Folie 1

Ladies and Gentlemen,

I will talk about S-terms in Combinatory Logic.

I ignore the word "logic"
and concentrate on the combinatory aspects.

Folie 2

So we have a term algebra
with a binary function symbol, called application
and some nullary symbols, called combinators.

We can rewrite such trees, because
each combinator has an associated reduction rule.
The fundamental combinators I, K, S are shown here.

Although they look innocent, SK-terms are complicated.
We can translate terms from lambda calculus into SK-terms,
and all results on undecidability carry over.

Now K-terms are indeed simple,
so the question was asked: how difficult are S-terms?

CL(S) has some nice properties: it is confluent, and it is non-erasing, so weak and strong normalization coincide.

On the other hand, S-terms have non-obvious behaviour.
Can you answer this question: which of these terms has a normal for

Folie 3

The second one has a normal form,
and the first one has not.

As all terms will be S-terms from now on,
I will just print a small circle instead of the letter S.

Folie 4

These are individual examples of S-terms.

Now I am going to present a whole set
of S-terms without normal form.
This will be a rational set,
given by a regular tree grammar.

Folie 5

I need some prerequisites.

Let M be the set of all terms,
and define Q1 and Q2 by excluding one or two small trees.

Then F is defined by this regular grammar.
The asterisk is shorthand for M, the set of all terms.
It's a "don't care" symbol.

You see that the term Q2 Q1 is essential (see the examples),
but it may be buried down the tree
by using some of the first two productions.

We are going to show that each term F F has an infinite reduction.

We need the lemma Q1 F <= F
which is shown by a simple case distinction.
Q1 has a left spine of length 1 or more,
(length 0 is forbidden).
In both cases Q1 F is in F, as is clear from the grammar.

Now we consider terms F F.

Folie 6

We have already seen this example, on the introductory slides.

To prove the proposition, we do a case distinction
as suggested by F's grammar.

In the first two cases, by direct application of the S rule
we see that a subterm F F is reached.

Also for the third case we are going to show
that we reach a subterm F F, after at least one reduction step.
This would then complete the proof.

Folie 7

We name the subterms X Y Z accordingly.

In case X alone has an infinite reduction, we are done.

Otherwise X has a normal form.
Since X was in Q2, its normal form must be in Q2 as well.

We verify the relation Q2 && N <= S Q1 ++ S any any.
A normal form from Q2 must have a left spine of length one or two.
Zero is forbidden.
If it is one, then the right child cannnot be S, so it must be Q1.

This gives two cases for X.
In the first case, we use the lemma from the previous slide,

and in the second case we apply the definitio of F.

So the proof is completed.

By similar arguments, we find two more rational sets
of non-normalizing S-terms.

Folie 8

Don't worry, I'm not going through the details.
We define Q3 and Q4 as an extension of Q1 and Q2.
Then we have propsitions Q3 Q2 Q1 ->> inf
and S Q3 any Q2 ->> inf.

If we read the asterisk as Q0, and S Q3 as Q4,
then there is a common pattern:
the sum of the indices in these three formulas is six.

I would really like to know if this does really say something,
or if it's just coincidence.

Next, I'm going to present sets of normalizing terms.
Again these will be given by regular grammars.

Folie 9

As in lambda calculus,
we call left children "active" or "functions",
and right children "passive" or "arguments".

We are investigating how terms behave
when supplied with an argument x.

The simple case is that x never gets activated,
as in this example.
I collect all terms like this boxed one
into the set NO.

The good thing is that NO is rational,
generated by this simple grammar here.

All terms NO N are normalizing,
where N denotes the set of all normal forms.
The reason is that the argument is completely ignored
during the reduction.

We extend this a little bit,
by allowing activation of arguments.

Folie 10

Now x may be activated,
but it shall be supplied with at most one argument,
as seen in the example.

Again we collect all terms for which this happens
into the set N01. Of course N0 is contained in N01.

And again, N01 turns out to be rational.
The grammar is a bit more complicated,
but still fits into one line.

Now I claim all N01 N0 terms are normalizing.
This can be seen from the example.
N1 N0 may produce redexes N0 any,
but they are normalizing by the previous proposition.
It does not hurt that these redexes may be nested.

However it produces longer reduction chains and larger results.
You have seen one already, on the introductory slides.

Now is the time to collect the results.

Folie 11

We have seen sets of terms with and without normal form.

Did we leave any terms out?
Yes, but not too many of them.

The remaining cases,
that is N N minus the sets we've already seen,
are exactly N1 N1 plus N2 P2.

This can be verified by manipulation of rational languages
by pen and paper, or by machine.
I've written a program RX that does such computations.

For N1, we use a case distinction L0 ++ L1 ++ L2.

For N2 P2 we do some reductions
until we reach one of the previous cases.

This gives a procedure that decides normalization,
provided that the input has form N N.

But this is enough: if it's not N N,
then we first check normalization of the subterms
by calling ourselves recursively.
We can stop if a subterm does not normalize,
and we compute the normal form otherwise.

So I've shown how I used rational languages
to decide normalization of S-terms.

Finally, let me mention some related problems.

Folie 12

Of course one is interested in the word problem,
that is, reachability under reduction.

I have a feeling that it's decidable,
but it still is a long way to go.

Statman solved the word problem for the L combinator.
L has only two arguments,
but the word problem required an ingenious and complicated proof.
This might be even harder for S, with 3 arguments.

Another point. I used rational languages,
but from the proof it's not obvious
whether the set of all normalizing S-terms is itself rational.

This is so because during the decision procedure,
we have to perform some reductions,
until we test membership in rational sets.

Still I conjecture the set in question is rational,
and I have a reasonable guess for its grammar.
I computed the canonical automaton.
Which turned out to be large.
And I checked closure under reduction.

Even if all of this is correct (I hope it is), it's still not very illuminating.

Well, I've shown you a lot of "ad hoc" constructions.
and it's not really clear why they worked,
at least not to me.

What is really missing here
is a general argument that says:
yes, the set of normalizing S-terms is rational,
and you can construct its grammar.

Thomas Genet, in his talk, will do such constructions.
But, we already checked that,
the approximations obtained by his methods
are not sharp enough for CL(S).

I see my results as a strong hint
that CL(S) belongs to a class of term rewriting systems
whose behaviour can still be described by rational tree languages,
and I'd really like to find out more about this class.

Thank-you and Good-bye