Notes prepared by
Stanley Burris
February 10, 1997

1 Comments on propositional proof systems

In Chapter II of **LMCS** we looked at the propositional proof system PC and resolution theorem proving. A number of ideas have been developed regarding just how one sets up the basic structure of a proof system. We will look at some of the main ones here.

1.1 Definition of a propositional logic

DEFINITION 1 A *propositional logic*[1] consists of

- S, a set of connectives
- $\mathbf{C}$, the associated algebra of connectives
- $\mathcal{X}$, a set of propositional variables
- $\mathcal{C}$, a set of propositional constants
- $\mathrm{T}(\mathcal{X})$, the set of propositional formulas over $\mathcal{X}$
- Axioms
- Rules of inference.

DEFINITION 2 Given a set of propositional formulas Σ and a propositional formula φ we say

- $\Sigma \models \varphi$, read "φ is a consequence of Σ" if every truth-table evaluation which makes all formulas in Σ true also makes φ true.
- $\Sigma \vdash \varphi$, read "φ can be derived from Σ", if there is a derivation $\varphi_1, \ldots, \varphi_n$ of φ, i.e., each φ_i is either an axiom, or a member of Σ, or the result of applying a rule of inference to previous φ_j's, and $\varphi_n = \varphi$.

[1] We consider only two-valued propositional logics in **LMCS**. And to be more precise, we work only with Frege/Hilbert and resolution propositional logics. However some other possibilites are discussed in this addendum to the book.

1.2 Algorithms for basic questions about the propositional calculi

During the early development of propositional calculi, especially in the Warsaw school of the late 1920's, three basic questions came to the surface — and in 1946, after Tarski had moved to the U.S., he posed them to his American audience. In the following we assume that we are working with propositional calculi whose only rule of inference is modus ponens, and which have a finite set Σ of axiom schemata.

Q1: Is there an algorithm to determine if Σ is sound and complete?

Q2: Is there an algorithm to determine if Σ is an independent set of schemata?[2]

Q3: For each finite Σ is there an algorithm to determine which propositional formulas are derivable?

In 1949 Lineal & Post [4] answered all three questions in the negative — such algorithms do not exist for Q1 and Q2, and there is a finite Σ for which there is no algorithm to determine which formulas can be derived. Perhaps this makes the difficulty of the proof of the completeness of Frege/Łukasiewicz seem a little less surprising.

In the late 1920's Łukasiewicz indicated that he was going to write a comprehensive account of research on the propositional calculi — obviously such a work would consist of a number of special cases, with no clear pattern of what to expect in a sound and compete calculus.

He was also interested in finding independent sets of schemata, and the method of 'matrices' was developed by Bernays and him for this purpose. The basic idea to show that a schema σ cannot be derived from $\Sigma \setminus \{\sigma\}$ is to find an algebra $\mathbf{A} = (\mathrm{A}, S)$, where S is the set of connectives, and an endomorphism $\varepsilon : \mathbf{T}^S \longrightarrow \mathbf{A}$ such that

i. $\varepsilon(\varphi)$ has some property $\mathcal{P}$ for each substitution instance of each schema different from σ

ii. $\mathcal{P}$ is preserved by the rules of inference

iii. $\varepsilon(\sigma)$ does not have $\mathcal{P}$.

[2] Σ is independent if the removal of any schema leads to a smaller set of derivable propositions.

In view of the answer to Q2 this method was doomed to be fragmentary. Nonetheless it is quite useful. Let us use it to show:

EXAMPLE 3 A3 cannot be derived from A1 and A2 in Frege/Łukasiewicz propositional calculus.

We can let A $= \{0, 1\}$ and let the connectives $\neg, \rightarrow$ be associated to the functions:

P	$\neg P$
1	1
0	0

P	Q	$P \rightarrow Q$
1	1	1
1	0	0
0	1	1
0	0	1

Let $\varepsilon : \mathbf{T} \longrightarrow \mathbf{A}$ be such that $\varepsilon(P) = 0$ and $\varepsilon(Q) = 1$. Then any instance φ of A1 or A2 is such that $\varepsilon(\varphi) = 1$. Also if $\varepsilon(\varphi) = \varepsilon(\varphi \rightarrow \psi) = 1$ then $\varepsilon(\psi) = 1$. But A3, which is $(\neg P \rightarrow \neg Q) \rightarrow (Q \rightarrow P)$, maps to 0 under ε.

Exercises

Problem 1 Show the Frege/Łukasiewicz axioms are independent.
[Hint: A 3-element matrix will work in each of the two cases remaining.]

Problem 2 Show the there exists a finite set of independent propositional axioms Σ such that one *cannot* use finite matrices to prove their independence.

1.3 Formal systems, proof checkers, and theorem proving

Once we have a formal system, i.e., axioms and rules of inference, it is fairly routine to write a program to check if a sequence of formulas is indeed a derivation. And such programs are usually rather fast. The best known example is Automath, developed by de Bruijn in the Netherlands. Automath has been used to check all proofs in Landau's famous elementary book Grundlagen der Analysis. Once one has a proof checker it is straightforward to consider the possibility of using the proof checker to build an automated theorem prover.

The most primitive version of a theorem prover would be to start generating all possible finite strings and check to see which are derivations. Each time we find a derivation we add the conclusion to our bag of theorems. If our formal system is complete then every theorem will turn up sooner or later. However attractive this might sound, we are rapidly defeated by large numbers.

Suppose we have only two symbols, and we want to look at all sequences of at most 10 strings, each string of length at most 10, to see which are derivations. Then we will have

$$\sum_{i=1}^{10} 2^i = 2,046$$

many distinct strings to consider, and then

$$\sum_{i=1}^{10} 2,046^i > 10^{33}$$

distinct lists of strings to consider, which is impossible.

The most obvious way to cut down on the time demand is to only look at strings which are well defined formulas for our formal system. Generating well defined formulas is quite straightforward – we can make this simpler by putting some restriction on the number of variables we would like to consider. Then, in the propositional calculus for example, we would take the variables and generate new formulas by closing under the propositional connectives.

In our propositional logics we stated that formulas can be thought of as the terms for the algebra of connectives. So let us give a general recursion for the number of terms of an algebraic system with at most v variables and of length at most l, where we measure the length of a term by putting it in prefix form and counting the number of symbols occurring. Thus if our language has m_0 constant symbols and m_i function symbols of arity i, for $i > 0$, then, letting $terms(n)$ be the number of terms of length n, we have:

$$\begin{aligned} terms(1) &= m_0 + v \\ terms(n) &= \sum_{i>0} \left(n_i \cdot \sum_{l_1+\cdots+l_i=n-1} terms(l_1)\cdots terms(l_i) \right) \quad \text{for } n > 1. \end{aligned}$$

As an example let us look at the Frege/Łukasiewicz propositional logic. We have one binary connective ($\rightarrow$) and one unary connective ($\neg$). Thus $m_0 = 0$, $m_1 = 1$, $m_2 = 1$, and $m_i = 0$ for $i > 2$. Using the above formulas we obtain the following values for $terms(n)$ in the cases $1 \leq v \leq 5$:

For $v = 1$

n	$terms(n)$
1	1
2	1
3	2
4	4
5	9
6	21
7	51
8	127
9	323
10	835

For $v = 2$

n	$terms(n)$
1	2
2	2
3	6
4	14
5	42
6	122
7	382
8	1 206
9	3 922
10	12 914

For $v = 3$

n	$terms(n)$
1	3
2	3
3	12
4	30
5	111
6	363
7	1 353
8	4 917
9	18 777
10	71 769

For $v = 4$

n	$terms(n)$
1	4
2	4
3	20
4	52
5	228
6	804
7	3 444
8	13 780
9	59 588
10	253 252

For $v = 5$

n	$terms(n)$
1	5
2	5
3	30
4	80
5	405
6	1 505
7	7 255
8	31 155
9	149 455
10	688 655

So if we wanted to sift through all lists of at most ten propositional formulas of length at most 10, even in one variable, we would have more that 10^{29} cases, so again we have impossibility. Proof systems in the propositional calculus offer an alternative to truth tables – but it seems that if they are to be used for computer theorem proving then one will need to be clever.

1.4 Frege/Hilbert propositional calculi

If we abstract from the various propositional calculi that we have encountered we see that the key notions are: propositional formula, axioms, rules of derivation, derivation, and derived formula. The propositional formulas depend on the choice of connectives. The axioms and most rules can be put in the form of schemata

$$\frac{\psi_1, \ldots, \psi_k}{\psi}.$$

If $k = 0$ we have an axiom, and if $k > 0$ we have a rule. The preferred approach to substitution (since von Neumann, 1927) is to treat the axioms and rules as schemata, i.e., one is allowed to use any substitution instance of them, but not to use substitution as an rule of inference. (Note that substitution cannot be expressed as a scheme of the above form.) A derivation is then a finite sequence of formulas $\varphi_1, \ldots, \varphi_n$ such that each φ_i is either an axiom or the result of applying a rule of inference to previous steps. The last step φ_n is the derived formula, and we write $\vdash \varphi_n$. The notation $\varphi_1, \ldots, \varphi_k \vdash \varphi$ means one has a derivation of φ from $\varphi_1, \ldots, \varphi_k$, i.e., the definition of derivation is now extended to permit the appearance of any of the formulas $\varphi_1, \ldots, \varphi_k$ in the derivation sequence. Any such propositional calculus will be called a *Frege/Hilbert propositional calculus*. We note that since substitution is not expressible as a rule (as defined above) we have the unfortunate situation that Frege's propositional calculus does not qualify as a modern Frege/Hilbert propositional calculus.

Given a Frege/Hilbert propositional calculus we say it is *sound* if $\vdash \varphi \Longrightarrow \models \varphi$, it is *complete* if $\models \varphi \Longrightarrow \vdash \varphi$, and it is *implicationally complete* if $\varphi_1, \ldots, \varphi_k \models \varphi \Longrightarrow \varphi_1, \ldots, \varphi_k \vdash \varphi$. A *Frege system* (see Cook and Rechkow [1],[2]) is an implicationally complete and sound Frege/Hilbert propositional calculus with an adequate set of connectives, each of which has at most two arguments (for example the propositional calculus PC).

One way to strengthen a Frege system is to permit extension by definitions, i.e., one is allowed to use definitions of the form $P \leftrightarrow \varphi$ in a derivation, where P does not occur in φ or any previous step. Such propositional calculi are called *extended Frege systems*. In extended Frege systems one can find short (i.e., polynomially bounded) derivations of the pigeonhole tautologies.

A second way to strengthen a Frege system is to permit substitution as a rule of inference — however note that when carrying out a derivation to show $\Sigma \vdash \varphi$, one does not want to apply substitution to the members of Σ. This can be handled by treating the propositional variables of Σ as constants.

1.5 Natural deduction propositional calculi

The deduction lemma is a powerful tool of everyday mathematics, namely to show $\varphi \rightarrow \psi$ we assume φ and prove ψ. To incorporate this into our propositional calculus as a rule of inference forces us to modify our notion of what a line of a proof should look like, namely we need to accommodate arbitrary finite sets[3] of propositions Σ to express the deduction rule:

$$\frac{\Sigma \cup \{\varphi\} \Rightarrow \psi}{\Sigma \Rightarrow (\varphi \rightarrow \psi)}$$

Note that we have replaced $\vdash$ by a new symbol $\Rightarrow$ which is now a part of our formal language. A *(natural deduction) line* is an expression $\Sigma \Rightarrow \varphi$,

[3] One can also work with finite sequences rather than finite sets.

where Σ is a finite (possibly empty) set of formulas. A derivation is a finite sequence of natural deduction lines, each of which is either an axiom or results from applying a rule of inference to previous members of the sequence. A derivation of a formula φ is a derivation whose last member is the natural deduction line $\Rightarrow \varphi$.

The following is an example of a natural deduction propositional calculus obtained from Frege/Łukasiewicz propositional calculus:

AXIOMS:

$$(1)\ \frac{}{\Sigma \Rightarrow (\varphi \to (\psi \to \varphi))}$$

$$(2)\ \frac{}{\Sigma \Rightarrow ((\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi)))}$$

$$(3)\ \frac{}{\Sigma \Rightarrow ((\neg\varphi \to \neg\psi) \to (\psi \to \varphi))}$$

RULES OF INFERENCE:

$$(1)\ \frac{\Sigma \Rightarrow \varphi,\ \Sigma \Rightarrow (\varphi \to \psi)}{\Sigma \Rightarrow \psi}$$

$$(2)\ \frac{\Sigma \cup \{\varphi\} \Rightarrow \psi}{\Sigma \Rightarrow (\varphi \to \psi)}$$

$$(3)\ \frac{\Sigma \Rightarrow (\varphi \to \psi)}{\Sigma \cup \{\varphi\} \Rightarrow \psi}$$

Similarly one can transform any Frege/Hilbert propositional calculus into a natural deduction propositional calculus. One can define the notions of sound, complete and implicationally complete in an obvious manner, namely sound means $\vdash \Sigma \Rightarrow \varphi \Longrightarrow \Sigma \models \varphi$, complete means $\models \varphi \Longrightarrow \vdash \Rightarrow \varphi$, and implicationally complete means $\Sigma \models \varphi \Longrightarrow \vdash \Sigma \Rightarrow \varphi$. Cook and Reckhow define a *natural deduction system* to be a natural deduction propositional calculus which has a finite number of axioms and rules, an adequate set of connectives, and is sound and implicationally complete (such as the example above).

1.6 Gentzen propositional calculi

Gentzen [3] looked at expressions (called *sequents*) of the form $\Sigma \Rightarrow \Gamma$, where Σ and Γ are finite sets[4] of formulas. The intended interpretation is that an evaluation which makes all members of Σ true will make some member of Γ true. A derivation is a finite sequence of sequents such that each member is either an axiom or the result of applying one of the rules of inference to previous members. We will call such a propositional calculus a *Gentzen propositional calculus.* Such a calculus is sound if $\vdash \Sigma \Rightarrow \Gamma \Longrightarrow \Sigma \models \Gamma$, complete if $\models \Gamma \Longrightarrow \vdash \Gamma$, and implicationally complete if $\Sigma \models \Gamma \Longrightarrow \vdash \Sigma \Rightarrow \Gamma$. A *Gentzen system* is a Gentzen propositional calculus which has an adequate set of connectives, is sound and implicationally complete. One obvious way to obtain a Gentzen system is to take a natural deduction system and replace natural deduction lines in the axioms and rules of deduction by sequents, e.g., replace $\Sigma \Rightarrow \varphi$ by $\Sigma \Rightarrow \{\varphi\}$. Gentzen made use of the popular, and powerful, *cut* rule:

$$\text{CUT}\ \frac{\Sigma \Rightarrow \Gamma \cup \{\varphi\},\ \Sigma' \cup \{\varphi\} \Rightarrow \Gamma'}{\Sigma \cup \Sigma' \Rightarrow \Gamma \cup \Gamma'}$$

References

[1] S.A. Cook and R.A. Rechkow, On the lengths of proof in the propositional calculus, Preliminary version. *Proc. Sixth Annual ACM Symposium on Theory of Computing* (1974), 135–148. Corrections for "On the lengths of proofs in the propositional calculus". SIGACT News (1974), 15–22.

[2] S.A. Cook and R.A. Rechkow, The relative efficiency of propositional proof systems. J. Symbolic Logic **44** (1979), 36–50.

[3] G. Gentzen, Untersuchung über das logische Schliessen. Math. Z. **39** (1934), 176–210, 405–431.

[4] S. Lineal and E.L. Post, Recursive unsolvability of the deducibility, Tarski's completeness and independence of of axioms problems of propositional calculus (Abstract). Bull. AMS **55** (1949), 50.

[4] Also one can take sequences of formulas.