formula classes.
Technical Report KRR 14–03, TU Dresden.
http:
//www.wv.inf.tu-dresden.de/Publications/2014/report-2014-03.pdf.

Appendix

Notes on the Relationship to Craig Interpolation (Addendtum to Slide 9)

- [Tarski 35]: **Definability w.r.t. first-order formulas can be reduced to first-order validity**

$$\mathrm{gsnc}_S(F,G) \models \mathrm{gwsc}_S(F,G) \text{ iff } F \land G \models F' \rightarrow G'$$

- The **interpolants** X in S such that

$$F \land G \models X \models F' \rightarrow G'$$

  are definitions

- The extreme definitions GSNC and GWSC are obtained as **uniform interpolants** – if the predicate elimination succeeds

More precisely: Let S specify a set of predicates. Let F, G be first-order. Let F', G' be F, G after systematically replacing all predicates not in S with new symbols. Then

$$\mathrm{gsnc}_S(F,G) \models \mathrm{gwsc}_S(F,G) \text{ iff } F \land G \models F' \rightarrow G'.$$

If $X \Subset S$, then $F \land G \models X$ iff $\mathrm{gsnc}_S(F,G) \models X$.

If $X \Subset S$, then $X \models F' \rightarrow G'$ iff $X \models \mathrm{gwsc}_S(F,G)$.

Notes About Unique Definability (Mentioned on Slides 10 and 14)

- If $S \equiv \overline{S}$, then a formula that is definable in terms of S within F is **uniquely definable** iff

$$\models \text{project}_S(F)$$

- **Conservativeness** with respect to all formulas in a scope and **definability** in terms of that scope together imply **unique definability**

See [W 14a]

Proof Sketch for Slide 10

Assumptions: $R \Subset U$, $Q \Subset D$

R is an exact rewriting of Q w.r.t. V
iff $\forall DB \Subset D : \text{project}_U(V \land DB) \models R \text{ iff } DB \models Q$
iff $\forall DB \Subset D : V \land DB \models R \text{ iff } DB \models Q$ — since $R \Subset U$
iff $\forall DB \Subset D : DB \models \neg V \lor R \text{ iff } DB \models Q$
iff $\text{project}_{\overline{D}}(V \land \neg R) \equiv \text{project}_{\overline{D}}(\neg Q)$
iff $\text{gwsc}_D(V, R) \equiv Q$. — since $Q \Subset D$

Assume A1: Unique definability of all $R \Subset U$ i.t.o. D within V, i.e.
$\forall R \Subset U : \text{gsnc}_D(V, R) \equiv \text{gwsc}_D(V, R)$.

$\text{gwsc}_D(V, R) \models Q$
iff $\text{gsnc}_D(V, R) \models Q$ — by assumption A1
iff $V \land R \models Q$ — since $Q \Subset D$
iff $V \land \neg Q \models \neg R$
iff $\text{project}_{\overline{U}}(V \land \neg Q) \models \neg R$ — since $R \Subset D$
iff $R \models \text{gwsc}_U(V, Q)$. — Note: for "sound views" just this direction is relevant

$Q \models \text{gwsc}_D(V, R)$
iff $\text{project}_{\overline{D}}(V \land \neg R) \models \neg Q$
iff $V \land \neg R \models \neg Q$ — since $Q \Subset D$
iff $V \land Q \models R$
iff $\text{gsnc}_U(V, Q) \models R$. — since $R \Subset U$

See [W 14a]

Proof Sketch for Slide 11

Assumption: $R \Subset D$

$$
\begin{array}{ll}
 & R \text{ is a split rewriting of } Q \text{ w.r.t. } V \text{ and } D \\
\text{iff} & \forall DB \Subset D : DB \models R \text{ iff } DB \wedge V \models Q \\
\text{iff} & \forall DB \Subset D : DB \models R \text{ iff } DB \models \neg V \vee Q \\
\text{iff} & \mathrm{project}_{\overline{D}}(\neg R) \equiv \mathrm{project}_{\overline{D}}(V \wedge \neg Q) \\
\text{iff} & \neg R \equiv \mathrm{project}_{\overline{D}}(V \wedge \neg Q) \qquad \text{since } R \Subset D \\
\text{iff} & R \equiv \mathrm{gwsc}_{D}(V, Q).
\end{array}
$$

- Note: The GWSC is the **only** solution!
- This seems to supersede material in [W 14a]

Proof Sketch for Slide 15

$$
\begin{aligned}
&\models \forall x\, F \wedge (x \leftrightarrow X) \rightarrow (G \leftrightarrow H) \\
\text{iff } &\models (\forall x\, F \wedge x \wedge X \rightarrow (G \leftrightarrow H)) \wedge (\forall x\, F \wedge \neg x \wedge \neg X \rightarrow (G \leftrightarrow H)) \\
\text{iff } &\models (X \rightarrow (\forall x\, F \wedge x \rightarrow (G \leftrightarrow H))) \wedge ((\exists x\, F \wedge \neg x \wedge \neg(G \leftrightarrow H)) \rightarrow X) \\
\text{iff } & X \models \forall x\, F \wedge x \rightarrow (G \leftrightarrow H) \text{ and } \exists x\, F \wedge \neg x \wedge \neg(G \leftrightarrow H) \models X.
\end{aligned}
$$