School of Computer Science

University of St Andrews

Automating Cut Admissibility Proofs in Sequent Calculi

Peter Chapman, Roy Dyckhoff

School of Computer Science, University of St Andrews

{pc,rd}@cs.st-andrews.ac.uk

Introduction

The property of Cut admissibility, or Haupstatz, is extremely useful in a Sequent Calculus, and is deeply related to normalisation in an appropriate Lambda Calculus, via the Curry-Howard Correspondence [3]. A calculus in which Cut is admissible is much better suited to proof search than one in which the removal of the Cut rule would change the set of provable theorems. Therefore, proving Cut admissibility is of great interest for proof theorists.

A completeness proof can be used [2], in which we show that the addition of the Cut rule to a system will not alter that system, in terms of which formulae are derivable. This is somehow unsatisfactory, since given a deduction in the calculus in question, containing instances of the Cut rule, we only know it can be replaced with a deduction of the same formula that uses no Cuts; it does not, however, provide us with a method for constructing this new deduction. So, of primary interest are constructive Cut admissibility proofs, where we give rules for transforming one deduction, containing Cuts, into another, which is Cut-free.

Such proofs proceed by induction on a variety of factors, for instance the weight of a formula or the level of a Cut[4], and then by case analysis. The large variety of cases means that the proofs are often tedious. The main focus of this work is to move towards an semi-automated procedure for providing these constructive Cut admissibility proofs.

The project will fall naturally into two sections. The first will be the foundational aspects of the problem [1], answering questions such as "What restrictions do we need to place on the calculus?" and examining how the number of cases needed to be analysed can be restricted. The second stage will then be to implement this in a dependently-typed framework, such as ISABELLE *or* EPIGRAM.

URL: http://www-old.cs.st-andrews.ac.uk/~pc/

Example – Rules of Gentzen System G3i

In what follows, P ranges over propositional formulae, Γ over multisets of formulae, and A,B,C are formulae.

$$P, \Gamma \Rightarrow P \quad (Ax) \qquad\qquad \bot, \Gamma \Rightarrow A \quad (L\bot)$$

$$\dfrac{A, B, \Gamma \Rightarrow C}{A \wedge B, \Gamma \Rightarrow C} L\wedge \qquad\qquad \dfrac{\Gamma \Rightarrow A \quad \Gamma \Rightarrow B}{\Gamma \Rightarrow A \wedge B} R\wedge$$

$$\dfrac{A, \Gamma \Rightarrow C \quad B, \Gamma \Rightarrow C}{A \vee B, \Gamma \Rightarrow C} L\vee \qquad\qquad \dfrac{\Gamma \Rightarrow A_i}{\Gamma \Rightarrow A_0 \vee A_1} R\vee, (i = 0, 1)$$

$$\dfrac{A \supset B, \Gamma \Rightarrow A \quad B, \Gamma \Rightarrow C}{A \supset B, \Gamma \Rightarrow C} L\supset \qquad\qquad \dfrac{A, \Gamma \Rightarrow B}{\Gamma \Rightarrow A \supset B} R\supset$$

$$\dfrac{\forall x A, [t/x]A, \Gamma \Rightarrow C}{\forall x A, \Gamma \Rightarrow C} L\forall \qquad\qquad \dfrac{\Gamma \Rightarrow [y/x]A}{\Gamma \Rightarrow \forall x A} R\forall$$

$$\dfrac{[y/x]A, \Gamma \Rightarrow C}{\exists x A, \Gamma \Rightarrow C} L\exists \qquad\qquad \dfrac{\Gamma \Rightarrow [t/x]A}{\Gamma \Rightarrow \exists x A} R\exists$$

$$\dfrac{\Gamma \Rightarrow A \quad A, \Gamma \Rightarrow C}{\Gamma \Rightarrow C} Cut$$

where we have y not free in the conclusions of $L\exists$ and $R\forall$.

The Cut rule is of particular importance. It expresses a form of transitivity; if A can be derived from the context Γ (left premiss), and also C can be derived from Γ and the formula A (right premiss), we can cut the formula A from our premisses and simply conclude that C can be derived from Γ. However, the Cut rule is harmful to proof search; the formula A can not be determined from the conclusion of the Cut rule.

This particular calculus has absorbed the structural rules into the logical rules. The structural rules usually consist of Weakening and Contraction, whereby we can, respectively, add additional formulae to, and remove multiple copies of the same formula from, an antecedent or succedent.

Example – Cut Admissibility for G3i

We show the analysis of the case where $A_0 \wedge A_1$ is principal in both the left and right premisses of an instance of the Cut rule.

$$\dfrac{\dfrac{\Gamma \Rightarrow A_0 \quad \Gamma \Rightarrow A_1}{\Gamma \Rightarrow A_0 \wedge A_1} R\wedge \quad \dfrac{\Gamma, A_0, A_1 \Rightarrow C}{\Gamma, A_0 \wedge A_1 \Rightarrow C} L\wedge}{\Gamma \Rightarrow C} Cut$$

which becomes, upon transformation,

$$\dfrac{\Gamma \Rightarrow A_1 \quad \dfrac{\Gamma \Rightarrow A_0 \quad \Gamma, A_0, A_1 \Rightarrow C}{\Gamma, A_1 \Rightarrow C} Cut}{\Gamma \Rightarrow C} Cut$$

Because of the way the inductive proof works, we can invoke the Induction Hypothesis and replace the Cut on $A_0 \wedge A_1$ with Cuts on the smaller formulae A_0 and A_1.

To illustrate how many cases a constructive Cut admissibility proof must analyse, the proof for G3i has 18 cases:

- ***9 cases** where at least one of the premisses is an axiom*
- ***4 cases** where the Cut formula is non-principal in at least one of the premisses*
- ***5 cases** (for the three connectives and two quantifiers) where the Cut formula is principal*

Symmetry accounts for many of these cases, but it is clear that a lot of work, most of which is very similar, still needs to be done to manipulate deductions for all cases in the manner of the above transformation.

References

[1] Ciabattoni, A. and Terui, K. Towards a Semantic Characterisation of Cut-elimination. In *Cut-elimination in Classical and Nonclassical Logic, Studia Logica*, pages 96-119. Springer 2006.

[2] Hermant, O. Semantic Cut Elimination in the Intuitionistic Sequent Calculus. In *Typed Lambda Calculi and Applications, Lecture Notes in Computer Science*, pages 221-223. Springer-Berlin, 2005.

[3] Sørensen, M.H. and Urzyczyn, P. Lectures on the Curry-Howard Isomorphism. Studies in Logic and the Foundations of Mathematics, Volume 149. Elsevier, 2006.

[4] Troelstra, A.S. and Schwictenberg, H. Basic Proof Theory, Second Edition. Cambridge Tracts in Theoretical Computer Science, 2000.

This work is supported by an EPSRC Grant.