



Heterogeneous Proofs

Spider Diagrams meet Higher-Order Provers

Matej Urbas and Mateja Jamnik

`{mu232, mj201}@cl.cam.ac.uk`

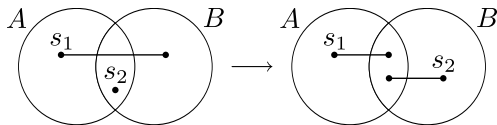
Heterogeneous reasoning?

Heterogeneous reasoning is...

... reasoning about **diagrammatic** and **sentential** propositions with **mixed** diagrammatic and sentential **inference steps**.

In other words:

- two languages (one of them is diagrammatic),
- two sets of inference rules, and
- a bi-directional translation between the two.



$$(\exists s \ s' . s \neq s' \wedge s \in A \cap B \wedge s' \in (A - B) \cup (B - A)) \longrightarrow (\exists s \ s' . s \neq s' \wedge s \in A \wedge s' \in B)$$

Main hypothesis:

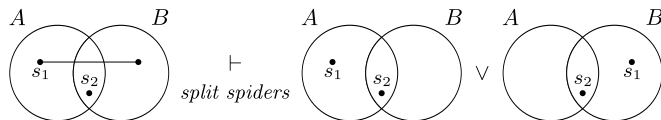
Interactive heterogeneous reasoning is feasible, and it can be done **formally** with **automated verification** of proofs (i.e.: with proof reconstruction).

We also hope to show that:

- HR can, for specific domains, produce more concise or intuitive formulae and proofs (we currently target MFOL).
- HR enables easier extensions to the diagrammatic logic (by adding new diagrammatic language elements and automatically importing or formalising new inference rules).

The diagrammatic language: **Spider Diagrams**

- A well-defined language on MFOL with a set of sound and complete inference rules.



- We are developing a diagrammatic reasoner for spider diagrams, called **Speedith** (sources at <http://gitorious.net/speedith>).

Heterogeneous framework

The sentential reasoner: **Isabelle**

- Formalisation of spider diagrams in Isabelle/HOL.

```
lemma sd_rule_split_spiders:
  "[[ habA = (h#hs); habB = h ] ]  $\implies$  sd_sem (PrimarySD habA shzs) =
  (sd_sem (PrimarySD (habA#hs) shzs)  $\vee$  sd_sem (PrimarySD (habB#hs) shzs))"
```

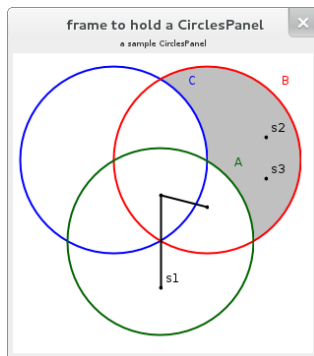
- ML-level translation procedures.
- Invocation of Speedith through custom tactics.

```
(* This lemma should land in the unit tests. *)
lemma testA: "( $\exists$ s1 s2. distinct[s1, s2]  $\wedge$  s1  $\in$  A  $\cap$  B  $\wedge$  s2  $\in$  (A - B)  $\cup$  (B - A))
 $\longrightarrow$  ( $\exists$ s1 s2. distinct[s1, s2]  $\wedge$  s1  $\in$  A  $\wedge$  s2  $\in$  B)"
  apply (sd_tac split_spiders sdi: 1 sp: "s2" r: "[([\"A\"],[\"B\"])]")
  apply (sd_tac add_feet sdi: 3 sp: "s2" r: "[([\"A\", \"B\"],[ ])]")
  apply (sd_tac add_feet sdi: 3 sp: "s1" r: "[([\"A\"],[\"B\"])]")
  apply (sd_tac add_feet sdi: 2 sp: "s2" r: "[([\"A\", \"B\"],[ ])]")
  apply (sd_tac add_feet sdi: 2 sp: "s1" r: "[([\"B\"],[\"A\"])]")
  apply (sd_tac idempotency sdi: 1)
  by auto
```

Heterogeneous framework

TODO: GUI integration

- Graphical interactive input and inference rule selection.
- Visualisation of spider diagrams with *iCircles* by Stapleton and Flower.



How should it work?

Something like this:

The screenshot shows the Isabelle/ML Playground interface. The main window displays the following code:

```
49 ML {* Diabelli.from_term_to_sd (@{term "( $\exists f. \text{sd } [s, s'] f (f s \in A \cap B \wedge f s' \in (A$   
50  
51 (* The proof of the main example using many first-order existential  
52 quantifiers. *)  
53 lemma example_1_b_1: "( $\exists s s'. \text{sp } [s, s'] (s \in A \cap B \wedge s' \in (A - B) \cup (B - A))$ )  $\longrightarrow$   
54 ( $\exists s s'. \text{sp } [s, s'] (s \in A \wedge s' \in B)$ )"  
55 apply (sd_reasoner split_spider "s'")  
56 apply (sd_interactive_reasoner)  
57 by (auto simp add: sp_def, iprover)  
58
```

The Result Window shows the "Spider diagrams" tab, which contains the following diagram:

Invoke Speedith

The diagram illustrates the transformation of a spider diagram. On the left, two overlapping circles labeled A and B are shown. The intersection contains two points: s' (top) and s (bottom). An arrow points to the right diagram, where the same two overlapping circles A and B are shown, but the intersection now contains s (top) and s' (bottom). This represents a swap of the elements s and s' within the intersection of sets A and B.

54 | 39 | INS Free: 68MB/227MB

Heterogeneous Proofs: Spider Diagrams meet Higher-Order Provers

Matej Urbas

`Matej.Urbas@cam.ac.uk`

Mateja Jamnik

`Mateja.Jamnik@cam.ac.uk`

Resources:

- Speedith:

`http://gitorious.net/speedith`

- iCircles:

`https://gitorious.org/speedith/inductive_circles`