Adding Recursion to SPJRUD

Jef Wijsen

May 10, 2019

Complexity

- An algorithm runs in $\mathcal{O}(f(n))$ time if there exists a constant k such that on inputs of sufficiently large size n, the algorithm terminates after at most $k \cdot f(n)$ steps.
- An algorithm runs in $\mathcal{O}(f(n))$ space if there exists a constant k such that on inputs of sufficiently large size n, the algorithm uses at most $k \cdot f(n)$ bits of auxiliary memory.
- A polytime algorithm runs in $\mathcal{O}(n^k)$ time for some constant k.
- A logspace algorithm runs in $\mathcal{O}(\log n)$ space.
- Explain $\mathbf{L} \subseteq \mathbf{P}$: with $k \cdot \log n$ bits, you can use at most $2^{k \cdot \log n} = n^k$ distinct auxiliary states.

Query Evaluation

For every fixed SPJRUD expression E, we define EVAL(E) as the following problem:

INPUT: A database $\mathcal{I}$ and a tuple t.

QUESTION: Does t belong to $[\![E]\!]^{\mathcal{I}}$?

Proposition

For every expression E in SPJRUD, there exists a logspace algorithm for the following problem:

Given a database $\mathcal{I}$, return $[\![E]\!]^{\mathcal{I}}$.

$\Longrightarrow$ EVAL(E) *is in* **L** *for every expression E in SPJRUD.*

Fixed Points

Let U be a finite set. A mapping $f : \mathcal{P}(U) \to \mathcal{P}(U)$ is

- inflationary (French: *inflationniste*) if for all $X \subseteq U$, $X \subseteq f(X)$;
- monotone if for all $X, Y \subseteq U$, $X \subseteq Y$ implies $f(X) \subseteq f(Y)$.

A set $X \subseteq U$ is a fixed point of f if $f(X) = X$.

Example

Let $U = \{a, b\}$ and f_1, f_2, f_3 as follows.

X	$f_1(X)$	$f_2(X)$	$f_3(X)$
$\emptyset$	$\{a, b\}$	$\emptyset$	$\{a, b\}$
$\{a\}$	$\{a\}$	$\{b\}$	$\{b\}$
$\{b\}$	$\{b\}$	$\{a\}$	$\{a\}$
$\{a, b\}$	$\{a, b\}$	$\{a, b\}$	$\emptyset$

Fixed Point Computation

Property

Define $X^0 := \emptyset$*, and for* $i = 0, 1, \ldots$*,* $X^{i+1} := f(X^i)$*.*

- *If* f *is inflationary or* f *is monotone, then for some* $n \leq |U|$*,* X^n *is a fixed point.*
- *Moreover, if* f *is monotone, then this fixed point* X^n *is included in every other fixed point of* f*. That is,* X^n *is the unique least fixed point of* f*.*

A Fixed Point Operator for SPJRUD

Let R and Δ be relation names s.t. $\text{sort}(R) = \text{sort}(\Delta) = \{A, B\}$.
Let

$$E := R \cup \pi_{AB}\left(\rho_{B \mapsto C}(R) \bowtie \rho_{A \mapsto C}(\Delta)\right).$$

Define f as the mapping s.t. for every relation X over $\{A, B\}$,

$$f(X) := [\![E]\!]^{\mathcal{I}_{\Delta \to X}}$$

Define $\Delta^0 := \emptyset$ and $\Delta^{i+1} := f(\Delta^i)$ for $i \geq 0$.

Questions

- Argue that f is both inflationary and monotone.
- Describe the fixed point reached by $(\Delta^i)_{i=0}^{\infty}$.

$\Longrightarrow$ New operator:

Syntax: $\mathbf{fp}_{\Delta:AB}(E)$

Semantics: $[\![\mathbf{fp}_{\Delta:AB}(E)]\!]^{\mathcal{I}}$ is the fixed point reached by $(\Delta^i)_{i=0}^{\infty}$.

Nesting is Allowed

Example

Let $\text{sort}(R) = \{A, B, C\}$.

$$\begin{aligned} E_1 &:= \mathbf{fp}_{\Delta:ABC}\left(R \cup \pi_{ABC}\left(\rho_{B\mapsto D}(R) \bowtie \rho_{A\mapsto D}(\Delta)\right)\right) \\ E_2 &:= \pi_{AB}(E_1) \\ E_3 &:= \mathbf{fp}_{\Delta':AB}\left(E_2 \cup \pi_{AB}\left(\rho_{B\mapsto C}(E_2) \bowtie \rho_{A\mapsto C}(\Delta')\right)\right) \end{aligned}$$

Example

Let $\text{sort}(R) = \{A\}$.

$$\mathbf{fp}_{\Delta:A}\left(\Delta \cup \left(R - \mathbf{fp}_{\Delta':A}\left(\Delta' \cup (R - \Delta)\right)\right)\right)$$

Problem: $(\Delta^i)_{i=0}^{\infty}$ May Reach No Fixed Point

Let $\mathrm{sort}(R) = \mathrm{sort}(\Delta)$.
Let
$$f(X) := [\![R - \Delta]\!]^{\mathcal{I}_{\Delta \to X}}.$$

Questions

- Does f have a fixed point for every database $\mathcal{I}$?
- Does f have a fixed point for some database $\mathcal{I}$?
- What if we replace R with an arbitrary SPJRUD expression of the same sort as Δ?

$$\Downarrow$$

Proposition

The following problem is undecidable: Given an expression E that uses Δ, does $\Delta^0, \Delta^1, \Delta^2, \ldots$ (as previously defined) reach a fixed point for every database $\mathcal{I}$?

Solution

Alike in *Bases de Données I*:

domain independence is an undecidable semantic property $\rightarrow$ safety is a decidable syntactic property

Proposition

Let $\mathbf{fp}_{\Delta:S}(E)$ *be syntactically well-defined.*
Let $\mathcal{I}$ *be any database, and* $f(X) := [\![E]\!]^{\mathcal{I}_{\Delta\to X}}$.
Then,

if all **fp**-*subexpressions*[1] *are of the form* $\mathbf{fp}_{\Delta':S'}(\Delta' \cup E')$	$\Longrightarrow$	f *is inflationary*

and

if for every **fp**-*subexpression* $\mathbf{fp}_{\Delta':S'}(E')$, *we have that* E' *is positive in* Δ'	$\Longrightarrow$	f *is monotone*

[1] Since an expression is a subexpression of itself, these conditions apply also to $\mathbf{fp}_{\Delta:S}(E)$ itself.

SPJRUD+FP

SPJRUD+FP extends SPJRUD with the **fp**-operator, but with the following syntactic restriction:
whenever you write $\mathbf{fp}_{\Delta:S}(E)$, it must be the case that either

- E is of the form $\Delta \cup E'$, or
- E is positive in Δ.

Moreover, avoid mixing up both forms in a same expression (because in database theory, it is common to separate **ifp** from **lfp**, which correspond, respectively, to the first and second syntactic form).

Proposition

For every expression E in SPJRUD+FP, there exists a polytime algorithm for the following problem:

Given a database $\mathcal{I}$, return $[\![E]\!]^{\mathcal{I}}$.

$\Longrightarrow$ EVAL(E) *is in* **P** *for every expression E in SPJRUD+FP.*

Fixed Point Operator in Relational Calculus

Syntax We add formulas of the form

$$[\mathbf{fp}_{\Delta:x_1,\ldots,x_k}(\varphi)](t_1,\ldots,t_k)$$

where

- Δ is a k-ary relation name;
- $x_1,\ldots,x_k$ are the free variables of φ; and
  $\Longrightarrow$ evaluating $\varphi(x_1,\ldots,x_k)$ on some database $\mathcal{I}_{\Delta\to\Delta^i}$ results in a k-ary relation $\Delta^{i+1} := \{(c_1,\ldots,c_k) \mid \mathcal{I}_{\Delta\to\Delta^i} \models \varphi(c_1,\ldots,c_k)\}$
- every t_i is a constant or a variable.

The free variables of $[\mathbf{fp}_{\Delta:x_1,\ldots,x_k}(\varphi)](t_1,\ldots,t_k)$ are the variables that occur in $t_1,\ldots,t_k$.

Semantics return all values for [the variables in] $(t_1,\ldots,t_k)$ that yield a tuple in the fixed point reached by $(\Delta^i)_{i=0}^{\infty}$ with $\Delta^0 = \emptyset$

Examples

- Transitive closure of a binary relation R.

$$\{\langle u, v\rangle \mid [\mathbf{fp}_{\Delta:x,y}\,(R(x,y) \lor \exists z\,(R(x,z) \land \Delta(z,y)))](u,v)\}$$

$\Longrightarrow$ all couples (u, v) in the transitive closure

- All nodes reachable from 0.

$$\{\langle v\rangle \mid [\mathbf{fp}_{\Delta:x,y}\,(R(x,y) \lor \exists z\,(R(x,z) \land \Delta(z,y)))](0,v)\}$$

- Is there a path from 0 to 4?

$$\{\langle\rangle \mid [\mathbf{fp}_{\Delta:x,y}\,(R(x,y) \lor \exists z\,(R(x,z) \land \Delta(z,y)))](0,4)\}$$

- All couples not in the transitive closure.

$$\{\langle u, v\rangle \mid \exists w\,(R(u,w) \lor R(w,u)) \land \exists w\,(R(v,w) \lor R(w,v)) \land \neg[\mathbf{fp}_{\Delta:x,y}\,(R(x,y) \lor \exists z\,(R(x,z) \land \Delta(z,y)))](u,v)\ \}$$

Example

Let R be ternary relation name with $\mathrm{sort}(R) = \{A, B, C\}$.
Let S be a unary relation name with $\mathrm{sort}(S) = \{A\}$.
An R-tuple $\{A : p, B : q, C : r\}$ encodes the propositional formula

$$p \wedge q \rightarrow r.$$

An S-tuple $\{A : p\}$ encodes that p has truth value **true**.

Which propositions r must be true in every model of the formulas in R, given the truth values in S?

$$\{r \mid [\mathbf{fp}_{\Delta:x}\,(S(x) \vee \exists p \exists q\,(R(p, q, x) \wedge \Delta(p) \wedge \Delta(q)))](r)\}$$

Syntactic Restrictions

$$[\mathbf{fp}_{\Delta:x_1,\ldots,x_k}(\varphi)](t_1,\ldots,t_k)$$

Question:

What syntactic restrictions on φ guarantee that

$$\emptyset = \Delta^0, \Delta^1, \Delta^2, \ldots$$

will reach a fixed point?

Exercise

Let R be a binary relation that encodes a directed graph.
Which vertices are in the answer of the following query?

$$\{z \mid \ [\mathbf{fp}_{\Delta:x}\,(\exists y\,(R(x,y) \vee R(y,x)) \wedge \forall y\,(R(y,x) \rightarrow \Delta(y)))](z) \\ \wedge \\ \exists x R(x,z)\ \}$$

Transitive Closure Logic

SPJRUD+TC adds a further restriction:

whenever you write $\mathbf{fp}_{\Delta:S}(E)$, it must be the case that sort$(E) = \vec{A}\vec{B}\vec{D}$ with $|\vec{A}| = |\vec{B}|$ and E computes, for every fixed $\vec{D}$-value $\vec{d}$, the transitive closure of the set of $(\vec{A}, \vec{B})$-values that occur with $\vec{d}$;

$\Longrightarrow$ if $\{\vec{A} : \vec{a}, \vec{B} : \vec{b}, \vec{D} : \vec{d}\}$ and $\{\vec{A} : \vec{b}, \vec{B} : \vec{c}, \vec{D} : \vec{d}\}$ are in the transitive closure, then so is $\{\vec{A} : \vec{a}, \vec{B} : \vec{c}, \vec{D} : \vec{d}\}$.

Note: separate transitive closure is computed for every value of $\vec{D}$.

Convenient notation: $\mathbf{tc}_{\vec{A};\vec{B}}(E)$

SPJRUD+TC has a lower complexity than SPJRUD+FP (**NL** versus **P**).

Discussion and Exercises

See course notes.