A CONCURRENT CUT-ELIMINATION

This message is about the computational content of proofs in Peano arithmetic. As shown in [1], it was implicit in Gentzen's 1936 unpublished (reproduced in [3]) paper that any proof in Peano arithmetic of an existential statement (sigma01) has a computational interpretation and it is natural to interpret Gentzen's procedure in a game theoretic way (following Lorenzen [5]). An alternative analysis of classical proofs is the technique of A-translation, by which proofs get interpreted essentially as functional programs, cf. [2].
In [1] and [4], is analysed
a simple example, due to G. Stolzenberg, for which it is possible to do a concurrent cut-elimination, which respects the symmetry of the data, whereas the functional program that we got by A-translation breaks this symmetry (see below.)

The natural question was then whether this procedure can be generalised to any proof.

This message contains a positive answer to this question, showing a way to do a truly concurrent cut-elimination for a classical sequent calculus. It can be interpreted as a strategy of reduction for a classical sequent calculus that is concurrent, but deterministic. The "concurrent" here means that, in general, a multiple cut cannot be reduced to simple cuts.

In order to be self contained, we recall the problem as it appears in a game theoretic terminology, and we present the solution on a simple, but characteristic, case.

We suppose given symbols for basic decidable relations on integers, like x=y, x<y,... and we consider only closed prenex formulae

A = (x)(E y)(z)P(x,y,z)

where P is an atomic decidable relation. We suppose that each atomic symbol P has an associate symbol P* that represents its complement, and we define as usual the negation of an arbitrary prenex formula

A* = (E x)(y)(E z)P*(x,y,z).

We define the depth of a prenex formula as its number of quantifiers. Here the depth of A is 3.

We will write A(n1) for (E y)(z)P(n1,y,z), A(n1n2) for (z)P(n1,n2,z) A*(n1) for (y)(E z)P*(n1,y,z). Since we are going to consider instantiation of prenex formulae, this notation is very convenient.

We define then by induction when a multiset of formulae

M = A1,...,An

is provable:

(1) if one of the Ai is atomic and true, then M is provable

(2) if all Ai are existential, and Ai = (E x)Bi(x), and for one integer n, the multiset M, Bi[n] is provable, then M is provable

(3) if some Ai are universal, let say A1 = (x)B1(x), A2 = (x)B2(x) and for all n1, n2 the multiset B1[n1], B2[n2], A3,...,An is provable, then M is provable.

As usual with such generalised inductive definitions, this definition has a natural interpretation in term of processes, or games: a proof of a multiset M can be seen as a winning strategy for a player I against a player II, where the rules are

(1) if one of the Ai is atomic and true, then I wins

(2) if all Ai are existential, and Ai = (E x)Bi(x), the player I has to choose one value n and the configuration becomes M, Bi[n]

(3) if some Ai are universal, let say A1 = (x)B1(x), A2 = (x)B2(x) then the player II has to choose values n1, n2 and the configuration becomes B1[n1], B2[n2], A3,...,An.

The MOVES of player in this game consist thus to choose a formula, and to instantiate it.

The moves of players that make a formula atomic has a crucial role in what follows. It is clear that we can as well impose that each such move from the player I has to make the atomic formula TRUE (we simply ignore the moves that do not follow this constraint), amd that the player II has lost the game when, whatever moves he can make, he instantiates an atomic formula to TRUE.

With this restriction, a proof for (x)(E y)R(x,y) is like a function f that satisfies (n)R(n,f(n)), and a proof of (E y)P(y) is like a witness for P.

The case of a proof for (E x)(y)R(x,y) is more complicated: the proof makes successive guesses for x, that are shown by player II to be false, until it finds a value n satisfying (y)R(n,y), that is such that the player II is forced to choose a value m such that R(n,m) holds. Typically, if f is seen as an oracle, a proof of (E x)(y)[f(x)<= f(y)], will make first a random guess x = 0, and then use each counter-example suggested by the player II as its next new guess. The fact that this strategy is winning is insured by the fact that N is well-founded.

It is direct also to reformulate this definition in terms of a sequent calculus with an omega-rule.

========

EXAMPLE: f is a function from N to {0,1}, seen as an oracle

Q0 proves (E a,b)[a<b & f(a)=f(b)] , (E x)(y) ~[x <= y & f(y) = 0]

Q1 proves (E a,b)[a<b & f(a)=f(b)] , (E x)(y) ~[x <= y & f(y) = 1]

P proves (x0)(E y0)[x0 <= y0 & f(y0) = 0],(x1)(E y1)[x1 <= y1 & f(y1) = 1]

and the behaviour of Qi is

send x = 0, waits for y = v. If f(v) = i, send x = v+1 waits for y = w. If v < w, and f(w) = i then send a = v, b = w;

this corresponds to the natural proof that, if the value i is taken infinitely often, it is taken twice.

The behaviour of P is

waits for x0 = u0, x1 = u1, and computes u = max(u0,u1), and f(u). If f(u) = 0 send y0 = u, else send y1 = u;

this corresponds to the proof that either the value 0 or the value 1 is taken infinitely many often.

Combining P, Q0, Q1 we expect to get a proof that f takes twice the same value.

The problem is that this is a classical argument: what are these two values actually contained in this proof??

This is G. Stolzenberg's example. Already in this case, the cut-protocol we will describe respects the symmetry between 0/1, and look only at most at 3 values of f, whereas if we analyse this argument via A-translation, the symmetry between 0/1 is broken, and sometimes the program looks 4 values of f (see [4]).

============

We consider then the following problem, that seems to contain the main difficulty for getting a concurrent cut-elimination:

PROBLEM

Let A0, A1 be two UNIVERSAL formulae.
Given a proof P of the multiset A0,A1, a proof Q0 of the multiset (E x)B(x), A0*, and a proof Q1 of the multiset (E x)B(x), A1*, where B is atomic, to give a "protocol" for using these proofs in computing a value n such that B[n] holds.

Furthermore, we ask this protocol to be "symmetric" in Q0, Q1.

REMARK: in term of sequent calculus, we have a multiple cut

Q0 CUT P CUT Q1.

One possibility is to eliminate the cut between Q0 and P, and then the remaining cut with Q1,

(Q0 CUT P) CUT Q1

this is the usual treatment of multiple cuts.
In the case where A0,A1 are UNIVERSAL, cut-elimination in sequent calculus is non Church-Rosser; the concurrent way we suggest can thus give in general different answers than what we get by reducing the multiple cut to single cut, and this is what happens on the 0/1 example.

=========

The idea is the following: the "cut-protocol" will try to ask the players Q0, P, and Q1 what moves they make, acting like a player II in virtual games against them.
In a way, the protocol let P, Q0, Q1 playing against each other as long as Q0, Q1 do not give a value n such that B[n] holds. The role of the cut-protocol is thus to observe what are the moves of each players, and according to these moves, to transmit these moves to the other players concerned.

Since we are only interested in the value n, the protocol can be described as a protocol for an internal communication between Q0, P and Q1, and we want that the protocol insures the termination of this internal communication.

The key is to define the protocol by induction on the depth of the cut-formulae A0 and A1, and to prove by induction that this cut-protocol insures the termination. We define it first for formulae of depth 2, and shows how it is defined for depth(A0) = n, if we have defined it for depth(A0) = n-1.

Definition of the cut-protocol for formulae of depth 2 (this contains the concurrent solution of the 0/1 example):

the protocol waits for the move of Q0 and Q1. If one of them instantiates (E x)B(x), then the protocol gets a value n that satisfies B[n], because B is atomic, and the protocol has finished. Otherwise, the moves are of the form

Q0: (E x)B(x), A0* =====> (E x)B(x), A0*, A0*(1)

Q1: (E x)B(x), A1* =====> (E x)B(x), A1*, A1*(1)

it transmits these moves to the player P, and waits for its answer when we instantiate A0 and A1 with the values chosen by Q0 and Q1

P: A0(1),A1(1) ======> ??

This answer is

either P: A0(1),A1(1) ======> A0(11),A0(1),A1(1)

or P: A0(1),A1(1) ======> A1(11),A0(1),A1(1)

In the first case, since A0 is of depth 2, the formula A0(11) is true, and hence P has given a conterexample to the moves A0*(1) of Q0 (the second case is completely symmetric.)
The protocol transmits this move to Q0, who is now in the position

Q0 : (E x)B(x), A0*, A0*(11) ====> ??

and either gives a value n for x, in which case the protocol has finished or backtrack in its choice A0*(11) and do the move

Q0 : (E x)B(x), A0*, A0*(11) ====> (E x)B(x), A0*, A0*(11), A0*(2).

The protocol transmits this value to P who has now the position

P : A0(2), A1(1) ====> ??

and P moves to

P : A0(2), A1(1) ====> A0(21), A0(2), A1(1) denying the last move of Q0

or

P : A0(2), A1(1) ====> A1(11), A0(2), A1(1) denying the last move of Q1.

In this last case, the cut-protocol transmits this to Q1,

Q1 : (E x)B(x), A1*, A1*(11) ====> ??

who either gives a value n such that B[n], or backtracks in its choice, moving to

Q1 : (E x)B(x), A1*, A1*(11) ====> (E x)B(x), A1*, A1*(11), A1*(2)

and P got the position

P : A0(2), A1(2) ====> ??

and so on.

This stops eventually, that is, eventually, Q0 or Q1 gives a value n such that B[n], because Q0 and Q1 are supposed to be proofs, i.e. winning strategies.

=========

At this point, the reader can stop and tries this protocol on the 0/1 example, for a fixed f, like

f(0) = 0, f(1) = 1, f(2) = 0 (1)

or

f(0) = 1, f(1) = 0, f(2) = 1 (2).

Let the protocol keeps track of all moves in term of LINES, that represents the history of all moves. Then the action of the protocol on (1) is (L1 and L2 can be interchanged, they are independent)

L1: A0*(0)	Q0 tries the value 0
L2: A1*(0)	Q1 tries the value 0
L3: A0(00)	P gets the values 0 and 0, and shows that the value 0 of Q0 is not correct
L4: A0*(1)	Q0 backtracks, tries the value 1
L5: A1(01)	P gets the values 1 and 0, and shows that the value 0 of Q1 is not correct
L6: A1*(2)	Q1 bactracks, tries the value 2
L7: A0(12)	P gets the values 1 and 2, and shows that the value 1 of Q0 is not correct
L8: a=0, b=2	Q0 is able to find a and b.

It will then be clear that the choice of the protocol is almost forced, except when it has to transmit two values to P, that is for the lines L3, L5 and L7. And it is only for the line L7 that there is a choice. Indeed, for this line, the protocol has to choose between values already sent by Q0, but they are two of them: 0 (in L1) and 1 (in L4). It seems clear that it has to choose 1 in L4, because 0 has been already shown to be incorrect in L3.
HOWEVER, and that is an important point, to reduce the multiple cut to single cuts has exactly the effect for the protocol of "forgetting" that 0 has been already shown to be incorrect, and to send the values 0 and 2 in place of the line L7.

=========

Definition of the cut-protocol, inductive step:

We suppose to have defined the cut-protocol for depth(A0)=n-1, and we define it for depth(A0) = n. The n-protocol simply follows the (n-1)-protocol as long as no atomic formula is created from A0 or A0*, and keep a list of all the moves that are going on.
Each such moves in this list is called a LINE.
Let us assume that n is odd (the case n even is similar).
Each moves of Q0 creating an atomic instance of A0, A0(k1...kn) is considered to be a backtracking point back to the line where A0*(k1...k(n-1)) was created by P. Indeed, the formula A0(k1...kn)

is true, and shows that the move A0*(k1...k(n-1)) of P cannot lead to anything for P, and that all the lines between this move of P and the move of Q0 can be as well forgotten. The protocol put them in parenthesis and follows the (n-1)-protocol on what it does for the new move of P.

The numbers of backtracking back to a given line is limited because P is supposed to follow a winning strategy. The (n-1)-protocol insures termination by induction hypothesis, hence the number of lines growing without backtracking is also limited.

This shows how to define the n-protocol, and why it insures termination.

TYPICALLY, if n = 3, and if the interaction starts like that the interaction shown for the 0/1 example, the move L8 can now be of the form

L8 : A0*(00n)   Q0 shows that the move L3 of P was actually incorrect

the protocol transmits this to P, and P has now the form

L9: P: A0(00n), A0(0), A1(0) =====> ??

and the protocol in the next lines, will backtrack from L9 to L3, that is will forget all the values proposed by Q0 and Q1 between the lines L4 and L8.

REMARK 1: this analysis applies in the case of a single cut,

M + A   A* + N

and gives an alternative proof for cut-elimination for arithmetic which is directly by induction on the depth of the cut-formulae A, different from Gentzen's argument, which reduces the problem to a multiple cut

(M + A + A(1) CUT A(1)* + N) CUT A* + N.

REMARK 2: if A0 or A1 is existential, the choice of te protocol is forced.

REMARK 3: the key here was to base the description on the depth of formulae. If we try to give a direct description of what the protocol is doing in general, it will be difficult, since for depth = 2, it involves backtracking, but for depth = 3, it involves backtracking in the backtracking of depth 2, and so on.

Already for depth = 3, the protocol gives a "concurrent" way of reducing a multiple cut

(x)(E y)(z)P0(x,y,z), (x)(E y)(z)P1(x,y,z)   (*)

that would be difficult to describe in term of cuts in the sequent calculus.

QUESTION: to find a simple example, generalising the 0/1 example, where we have a cut on a formula of depth 3. It involves finding a lemma of the form (*), proved by classical means, generalising the lemma P used in the 0/1 example (the "infinite box principle": there are infinitely many 0 or infinitely many 1).

QUESTION: is there a known formalism, in concurrency or elsewhere, in which such game-theoretic problems and winning strategies and "dialogue" are naturally expressed, and in which one can elegantly formalise termination argument.

REFERENCES:

[1] "A semantics of evidence for classical arthmetic", Th. Coquand, in the proceeding of Logical Framework, Edinburgh 1991, Eds G. Huet, G. Plotkin and C. Jones.

[2] Extracting Cosntructive Content from Classical Proofs, Ch. Murthy, PhD thesis, Cornell U, 1990.

[3] Collected Work of G. Gentzen, Szabo Ed, North Holland, 1969.

[4] Extraction de programmes a partir de preuves classiques, etude d'un exemple simple, magistere, H. Herbelin, 1991.

[5] Ein dialogisches Konskruktivitatskriterium, Lorenzen, in Infinistic Methods, Proceed. Symp. Found. of Math., PWN, Warszawa, pp. 193-200.