Università degli Studi Roma Tre
Facoltà di Scienze M.F.N.
Corso di Laurea in Matematica

Tesi di Laurea in Matematica

Undecidability of the word problem for groups: the point of view of rewriting theory

Candidato
Matteo Acclavio

Relatori
Prof. Y. Lafont

....

Prof. L. Tortora de falco

....

UNIVERSITÉ
FRANCO
ITALIENNE

questa tesi é stata redatta nell'ambito del Curriculum Binazionale di Laurea Magistrale in Logica, con il sostegno dell'Universitá Italo-Francese (programma Vinci 2009)

Anno Accademico 2011-2012
Ottobre 2012

Classificazione AMS:
Parole chiave:

"There once was a king, Sitting on the sofa,
He said to his maid, Tell me a story, And the maid began:

There once was a king, Sitting on the sofa,
He said to his maid, Tell me a story, And the maid began:

There once was a king, Sitting on the sofa,
He said to his maid, Tell me a story, And the maid began:

There once was a king, Sitting on the sofa,

$\ddots$

"

Italian nursery rhyme

Even if you don't know this tale, it's easy to understand that this could continue indefinitely, but it doesn't have to. If now we want to know if the narration will finish, this question is what is called an undecidable problem: we'll need to listen the tale until it will finish, but even if it will not, one can never say it won't stop since it could finish later... those things make some people loose sleep, but usually children, bored, fall asleep.

More precisely a decision problem is given by a question regarding some data that admit a negative or positive answer, for example: "is the integer number n odd?" or " does the story of the king on the sofa admit an happy ending?". The concept of *algorithm*, mathematically "well-defined", was introduced at the beginning of 20^{th} century by Church ([9],1936) and Turing ([21],1937) who introduced two models of computation: *λ-calculus* and *Turing machines* respectively. This two model of computation are capable, with a succession of simple and mechanizable instructions, to compute functions: it's the birth of theoretical computer science since by the *Turing-Church thesis*, both these two models are equivalent and capable to represent any computable function (this property is called *Turing completeness*). A problem is *decidable* if there exists a computable function giving an exact answer for any instance of a problem, *undecidable* when no computable function that can give an answer to every instance there exists.

The answer to the word problem refers to rewriting systems. A rewriting system is the data of a finite set of symbols called *alphabet* used to write finite sequences of letters called *words* and a finite set of rules to rewrite some words into others. Given a rewriting system and two words of its alphabet, the word problem asks: "can i write one of the two words starting from the other using only the given rewriting rules ?". Rewriting systems are used in algebra and geometry in order to give partial descriptions of objects without giving explicit exhaustive description. In fact we are capable to rebuild complete descriptions of objects that could be infinite or with very complex interaction between its element by rewriting systems that report complex interactions to more simpler ones regarding only some particular elements (called generators). On the other hand, rewriting systems are related with logic and theoretical computer science since they are a Turing complete model of computation.

Rewriting systems are naturally linked with some algebraic structures called monoids: the orientation of the rewriting rules impede to reverse the equality in order to obtain the initial element. This impossibility to reverse some interactions between elements is typical of the monoid structures which, in general, have not *inverse elements.* The first results on the word problem for monoid are from 1947 by Markov [18] and Post [20], showing the possibility to encode the computation of a Turing machine by a rewriting system. This encoding allow to prove the undecidability of the word problem for monoids. Unfortunately, the same argumentation cannot be used in the case of the word problem for groups: the existence of inverses for every element, characteristic of groups, can produce "interferences" during the rewriting process since every element followed by its inverse, even if equal to the empty word, can interact with adjacent letters by some rule. The first results are from 1954 (Novikov [19]) and 1955 (Boone[7]), that prove independently the existence of an explicit procedure to build groups for which they prove that the word problem is undecidable.

The central part of this thesis is a comparison of two proofs of the Novikov-Boone theorem, one by Bokut[5] and the other by Lafont[15], suggested me by Prof. Lafont in order to check the differences between the two proofs, both based on the use of rewriting systems.

The thesis is organized as follows:

- Chapter 1 contains some background: some basic monoid and group theory, rewriting theory and presentation of monoids and groups, a proof of the existence of an embedding $F_\infty \hookrightarrow F_2$ of the free group with a denumerable number of generators into the free group with two generators (Theor. 2), some notions of computability theory defining the Turing machines, some of their properties and some other models of computation which will be used in the sequel;

- In chapter 2 we present some classical undecidability results: the Gödel's incompleteness theorems, Church's theorem on the undecidability of predicate calculus, the undecidability of the halting problem for Turing machines and some similar results for other models of computation;

- The whole chapter 3 is dedicated to a detailed proof of the Higman-Neuman-Neumann extension theorem [14]. We present here the proof given in [15] using rewriting systems as a tool to extend groups in order to have a suitable combinatorial property;

- Chapter 4 is devoted to present a family of groups introduced by Novikov in term of *groups with stable letters* [3], called Novikov-Boone groups. This groups have a particular combinatorial property useful to prove some results of undecidability for groups. In the second part we prove some properties for such type of groups like the fact that they are a particular case of HNN-extension;

- In chapter 5 the proofs of Bokut' and Lafont are analyzed step by step;

- In chapter 6 we present an application of the undecidability of the word problem to prove the undecidability of non-commutative linear logic after a summary of the differences between the two proofs:

  - Bokut's proof [5] [6] is based on a rewriting system induced by the relations of the defining presentation of the *Boone group* $G(T, q)$. This new infinite rewriting system is built to be convergent. So, in order to test if a word W is equal to the letter q, it will suffice to compute the normal form of the word W and compare it with q (since q is in normal form). The undecidability of the word problem for $G(T, q)$ will follow from the undecidability of the word problem for a *special monoid* T encoding of a Turing machine;
  - Lafont's proof [15] is inspired by Aandreaa and Cohen's [1]. It also uses rewriting, but the only essential point is the notion of *convergent rewriting system* and not the study of a particular system. It uses the undecidability of the halting problem for a particular class of abstract machines called *affine machines.* Using some properties of the free group F_∞, it is possible to associate a *local isomorphism* of an extension of F_2 with every transition of an affine machine $\mathcal{A}$. By the *HNN embedding theorem*, the configurations of the machine live in some group $G_\mathcal{A}$, where transitions are represented by elements of $G_\mathcal{A}$. In that group, the word problem is equivalent to accessibility for the affine machine of a fixed configuration, a problem which is proven to be undecidable ([15]).

- Appendix A gives an overview of combinatorial systems showing some properties for string rewriting systems; we then prove the undecidability of the word problem for monoids and we explain why the results for groups can't be given in the same way;

- Appendix B gives an overview of linear logic sequent calculus and presents some classical results.

Chapter 1

Some backgrounds

1.1 Basic Group and Monoid Theory

A *group* is an algebraic structure consisting of a set together with a *binary operation* that with two given elements of the set, associates a third one. The set with an operation, to be a group, have to satisfy a four of axioms: closure, associativity, identity and invertibility. A group is surely one of the simplest algebraic structures and it was the first one studied with this modern point of view. The concept of group arose from Évariste Galois' studies (1830's) on polynomial equations: he linked their solubility to some particular property of a group associated to each polynomial. The study of the group was also developed in other field of math: Felix Klein in 1872 in his *Erlagen program* classify the new geometries discovered in the 19th century (non-euclidean and projective) considering them groups of symmetries. Moreover, in number theory, in order to solve the last Fermat's problem, this new notion was used to generalize results to class of object with similar numerical property.

Even though the notion of *monoid* is differs from group's one for the absence of the invertibility axiom, it started to be studied later at the beginning of 20th century. Eliminating the notion of *inverse elements* is obtained a structure which can better represents the concept of function composition and computing process: even if we know a result and what kind of transformation we have done to obtain it, we can't always find out the initial data since such transformation could be not reversible. For this reason monoids are used every time there is an irreversible process and so it found large application in theoretical computer science, category theory but also probability.

1.2 Monoid presentations

Monoid's and groups' presentation are strictly related with algebra, geometry and model of computation. Them was introduced in the end of 19th century by Walter von Dyck to study groups in term of generators and relation. Study groups in this way permits to analyze some property without know exactly the group: algebraic geometry use them to compute *fundamental groups* of topological spaces groups' amalgams arising from the Seifert-Van Kampen theorem. The link with the computer since is due to the fact that *semi-Thue system* are a

Turing-complete model of computation (see Chapt. 1.3 and Appendix A) with interesting behaviors about convergence linked with the monoid's homology and finiteness propert.

Definition 1 (Monoid Presentation) *Let Σ be an* alphabet*, Σ^* the set of* word *on Σ (i.e. the set of all possible finite concatenation of symbols of the alphabet including the* empty word 1*) and $\mathcal{R} \subset \Sigma^* \times \Sigma^*$ a set of* rules *on Σ. A presentation P is a couple $(\Sigma|\mathcal{R})$ given by an alphabet and a set of rule on the alphabet. A presentation is called* finite *if Σ and $\mathcal{R}$ are finite sets.*

Remark 1 *A group is* finitely presented *if it can be is finitely presented as monoid.*

Definition 2 (Translation) *Let $(\Sigma|\mathcal{R})$ and $(\Sigma'|\mathcal{R}')$ two presentation of monoids. A* translation *it's a function $\bar{\phi} : (\Sigma|\mathcal{R}) \to (\Sigma'|\mathcal{R}')$ obtained extending a map $\phi : \Sigma \to \Sigma'^*$ on presentation such that:*

1. $\forall w \in \Sigma,\ \bar{\phi}(w) = \phi(w)$;
2. $\forall \mathfrak{r} = (u, v) \in \mathcal{R},\ \bar{\phi}(\mathfrak{r}) = (\phi(u), \phi(v)) \in \leftrightarrow^*_{\mathcal{R}'}$.

Lemma 1 (Lafont embedding lemma) *Let $(\Sigma|\mathcal{R})$ and $(\Sigma'|\mathcal{R}')$ be two presentations such that:*

- $\Sigma \subseteq \Sigma'$;
- $(\Sigma'|\mathcal{R}')$ *is convergent;*
- $\mathcal{R} = \{(u, v) \in \mathcal{R}' | u \in \Sigma^*\}$.

Then the inclusion $\phi : \Sigma \hookrightarrow \Sigma'$ defines a translation $\bar{\phi} : (\Sigma|\mathcal{R}) \to (\Sigma'|\mathcal{R}')$ and $\hat{\phi}$ is injective.

Theorem 2 *It exists an embedding of F_ω into F_2.*

Lemma 3 $\forall p, q \in \mathbb{Z}, q \neq 0$ *the family $\{a_p, b^q\}$ is free in F_2*

Definition 3 (Word problem) *Given a rewriting system $(\Sigma|\mathcal{R})$, the word problem consists to answer the following question:*

$$\text{Given } v, w \in \Sigma^* \text{ are } v \leftrightarrow^*_{\mathcal{R}} w?$$

The word problem can be defined in the same way for monoids and groups which admits a finite presentation. It can also be defined with $\to^*$ at the place of $\leftrightarrow^*_{\mathcal{R}}$, the two formulation are equivalent. Some texts highlight the difference obetween words probelem (as defined above) and word problem (where $w = 1$). Both problems have the same answer.

1.3 Basic Computability Theory

In the world meeting of 1900 Hilbert proposed some mathematical problem to work on in 20th century. One of them, the 2nd one, asked if the axiom of arithmetics are *consistent* (i.e. not contradictory). To answer this question, the study of logic was deepened bringing out new problem like the entscheidungsproblem (decision problem) posed by Hilbert in 1928 which asks the existence of an algorithm capable to test if a first-order logic statement is universallly valid. At that time there was not a proper definition of algorithm which was formally defined by Alonzo Church in 1936 and independently by Alan Turing respectively in term of *λ-calculus* and *Turing machines*. By the *Church-Turing thesis* these two models of computation are equivalent and they can express all and every computable function (for every computable function exists a λ-term and a Turing machine that can calculate it).

Here we'll present also some other models of computation *Turing complete* (i.e. they can compute the same class of functions of a Turing machine) that we'll use during the proofs.

For reasons related to the proofs we'll study, we'll focus on the Turing model instead of the λ-calcul (which has a more manageable definition). Moreover, the definition of a computability with Turing machine permits to define the concept of complexity of a computation in term of number of transitions that the machine have to do to compute it.

1.3.1 Turing Machines

Proposition 1 (Existence of Universal Turing Machine [21]) *It is possible to invent a single machine which can be used to compute any computable sequence. If this machine U is supplied with a tape on the beginning of which is written the "standard description" of an action table of some computing machine M, then U will compute the same sequence as M.*

Definition 4 (Decidability) *A function is* decidable *if there is a Turing machine which can compute it for any initial data. A property P is decidable if the characteristic function the set representing P.*

1.3.2 Some other model of computation

We'll now present some model of computation we'll need in the prove we'll study.

Theorem 4 *Every n-register machine R can be simulated by a 2-register machine $R_2(R)$.*

Definition 5 (Affine machine [15]) *An* affine machine*, fixed an $m \in \mathbb{N}$, is a finite set $\mathcal{A} \subset \mathbb{Z} \times \mathbb{Z}^* \times \mathbb{Z} \times \mathbb{Z}^*$. Every $(p, q, p', q') \in \mathcal{A}$ define an* affine transition *$p + qz \to_{\mathcal{A}} p' + q'z$ $(z \in \mathbb{Z})$.*

Remark 2 *Every 2-register machines $\mathcal{M}$ can be simulated by an affine machine.*

Chapter 2

Some Undecidability Results

The negative answer to the decision problem corresponds to the existence of *uncomputable* function. The first step was to answer the entscheidungsproblem is due to Gödel: in his incompleteness theorems he proved that in sufficient expressive theory $\mathcal{T}$ which can axiomatize the arithmetic, if it is coherent, is always possible to found a formula that ϕ such that booth ϕ and $\neg\phi$ can't be proved in $\mathcal{T}$. To prove this, he utilize the notion of *primitive recursive function* to encode the syntax of logic by numbers. The same encoding was used by Church to answer to generalize the results and answer to the problem.

Also Turing proved that a *Turing machine* which decide if a given Turing machine will stop its computation starting by a given initial data, can't exists: this problem, the *halting problem*, is undecidable too. Using that results he proved that the entscheidungsproblem is undecidable.

We'll use this result to prove the undecidability of some other problems showing a way to *reduce* their instance to an instance for the halting problem and so showing this problem can't be decidable.

2.1 Gödel's Theorems

In his work [13], Gödel give a method to associate to every formula its *Gödel number* (a natural number) which encode it. Moreover he gives a way to express the logical derivation in the language $\mathcal{L}_0 = \{\simeq, \underline{0}, \underline{+}, \underline{\times}, s\}$ of arithmetic. In that system he shows the existence of a *fixed point* for any formula and, using it, the existence of formulas which can't be proved in theories.

Theorem 5 (First Gödel incompleteness theorem) *In any coherent theory containing PA there is a statement G such that booth G and $\neg G$ cannot be proved in the theory.*

Theorem 6 (Second Gödel incompleteness theorem) *Any coherent theory T containing PA cannot prove its own consistency.*

2.2 Church's Theorems

The first Gödel theorem can be simply extended to PA_0 by the following:

Theorem 7 *Let T be a theory extending PA_0, if T is consistent so T is undecidable*[1]*.*

In [9] Church utilize the Gödel's results to proper answer to the decision problem generalized for the propositional logic.

Theorem 8 (Church) *First order logic expressed in the lenguage $\mathcal{L}_0$ is undecidable.*

2.3 Undecidability of the halting problem for Turing machines

The halting problem for Turing machines asks if, given a Turing machine M and a configuration s_0, the computation of M starting from s_0 terminates.

Theorem 9 (Undecidability of the halting problem) *The halting problem for Turing machine is undecidable.*

2.4 Some other results

Theorem 10 (Undecidability of the halting problem for 2-register machine) *There exist a 2-register machine with undecidable halting problem*

Theorem 11 (Undecidability of $\mathcal{H}alt$ problem for modular machines) *There exist an affine machine $\mathcal{A}$ such that $\mathcal{H}alt_{\mathcal{A}}$ is undecidable.*

The equivalence problem for an affine machine $\mathcal{A}$ asks if, given a $z \in \mathbb{Z}$, $z \leftrightarrow^*_{\mathcal{A}} m$.

Theorem 12 (Undecidability of equivalence problem for affine machines) *There exists a machine affine $\mathcal{A}$ and an integer m such that the equivalence problem it's undecidable.*

2.5 Undecidability of Propositional Linear Logic

Theorem 13 *If there is a proof of $\vdash \Gamma$ in theory T, then there is a directed proof of $\vdash \Gamma$ in theory T.*

Lemma 14 *It is undecidable if an ACM accepts from a given ID.*

Lemma 15 *An ACM accepts form an ID s iff every sequent in $\Theta(s)$ is provable in the theory derived from M.*

[1]A theory is undecidable if the set of provable closed formula is undecidable

Chapter 3

The Higman-Neuman-Neuman Extension Theorem

In order to build groups' extensions with particular combinatorial propriety, it will be useful to use the HNN-theorem *for the groups.*

3.1 HNN extension theorem

Theorem 16 (HNN extension associated with a subgroup) *Let G be a group, $\forall H < G, \exists F > G$ and $b \in F$ such that $H = C_G(b)$.*

3.1.1 HNN extension theorem proof
Part I: A non convergent presentation of F

In order to demonstrate the theorem, we'll build an "ad hoc" extension F of G and we'll show that exist an element $b \in F$ such that $H = C_G(b)$.
Let $F = \frac{G*\langle b\rangle}{\leftrightarrow^*_C}$ where $\leftrightarrow^*_C$ is the smallest equivalence relation containing the set $C = \{(bh, hb)|h \in H\}$. The free product $G * \langle b\rangle$ can be presented, given the standard presentation of G and the minimal presentation of $\mathbb{Z}$ as monoid[1]*, by $(\Sigma_G \cup \{b, \bar{b}\}|\mathcal{R}_G \cup \{(b\bar{b}, 1), (\bar{b}b, 1)\})$, so we have a presentation of F*

$$(\Sigma_F = \Sigma_G \cup \{b, \bar{b}\}|\mathcal{R}_F = \mathcal{R}_G \cup R_b \cup \mathcal{R}_H\})$$

where $\mathcal{R}_H = \{(\beta a_h, a_h\beta)|h \in H, \beta \in \{b, \bar{b}\}\}$.

Remark 3 *The presentation $\langle\Sigma_F|\mathcal{R}_F\rangle$ is not convergent.*

3.1.2 HNN extension theorem proof
Part II: A convergent presentation of F

Using a properties of groups is possible to give another presentation of F adding new superfluous generators and new relation. Let fix an $H^\perp$ with $1 \in H^\perp$, we

[1]see **??** pag. **??**

define the superfluous generators $b_v = ba_v$ and $b'_v = \bar{b}a_v$ ($\Sigma_\perp := \{b_v, b'_v | v \in H^\perp\}$).[2] *Using the relation of $\mathcal{R}_F$ and the fact that, by the Prop.??, is possible to derivate the following set $\mathcal{R}_\perp$ of relations:*

$$\forall v \in H^\perp \qquad b_1 b'_v \to a_v \qquad b'_1 b_v \to a_v$$

$$b_v a_x \to a_h b_w \ \exists! h \in H, w \in H^\perp \text{ such that } vx = hw$$

$$b'_v a_x \to a_h b'_w \ \exists! h \in H, w \in H^\perp \text{ such that } vx = hw$$

Proposition 2 *The presentation $(\Sigma_G \cup \Sigma_\perp | \mathcal{R}_G \cup \mathcal{R}_\perp)$ is convergent.*

3.1.3 HNN extension theorem proof Part III: Concluding

It's easy to prove by that $F' \geqslant G$ and $F' \geqslant \langle b \rangle$. It's also evident for construction that $C_G(b) \geqslant H$. To prove the equality it suffices to show that only the elements of H commutes with b. Let $x = hw$ with $h \in H$ and $w \in H^\perp$, we have $b_1 a_x \to_{\mathcal{R}_{F'}} a_h b_w$ and so $ba_x = \psi(b_1 a_x) = \psi(\phi(ba_x)) \to_{\mathcal{R}_F} \psi(a_h b_w)$. But $a_h b_w$ is reduced and so $\psi(a_h b_w) = a_h b a_w$ is. So $\forall x \in G$ $xb = hbw = xb$ iff $x \in H$(i.e. $w = 1$) that mean $C_G(b) = H$.

3.2 HNN extention theorem application

Corollary 17 *If G is finitely presented and H is finitely generated in G, then the HNN-extension F of G associated with H is finitely presented.*

Theorem 18 (HNN extension associated with an local isomorphism) *Let G be a group, $\forall \phi : H \to H'$ local isomrphism, $\exists F > G$ and $b \in F$ such that:*

1. *b represents ϕ ;*
2. *$\langle K, b \rangle_F \cap G = K$ for all K ϕ-invariant ;*
3. *if G is finitly presented and H finitely generated F is finitely presented .*

Theorem 19 (HNN extension associated with several local isomorphism) *Let G be a group, $\forall \phi_1 : H_1 \to H'_1, \ldots, \phi_n : H_n \to H'_n$ local isomorphism, $\exists F > G$ and $b \in F$ such that:*

1. *b_i represents ϕ_i $\forall i$*
2. *$\langle K, b_1, \ldots, b_n \rangle_F \cap G = K$ for all K invariant for all ϕ_i*
3. *if G is finitely presented and all H_i finitely generated F is finitely presented*

[2] $b_v = ba_v$ and $b'_v = \bar{b}a_v$ essentially means that b_v and b'_v are abbreviation respectively for the words ba_v and $\bar{b}a_v$

Chapter 4

Novikow-Boone's groups

Independently of Higman, Neumann and Neumann's work oriented to a purely algebraic and topological application, Novikow in [19] discovered the HNN-extension and approach the subject in a more constructive way. With Boone [7] they connect it to algorithmic and combinatorial algebra demonstrating the undecidability of the word problem for the groups.

4.1 A Novikov-Boone's group zoo

Here will be presented some Novikov-Boone's groups, stating some their properties that permits to demonstrate the undecidability of some of their property.

4.2 Group with standard basis

Introducted by Bokut' in [3] a standard basis *or* standard normal form *permits to have a canonical form to write an element of a Novikov-Boone group given one of its presentation.*

Chapter 5

Undecidibility of the word problem for the groups

5.1 Bokut' proof

In [3] Bokut gives the proofs of Novikov-Boone's theorem proving that Boone's groups $G(T, q)$ has standard basis. It make it easyer (Bokut' [4]) to prove that exists a finitely presented group in which the word problem for the group $G(T, q)$ can have any fixed Turing degree of unsolvability.

5.1.1 The Boone group

Let's now build the Boone group like a succession of HNN extension, for every extension will be given them additional generators and relations, the letters of maximal weight that will appear in the definition of prohibiten words will be highlited and there will be explicitated the twin words form.

Definition 6 (Boone group) *Let T be a* special semigroup[1], *i.e. a semgroup generated by $\{s_d, q_e\}_{d\in D, e\in E}$ and relations $A_i = B_i, 1 \leqslant i \leqslant N$ where A_i, B_i* special words *($A_i, B_i \equiv S q_e S'$ where S, S' are $\{s_d\}$-words).*

- $G_0 = \langle x, y\rangle$;
- G_1: $\{s_d | d \in D\} \quad | \quad \mathbf{y} y s_d = s_d \mathbf{y},\ \mathbf{x} s_d = s_d x \mathbf{x}$,
  $\mathcal{A}_{s_d} = V(x, y^2)$, $\quad \mathfrak{B}_{s_d} = V(x^2, y)$;
- G_2: $\{l_i, r_i | 1 \leqslant i \leqslant N\} \quad | \quad \mathbf{s_d} l_i = y l_i y \mathbf{s_d},\ \mathbf{s_d} x r_i x = r_i \mathbf{s_d}$,
  $\mathcal{A}_{l_i} = V(y^{-1} s_d)$, $\mathfrak{B} = V(y s_d)$, $\mathcal{A}_{r_i} = V(s_d x)$, $\mathfrak{B}_{r_i} = V(s_d x^{-1})$;
- G_3: $\{q_e | e \in E\} \quad | \quad A_i = \mathbf{l_i} B_i \mathbf{r_i}, A_i \equiv A'_i q_{n_i} A''_i, B_i \equiv B'_i q_{m_i} B''_i$,
  where A'_i, A''_i, B'_i, B''_i are $\{s_b\}$-words and
  $\mathcal{A}_{q_{m_i} q_{n_i}} = V(A'^{-1}_i l_i B'_i)$, $\mathfrak{B}_{q_{n_i} p_{m_i}} = V(A''_i r_i^{-1} B''^{-1}_i)$;
- G_4: $\{t\} \quad | \quad \mathbf{l_i} t = t \mathbf{l_i},\ \mathbf{y} t = t \mathbf{y}$,
  $\mathcal{A}_t = V(l_i, y) = \mathfrak{B}_t$;

[1] App.A

- *fixed a $q \in \{q_e\}$, G_5: $\{k\} \mid \mathbf{r_i}k = k\mathbf{r_i},\ \mathbf{x}k = k\mathbf{x}, q^{-1}\mathbf{t}qk = kq^{-1}\mathbf{t}q$,*
  $\mathcal{A}_k = V(r_i, x, q^{-1}tq) = \mathfrak{B}_k$.

Theorem 20 *The Boone group $G(T, q) = G_5$ have a standard basis.*

Lemma 21 *The word problem for the group G_4 is solvable.*

Lemma 22 *Let S, S' special word in T then $S =_T S'$ iff there exist $V(l_i, y), W(r_i, x)$ such that:*

$$S =_{G_3} V(l_i, y)S'W(r_i, x).$$

Lemma 23 *The problem for a word U of the group G_3 to equal to a word in the form $V(l_i, y)SW(r_i, x)$ with S a special word is solvable.*

Theorem 24 *The decidability of the word problem for the group $G(T, q)$ coincides with the decidability of the problem to verify the equality of special word of a monoid T to a word q.*

Corollary 25 (Undecidability of word problem for the groups) *There exists a finitely presented gruop with undecidable word problem.*

Proof: *By Theo.29 exists a finite presented monoid T with defining relation given by special words and undecidable word problem, so by Theo.24 the associated Boone group will have undecidable word problem.*

5.2 Lafont proof

Using the affine machines Lafont in [15] give another proof of the theorem, similar to the proof given in [1] by Cohen Aandrea, in a simply way. We'll use the same notation of Theor.2 : $F_2 = \langle a, b\rangle$ and $a_n = b^n ab^{-n}$.

Lemma 26
For all $p, p', q, q', z \in \mathbb{Z}$, $q, q' \neq 0$, there is an isomorphism $\phi : F_2 \to F_2$ such that $\phi(a_{p+qz}) = a_{p'+q'z}$.

Notation: *Let $I \subset \mathbb{Z}$, $[P]_{F_2}$ is the subset of F_2 generated by the set $\{a_z | z \in \mathbb{Z}\}$.*

Lemma 27 *Let $p, q \in \mathbb{Z}$, so $\langle a_p, b^q\rangle \cap [\mathbb{Z}]_{F_2} = [p + q\mathbb{Z}]_{F_2}$.*

Proof: *[Undecidability of word problem for the groups] Let $m \in \mathbb{Z}$ and $\mathcal{A}$ machine affine. It's possible to associate for every transition of $\mathcal{A}$ a local isomorphism ϕ_i. By the Theor.19 is possible to obtain an extension of $F_\mathcal{A}$ of F_2 with stable letters $t_1 \ldots t_n$ which represents the local isomorphism $\phi_1 \ldots \phi_n$. Let $P = \{z \in \mathbb{Z} | z \leftrightarrow^*_\mathcal{A} m\}$ and $H = \langle a_m, t_1, \ldots t_n\rangle$. By Lemma 26 follow:*

- *if $z \to_\mathcal{A} z'$ so $a_{z'} = \phi_i(a_z) = t_i a_z t_i^{-1}$ exist an $i \in \{1, \ldots, n\}$*
- *if $z \leftrightarrow^*_\mathcal{A} z'$ so $a_{z'} = \phi_{i_n} \circ \ldots \circ \phi_{i_1}(a_z) = u a_z u^{-1}$ exist an $u \in \langle t_1, \ldots t_n\rangle$*

so $K \subseteq H$ because $a_m \in K$ and for every $z \leftrightarrow^*_{\mathcal{A}} m$, $a_m \leftrightarrow^*_{\mathcal{A}} a_z$ and $K = K \cap [\mathbb{Z}]_{F_2} = H \cap [\mathbb{Z}]_{F_2}$. Moreover K it's invariant for every local isomorphism ϕ_i because

$$\langle a_p, b^q \rangle \cap K = \langle a_p, b^q \rangle \cap [\mathbb{Z}]_{F_2} \cap K = [p + q\mathbb{Z}]_{F_2} \cap [P]_{F_2} = [(p + q\mathbb{Z}) \cap P]_{F_2}$$

and so (see Theo. 19) $K = H \cap F_2$.
So is possible to see that exists an extension $F_{\mathcal{A}}$ finitely presented of F_2 and $u \in F$ such that

$$a_z u = u a_m \text{ in } F_{\mathcal{A}} \Leftrightarrow a_z \in H \Leftrightarrow a_z \in K = [P]_{F_2} \Leftrightarrow z \leftrightarrow^*_{\mathcal{A}} m$$

Therefore the word problem for group $F_{\mathcal{A}}$ is reducible to the $\mathcal{H}$alt problem for the machine affine $\mathcal{A}$ which can be undecidable (see Prop.12).

Chapter 6

Conclusion

Now that we have analyzed the two proofs, we can find out the differences of the use of rewriting:

- *Bokut' starts from the definition of Boone group to create a convergent rewriting system on its elements. Using it, he shows that any elements will have a canonical representative and so the word problem will be reduced to the verification if two different elements have the same canonical representative.*

  This process will be equivalent to every implementable test on words to verify the equivalence since this new rewriting system is derived by the presentation and can be viewed as an order on the equivalence classes of words in $G(T, q)$, where the normal forms are the minimums, so it is just a test on two representatives of the equivalence classes. This rewriting system will work progressively on the alphabets used to build $G(T, q)$ computing its normal form. This computation will be decidable iff the word problem of the monoid T is since, during the computation, the rewriting system needs to know if an elements of T is equivalent to q and so, by the construction of T, if an element of the monoid T represents a configuration of a Turing machine contained in a terminating computation;

- *The Lafont's idea is to consider a copy of F_∞ contained in F_2 to simulate the natural numbers and, usind the affine machines $\mathcal{A}$, to consider a subset of $n \in \mathbb{N}$ "equivalent" to a fixed $m \in \mathbb{N}$. With HNN theorem will be possible to extend F_2 with elements that will simulate the affine transition of $\mathcal{A}$. In this new group $L(\mathcal{A})$ there will be some elements corresponding to natural numbers and some others to affine transitions, so there will be some elements capable to represent a computation of $\mathcal{A}$. To test if an element which represent a computation of $\mathcal{A}$ is equal to the element representing the number m will be decidable if and only if the equivalence problem for $\mathcal{A}$ is. Taking an affine machine with undecidable equivalence problem, $L(\mathcal{A})$ will be a group with undecidable word problem.*

Appendix A

Combinatorial system

A.1 Undecidability of word problem for monoids

Proposition 3 *Every non deterministic Turing machine can be simulated by a semi-Thue system.*

Corollary 28 *Semi-thue system are a Turing complete computation model.*

This proves the following:

Theorem 29 (Post-Markov ([20],[18]) *Exists a finite semigroup with undecidable word problem.*
More precisely it exists a monoid finitely presented with rewriting rule expressed by special words.

A.2 Why undecidability of word problem for groups is more difficult

We'll give an example to show why to prove the undecidability of word problem for groups we can't use the same methods used to prove the same results for monoid.
Example: *Let $\mathcal{M}$ be a 2-register machine given by the following instruction set:*

s_1 $JZDEC(a, s_\perp, s_2)$;

s_2 $INC(a, s_3)$;

s_3 $JZDEC(b, s_\perp, s_3)$;

$s_\perp$ $HALT$.

All the transition of $\mathcal{M}$ will be in the form:

$$(1,0,y) \to_{\mathcal{M}} (\perp,0,y)\ , \qquad (1,x{+}1,y) \to_{\mathcal{M}} (2,x,y)\ , \qquad (2,x,y) \to_{\mathcal{M}} (3,x{+}1,y),$$

$$(3,x,0) \to_{\mathcal{M}} (\perp,x,0)\ , \qquad (3,x,y+1) \to_{\mathcal{M}} (3,x,y).$$

If we encode the configuration (i, x, y) of $\mathcal{M}$ as $[i, x, y] = \alpha a^x s_i b^y \omega$ we can build a monoid

$$M_{\mathcal{M}} = \langle \Sigma_{\mathcal{M}} = \{\alpha, \omega, a, b, s_1, s_2, s_3, s_\perp\} | \mathcal{R} \rangle^+$$

where $\mathcal{R}$ is the set of rules applied on word in the form $\alpha a^x s_i b^y \omega$ represent the transition of $\mathcal{M}$:

$$\alpha s_1 \rightarrow \alpha s_\perp, \qquad a s_1 \rightarrow s_2, \qquad s_2 \rightarrow a s_3, \qquad s_3 \omega \rightarrow s_\perp, \qquad s_3 b \rightarrow s_3.$$

In this monoid $s_1 \neq_M s_3$ but in the group $G = \langle \Sigma_{\mathcal{M}} | \mathcal{R} \rangle$ we have $a^{-1} a s_1 \rightarrow s_1$ and $a^{-1} a s_1 \rightarrow a^{-1} s_2 \rightarrow a^{-1} a c_3 \rightarrow s_3$ so $s_1 =_G s_3$.
This is not coherent with the register machine computing since:

$$(1, 0, 1) \rightarrow^*_{\mathcal{M}} (\perp, 1, 0) \qquad \textit{and} \qquad (1, 0, 1) \not\leftrightarrow^*_{\mathcal{M}} (\perp, 0, 0)$$

while in G we have $[1, 0, 1] = \alpha s_1 b \omega =_G \alpha s_3 b \omega =_G \alpha s_3 \omega =_G \alpha s_\perp \omega = [\perp, 0, 0]$.

The presence of the inverse element for every element of G create interferences which doesn't respect the derivation of a model of computation, because, in general, transitions don't admit an inverse transition capable to restore the changes done.

Theorem 30 (Word problem for commutative rewriting system)
Let $(\Sigma | \mathcal{R})$ a presentation of a commutative monoid, so the word problem for $\langle \Sigma | \mathcal{R} \rangle^+$ is decidable. (i.e.)
Proof: *since $\{(a_i a_j, a_j a_i), (a_j a_i, a_i a_j)\}_{a_i, a_j \in \Sigma} \subseteq \leftrightarrow^*_{\mathcal{R}}$ letters commute, so it only matters the number of occurrence of any letters to know if it is possible to apply a rule. It will be possible to compute, form any word v, the set S_v of words in which it can be rewritten and verify if for some of word in S_v and S_w contain the same number of occurrence of the same letters.*

Appendix B

Linear Logic

A linear logic sequent is a $\vdash$ followed by a multiset of linear logic formulas. We assume a set of proposition p_i given, along with their associated negation $p_i^{\perp}$. Below we give the inference rules for the linear sequent calculus, along with the definition of negation and implication. The negation is a defined concept, not an operator.

B.1 Cut Elimination

Like in classical sequent calculus, also in linear logic we have an analogous cut elimination theorem:

Theorem 31 *If A is provable in linear logic , there exists a prove of A without Cut rules.*

The theorem is still valid also in NCL fragment:

Theorem 32 *If A is provable in NCL , there exists a prove of A without Cut rules.*

As seen in Ch.2.5, we may need work in a fragment of linear logic with some non-logical axioms *i.e. rules in the form:*

$$\frac{}{\vdash C, p_{a_i}^{\perp}, \ldots, p_{a_j}^{\perp}}\, T$$

where C is a formula in $MALL$ and $p_{a_j}^{\perp}$ negative literals. If we define a direct cut *an application of a cut rule where one of the premises is consequence of a non-logical axiom and a* direct proof *is a proof without non-direct cut, the cut elimination theorem is still valid as enunciated in Teor.13.*

Bibliography

[1] *S. Aandreaa & E. Cohen,* Modular machines, the word problem for finitely presented groups, Collins' theorem*, Word Problems II. (1980) p.1-16*

[2] *V.M. Abrusci, L. Tortora de Falco,* Appunti del corso di logica*, (2009)*

[3] *L.A. Bokut',* Groups with a relative standard basis*, Sibirskii matematichan Z. 9, N.3 (1968) p.4-52, 1980, p.29-53*

[4] *L.A.Bokut',* The degrees of unsolvability for the conjugacy problem for finitely presented groups*, Algebra and Logic7 , N.5 (1968) p.4-70, N.6 (1968) p.4-52*

[5] *L.A. Bokut',* Malcev's problem and groups with a normal form*, Word Problems II (1980) p.29-53*

[6] *L.A. Bokut',* Algorithmic and Combinatorial Algebra*, Kluwer Academic Publisher (1994), chap.6-7*

[7] *W.W. Boone* The word problem*, Annals of mathematics (1959), vol.70, N.2, p.207-265*

[8] *G.S.Ceitin,* Associative calculus with undecidable equivalence problem*, Dokl. Akad. Nauk SSSR (1956), vol.107, N. 3 , p.370-371(in russian)*

[9] *A.Church,* A note on the entscheidungproblem*, The journal of Symbolic Logic Vol.1, Number1, March 1936*

[10] *R.Cori and D.Lascar* Logique mathèmatique*, Masson, Paris, 1993*

[11] *M.D.Davis,* Computability & unsolvability*, McGraw-Hill Book Company(1958), ch.6*

[12] *M.D.Davis and E.J.Weyuker,* Computability, complexity and lenguages*, Accademic Press (1983), p.118-146*

[13] *K.Gödel,* Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I*, Monatshefte für Mathematik und Physik, 38 (1931), pp.173-198.*
Translated by Martin Hirzel, 2000: On Formally undecidable proposition of *Principia Mathematica* and related systems

[14] *G.Higman, B.H. Neumann, H. Neumann* Journal of the London Mathematical Society *s1-24 (4): 247254*

[15] Y.Lafont, Réécriture et problem du mot*, Gazette des mathématiciens 120 (2009) p.27-38*

[16] Patrick Lincoln, Kohn Mitchell, Andre Scedrov, Natarajan Shankar, Decision Problems for Propositional Linear Logic*, Foundations of Computer Science, 1990. Proceedings., 31st Annual Symposium on*

[17] M.L.Minsky, Recursive unsolvability of Post problem of "Tag" and other topics in theory of Turing machines *, Annals of math. (1961), Vol.74,N.3, p.437-455*

[18] A.A.Markov, On the impossibility of certain algorithm in the theory of associative systems*, Dokl. Akad. Nauk SSSR (1947), vol.55, p.587-590(in russian). English translation,* Compte rendus de l'academie des sciences de l'U.R.S.S.*, n.s, vol. 55, p. 583-586*

[19] P.S. Novikov On algorithmic undecidability of the word problem*, Dokl. Akad. Nauk SSSR (1952), vol.85, N.4, p.485-524 (in russian)*

[20] E.L. Post, Recursive unsolvability of a problem of Thue*, The journal of sybolic logic(1947), vol.12, p.1-11*

[21] A.Turing, On computable numbers, with an application to the entscheidungsproblem*, Proceedings of the London Mathematical Society. 2 42 : p230-265. 1937*

[22] M.K.Veliev', On a problem of G. Higman*, Algebra i logika, 1968, Vol7, N.3, O.9-22 (in russian)*