FORMALISM B

Alessio Guglielmi (TU Dresden)
20.12.2004

In this note (originally posted on 9.2.2004 to the Frogs mailing list) I would like to suggest an improvement on the current notions of deep inference, including what I called formalism A [AG11]. The suggested formalism is called B, for the time being.

The slogan for this note is very simple:

   inference rules operate on derivations .

Not on formulae, not on structures, but on entire derivations. It's a simple idea, but there are non-trivial consequences. What I find difficult, of course, is assessing the impact of this notion in the long run, because the change is almost like moving from the sequent calculus to the calculus of structures (CoS).

In any case, the general idea is that this approach should be more general than CoS (which in turn is more general than the sequent calculus). As you will see, the problem with the new formalism is that a decent concrete syntax is currently unknown: we will have one when we will have deductive proof nets, but for the time being there's nothing viable. On the other hand, I hope that formalism B will make us progress towards the goal of getting deductive proof nets, because at least it describes nice properties they should have.

For this reason, it's important that CoS is a faithful description of formalism B, meaning that we can study proofs and their properties in the syntax of CoS, and then we see them in the more semantic setting of formalism B. For example: decomposition and cut elimination are done in CoS; proof identity in formalism B.

There are many motivations for using deep inference, but the one I'm going to use here is `getting rid of bureaucracy´, which, of course, is a (or *the*?) problem for proof identity.

Bureaucracy in Derivations

Let us distinguish two kinds of annoying trivialities that occur in CoS derivations (the situation in the sequent calculus is even worse):

Type A Bureaucracy

We have type A bureaucracy whenever we find ourselves in the following situation: `there are two non-overlapping redexes´, or, in other words, `there's a diamond in the proof search space´. More precisely, let S{R}{T} be a structure where R and T appear in the

context S{ }{ } and they don't overlap. We are in a situation in
which there exist two proofs

```
    S{R'}{T'}        S{R'}{T'}
        |                |
     Δ₂|             Δ₁|
        |                |
    S{R'}{T}   and   S{R}{T'} .
        |                |
     Δ₁|             Δ₂|
        |                |
     S{R}{T}          S{R}{T}
```
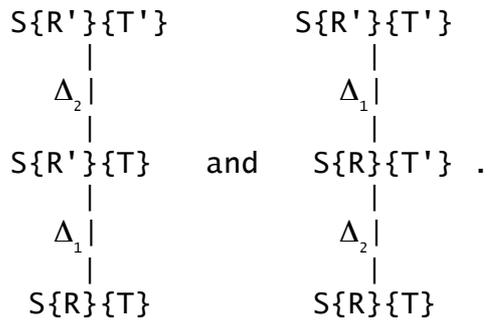
They are made up from the more elementary derivations

```
     R'             T'
      |              |
   Δ₁|    and     Δ₂|  .
      |              |
      R              T
```
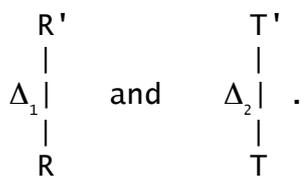
This is an obvious case of bureaucracy, corresponding to a trivial
case of permutability (of inference rules, or of the entire
derivations Δ_1 and Δ_2).

Formalism A takes care of this problem, by allowing the expressing
of the derivation

```
    S{R'}{T'}
       |   |
    Δ₁|   |Δ₂  ,
       |   |
    S{R}{T}
```
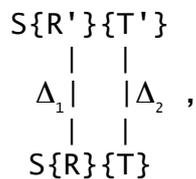
in which Δ_1 and Δ_2 are simply put `in parallel´.

Type B Bureaucracy

We have type B bureaucracy when redexes are nested. For example,
take Δ_1 as above, but put it `inside´ a switch, as follows:

```
    (R',[T,U])
       |
    Δ₁|
       |
    (R,[T,U])
  s ---------- ;
    [(R,T),U]
```
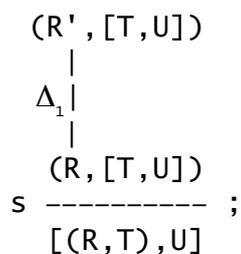
Clearly, one can permute the switch all the way up and do

```
        [(R',T),U]
    s ────────────
        (R',[T,U])
           |
        Δ₁ |              ,
           |
          (R,[T,U])
```

or any of the intermediate derivations. In general, the phenomenon
is the same when in the place of a simple switch one has an entire
derivation: a clear case of bureaucracy, but how can we get rid of
it?

Inference on Derivations

The simple solution is to allow inference rules on derivations, for
example the switch rule becomes

```
        (Δ,[Δ',Δ"])
    s ─────────────  ,
        [(Δ,Δ'),Δ"]
```

where Δ, Δ' and Δ'' are derivations. This also takes care of more
complex cases in which you have *three* `parallel´ derivations Δ, Δ'
and Δ'' inside a switch. At this time, it's the most general notion
of deep inference I can think of.

Note that one extends the equivalence classes of structures to
derivations, meaning that the above object is also to be taken
modulo associativity, commutativity, etc.

I believe that this notion actually further simplifies the picture,
because we now have just one first-class object, the derivation, and
all the other objects are projections, special cases of it: CoS
derivations, sequent calculus derivations, structures, formulae,
atoms,...

Now, there is a series of simple considerations one should make.
Let's call these new rules `B rules´.

At face value, a B rule *increases* bureaucracy simply because it
copies (from premise to conclusion) entire derivations. This is why
I'm saying that the syntax is an insufficient approximation. *Of
course*, one should not think of a B rule as a term rewriting rule;
rather it's a moral, semantical understanding of a more concrete
(CoS, sequent calculus) object.

The example above with Δ_1 and switch should correspond to something
like this:

```
   (  R' ,[T,U] )
    +---+  \   /
    |   |   \ /
    |  Δ₁|    X    ,
    |   |   / \
    +---+  /   \
   [( R ,T) ,U  ]
```
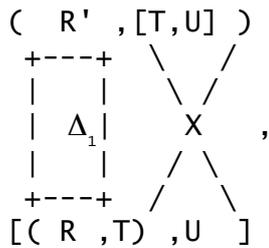
where somehow the logical relations get crossed by some net. But *there is no duplication* of Δ_1. If one gets the idea of this one, it's easy to get the idea in general, where one has three subderivations for switch. It clearly is worse than DNA in mid-mitosis as far as drawing these beasts goes.

Now, an even more delicate case occurs when you have *moral* duplication (as opposed to the fake syntactical one above):

```
        [Δ,Δ]
   c↓ ------    (contraction).
         Δ
```

There are two considerations one can make for contraction and similar cases:

1   Contraction can be pushed up along Δ in such a way that it duplicates a minimum of information (contraction can always be reduced to structures, and then to atoms for most logics).

2   CoS already does an excellent job at making rules linear (in the sense of term rewriting, meaning: `no duplication´). While in the sequent calculus one has plenty of duplication, in CoS this is always reduced to a minimum.

But now one asks oneself a strange question.

Negation??

What is

```
         t
   i↓ ------    (interaction)?
      [Δ,¬Δ]
```

The problem is, of course, $\neg\Delta$. Maybe I'm wrong, but this is not a stupid question.

Reference

[AG11] Alessio Guglielmi. Formalism A. Manuscript, 2004. URL: http://iccl.tu-dresden.de/~guglielm/p/AG11.pdf.

Web Site

http://alessio.guglielmi.name/res/cos.