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,
> > >
> > > ⋱

"

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. 3), 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.

Contents

Introduction **i**

1 Some backgrounds **1**

1.1 Basic Group and Monoid Theory 1

1.1.1 Group Theory 1

1.1.2 Monoid theory 3

1.2 Monoid presentations 4

1.3 Basic Computability Theory 10

1.3.1 Turing Machines 10

1.3.2 Some other model of computation 12

2 Some Undecidability Results **16**

2.1 Gödel's Theorems 16

2.2 Church's Theorems 20

2.3 Undecidability of the halting problem for Turing machines 20

2.4 Some other results 21

2.5 Undecidability of Propositional Linear Logic 22

3 The Higman-Neuman-Neuman Extension Theorem **27**

3.1 HNN extension theorem 27

3.1.1 HNN extension theorem proof Part I: A non convergent presentation of F 27

3.1.2 HNN extension theorem proof Part II: A convergent presentation of F 28

3.1.3 HNN extension theorem proof Part III: Concluding 29

3.2 HNN extention theorem application 30

4 Novikow-Boone's groups **32**

4.1 A Novikov-Boone's group zoo 32

4.1.1 Novikow group $\mathfrak{A}_{p_1,p_2}$ 32

4.1.2 Novikow group $\mathfrak{A}_p$ 33

4.1.3 Boone group 33

4.1.4 Borisov group 34

4.1.5 Aandrea group 34

4.1.6 Valiev group 35

4.2 Group with standard basis 36

4.2.1 The definition of groups with standard normal form . . . 38

5 Undecidibility of the word problem for the groups 40

5.1 Bokut' proof 40

5.1.1 The Boone group 40

5.2 Lafont proof 45

6 Conclusion 47

6.1 Word problem and Propositional linear logic 48

A Combinatorial system 50

A.1 Undecidability of word problem for monoids 50

A.2 Why undecidability of word problem for groups is more difficult . 51

B Linear Logic 53

B.1 Cut Elimination 55

Bibliography 57

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.

1.1.1 Group Theory

Definition 1 (Group) *A* group $G = (S, *)$ *is given by a set* S *(support of* G*) and a* binary operation $*$ *on* S*:*

$$*_G : S \times S \to S$$

satisfying the following axioms (group axiom):

- *(Closure):* $\forall g, g' \in G,\ g * g' \in G$*;*

- *(Associativity):* $\forall g, g', g'' \in G,\ g * (g' * g'') = (g * g') * g''$*;*
- *(Identity):* $\exists e \in G$ *such that* $\forall g \in G, e * g = g = g * e$*;*
- *(Invertibility):* $\forall g \in G\ \exists g^{-1} \in G$ *such that* $g * g^{-1} = e = g^{-1} * g$

where is used to write $g \in G$ *to note* $G = (S, *)$ *and* $g \in S$*.*

Notation: Since every operation on a set induce an unique group, in case of ambiguity in presence of different group operations, we'll write $*_G$ to note the operation of the group G.

Definition 2 (Order of a group's element) *Let* $G = (S, *)$ *be a group, the* order of G *or* cardinality of G *is the number of elements of* S*. If* $g \in G$ *the* order of $g := ord(g)$ *is the minimum* $n \in N$ *such that* $g^n = \underbrace{g * \ldots * g}_{n} = 1$*. If there is not* $n \in \mathbb{N}$ *such that* $g^n = 1$ *that will be noted* $ord(g) = \infty$*.*

Definition 3 (Subgroup) *If* $H \subseteq G$ *is a subset of elements of* G*,* H *is a* subgroup of G *(noted* $H \leqslant G$*) if* H *satisfies the group axiom under the binary operation of* G*.*

Definition 4 (Subgroup generated by a subset of a group G**)** *If* S *a subset of a group* G*, the subgroup generated by* S *is* $\langle S \rangle_G = \{s_1^{\epsilon_1} \ldots s_k^{\epsilon_k} | s_i \in S\}$*. A subgroup* $H \leqslant G$ *is* finitely generated *if there is a finite* $S \subseteq G$*, such that* $H = \langle S \rangle_G$*.*

Definition 5 (Coset, Normal Subgroup) *If* $H \leqslant G$ *are defined the following subset of* G*:*

- $gH = \{gh | h \in H\}$ *the* left coset of H
- $Hg = \{hg | h \in H\}$ *the* right coset of H.

Remark 1 *Two elements* g *and* g' *define the same left (right) coset* $gH = g'H$ *(*$Hg = Hg'$*) iff* $g^{-1}g' \in H$ *(iff* $g'g^{-1} \in H$*). That induce a partition on the elements of the group given by the left (right) cosets.*

Definition 6 (Transversal set) *If* $H \leqslant G$*, we can define a* transversal set $H^\perp$ *of* H *by choosing*[1] *a random elements of each coset.*

Given a subgroup H of G and a set $H^\perp$ of representatives of right cosets, we have a unique decomposition of each element of G:

Proposition 1 *For every* $g \in G$*, there is a unique decomposition* $g = hv$ *with* $h \in H$ *and* $v \in H^\perp$*.*

Proof: Because H induces a partition on G (given by its right cosets) and $g \in Hg$, there exists a unique $v \in H^\perp$ such that $Hg = Hv$. So $h = gv^{-1}$ is an element of H and $g = hv$.

Definition 7 (Normal Subgroup) *A subgroup* H *of* G *is* normal *if* $Hg = gH$ *for all* $g \in G$*. It is noted by* $H \trianglelefteq G$*.*

[1]We need the axiom of choice if [G:H] is not finite.

Definition 8 (Centralizer of x) *If $H \leqslant G$ and $x \in G$, the* centralizer of x in H *is the subgroup of H consisting of elements which commute with x: $C_H(x) = \{h \in H | xh = hx\}$.*

Definition 9 (Quotient Group) *A* quotient group *$Q = \frac{G}{\mathcal{R}}$ is a group obtained identifying together elements of a group G using an equivalence relation $\mathcal{R}$ compatible with the group operation. The elements of Q (class of equivalence) are usually noted by $[g]$. Furthermore the set $N := [1]_{\mathcal{R}}$ of elements equivalent to 1_G is a normal subgroup: for every $x \in G$, $Nx = xN$. Vice versa every $N \trianglelefteq G$ induce an equivalence relation on G given by $g \sim g' \Leftrightarrow gg'^{-1} \in N$.*

Definition 10 (Group Homomorphism) *Let G, G' be groups an* group homomorphism *ϕ is a map $\phi : G \to G'$ such that: $\phi(g *_G g') = \phi(g) *_{G'} \phi(g')$*

Definition 11 (Group Isomorphism) *If $\phi : G \to G'$ is a group homomorphism, ϕ is an isomorphism if it is bijective.*

Definition 12 (Local isomorphism) *A* local isomorphism *of G is an isomorphism $\phi : H \to H'$ between two subgroups H and H' of G. An element $t \in G$* represents *ϕ if $\forall x \in H$, $\phi(x) = txt^{-1}$. A subgroup K of G is ϕ-*invariant *if $\phi(H \cap K) = \phi(H' \cap K)$.*

1.1.2 Monoid theory

Definition 13 (Monoid) *A* monoid *$M = (S, *)$ is given by a set S and a* binary operation *$*$ on S:*

$$* : S \times S \to S$$

satisfying the following axioms:

- *(Closure): $\forall g, g' \in G,\ g * g' \in G$;*
- *(Associativity): $\forall g, g', g'' \in G,\ g * (g' * g'') = (g * g') * g''$;*
- *(Identity): $\exists e \in G$ such that $\forall g \in G, e * g = g = g * e$.*

Definition 14 (Submonoid) *A subset $N \subseteq M$ is a submonoid of M if it contains the unity and it is closed under the binary operation induced by that of M (i.e. for every $x, y \in N$, $xy \in N$).*

Definition 15 (Quotient Monoid) *If M is a monoid and $\sim$ an equivalence realtion on M, the* quotient monoid *$\frac{M}{\sim}$ is the monoid with elements the equivalene class of M relative to $\sim$ and the operation is derived by the M's one.*

Definition 16 (Monoid Homomorphism/Isomorphism) *If M and N are two monoids, a map $f : M \to N$ is an* homomorphism *if $f(xy) = f(x)f(y)$ for all $x, y \in M$. An homomorphism f is an* isomorphism *if it is bijective.*

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 17 (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.*

Notation: In order to view a presentation as a *string rewriting system*[2], the pair (w, w') will be also denoted as a *reduction rule* $w \to w'$.

Notation: $M = \langle\Sigma|\mathcal{R}\rangle^+$ means that M is equal to the quotient of Σ^* by the congruence $\leftrightarrow^*_{\mathcal{R}}$ generated by $\mathcal{R}$ (the smallest equivalence relation containing $\mathcal{R}$ and compatible with the multiplication). Similarly, a presentation of a group is given by an alphabet Σ and a set of pair of words on the alphabet $\Sigma \cup \bar{\Sigma}$ where $\bar{\Sigma} = \{\bar{\sigma}|\sigma \in \Sigma\}$. $G = \langle\Sigma|\mathcal{R}\rangle$ means that G it's equal to the monoid $\langle\Sigma \cup \bar{\Sigma}|\mathcal{R} \cup \mathcal{I}_\Sigma\rangle^+$ where $\mathcal{I}_\Sigma = \{(\sigma\bar{\sigma}, 1)(\bar{\sigma}\sigma, 1)|\sigma \in \Sigma\}$.

Notation: Given a presentation $(\Sigma|\mathcal{R})$ and two words $v, w \in \Sigma^*$, $v \equiv w$ means that v and w are the same word (written with the same letters in the same order), while $v =_M w$ means that they are equivalent in the quotient M (if there is no ambiguity it will be denoted $=$).

Example: $\mathbb{Z} \simeq \langle b|\varnothing\rangle =: F_1$ has a *canonical presentation* $\langle b\rangle := \langle b|\varnothing\rangle$ as a group and a *canonical presentation* $(\{b, \bar{b}\}|\mathcal{R}_b = \{(\bar{b}b, 1), (b\bar{b}, 1)\})$ like monoid. If $w \equiv b\bar{b}, w' \equiv \bar{b}b$ so $ww' \equiv b\bar{b}^2b = 1$.

Notation: Words on an alphabet Σ will be noted by small or capital letters. If $w_1, \ldots, w_n \in \Sigma^*$, then $W(w_1, \ldots, w_n)$ is a word $W \in \Sigma^*$ such that every word is written in therm of $w_1, \ldots, w_n$ i.e. $W \equiv W_1 \ldots W_k$ with $W_j \equiv w_i, \forall 1 \leqslant j \leqslant k \exists 1 \leqslant i \leqslant n$.

It's preferable to continue to distinguish the two equivalences $=$ and $\leftrightarrow^*_{\mathcal{R}}$ since there is a subtle difference between $=$ and $\leftrightarrow^*_{\mathcal{R}}$: the first is the equality in the semantics of M, the abstract algebraic object, while the second is the equality in the syntax of the quotient, so this equality depends from the rewriting system chosen. If there is not ambiguity (a single system is given) or if all

[2]see. Appendix A

systems have the same property booth notation will be used with the same meaning.

Remark 2 *A group is* finitely presented *if it can be is finitely presented as monoid.*

Proof: Given a finite presentation $(\Sigma|\mathcal{R})$ of a group G, we get

$$G = \langle \Sigma \cup \bar{\Sigma} | \mathcal{R} \cup \mathcal{I}_\Sigma \rangle^+$$

where, if Σ is finite and so $\bar{\Sigma} = \{\bar{\sigma} | \sigma \in \Sigma\}$ and $\mathcal{I}_\Sigma = \{(\sigma\bar{\sigma}, 1), (\bar{\sigma}\sigma, 1) | \sigma \in \Sigma\}$ are finite too.

Definition 18 (Reductions) *If $u, v \in \Sigma^*$ and $(r, s) \in \mathcal{R}$, we get an* elementary reduction *$urv \to_{\mathcal{R}} usv$. If there is a sequence $u_0, u_1 \ldots u_n$ in Σ^* such that $u_i \to_{\mathcal{R}} u_{i+1}$ for $i = 0 \ldots n-1$ we get a* composed reduction *$u_0 \to^*_{\mathcal{R}} u_n$. A word w is* reduced *if there is no v such that $w \to_{\mathcal{R}} v$. If a word u admits a single reduced word $\hat{u}$ such that $u \to^*_{\mathcal{R}} \hat{u}$, then $\hat{u}$ is called its* normal form.

Definition 19 (Convergent presentation) *A presentation $(\Sigma|\mathcal{R})$ is* noetherian *if there are not infinite sequence $\{u_i\}_{i\in\mathbb{N}}$ such that $u_i \to_{\mathcal{R}} u_{i+1} \forall i \in \mathbb{N}$. A presentation is convergent if, moreover, it have the confluence propriety: for every u, v, v' such that $u \to^*_{\mathcal{R}} v$ and $u \to^*_{\mathcal{R}} v'$ there is some w such that $v \to^*_{\mathcal{R}} w$ and $v' \to^*_{\mathcal{R}} w$.*

$$\begin{tikzcd} & u \arrow[dl, "*"'] \arrow[dr, "*"] & \\ v \arrow[dr, "*"'] & & v' \arrow[dl, "*"] \\ & w & \end{tikzcd}$$

Figure 1.1: Confluent diagram

Definition 20 *A* subword *of a word v is a word w such that $v = uwu'$ for some $u, u' \in \Sigma^*$ (u u' can be empty). The* overlap *of two subword u and w of v is the longest word v' such that $u = u'v'$, $w = v'w'$ and $u'v'w'$ is a subword of v. If w is a subword of v we say that v contains w, moreover if $v = wu$ ($v = uw$) $\exists u \in \Sigma^*$, w it's a prefix (suffix) of v.*

Notation: We write $sub(v)$ for the set of subwords of v.

Definition 21 (Critical Peak) *Given a presentation $(\Sigma|\mathcal{R})$, a* critical peak *is a word w containing two subwords v and v' with non-empty overlap such that v and v' are respectively the prefix and the suffix of w (or $v = w$ and $v' \in sub(w)$) and $(v, u), (v', u') \in \mathcal{R}$ for some u, u'. We'll say that a critical peak w is* solvable *if booth path of reduction starting from the word w converge to some word w'.*

Definition 22 (Standard presentation of a group) *If G is a group, its* standard presentation $(\Sigma_G|\mathcal{R}_G)$ *is given by* $\Sigma_G = \{a_x|\ x \in G\}$ *and*

$$\mathcal{R}_G = \{a_1 \to 1, a_x a_y \to a_{xy} | x, y \in G\}.$$

Remark 3 *The standard presentation of G is convergent.*

Proof: The confluence follows from the associativity of the group operation (i.e. $\forall x, y, z \in G,\ x(yz) = (xy)z$. Indeed, critical peaks are of the following forms:

$$\begin{tikzcd}
& a_x a_y a_z \arrow[dl] \arrow[dr] & \\
a_{xy} a_z \arrow[d] & & a_x a_{yz} \arrow[d] \\
a_{(xy)z} & = & a_{x(yz)}
\end{tikzcd}$$

Figure 1.2: Critical peaks for the standard presentation of a group G

The termination, instead, is guaranteed by the fact that every reduction reduces the length of a word by one, and so, the reduced words are the letters and the empty word:

$$a_{x_1} a_{x_2} \dots a_{x_n} \to^*_{\mathcal{R}_G} a_{x_1 \dots x_n}, \qquad a_1 \to^*_{\mathcal{R}} 1$$

Definition 23 (Free Product) *Let* $G = \langle \Sigma_G|\mathcal{R}_G\rangle^+$ *and* $H = \langle \Sigma_H|\mathcal{R}_H\rangle^+$. *Then the* free product $F = G * H$ *has a presentation, as monoid, given by the disjoint union of the presentations of G and H:* $F = \langle \Sigma_G \uplus \Sigma_H|\mathcal{R}_G \uplus \mathcal{R}_H\rangle^+$.

Notation: we'll note F_1 the free group with one generato (i.e. $F_1 \simeq \mathbb{Z}$). F_n will note the free group on n generators $F_n = \langle a_1, \dots, a_n\rangle = F_{1_1} * \dots * F_{1_n}$ and F_ω the free group of $\aleph_0$ generators $\{\alpha_n\}_{n\in\mathbb{N}}$.

Definition 24 (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.*

Proof: *Let $[v]_{\mathcal{R}}$ be the equivalence classes of v with respect to $\leftrightarrow^*_{\mathcal{R}}$, it suffice to prove that* $[v]_{\mathcal{R}} = [v]_{\mathcal{R}'} \cap \Sigma^*$

$\subseteq$) *Since $\mathcal{R} \subseteq \mathcal{R}'$ if $w \in \Sigma^*$ and $w \leftrightarrow^*_{\mathcal{R}} v$ then $w \leftrightarrow^*_{\mathcal{R}'} v$;*

$\supseteq$) *Let $w \in \Sigma'^*$ such that $w \leftrightarrow^*_{\mathcal{R}'} v$. Then, since $(\Sigma'|\mathcal{R}')$ is convergent, there is $u \in \Sigma'^*$ such that $w \rightarrow^*_{\mathcal{R}'} u$ and $v \rightarrow^*_{\mathcal{R}'} u$. For every $v \in \Sigma^*$, applying a rewriting rule of $\mathcal{R}'$, we get a word in Σ^*, so that $u \in \Sigma^*$. If also $w \in \Sigma^*$ then $w \leftrightarrow^*_{R} v$.*

Since $\bar{\phi}$ is well defined and for every $v, w \in \Sigma^*$, $v \leftrightarrow^*_{\mathcal{R}'} w$ iff $v \leftrightarrow^*_{\mathcal{R}}$, $\hat{\phi}$ is an injective homomorphism.

Definition 25 (Embedding Translation) *An* embedding translation $\bar{\phi} : (\Sigma|\mathcal{R}) \rightarrow P' = (\Sigma'|\mathcal{R}')$ *is a translation such that:*

- *P' is convergent on $\phi(\Sigma^*)$;*
- $\exists$ *a* control function[3] $\psi : \Sigma'^* \rightharpoonup \Sigma^*$ compatible with $\leftrightarrow^*_{\mathcal{R}'}$, *i.e. such that $\forall v \in \Sigma^*$, $\psi(\phi(v)) \leftrightarrow^*_{\mathcal{R}} v$.*

Lemma 2 (Extended embedding lemma) *If exists a embedding translation $\bar{\phi} : (\Sigma|\mathcal{R}) \rightarrow P' = (\Sigma'|\mathcal{R}')$, so exist an homomorphism $\hat{\phi} : \langle\Sigma|\mathcal{R}\rangle^+ \hookrightarrow \langle\Sigma'|\mathcal{R}'\rangle^+$.*
Proof: *We define $\hat{\phi}([w]_{\mathcal{R}}) = [\phi(w)]_{\mathcal{R}'}$ and $\hat{\psi}([v]_{\mathcal{R}'}) = [\psi(v)]_{\mathcal{R}}$. Like in Lemma 1 will be necessary to demonstrate $\bar{\phi}([v]_{\mathcal{R}}) = [\bar{\phi}(v)]_{\mathcal{R}'}$:*

$\subseteq$) *since $\bar{\phi}(\mathcal{R}) \subseteq \leftrightarrow^*_{\mathcal{R}'}$ if $w \in \Sigma^*$ and $w \leftrightarrow^*_{\mathcal{R}} v$ then $\bar{\phi}(w) \leftrightarrow^*_{\mathcal{R}'} \bar{\phi}(v)$*

$\supseteq$) *let $w \in \Sigma^*$ if $\bar{\phi}(w) \leftrightarrow^*_{\mathcal{R}'} \bar{\phi}(v)$ then $w \leftrightarrow^*_{\mathcal{R}} v$. Since $(\Sigma'|\mathcal{R}')$ is convergent, exists an unique $\hat{v} \in \Sigma^*$ such that $\bar{\phi}(v) \rightarrow^*_{\mathcal{R}'} \bar{\phi}(\hat{v})$ and $w \rightarrow^*_{\mathcal{R}'} \bar{\phi}(\hat{v})$. Since ψ is compatible with $\leftrightarrow^*_{\mathcal{R}'}$ and if $z \in \Sigma^*$ every rewriting rule in the path of reduction from a $\bar{\phi}(z)$ to $\bar{\phi}(\hat{v})$ is in $\bar{\phi}(\mathcal{R})$, so $\bar{\phi}(z) \leftrightarrow^*_{\mathcal{R}'} \bar{\phi}(\hat{v})$ iff $z \leftrightarrow^*_{\mathcal{R}} \psi\bar{\phi}(z) \leftrightarrow^*_{\mathcal{R}} \psi\bar{\phi}(\hat{v}) \leftrightarrow^*_{\mathcal{R}} \hat{v}$.*

Definition 26 (Iso-translation) *An* iso-translation *between two presentation $\bar{\phi} : (\Sigma|\mathcal{R}) \rightarrow (\Sigma'|\mathcal{R}')$ is an embedding translation such that for every equivalence class $[v']_{\mathcal{R}'}$ of Σ'^* exists at least a $v \in \Sigma^*$ such that $\phi(v) \in [v']_{\mathcal{R}'}$.*

Proposition 2 *If exists a iso-translation $\bar{\phi} : (\Sigma|\mathcal{R}) \rightarrow (\Sigma'|\mathcal{R}')$, so $M = \langle\Sigma|\mathcal{R}\rangle^+$ and $M' = \langle\Sigma'|\mathcal{R}'\rangle^+$ are isomorph.*
Proof: *By lemma 2 $M \hookrightarrow M'$. Moreover $\bar{\phi}(\Sigma^*) = \Sigma'^*$ is a bijection with the property $\bar{\phi}(ww') = \bar{\phi}(w)\bar{\phi}(w')$, so an isomorphism.*

Definition 27 (Lexico-metric order) *Given an alphabet Σ equipped with an order $<_\Sigma$ ($\alpha =_\Sigma \beta$ means $\alpha \leqslant \beta \wedge \beta \leqslant \alpha$), $v = \alpha_{i_1} \cdots \alpha_{i_n}$ and $w = \alpha_{j_1} \cdots \alpha_{j_m}$, it's possible to extend it to a* lexicografic order *on the set of words on Σ:*

$$v <_\Sigma w \Leftrightarrow \exists k \forall h < k(\alpha_{i_h} =_\Sigma \alpha_{j_h} \wedge (k = n < m \vee (k \leqslant m \wedge k < n \wedge \alpha_{i_k} <_\Sigma \alpha_{j_k})$$

and also to lexico-metric order*:*

$$v \lhd_{(\Sigma,<_\Sigma)} w \Leftrightarrow n < m \text{ or } \exists k \leqslant n \forall h < k(\alpha_{i_h} =_\Sigma \alpha_{j_h} \wedge \alpha_{i_k} <_\Sigma \alpha_{j_k})$$

Example: Let $\Sigma = \{a, b, c\}$ and with the order $a =_\Sigma b <_\Sigma c$ so $abc <_\Sigma bca$, and $abc \lhd bca$ but $aabca <_\Sigma bca$ and $bca \lhd aabca$.

[3]it can be a partial function

Theorem 3 *It exists an embedding of F_ω into F_2.*
Proof: *Like in [15], showing that the family $\{b^n ab^{-n}\}_{n\in\mathbb{Z}}$ is free*[4] *in the group $F_2 = \langle a, b\rangle$, it's possible to have the embedding translation of $\bar{\phi} : F_\omega \to F_2$ given by $\phi(\alpha_n) = b^n ab^{-n}$ and so the proof by Lemma 2. In order to founf the control function of the translation, we'll build a new convergent presentation of*

$$F_2 = \langle \Sigma = \{a, \bar{a}, b, \bar{b}\} | \mathcal{R} = \{a\bar{a} \to 1, \bar{a}a \to 1, b\bar{b} \to 1, \bar{b}b \to 1\}\rangle^+$$

it suffices to add, for every $n > 0$, the superfluous generators[5] *given by the relation:*

$$a_n = b^n a\bar{b}^n \qquad \bar{a}_n = b^n \bar{a}\bar{b}^n \qquad a_{-n} = \bar{b}^n ab^n \qquad \bar{a}_{-n} = \bar{b}^n \bar{a}b^n.$$

The following relation will be derivable for every $n \in \mathbb{Z}$ (nominally $a_0 := a$):

$$a_n\bar{a}_n = 1 \quad \bar{a}_n a_n = 1$$

$$ba_n = a_{n+1}b \quad b\bar{a}_n = \bar{a}_{n+1}b \quad \bar{b}a_n = a_{n-1}\bar{b} \quad \bar{b}\bar{a}_n = \bar{a}_{n-1}\bar{b}.$$

Let $\Sigma_2 = \{b, \bar{b}\} \cup \{a_n, \bar{a}_n\}_{n\in\mathbb{Z}}$, a new presentation of F_2 is given by $\langle \Sigma_2 | \mathcal{R}_2\rangle$ where $\mathcal{R}_2$ consists of the following reduction rules (varying $n \in \mathbb{Z}$):

$$a_n\bar{a}_n \to 1 \qquad \bar{a}_n a_n \to 1 \qquad b\bar{b} \to 1 \qquad \bar{b}b \to 1$$

$$ba_n \to a_{n+1}b \qquad b\bar{a}_n \to \bar{a}_{n+1}b \qquad \bar{b}a_n \to a_{n-1}\bar{b} \qquad \bar{b}\bar{a}_n \to \bar{a}_{n-1}\bar{b}.$$

Defining the order on Σ_2 by $\forall n,\ a_n =_{\Sigma_2} a_{n+1} =_{\Sigma_2} \bar{a}_n <_{\Sigma_2} b =_{\Sigma_2} \bar{b}$, is possible to define a lexico-metric order $\lhd_{\Sigma_2}$ on Σ_2^*. The rewriting system is so noetherian since for every reduction $w \to_{\mathcal{R}_2} w'$, $w' \lhd w$ and $\lhd$ it's a well-order on Σ_2^*. By this order every reduced word will be in the form $\alpha_1 \dots \alpha_n \beta_i^k$ with $\alpha_i \in \{a_n, \bar{a}_n\}$ and $\beta \in \{b, \bar{b}\}$. Moreover all the critical picks are solvable:

- *For every $(\gamma, \gamma') \in \{(a_n, \bar{a}_n), (\bar{a}_n, a_n), (b, \bar{b}), (\bar{b}, b)\}$*

$$\begin{tikzcd}
 & \gamma\gamma'\gamma \arrow[dl] \arrow[dr] & \\
\gamma \arrow[rr, phantom, "="] & & \gamma
\end{tikzcd}$$

- *For every $(\alpha_n, \alpha'_n) \in \{(a_n, \bar{a}_n), (\bar{a}_n, a_n)\}$*

$$\begin{tikzcd}
 & \bar{b}\alpha_n\alpha'_n \arrow[dl] \arrow[ddd] & & b\alpha_n\alpha'_n \arrow[ddd] \arrow[dr] & \\
\alpha_{n-11}\bar{b}\alpha'_n \arrow[d] & & & & \alpha_{n+1}b\alpha'_n \arrow[d] \\
\alpha_{n+1}\alpha'_{n+1}\bar{b} \arrow[dr] & & & & \alpha_{n+1}\alpha'_{n+1}b \arrow[dl] \\
 & \bar{b} & & b &
\end{tikzcd}$$

[4]i.e. there are not relations between the elements
[5]them can be viewed like some abbreviation of some word in F_2

- *For* $(\gamma, \gamma', \delta) \in \{(b, \bar{b}, -1), (\bar{b}, b, +1)\}$

$$\begin{tikzcd}
 & \gamma\gamma'\alpha_n \arrow[dl] \arrow[ddd] \\
\gamma\alpha_{n+\delta}\gamma \arrow[d] & \\
\alpha_n\gamma\gamma' \arrow[dr] & \\
 & \gamma
\end{tikzcd}$$

The equivalence of the two presentation is provable showing the existence of an iso-translation $\bar{\phi}' : (\Sigma|\mathcal{R}) \to (\Sigma_2|\mathcal{R}_2)$ *given by* $\bar{\phi}'(a) = a_0$, $\bar{\phi}'(\bar{a}) = \bar{a}_0$, $\bar{\phi}'(\beta) = \beta$ *where* $\beta = b, \bar{b}$*. The control function* ψ' *is defined by* $\psi'(\beta) = \beta$ *and* $\psi'(\alpha_n) = b^n\alpha b^{n-1}$ *where* $\beta = b, \bar{b}$ *and* $\alpha = a, \bar{a}$*.*

Now it's easy to show that the map $\phi : \Sigma_\omega = \{\alpha_n, \bar{\alpha}_n\}_{n\in\mathbb{Z}} \to \Sigma_2^*$ *such that* $\phi(\alpha_n) = a_n$ *and* $\phi(\bar{\alpha}_n) = \bar{a}_n$ *induce an embedding translation* $\bar{\phi} : \langle\Sigma_\omega\rangle \to (\Sigma_2|\mathcal{R}_2)$*. Since every word in* $\phi(\Sigma_\omega)$ *are in* $\{a_n, \bar{a}_n\}^*_{n\in\mathbb{N}}$*, them are in normal form in* $(\Sigma_2|\mathcal{R}_2)$ *and it's possible to define* $\psi : \Sigma_2^* \to \Sigma_\omega$ *inductively on the number* N_w *of* a *and* $\bar{a}$ *in the word* w*: if* $N_w = 0$ *so* $\psi(w) = 1$*. Else* $w = B(b, \bar{b})\alpha w'$ *where* $\alpha = a$ *or* $\bar{a}$*,* $B(b, \bar{b})$ *is a word containing only occurrences of* b *and* $\bar{b}$*, so* $\psi(w) = a_n\psi(\beta w')$ *where* $n = (\#$*occurence of* b *in* $B(b, \bar{b})) - (\#$*occurence of* $\bar{b}$ *in* $B(b, \bar{b}))$ *and* $N_{w'} < N_w$*. That satisfy* $\forall w \in \Sigma_\omega$*,* $\psi(\phi(w)) = w$*. So* $F_\omega = \langle\Sigma_\omega\rangle \hookrightarrow (\Sigma_2|\mathcal{R}_2) = \langle a, b\rangle = F_2$*.*

Lemma 4 $\forall p, q \in \mathbb{Z}, q \neq 0$ *the family* $\{a_p, b^q\}$ *is free in* F_2

Proof: Because $\{a, b\}$ is free in F_2 and $ord(b) = \infty$, $\{a, b^q\}$ is free in F_2 (if not it means exists relations between a and b). So $\{a_p, b^q\}$ have to be free since $\langle a_p, b_q\rangle$ can be obtained from the free sub-group of F_2 generated by $\{a, b^q\}$ applying the internal isomorphism $x \to b^p x b^{-p}$.

Definition 28 (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

Definition 29 (Turing Machine) *A Turing machine is a abstract machine consisting of:*

- *An* infinite tape *containing* cells *in which are written symbol of a fixed* alphabet*;*
- *A* head *which can* read and write *symbols on the tape and move left (L) or right (R) on it;*
- *A* set of instruction *for the head depending of the input and the current state.*

it can be visualized as above:

$$q_j$$
$$\leftarrow \bigtriangledown \rightarrow$$

$\dots$	$a_{i_{k-1}}$	a_k	a_{k+1}	$\dots$

More formally a Turing machine M is a 5-tuple $(Q, \Sigma, q_0, \perp, \delta)$ where:

- *Q is the set of states;*
- *Σ an alphabet with special symbol $\square$ representing an empty cell (it is not explicitly written in the list of symbols in Σ);*
- *$q_0 \in Q$ is the* initial state*;*
- *$\perp \subset Q$ is a set of* final states*;*

- $\delta : \Sigma \times Q \backslash \perp \to \Sigma \times Q \times \{L, R\}$ *is the* transition function.

Definition 30 (Configuration for a Turing machine) *Given a Turing machine M, a* configuration *for M is a string $s = a_{i_1} \dots a_{i_{k-1}} q_j a_{i_k} \dots a_{i_n}$ where $a_{i_1} \dots a_{i_k-1} a_{i_k} \dots a_{i_n}$ is a word in the alphabet Σ representing the content of the tape and a state q_j is positioned before the symbol read by the head.*

An initial configuration *is a configuration containing q_0, a* final configuration *is a configuration where $q_j \in \perp$.*

A configuration s_{n+1} is derivable *by $s_n = a_{i_1} \dots a_{i_{k-1}} q_j a_{i_k} \dots a_{i_n}$ if*

$$s_{n+1} = \begin{cases} a_{i_1} \dots a_{i_{k-2}} q'_j a_{i_{k-1}} a'_{i_k} a_{i_k} \dots a_{i_n} & \text{if } \epsilon = L \\ a_{i_1} \dots a_{i_{k-1}} a'_{i_k} q'_j a_{i_{k+1}} \dots a_{i_n} & \text{if } \epsilon = R \end{cases} .$$

where $\delta(a_{i_k}, q_j) = (a'_i, q'_j, \epsilon)$. In that case we'll note $s_n \to_M s_{n+1}$ a transition *of T.*

Definition 31 (Computation) *A computation of a Turing machine M is a sequence of configurations $s_0, s_1, \dots$ starting with an initial configuration such that $s_n \to_M s_{n+1}$ for all $n \geqslant 0$. Intuitively is sequence of the configurations of the machine during the computing started with the configuration s_0. A computation terminate if it is in the form $s_0 \to \dots \to s_n$ with s_n such that there exist not a configuration s_x such that $s_n \to s_x$. Else we say that the computation diverge.*

Remark 4 *If $s_\perp$ is a transition containing a final state (a final configuration), there exists not s such that $s_\perp \to_M s$, so a computation can contain at most one final configuration at the end of a terminating computation. Moreover form a configuration s_0 there exists only one computation starting with s_0.*

With that definition of computation will so be possible to givean idea of *complexity* of a terminating computation $s_0, \dots, s_\perp$ of M in term of length $|s_0| = n$ of the initial configuration: we'll say that M operate in time $f(n)$ if $s_{f(n)} = s_\perp$ or in space $g(n)$ if it is the size of longer configuration of the computation (i.e. $g(n) = max\{|s_i|\}$).

There exist some variant of Turing machine equivalent: multi-tape machine (with dependent or independent heads) and machine where head can write without moving. A *non-deterministic* Turing machine is a machine where δ is not a function but a relation: is so possible that the computation will "branch" at some point of the computing. That defines different classes of complexity of computation.

Like we have seen, every algorithm can be implemented by a Turing machine. We'll prove the next proposition (in the original formulation given by Turing) using this fact, in order to to give the idea of what we are trying to build, without loosing ourself in the mathematical rigorous formalism.

Proposition 3 (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.*

Proof: First of all we'll need to explicit what is a "standard description" of a Turing machine. The idea is to encode the machine in enother language (for example the binary one) encoding like a list the alphabet (also the empty symbol $\square$ of the machine), the set of states, L and R and the function δ in form of 5-tuple.

Now is possible to build a *Universal Machine U* capable to contain on the tape the encoding of the machine M we want to simulate and an initial configuration for this machine (encoded with the same encoding used for M) in two different part of the tape. The computation will proceed on the part of the tape containing the configurations, searching every time the transition to execute in order to obtain the new one, in the part of the tape containing the machine encoding.

Definition 32 (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.

Definition 33 (Register Machine) *A* register machine *is an abstract machine $\mathcal{M}$ consisting of:*

- *Labeled unbounded integer-value register: any labeled register can hold a single non-negative integer;*
- *A list of (labeled) sequential instructions s_i (we'll call them* routine*) in the form:*
  - *$INC(r, s_j)$ = increase r and go to s_j ;*
  - *$JZDEC(r, s_j, s_k)$ = if $r = 0$ go to s_j, else decrease r and go to s_k;*
  - *$HALT$;*
- *A state register: which hold the label of the instruction to execute. A configuration for a 2- register machine $\mathcal{M}$ is a triple (s, a, b) where a, b represent the integers in registers and s a state. The writing $\mathfrak{s} \to_{\mathcal{M}} \mathfrak{s}'$ ($\mathfrak{s} \to^*_{\mathcal{M}} \mathfrak{s}'$) denote that $\mathcal{M}$ transform a configuration $\mathfrak{s}$ in a configuration $\mathfrak{s}'$ in one step (a finite number of steps). A state $(0, a, b)$ will denote a final state.*

A non-deterministic register machine is a machine where there could be instructions labeled by the same letter, in that case the machine will branch the computation executing one of them for every branch.

Remark 5 *It's possible to define different equivalent machine with different form of instructions [17], for example if we build a machine without $JZDEC$ instructions but with instructions $DEC(r, s_j)$ (decrease r go to s_j) and $ZeroTest(r, s_i)$ (if $r = 0$ do s_i else execute the next instruction of the list) instead of the instruction $JZDEC$ in the list of instruction we can express machines computing the same functions, will refer to this particular formulation of register machine as* Minsky machine *$M = (Q, \delta, a_1, \dots, a_n)$ where Q is the set of labeling for instruction, δ the list of instruction of M and the a_i the registers.*

Lemma 5 *There is a routine for a 2-register machine with the second register empty which computes the multiplication by a fixed k of the content of the first register in it.*
Proof: *Let s_0 the initial state and the number α content in the register a. The list instruction will be in the form:*

s_0 $JZDEC(a, s_t, s_1)$;

s_i $INC(b, s_{i+1})$ *for all* $i = 1\ldots, k-1$;

s_k $INC(b, s_0)$;

s_t $JZDEC(b, s_\perp, s_c)$;

s_c $INC(a, s_t)$;

$s_\perp$ $HALT$.

It will need $k + 4$ states.

Lemma 6 *There is a routine for a 2-register machine with the second register empty which compute the division of the content α of the first register by k (recording it in the first register), and if the remainder is different from 0 it will preserve α.*
Proof: *Let s_0 the initial state and the number α content in the register a and β in b. The machine will subtract if is possible $i = k$ times 1 to α adding 1 to β and, at the end this procedure, it will copy β in the empty register a, else will re-add $i < k$ times 1 to α (where i is the time it subtracted 1) and after β times k times 1. The list instruction will be in the form:*

s_i $JZDEC(a, s_{b(i)}, s_{i+1})$ *for all* $i = 0\ldots, k-1$;

s_k $JZDEC(a, s_p, s_+)$;

s_+ $INC(b, s_0)$;

s_p $INC(b, s_f)$;

s_f $JZDEC(b, s_\perp, s_c)$;

s_c $INC(a, s_f)$;

$s_{b(i)}$ $INC(a, s_{b(i-1)})$ *for all* $i = 1, \ldots, k-1$;

$s_{b(0)}$ $JZDEC(b, s_\perp, s_{+_k})$;

s_{+_i} $INC(a, s+_{i-1})$ *for* $i = k, \ldots, 1$;

s_{+_0} $JZDEC(b, s_\perp, s_{+_k})$;

$s_\perp$ $HALT$;

It will need $2k + 7$ states.

Theorem 7 *Every n-register machine R can be simulated by a 2-register machine $R_2(R)$.*
Proof: *Let $p_1, \ldots, p_n$ be the first n prime numbers. Is so possible to encode the string of the registers' contents $a_1, \ldots, a_n$ as $\gamma(a_1, \ldots, a_n) = p_1^{a_1} \cdots p_n^{a_n}$. In this encoding add(subtract) 1 to the i-esim register correspond to multiply (divide) for p_i the number $\gamma(a_1, \ldots, a_n)$. Will be so possible to simulate the computing of R_n by a 2-register machine $R_2(R)$ since every configuration $(s, \alpha_1, \ldots, \alpha_n)$ will correspond to the configuration $(s_0(s), \Pi p_i^{\alpha_i}, 0)$ of $R_2(R)$, where $s_0(s)$ is the first instruction of the routine used to simulate the instruction of s using the a variant of Lemma 5 for the instruction INC and Lemma 6 for $JZDEC$.*

Definition 34 (Modular machines [1]) *A* modular machine *$\mathcal{M}od$ is defined, fixed an $m \in \mathbb{N}$, by a "set of instruction" (a, b, c, ϵ) of quadruples where $0 \leqslant a, b \leqslant m$, $0 \leqslant c \leqslant m^2$, $\epsilon = R, L$ (at most one quadruple can begin with the same pair a and b), and an integer $0 < n < m$ to define input and output functions. A configuration for $\mathcal{M}od$ is a pair (α, β) where $\alpha = um + a$, $\beta = vm + b$. If no quadruple begins with a, b, (α, β) it's called* terminal, *else $(\alpha, \beta) \rightarrow_{\mathcal{M}od} (\alpha', \beta')$ where*

$$(\alpha', \beta') = \begin{cases} (um^2 + c, v) & \text{if } \epsilon = R \\ (u, vm^2 + c) & \text{if } \epsilon = L \end{cases}$$

The computing function of $\mathcal{A}$ is the partial function $u_{\mathcal{M}od} g_{\mathcal{M}od} i_{\mathcal{M}od} : \mathbb{N} \rightarrow \mathbb{N}$ defined by:

$$i_{\mathcal{M}od} : \mathbb{N} \rightarrow \mathbb{N}^2, \qquad r \rightarrow (\sum b_i n^i, n+1) \text{ where } r = \sum b_i n^i,\ 0 \leqslant b_i < n$$

$$g_{\mathcal{M}od} : \mathbb{N}^2 \rightarrow \mathbb{N}^2, \qquad (\alpha, \beta) \rightarrow^*_{\mathcal{A}} (\alpha', \beta'), \qquad (\alpha', \beta') \text{ terminal}$$

$$u_{\mathcal{M}od} : \mathbb{N}^2 \rightarrow \mathbb{N}, \qquad (\alpha', \beta') \rightarrow \sum_1^k b_i m^{i-1} \text{ where } \alpha = \sum b_i m^i,\ 0 \leqslant b_i < n$$

where $k = min\{i | b_i = 0\}$.

Remark 6 *It is so possible, with a proper encoding, to utilize a modular machine to simulate a Turing machine: starting by a Turing machine T on the alphabet $\{b_i\}_{0 \leqslant i < m}$ is possible to associate the coding of the two parts of the $\alpha = \sum_{i=h}^{0} c(i) m^i$ and $\beta = \sum_{i \geqslant h+1} c(i) m^{h+1-i}$ respectively at left and right of the head ($c(i)$ is the content of the the i^{th} cell of the tape), at every state of T a quadruple of $\mathcal{M}od$.*

Definition 35 (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 *$p + qz \rightarrow_{\mathcal{A}} p' + q'z$ ($z \in \mathbb{Z}$).*

Remark 7 *Every 2-register machines $\mathcal{M}$ can be simulated by an affine machine: let (s, a, b) a configuration for $\mathcal{M}$, coding it in the integer $[s, a, b] = s + m2^a 3^b$, every transition will be in the form:*

$$i + mk \rightarrow i + 2mk \qquad i + m(2z+1) \rightarrow j + m(2z+1) \qquad i + 2mz \rightarrow k + mz$$

$$i + mk \rightarrow i + 3mk \qquad i + m(3z+1) \rightarrow j + m(3z+1) \qquad i + 3mz \rightarrow k + mz$$

$$i + m(3z + 2) \to j + m(3z + 2)$$

so if z, z' are two integer, $z \leftrightarrow^*_{\mathcal{A}} z'$ so z is the code of a configuration iff z' is. Futhermore:

$$(s, a, b) \to_{\mathcal{M}} (s', a', b') \textit{ imply } (s, a, b) \leftrightarrow^*_{\mathcal{M}} (s', a', b') \textit{ iff } [s, a, b] \leftrightarrow^*_{\mathcal{A}} [s', a', b'].$$

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.*

Definition 36 (Primitive Recursive Function/Relation) *The class of primitive recursive function is the smallest class of function $\mathbb{N}^p \to \mathbb{N}$ for some $p \in N$ containing:*

- *Constant function: $C_n(x_1, \dots x_p) = n$ for any $n, p \in \mathbb{N}$;*
- *The function* successor*: $s(n) = n + 1$;*
- *The projections: $\pi_i^p(x_1, \dots, x_p) = x_i$;*

closed under:

- *Composition* $\circ$*: let* $h : \mathbb{N}^k \to N$ *and* $g_i : \mathbb{N}^p \to \mathbb{N}$ *so* $f = h \circ (g_1, \ldots, g_p) : \mathbb{N}^p \to \mathbb{N}$ *is defined* $f(x_1, \ldots x_p) = h(g_1(x_1, \ldots x_p), \ldots, g_k(x_1, \ldots x_p))$
- *Recursion schema* ρ*: let* $g : \mathbb{N}^p \to \mathbb{N}$ *and* $h : \mathbb{N}^{p+2} \to \mathbb{N}$*,*
  $f = \rho(g, h) : \mathbb{N}^{p+1} \to N$ *is defined* $f(0, x_1, \ldots x_p) = g(x_1, \ldots x_p)$ *and*
  $f(y+1, x_1, \ldots x_p) = h(y, f(y, x_1, \ldots x_p), x_1, \ldots x_p)$*;*

A relation is primitive recursive if its characteristic function is.

Proposition 4 *Any primitive recursive function is decidable.*
Proof: *constant, successor and projection are decidable. There exist also Turing machines to compute the composition of two function executing the second program on the output of the first and the recursion schema too.*

Proposition 5 *The following function and relation are primitive recursive:*

- *addiction;*
- *multiplication;*
- *pseudo-sottraction (* $n \dot{-} m = n - m$ *if* $n - m > 0$*, else* 0 *);*
- $mod(x, y)$*,* $div(x, y)$ *if* $x > y$ *where* $x = y\ div(x, y) + mod(x, y)$*;*
- $\alpha_2(x, y) = \frac{1}{2}(x + y)(x + y + 1) + y$*;*
- $\alpha_p(x_1, \ldots x_p) = \alpha_2(x_1, \alpha(x_2, \ldots, x_p))$*.*

Definition 37 (Recursive function) *The class of recursive function is the smallest class of function* $\mathbb{N}^p \to \mathbb{N}$ *for some* $p \in N$ *containing:*

- *Constant function:* $C_n(x_1, \ldots x_p) = n$ *for any* $n, p \in \mathbb{N}$*;*
- *The projections:* $\pi_i^p(x_1, \ldots, x_p) = x_i$*;*
- *Addiction;*
- *Multiplication;*
- *The characteristic function of the binary relation* $<$*;*

closed under:

- *Composition;*
- μ *schema:* $g(x_1, \ldots, x_n) = \mu y(f(x_1, \ldots, x_n, y) = 0)$ *is the smallest* z *such that* $f(x_1, \ldots, x_n, z) = 0$ *and for all* $z' < z$ $f(x_1, \ldots, x_n, z')$ *is defined. If such* z *doesn't exist* $g(x_1, \ldots, x_n)$ *is not defined.*

Since the function α_p *can encode the p-uple, we need a function capable to do the revers encoding:*

Proposition 6 *There is a recursive function* $\beta : \mathbb{N}^2 \to \mathbb{N}$ *such that, for all succession* $a_1, \ldots, a_n$ *of natural number, it exists* $c \in \mathbb{N}$ *such that* $\beta(i, c) = a_i$ *for all* $i \in \{1, \ldots, n\}$*.*

***Theorem 8 (Representability of recursive function in** PA_0**)** Every recursive total function is representable in PA_0 by a formula.*

We'll be so possible to encode the set of terms and formulas with natural number enumerating the set of variables and symbols of the syntax. The number associated to a term t or a formula F with this encoding is called Gödel number of t (F) noted $\#t$ ($\#F$).
Notation: *$\underline{n}$ is the term representing the number $n \in \mathbb{N}$ in the language $\mathcal{L}_0$ i.e. $s^n(0)$.*

Definition 38 (Encoding of** $\mathcal{L}_0$**)
Encoding of terms t:

- *if $t = \underline{0}$, $\#t = \alpha_3(0, 0, 0)$;*
- *if $t = x_n$ the n^{th} variable , $\#t = \alpha_3(n + 1, 0, 0)$;*
- *if $t = \underline{s}t_1$, $\#t = \alpha_3(\#t_1, 0, 1)$;*
- *if $t = t_1\underline{+}t_2$, $\#t = \alpha_3(\#t_1, \#t_2, 2)$;*
- *if $t = t_1\underline{\times}t_2$, $\#t = \alpha_3(\#t_1, \#t_2, 3)$;*

Encoding of formulas F:

- *if $F = t_1 \simeq t_2$, $\#t = \alpha_3(\#t_1, \#t_2, 0)$;*
- *if $F = \neg t_1 \simeq t_2$, $\#t = \alpha_3(\#t_1, \#t_2, 1)$;*
- *if $F = F_1 \wedge F_2$, $\#t = \alpha_3(\#F_1, \#F_2, 2)$;*
- *if $F = F_1 \vee F_2$, $\#t = \alpha_3(\#F_1, \#F_2, 3)$;*
- *if $F = \forall x_n G$, $\#t = \alpha_3(\#G, n, 4)$;*
- *if $F = \exists x_n G$, $\#t = \alpha_3(\#G, n, 4)$.*

Will be possible to encode the derivations and sets too.

Proposition 7 *The following function, relations and sets are recursive:*

- *$Ter_{\mathcal{L}_0}(n)$, n is the encoding of a term of $\mathcal{L}_0$;*
- *$For_{\mathcal{L}_0}(n)$, n is the encoding of a formula of $\mathcal{L}_0$;*
- *$\Phi(n)$, n is the encoding of a closed formula of $\mathcal{L}_0$;*
- *$Dim(x, \#F)$, if x is the encoding of a derivation of the formula $\#F$;*
- *$Sub(n, \#t, X) = \#X[t/x_n]$ where X is the number of a formula or a term.*

Definition 39 (Peano Axioms) *We'll denote PA_0 the following set of axioms:*

- *$\forall x \neg(sx \simeq \underline{0})$;*
- *$\forall x \exists y(\neg(x \simeq 0) \rightarrow sy \simeq x)$;*

- $\forall x \forall y (sx \simeq sy \rightarrow x \simeq y)$;
- $\forall x (x \underline{+} \underline{0} \simeq x)$;
- $\forall x \forall y (x \underline{+} sy \rightarrow x \simeq s(x \underline{+} y)$;
- $\forall x (x \underline{\times} \underline{0} \simeq \underline{0})$;
- $\forall x \forall y (x \underline{\times} sy \simeq (x \underline{\times} y) \underline{+} y)$.

PA is the set PA_0 united with the following infinite set of axioms definite by the variation of $F[x_0, \ldots, x_k]$ formula of $\mathcal{L}_0$:

- $\forall x_1, \ldots \forall x_k (F(\underline{0}, x_0, \ldots x_k) \wedge (F(y, x_1, \ldots, x_k) \rightarrow F(sy, x_1, \ldots, x_k)) \rightarrow \forall x_0 F(x_0, x_1, \ldots, x_k)$

Theorem 9 (Fixed point) *For all $A(x)$ closed formula with one free variable, there exist a closed formula B such that $AP_0 \vdash B \leftrightarrow A(x)$.*
Proof: *Let $x = x_k$ the free variable of A, $Num(n)$ is the function that given $n \in \mathbb{N}$ gives $\#n$ and $S(v, n, \underline{\#F})$ the formula in $\mathcal{L}_0$ who represents $v \simeq Sub(k, Num(n), \#F)$. Since $PA_0 \vdash \forall v (v \simeq \underline{Sub(k, n, n)} \leftrightarrow S(v, n, n))$ for all $n \in \mathbb{N}$, we pose:*

$$\Theta(x) = \forall x (S(y, x, x) \rightarrow A(y))$$

and $m = \#\Theta(x)$. Proving that $PA_0 \vdash \Theta(\underline{m}) \leftrightarrow A(\underline{Sub(k, m, m)})$, we found that $B = \Theta(\underline{m})$ is the searched formula.

Lemma 10 *Let $DIM(x, y)$ the formula of $\mathcal{L}_0$ to exprime $(x, y) \in Dim$ and $TH(x) = \exists y DIM(y, x)$ the proposition that exprim the existence of a derivation of the formula F, so:*

- *if $PA \vdash F$ so $PA \vdash TH(\underline{\# F})$;*
- $PA \vdash TH(\underline{\#F}) \rightarrow TH(\underline{\#TH(\underline{\#F})})$;
- $PA \vdash TH(\underline{\#A}) \wedge TH(\underline{\#A \rightarrow B}) \rightarrow TH(\underline{\#B})$;
- *if $PA \vdash A \rightarrow B$ so $PA \vdash TH(\underline{\#A}) \rightarrow TH(\underline{\#B})$.*

Theorem 11 (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.*
Proof: *By Teor.9 we know there is the formula G, fixed point of $\neg TH(x)$ such that $PA \vdash G \leftrightarrow \neg TH \underline{\#G})$.*

$PA \nvdash G$) *if $PA \vdash G$ so $PA \vdash TH(\underline{\#G})$ by lemma 10 and, by hypotesis $PA \vdash G \leftrightarrow \neg TH(\underline{\#G})$. Will so possible to deduce from $PA \vdash G$ and $PA \vdash G \leftrightarrow \neg TH(\underline{\#G})$ that $PA \vdash \neg TH(\underline{\#G})$ and so (by $PA \vdash TH(\underline{\#G})$) that $PA \vdash$, i.e. PA is not coherent.*

$PA \nvdash \neg G$) *prove $PA \vdash G \leftrightarrow \neg TH(\underline{\#G})$ is equal to prove*

$$PA \vdash (\neg G \vee \neg TH(\underline{\#G})) \wedge (G \vee \neg(\neg TH(\underline{\#G})))$$

that needs booth $PA \vdash \neg G \vee \neg TH(\underline{\#G})$ and $PA \vdash G \vee \neg(\neg TH(\underline{\#G}))$ can be proved. To prove the second one it's equivalent to prove $PA \vdash \neg G \rightarrow$

$TH(\underline{\#G})$ and so, by hypotesis $PA \vdash \neg G$, we obtain $PA \vdash TH(\underline{\#G})$ (by modus ponens) and $PA \vdash TH(\underline{\# \neg G})$ by lemma 10. Both statement can't be true since PA is coherent, otherwise we could take one proof of G, one of $\neg G$ to obtain a proof of the empty sequent.

Theorem 12 (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 13 *Let T be a theory extending PA_0, if T is consistent so T is undecidable*[1]*.*
Proof: *Let $\Theta = \{(m,n)|\ m = \#F[x_0] \text{ and } T \vdash F[\underline{n}]\}$. By contradiction T is decidable, so are Θ and $B = \{n \in \mathbb{N}|(n,n) \notin \Theta\}$. Let $G[x]$ the formula representing B (i.e. $G[n] \Leftrightarrow B$), so for any $n \in \mathbb{N}$:*

- *$n \in B$ implies $PA_0 \vdash G[\underline{n}]$ and so $T \vdash G[\underline{n}]$;*
- *$n \notin B$ implies $PA_0 \vdash \neg G[\underline{n}]$ and so $T \vdash \neg G[\underline{n}]$.*

Let $\#G[x] = g$ so $g \notin B$: if $g \in B$ so $(g,g) \notin \Theta$ and $T \nvdash G[\underline{g}]$ while $T \vdash G[\underline{g}]$ by definition of G. Conversely if $g \notin B$ implies $(g,g) \in \Theta$ and $T \vdash G[x]$ but also $T \vdash \neg G[x]$ so T is not coherent.

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

Theorem 14 (Church) *First order logic expressed in the lenguage $\mathcal{L}_0$ is undecidable.*
Proof: *Let G the conjunction of the axioms in PA_0:*

$$PA_0 \vdash F \Leftrightarrow G \to F \in T_0$$

where and $T_0 = \{F|F \text{ is a closed formula of } \mathcal{L}_0 \text{and} PA_0 \vdash F\}$.
By Theor.13, PA_0 is not decidable so T_0 will be not recursive.

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 15 (Undecidability of the halting problem) *The halting problem for Turing machine is undecidable.*
Proof: *by contradiction we suppose, by Theor.3, that exists a universal Turing machine N, which given the number M^* encoding of a Turing machine M and the number n, it computes:*

$$N(M^*,n) = \begin{cases} 1 & \text{if the computation of } M \text{ starting with } dec_M(n) \text{termines} \\ 0 & \text{else} \end{cases}$$

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

where without loosing of generality, we can suppose that $dec_M(n)$ *is a well defined configuration for* M *for every* $n \in \mathbb{N}$.

Let $\mathcal{N}$ *be the machine such that, for every Turing machine* M*, it computes* $\mathcal{N}(M) = N(M^*, M^*)$ *so:*

$$\mathcal{N}(M) = \begin{cases} 1 & \textit{if the computation of } M \textit{ starting with } dec_M(M^*) \textit{termines} \\ 0 & \textit{if the computation of } M \textit{ starting with } dec_M(M^*) \textit{ diverges} \end{cases}$$

and $\mathcal{D}$ *the machine such that:*

$$\mathcal{D}(M) = \begin{cases} \textit{termines} & \textit{if } \mathcal{N}(M) = 0 \\ \textit{diverges} & \textit{if } \mathcal{N}(M) = 1 \end{cases}$$

The absurd follow by the facts:

- *If* $\mathcal{D}(\mathcal{D})$ *diverges and* $\mathcal{N}(\mathcal{D}) = 1$*, so* $\mathcal{N}(\mathcal{D}) = \mathcal{D}(\mathcal{D})$ *termines;*
- *If* $\mathcal{D}(\mathcal{D})$ *termines and* $\mathcal{N}(\mathcal{D}) = 0$*, so* $\mathcal{N}(\mathcal{D}) = \mathcal{D}(\mathcal{D})$ *diverges.*

2.4 Some other results

Theorem 16 (Undecidability of the halting problem for 2-register machine)
There exist a 2-register machine with undecidable halting problem

Proof: *We can build a register machine* R *from a Turing machine* T *(on binary alphabet) with one* h *register containing the letter canned by the head, two register* l *and* r *which contain a number that express the content of the left and right part of the tape (read from the center to the border).*

				q $\bigtriangledown$				
$\dots$	a_0	$\dots$	a_{i-1}	a_i	a_{i+1}	$\dots$	a_n	$\dots$

$$l = a_0(2^{i-1}) + a_1(2^{i-2}) + \dots + a_{i-1}(2^0)\ ,\ h = a_i\ ,\ r = a_n(2^{n-i}) + a_{n-1}(2^{n-i-1}) + \dots + a_{i+1}(2^0)$$

Any transition of the turing machine will so be represented by a sequence of:

- *to represent the head moving left:*
  - *multiply for* 2 *the register* r *adding* 1 *if* $h = 1$*;*
  - *if* $l \equiv 1$ *mod* 2 *subtract* 1*, set* $h = 1$*;*
  - *if* $l \equiv 1$ *mod* 2 *set* $h = 0$*;*
  - *divide* l *for* 2 *;*
- *to represent the head moving left:*
  - *multiply for* 2 *the register* l *adding* 1 *if* $h = 1$*;*
  - *if* $r \equiv 1$ *mod* 2 *subtract* 1*, set* $h = 1$*;*
  - *if* $r \equiv 1$ *mod* 2 *set* $h = 0$*;*
  - *divide* r *for* 2 *;*
- *to represent the writing in the cell read by the head:*

- *set* $h = 0$*;*
- *add* 1 *if needed;*

With a proper list of instruction there will be possible to simulate the computing of any Turing machine. The halting problem for such register machine R will be equivalent to the relative halting problem for T. Using Theor. 7 the computing for R can be simulated by a 2-register machine that will have undecidable halting problem.

Theorem 17 (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.*

Proof: *Let T_S a Turing machine computing an recursively enumerable set S. Since is possible to encode its computing by a modular machine, so it exists a modular machine $\mathcal{M}od$ such that it computes S. Then $\mathcal{H}alt_{\mathcal{M}od} \simeq \mathcal{H}alt_{T_s}$ is indecidable.*

Theorem 18 (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.*

Proof: *The equivalence problem ask if, given a $z \in \mathbb{Z}$, $z \leftrightarrow^*_{\mathcal{A}} m$. Let $\mathcal{M}$ a 2-register machine and n instructin. If we pose $m = n + 1$ and if we encode every configuration (i, axy) by the integer $[i, x, y] = i + m2^x3^y$, every trasition of $\mathcal{M}$ correstond to at most two transition affine. We obtain an affine machine $\mathcal{A}_{\mathcal{M}}$ wit the following properties:*

- *if $z \rightarrow_{\mathcal{A}} z'$ so z is the encoding of a cofiguration iff z' is;*
- $(i, x, y) \rightarrow_{\mathcal{M}} (i', x', y)$ *iff* $[i, x, y] \rightarrow_{\mathcal{A}} [i', x', y']$*;*
- $(i, x, y) \leftrightarrow^*_{\mathcal{M}} (i,' x,' y')$ *iff* $[i, x, y] \rightarrow_{\mathcal{A}} [i', x', y']$.

Since $m = [0, 0, 0]$ and $z = [s_z, a_z, b_z]$ so $z \leftrightarrow^*_{\mathcal{A}} m$ iff $(s_z, a_z, b_z) \leftrightarrow^*_{\mathcal{M}} (0, 0, 0)$, i.e. the problem of equivalence will correspond to the $\mathcal{H}alt$ problem for $\mathcal{M}$ (is possible to suppose that the final state for $\mathcal{M}$ is $(0, 0, 0)$).

2.5 Undecidability of Propositional Linear Logic

The proof of undecidability of propositional linear logic[2] *is proved in [16] reducing the halting problem for a form of* and-branching *2-register machine to a decision problem for linear logic.*

For the theories of interest here, an axiom *may be any linear logic sequent in the form $\vdash C, p^{\perp}_{i_1}, \ldots, p^{\perp}_{i_n}$ with C a MALL formula. Any finite set of axiom is a* theory*. For any theory T, we say that the sequent $\vdash \Gamma$ is provable in T when we are able to derive $\vdash \Gamma$ using the standard set of linear logic proof rules in combination with axiom $t_i \in T$. We'll note an axiom rule of T with*

$$\frac{}{\vdash C^{\perp} \otimes p_a \otimes p_b \otimes \ldots \otimes p_z} T$$

[2]See Appendix B for detail.

A direct cut *is where at least one premise is an axiom in T, a* direct proof *is a proof where all cut are direct cut. When theories are added to linear logic the cut elimination theorem no longer holds due to axioms which may participate directly in cuts. However we have the following result:*

Theorem 19 *If there is a proof of $\vdash \Gamma$ in theory T, then there is a directed proof of $\vdash \Gamma$ in theory T.*
Proof: *The proof is a variant of the cut elimination theorem for linear logic. It will suffice modify the definition of degree of a derivation in order to consider direct cuts' value null.*

We define the translation $[T]$ of a theory T with axioms $t_1, \ldots, t_k$ into a multiset of pure linear logic formulas by

$$[\{t_1, \ldots, t_k\}] = ?[t_1], \ldots, ?[t_k]$$

where $[t_i]$ is defined for each axiom as follow:

$$[C, p_a^\perp, p_b^\perp, \ldots, p_z^\perp] = (C \parr p_a^\perp \parr p_b^\perp, \parr \ldots \parr p_z^\perp)^\perp = C^\perp \otimes p_a \otimes p_b \otimes \ldots \otimes p_z'.$$

Lemma 20 *For any finite set of axioms T, the sequent $\vdash \Gamma$ is provable in theory T iff $\vdash [T], \Gamma$ is provable without non-logical axioms.*
Proof:

- $\Rightarrow$ *Given some proof of $\vdash \Gamma$ in theory T, we have a linear logic proof tree where each axiom leaf $\vdash \Delta$ is in the form $\Delta = t_i$ for some $t_i \in T$ or $\Delta = p_j, p_j^\perp$. In the first case, we replace it with a proof of $\vdash [t_i], \Delta$ (provable by definition of $[\cdot]$) and then by application of dereliction rule to $\vdash ?[t_i], \Delta$. Anyway, since each formulas in $[T]$ begin with $?$, with a proper number of weakening we can obtain a proof for $\vdash [T], \Delta$ booth if $\Delta = t_i$ or $\Delta = p_j, p_j^\perp$. If we replacing every axiom leaf with a such proof, we'll obtain a new proof tree where every binary rules increase the number of occurrence of $[T]$. Since every formula in T is in the form $?[t_i]$, it will be possible, with a proper number of contraction rules, to derive $\vdash [T], \Gamma$.*

- $\Leftarrow$ *For any axiom $[t_i]$ we can may prove $!([t_i]^\perp) = !(C \parr p)a^\perp \parr \ldots \parr p_z^\perp)$.By cutting this proof against a proof of $\vdash [T], \Gamma$ we obtain $\vdash [T \backslash \{t_i\}], \Gamma$. Thus by induction on the number of axioms in T we can derive $\vdash \Gamma$ in the theory T.*

Definition 40 (And-branching 2-register machine) *An and-branching 2-register machine is a non-deterministic Minsky machine without $ZeroTest$ instruction adding the instuctions $DEC(r, q)$ which decrease the register r and do to q and the* fork *instruction representing the and-braching:*

$$FORK q_i \; q_j$$

which allows a machine to continue the computation in both states. We'll call that kind of 2-counter machine with set of instruction Q, ACM.

An istantaneous description*, or ID, for an ACM is a finite list of ordered triples $\langle q_i, a, b\rangle$ where $q_j \in Q$ and a, b are the natural numbers in the registers. Intuitively it's the list of triples representing the configurations of the parallel* *computation of the machine which terminates successfully only if all its concurrent computation fragments terminate successfully. The ACM will take an ID s to an ID s' applying one instruction on one of the triples according to the list of instruction (the instruction $DEC(r, q)$ will not be execute if $r = 0$).*

We define the accepting triple $\langle q_\perp, a, b\rangle$ *where* $q_\perp = HALT$. *An* accepting *ID is an ID where every elements are accepting triples, that is, every branch of computation has reached the accepting triple. An ACM accepts form an ID $s = \{\langle q_0, a, b\rangle\}$ iff there is some computation form S to an accepting ID.*

The reason to take ACM is that the zero-test instruction, which is the most difficult to encode in linear logic but necessary to have an halting problem undecidable, can be simulated by the more basic and-branching. Otherwise the halting problem for a register machine without zero-test become equivalent to the word problem for commutative semi-Thue system which is decidable by Theo 43.

Lemma 21 *It is undecidable if an ACM accepts from a given ID.*
Proof: *We can suppose without loosing generality to have a Minsky machine R with an unique final state $q_\perp$ with instructions INC, DEC and $ZeroTest$ (as seen in Oss. 5).*

We can so build an ACM A_R replacing the instruction $ZeroTest$. We first replace the instructions $ZeroTest(r, q_i)$ (with $r = a, b$) with the instruction $FORK(z_r, q_i)$. The two instructions for z_a and z_b will simulate the test on register a and b respectively. For z_a (and in the same way for z_b) we add two instructions: $DEC(b, z_a)$ and $FORK(q_f, q_f)$.

Every branching form a state z_r is at once another z_r or $q_\perp$. The idea is that the computation will branch in two parallel computation: one will continue like the test had succeeded, the other one will effectively verify if it is only decreasing the other register. If a_z is the value in the register a when branch of computation that simulates the test, it will reaching only configuration $\langle q_\perp, a_z, 0\rangle$, so the machine will accept only if a_z was effectively 0.

A computation of A_R will so terminate only if the corresponding computation of R will and so the decidability of halting problem for ACM is equivalent to halting problem for Minsky machine.

Let $M = (Q, \delta, a, b)$ be an ACM we define a set of formulas:

$$\{q_i, q_i^\perp | q_i \in Q\} \bigcup \{a, a^\perp, b, b^\perp\}$$

We then define the linear logic theory for the list of instruction δ as the set of axioms determined as follows:

$$\begin{array}{rcl}
q_i = INC(a, q_j) & \mapsto & \vdash q_i^\perp, (q_j \otimes a) \\
q_i = INC(b, q_j) & \mapsto & \vdash q_i^\perp, a^\perp, q_j \\
q_i = DEC(a, q_j) & \mapsto & \vdash q_i^\perp, a^\perp, q_j \\
q_i = DEC(b, q_j) & \mapsto & \vdash q_i^\perp, b^\perp, q_j \\
q_i = FORK(q_j, q_k) & \mapsto & \vdash q_i^\perp, q_j, q_k
\end{array}$$

Using the linear implication a transition like $q_i = INC(a, q_j)$ may viewed as $\vdash q_i \multimap (q_j \otimes a)$ i.e. the state q_i move to the state q_j and add 1 to a. Denoting with C^n the sequence

$$C^n = \overbrace{C, \ldots C}^{n}$$

is possible to define a translation Θ converting a configuration for an ACM into a linear logic sequent

$$\Theta(\langle q_i, x, y\rangle) = \vdash q_i^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f$$

containing an occurrence of q_f, a negative occurrence of a q_j and a number of negative occurrence of a and b equal to the value of the corresponding register. The translation of an ID s is the set of sequent translation of the configuration of s.

Lemma 22 *An ACM accepts form an ID s iff every sequent in $\Theta(s)$ is provable in the theory derived from M.*
Proof: *we'll not give a proof of the lemma but we'll show an example to understand how LL simulate the computing or the register machine.*

- $\Theta(\langle q_f, 0, 0\rangle) = \vdash q_f^{\perp}, {a^{\perp}}^{0}, {b^{\perp}}^{0}, q_f$ *is provable since*

$$\dfrac{}{\vdash q_f^{\perp}, q_f}\ I$$

  is a logical axiom.

- $\Theta(\langle q_j, x+1, y\rangle) = \vdash q_j^{\perp}, {a^{\perp}}^{x+1}, {b^{\perp}}^{y}, q_f$ *obtained by an instruction* $q_i = INC(a, q_j)$ *will be provable iff it will be* $\Theta(\langle q_i, x, y\rangle) = \vdash q_i^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f$*:*

$$\dfrac{\dfrac{}{\vdash q_i^{\perp}, (q_j \otimes a)}\ T \qquad \dfrac{\begin{array}{c}\vdots\\ \vdash q_j^{\perp}, {a^{\perp}}^{x+1}, {b^{\perp}}^{y}, q_f\end{array}}{\vdash (q_j^{\perp} \parr a^{\perp}), {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f}\ \parr}{\vdash q_i^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f}\ Cut$$

  and in a similar way for $q_i = INC(b, q_j)$;

- $\Theta(\langle q_j, x, y\rangle) = \vdash q_j^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f$ *obtained by an instruction* $q_i = DEC(a, q_j)$ *will be provable iff it will be* $\Theta(\langle q_i, x+1, y\rangle) = \vdash q_i^{\perp}, {a^{\perp}}^{x+1}, {b^{\perp}}^{y}, q_f$*:*

$$\dfrac{\dfrac{}{\vdash q_i^{\perp}, q_j, a^{\perp}}\ T \qquad \begin{array}{c}\vdots\\ \vdash q_j^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f\end{array}}{\vdash q_i^{\perp}, {a^{\perp}}^{x+1}, {b^{\perp}}^{y}, q_f}\ Cut$$

  and in a similar way for $q_i = DEC(b, q_j)$;

- $\Theta(\langle q_j, x, y\rangle) = \vdash q_j^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f$ *and* $\Theta(\langle q_k, x, y\rangle) = \vdash q_k^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f$ *obtained by an instruction* $q_i = FORK(q_j, q_k)$ *will be provable iff it will be* $\Theta(\langle q_i, x, y\rangle) = \vdash q_i^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f$*:*

$$\dfrac{\dfrac{}{\vdash q_i^{\perp}, q_j \oplus q_k}\,T \qquad \dfrac{\begin{matrix}\vdots\\ \vdash q_j^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f\end{matrix} \qquad \begin{matrix}\vdots\\ \vdash q_k^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f\end{matrix}}{\vdash (q_j^{\perp} \mathbin{\&} q_k^{\perp}), {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f}\,\&}{\vdash q_i^{\perp}, {a^{\perp}}^{x}, {b^{\perp}}^{y}, q_f}\,Cut$$

So an ID is accepted by an ACM if it represents the branching of an halting computation corresponding to a direct proof of $\vdash q_f^{\perp}, q_f$.

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 23 (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 8 *The presentation $\langle \Sigma_F | \mathcal{R}_F \rangle$ is not convergent.*

Proof: *We just need to observe all the critique peaks:*

- *if the critique pick it's a word of the alphabet of G, it's soluble because it's in the standard presentation of G*

[1]see 1.2 pag. 4

- *if the critique pick it's a word of the alphabet of $\langle b, \bar{b}\rangle$, it is solvable:*

$$\begin{tikzcd}[column sep=small]
& b\bar{b}b \arrow[dl] \arrow[dr] & \\
b \arrow[rr, phantom, "="] & & b
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
& \bar{b}b\bar{b} \arrow[dl] \arrow[dr] & \\
\bar{b} \arrow[rr, phantom, "="] & & \bar{b}
\end{tikzcd}$$

- *if the critique peak contain only the letters of Σ_b and a_h with $h, k \in H$, it's solvable:*

$$\begin{tikzcd}
b\bar{b}a_h \arrow[ddd] \arrow[dr] & \\
& ba_h\bar{b} \arrow[d] \\
& a_hb\bar{b} \arrow[dl] \\
a_h &
\end{tikzcd}
\quad
\begin{tikzcd}
\bar{b}ba_h \arrow[ddd] \arrow[dr] & \\
& \bar{b}a_hb \arrow[d] \\
& a_h\bar{b}b \arrow[dl] \\
a_h &
\end{tikzcd}
\quad
\begin{tikzcd}
ba_ha_k \arrow[ddd] \arrow[dr] & \\
& a_hba_k \arrow[d] \\
& ba_ha_k \arrow[dl] \\
ba_{hk} &
\end{tikzcd}
\quad
\begin{tikzcd}
\bar{b}a_ha_k \arrow[ddd] \arrow[dr] & \\
& a_h\bar{b}a_k \arrow[d] \\
& \bar{b}a_ha_k \arrow[dl] \\
\bar{b}a_{hk} &
\end{tikzcd}$$

- *all the non-solvable peak are all in the form ($\beta \in \Sigma_b$, $h \in H$, $x \in G\backslash H$) :*

$$\begin{tikzcd}[column sep=small]
& \beta a_h a_x \arrow[dl] \arrow[dr] & \\
\beta a_{hx} \arrow[rr, phantom, "\neq"] & & a_h\beta a_x
\end{tikzcd}$$

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

Using the Lemma1 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 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.1, is possible to derivate the following set $\mathcal{R}_{\perp}$ of relations:

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

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

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

Proposition 8 *The presentation $(\Sigma_G \cup \Sigma_{\perp} | \mathcal{R}_G \cup \mathcal{R}_{\perp})$ is convergent.*

Proof: *Like in 8, a critique peak of the alphabet Σ_G or $\{b_1, b'_1\}$ is solvable. The others critique peak are all in the form $\beta_v a_x a_y$ or $b_1 b'_v a_x$ or $b'_1 b_v a_x$. These three kind of critique peak are solvable:*

[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 $\bar{b}a_v$

$$\begin{tikzcd}
& \beta_v a_x a_y \arrow[dl] \arrow[dr, "{vx = kw', k \in H, w' \in H^\perp}"] & \\
\beta_v a_{xy} \arrow[ddd, "{v(xy) = hw, h \in H, w \in H^\perp}"'] & & a_k \beta_{w'} a_y \arrow[d, "{w'y = k'w'', k' \in H, w'' \in H^\perp}"] \\
& & a_k a_{k'} \beta_{w''} \arrow[dd] \\
& & \\
a_h \beta_w & = & a_{kk'} \beta_{w''}
\end{tikzcd}$$

because $hw = v(xy) = (vx)y = (kw')y = k(w'y) = k(k'w'') = (kk')w''$ and by the lemma 1 $w = w''$ and $h = kk'$.

$$\begin{tikzcd}
& b_1 b'_v a_x \arrow[dddl] \arrow[dr, "{vx = hw, h \in H, w \in H^\perp}"] & \\
& & b_1 a_h b'_w \arrow[d] \\
& & a_h b_1 b'_w \arrow[d] \\
a_v a_x \arrow[d] & & a_h a_w \arrow[d] \\
a_{vx} & = & a_{hw}
\end{tikzcd}$$

the same for the pick $b'_1 b_v a_x$ changing b_1 with b'_1 and b'_v with b_v.

Remark 9 *Every reduced words of this presentation of F' are in the form $\alpha\beta_1 \dots \beta_n$ with $\alpha \in \Sigma_G \cup \{1\}$, $n \geqslant 0$ and $\beta_i \in \Sigma_\perp$ ($n \neq 1 \Rightarrow \forall i,\ \beta_i \neq b_1$ and $\beta_i \neq b'$).*

Proposition 9 *$F' = \langle \Sigma_{F'} = \Sigma_G \cup \Sigma_\perp | \mathcal{R}_{F'} = \mathcal{R}_G \cup \mathcal{R}_\perp \rangle \simeq F$.*

Proof: *By Prop.2 , it suffices to show that exists an iso-translation from F to F'. Let $\phi : \Sigma_F \to \Sigma'_F$ such that $\phi(a_x) = a_x$, $\phi(b) = b_1$ and $\phi(\bar{b}) = b'_1$ we can define $\bar{\phi}$ and so:*

- $\forall \mathfrak{r} \in \mathcal{R}, \bar{\phi}(\mathfrak{r}) \in \leftrightarrow^*_{\mathcal{R}'}$;
- *exists a control function ψ given by $\psi(a_x) = a_x$, $\psi(b_v) = ba_v$ and $\psi(b'_v) = \bar{b}a_v$;*
- *since ψ is always defined $\bar{\phi}(\psi(v')) \leftrightarrow^*_{\mathcal{R}'} v'$.*

Since$\bar{\phi}$ is an iso-translation so $F \simeq F'$.

3.1.3 HNN extension theorem proof Part III: Concluding

It's easy to prove by prop.2 that $F' \geqslant G$ and $F' \geqslant \langle b \rangle$ because the functions $id_G : \Sigma_G \to \Sigma'^*_F$ and $id_b : \{b, b' = \bar{b}\} \to \Sigma^*_F$ are embedding translation. 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 $\psi(a_h b_w) = a_h ba_w$ 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 24 *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.*

Proof: *It just needs to change a little bit the construction of F used in the demonstration of Th.23. Let $u_1, \dots u_n \in \Sigma_G$ such that $H = \langle u_1, \dots, u_n \rangle$ since $\forall h \in H$, $h = u_{i_1} \dots u_{i_m}$, $\exists m > 0$ and $i_j \in \{1, \dots, n\}$. F will be presented by $\langle \Sigma_G \cup \{b, \bar{b}\} | \mathcal{R}_G \cup R_{gen} \rangle$ where $R_{gen} = \{b\bar{b} \to 1, \bar{b}b \to 1, bu_1 \to u_1 b, \dots, bu_n \to u_n b\}$. By the transitive and operation-compatible closure of R_{gen}, $\forall h \in H$ the relation $(a_h b, ba_h) \in \leftrightarrow^*_{\mathcal{R}_{gen}}$ so $\leftrightarrow^*_{\mathcal{R}_F} \subseteq \leftrightarrow^*_{\mathcal{R}_G \cup \mathcal{R}_{gen}}$ where $\mathcal{R}_F := \mathcal{R}_G \cup \{(hb, bh) | h \in H\}$. Moreover every u_i are elements of H so $R_{gen} \subseteq \mathcal{R}_F$ and $\leftrightarrow^*_{\mathcal{R}_G \cup \mathcal{R}_{gen}} \subseteq \leftrightarrow^*_{\mathcal{R}_F}$.*

Theorem 25 (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 .*

Proof: *Let $F = \frac{G * \langle b \rangle}{\leftrightarrow^*_C}$ where $\leftrightarrow^*_C$ is the smallest equivalence relation containing the set $C = \{(bh, \phi(h)b) | h \in H\}$. Fixed $H^\perp, H'^\perp$ transversal sets respectively of cosets of H and H' ($1 \in H^\perp$ and $1 \in H'^\perp$) is possible to give the following convergent presentation of $F = (\Sigma_\phi | \mathcal{R}_\phi)$ built in the similar way of 3.1.2 ($b_u = ba_u$ and $b'_v = \bar{b}a_v$):*

$$\Sigma_\phi = \{a_x\}_{x \in G} \cup \{b_u\}_{u \in H^\perp} \cup \{b'_v\}_{v \in H'^\perp}$$

and the following rewriting rules $\mathcal{R}_\phi$:

$$a_x a_y \to a_{xy} \qquad a_1 \to 1 \qquad b_1 b'_v \to a_v \qquad b'_1 b_u = a_u$$

$$b_v a_x \to a_{\phi(h)} b_w \qquad \exists! h \in H, v, w \in H^\perp \text{ such that } vx = hw$$

$$b'_v a_x \to a_{\phi(h')} b'_w \qquad \exists! h' \in H', v, w \in H^\perp \text{ such that } vx = h'w$$

Like in Th.23 $(\Sigma_\phi | \mathcal{R}_\phi)$ is a convergent presentation and F is an extension of G and $\langle b \rangle$.
1) b represents ϕ since $\forall u \in H$, $b_1 a_u b'_1 = a_{\phi(u)}$.
2) For every $K < G$ is possible to choose the elements of $H^\perp$ and $H'^\perp$ such that for every $k \in K$, $k = hv$ where $h \in K \cap H$ and $v \in K \cap H^\perp$, under that conditions if K is ϕ-invariant if a word is written in the alphabet

$$\Sigma_\phi|_K = \{a_k\}_{k \in K} \cup \{b_u\}_{u \in H^\perp \cap K} \cup \{b'_v\}_{v \in H'^\perp \cap K}$$

so it is a normal form since every K is a subgroup. That means $\langle K, b \rangle_F \cap G \subseteq K$ and so the equality while $K \subseteq \langle K, b \rangle_F \cap G$.
3) Follow from Cor.24.

Theorem 26 (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*

Proof: *Induction on the number of local isomorphism n using Th.25*

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.

4.1.1 Novikow group $\mathfrak{A}_{p_1,p_2}$

Let K a Post system[1] $[\Sigma_a;\mathcal{R}]$ *on the alphabet* $\Sigma_a = \{a_1,\ldots,a_n\}$ *and* $\mathcal{R} = \{(A_i,B_i), 1 \leqslant i \leqslant \lambda\}$*,* A_i, B_i *nonempty, is possible to build the Novikow group* $\mathfrak{A}_{p_1,p_2}$ *associated with K on the alphabet Σ consisting of:*

$$a_1,\ldots,a_n,q_1,\ldots,q_\lambda,r_1,\ldots,r_\lambda,l_1,\ldots,l_\lambda$$

one of his copy, namely:

$$a_1^+,\ldots,a_n^+,q_1^+,\ldots,q_\lambda^+,r_1^+,\ldots,r_\lambda^+,l_1^+,\ldots,l_\lambda^+$$

and two supporting letters p_1, p_2*. It's defined by the following relations:*

1. $q_i a = a q_i q_i$, $q_i^+ q_i^+ a^+ = a^+ q_i^+$;
2. $r_i r_i a = a r_i$, $r_i^+ a^+ = a^+ r_i^+ r_i^+$;
3. $a l_i = l_i a$, $a^+ l_i^+ = l_i^+ a^+$;
4. $q_i^+ l_i^+ p_1 l_i q_i = A_i^+ p_1 A_i$;
5. $r_i^+ p_1 r_i = p_1$;
6. $r_i l_i p_2 l_i^+ r_i^+ = B_i p_2 B_i^+$;

[1] see. Appendix A

7. $q_i p_2 q_i^+ = p_2$;

for $1 \leqslant i \leqslant \lambda, a \in \Sigma_a$ *and* $(a_{s_1}, \ldots, a_{s_k})^+ = a_{s_1}^+, \ldots, a_{s_k}^+$.

Proposition 10 (Novikow property) *The words* $p_1 X p_2 X^+$ *and* $p_1 Y p_2 Y^+$ *are conjugate in the group* $\mathfrak{A}_{p_1 p_2}$ *iff* $X \sim_K Y$ *in the associated* Post system[2] K *where* $X, Y \in \Sigma_a$.

4.1.2 Novikow group $\mathfrak{A}_p$

Let $\Sigma_a = \{a_1, \ldots, a_n\}$ *and* (A_i, B_i) *pairs of nonempty* Σ_a*-word for* $1 \leqslant i \leqslant m$.

$$A_{d\mu l\rho} = \langle \Sigma_a \cup \{\rho, \tilde{\rho}, \mu_{1i}, \tilde{\mu}_{1i}, \mu_{2i}, \tilde{\mu}_{2i}, l_{ai}, d_i\}_{1 \leqslant i \leqslant m} | \mathcal{R} \rangle$$

where $\mathcal{R}$ *is the set of the following relation:*

1. $\rho_i a = a\rho_i^2$, $\quad \tilde{\rho}_i a = a\tilde{\rho}_i^2$;
2. $bl_{ai} = l_{ai} b$;
3. $a\mu_{1i} lai = \mu_{1i} a, \quad a\tilde{\mu}_{1i} lai = \tilde{\mu}_{1i} a$;
4. $al_{ai}\mu_{2i} = \mu_{2i} a, \quad al_{ai}\tilde{\mu}_{2i} = \tilde{\mu}_{2i} a$;
5. $\tilde{\mu}_{1i}\tilde{\rho}_i d_i \tilde{\mu}_{2i} = \tilde{\mu}_{1i}\rho_i d_i \tilde{\mu}_{2i} A_i^{-1} B_i$;
6. $ad_i = d_i a$;

for $1 \leqslant i \leqslant \lambda$ *and* $a, b \in \Sigma_a$.

$$\mathfrak{A}_p = \frac{A_{d\mu l\rho} * A_{d\mu l\rho}^+ * p}{\leftrightarrow_{\mathcal{R}_p}^*}$$

where $A_{d\mu l\rho}^+$ *is an antiisomorphic copy of* $A_{d\mu l\rho}$ *given by the antiisomorphism* [3] $x \to x^+$ *and* $\mathcal{R}_p = \{EpE^+ \to p\}$ *where* $E \in A_{d\mu l\rho}$.

4.1.3 Boone group

Let $T = (\Sigma_T = \{s_d, q_e\}_{d\in D, e\in E} | \mathcal{R}_T = \{A_i \to B_i, \}_{1\leqslant i \leqslant N})$ *with* $q_1 = q$*, a monoid with* A_i, B_i special words *in the alphabet* Σ_a *(i.e. word in the form* $sq_e s'$ *with* s, s' *words of the alphabet* $\{s_d\}$*), the Boone group* $G(T, q)$ *with corresponding monoid* T *is given by the alphabet:*

$$\Sigma = \{s_d, q_e, x, y, l_i, r_i, k, t\}_{d\in D, e\in E,\ 1\leqslant i\leqslant N}$$

and the following relations:

1. $y^2 s_d = s_d y, \quad xs_d = s_d x^2$;
2. $s_d l_i = yl_i ys_d, \quad s_d x r_i x = r_i s_d$;
3. $l_i B_i r_i = A_i$;

[2]see App.A

[3]an antiisomorphisme $\phi : G \to G'$ is a map such that $\phi(1_G) = 1_{G'}$ and $\phi(xy) = \phi(y)\phi(x)$

4. $l_i t = t l_i$, $yt = ty$;

5. $r_i k = k r_i$, $xk = kx$;

6. $q^{-1}tqk = kq^{-1}tq$;

Proposition 11 (Boone property) *Let S, S' special words of Σ_a, than $S \leftrightarrow^*_{R_T} S'$ iff $\exists V(l_i, y), W(r_i, x)$ such that $S = V(l_i, y)S'W(r_i, x)$ in $G(T, q)$.*

4.1.4 Borisov group

Let $\Sigma_a = \{s_j\}_{1 \leqslant j \leqslant n}$ and $\mathcal{R}_\Pi = \{(F_i, G_i), 1 \leqslant i \leqslant m\}$ a set of pairs of nonempty words of Σ_a and P a fixed arbitrary word of Σ_a. The Borisov group $G(\Pi, P)$ can be presented by the alphabet:

$$\Sigma = \Sigma_a \cup \{d, e, c, t, k\}$$

and the following relation:

1. $d^{m+1}s = sd$, $es = se^{m+1}$;

2. $sc = cs$;

3. $d^i F_i e^i c = c d^i G_i e^i$;

4. $ct = tc$, $dt = td$;

5. $ck = kc$, $ek = ke$;

6. $P^{-1}tPk = kP^{-1}tP$;

for every $1 \leqslant i \leqslant m$, $s \in \Sigma_a$. Let $\Pi = (\Sigma_a|\mathcal{R}_\Pi)$ the monoid associated with $G(\Pi, P)$.

Proposition 12 (Borisov property) *Let Q be a Σ_a-word then $Q = P$ in the associated monoid iff $Q^{-1}tQk = kQ^{-1}tQ$ in $G(\Pi, P)$.*

4.1.5 Aandrea group

In [6] its presentation is linked with Aandrea's modular machine instruction set [1]. It's presented by an integer $m > 0$ and a set of triples of integer $M = \{(s_i, a_i, b_i)\}_{i \in I} \cup \{(s_j, a_j, b_j)\}_{j \in J}$ where $0 \leqslant a_k, b_k < m$ and $0 \leqslant c_k < m^2$ for every $k \in I \cup J$.

$$G(M) = (r_i, l_j, x, y, t, r, , k; i \in I, j \in J|\mathcal{R}_M)$$

where, denoting $t(\alpha, \beta) = x^{-\alpha}y^{-\beta}tx^\alpha y^\beta$ for $\alpha, \beta \geqslant 0$, the relation of $\mathcal{R}_M$ are:

1. $xy = yx$;

2. $x^m r_i = r_i x^{m^2}$, $y^m r_i = r_i y$;

3. $t(a_i, b_i) r_i = r_i t(s_i, 0)$;

4. $x^m l_j = l_j x$, $y^m l_j = l_j y^{m^2}$;

5. $t(a_j b_j) l_j = l_j t(0, s_j)$;

where $i \in I, j \in J$.

Proposition 13 *For every modular machine $\mathcal{M}od$, it exists an Aandrea group $G(M_{\mathcal{M}od})$ associated.*

4.1.6 Valiev group

Differentrly form the previous groups, the Valiev group [22] does not depend on a monoid, Post system or a Turing or Modular machine, it can interpretate any recursively enumerable set of natural number. It'll be presented by the alphabet:

$$\Sigma = \{a_i, b_i, c_i, t_i, i_{ijk}, d\}_{0 \leqslant i \leqslant m,\ 0<k<i,j<m}$$

and the relations:

1. $t_0^{-1} b_0 t_0 = a_0^{-1} b_0 a_0$;
2. $t_i^{-1} b_i t_i = a_i b_i c_i \qquad (1 \leqslant i \leqslant m)$;
3. $t_i a_j = a_j t_i\ , \qquad t_i c_j = c_j t_i\ , \qquad (0 \leqslant i, j \leqslant m)$;
4. $a_m d = d a_m^2\ , \qquad c_m d = d c_m^2 \qquad b_{m-1} d a_{m-1} b_{m-1} c_{m-1}$;
5. $a_i d = d a_i (i \neq m)\ , \qquad b d_i = d_i b (i \neq m-1) \qquad c_i d = d c_i (i \neq m)$;
6. $b_i t_{ijk} = t_{ijk} a_1 b_i c_i\ , \qquad c_i t_{ijk} = t_{ijk} t_k c_j\ , \qquad t_{ijk} t_k = t_k t_{ijk},$
   $t_{ijk} a_s = a_s t_{ijk} (s \neq i),\ t_{ijk} b_s = b_s t_{ijk} (s \neq i),\ t_{ijk} c_s = c_s t_{ijk} (s \neq j)\ .$

4.2 Group with standard basis

Definition 41 (Group with stable letters) *Let $\hat{G} = \langle \hat{\Sigma} | \hat{\mathcal{R}} \rangle$ be a group, the group with a system of* stable letters $\{p\}$ *and* base group $\hat{G}$ *is defined by:*

$$G = \langle \Sigma = \hat{\Sigma} \cup \{p\} | \mathcal{R} = \hat{\mathcal{R}} \cup \mathcal{R}_p = \{A_i p \to pB_i\}_{i \in I} \rangle$$

where $p \notin \Sigma$ and $\forall i \in I$ $A_i, B_i \in \hat{\Sigma}^*$. A pair of corresponding *or* twin *words will be in the form:*

$$\mathcal{A}_p = \mathcal{A}_{i_1}^{\pm 1}, \ldots, \mathcal{A}_{i_k}^{\pm} \qquad \mathfrak{B}_p = \mathfrak{B}_{i_1}^{\pm 1}, \ldots, \mathfrak{B}_{i_k}^{\pm}$$

thus, for $\epsilon = \pm 1$, the equality $\mathcal{A}_{p^\epsilon} p^\epsilon = p^\epsilon \mathfrak{B}_{p^\epsilon}$ where $\mathcal{A}_p^{-1} = \mathfrak{B}_p$ and $\mathfrak{B}_{p^{-1}} = \mathcal{A}_p$.

The extended system of relation *of the group G is the system of rule $\mathcal{R}_p \cup \mathcal{R}_p^{-1}$ where $\mathcal{R}_p^{-1} = \{B_i^{-1} p^{-1} \to p^{-1} A_i^{-1}$ such that $A_i p \to pB_i \in \mathcal{R}_p\}_{i \in I}$. In that system it's possible to define the* individuality of a letter*: since every transformation is in the form:*

$$uwv \to uw'u \text{ with } (w \equiv A_i p,\ w' \equiv pB_i) \text{ or } (w \equiv B_i^{-1} p,\ w' \equiv p^{-1} A_i^{-1}),\ u, v \in \hat{\Sigma}^*$$

the individuality of the letter p and all the letters in u and v will be preserved.

Definition 42 (Regular system) *A system of stable letters is called* regular *if $\mathcal{A}_{p^\epsilon} \leftrightarrow^*_{\hat{\mathcal{R}}} 1 \Leftrightarrow \mathfrak{B}_{p^\epsilon} \leftrightarrow^*_{\hat{\mathcal{R}}} 1$ for any corresponding words $\mathcal{A}_p, \mathfrak{B}_p$.*

Proposition 14 *If $\{p\}$ is a regular system for $\hat{G}$, so G is an HNN-extension of $\hat{G}$.*

Proof: *See Cor.30.*

Definition 43 (Insertion/cancellation) *An* insertion *is a transformation in the form $1 \to pp^{-1}$ or $1 \to p^{-1}p$. Its inverse it's called* cancellation.

Lemma 27 *Let $Wp^\epsilon U \to W_1 p^\epsilon U_1 \to \ldots \to W_n p^\epsilon U_n$ be a chain of extended transformations, where the individuality of p^ϵ is preserved. Then there exists twin words $\mathcal{A}_{p^\epsilon}$ and $\mathfrak{B}_{p^\epsilon}$ such that:*

$$W = W_n \mathcal{A}_{p^\epsilon} \qquad U = \mathfrak{B}_{p^\epsilon}^{-1} U_n \ .$$

If there are insertion of stable letters in the chain then the words W and U can be respectively transformed into the words $W_n \mathcal{A}_{p^\epsilon}$ and $\mathfrak{B}_{p^\epsilon}^{-1} U_n$ without applying such transformations.

Proof: *Proved by induction on the length n of the chain. For $n = 0$ is trivial. If a transformation of the chain does not apply on p^ϵ than the lemma is clear, else it is in the form $W_i \mathcal{A}_l p U_i \to W_i p \mathfrak{B}_l U_i + 1$ or $W_i \mathfrak{B}_l p^{-1} U_i \to W_i p^{-1} \mathcal{A}_l U_i$ so $W_{i+1} = W_i \mathcal{A}_{ip^\epsilon}$ and $U_{i+1} = \mathfrak{B}_{ip^\epsilon}^{-1} U_i$. Moreover in passing from the words W_i, U_i to W_{i+1}, U_{i+1} there is not insertion of stable letters.*

Lemma 28 (The Novikov lemma) *Let $\{p\}$ be a regular system of stable letters and W a word in G satisfying $W = 1$. Than W can be rewrited in 1 by a chain of extended transformation, each of them is not an insertion of stable letters.*

Proof: *Let consider a step of a chain of an extended transformation $W \to \ldots \to 1$ in which there is an insertion of the letter p:*

$$W \to \ldots \to W_{i-1} = VV' \to W_i = Vp^{\epsilon}p^{-\epsilon}V' \to \ldots \to 1$$

since the letters p^{ϵ} and $p^{-\epsilon}$ should be cancelled during the transformation, there are two cases:

- *the cancellation involves only the this two letters:*

  $$W \to \ldots \to W_i = V_1p^{\epsilon}p^{-\epsilon}V_1' \to \ldots \to V_kp^{\epsilon}p^{-\epsilon}V_k' \to V_kV_k' = W_k \to \ldots \to 1$$

  so by the Lemma 27 there exist twin words $\mathcal{A}_{1p^{\epsilon}}, \mathcal{A}_{2p^{\epsilon}}\mathfrak{B}_{1p^{\epsilon}}, \mathfrak{B}_{2p^{\epsilon}}$ such that the words $V_1, 1, V_1'$ can be transformed into the words $V_k\mathcal{A}_{1p^{\epsilon}}, \mathfrak{B}_{1p^{\epsilon}}^{-1}\mathfrak{B}_{2p^{\epsilon}}$ and $\mathcal{A}_{2p^{\epsilon}}^{-1}V_k'$ without insertion of stable letters. Since $\{p\}$ is regular in G holds $\mathfrak{B}_{1p^{\epsilon}}^{-1}\mathfrak{B}_{2p^{\epsilon}} = 1$ iff $\mathcal{A}_{1p^{\epsilon}}\mathcal{A}_{2p^{\epsilon}}^{-1} = 1$. So W_i can be transformed in W_k without insertion of stable letters, then is possible to obtain the same transformation eliminating this insertion of stable letters;

- *else the chain is in the form:*

  $$W \to \ldots \to W_i = V_1p^{\epsilon}V_1'p^{-\epsilon}p^{\epsilon}V_1'' \to \ldots$$

  $$\ldots \to V_kp^{\epsilon}p^{-\epsilon}V_k'p^{\epsilon}V_k'' \to V_kV_k'p^{\epsilon}V_k'' = W_k \to \ldots \to 1$$

  by lemma 27 there exists pairs of twin words $\mathcal{A}_{ip^{\epsilon}}, \mathfrak{B}_{ip^{\epsilon}}, i = 1, 2, 3$ such that the words $V_1, V_1', 1$ and V_1'' can be transformed respectively in $V_k\mathcal{A}_{1p^{\epsilon}}$, $\mathfrak{B}_{1p^{\epsilon}}^{-1}\mathcal{A}_{2p^{-\epsilon}}$, $\mathfrak{B}_{2p^{-\epsilon}}^{-1}V_k'\mathcal{A}_{3p^{\epsilon}}$ and $\mathfrak{B}_{3p^{\epsilon}}^{-1}V_k''$, hence the word W_i can be transformed into

  $$V_k\mathcal{A}_{1p^{\epsilon}}p^{\epsilon}\mathfrak{B}_{1p^{\epsilon}}^{-1}\mathcal{A}_{2p^{-\epsilon}}\mathfrak{B}_{3p^{\epsilon}}^{-1}V_k''$$

  and applying the transformations in the extended system W_i become

  $$V_k\mathcal{A}_{1p^{\epsilon}}\mathcal{A}_{1p^{\epsilon}}^{-1}\mathcal{A}_{2p^{\epsilon}}\mathcal{A}_{3p^{\epsilon}}^{-1}p^{\epsilon}V_k''$$

  which can be transformed in

  $$V_k\mathcal{A}_{2p^{\epsilon}}\mathcal{A}_{3p^{\epsilon}}^{-1}p^{\epsilon}V_k'' = V_k\mathfrak{B}_{2p^{-\epsilon}}\mathcal{A}_{3p^{\epsilon}}^{-1}p^{\epsilon}V_k''.$$

  By the insertion of $1 = \mathfrak{B}_{2p^{-\epsilon}}^{-1}V_k'\mathcal{A}_{3p^{\epsilon}}$ (which doesn't contain stable letters), we have:

  $$V_k\mathfrak{B}_{2p^{-\epsilon}}\mathcal{A}_{3p^{\epsilon}}^{-1}p^{\epsilon}V_k'' \to V_k\mathfrak{B}_{2p^{-\epsilon}}1\mathcal{A}_{3p^{\epsilon}}^{-1}p^{\epsilon}V_k'' \to$$

  $$\to V_k\mathfrak{B}_{2p^{-\epsilon}}\mathfrak{B}_{2p^{-\epsilon}}^{-1}V_k'\mathcal{A}_{3p^{\epsilon}}\mathcal{A}_{3p^{\epsilon}}^{-1}p^{\epsilon}V_k'' \to^2 V_kV_k'p^{\epsilon}V_k'' = W_k.$$

  This permits to decrease the number of insertions in the chain.

The lemma follows by induction on the number of insertion in the chain.

Lemma 29 (The Britton's lemma) *Let $\{p\}$ be a regular system of stable letters for the group G over $\hat{G}$ and W a word in G such that $W = 1$ in G. Than W is a word in $\hat{G}$ and $W =_{\hat{G}} 1$ or W includes the subword $p^{-\epsilon}Ap^{\epsilon}$ where $A \in \hat{G}$ and $A =_{\hat{G}} \mathcal{A}_{p^{\epsilon}}$.*

Proof: *By Novikov's lemma, the word W can be transformed in 1 without insertion of stable letters, so if the chain*

$$W \to W_1 \to \ldots \to W_n = 1$$

contain no stable letters then $W \in \hat{G}$ and $W =_{\hat{G}} 1$. If W contains the letter p, then it should be cancelled during the transformation. Considering the first cancellation of a stable letters occurring in the chain

$$W = Vp^{-\epsilon}V'p^{\epsilon}V'' \to \ldots \to W_k = V_k p^{-\epsilon}p^{\epsilon}V'_k \to V_k V'_k = W_{k+1}$$

where V' does not contain the stable letters. By lemma 27 there exists a pair of twin words $\mathcal{A}_{ip^{\epsilon}}, \mathfrak{B}_{ip^{\epsilon}}, i = 1, 2$ such that the words V, V', V'' can be transformed into the words $V_k\mathcal{A}_{1p^{-\epsilon}}, \mathfrak{B}^{-1}_{1p^{-\epsilon}}\mathcal{A}_{2p^{\epsilon}}$ and $\mathfrak{B}^{-1}_{2p^{\epsilon}}V'_k$ without insertion of stable letters. Hence $V' \in \hat{G}$ since $V' = \mathfrak{B}^{-1}_{1p^{-\epsilon}}\mathcal{A}_{2p^{\epsilon}} = \mathcal{A}{-1}_{1p^{\epsilon}}\mathcal{A}_{2p^{\epsilon}} = \mathcal{A}_{p^{\epsilon}}$.

Corollary 30 *If $\{p\}$ is a regular system of stable letters of the group G over $\hat{G}$ than $\hat{G} < G$.*

Definition 44 *A word W of a group with stable letters $\{p\}$ is called p-reducible if W includes a subword in the form $p^{-\epsilon}Ap^{\epsilon}$ where $A \in \hat{G}$ and $A =_{\hat{G}} A_{p^{\epsilon}}$.*

With this definition is possible to reformulate the Britton's lemma: if $W =_G 1$ and W contains stable letters, so for some stable letters W is p-reducible.

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.*

4.2.1 The definition of groups with standard normal form

Let's consider a sequence of HNN-extension $G_0, G_i, \ldots, G_n$ where G_0 is a free group and the group G_{i+1} is obtained adjoining to the group G_i letters $\{p\}$ and defining relation

$$A_l p = pB_l$$

where $p \in \{p\}$ it's called letter of weight $i+1$ *and $A_l, B_l \in G_i$ contain exactly one letter of the highest weight. So in the group G_{i+1} an arbitrary relation can be represented in the form*

$$A'xA''p = pB'yB''$$

where x and y are the letters of highest weight (if the power of these letters are different from ± 1 will be considered its first or last occurrence). For every relation will be associated four types of prohibited words*:*

$$x\mathfrak{B}_x A''p \qquad x^{-1}\mathfrak{B}_{x^{-1}}A'^{-1}p \qquad y\mathfrak{B}_y B''p^{-1} \qquad y^{-1}\mathfrak{B}_{y^{-1}}B'^{-1}p^{-1}$$

Is so possible to define by induction on i the notion of canonical word*: every reduced word of G_0 are in canonical form, an irreducible word*

$$U = U_1 p^{\epsilon_1} U_2 p^{\epsilon_2} \ldots U_k p^{\epsilon_k} U_{k+1}$$

in the group G_{i+1} where $U_j \in \hat{G}$ and p_j are letters of weight $i+1$ $k \geqslant 0$ is canonical if, for every j:

- *U_j are canonical words in the group G_i;*
- *U doesn't include subword of an any prohibited types in G_{i+1}.*

Is so possible to reduce a word $U \doteq U_1 p^\epsilon U_2 \dots U_{n-1} p^\epsilon U_n$ in canonical $C(U)$ by the following algorithmic process:

1. *reduce every word U_j to canonical form in the group G_i;*
2. *perform all possible cancellation of letters of weight $i+1$;*
3. *eliminate the first occurrence (from the right) of a prohibited word following the following role[4]:*

$$x\mathfrak{B}_x A'' p \to \mathcal{A}_x A'^{-1} p B \qquad x^{-1}\mathfrak{B}_{x^{-1}} A'^{-1} p \to \mathfrak{B}_x A'' p B^{-1}$$

$$y\mathfrak{B}_y B'' p^{-1} \to \mathcal{A}_y^{-1} B'^{-1} p^{-1} A \qquad y^{-1}\mathfrak{B}_{y^{-1}} B'^{-1} p^{-1} \to \mathfrak{B}_y B'' p^{-1} A^{-1}$$

   where $\mathcal{A}_z$ and B_z (with $z = x$ or y) are twin words;

4. *return to step 1 .*

Definition 45 *The group G_{i+1} is called group with* standard normal form *or group with* standard basis *if every word U can be reduced to canonical form $C(U)$ in a finite number of steps. If that condition it's satisfy for every i the group G is a group with standard normal form.*

Lemma 31 *Let G_i a group with standard normal form then the canonical for of an arbitrary word of the group G_{i+1} is unique iff the following condition are met:*

- *p is a system of stable letters;*
- *If the word Up^ϵ and Vp^ϵ are canonical $U, V \in G_i$, p letter of weight $i+1$ and $U = V\mathcal{A}_{p^\epsilon}$ then the equality $\mathcal{A}_{p^\epsilon} =_{G_i} 1$ holds.*

Lemma 32 *Let G_i be a group with standard normal form and $\{p\}$ a regular system of stable letters. Suppose that any word $\mathcal{A}_{p^\epsilon} \neq_{G_i} 1$ with the letter p of weight $i+1$ is representable as:*

$$\mathcal{A}_{p^\epsilon} =_{G_i} V_1 x_1 V_2 x_2 V_3$$

where x_1, x_2 are letters of highest weight and the word is x-irriductible for every letter x of higher weight. If an arbitrary word of the form

$$x_2 C(\mathfrak{B} x_2 V_3) p^\epsilon \qquad , \qquad x_1^{-1} C(\mathfrak{B} x_1^{-1} V_1^{-1}) p^\epsilon$$

is prohibited or includes a prohibited subword (with respect to the letter p) then the second condition of Lemma 31 are satisfied.

[4]Every of these role derive by the relation $A'\mathcal{A}_x x \mathfrak{B}_x A'' p = pB'\mathcal{A}_y y \mathfrak{B}_y B''$, where $B \doteq B'yB'' and A \doteq A'xA''$

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

To introduce the Boone group $G(T,q)$ is needed to extend the concept of stable letters to system with more than one letter. A set $P = \{p_m\}$ is a system of stable letters of a group G over $\hat{G}$ if the group G can be presented by

$$G = \langle \Sigma_{\hat{G}} \cup \{p_m\} | \mathcal{R}_{\hat{G}} \cup \{A_i p_{m_i} = p_{n_i} B_i | A_i, B_i \in \hat{G}\} \rangle.$$

The letters involved in the same relation are called contiguous. *Completing this definition with transitivity and reflexivity is obtained a partition of P given by $\bigcup_{n \in I} \{p_m\}_{m \in P_n}$ where all the $p_m \in P_n$ are contiguous to a fixed p_n for every $n \in I$. Since exist A'_{n_i}, B'_{n_i} such that $A_{n_i} p_{n_i} = p_n B'_{n_i}$ so by $p_{n_i} = A'^{-1}_{n_i} p_n B'_{n_i}$ is possible to eliminate all the p_m with $m \notin I$ and so present the group in the form:*

$$G = \langle \Sigma_{\hat{G}} \cup \{p_m\}_{m \in I} | A'_{n_l} p_n = p_n F_{n_l} \rangle.$$

Definition 46 *The system P of stable letters is* regular *if every $p_m \in I$ are stable letters. For p_{n_i}, p_{n_j} contiguous is possible to define*

$$\mathcal{A}_{p_{n_i}, p_{n_j}} = A'_{n_j} \mathcal{A}_{p_n} A'_{n_i} \qquad \mathfrak{B}_{p_{n_i}, p_{n_j}} = B'_{n_j} \mathfrak{B}_{p_n} B'_{n_i}$$

where $A'_{n_j}, A'_{n_i}, B'_{n_j}$ and B'_{n_i} are words participating in the relation which links letters p_{n_i}, p_{n_j} to p_n. It is also valid the following notational equality:

$$\mathcal{A}_{p^{\epsilon}_{n_i} p^{\epsilon}_{n_i}} \Bumpeq \mathfrak{B}_{p^{-\epsilon}_{n_j} p^{-\epsilon}_{n_i}} \; .$$

In the same manner of Chap.4.2 is possible to define the individuality of a letter and extended system of transformation to reformulate the lemmas 27, Britton's and Novikov's lemmas. For example the analogous of lemma 27 tell that, given a chain of extended transformation

$$Wp^{\epsilon}_{n_1}U \to W_1p^{\epsilon}U_1 \to \ldots \to W_kp^{\epsilon}_{n_k}U_k$$

where $p^{\epsilon}_{n_i}$ have the same individuality, there exists twin words $\mathcal{A}_{p^{\epsilon}_{n_i}p^{\epsilon}_{n_i}}$ and $\mathfrak{B}_{p^{\epsilon}_{n_i}p^{\epsilon}_{n_i}}$ such that:

$$W = W_n\mathcal{A}_{p^{\epsilon}_{n_k}p^{\epsilon}_{n_1}} \text{ and } U = \mathfrak{B}^{-1}_{p^{\epsilon}_{n_1}p^{\epsilon}_{n_k}}U_n$$

while Britton lemma tells that given a regular system of stable letters P of a group G over $\hat{G}$ and a word $W =_G 1$ than either $W \in \hat{G}$ and $W =_{\hat{G}} 1$ or W includes subword of the form $p^{-\epsilon}_{n_j}\mathcal{A}_{p^{\epsilon}_{n_i}p^{\epsilon}_{n_j}}p^{\epsilon}_{n_i}$.

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 47 (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 *($A_i, B_i \equiv Sq_eS'$ where S, S' are $\{s_d\}$-words).*

- $G_0 = \langle x, y\rangle$;
- G_1: $\{s_d | d \in D\} \quad | \quad \mathbf{y}ys_d = s_d\mathbf{y},\ \mathbf{x}s_d = s_dx\mathbf{x}$,
  $\mathcal{A}_{s_d} = V(x, y^2)$, $\quad \mathfrak{B}_{s_d} = V(x^2, y)$;
- G_2: $\{l_i, r_i | 1 \leqslant i \leqslant N\} \quad | \quad \mathbf{s_d}l_i = yl_iy\mathbf{s_d},\ \mathbf{s_d}xr_ix = r_i\mathbf{s_d}$,
  $\mathcal{A}_{l_i} = V(y^{-1}s_d)$, $\mathfrak{B} = V(ys_d)$, $\mathcal{A}_{r_i} = V(s_dx)$, $\mathfrak{B}_{r_i} = V(s_dx^{-1})$;
- G_3: $\{q_e | e \in E\} \quad | \quad A_i = \mathbf{l_i}B_i\mathbf{r_i}, A_i \equiv A'_iq_{n_i}A''_i, B_i \equiv B'_iq_{m_i}B''_i$,
  where A'_i, A''_i, B'_i, B''_i are $\{s_b\}$-words and
  $\mathcal{A}_{q_{m_i}q_{n_i}} = V(A'^{-1}_i l_iB'_i)$, $\mathfrak{B}_{q_{n_i}p_{m_i}} = V(A''_i r^{-1}_i B''^{-1}_i)$;
- G_4: $\{t\} \quad | \quad \mathbf{l_i}t = t\mathbf{l_i},\ \mathbf{y}t = t\mathbf{y}$,
  $\mathcal{A}_t = V(l_i, y) = \mathfrak{B}_t$;
- *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 33 *The Boone group $G(T, q) = G_5$ have a standard basis.*

Proof: *Let's build the set C_i of the words in standard normal form for every G_i:*

- *C_0 is equal to the set of all irreducible words on the alphabet $\{x, y\}$ (also negative letters), by definition $\mathcal{A}_x \equiv \mathfrak{B}_x \equiv \mathcal{A}_y \equiv \mathfrak{B}_y \equiv 1$;*

[1]App.A

- *the set C_1 it's constituited by words in the form*

$$C(W) = U_1 s_{d_1} U_2 \dots U_k s_{d_k} U_{k+1}$$

where $U_i \in C_0$ and $C(W)$ does not contain subword in the form:

$$\alpha \mathfrak{B}_\alpha A'' p, \qquad \alpha^{-1} \mathfrak{B}_{\alpha^{-1}} A'^{-1} p, \qquad \beta \mathfrak{B}_\beta B'' p^{-1}, \qquad \beta^{-1} \mathfrak{B}_{\beta^{-1}} B'^{-1} p^{-1}.$$

Since $A = \mathbf{y}y, B = \mathbf{y}$ ($A' \doteq 1, A'' \doteq yB' \doteq B'' \doteq 1$) or $A = \mathbf{x}, B = x\mathbf{x}$ ($A' \doteq A'' \doteq 1, B' \doteq x, B'' \doteq 1$), the prohibiten words wil be in the form:

$$yV(y)A''s_d, \qquad y^{-1}V(y)A'^{-1}s_d, \qquad yV(y)B''s_d^{-1}, \qquad y^{-1}V(y)B'^{-1}s_d^{-1},$$

$$xV(x)A''s_d, \qquad x^{-1}V(x)A'^{-1}s_d, \qquad xV(x)B''s_d^{-1}, \qquad x^{-1}V(x)B'^{-1}s_d^{-1},$$

so them have to contain a subword in the form:

$$y^2 s_d, \qquad y^{-1} s_d, \qquad y s_d^{-1}, \qquad y^{-2} s_d^{-1},$$

$$x s_d, \qquad x^{-1} s_d, \qquad x s_d^{-1}, \qquad x^{-2} s_d.$$

In that simple case is possible to see that in a normal form word in G_1 before a positive s_d there could be:

1. *the word before a positive s_d have to terminate with a single occurrence of an y;*
2. *the word before a negative s_d have to terminate with a single occurrence of a negative x.*

- *the set C_2 it's consists of reduced word in the form*

$$U_1 \alpha_{i_1} U_2 \dots U_k \alpha_{i_k} U_{k+1}$$

where $U_i \in C_1$, $\alpha_{i_j} \in \{r_i, l_i | i \leqslant i \leqslant N\}$ containing no subword in the form:

$$s_d V(x^2, y) l_i^\epsilon, \qquad s_d^{-1} V(x, y^2) y^\epsilon l_i^\epsilon,$$

$$s_d V(x^2, y) x^\epsilon r_i^\epsilon, \qquad s_d^{-1} V(x, y^2) r^\epsilon,$$

where V, Vx^ϵ ,Vy^ϵ (where$V = V(x^2, y)$ or $V(x, y^2)$) are reduced, $d \in D$, $1 \leqslant i \leqslant N$. Since a word $\mathcal{A}_l$ can be in the form ySy^{-1} with S reduced word in $\{s_d\}$, elimination rule could not and the word in the form

$$s_d V(x^2, y) l_i^\epsilon \qquad \text{or} \qquad s_d^{-1} V(x, y^2) y^\epsilon l_i^\epsilon$$

are prohibited, lemma 32 is verified for that kind of word (choosing x_1 the first letter of S and x_2 the last one), else lemma 31 holds.

- *To verify the existence of the standard basis for G_3 will suffice to use the lemma 31: since a word $\mathcal{A}_{q_m q_n}, \mathfrak{B}_{q_m q_n}$ are equal to 1 iff his projection on the alphabet $\{l_i, r_i\}$ is equal to 1. It follows that the letters q_e are regoular and as above is possible to apply the lemma 32, so G_3 is a gruop with standard basis;*

- *In G_4 the prohibited word are in the form*

$$y^\delta t^\epsilon \qquad l_i C(y^{-1}SyV(y))t^\epsilon \qquad l_i^{-1}C(ySy^{-1}V(y))t^\epsilon$$

  where $\delta = \pm 1$ and S a reduced $\{s_d\}$-word. Since every elimination of prohibited word reduce the number of l_i or y. The lemma 31 is proved because if two reduced word Ut, Wt where $U = W\mathcal{A}_t = WV(l_i, y)$ than $V(l_i, y) \equiv 1$.

- *Finally a normal form word contains no subword of form:*

$$r_i^\delta k^\epsilon \ , \qquad x^\delta k^\epsilon \ , \qquad t^\delta C(V(l_i, y)qW(r_i, x))k^\epsilon$$

  where $\delta = \pm 1$. The presence of $W(r_i, x)$ in the last class of prohibited words is due to the fact that $W(r_i, x)$ commute with k and by the fact that, if Σ is a special word of T such that $\Sigma =_T q$, then $\Sigma^{-1}t\Sigma k =_{G_5} k\Sigma^{-1}t\Sigma$.

Lemma 34 *The word problem for the group G_4 is solvable.*

Proof: *Let us verify that, for every word $X \in G_4$, the canonical word $C(X)$ equal to X is effectively calculable. As observed, the problem reduction to canonical form it depends of the recognition of prohibited words. The last one is related to the recognition problem for the subwords in the form $\mathcal{A}_x, \mathfrak{B}_x$ where x is a stable letter of the group G_i, $i < 4$, but also $\mathcal{A}_{x,x'}, \mathfrak{B}_{x,x'}$ where x, x are contiguous stable letters (in the case of $i = 3$). For the groups G_1, G_2 to recognize prohibited words is trivial. For example in G_2 we need to recognize a word $V(x, y^2)$ among the word of the group G_0 and an irreducible word equal to $V(x, y^2)$ coincides with it.*

In the group G_3 we have to solve the membership problem for words of G_1 and each of the following subgroup

$$A_1 = \langle y^{-1}s_b\rangle, \qquad A_2 = \langle ys_b\rangle, \qquad A_3 = \langle s_b x\rangle, \qquad A_4 = \langle s_b x^{-1}\rangle.$$

In the case of A_1, for example, an arbitrary element is representable in the form ySy^{-1} where S is an irreducible word in $\{s_b | b \in B\}$. Then a word $U \in G_1$ is equals of a word in the subgroup A_1 iff $U = C(ySy^{-1})$.

For the group G_4 we have to recognize, for example, the word in the form $V(y^{-1}s_b)y^l = ySy^{-1}y^l$ among the words of the group G_1. If a canonical word U equals a word, of the form tackled, in the group G_1 then the word S is the projection of the word U on the alphabet $\{s_b\}$ and $C(yS^{-1}y^{-1}U) \equiv y^l$, which can be recognized by reducing to canonical form in the group G_1.

Lemma 35 *Let S, S' special word in T then $S =_T S'$ iff there exist $V(l_i, y), W(r_i, x)$ such that:*

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

Proof:

$\Leftarrow$ *Let S, S' be special words in T such that $S =_T S'$. Inductively it suffice to consider the case $S \to S'$: if $S \equiv U(s_b)A_iU'(s_b)$ and $S' \equiv U(s_b)B_iU'(s_b)$ exist $U(s_b), U'(s_b)$. Since $A_i = l_iB_ir_i$, for some $V(l_i, y)$ and $W(r_i, x)$, the equality $S = V(l_i, y)S'W(r_i, x)$ holds in the group $G(T, q)$.*

$\Rightarrow$ *Let $S = U(s_b)A_iU'(s_b)$, $S' = U(s_b)B_iU'(s_b)$ and $V(l_i, y) = y^{n_0}l_{i_i}^{\epsilon_1}\cdots l_{i_k}^{\epsilon_k}y^{n)k}$ with V reduced and $C(y^{n_0}l_{i_i}^{\epsilon_1}\cdots l_{i_k}^{\epsilon_k}S'_1q_m) = S_1q_mU$ with $U \in G$. However, the normal form of*

$$y^{n_0}l_{i_i}^{\epsilon_1}\cdots l_{i_k}^{\epsilon_k}S'_1q_m$$

given by the standardization process will give us

$$y^{n_0}l_{i_i}^{\epsilon_1}\cdots l_{i_k}^{\epsilon_k}S'_1 =_{G_1} S_1\mathcal{A}(l_{i_i}^{\epsilon_1}\cdots l_{i_k}^{\epsilon_k})$$

where $\mathcal{A}(l_{i_i}^{\epsilon_1}\cdots l_{i_k}^{\epsilon_k}) = \mathcal{A}_{q_mq_n}$ and $\mathcal{A}_{q_mq_n}$ is reduced.

It follows that $\mathfrak{B}(r_{i_i}^{-\epsilon_1}\cdots r_{i_k}^{-\epsilon_k})S'_2W(r_i, x) = S_2$ and by Britton's lemma we have $W(r_i, x) = x^{m_k}r_{i_k}^{\epsilon_k}\cdots l_{r_1}^{\epsilon_1}x^{m_0}$.

By induction if $k = 0$, then $S = S'$. If $k \geqslant 1$ an , for example $\epsilon_k = 1$ than we have

$$l_{i_k}y^{n_k}S'_1q_{m_{i_k}} = l_{i_k}y^{-1}SyB_{i_k1}q_{m_{i_k}}$$

where S is a positive $\{s_d\}$-word, From a first reduction of the right side we obtain

$$S = y^{n_0}l_{i_i}^{\epsilon_1}\cdots l_{i_{k-1}}^{\epsilon_{k-1}}y^{n_{k-1}}SA_{i_k1}q_{n_{i_k}}A_{i_k2}r_{i_k}^{-1}B_{i_k2}^{-1}S'_2x^{m_k}r_{i_k}^{\epsilon_k}\cdots l_{r_1}^{\epsilon_1}x^{m_0}$$

and by $W(r_i, x) = x^{m_k}r_{i_k}^{\epsilon_k}\cdots l_{r_1}^{\epsilon_1}x^{m_0}$

$$B_{i_k2}^{-1}S'_2x^{m_k} = xS'x^{-1}$$

where S' is a positive word, so $S'_2 = \Gamma_{i_k2}S'$. In this way we obtain

$$S = y^{n_0}l_{i_i}^{\epsilon_1}\cdots l_{i_{k-1}}^{\epsilon_{k-1}}y^{n_{k-1}}SA_{i_k1}q_{n_{i_k}}A_{i_k2}x^{m_{k-1}}r_{i_{k-1}}^{\epsilon_k}\cdots l_{r_1}^{\epsilon_1}x^{m_0}$$

and $S' = SB_{i_k1}q_{m_{i_k}}B_{i_k2}S'$ in T and so $S =_T S'$ as required.

Lemma 36 *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.*
Proof: *By induction, it is possible to define $\mathfrak{B}(r_{i_1}^{\epsilon_1}, \ldots, r_{i_k}^{\epsilon_k})$ for $k \geqslant 0$;*

$$\mathfrak{B}(1) = 1, \qquad \mathfrak{B}(r_i^{\epsilon}) = (A_{i2}r_iB_{i2}^{-1})^{\epsilon}, \qquad \mathfrak{B}(r_{i_1}^{\epsilon_1}, \ldots, r_{i_k}^{\epsilon_k}) = \mathfrak{B}(r_{i_1}^{\epsilon_1}, \ldots, r_{i_{k-1}}^{\epsilon_{k-1}})\mathfrak{B}(r_{i_k}^{\epsilon_k}).$$

Suppose that Q is a canonical word for G_3 in the form $V(l_i, y)SW(r_i, x)$ where V, W are in canonical word and S special. Among all the word inthe form $V(l_i, y)SW(r_i, x)$ equal to Q, if we take the one with the minimum number of l_i we have $Q = C(VSW) = Q_1(l_i, y, s_d)q_nC(\mathfrak{B}_{q_nq_m}S''W(r_i, x))$ where $S = S'q_mS''$,the word Q_1 does not contain s_d^{-1} and $\mathfrak{B}_{q_nq_m}$ is an r_j-irriductible word. In particolar we conclude that

$$C(Q_1\mathcal{A}_{q_nq_m}) = C(VS')$$

therefore this word does not contain s_d^{-1}. If a word $\mathfrak{B}_{q_nq_m}$ ends with the word $r_i^{-1}B_{i2}^{-1}$ or $r_iB_{i2}^{-1}$ then the wrod S'' does not begin with the word B_{i2} or A_{i2} respectively; otherwise the word Q is representable the form $V(l_i, y)SW(r_i, x)$ having less l_i letters than V. This imply that during the process to reduce the word $\mathfrak{B}_{q_nq_m}S''W$ to a canonical form, no letters r_i are cancelled.

So Q is in the form $V(l_i, y)SW(r_i, x)$ iff $Q = Q_1(l_i, y, s_d)q_nQ_2(r_i, c, s_d)q_n$ where Q_1 does not contain s_d^{-1} and if $Q_2 = R_0r_{i_1}^{\epsilon_1}, \dots, r_{i_k}^{\epsilon_k}R_k$ with $k \geqslant 0$, where R_j does not contain r_i^{ϵ}, then there exists $h \in \{0, \dots, k\}$ such that $\mathfrak{B}(r_{i_1}^{\epsilon_1}, \dots, r_{i_k}^{\epsilon_k})$ is a word in the form $\mathfrak{B}_{q_nq_m}$ for a suitable q_m and all the words $C(Q_1\mathcal{A}_{q_mq_n})$, $C(\mathfrak{B}_{q_nq_m}^{-1}Q_2)$ do not contain the letters s_d^{-1}. All these conditions are algorithmically recognizable.

Theorem 37 *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.*

Proof: *By lemma 34 and Theor.33 is possible, for all words $W \in G$, to calculate its normal form $C(W) = U_1kU_2k \dots U_nkU_{n+1}$ in a finite number of reduction. Since the word problem of G_4 is solvable the problem is deduced determinate if a word Q in G_3 is equal or not to a word $V(l_i, y)qW(x, r_i)$. By lemma 36 is possible to determinate if a word Q is equal to a word in the form $V(l_i, y)\Sigma W(x, r_i)$. So lemma 31 the decidability of word problem for $G(T, q)$ can be reduced to decidability of word problem for the monoid T.*

Corollary 38 (Undecidability of word problem for the groups) *There exists a finitely presented gruop with undecidable word problem.*

Proof: *By Theo.42 exists a finite presented monoid T with defining relation given by special words and undecidable word problem, so by Theo.37 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.3 : $F_2 = \langle a, b\rangle$ and $a_n = b^nab^{-n}$.

Lemma 39
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 $\phi(a_{p+qz}) = a_{p'+q'z}$.
Proof: *By Lem. 4 $\langle a_p, b^q\rangle = F_2 = \langle a_{p'}, b^{q'}\rangle$ so it exists an isomorphism ϕ such that $\phi(a_p) = a_{p'}$ and $\phi(b^q) = b^{q'}$ and so:*

$$\phi(a_{p+qz}) = \phi((b^q)^za_p(b^{-q})^z) = \phi((b^q)^z)\phi(a_p)\phi((b^{-q})^z) = (b^{q'})^za_{p'}(b^{-q'})^z = a_{p'+zq'}.$$

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 40 *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: *Let $K = [p + q\mathbb{Z}]_{F_2}$. Every reduced word w in $\langle a_p, b^q\rangle$ can be written in the form uv with $u \in K$ and $v \in \langle b^q\rangle$, because there are $k_i \in \mathbb{Z}$ and $\delta_i \in \{-1, 1\}$ such that:*

$$w = b^{k_0q}a_{\delta_1p}b^{k_1q}a_{\delta_2p}\cdots a_{\delta_np}b^{k_nq} =$$

$$= b^{m_0=k_0q+\delta_1p}ab^{m_1=k_1q+(\delta_2-\delta_1)p}a\cdots ab^{m_n=k_nq-\delta_np} =$$

$$= a_{m_0}a_{m_1+m_0}\cdots a_{\Sigma_{i=0}^{j}m_i}\cdots a_{\Sigma_{i=0}^{n-1}m_i}b^{\Sigma_{i=1}^{n}m_i} .$$

Let $\pi : F_2 \to \langle b \rangle$ the projection of F_2 on $\langle b \rangle$ (i.e. $\pi(a) = 1, \pi(b) = b$), so $K \subseteq [\mathbb{Z}]_{F_2} \subseteq ker(\pi)$ and $\forall x \in \langle a_p, b_q \rangle, \pi(x) = \pi(uv) = \pi(u)\pi(v) = \pi(v) = v$ so $[\mathbb{Z}]_{F_2} \cap \langle a_p, b^q \rangle \subseteq K$. By $K \subseteq [\mathbb{Z}]_{F_2}$ and $k \subseteq \langle a_p, b_q \rangle$ follows the equality.

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.26 is possible to obtain an extension of $F_{\mathcal{A}}$ of F_2 with stable letters $t_1 \dots t_n$ which represents the local isomorphism $\phi_1 \dots \phi_n$. Let $P = \{z \in \mathbb{Z} | z \leftrightarrow^*_{\mathcal{A}} m\}$ and $H = \langle a_m, t_1, \dots t_n \rangle$. By Lemma 39 follow:*

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

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. 26) $K = H \cap F_2$.
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.18).

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.*

6.1 Word problem and Propositional linear logic

The results of the undecidability of word problem for monoids can be used to prove the undecidability of non commutative propositional linear logic (NCL, see App.B) even in the multiplicative fragment [16]. We'll encode the word problem for monoids in a similar way of Chap.2.5 encoding axioms capable to represent derivation correspondig to computation of rewriting systems.

We'll define the translation of the word $[ab \dots z]$ *as list of NCL formulas* $p_a^\perp, p_b^\perp, \dots, p_z^\perp$, $[ab \dots z]^\perp$ *as* $p_a^\perp \otimes p_b^\perp \otimes \dots \otimes p_z^\perp$ *and of the rules* $v \to w \in \mathcal{R}$ *as an axiom* $[v \to w]$ *interpreted so* $\vdash [v], [w]^\perp$.

For any rewriting system $(\Sigma|\mathcal{R})$ *we'll be possible to associate a set of formula* $\{p_{a_i}, p_{a_i}^\perp\}_{a_i \in \Sigma}$ *and a theory* $T_{(\Sigma|\mathcal{R})} = \{[v \to w]\}_{v \to w \in \mathcal{R}}$.

With this encoding, a "pair of words" problem $P(v, w) : v \to_{\mathcal{R}}^* w$*? for two words* v *and* w *it's equivalent to prove the sequent*

$$\vdash [v][w]^\perp$$

in the relative NCL theory encoding the rewriting system.

Proposition 15 *A "pair of words" problem* $P(v, w)$ *is solvable in* $(\Sigma|\mathcal{R})$ *iff* $\vdash [v][w]^\perp$ *is provable in the theory* $T_{(\Sigma|\mathcal{R})}$.
Proof: *we'll not give a proper proof of the lemma but we'll show an example to understand how NCL simulate a rewriting system. First we note that if* $U = u_1 \dots u_k$ *we have:*

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash p_{u_1}^\perp, p_{u_1}}\,I \qquad \dfrac{}{\vdash p_{u_2}^\perp, p_{u_2}}\,I}{\vdash p_{u_1}^\perp, p_{u_2}^\perp, (p_{u_1} \otimes, p_{u_2})}\,\otimes \qquad \dfrac{}{\vdash p_{u_3}^\perp, p_{u_3}}\,I}{\vdash p_{u_1}^\perp, p_{u_2}^\perp, p_{u_3}^\perp, (p_{u_1} \otimes p_{u_2} \otimes, p_{u_3})}\,\otimes}{}$$

$$\dfrac{\vdots \qquad\qquad\qquad\qquad \vdots}{\vdash p_{u_1}^\perp, \dots p_{u_k}^\perp, (p_{u_1} \otimes \dots \otimes p_{u_k})}\,\otimes$$
$$\|$$
$$\vdash [U], [U]^\perp$$

so if $U = V$, $\vdash [U], [V]^\perp$ *is provable.*

The sequent calculus to simulate the application of the rule $W \to W'$ *on the word* UWV *to obtained the word* $UW'V$ *is the following:*

$$\dfrac{\dfrac{}{\vdash W, [W']^\perp}\,T \qquad \dfrac{\dfrac{\dfrac{\vdots}{\vdash [U], [W'], [V]}\,\parr}{\vdots}}{\vdash [U], \gamma([W']), [V]}\,\parr}{\vdash [U], [W], [V]}\,Cut$$

where $\gamma([W'])$ *is the formula derived by* $[W]$ *applying all the possible* $\parr$ *rules.*

The following concrete example simulates the application of the rule $cd \to xy$ *on the word* $UcdV$ *to obtained the word* $UxyV$*:*

$$\dfrac{\dfrac{}{\vdash p_c^\perp, p_d^\perp, (p_x \otimes p_y)}\,T \qquad \dfrac{\begin{array}{c}\vdots\\ \vdash [U], p_x^\perp, p_y^\perp, [V]\end{array}}{\vdash [U], (p_x^\perp \parr p_y^\perp), [V]}\,\parr}{\vdash [U], p_c^\perp, p_d^\perp, [V]}\,Cut$$

In such proofs, the application of Cut rules correspond to an application of a rewriting rule in the rewriting system.

Appendix A

Combinatorial system

Rewriting system, Post system, Thue system are different system of substitution of substrings in strings with the same base concept:

Definition 48 (production) *[11] Fixed an alphabet Σ, a* rewrite rule, semi-Thue production*or simply* production *is an expression*

$$u \to v$$

if P is a semi-Thue production $u \to v$, $A, B \in \Sigma^*$

$$A \to_P B$$

mean that exists $A', A'', B', B'' \in \Sigma^*$ such that $A = A'uA''$ and $B = B'vB''$ A normal production *is a produictin in the form $uv \to vu'$. Two word in $u, w \in \Sigma^*$.*

Definition 49 *A* combinatorial system *consists of an alphabet and a set of pair of words callad production.*

A semi-Thue system *or* string rewrite system $S = (\Sigma|\mathcal{R})$ *is given by an alphabet and a finite set of rewriting rule. A* Thue system *is a semi-Thue system where for every rewriting rule $u \to v$ exists its inverse $v \to u$. A* Post system $P = [\Sigma; \Phi]$ *is a combinatorial system with a finite set of normal production. Two word are called* equivalent in P *(written $u \sim_P w$) if there exists a sequence of normal production which transform u in w.*

A.1 Undecidability of word problem for monoids

Proposition 16 *Every non deterministic Turing machine can be simulated by a semi-Thue system.*

Proof: *Let Σ the alphabet of T and $Q = \{q_i\}$ the states of T. Is so possible ([12]) code the configurations of a Turing machine T writing the contents of the tape as a word on the alphabet Σ and placing the letter q_i before the letter read by the head. We'll call a word on a bipartite alphabeth $\Sigma \cup Q$* special *if in it there will be a single occurrence of letters in Q. To understand the intuition we'll give the following:*

Example: *the special word $a_0 \dots a_{i-1} q a_i \dots a_n$ will correspond to the following configuration of the machine:*

				q				
				$\triangledown$				
$\dots$	a_0	$\dots$	a_{i-1}	a_i	a_{i+1}	$\dots$	a_n	$\dots$

So every transition given by δ will correspond a rewriting rule in the form:

$$q a_i \to a_i' q' \qquad \text{if} \qquad \delta(a_i, q) = (a_i', q', R)$$

$$a_{i-1} q a_i \to a_{i-1} a_i' q' \qquad \text{if} \qquad \delta(a_i, q) = (a_i', q', L)$$

Let R be the set of rule corresponding to the set of 5-tuple describing δ, so the presentation $(\Sigma \cup Q|\mathcal{R})$ of the monoid $M(T)$ can simulate every computation of T. If w is a special word corresponding to a final configuration (i.e. the letters in Q is in $\perp$), to test if a word v is equal to w we need to verify if a computation starting from the configuration v will terminate with w, so in particular, if T terminate some of its computation. This problem is exactly the halting problem.

Corollary 41 *Semi-thue system are a Turing complete computation model.*

This proves the following:

Theorem 42 (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.

The following example is given by Ceitin in [8].

Example: *The semigroup $\langle a, b, c, d|\mathcal{R}\rangle^+$ where $\mathcal{R}$ are the relations:*

$$ac = ca, \qquad ad = da, \qquad bd = db, \qquad ce = eca, \qquad dc = edb, \qquad cca = ccae$$

has unsolvable word problem.

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 $\mathcal{M}$:

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

In this monoid $s_1 \neq_M s_3$ but in the group $G = \langle \Sigma_{\mathcal{M}} | \mathcal{R}\rangle$ we have $a^{-1}as_1 \to s_1$ and $a^{-1}as_1 \to a^{-1}s_2 \to a^{-1}ac_3 \to s_3$ so $s_1 =_G s_3$.
This is not coherent with the register machine computing since:

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

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 43 (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 $\{(a_ia_j, a_ja_i), (a_ja_i, a_ia_j)\}_{a_i,a_j\in\Sigma} \subseteq \leftrightarrow^*_{\mathcal{R}}$ 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. We'll note with $p_i, p_i^\perp$ propositional literal, $A, B, C, \ldots$ for formulas and Σ, Γ, Δ for multisets of formulas.

- *Identity*

$$\frac{}{\vdash A, A^\perp}\, I$$

- *Cut*

$$\frac{\vdash \Sigma, A \qquad \vdash \Gamma, A^\perp}{\vdash \Sigma, \Gamma}\, Cut$$

- *Tensor*

$$\frac{\vdash \Sigma, A \qquad \vdash B, \Gamma}{\vdash \Sigma, (A \otimes B), \Gamma}\, \otimes$$

- *Par*

$$\frac{\vdash \Sigma A, B}{\vdash \Sigma, A \parr B}\, \parr$$

- *Plus*

$$\frac{\vdash \Sigma A}{\vdash \Sigma, A \oplus B}\, \oplus \qquad \frac{\vdash \Sigma B}{\vdash \Sigma, A \oplus B}\, \oplus$$

- *With*

$$\frac{\vdash \Sigma, A \qquad \vdash \Sigma, B}{\vdash \Sigma, (A \& B)}\, \&$$

- *Weakening*

$$\frac{\vdash \Sigma}{\vdash \Sigma, ?A}\ ?\mathbf{W}$$

- *Contraction*

$$\frac{\vdash \Sigma, ?A, ?A}{\vdash \Sigma, ?A}\ ?\mathbf{C}$$

- *Dereliction*

$$\frac{\vdash \Sigma, A}{\vdash \Sigma, ?A}\ ?\mathbf{D}$$

- *Of course / Bang*

$$\frac{\vdash ?\Sigma, A}{\vdash ?\Sigma, !A}\ !\mathbf{S}$$

- *Botton*

$$\frac{\vdash \Sigma}{\vdash \Sigma, \bot}\ \bot$$

- *1*

$$\frac{}{\vdash 1}\ 1$$

- *True*

$$\frac{}{\vdash \Sigma, T}\ T$$

There also is an exchange rule usually omitted in linear logic:

$$\frac{\vdash A_1, \ldots, A_k}{\vdash A_{\sigma(1)}, \ldots, A_{\sigma(k)}}\ \sigma \in S_k$$

The principal formula *of an inference rules is any formula introduced by that rule, we say that a formula is* active *in a Cut if it appear in one of premises but not in the conclusions.*

With the term multiplicative fragment *(MLL) is denoted the calculus with inference rules $I, Cut, \otimes, \parr$ and with MALL the* multiplicative-additive fragment *with rules $I, Cut, \otimes, \parr, \&, \oplus$. The non-commutative linear logic NCL is given prohibiting the use of the exchange rules.*

B.1 Cut Elimination

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

Theorem 44 *If A is provable in linear logic , there exists a prove of A without Cut rules.*
Proof: *We have to give a procedure to eliminate Cut rules form derivation:*

- *if both formulas active in the cut are not principal, it will be possible to simply switch the position of the cut going up in the proof tree;*
- *if at least one of the formulas active in the cut is not principal, it will be possible to simply switch the position of the cut:*
  - *In the case of binary rules (different from cut):*

$$\dfrac{\dfrac{\overset{\vdots}{\vdash \Sigma, A} \qquad \overset{\vdots}{\vdash \Sigma', C, B}}{\vdash \phi(\Sigma, \Sigma'), C, (A \square B)}\,\square \qquad \overset{\vdots}{\vdash \Gamma, C^{\perp}}}{\vdash \phi(\Sigma, \Sigma'), (A \square B)}\,Cut$$

became

$$\dfrac{\overset{\vdots}{\vdash \Sigma, A} \qquad \dfrac{\overset{\vdots}{\vdash \Sigma', C, B} \qquad \overset{\vdots}{\vdash \Gamma, C^{\perp}}}{\vdash \Sigma', \Gamma, B)}\,Cut}{\vdash \phi(\Sigma, \Sigma'), \Gamma, (A \square B)}\,\square$$

where

$$\phi(\Sigma, \Sigma') = \begin{cases} \Sigma, \Sigma' & \textit{if } \square = \otimes \\ \Sigma & \textit{if } \Sigma = \Sigma', C \textit{ and } \square = \& \end{cases} .$$

  - *In case of unary rules:*

$$\dfrac{\dfrac{\overset{\vdots}{\vdash \Sigma, C, B, A}}{\vdash \phi(\Sigma, B), C, \phi(A, B)}\,\square \qquad \overset{\vdots}{\vdash \Gamma, C^{\perp}}}{\vdash \phi(\Sigma, B), \Gamma, \phi(B, A)}\,Cut$$

became

$$\dfrac{\dfrac{\overset{\vdots}{\vdash \Sigma, C, B, A} \qquad \overset{\vdots}{\vdash \Gamma, C^{\perp}}}{\vdash \Sigma, B, A}\,Cut}{\vdash \phi(\Sigma, B), \phi(A, B)}\,\square$$

where

$$(\phi(\Sigma, B); \phi(B, A)) = \begin{cases} (\Sigma; A \parr B) & \textit{if } \Box = \parr \\ (\Sigma, B; A \oplus D) & \textit{if } \Box = \oplus \\ (\Sigma, B, A; ?D) & \textit{if } \Box = ?\mathbf{W} \\ (\Sigma, B, A; ?D) & \textit{if } \Box = ?\mathbf{C} \textit{ and } \Sigma = \Sigma', ?D, ?D \\ (\Sigma, B; ?A) & \textit{if } \Box = ?\mathbf{A} \end{cases}.$$

- *if both formulas active in the cut are principal, it will be possible to simply switch the position of the cut: In the case of binary rules (different from cut):*

$\otimes$ *vs* $\parr$

$$\dfrac{\dfrac{\overset{\vdots}{\vdash \Sigma, A} \qquad \overset{\vdots}{\vdash B, \Sigma'}}{\vdash \Sigma, A \otimes B, \Sigma'}\otimes \qquad \dfrac{\overset{\vdots}{\vdash \Gamma, A^\perp, B^\perp}}{\vdash \Gamma, A^\perp \parr B^\perp}\parr}{\vdash \Sigma, \Gamma, \Sigma'}\mathit{Cut}$$

became

$$\dfrac{\overset{\vdots}{\vdash \Sigma, A} \qquad \dfrac{\overset{\vdots}{\vdash B, \Sigma'} \qquad \overset{\vdots}{\vdash \Gamma, A^\perp, B^\perp}}{\vdash \Gamma, A^\perp, \Sigma'}\mathit{Cut}}{\vdash \Sigma, \Gamma, \Sigma'}\mathit{Cut}$$

$\&$ *vs* $\oplus$

$$\dfrac{\dfrac{\overset{\vdots}{\vdash \Sigma, A} \qquad \overset{\vdots}{\vdash \Sigma, B,}}{\vdash \Sigma, A \mathbin{\&} B}\& \qquad \dfrac{\overset{\vdots}{\vdash \Gamma, A^\perp}}{\vdash \Gamma, A^\perp \oplus B^\perp}\oplus}{\vdash \Sigma, \Gamma}\mathit{Cut}$$

became

$$\dfrac{\overset{\vdots}{\vdash \Sigma, A} \qquad \overset{\vdots}{\vdash \Gamma, A^\perp}}{\vdash \Sigma, \Gamma}\mathit{Cut}$$

$?\mathbf{W}$ *vs* $!S$

$$\dfrac{\dfrac{\overset{\vdots}{\vdash \Sigma}}{\vdash \Sigma, ?A}?\mathbf{W} \qquad \dfrac{\overset{\vdots}{\vdash ?\Gamma, A^\perp}}{\vdash ?\Gamma, !A^\perp}!\mathbf{S}}{\vdash \Sigma, ?\Gamma}\mathit{Cut}$$

became

$$\dfrac{\dfrac{\vdots}{\dfrac{\vdash \Sigma}{\vdots}\ ?\mathbf{W}}}{\vdash \Sigma, ?\Gamma}\ ?\mathbf{W}$$

?**C** *vs* !*S*

$$\dfrac{\dfrac{\begin{matrix}\vdots\\ \vdash \Sigma, ?A, ?A\end{matrix}}{\vdash \Sigma, ?A}\ ?\mathbf{C} \qquad \dfrac{\begin{matrix}\vdots\\ \vdash ?\Gamma, A^{\perp}\end{matrix}}{\vdash ?\Gamma, !A^{\perp}}\ !\mathbf{S}}{\vdash \Sigma, ?\Gamma}\ Cut$$

became

$$\dfrac{\dfrac{\vdots}{\dfrac{\vdash \Sigma}{\vdots}\ ?\mathbf{W}}}{\vdash \Sigma, ?\Gamma}\ ?\mathbf{W}$$

?**D** *vs* !*S*

$\perp$ *vs* 1

The theorem is still valid also in NCL fragment:

Theorem 45 *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:*

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

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.19.*

All the proofs of these theorem can be founded on [16], App.A.

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)*