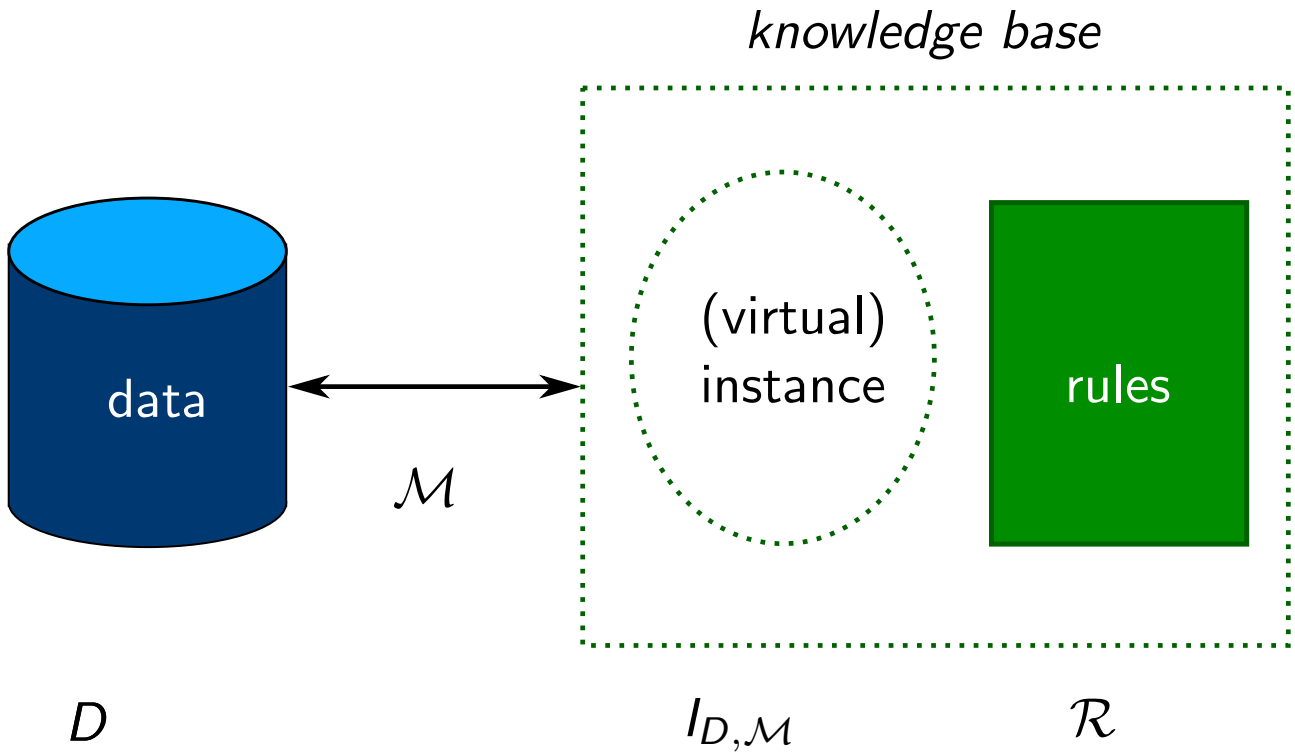
Parallelisable Existential Rules (KR 2021)

Maxime Buron, Marie-Laure Mugnier, Michael Thomazo

GraphIK seminar, 2/9/2021

Context

Ontology-Based Data Access



Existential rules as a uniform language for both $\mathcal{M}$ and $\mathcal{R}$

Existential rules as Mappings

Global-Local-As-View Mappings (GLAV)

Existential rules

$$\forall \vec{x}\, \forall \vec{y}\, (\ \mathsf{Body}[\vec{x}, \vec{y}] \rightarrow \exists \vec{z}\, \mathsf{Head}[\vec{x}, \vec{z}]\ )$$

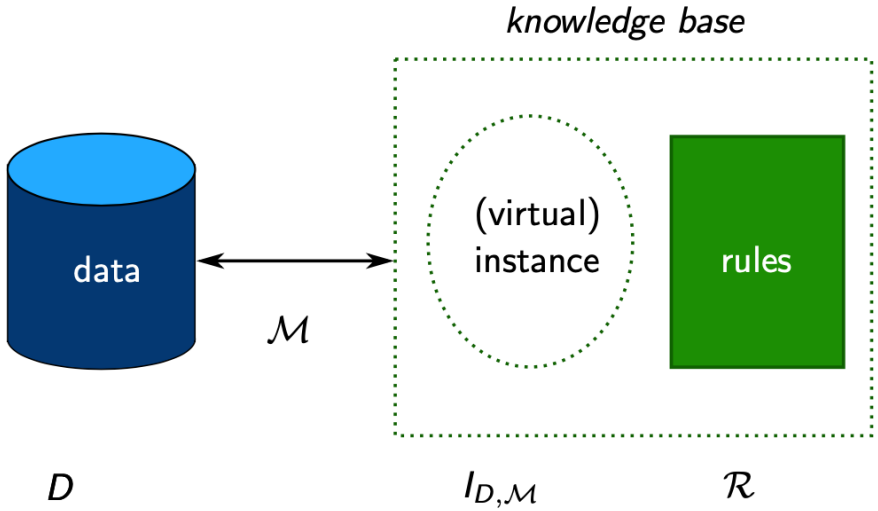
$$\forall \vec{x}\, (\ \exists \vec{y}\, \mathsf{Body}[\vec{x}, \vec{y}] \rightarrow \exists \vec{z}\, \mathsf{Head}[\vec{x}, \vec{z}]\ )$$

Mappings (aka source-to-target Tuple-Generating-Dependencies)

- Body is a (conjunctive) query on the schema of D with answer variables $\vec{x}$ (we assume that D is a relational database - can be extended to any kind of database by taking the view associated with tuples of answers)
- Head is a (conjunctive) query on the KB vocabulary with answer variables $\vec{x}$

Context

Answering (conjunctive) queries



Either: materialize the KB

Materialize $I_{D,\mathcal{M}}$, then query the obtained KB (saturation / query reformulation)

Or: leave the instance (factbase) virtual

Reformulate the query with $\mathcal{R}$, then rewrite it with $\mathcal{M}$

Query reformulation is expensive: it is a recursive process

Context

Compiling $\mathcal{R}$ into $\mathcal{M}$

Aim: compute a new mapping $\mathcal{M}'$ such that: for all query q,
$\mathit{rewriting}(\mathit{reformulation}(q, \mathcal{R}), \mathcal{M}) \equiv \mathit{rewriting}(q, \mathcal{M}')$

Dually: for all database D, $\mathit{chase}(I_{D,\mathcal{M}}, \mathcal{R}) \equiv I_{D,\mathcal{M}'}$

Simple example ($\mathcal{R}$ restricted to a class hierarchy)

$\mathcal{M}$: $M_1 = s_1(x, y) \to t_1(x, y)$
$M_2 = s_2(x, y) \to t_2(x)$

$\mathcal{R}$: $R_1 = t_2(x) \to t_3(x)$
$R_2 = t_2(x) \to t_4(x)$
$R_3 = t_3(x) \to t_5(x)$
$R_4 = t_4(x) \to t_5(x)$

$\mathcal{M}'$: $M'_1 = M_1 = s_1(x, y) \to t_1(x, y)$
$M'_2 = s_2(x, y) \to t_2(x) \wedge t_3(x) \wedge t_4(x) \wedge t_5(x)$

$q() = t_1(u, v) \wedge t_5(u) \wedge t_5(v)$
$\mathit{ref}(q, \mathcal{R})$ contains 16 queries
$\mathit{rew}(\mathit{ref}(q, \mathcal{R}), \mathcal{M}) =$
$\{s_1(u, v) \wedge s_2(u, w_1) \wedge s_2(v, w_2)\}$
$= \mathit{rew}(q, \mathcal{M}')$

Here, it suffices to "chase" each mapping assertion independently

Context

Compiling $\mathcal{R}$ into $\mathcal{M}$

A sightly more complex example

$\mathcal{M}$: $M_1 = s_1(x, y) \rightarrow t_1(x, y)$
$M_2 = s_2(x, y) \rightarrow t_2(x)$
$\mathcal{R}$: $R_1 = t_2(x) \rightarrow \exists z\ t_3(x, z)$
$R_2 = t_1(x, y) \wedge t_3(x, z) \rightarrow t_4(y)$

$\mathcal{M}'$: $M_1 = s_1(x, y) \rightarrow t_1(x, y)$
$M_2 = s_2(x, y) \rightarrow t_2(x)$
$M_3 = s_2(x, y) \rightarrow \exists z\ t_3(x, z)$
$M_4 = s_1(x, y) \wedge s_2(x, z) \rightarrow t_4(y)$

Here, mapping assertions cannot be considered independently

Idea: $\mathcal{M}'$ is obtained by composing the rules from $\mathcal{M} \cup \mathcal{R}$ until fixpoint, then keeping only mapping assertions

Desired property

For all database D, $chase(I_{D,\mathcal{M}}, \mathcal{R}) \equiv I_{D,\mathcal{M}'}$
where $I_{D,\mathcal{M}'}$ is obtained by a single breadth-first step of $chase(D, \mathcal{M}')$

Research question

Under which conditions on $\mathcal{R}$
can the chase of $(I, \mathcal{R})$ be simulated in a single (breadth-first) step?

- Parallelisability: $\mathcal{R}$ is parallelisable if there exists a finite rule set *independent from any instance* able to produce (an equivalent superset of) the chase of $\mathcal{R}$ in a single step.
- How to characterize parallelisable rule sets?

Key notion: Piece

Piece

Minimal set of atoms 'glued' by existential variables (in a rule) or nulls (in the chase)

$r(x, y) \rightarrow \exists z_1 \exists z_2 \exists z_3 \; p(x, z_1) \wedge p(z_1, z_2) \wedge p(z_2, z_3) \wedge p(x, y)$

Any rule can be decomposed into an equivalent set of rules with a single-piece head

In the following:

- Rules have a single-piece head (and have no constant)
- Semi-oblivious chase (two applications of R that coincide on the frontier produce the same result); moreover, breadth-first
- Note: single-piece decomposition can only make the so-chase to halt more often (see Lucas' work).

Parallelisability

(We consider the semi-oblivious chase)

$\mathcal{R}$ is parallelisable if there exists a finite rule set $\mathcal{R}'$ such that for any instance I:

1. there is an injective homomorphism from $chase_\infty(I, \mathcal{R})$ to $chase_1(I, \mathcal{R}')$
2. there is a homomorphism from $chase_1(I, \mathcal{R}')$ to $chase_\infty(I, \mathcal{R})$

Relationship with boundedness

$\mathcal{R}$ is bounded if there is k s.t. for any instance I, $chase_k(I, \mathcal{R}) = chase_\infty(I, \mathcal{R})$

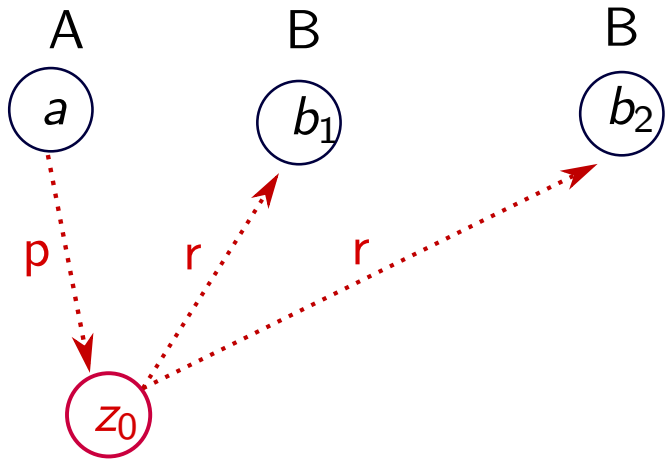
If $\mathcal{R}$ is *parallelisable* then it is bounded, but the converse does not hold.

Boundedness does not ensure parallelisability

'Prime example' (bounded)
$R_1 : A(x) \to \exists z\ p(x, z)$
$R_2 : p(x, z) \wedge B(y) \to r(z, y)$

$I = \{A(a), B(b_1), B(b_2)\}$

$chase_\infty(I, \mathcal{R}) = I \cup \{p(a, z_0), r(z_0, b_1), r(z_0, b_2)\}$



For any n:
$I_n = \{A(a), B(b_1), \ldots, B(b_n)\} \Rightarrow$ there is a null that appears in $n + 1$ atoms
Hence this rule set is not parallelisable

A new class: Pieceful

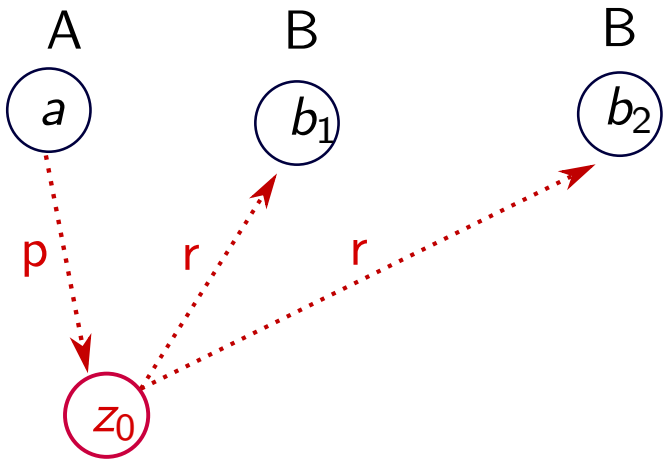
$\mathcal{R}$ is pieceful if for any trigger (R, π) in any derivation with $\mathcal{R}$,

- either $\pi(\mathit{frontier}(R))$ belongs to the terms of the initial instance
- or $\pi(\mathit{frontier}(R))$ belongs to the terms of atoms brought by a *single* previous rule application.

Prime example is not pieceful

Prime example (bounded)
$R_1 : A(x) \to \exists z\ p(x, z)$
$R_2 : p(x, z) \wedge B(y) \to r(z, y)$

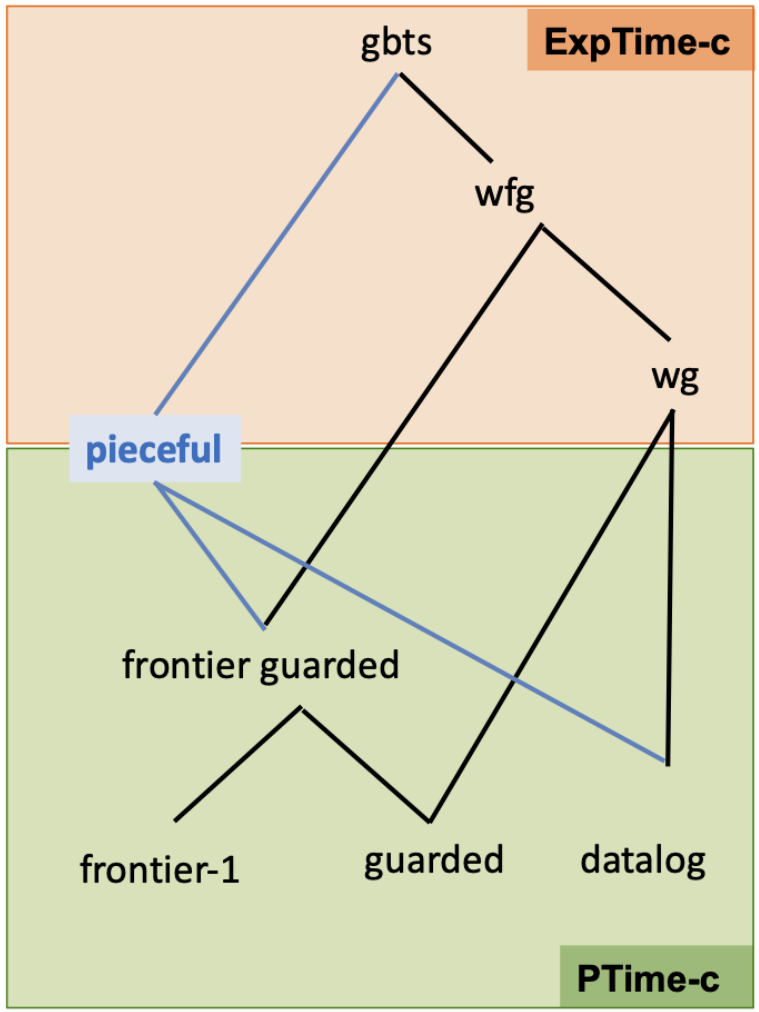
$I = \{A(a), B(b_1), B(b_2)\}$
First trigger: $(R_1, \{x \mapsto a\}$; creates $p(a, z_0)$
Then: $(R_2, \{x \mapsto a, z \mapsto z_0, y \mapsto b_1\})$



Parallelisability $\Rightarrow$ Piecefulness

Why? If a rule set $\mathcal{R}$ is not pieceful, one can create an instance I_n s.t. $chase(I, \mathcal{R})$ has a null that occurs in at least n atoms.

New landscape



(with data complexity of conjunctive query entailment)

Parallelisability = Boundedness + Piecefulness

What we have so far:

- Parallelisability ⇒ Boundedness (but the converse is false: see prime example)
- Parallelisability ⇒ Piecefulness (but the converse is false: see transitivity)

Boundedness + Piecefulness ⇒ Parallelisability

- If $\mathcal{R}$ is pieceful, the size of a piece in $chase_k(I, \mathcal{R})$ is bounded independently from I
- If $\mathcal{R}$ is pieceful *and bounded*, the size of a piece in the chase is bounded independently from I. Hence, there is a finite number of 'non-isomorphic' pieces associated with $\mathcal{R}$
- If $\mathcal{R}$ is bounded, each piece (seen as a query) has a finite set of rewritings (reformulations) with $\mathcal{R}$
  ⇒ roughly, $\mathcal{R}'$ is the set of all rules of the form $rewriting(P) \to P$

Now, how to compute a parallelisation?

Rule composition

Datalog unfolding

For datalog rules: parallelisability = boundedness

A parallelisation of $\mathcal{R}$ can be computed by 'unfolding' the rules from $\mathcal{R}$
$\mathcal{R}^\star$: starting from $\mathcal{R}$, we repeatedly unfold a rule from $\mathcal{R}^\star$ with a rule from $\mathcal{R}$

$\mathcal{R} = \{R_1, R_2, R_3\}$

$R_1 : A(x) \to B(x)$
$R_2 : C(x) \to D(x)$
$R_3 : B(x) \wedge D(x) \to G(x)$
Denoting $R_i \circ R_j$ the unfolding of R_i by R_j, we obtain:
$R_3 \circ R_1 : A(x) \wedge D(x) \to G(x)$
$R_3 \circ R_2 : C(x) \wedge B(x) \to G(x)$
$(R_3 \circ R_1) \circ R_2 : A(x) \wedge C(x) \to G(x)$
$(R_3 \circ R_2) \circ R_1 = (R_3 \circ R_1) \circ R_2.$
$\mathcal{R}^\star = \mathcal{R} \cup \{R_3 \circ R_1, R_3 \circ R_2, (R_3 \circ R_1) \circ R_2\}$

Soundness and completeness of $\mathcal{R}^\star$: $I, \mathcal{R} \models q$ iff $chase_1(I, \mathcal{R}^\star) \models q$

Rule composition

Existential rules

Unfolding extended to (single-piece) existential rules

- Based on piece-unifiers instead of classical unifiers
- Keeps single-piece rules

$R_1 : A(x) \rightarrow \exists z\ p(x, z)$
$R_2 : p(x, z) \rightarrow B(z)$
$R_3 : C(x) \wedge B(y) \rightarrow r(x, y)$

$R_2 \circ R_1 = A(x) \rightarrow \exists z\ p(x, z) \wedge B(z)$
$R_3 \circ R_2 = p(x', z) \wedge C(x) \rightarrow \cancel{B(z)} \wedge r(x, z)$
Note that $p(x', z) \wedge C(x) \rightarrow B(z)$ is useless w.r.t. R_2

Details on existential rule composition $R_2 \circ R_1$

Given $R_1 : B_1 \to H_1$ and $R_2 : B_2 \to H_2$
and $\mu = (B_2', H_1', u)$ a piece-unifier of B_2 with R_1:

1. If $u(\mathit{frontier}(R_2)) \cap \mathit{exist}(R_1) = \emptyset$:

$$R_2 \circ_\mu R_1 = u(B_1) \cup u(B_2 \setminus B_2') \to u(H_2)$$

2. Otherwise:

$$R_2 \circ_\mu R_1 = u(B_1) \cup u(B_2 \setminus B_2') \to u(H_1) \cup u(H_2)$$

In short: if no frontier variable of R_2 is unified with an existential variable of R_1, the head of R_1 can be safely ignored, which allows to keep single-piece rules

Rule composition on the prime example

$R_1 : A(x) \to \exists z\ p(x, z)$
$R_2 : p(x, z) \wedge B(y) \to r(z, y)$

Let us build $\mathcal{R}^\star$:
$R_2 \circ R_1 : A(x) \wedge B(y) \to \exists z\ p(x, z) \wedge r(z, y)$
$R_2 \circ (R_2 \circ R_1) : A(x) \wedge B(y) \wedge B(y_1) \to \exists z\ p(x, z) \wedge r(z, y) \wedge r(z, y_1)$
etc.
At each step, a new rule $R_2 \circ R^*$, where R^* is the rule created at the preceding step:
$A(x) \wedge B(y) \wedge B(y_1) \ldots B(y_i) \to \exists z\ p(x, z) \wedge r(z, y) \wedge r(z, y_1) \ldots \wedge r(z, y_i)$

What this example shows:

- Completeness requires composition of the form $R \circ R^*$ (and not only $R^* \circ R$ as in datalog)
- $\mathcal{R}^\star$ may be infinite even if $\mathcal{R}$ is bounded, with no finite subset of $\mathcal{R}^\star$ being complete.

Parallelisation by rule composition

Completeness of $\mathcal{R}^\star$

If $\mathcal{R}$ is pieceful, then for any instance I, each piece of $chase_\infty(I, \mathcal{R})$ can be obtained by applying a rule from $\mathcal{R}^\star$ to I.

Conjecture: this is true even if $\mathcal{R}$ is not pieceful

Corollary

If $\mathcal{R}$ is parallelisable (ie pieceful and bounded)
then it is parallelisable by a (finite) subset of $\mathcal{R}^\star$

Another characterization of piecefulness

(Existential) stability

- For a piece-unifier of $body(R_2)$ with R_1: if a frontier variable of R_2 is unified with an existential variable of R_1, then the whole frontier of R_2 is unified
- For $\mathcal{R}$: all piece-unifiers with rules of $\mathcal{R}$ have the stability property

Existential stability may be lost when a composed rule is added

We say that $\mathcal{R}$ has the existential stability 'at the infinite' if $\mathcal{R}^\star$ has the existential stability

Piecefulness = Stability at the infinite

- If $\mathcal{R}$ is pieceful then it has the existential stability
- If $\mathcal{R}$ is pieceful then $\mathcal{R}^\star$ is pieceful (hence, $\mathcal{R}^\star$ has the existential stability)
- If $\mathcal{R}$ is stable at the infinite then it is pieceful

Beyond 'existential' composition

Previous composition does not fit with our intuition

Prime example again

$R_1 : A(x) \to \exists z\ p(x, z)$
$R_2 : p(x, z) \wedge B(y) \to r(z, y)$
$I = \{A(a), B(b), B(c)\}$

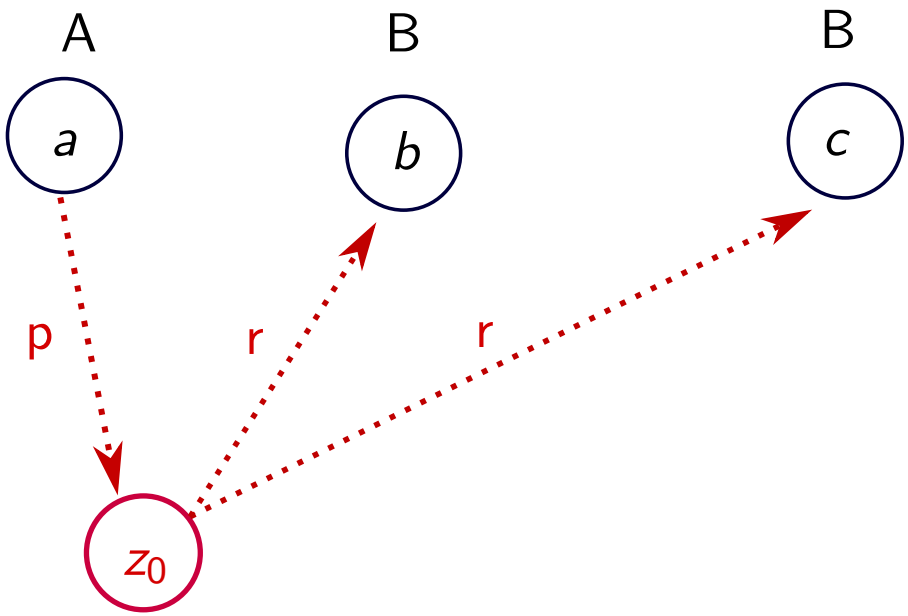
$chase_\infty(I, \mathcal{R}) = I \cup \{p(a, z_0), r(z_0, b), r(z_0, c)\}$
1 application of R_1 followed by 2 parallel applications of R_2

$R_2 \circ R_1 = A(x) \wedge B(y) \to \exists z\ p(x, z) \wedge r(z, y)$:
does not capture 'all applications of R_2 that use the atom brought by R_1'

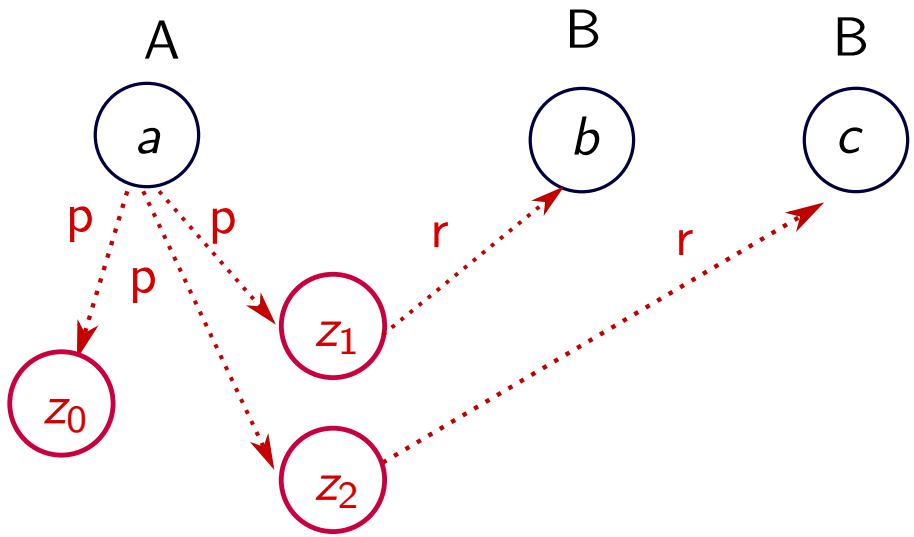
1 (breadth-first) chase step with $\mathcal{R} \cup \{R_2 \circ R_1\}$:
$I \cup \{p(a, z_0), p(a, z_1), r(z_1, b), p(a, z_2), r(z_2, c)\}$

Prime example (cont')

$R_1 : A(x) \rightarrow \exists z\ p(x, z)$
$R_2 : p(x, z) \wedge B(y) \rightarrow r(z, y)$
$I = \{A(a), B(b), B(c)\}$



$chase_\infty(I, \mathcal{R})$



One step of $\{R_1, R_2, R_2 \circ R_1\}$

Compact composition

Instead of:

$$R_2 \circ R_1 = \forall x \forall y \ (A(x) \wedge B(y) \rightarrow \exists z \ (p(x,z) \wedge r(z,y)))$$

we define:

$$R_2 \bullet R_1 = \forall x \exists z \forall y \ (A(x) \wedge B(y) \rightarrow p(x,z) \wedge r(z,y))$$

- $\bullet$ is more succint than $\circ$

  but the result may not be (equivalent to) an existential rule

When the piece-unifier satisfies existential stability, $\bullet$ is equivalent to $\circ$

Relationship with skolemization

Let's skolemize the prime example

$R_1 : A(x) \to \exists z\ p(x, z)$
$R_2 : p(x, z) \wedge B(y) \to r(z, y)$

$sk(R_1) : A(x) \to p(x, f(x))$
$sk(R_2) = R_2 : p(x, z) \wedge B(y) \to r(z, y)$

Composition of $sk(R_2)$ with $sk(R_1)$ (with classical unifiers) yields:
$A(x) \wedge B(y) \to p(x, f(x)) \wedge r(f(x), y)$ (where $p(x, f(x))$ could be removed)

It is not the skolemization of an existential rule but rather $sk(R_2 \bullet R_1)$
$R_2 \bullet R_1 = \forall x \exists z \forall y\ (A(x) \wedge B(y) \to p(x, z) \wedge r(z, y))$

E.g. on $I = \{A(a), B(b), B(c)\}$, one would obtain the expected result:
$I \cup \{p(a, f(a)), r(f(a), b), r(f(a), c) = chase_\infty(I, sk(\mathcal{R}))$

Many perspectives

- Better understand rule composition to compute parallelisation in practice
  - Is $\mathcal{R}^\star$ always complete?
  - When $\mathcal{R}$ is pieceful, can we consider only compositions of the form $R^* \circ R$?
  - When $\mathcal{R}$ is parallelisable, how to compute a complete finite subset of $\mathcal{R}^\star$?
- Go beyond parallelisable rules by partitioning / combining rule subsets
- Better understand the properties of pieceful rules
- Extend the notion of parallelisability
- Composition of skolemized rules goes beyond existential rules. Can we still work with it?