Access Predicates – Examples from the Literature

Revision: March 28, 2019; Rendered: May 25, 2021

Some examples of solutions to view-based query processing and query optimization tasks from the literature. Makes use of scratch_forgetting and scratch_definientia. Formalized with the *PIE* system.

Contents

1 Examples from Benedikt, ten Cate and Tsamoura: Generating Low-Cost Plans from Proofs

These examples stem from [BtCT14]. The numbering of examples refers to that paper. The representation here is different from the paper – (it seem new and is called here “SV-modeling”). Also the methods and solutions are different.

1.1 Example 1

Note: ia in the background formula represents that the name (“smith” in the paper) is given. However ia is actually not used to compute the interpolant.

exbct_1$_a$

Defined as

$$\begin{array}{ll}\forall noe\,(\mathsf{i}e \rightarrow (\mathsf{profinfo_a}(n,o,e) \leftrightarrow \mathsf{profinfo}(n,o,e))) & \wedge \\ \forall noe\,(\mathsf{profinfo}(n,o,e) \rightarrow \mathsf{i}n \wedge \mathsf{i}o \wedge \mathsf{i}e). & \end{array}$$

exbct_1$_b$

Defined as

$$\begin{array}{ll}\forall ne\,(\mathsf{udirect}(n,e) \rightarrow \mathsf{i}n \wedge \mathsf{i}e) & \wedge \\ \forall noe\,(\mathsf{profinfo}(n,o,e) \rightarrow \mathsf{udirect}(n,e)). & \end{array}$$

exbtc_1$_c$

Defined as

$$\begin{array}{l}\mathit{definiens}(\mathsf{profinfo}(\mathsf{a},\mathsf{b},\mathsf{c}), \\ \qquad \mathit{exbct\_1_a} \wedge \mathit{exbct\_1_b} \wedge \mathsf{ia}, \\ \qquad [\mathsf{profinfo_a}, \mathsf{udirect}]).\end{array}$$

Input: *exbtc_1$_c$*.
Result of interpolation:

$$\mathsf{udirect}(\mathsf{a},\mathsf{c}) \wedge \mathsf{profinfo_a}(\mathsf{a},\mathsf{b},\mathsf{c}).$$

In the following formula the query is expressed more accurately with existential midle argument, as described in Example 3 of [BtCT14].

exbtc_1$_d$

Defined as

$$\begin{array}{l}\mathit{definiens}(\exists o\, \mathsf{profinfo}(\mathsf{a},o,\mathsf{c}), \\ \qquad \mathit{exbct\_1_a} \wedge \mathit{exbct\_1_b} \wedge \mathsf{ia}, \\ \qquad [\mathsf{profinfo_a}, \mathsf{udirect}]).\end{array}$$

Input: *exbtc_1$_d$*.
Result of interpolation:

$$\exists x\,(\mathsf{udirect}(\mathsf{a},\mathsf{c}) \wedge \mathsf{profinfo_a}(\mathsf{a},x,\mathsf{c})).$$

1.2 Variant of Example 1

This is the variant of Example 1 from [BtCT14, p. 101 Left Column]. It leads to different interpolants, obtained with the enum_ips option. Here the first 3 interpolants are shown.

$exbtc\_1_e$

Defined as

$$
\begin{aligned}
\mathit{definiens}(\mathsf{profinfo}(\mathsf{a},\mathsf{b},\mathsf{c}),\ & \\
& exbct\_1_a & \land \\
& \forall ne\,(\mathsf{udirect}_1(n,e) \rightarrow \mathsf{i}n \land \mathsf{i}e) & \land \\
& \forall noe\,(\mathsf{profinfo}(n,o,e) \rightarrow \mathsf{udirect}_1(n,e)) & \land \\
& \forall ne\,(\mathsf{udirect}_2(n,e) \rightarrow \mathsf{i}n \land \mathsf{i}e) & \land \\
& \forall noe\,(\mathsf{profinfo}(n,o,e) \rightarrow \mathsf{udirect}_2(n,e)) & \land \\
& \mathsf{ia}, & \\
& [\mathsf{profinfo_a}, \mathsf{udirect}_1, \mathsf{udirect}_2]).
\end{aligned}
$$

Input: $exbtc\_1_e$.
Result of interpolation:

$$\mathsf{udirect}_1(\mathsf{a},\mathsf{c}) \land \mathsf{profinfo_a}(\mathsf{a},\mathsf{b},\mathsf{c}).$$

Input: $exbtc\_1_e$.
Result of interpolation:

$$\mathsf{udirect}_2(\mathsf{a},\mathsf{c}) \land \mathsf{profinfo_a}(\mathsf{a},\mathsf{b},\mathsf{c}).$$

Input: $exbtc\_1_e$.
Result of interpolation:

$$\mathsf{udirect}_1(\mathsf{a},\mathsf{c}) \land \mathsf{udirect}_2(\mathsf{a},\mathsf{c}) \land \mathsf{profinfo_a}(\mathsf{a},\mathsf{b},\mathsf{c}).$$

1.3 Example 4

Note that we have the id as last argument of profinfo, following the natural language description of Example 1 of the paper. In the formal version, of Example 4 in the paper the id is the first argument. The $i(a)$ has been dropped here.

$exbtc\_4_a$

Defined as

$$
\begin{aligned}
\mathit{definiens}(\exists c\, \mathsf{profinfo}(\mathsf{a},\mathsf{o},c),\ & \\
& exbct\_1_a \land exbct\_1_b, \\
& [\mathsf{profinfo_a}, \mathsf{udirect}]).
\end{aligned}
$$

Input: $exbtc\_4_a$.
Result of interpolation:

$$\exists x\,(\mathsf{udirect}(\mathsf{a},x) \land \mathsf{profinfo_a}(\mathsf{a},\mathsf{o},x)).$$

1.4 Example 5

Note that we have the id as last argument of profinfo, following the natural language description of Example 1 of the paper. In the formal version, of Examples 4 and 5 in the paper the id is the first argument.

Here we need $i(b)$ because, as described in the paper, the access to profinfo requires all arguments bound, where only the first and last can be bound at all by udirect. Perhaps this is a bug in the paper.

The point of the example seems that the method of the paper involves all three udirect accesses (in some order), but it is hard to see why this is useful, when the query could be answered by accessing just one of them.

---

exbtc_5a

Defined as

$$
\begin{array}{ll}
\mathit{definiens}(\exists c\, \mathsf{profinfo}(\mathsf{a}, \mathsf{b}, c), & \\
\quad \forall noe\, (\mathsf{i}n \land \mathsf{i}o \land \mathsf{i}e \rightarrow (\mathsf{profinfo_a}(n, o, e) \leftrightarrow \mathsf{profinfo}(n, o, e))) & \land \\
\quad \forall noe\, (\mathsf{profinfo}(n, o, e) \rightarrow \mathsf{i}n \land \mathsf{i}o \land \mathsf{i}e) & \land \\
\quad \mathit{exbct\_1_a} & \land \\
\quad \forall ne\, (\mathsf{udirect_1}(n, e) \rightarrow \mathsf{i}n \land \mathsf{i}e) & \land \\
\quad \forall noe\, (\mathsf{profinfo}(n, o, e) \rightarrow \mathsf{udirect_1}(n, e)) & \land \\
\quad \forall ne\, (\mathsf{udirect_2}(n, e) \rightarrow \mathsf{i}n \land \mathsf{i}e) & \land \\
\quad \forall noe\, (\mathsf{profinfo}(n, o, e) \rightarrow \mathsf{udirect_2}(n, e)) & \land \\
\quad \forall ne\, (\mathsf{udirect_3}(n, e) \rightarrow \mathsf{i}n \land \mathsf{i}e) & \land \\
\quad \forall noe\, (\mathsf{profinfo}(n, o, e) \rightarrow \mathsf{udirect_3}(n, e)) & \land \\
\quad \mathsf{ib}, & \\
\quad [\mathsf{profinfo_a}, \mathsf{udirect_1}, \mathsf{udirect_2}, \mathsf{udirect_3}]). &
\end{array}
$$

Input: *exbtc_5a*.
Result of interpolation:

$$\exists x\, (\mathsf{udirect_1}(\mathsf{a}, x) \land \mathsf{profinfo_a}(\mathsf{a}, \mathsf{b}, x)).$$

Input: *exbtc_5a*.
Result of interpolation:

$$\exists x\, (\mathsf{udirect_2}(\mathsf{a}, x) \land \mathsf{profinfo_a}(\mathsf{a}, \mathsf{b}, x)).$$

1.5 Example 2

There seems a bug in the paper: Actually a referential constraint from direct2 into direct1 w.r.t. n, a is required here. The paper suggests the reverse direction.

exbtc_2_schema

Defined as

$$
\begin{array}{ll}
\forall nau\,(\mathsf{i}n \land \mathsf{i}u \to (\mathsf{direct1_a}(n,a,u) \leftrightarrow \mathsf{direct_1}(n,a,u))) & \land \\
\forall nau\,(\mathsf{direct_1}(n,a,u) & \to \\
\quad \mathsf{i}n \land \mathsf{i}a \land \mathsf{i}u) & \land \\
\forall nau\,(\mathsf{direct_1}(n,a,u) \to \mathsf{ids}(u)) & \land \\
\forall u\,(\mathsf{ids}(u) \to \mathsf{i}u) & \land \\
\forall nap\,(\mathsf{i}n \land \mathsf{i}a \to (\mathsf{direct2_a}(n,a,p) \leftrightarrow \mathsf{direct_2}(n,a,p))) & \land \\
\forall nap\,(\mathsf{direct_2}(n,a,p) & \to \\
\quad \mathsf{i}n \land \mathsf{i}a \land \mathsf{i}p) & \land \\
\forall nap\,(\mathsf{direct_2}(n,a,p) \to \mathsf{names}(n)) & \land \\
\forall n\,(\mathsf{names}(n) \to \mathsf{i}n) & \land \\
\forall nap\,(\mathsf{direct_2}(n,a,p) \to \exists u\, \mathsf{direct_1}(n,a,u)). &
\end{array}
$$

exbtc_2$_a$

Defined as

$$
\begin{array}{l}
\mathit{definiens}(\exists na\, \mathsf{direct_2}(n,a,\mathsf{p}), \\
\qquad \mathit{exbtc\_2\_schema}, \\
\qquad [\mathsf{direct1_a}, \mathsf{direct2_a}, \mathsf{ids}, \mathsf{names}]).
\end{array}
$$

Input: *exbtc_2*$_a$.
Result of interpolation:

$$
\begin{array}{ll}
\exists x, y, z\,(\mathsf{ids}(z) & \land \\
\qquad \mathsf{names}(x) & \land \\
\qquad \mathsf{direct1_a}(x,y,z) & \land \\
\qquad \mathsf{direct2_a}(x,y,\mathsf{p})). &
\end{array}
$$

2 Examples from Toman and Wedell: Fundamentals of Physical Design and Query Compilation

These examples are from [TW11, Chapters 3 and 5].

2.1 Example 5.14

extw_514$_a$

Defined as

$$\begin{array}{ll}\forall xy\,(\mathsf{v_1}xy \leftrightarrow \exists uw\,(\mathsf{r}ux \land \mathsf{r}uw \land \mathsf{r}wy)) & \land\\ \forall xy\,(\mathsf{v_2}xy \leftrightarrow \exists uw\,(\mathsf{r}xu \land \mathsf{r}uw \land \mathsf{r}wy)) & \land\\ \forall xy\,(\mathsf{v_3}xy \leftrightarrow \exists u\,(\mathsf{r}xu \land \mathsf{r}uy)). & \end{array}$$

extw_514$_b$

Defined as

$$\begin{array}{l}\mathit{definiens}(\exists uvw\,(\mathsf{r}u\mathsf{x} \land \mathsf{r}uw \land \mathsf{r}wv \land \mathsf{r}v\mathsf{y}),\\ \qquad \mathit{extw\_514}_a,\\ \qquad [\mathsf{v_1}, \mathsf{v_2}, \mathsf{v_3}]).\end{array}$$

Input: *extw_514$_b$*.
Result of interpolation:

$$\exists z\,\forall u\,(\mathsf{v_1}(\mathsf{x}, z) \land (\mathsf{v_3}(u, z) \rightarrow \mathsf{v_2}(u, \mathsf{y}))).$$

Notes: The book shows this solution, but also another, longer formula which is then used as basis for plan generation. The longer formula seems not easily to obtain as alternative interpolant with CM prover.

extw_514_altsol

Defined as

$$\exists uv\,(\mathsf{v_1}\mathsf{x}u \land \mathsf{v_3}vu \land \mathsf{v_2}v\mathsf{y} \land \forall w\,(\lnot \mathsf{v_3}wu \lor \mathsf{v_2}w\mathsf{y})).$$

extw_514_query

Defined as

$$\exists uvw\,(\mathsf{r}u\mathsf{x} \land \mathsf{r}uw \land \mathsf{r}wv \land \mathsf{r}v\mathsf{y}).$$

extw_514_check_altsol$_1$

Defined as

$$\mathit{extw\_514}_a \rightarrow (\mathit{extw\_514\_altsol} \leftarrow \mathit{extw\_514\_query}).$$

$extw\_514\_check\_altsol_2$

Defined as

$$extw\_514_a \rightarrow (extw\_514\_altsol \rightarrow extw\_514\_query).$$

The next formula uses literal forgetting for $v3$. This seems hard for the CM prover (12 sec, 3 of them in the last depth 7). When literal forgetting is used to restrict polarities for all three access paths, i.e., $[v1-p, v2-p, v3-n]$, the CM prover does not succeed in a few minutes.

$extw\_514_c$

Defined as

$$\begin{array}{l} definiens\_lit(\exists uvw\,(\mathsf{r}u\mathsf{x} \land \mathsf{r}uw \land \mathsf{r}wv \land \mathsf{r}v\mathsf{y}),\\ \quad extw\_514_a,\\ \quad [\mathsf{v}_1\text{-}\mathsf{pn}, \mathsf{v}_2\text{-}\mathsf{pn}, \mathsf{v}_3\text{-}\mathsf{n}]).\end{array}$$

$extw\_514_d$

Defined as

$$\begin{array}{l} definiens\_lit\_lemma(\exists uvw\,(\mathsf{r}u\mathsf{x} \land \mathsf{r}uw \land \mathsf{r}wv \land \mathsf{r}v\mathsf{y}),\\ \quad extw\_514_a,\\ \quad [\mathsf{v}_1\text{-}\mathsf{pn}, \mathsf{v}_2\text{-}\mathsf{pn}, \mathsf{v}_3\text{-}\mathsf{n}]).\end{array}$$

$extw\_514_e$

Defined as

$$\begin{array}{l} definiens\_lit\_lemma(\exists uvw\,(\mathsf{r}u\mathsf{x} \land \mathsf{r}uw \land \mathsf{r}wv \land \mathsf{r}v\mathsf{y}),\\ \quad extw\_514_a,\\ \quad [\mathsf{v}_1\text{-}\mathsf{p}, \mathsf{v}_2\text{-}\mathsf{p}, \mathsf{v}_3\text{-}\mathsf{n}]).\end{array}$$

2.2 Examples 3.2, 3.4, 3.5

These examples are also discussed in the book in Examples 5.3 and 5.4.

The access paths have arguments *Salary*, *Number*, *Name*. The employee/3 relation has these arguments in the ordering *Number*, *Name Salary*.

We give the access paths here different number suffixes than in the book to utilize that lexically smaller predicates are preferred by the interpolant computation (the ordp option of the CM prover) and thus returned in earlier solutions.

NOTE: Currently the ordp option prefers solution with lexically smaller predicates, however for the price of possibly choosing a larger clause, which might introduce redundancies. The ordp option is not required if access patterns are disjoint.

---

extw_3_schema

Defined as

$$\begin{array}{ll}
\forall xyz\,(\mathsf{emp_array_3}(z,x,y) \leftrightarrow \mathsf{employee}(x,y,z)) & \land\\
\forall xyz\,(\mathsf{i}z \rightarrow (\mathsf{emp_array_2}(z,x,y) \leftrightarrow \mathsf{employee}(x,y,z))) & \land\\
\forall xyz\,(\mathsf{i}z \land \mathsf{i}x \rightarrow (\mathsf{emp_array_1}(z,x,y) \leftrightarrow \mathsf{employee}(x,y,z))) & \land\\
\forall xyz\,(\mathsf{employee}(x,y,z) \rightarrow \mathsf{i}x \land \mathsf{i}y \land \mathsf{i}z). &
\end{array}$$

---

extw₃₂

Defined as

$$\begin{array}{l}
\mathit{definiens}(\mathsf{employee}(\mathsf{x},\mathsf{y},\mathsf{z}),\\
\qquad \mathit{extw_3_schema},\\
\qquad [\mathsf{emp_array_1}, \mathsf{emp_array_2}, \mathsf{emp_array_3}]).
\end{array}$$

$extw_{34}$

Defined as

$$\begin{array}{l}
\mathit{definiens}(\mathsf{employee}(\mathsf{x},\mathsf{y},\mathsf{z}),\\
\qquad \mathit{extw_3_schema} \land \mathsf{iz},\\
\qquad [\mathsf{emp_array_1}, \mathsf{emp_array_2}, \mathsf{emp_array_3}]).
\end{array}$$

$extw_{35}$

Defined as

$$\begin{array}{l}
\mathit{definiens}(\mathsf{employee}(\mathsf{x},\mathsf{y},\mathsf{z}),\\
\qquad \mathit{extw_3_schema} \land \mathsf{ix} \land \mathsf{iz},\\
\qquad [\mathsf{emp_array_1}, \mathsf{emp_array_2}, \mathsf{emp_array_3}]).
\end{array}$$

Input: extw_{32}.
Result of interpolation:

$$\mathsf{emp\_array_3}(\mathsf{z},\mathsf{x},\mathsf{y}).$$

Input: extw_{34}.
Result of interpolation:

$$\mathsf{emp\_array_2}(\mathsf{z},\mathsf{x},\mathsf{y}).$$

Input: $extw_{35}$.
Result of interpolation:

$$\mathsf{emp\_array}_1(\mathsf{z}, \mathsf{x}, \mathsf{y}).$$

2.3 Alternate Modeling, Similar to "Option 3"

This is just similar to "Option 3" in the book, but we use employee numbers directly as IDs of employees. We map the schema to employee/3:

---

extw_3_o3_schema_addition

Defined as

$$\begin{array}{ll}
\forall x\,(\mathsf{emp}(x) \leftrightarrow \exists yz\,\mathsf{employee}(x, y, z)) & \land \\
\forall xy\,(\mathsf{name}(x, y) \leftrightarrow \exists z\,\mathsf{employee}(x, y, z)) & \land \\
\forall xy\,(\mathsf{salary}(x, y) \leftrightarrow \exists z\,\mathsf{employee}(x, z, y)). &
\end{array}$$

---

extw_o3₃₂

Defined as

$$\begin{array}{l}
\mathit{definiens}(\mathsf{emp}(\mathsf{x}) \land \mathsf{name}(\mathsf{x}, \mathsf{y}) \land \mathsf{salary}(\mathsf{x}, \mathsf{z}), \\
\quad \mathit{extw_3_schema} \land \mathit{extw_3_o3_schema_addition}, \\
\quad [\mathsf{emp_array}_1, \mathsf{emp_array}_2, \mathsf{emp_array}_3]).
\end{array}$$

---

extw_o3₃₄

Defined as

$$\begin{array}{l}
\mathit{definiens}(\mathsf{emp}(\mathsf{x}) \land \mathsf{name}(\mathsf{x}, \mathsf{y}) \land \mathsf{salary}(\mathsf{x}, \mathsf{z}), \\
\quad \mathit{extw_3_schema} \land \mathit{extw_3_o3_schema_addition} \land \mathsf{iz}, \\
\quad [\mathsf{emp_array}_1, \mathsf{emp_array}_2, \mathsf{emp_array}_3]).
\end{array}$$

---

extw_o3₃₅

Defined as

$$\begin{array}{l}
\mathit{definiens}(\mathsf{emp}(\mathsf{x}) \land \mathsf{name}(\mathsf{x}, \mathsf{y}) \land \mathsf{salary}(\mathsf{x}, \mathsf{z}), \\
\quad \mathit{extw_3_schema} \land \mathit{extw_3_o3_schema_addition} \land \mathsf{ix} \land \mathsf{iz}, \\
\quad [\mathsf{emp_array}_1, \mathsf{emp_array}_2, \mathsf{emp_array}_3]).
\end{array}$$

Input: $extw\_o3_{32}$.
Result of interpolation:

$$\exists u, v\,(\mathsf{emp\_array}_3(\mathsf{z}, \mathsf{x}, v) \land \mathsf{emp\_array}_3(u, \mathsf{x}, \mathsf{y})).$$

Input: $extw\_o3_{34}$.
Result of interpolation:

$$\exists u, v\,(\mathsf{emp\_array}_2(\mathsf{z}, \mathsf{x}, v) \wedge \mathsf{emp\_array}_3(u, \mathsf{x}, \mathsf{y})).$$

Input: $extw\_o3_{35}$.
Result of interpolation:

$$\exists u, v\,(\mathsf{emp\_array}_1(\mathsf{z}, \mathsf{x}, v) \wedge \mathsf{emp\_array}_3(u, \mathsf{x}, \mathsf{y})).$$

References

[BtCT14] Michael Benedikt, Balder ten Cate, and Efthymia Tsamoura. Generating low-cost plans from proofs. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS'14*, pages 200–211, 2014.

[TW11] David Toman and Grant Weddell. *Fundamentals of Physical Design and Query Compilation.* Synthesis Lectures on Data Management. Morgan and Claypool, 2011.

Index