CSE303 Report

COMBINATORIAL STUDY OF ORDERED LINEAR SEQUENT CALCULUS

Aarrya Saraf supervised by Dr. Noam Zeilberger

February 13, 2024

Contents

1 Abstract

Fang [1] has recently described a bijection between rooted planar maps and normal linear ordered lambda terms. This bijection seems more natural than an earlier one described by Zeilberger and Giorgetti [2]. More precisely, it is easier to study from the point of view of logic since it relies on the correspondence between planar lambda terms and proofs in a very restrictive "logic of pure implications".
As it turns out, some formulas admit more than one proof and hence correspond to more than one planar map. In this project, we have further analyzed this bijection by creating a system to generate all possible formulae. The system also tells us whether, in our restrained logic, it is provable or not. If it is provable it also lists the number of proofs for the formula. Based on this we have tried to find patterns among the formulae to give necessary and sufficient or simply sufficient conditions to have a unique proof.

2 Sequent Calculus

2.1 Rules

Given our restrained system of logic and the corresponding lambda terms, we worked with the following rules in our Sequent Calculus.

- Identity Rule:

$$\frac{}{o \vdash o}\,\mathrm{I}$$

- Right Rule

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash (A \multimap B)}\,\mathrm{R}$$

- Left Rule

$$\frac{\Gamma \vdash A \qquad B, \Delta \vdash o}{(A \multimap B), \Gamma, \Delta \vdash o}\,\mathrm{L}$$

2.2 Notes

- Small letters are for atomic formulae and hence the identity rule can be applied only to atomic formulae, and before applying the left rule we must ensure the right-hand side is an atom. Furthermore, on the right sequent after the left rule, the right side is always an atom as well.

- Our system is ordered and we cannot freely interchange two sequents

- Proofs are not necessarily unique. This is because of the potential ambiguity in the choice of Δ and Γ in the left rule. In no other place can we have a different proof tree.

- This structure also means that at each time we can apply only 1 of the 3 rules and it will always be in the order identity followed by right followed by left recursively until we reach a point where we prove it or are unable to prove it.

- A sequent cannot be proved iff no rules can be applied to it. For example $o, o \vdash o$ and $a \vdash b$.

2.3 Illustrations

Let us show some sequents and their proof(s). We define the two sequents beforehand

$$A := (((o \multimap o) \multimap (o \multimap o)) \multimap o) \multimap ((o \multimap o) \multimap o)$$

$$B := ((o \multimap o) \multimap ((o \multimap o) \multimap o)) \multimap ((o \multimap o) \multimap o)$$

Notice that the Formula more or less remains the same: the only change is in the bracketing.

2.3.1 Proof of A

$$\begin{prooftree}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\RightLabel{L}
\BinaryInfC{$(o \multimap o), o \vdash o$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\RightLabel{L}
\BinaryInfC{$(o \multimap o), (o \multimap o), o \vdash o$}
\RightLabel{R^2}
\UnaryInfC{$(o \multimap o) \vdash ((o \multimap o) \multimap (o \multimap o))$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\RightLabel{L}
\BinaryInfC{$(((o \multimap o) \multimap (o \multimap o)) \multimap o), (o \multimap o) \vdash o$}
\RightLabel{R^2}
\UnaryInfC{$\vdash (((o \multimap o) \multimap (o \multimap o)) \multimap o) \multimap ((o \multimap o) \multimap o)$}
\end{prooftree}$$

2.3.2 First Proof of B

$$\begin{prooftree}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\RightLabel{L}
\BinaryInfC{$o \multimap o, o \vdash o$}
\RightLabel{R}
\UnaryInfC{$o \multimap o \vdash o \multimap o$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\RightLabel{R}
\UnaryInfC{$\vdash o \multimap o$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\RightLabel{L}
\BinaryInfC{$(o \multimap o) \multimap o \vdash o$}
\RightLabel{L}
\BinaryInfC{$((o \multimap o) \multimap ((o \multimap o) \multimap o)), (o \multimap o) \vdash o$}
\RightLabel{R^2}
\UnaryInfC{$\vdash ((o \multimap o) \multimap ((o \multimap o) \multimap o)) \multimap ((o \multimap o) \multimap o)$}
\end{prooftree}$$

2.3.3 Second Proof of B

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\RightLabel{R}
\UnaryInfC{$\vdash o \multimap o$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\RightLabel{L}
\BinaryInfC{$o \multimap o, o \vdash o$}
\RightLabel{R}
\UnaryInfC{$o \multimap o \vdash o \multimap o$}
\AxiomC{}
\RightLabel{I}
\UnaryInfC{$o \vdash o$}
\RightLabel{L}
\BinaryInfC{$(o \multimap o) \multimap o \vdash o$}
\RightLabel{L}
\BinaryInfC{$((o \multimap o) \multimap ((o \multimap o) \multimap o)), (o \multimap o) \vdash o$}
\RightLabel{R^2}
\UnaryInfC{$\vdash ((o \multimap o) \multimap ((o \multimap o) \multimap o)) \multimap ((o \multimap o) \multimap o)$}
\end{prooftree}
$$

As a last remark, we can also see that

$$((((o \multimap o) \multimap (o \multimap o)) \multimap o) \multimap (o \multimap o)) \multimap o$$

is not provable (and hence false) but also shares the same structure with a different bracketing.

2.4 A Prover

Working on such a constrained logic made this task highly repetitive and simple sometimes. To find patterns in this, one approach would be to have the provability (and uniqueness of the proof) for all different formulae and try and identify patterns there. To that end, we wrote code that could take in a sequent and prove whether it is true or not. The language chosen for this is Haskell because of both convenience and performance reasons: the lazy evaluation and the thunks make the code much faster since calculations are often repetitive, but pattern matching makes code much smaller compared to Python for example.
The approach was to put the sequent in a list and apply the identity rule first. If applicable we can stop there. If not we apply the right rule as many times as possible. After this we apply all possible left rules and if any of them evaluate to true we are done.
A small change done towards the end was assigning a sequent the value of the number of distinct proofs it has if true and 0 if false. Then instead of running an or over all the boolean evaluation of the subsequents after a left rule, we could add the integer evaluations after a left rule and this would give us the number of proofs. If this number is 0 the sequent is not provable. If it is 1 then the proof is unique. If it is more than 1 then there are that many possible proofs.

3 Generating Sequents

With a prover ready if we could generate all the possible sequents then we could figure out the pattern. The following idea was essential to it

3.1 Generating Binary Trees

Since we have only 1 possible binary operator, generating a sequent with n atoms is equivalent to generating a binary tree with n leaves. This can be done by replacing the nodes with linear implications. This is something we have put before and can be done recursively. Here is the pseudo-code for that

Algorithm 1 bingen

```
Require: n ≥ 1
  arr ←[]
  i ← 0
  while i ≤ n do
      l ← bingen(i)
      r ← bingen(n − i)
      while lt ∈ l do
          while rt ∈ r do
              arr append (Node (lt)(rt))
          end while
      end while
      i ← i + 1
  end while
```

The lazy evaluation of Haskell makes this faster than the typical $\mathcal{O}(C_{n-1})$ where C_n denotes the nth Catalan number.

However, this just gives us the general structure. It is still upon us to decide the atoms and putting them differently. For example we $a \multimap b$ is very different from $a \multimap a$ and hence we had to generate the atoms which would then replace the leaves

3.2 Generating Atoms

3.2.1 A Naive Approach

There are many naive approaches to do this but all of them have the same problem. One of the Algorithms is given below.

For $n > 0$ atoms, define a list arr with n of each number from 0 to $n-1$. Then from this list of n^2 elements, we pick all possible combinations of n elements. The problem with this algorithm is the following- The time complexity is $\mathcal{O}(C_{n-1}n^n)$ but we can do better than this. This is because

$$a \multimap a$$

and

$$b \multimap b$$

are equivalent.

3.2.2 A Better Approach

Instead, we can look at all possible partitions of the list and assign each partition the same letter. This is much better as this follows the Bell numbers $(B_n)_n, n \in \mathbb{N}$. [3]. Again, Haskell is much faster and concise.

```
appender :: a -> [[a]] -> [[[a]]]
appender x lists = zipWith (\i lst -> take i lists ++
                   [lst ++ [x]] ++
                   drop (i+1) lists) [0..] lists

applier :: Int -> [[[Int]]] -> [[[Int]]]
applier x [] = []
applier x (y:ys) = (appender x y) ++( ([x]:y):
                   (applier x ys))

part :: [Int] -> [[[Int]]]
part [] = [[]]
part [x] = [[[x]]]
part [x, y] = [[[x], [y]], [[x, y]]]
part (x:xy) = applier x (part xy)
```

With this, we generate formulae that are all different, and we bring down our complexity to $\mathcal{O}(C_{n-1}B_n)$

4 Final Remarks

4.1 Experiments

With the system ready, the following 3 things were used as parameters to find patterns. There are essentially 3 broad things we can vary in order to find patterns.

- Size of the Sequent or the number of atoms: As seen later, for an odd number of atoms we find 0 proofs.
- Number of distinct atoms: A formula with n atoms all of which are distinct can never be true. Experiments were conducted where we set the number of atoms to 1.
- Structure of the tree: Lastly, the arrangement is what can affect the proof and this is what prevents us from having a map from $\mathbb{N}$ to $\mathbb{N}$, but structures can be fixed. Consider

$$D_{n\geq 3} = \underbrace{(o \multimap \ldots (o \multimap}_{n-2 \text{ times}}(o \multimap o))) \multimap \underbrace{(o \multimap \ldots (o \multimap}_{n-2 \text{ times}}(o \multimap o)))$$

It can be shown that this has a unique proof for all such n.

Keeping these factors, in mind the following experiments were performed:

- Experiment 1: We generate different tree structures for a fixed size and assign each root the same atom. We then take the sum of the number of proofs of all these trees and generate our sequence by increasing the size which is as follows:

  $$0, 1, 0, 2, 0, 9, 0, 54, 0, 378, 0, 2916, 0, 24057$$

  By plugging this into the encyclopedia [4], this time we obtain that the sequence $(a_n)_n, n \in \mathbb{N}$ is defined as follows:

  $$a_n = \begin{cases} \frac{2(3^k(2k)!)}{k!(k+2)!} & \text{where } k = \frac{n}{2} - 1, \text{if } x \equiv 0 \pmod 2 \\ 0, & \text{otherwise} \end{cases}$$

  The odd terms correspond to the number of rooted planar maps with k edges. This is in fact the initial bijection found by Zeilberger [2]. This was done as a verification for our system.

- Experiment 2: In this experiment we tried to find patterns by taking the sum of all possible proofs over all possible formulae with a fixed size (so we vary the trees and the atoms). Since this grows so fast not many results could be calculated but we obtained the following sequence of length 8

  $$0, 1, 0, 4, 0, 45, 0, 810$$

  It is clear that with an odd number of atoms, we cannot get any proofs and this is very intuitive. Moreover, From the encyclopedia of integer sequences [4] this does not follow any standard known sequence. However, upon looking more closely one realizes that this is somehow related to the initial sequence found but we need to multiply it by some factor (even terms only). The sequence of the multiplication factor is as follows

  $$1, 2, 5, 15...$$

  This in fact gives us the Bell Numbers. We recall this was the total number of partitions for the set. It is slightly more strange though. Recall that we are not considering the odd formulae and hence the number of trees is growing twice as fast. There appears to be some connection here.

- Experiment 3: Since the second experiment was successful but mysterious, we tried to replicate this for n distinct atomic formulae in general and see if any pattern springs out from that. We have listed the number of proofs and the number of distinct atoms(Q) below

Table of number of proofs		
Q	# Sequents	# Proofs
1	[1,1,2,5,14,42,132,429,1430,4862,16796,58786,208012, 742900]	[0,1,0,2,0,9,0,54,0,378,0,2916,0,24057]
2	[0,1,6,35,210,1302,8316,54483,364650, 2484482]	[0,0,0,2,0,27,0,378,0,5670]
3	[0,0,1,30,350,3780,39732,414414]	[0,0,0,0,0,9,0,324,0,9450]
4	[0,0,0,1,140,2730,46200,729729]	[0,0,0,0,0,0,0,0,54,0,3780]
5		[0,0,0,0,0,0,0,0,0,0,378]

Upon closely examining the sequences we find further patterns here. Firstly, we notice that, in general, with Q distinct atoms and $2Q$ sequents we re-obtain the sequence that follows the number of rooted planar maps found before.
Moreover, our sequences for $Q > 1$ seem to be related to $Q = 1$ in a pattern. Observe that for $Q = 2$ the number of proofs for size 4 remains the same. For size 6 it triples. This is because we can find 3 generalized sequents for every sequent that was true in the $Q = 1$ case. For example, if

$$(a \multimap a) \multimap (a \multimap a)$$

then we could generalise this by saying

$$(a \multimap b) \multimap (a \multimap b)$$

For the size 8 it is in fact 7 times that for 1 and for 10 it is 15 times that. Indeed we notice a sequence $(2^m - 1)\left(\frac{2(3^k(2k)!)}{k!(k+2)!}\right)$ where $m = \frac{\textbf{\# of atoms}}{2} - 1$
We also observe that for the size 3 we get the pattern $1, 6, 15...$ if we look at this in context with size 2 we also obtain the triangle read by rows sequence (A354977 [4]). The same goes for the case of size 4 where we obtain the sequence $1, 10$. However, these limit the sizes and say nothing about the size of say 10 with 2 distinct atoms. Hence looking further into this we conclude that the nth entry of column k is given by $S(n + k, k)$ which are the Sterling numbers of the second kind. This also means that running a sum over the column gives us the pattern for the sequence generated in experiment 1: This means that for all possible sequents of a certain size, we will have

$$\sum_{i=1}^{i=\frac{n}{2}} S(n+i, n)\left(\frac{2(3^n(2n)!)}{n!(n+2)!}\right)$$

Further, the proof of this property can be seen using [5] where we say that a proof for a sequent is a specialized version of a proof for a balanced sequent. In our case, we can identify some atoms and this follows the $S(n + k, k)$ sequence discussed earlier.
It is also to be noted that after a certain size generating all the sequents

is not possible without much better computers. Hence we filtered the number of sequents by considering only the ones that have at least two of every variable. Hence while we couldn't provide the number of sequents we could still analyse the answer.
This is a new and interesting result we need to study more. One example might be to consider the original bijection with rooted planar maps with a "half-open" edge instead which by Zeilberger [6] is also in bijection with rooted planar maps. By doing so we have that the number of edges in the map is equal to the number of atoms in our sequent and we can consider partitions of those to be corresponding to the proofs.

- Experiment 4: With this done, we fixed the number of distinct atoms to 1, and for each sequent with $n, n \in \mathbb{N}$ we noted the number of proofs. The following results were obtained.

$$
\begin{aligned}
&[1, 0...] \\
&[0, 1, 0...] \\
&[2, 0...] \\
&[3, 2, 0...] \\
&[14, 0...] \\
&[33, 9, 0...] \\
&[132, 0...] \\
&[377, 50, 2, 0...] \\
&[1430, 0...] \\
&[4518, 314, 26, 4, 0...] \\
&[16796, 0...] \\
&[56304, 2137, 270, 67, 4, 2, 2, 0...] \\
&[208012, 0...] \\
&[723872, 15398, 2614, 815, 106, 42, 39, 6, 0, 4, 4, 0...] \\
&[2674440, 0...]
\end{aligned}
$$

The odd terms had no proofs and we can discard them while looking at the patterns. We thus obtain this table:

Table of number of proofs											
size	0	1	2	3	4	5	6	7	8	9	10
2	0	1	0	0	0	0	0	0	0	0	0
4	3	2	0	0	0	0	0	0	0	0	0
6	33	9	0	0	0	0	0	0	0	0	0
8	377	50	2	0	0	0	0	0	0	0	0
10	4518	314	26	4	0	0	0	0	0	0	0
12	56304	2137	270	67	4	2	2	0	0	0	0
14	723872	15398	2614	815	106	42	39	6	0	4	4

The verification for this is easy. The sum of each row follows C_n and the weighted sum for each row follows the sequence found in 3. Hence, every odd row will be of the form $R_{2n+1} = [C_{2n+1}, 0...]$. As for even rows, we were unable to find any patterns.
The next attempt was to check for the ratio of the number of unprovable sequents to the total number of sequents. With that, we observe the following sequence:

$$0, \frac{3}{5}, \frac{33}{42}, \frac{377}{429}, \frac{4518}{4862}, \frac{56304}{58786}, \frac{723872}{742900}, \frac{9xxxxxx}{9694845} \dots$$

- Experiment 5: Since we are dealing with binary trees here, one interesting way to find relations might be using the associahedron [7]: a polytope with nodes that represent different binary trees and edges that relate them. This however would require a lot more prerequisite knowledge and time which was not available to us.

- Experiment 6: The only reason we cannot have a sequence is because of the branching of the binary trees. Hence if we can fix a structure for binary trees and a way to maintain that structure by adding more items then we could construct a sequence and try to find the proofs associated with them. Again this would require a lot more time and could not be completed.

4.2 Conclusion

To summarise, we wanted to analyze a bijection between certain lambda terms and certain trees. By Fang, [1], we could reduce this problem to the proofs of sequents in a restricted logic system. To better study that, we built a system that produces all possible formulae with n atoms and then checks whether it is provable or not, and if provable it also checks if the proof is unique. With the system built, we conducted experiments; What we did find is very interesting. We show that using Sterling numbers and the number of rooted planar graphs we can predict the number of proofs of any sequent of any length with a certain number of distinct atoms. Indeed, by summing over all these distinct numbers we have the proofs for all the sequents. This is a new finding and some further analysis needs to be done on this interesting connection. Further, we have also mentioned additional experiments which are now easier to conduct based on the made tools.

5 Appendix

- Lambda Calculus: The lambda calculus is a theory of functions as formulas. It is a system for manipulating functions as expressions. For example, The expression $\lambda x.x^2$ stands for the function that maps x to x^2. An occurrence of a variable x inside a term of the form $\lambda x.N$ is said to be bound. [8]

- Normal Linear Ordered Lambda Term: A Lambda term where each bound variable is used exactly once and exactly in the order. Further, a term is normal if it contains no β-redexes For example, $\lambda x \lambda y x + y$ is valid but $\lambda x \lambda y x^2 + y$ or $\lambda x \lambda y y + x$ are not. [9]

- Linear Logic: Linear logic is a refinement of classical and intuitionistic logic. Instead of emphasizing truth, as in classical logic, or proof, as in intuitionistic logic, linear logic emphasizes the role of formulas as resources. To achieve this focus, linear logic does not allow the usual structural rules of contraction and weakening to apply to all formulas but only those formulas marked with certain modals. Linear logic contains a fully involutive negation while maintaining a strong constructive interpretation. Linear logic also provides new insights into the nature of proofs in both classical and intuitionistic logic. Given its focus on resources, linear logic has found many applications in Computer Science. What is important to note is that for our project we have considered linear implication as the only operation in this logic. [10, 11]

- Rooted Planar Map: A rooted planar map is a connected graph embedded in the 2-sphere, with one edge marked and assigned an orientation. [2]

- Linearly Implies ($\multimap$) the linear implication $B \multimap C$ can be defined as $B^{\perp} \invamp C$, while the intuitionistic implication $B \Rightarrow C$ can be defined as $!B \multimap C$.

- Sterling Numbers Stirling number of the second kind (or Stirling partition number) is the number of ways to partition a set of n objects into k [12]

References

[1] Wenjie Fang. Bijections between planar maps and planar linear normal λ-terms with connectivity condition, 2023.

[2] Noam Zeilberger and Alain Giorgetti. A correspondence between rooted planar maps and normal planar lambda terms. *Logical Methods in Computer Science*, Volume 11, Issue 3, September 2015.

[3] Paul Tarau and Valeria de Paiva. Deriving theorems in implicational linear logic, declaratively. *Electronic Proceedings in Theoretical Computer Science*, 325:110–123, September 2020.

[4] Oeis: The on-line encyclopedia of integer seqeunces.

[5] Harry G. Mairson. Linear lambda calculus and ptime-completeness. *J. Funct. Program.*, 14(6):623–633, nov 2004.

[6] Julien Courtiel, Karen Yeats, and Noam Zeilberger. Connected chord diagrams and bridgeless maps, 2017.

[7] Wikipedia contributors. Associahedron — Wikipedia, the free encyclopedia, 2023. [Online; accessed 7-December-2023].

[8] Peter Selinger. Lecture notes on the lambda calculus, 2013.

[9] Noam Zeilberger Anupam Das, Damiano Mazza and Le Thanh Dung Nguyen. On the complexity of normalization for the planar λ-calculus. Trends in Linear Logic and Applications, 2023.

[10] Frank Pfenning. Linear logic lectures at cmu, 2012.

[11] Roberto Di Cosmo and Dale Miller. Linear Logic. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Fall 2023 edition, 2023.

[12] Wikipedia contributors. Stirling numbers of the second kind — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Stirling_numbers_of_the_second_kind&oldid=1186379985`, 2023. [Online; accessed 7-December-2023].