Semi-Thue Systems and Word Problems

Benjamin Maier

"Models of Computation" – Department of Philosophy

May 10, 2012

Universiteit Utrecht

Table of Contents

Universiteit Utrecht

Table of Contents

Universiteit Utrecht

Axel Thue (1863-1922)

- Norwegian mathematician
- studied long time in Germany, became professor in Norway
- worked mostly on Diophantine equations and number theory *($\rightarrow$ Hilbert's 10^{th} problem)*
- formulated the word problems, which later became subject of $\rightarrow$ *Combinatory Logic*

Axel Thue

http://commons.wikimedia.org/wiki/File:Axel_Thue.jpg

(Semi-)Thue Systems

- a semi-Thue system is a 2-tupel $T := (\Sigma, S)$
- Σ - alphabet
- $S \subseteq \Sigma^* \times \Sigma^*$ is a set of pairs of words (binary relations)

  $$S = \{(a_1, b_1), (a_2, b_2), ...\}$$

  called the **rewriting rules**
- let u, v, x, y be words, s.t. $u, v, x, y \in \Sigma^*$
  (note that the empty word ϵ may be part of Σ^*)
- a string uxv may be rewritten

  $$uxv \longrightarrow_S uyv$$

  if and only if $(x, y) \in S$ (uxv denotes the concatenation of the words u, x and v)
- a word y is called *derivable from* x *in* S or ($x \overset{*}{\longrightarrow}_S y$) if there exists a sequence of words

  $$uxv \longrightarrow_S uw_1v \longrightarrow_S ... \longrightarrow_S uw_nv \longrightarrow_S uyv$$
- original Thue systems: pair (x, y) meant $uxv \longleftrightarrow_S uyv$

Example

instance semi-Thue system $T = (\Sigma, S)$, with $\Sigma = \{a, b, c\}$,
$S = \{(a, c), (aa, b)\}$

question Is the word $y = cbbbc$ derivable from the word $x = abaabc$ in S $(x \xrightarrow{*}_S y)$?

- from $abaabc$ we can derive $abbbc$ in a single step
- from $abbbc$ we can derive $cbbbc$ in a single step
- therefore the word y is derivable from x

Table of Contents

Universiteit Utrecht

Semi-Thue Systems as a Model of Computation - Easy examples

- indications for semi-Thue systems performing computations
- successor/predecessor functions in unary representation

$$\Sigma = \{1\}, \qquad S = \{(\epsilon, 1), (1, \epsilon)\}$$

- constant functions or identity in any representation

$$\Sigma = \{\sigma_1, ..., \sigma_n\}, \qquad S = \{(\sigma_1, \sigma_1), ..., (\sigma_n, \sigma_n)\}$$

- recursion is possible through derivations
- Is it possible to show equivalence to other models of computation?

Equivalence to Turing Machines

- semi-Thue systems can be shown to be isomorphic to unrestricted grammars
- unrestricted grammars: have additional terminal letters which may not be removed during derivation and always start with a symbol S
- Theorem: the languages of unrestricted grammars are the recursively enumerable languages
- sketch of proof for an unrestricted grammar G to construct a Turing machine M
  1. construct a two tape non-deterministic Turing machine M
  2. tape 1 holds the input
  3. tape 2 holds the state of the current derivation, starting with the starting symbol S
  4. at each step, M chooses nondeterministically a rule from G, applies it to tape 2
  5. if the tapes are same, the input is accepted. if not, the machine chooses another rule to apply
- idea of proof for a Turing machine M to construct an unrestricted grammar G
  1. choose Turing machine to halt in empty state
  2. construct G, s.t. it analyses the behavior of M backwards

Table of Contents

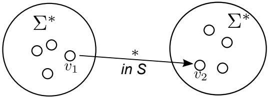
Word Problems

Given a semi-Thue system $T = (\Sigma, S)$

The accessibility problem

instance two arbitrary words $v_1, v_2 \in \Sigma^*$

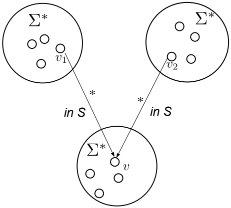
question $v_1 \xrightarrow{*}_S v_2$?



The common descendant problem

instance two arbitrary words $v_1, v_2 \in \Sigma^*$

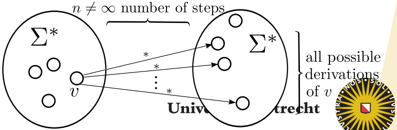
question Is there a $v \in \Sigma^*$, s.t.
$v_1 \xrightarrow{*}_S w$ and
$v_2 \xrightarrow{*}_S w$?



The termination problem

instance an arbitrary word $v \in \Sigma^*$

question Is every derivation starting from v possible to do in a finite number of steps?



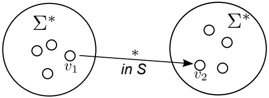
Word Problems

Given a semi-Thue system $T = (\Sigma, S)$

The accessibility problem

instance two arbitrary words $v_1, v_2 \in \Sigma^*$

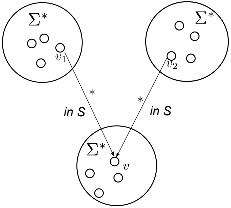
question $v_1 \xrightarrow{*}_S v_2$?



The common descendant problem - UNDECIDABLE

instance two arbitrary words $v_1, v_2 \in \Sigma^*$

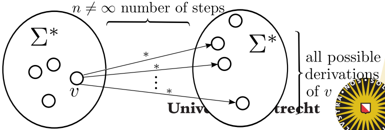
question Is there a $v \in \Sigma^*$, s.t.
$v_1 \xrightarrow{*}_S w$ and
$v_2 \xrightarrow{*}_S w$?



The termination problem

instance an arbitrary word $v \in \Sigma^*$

question Is every derivation starting from v possible to do in a finite number of steps?



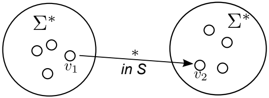
Word Problems

Given a semi-Thue system $T = (\Sigma, S)$

The accessibility problem

instance two arbitrary words $v_1, v_2 \in \Sigma^*$

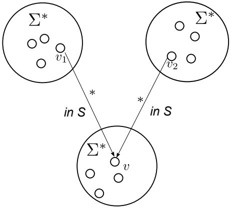
question $v_1 \xrightarrow{*}_S v_2$?



The common descendant problem - UNDECIDABLE

instance two arbitrary words $v_1, v_2 \in \Sigma^*$

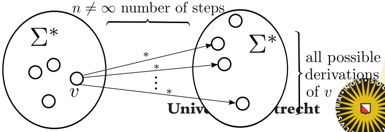
question Is there a $v \in \Sigma^*$, s.t.
$v_1 \xrightarrow{*}_S w$ and
$v_2 \xrightarrow{*}_S w$?



The termination problem - UNDECIDABLE

instance an arbitrary word $v \in \Sigma^*$

question Is every derivation starting from v possible to do in a finite number of steps?



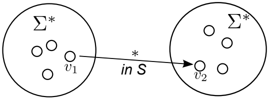
The Accessibility Problem - Proof Sketch

Given a semi-Thue system $T = (\Sigma, S)$

The accessibility problem

instance two arbitrary words $v_1, v_2 \in \Sigma^*$

question $v_1 \xrightarrow{*}_S v_2$?



Theorem

The accessibility problem is unsolvable over an arbitrary finite alphabet.

Proof

following Z. Manna, first proof by E. Post (1947)

1. choose an alphabet Σ, binary relations S, an arbitrary word $x \in \Sigma^*$ and the word $y = \epsilon$
2. construct a Turing or Post machine which halts if and only if $x \xrightarrow{*}_S y$
3. this means that the problem is reduced to the *halting problem* of Post/Turing machines
4. *halting problem* is unsolvable $\Rightarrow$ accessibility problem is unsolvable

The Accessibility Problem is undecidable - Proof

- take $\Sigma = \{B_0, ..., B_m, \vdash, \dashv, a, b\}$
- out of these, construct a Post machine with m test/assignment states $B_1, ..., B_m$ the starting state is B_0
- the Post machine is a machine of alphabet $\Sigma' = \{a, b\}$
- the input word for the machine is supposed to be $w = \sigma_1\sigma_2...\sigma_n$
- for the semi-Thue system take the words

$$x = B_0 \vdash \sigma_1\sigma_2...\sigma_n \dashv$$
$$y = \epsilon$$

- x is supposed to be the initial configuration of the machine, where the B's can be interpreted as the "head"
- the current word w' of the machine is always in between $\vdash$ and $\dashv$
- next: constructing S connected to the behavior of the Post machine

The Accessibility Problem is undecidable - Proof

S is made of the following rules

1. Start $\rightarrow (B_0, B_1)$
2. read first letter and statement transition

$$(B_i \vdash a, B_j \vdash)$$
$$(B_i \vdash b, B_k \vdash)$$
$$(B_i \vdash \dashv, \epsilon)$$

$B_6 \vdash abaab \dashv$

read and go to next statement

$B_9 \vdash baab \dashv$

or

$B_5 \vdash \dashv$

read and halt / grammar yields empty word

$\mathrm{HALT} \ / \ \epsilon$

important! only possibility to **halt**: head reads the empty word between $\vdash$ and $\dashv$

3. add a letter in the end and state transition

$$(\dashv B_i, \sigma' \vdash B_j)$$
$$\sigma' \in \{a, b\}$$

$\vdash babb \dashv B_3$

add and go to next statement

$\vdash babba \dashv B_{538}$

4. Cycling (thus enabling the head to move between the beginning and the end of the current word w')

$$(\sigma B_i, B_i \sigma)$$
$$(B_i \sigma, \sigma B_i)$$
$$\sigma \in \{\vdash, \dashv, a, b\}$$

$B_3 \vdash aabba \dashv$

cycle

$\vdash aabba \dashv B_3$

iversiteit Utrecht

The Accessibility Problem is undecidable - Proof Summed Up

- we have $\Sigma = \{B_0, ..., B_m, \vdash, \dashv, a, b\}$, parallel a Post machine with $\Sigma' = \{a, b\}$
- we constructed S and connected transitions for the machine
- we choose an arbitrary word $w \in \Sigma'^*$ and construct a word for the semi-Thue system

$$x = B_0 \vdash w \dashv$$

$$y = \epsilon$$

- every move of the rules in S is connected to a certain move of the machine
- the only halt statement is given for the head to read ϵ, yielding y
- thus, the machine halts if and only if y is derivable from x

But it is undecidable to determine whether the machine halts or not.

Hence, the accessibility problem is undecidable.

Q.E.D.

Table of Contents

Universiteit Utrecht

Summary

We

- introduced semi-Thue systems as string rewriting systems
- indicated: semi-Thue systems are equivalent to model of Turing machines $\Leftrightarrow$ are a model of computation
- introduced three word problems — indicated that two are undecidable — proved that the third is undecidable, too
- will probably hear more in: Lindenmayer systems, Combinatory Logic, Markov Algorithms, Correspondence Problem