Magic Sets

CS240B Notes

Notes based on Section 9.5 of *Advanced Database Systems*—Morgan Kaufmann, 1997

C. Zaniolo, March 2002

The Same-Generation Example

People are of the same generation if their parents are of the same generation

$?\texttt{sg}(\texttt{marc}, \texttt{Who}).$

$\texttt{sg}(\texttt{X}, \texttt{Y}) \leftarrow \quad \texttt{parent}(\texttt{XP}, \texttt{X}), \texttt{sg}(\texttt{XP}, \texttt{YP}), \texttt{parent}(\texttt{YP}, \texttt{Y}).$

$\texttt{sg}(\texttt{A}, \texttt{A}).$

This program cannot be computed in a bottom-up fashion because the exit rule is not safe.

Even if we make it safe by adding a goal such as $\texttt{people}(\texttt{A})$, bottom-up computation would be inefficient since all same-generation pairs are produced, while we only want those that have `marc` as their first component.

Top-Down Computation for same-generation

$?\texttt{sg}(\texttt{marc}, \texttt{Who}).$

$\texttt{sg}(\texttt{X}, \texttt{Y}) \leftarrow \quad \texttt{parent}(\texttt{XP}, \texttt{X}), \texttt{sg}(\texttt{XP}, \texttt{YP}), \texttt{parent}(\texttt{YP}, \texttt{Y}).$

$\texttt{sg}(\texttt{A}, \texttt{A}).$

- If $\texttt{parent}(\texttt{tom}, \texttt{marc})$ is in the database, the resolvent of the query goal with the first rule is

  $\leftarrow \texttt{parent}(\texttt{XP}, \texttt{marc}), \texttt{sg}(\texttt{XP}, \texttt{YP}), \texttt{parent}(\texttt{YP}, \texttt{Y}).$

  Then, by unifying the first goal with the fact $\texttt{parent}(\texttt{tom}, \texttt{marc})$, the new goal list becomes:

  $\leftarrow \texttt{sg}(\texttt{tom}, \texttt{YP}), \texttt{parent}(\texttt{YP}, \texttt{Y}).$

- The binding has been passed from the first argument in the head to the first argument in the body of the recursive predicate. $\texttt{X}$ and $\texttt{XP}$ are instantiated, while $\texttt{Y}$ and $\texttt{YP}$ remain unbound.

Only ancestors of marc are of interest

The $\mathtt{sg}$ rules are linear, but not left-linear or right-linear.

Say that the predicate $\mathtt{m.sg(X)}$ computes the ancestors of $\mathtt{marc}$.

We can add $\mathtt{m.sg(X)}$ to the exit rule to make it safe, and to the recursive rule to make it more selective:

$$?\mathtt{sg'(marc, Z)}.$$

$$\mathtt{sg'(X, X)} \leftarrow \mathtt{m.sg(X)}.$$
$$\mathtt{sg'(X, Y)} \leftarrow \mathtt{parent(XP, X), sg'(XP, YP), parent(YP, Y), m.sg(X)}.$$

$\mathtt{m.sg(X)}$ is called the *magic* predicate for $\mathtt{sg}$ and can be computed from the original program as follows:

$$\mathtt{m.sg(marc)}.$$
$$\mathtt{m.sg(XP)} \leftarrow \mathtt{m.sg(X), parent(XP, X)}.$$

Computing the Magic Predicate

$?\texttt{sg}(\texttt{marc}, \texttt{Who}).$

$\texttt{sg}(\texttt{X}, \texttt{Y}) \leftarrow \quad \texttt{parent}(\texttt{XP}, \texttt{X}), \texttt{sg}(\texttt{XP}, \texttt{YP}), \texttt{parent}(\texttt{YP}, \texttt{Y}).$

$\texttt{sg}(\texttt{A}, \texttt{A}).$

Binding analysis of the top-down behavior.
The first argument in the query: thus $\texttt{X}$ is bound and through goal $\texttt{parent}(\texttt{XP}, \texttt{X})$ the binding is passed to $\texttt{XP}$ in the recursive goal.
The variables $\texttt{Y}, \ \texttt{YP}$ remain unbound.
The rules for the magic predicates can be obtained by:
(1) using the query constant as the exit rule (a fact).
(2) using the top-down bound arguments and predicates for the exit rule—however head and tail must be reversed!

The Counting Method

"People who are of the same generation as `marc`". Is logically equivalent to:

1. Find the ancestors of `marc` and their levels, where `marc` is a zero-level ancestor of himself, his parents are first-generation (i.e., first-level) ancestors, his grandparents are second-generation ancestors, and so on. (This computation is performed by the predicate `sg_up`)
2. Switch to the computation of descendants
3. Perform the computation of descendants—while descreasing the level by one at each step
   This is performed by the predicate `sg_dwn` in
4. Check when you return to level 0 to find those who are of the same generation as `marc`.

The Counting Method—cont.

Find ancestors of marc, and then their descendants

$$\texttt{sg\_up}(0, \texttt{marc}).$$

$$\texttt{sg\_up}(\texttt{J}+1, \texttt{XP}) \leftarrow \texttt{parent}(\texttt{XP}, \texttt{X}), \texttt{sg\_up}(\texttt{J}, \texttt{X}).$$

$$\texttt{sg\_dwn}(\texttt{J}, \texttt{X}) \leftarrow \texttt{sg\_up}(\texttt{J}, \texttt{X}).$$

$$\texttt{dwn}(\texttt{J}-1, \texttt{Y}) \leftarrow \texttt{sg\_dwn}(\texttt{J}, \texttt{YP}), \texttt{parent}(\texttt{YP}, \texttt{Y}).$$

$$?\texttt{dwn}(0, \texttt{Z}).$$

The counting method: pros and cons

- The counting method is often more efficient than the magic-set method.
- However it is not as general: e.g. if we add the goal $\texttt{X} \neq \texttt{Y}$ to ensure that you to leave out `marc` from the people who are of the same generation as `marc`, then we must memorize.
- Cycles in the database will throw it into a loop–just as Prolog
- Many approaches to get methods that combine the strengths of magic and counting have been developed.

Supplementary Magic Sets

$$\begin{aligned}
&\mathtt{m.sg(marc)}.\\
&\mathtt{m.sg(XP)} \leftarrow && \mathtt{m.sg(X), parent(XP, X)}.\\
&\mathtt{spm.sg(X, XP)} \leftarrow && \mathtt{parent(XP, X), m.sg(X)}.\\
&\mathtt{sg'(X, X)} \leftarrow && \mathtt{m.sg(X)}.\\
&\mathtt{sg'(X, Y)} \leftarrow && \mathtt{sg'(XP, YP), spm.sg(X, XP), parent(YP, Y)}.\\
&?\mathtt{sg'(marc, Z)}.\\
\%&\mathtt{sg'(X, Y)} \leftarrow && \mathtt{parent(XP, X), sg'(XP, YP), parent(YP, Y), m.sg(X)}
\end{aligned}$$

In addition to the magic predicates, supplementary predicates are used to store the pairs bound-arguments-in-head/bound-arguments-in-recursive-goal.

Supplementary Magic Sets

The magic set method and supplementary magic set method are very similar–often the first term is used to refer to both methods.
The magic predicate and the supplementary magic predicate are normally written in a mutually recursive form.

$$\begin{aligned}
&\texttt{m.sg}(\texttt{marc}).\\
&\texttt{spm.sg}(\texttt{X},\texttt{XP}) \leftarrow \quad \texttt{m.sg}(\texttt{X}),\texttt{parent}(\texttt{XP},\texttt{X})\\
&\texttt{m.sg}(\texttt{XP}) \leftarrow \qquad\quad \texttt{spm.sg}(\texttt{X},\texttt{XP}).
\end{aligned}$$

Benefits of Memorizing

People who are of the same generation through common ancestors who are less than 12 *levels remote and always lived in the same state*

$$?\texttt{stsg}(\texttt{marc}, 12, \texttt{Z}).$$

$$\begin{aligned}\texttt{stsg}(\texttt{X},\texttt{K},\texttt{Y}) \leftarrow \quad & \texttt{parent}(\texttt{XP},\texttt{X}), \texttt{K} > 0, \texttt{KP} = \texttt{K} - 1,\\ & \texttt{born}(\texttt{X},\texttt{St}), \texttt{born}(\texttt{XP},\texttt{St}),\\ & \texttt{stsg}(\texttt{XP},\texttt{KP},\texttt{YP}),\\ & \texttt{parent}(\texttt{YP},\texttt{Y}).\end{aligned}$$

$$\texttt{stsg}(\texttt{X},\texttt{K},\texttt{X}).$$

Benefits of Memorizing—cont.

Since the first two arguments of `stsg` are bound. The supplementary magic method for this example is:

$$\texttt{m.stsg}(\texttt{marc}, 12).$$

$$\texttt{spm.stsg}(\texttt{X}, \texttt{K}, \texttt{XP}, \texttt{KP}) \leftarrow \texttt{m.stsg}(\texttt{X}, \texttt{K}), \texttt{parent}(\texttt{XP}, \texttt{X}), \texttt{K} > 0, \texttt{KP} = \texttt{K} - 1, \texttt{born}(\texttt{X}, \texttt{St}), \texttt{born}(\texttt{XP}, \texttt{St}).$$

$$\texttt{m.stsg}(\texttt{X}, \texttt{K}) \leftarrow \texttt{spm.stsg}(\texttt{X}, \texttt{K}, \texttt{XP}, \texttt{KP}).$$

$$\texttt{stsg}(\texttt{X}, \texttt{K}, \texttt{X}) \quad \texttt{m.stsg}(\texttt{X}, \texttt{K}).$$

$$\texttt{stsg}(\texttt{X}, \texttt{K}, \texttt{Y}) \leftarrow \texttt{stsg}(\texttt{XP}, \texttt{KP}, \texttt{YP}), \texttt{spm.stsg}(\texttt{X}, \texttt{K}, \texttt{XP}, \texttt{KP}), \texttt{parent}(\texttt{YP}, \texttt{Y}).$$

Supplementary Magic Sets

- Only those variables that are needed for the second fixpoint are stored in the supplementary magic relations: thus `St` is not included.
- The method of choice in many prototypes because of generality and robustness.
- the method works with cycles in the database
- Ability of storing one-way predicates, such as $\mathtt{X} = \mathtt{f}(\mathtt{Y}, \mathtt{Z}, \ldots)$.