SUBATOMIC LOGIC

Alessio Guglielmi (TU Dresden)
19.11.2002- updated on 21.11.2002

One can unify classical and linear logic by using only two simple, linear, \`hyper´ inference rules; they generate nice systems for all the fragments, where all rules are local. The various logics are determined by the algebra of their units, for example boolean algebra determines classical logic. We can prove cut elimination easily and once and for all in the two-rule system, and the procedure scales down to all the derived fragments. This note shows the main ideas of the construction; I will follow with a detailed technical report soon.

Classical Logic

In order to achieve the result, I need to consider atoms as composed of more elementary matter. For example, take classical logic, and suppose that a, b, c, ... are the propositional variables (*positive atoms*). Every atom corresponds to a binary logical relation (like disjunction and conjunction) that I write a<A;B>, b<A;B>, c<A;B>, ..., where A and B can be any structures of the language.

The language of *subatomic classical propositional logic* is freely built on the two units **0** and **1** by the binary operators [_,_] (also written **D**<_;_>, *disjunction*), (_,_) (also written **C**<_;_>, *conjunction*) and a<_;_>, b<_;_>, c<_;_>, ... (*atoms*); there's no negation.

We can distinguish two kinds of atoms, the *tame atoms*, which are relations among units, like a<**1**;**0**> and b<**0**;**0**> and the *wild atoms*, like a<b<**0**;**1**>;**1**>. Wild atoms do not correspond to anything in classical logic; tame ones instead correspond to normal formulae this way:

a<**0**;**0**> corresponds to falsehood **f**;
a<**0**;**1**> corresponds to the atom a;
a<**1**;**0**> corresponds to the negated atom ¬a;
a<**1**;**1**> corresponds to truth **t**.

We will always enforce the equations

a<**0**;**0**> = **0** ,
a<**1**;**1**> = **1** .

So, we establish that every atom is a non-commutative self-dual binary logical relation, whose De Morgan laws respect order:

¬α<A;B> = α<¬A;¬B> .

Since negation can always be pushed *to units* until it disappears, we don't need it explicitly in the language.

Now, we know already how to present classical [BT] and linear logic [LS], and related fragments, in the calculus of structures [WS]. We also know that we can present all these systems with local rules--an impossible task in the sequent calculus [KB]. The system for linear logic is particularly interesting, because it is big (34 rules) but all rules are variations on a couple of themes. There clearly is some hidden scheme at work there, and here I'm going to address this issue: how can we *generate* that system in a way that the scheme is evident, and *useful*?

It turns out that, unless I'm terribly mistaken, we can do much more, and namely we can unify classical and linear logic under the same scheme. This scheme consists of two *hyperrules*, is *very simple* and provides cut elimination for all the generated systems. I also conjecture that we can extend the scheme to various modal logics and to quantifiers of varying order. The trick is to deal with atoms as if they were normal logical relations, as shown above. We also need to use the calculus of structures, mainly because of its ability to deal with *couples* of logical relations in rules.

One of the two hyper rules is *hypermedial*:

```
    α< β<A;E> ; β<B;F> >
mαβ--------------------- ,
    β< α<A;B> ; α<E;F> >
```

where α and β are any two logical relations (**D**, **C**, a, b, c, ...).

For α = **D** and β = **C** the hypermedial becomes

```
    [ (A,E) , (B,F) ]
mDC-------------------
    ( [A,B] , [E,F] )
```

(I'm applying the usual conventions of the calculus of structures for the sake of readability, so, instead of **D**<A;B>, I write [A,B]). This rule is equivalent to the normal medial we typically use.

For every atom a and α = a and β = **D**, the hyperrule becomes the crazy rule

```
    a< [A,E] ;  [B,F] >
maD--------------------- ;
    [ a<A;B> , a<E;F> ]
```

this rule is new and generates identity and weakening, all of which in their atomic form. We can also generate m**D**a, which will give us contraction:

```
          [ a<A;E> , a<B;F> ]
      mDa--------------------- ;
          a< [A,B] ;  [E,F] >
```

and we can generate m**C**a, which will give us cut:

```
          ( a<A;E> , a<B;F> )
      mCa--------------------- .
          a< (A,B) ;  (E,F) >
```

Let us see why these rules generate familiar rules, by only considering tame atoms and by checking some possible assignments of units to A, B, E and F. We are using the rules for classical logic, so we have the boolean equations:

```
      [0,0] = (0,0) = (0,1) = (1,0) = 0 ;
      [1,1] = [1,0] = [0,1] = (1,1) = 1 .
```

Then

```
          a< [0,1] ;  [1,0] >              t
      maD---------------------   yields  ------   (identity),
          [ a<0;1> , a<1;0> ]            [a,¬a]

          a< [0,1] ;  [1,1] >              t
      maD---------------------   yields  -----   (weakening),
          [ a<0;1> , a<1;1> ]            [a,t]

          [ a<0;1> , a<0;1> ]            [a,a]
      mDa---------------------   yields  -----   (contraction),
          a< [0,0] ;  [1,1] >              a

          ( a<0;1> , a<1;0> )            (a,¬a)
      mCa---------------------   yields  ------   (cut).
          a< (0,1) ;  (1,0) >              f
```

The reader can continue the exercise and see that all generated rules are sound, and that all rules of classical logic are generated except for switch.

We can instantiate hypermedial to all ma**D**, m**D**a, ma**C**, m**C**a, for all atoms a, and to m**DC** (which is the ordinary medial found by Alwen).

Switch is generated, for α = **C** and β = **D**, by the other hyper rule *hyperswitch*:

```
          α< β<A;E> ;  β<B;F> >
      sαβ----------------------- ,
          β< α<A;B> ; ~α<E;F> >
```

where ~α is the dual relation of α (in the case of atoms, which are self-dual, hyperswitch coincides with hypermedial).

Comments

Is it a trick? What can we gain from this little game?

First, we have to address the problem of wild atoms. Clearly, the system can prove more than classical logic does, for example [a<**0**;[a<**0**;**1**>,**1**]>,**1**] can be proved by

```
                   1
        maD------------
              [a<0;1>,1]
   maD--------------------- .
         [a<0;[a<0;1>,1]>,1]
```

But if we restrict provability to tame atoms, then the system coincides with propositional classical logic, so what we have is a conservative extension. This can be seen easily by noticing that all the rules involving atoms generated by hypermedial are invertible.

What can we gain? Primarily, we can gain a simplification of the cut elimination argument, since we only need to cope with two rules. The rules are `hyper´, but it’s not more difficult to cope with hyper rules than with ordinary ones. Anyway, if it were only for classical logic, then this wouldn’t be so much.

Then the next question is whether this is just a trick or whether it really provides some useful insight. The next step is then to see what happens with linear logic. We expect, of course, that weakening and contraction somehow disappear. Well, they do!

Multiplicative Linear Logic

Here, everything is as above, but we interpret **D** and **C** as, respectively, par and tensor; then **0** and **1** are ⊥ and 1. If this were the only change, then contraction and weakening would still be there, but we also have to change the unit equations, which now are:

```
      [0,0] = (0,1) = (1,0) = 0 ;
      [1,0] = [0,1] = (1,1) = 1 .
```

Then

```
        a< [0,1] ;  [1,0] >              1
maD---------------------   yields   ------   (identity),
        [ a<0;1> , a<1;0> ]           [a,¬a]

        a< [0,1] ;  [1,1] >     a<1;[1,1]>
maD--------------------- is ---------- (not weakening!),
        [ a<0;1> , a<1;1> ]       [a,1]
```

```
     [ a<0;1> , a<0;1> ]        [a,a]
mDa--------------------- is ---------- (not contraction!),
     a< [0,0] ;  [1,1] >     a<0;[1,1]>

     ( a<0;1> , a<1;0> )            (a,¬a)
mCa---------------------   yields   ------   (cut).
     a< (0,1) ;  (1,0) >              ⊥
```

It's not difficult to interpret what is happening: for example a<**0**;[**1**,**1**]> means keeping track of two positive occurrences of atom a; a<**1**;[**1**,**1**]> means **1** (i.e., a<**1**;**1**>) plus a, and so on. In other words, \`resource consciousness´ is faithfully taken care of by the units behaviour.

Additive Linear Logic

Let's perform another test. This time we want to get the additive linear logic fragment. **D** and **C** stay as they are in multiplicative linear logic, as well as the units. We have to add two mutually dual logical relations **D'** and **C'** for plus and with, which we respectively denote by *[_,_] and *(_,_).

With respect to the multiplicative units, the equations now become

```
*[0,0] = *(0,0) = 0 ;
*[1,1] = *(1,1) = 1 .
```

We are now interested in the rules ma**D'**, m**D'**a, m**C'**a and their inverted and dual rules. As expected, almost every recognisable rule is destroyed, so no identity, no cut and no weakening. But there is a surprise.

Consider m**D'**a in this case

```
      *[ a<0;1> , a<0;1> ]            *[a,a]
mD'a----------------------   yields   ------ .
      a< *[0,0] ; *[1,1] >              a
```

This strange rule of contraction, which applies to atoms in a plus context, is enough to recover both contraction in the normal linear logic sense (so, contraction applied to modalised formulae) and \`contraction´ in additive contexts. In addition, it is *atomic*.

Actually, this rule was already discovered by Lutz in [LS], which the reader should consult for details. It is in my opinion absolutely remarkable that this rule can be obtained *automatically* by the methods outlined above.

Final Remarks

It looks to me that the observations above could lead to a deep understanding of proof-theoretical mechanisms. Contrary to other monster, unifying logical systems, like Girard's LU (which has around 100 rules), we have to deal here with the combinatorial behaviour of *just two* rules, which, in addition, are similar.

What is possible to do then, is to study the cut elimination problem in a totally generic system made of the two hyperrules, and then specialise the system by providing the unit equations and a choice of couples of logical relations to be subject to medial and switch rules. The specialisation will be a *triviality*.

I started doing this already, and I found several surprising facts about the complexity of the cut elimination procedure. For example, contrary to common perception, contraction is not the real responsible for the growth in size of derivations during normalisation. Actually, the real responsible is medial (in the sense of [BT])! The observations above give a possible explanation of this fact, since we now see that atomic contraction is just one of the many sides of a rule that actually behaves well with respect to cut elimination (with weakening and identity).

References

[BT] Kai Brünnler and Alwen Tiu. A local system for classical logic. In R. Nieuwenhuis and A. Voronkov, editors, *LPAR 2001*, volume 2250 of *Lecture Notes in Artificial Intelligence*, pages 347-361. Springer-Verlag, 2001. URL: http://www.ki.inf.tu-dresden.de/~kai/LocalClassicalLogic-lpar.pdf.

[KB] Kai Brünnler. Two restrictions on contraction. Technical Report WV-02-04, Dresden University of Technology, 2002, URL: http://www.ki.inf.tu-dresden.de/~kai/RestrictionsOnContraction.pdf, submitted.

[LS] Lutz Straßburger. A local system for linear logic. In M. Baaz and A. Voronkov, editors, *LPAR 2002*, volume 2514 of *Lecture Notes in Artificial Intelligence*, pages 388-402. Springer-Verlag, 2002. URL: http://www.ki.inf.tu-dresden.de/~lutz/lls-lpar.pdf.

Web Site

[WS] http://www.ki.inf.tu-dresden.de/~guglielm/Research.