Journal of Universal Computer Science, vol. 5, no. 9 (1999), 563-573
submitted: 10/6/99, accepted: 15/7/99, appeared: 28/9/99 © Springer Pub. Co.

A Polynomial Solution for 3-SAT in the Space of Cellular Automata in the Hyperbolic Plane

Maurice Margenstern
G.I.F.M., Université de Metz,
I.U.T. de Metz, Département d'Informatique,
Île du Saulcy,
57045 Metz Cedex, France,
e-mail : margens@iut.univ-metz.fr

Kenichi Morita
University of Hiroshima
Faculty of Engineering
Higashi-Hiroshima, 739-8527
e-mail : morita@ke.sys.hiroshima-u.ac.jp

Abstract. In this paper, we define cellular automata on a grid of the hyperbolic plane, based on the tessellation obtained from the regular pentagon with right angles. Taking advantage of the properties of that grid, we show that 3-SAT can be solved in polynomial time in that setting, and then we extend that result for any NP problem. Several directions starting from that result are indicated.

Categories: F.1.1, F.1.3

Keywords: cellular automata, hyperbolic plane, complexity theory

1 Introduction

It is not the first time that the hyperbolic plane appears in theoretical computer science, see for instance [Robinson 78], but it seems that it is the first time that the possibilities it contains are explicitly indicated.

We shall see how advantage can be taken from the rich properties of the hyperbolic plane for transforming 3-SAT problems into a tractable one.

In this extended abstract, we cannot introduce the reader to hyperbolic geometry. We assume he/she is familiar with elementary properties of the hyperbolic plane, see for instance [Meshkowski 64], [Magnus 74] and [Millman et al. 81]. The reader is also referred to [Margenstern et al. 2000] which contains what is needed for our constructions.

In the second section, we construct the basic tiling we shall use to devise cellular automata in the hyperbolic plane. A tool is provided, making it possible to easily 'find one's way' in that non-familiar plane where our usual four directions are of no use. In the third section, using that tool, we give a solution for 3-SAT in quadratic time, and we deduce from that the main result of this paper, namely:

Theorem 1 – *NP-problems can be solved in polynomial time in the space of cellular automata in the hyperbolic plane.*

In the fourth section, we discuss the consequences of that result and indicate several possible continuations.

2 Cellular automata in the hyperbolic plane

We shall make use of Poincaré's disk model for representing the hyperbolic plane. We shall often speak of h-lines when the word line will be kept for indicating the euclidean ones. We assume the reader is familiar with the elementary features of that model: if needed, see the already pointed at literature.

2.1 The pentagrid

We shall make use of a tiling of the hyperbolic plane constructed by tessellation on the regular pentagon with right angles. The tessellation consists in taking the reflections through the sides of the pentagon, then the sides of the images through those reflections of the original pentagon and so on. This process can be generalized, and it was proved by Poincaré, see [Poincaré 1882], that any triangle with angles π/ℓ, π/m and π/n with $\frac{1}{\ell}+\frac{1}{m}+\frac{1}{n}<1$ generates a unique tessellation defining a tiling of the hyperbolic plane. Elementary proofs of that theorem can be found in [Carathéodory 54] and [Maskit 71].

Applying that theorem to the triangle with the following angles : $\frac{\pi}{5}, \frac{\pi}{4}, \frac{\pi}{2}$, we obtain the existence and uniqueness, up to isometries, of the regular pentagon with right angles and the tiling associated to it which we call from now on *pentagrid*. It is partially represented below, in figure 1.

2.2 The Fibonacci tree

We now construct the tool we need for the third section. It also provides an alternative proof of the existence of the pentagrid.

It may be assumed that the center of the unit disk, say A, is a vertex of the pentagrid. Now, exactly four pentagons share A as a same vertex. By a rotation, it may be assumed that the sides of those pentagons are supported by the vertical and the horizontal diameters of the unit circle. And so, we may fix our attention on the south-western quarter of the unit disk later denoted by $\mathcal{Q}$. Our tessellation is a tiling of the whole hyperbolic plane if and only of it is a tiling of $\mathcal{Q}$.

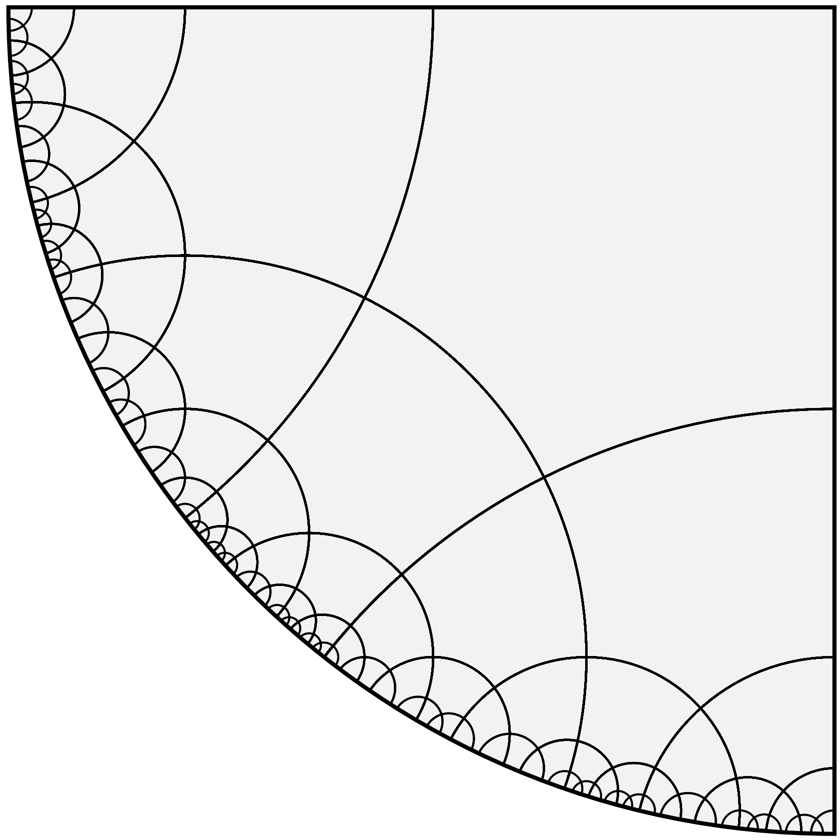

Figure 1 The south-west quarter of the hyperbolic plane in the representation of the Poincaré's disk.

Let P_0 be the pentagon in $\mathcal{Q}$ with A as a vertex. Call it the *leading* pentagon of $\mathcal{Q}$. Clockwise number its sides by **1**, **2**, **3**, **4** and **5** as indicated on figure 2. Two non consecutive edges of the pentagon have a common perpendicular and so, h-lines **2** and **3** do not intersect **5**. The complement of P_0 in $\mathcal{Q}$ can be split into three regions as follows. Call R_1 the closure of the trace in $\mathcal{Q}$ of the half-plane delimited by **2** which does not contain P_0. What remains in $\mathcal{Q}\backslash R_1$ is in the same way separated by **3** and, similarly, call R_2 the region delimited by **3** whose interior does not intersect P_0. Define R_3 in the same way with respect to **4**, R_1 and R_2 being deleted from $\mathcal{Q}$, see figure 2.

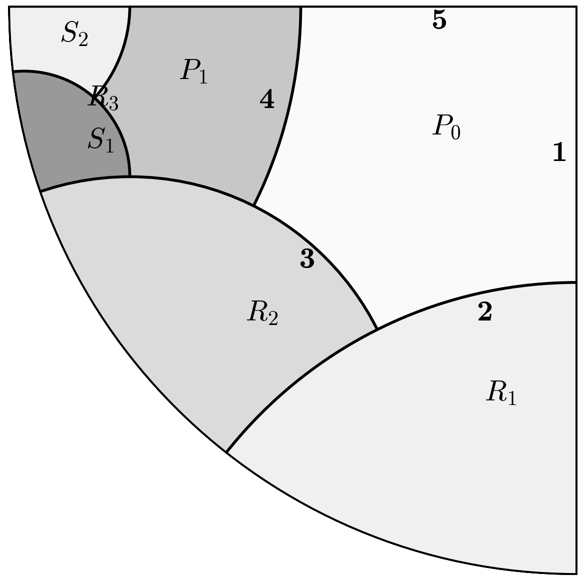

Figure 2 Splitting the quarter into four parts and R_3 into three parts
In light lines, the tessellation based on the leading pentagon of $\mathcal{Q}$.

Regions R_1 and R_2 are images of $\mathcal{Q}$ by, respectively, the hyperbolic shift (product of two reflections through lines possessing a common perpendicular) along **1** transforming **5** into **2** and the hyperbolic shift along **4** transforming **5** into **3**. Define a tree as follows: P_0 is associated to its root and let us consider that the root has three sons, ordered from the left to the right and respectively associated to pentagons in R_3, R_2 and R_1 that will later be defined. We shall say that the root is a 3-node because it has three sons.

Now, regions R_1 and R_2 are isometric to $\mathcal{Q}$ by the hyperbolic shifts above described. Accordingly, the same splitting can be done for these regions. The leading pentagons of R_1 and R_2 will be associated to the above indicated sons of the root. The above hyperbolic shifts indicate that in their turn, these sons are also 3-nodes. Region R_3 *is not* isometric to $\mathcal{Q}$. Let P_1 be the reflection of P_0 through **4** with sides numbered this time anticlockwise, in a way which allows to give the same number to edges supported by the same h-line. For avoiding possible confusions, we put the name of the considered pentagon as an index, when needed. P_1 is the pentagon associated to the leftmost son of the root. Line $\mathbf{2}_{P_1}$ delimits the image of $\mathcal{Q}$ through the hyperbolic shift along $\mathbf{1}_{P_1}$ transforming **5** into $\mathbf{2}_{P_1}$. Let S_1 be the considered region. Now $\mathbf{1}_{P_1}$ delimits S_2 which is the closure of $R_3 \backslash (S_1 \cup P_1)$. We associate two sons to the node associated to P_1 : the left one to the leading pentagon of S_2, *i.e.* the reflection of P_1 through $\mathbf{1}_{P_1}$, and the leading pentagon of S_1. Say that the node associated to P_1 is a 2-node, and notice that $S_2 \cup P_1$ is obtained from $R_3 \cup P_0$ by the hyperbolic shift along $\mathbf{5}_{P_0}$ transforming $\mathbf{1}_{P_0}$ into $\mathbf{4}_{P_0}$.

Starting from the axiom telling that the root is a 3-node, the tree is constructed by applying the rule: the leftmost son of a node is a 2-node and the other son(s) is/are 3-node(s). The uniquely defined tree in that way is called *Fibonacci tree* for the following reason. Let s_n be the number of nodes of height n in a Fibonacci tree, the root being of height 0, as usual, and let d_n, t_n be, respectively, the numbers of 2- and 3-nodes of height n. Then, by induction on n, it can be seen that $d_n = f_{2n-1}$, $t_n = f_{2n}$ and $s_n = f_{2n+1}$, where f_n is the Fibonacci sequence.

In order to set up a bijection with the considered tiling of $\mathcal{Q}$, it is enough to label the edges of the tree, considered as a non-directed graph, in the following way: the label leading from a node to one of its sons is the number in $\{\mathbf{1},\mathbf{2},\mathbf{3},\mathbf{4},\mathbf{5}\}$ of the reflection transforming its associated pentagon into the pentagon associated to the considered son.

We can notice the following property of the numbering the first three rows of which are given by figure 3, which can be proved by induction on the height of the considered row: on odd rows numbers start with 4 and are decreasing; on even rows, they start with 1 and are increasing.

A bit more can be said : call *missing numbers* the numbers which are indicated over dotted edges in figure 3. They correspond to the reflection which directly transforms the pentagon associated to the above node to the pentagon associated to the below node as indicated in the figure by dotted edges. If the numbering of a row is mixed with the missing numbers above indicated and at the corresponding place, we obtain a regular sequence of numbers : a beginning of $(43215)^{+\infty}$ for odd rows and a beginning of $(12345)^{+\infty}$ for even rows.

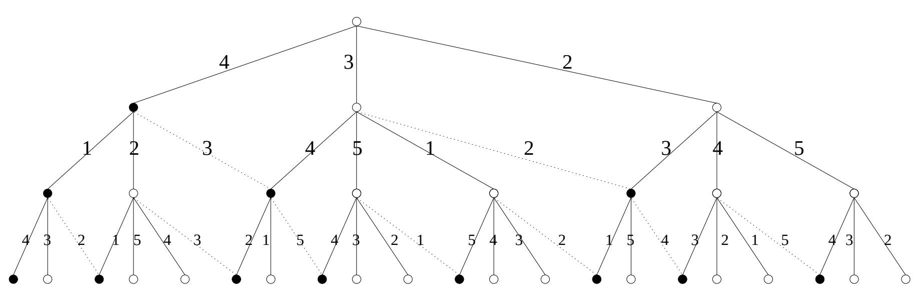

Figure 3 The Fibonacci tree with the numbering of its connections

Finally, notice that for all nodes, except those belonging to the rightmost and leftmost branches of the tree, root included, there are five edges connecting this node to others taking into account the *parental* edge and one or two possible doted edges, associated to missing numbers. Notice that for each node, the considered five edges are labeled from **1** up to **5** alternatively, in a clockwise and an anticlockwise way.

It is now not difficult to see that the tessellation defines a tiling of $\mathcal{Q}$. It is enough for that to see that any point of the interior of $\mathcal{Q}$ falls inside of at least one pentagon associated to the Fibonacci tree. This is made on the basis of an inductive argument using the described splitting of regions. Splitted regions are closer and closer to the unit circle and their diameter tends to zero. See [Margenstern et al. 2000] for a detailed proof.

2.4 Cellular automata in the pentagrid

Each quarter of the hyperbolic plane can be fitted with a similar Fibonacci tree and the numbering of edges in the one associated to $\mathcal{Q}$ naturally extends to the other quarters in a unique way, see [Margenstern et al. 2000] for details.

It may be assumed that such a representation is imbedded in the cellular automata as a 'hardware' feature: otherwise, a cell could not distinguish its neighbouring cells from each other. However, the corresponding information could be displayed in a 'software' way. Notice that the orientation of the numbering is changed by reflection, each cell receiving its orientation, provided the orientation of the leading pentagon of $\mathcal{Q}$ is defined. The distinction between neighbours is immediate as far as a cell knows its *father number* and its orientation. By definition, the root of a Fibonacci tree, later on called the *central cell*, has no father. A father number is nonetheless given to the central cell for determining the edges leading to its sons. It may be taken greater than 5 for signalizing the central cell.

This makes it possible for us to only use the Fibonacci trees for representing the cellular automata we shall construct in the next section for solving 3-SAT problem in polynomial time.

3 Solving 3-SAT in the pentagrid

We have now the tools for proving the main theorem which is a corollary of the following result:

Theorem 2 – *The problem* 3-*SAT can be solved in quadratic time by a* *cellular automaton on the pentagrid.*

The proof is split in two steps : first, taking advantage of the Fibonacci tree, we show how to delimitate a working space of size 2^n starting from a unary representation of n in $O(n)$ time. We then show how to adapt the algorithm for marking the exponential area into an algorithm for computing a solution of 3-SAT in quadratic time under the assumption that data are given with numbers written in unary representation. Then, by standard methods, we go then from a configuration using the binary representation of numbers into a configuration using the unary representation in polynomial time in the size of the initial representation. It will finally be possible to solve 3-SAT in time $O(k^2)$, assuming that k is the size of the data where numbers are written in the binary representation.

Within the frame of this extended abstract, we give only the main ideas used for proving the first step of the proof. Although the second step uses standard techniques, it must be checked that those techniques are applicable in the frame of our pentagrid. This is done in detail in [Margenstern et al. 2000].

3.1 A sketchy proof a theorem 2

We shall not here define 3-SAT problem, referring the reader to the abundant literature, see for instance [Garey and Johnsohn 79].

Intuitively, we presently know that the number of nodes with the same height in a Fibonacci tree is an exponential function of that height. This suggests that it is possible to mark an area of size 2^n in linear time in n.

Marking an exponential area

The idea is to spread the units arriving one by one to the central cell over all the branches of a binary tree $\mathcal{B}$ defined as follows. Tree $\mathcal{B}$ is rooted on the central cell and its nodes are recursively determined by taking 3-nodes only.

Let us describe that more precisely.

Assume that n, the exact number of variables, is given in unary, on the other side of **1** along **5**, starting from that neighbour of the central cell, represented by $\square$:

$$\_\,\square\circ\ldots\circ\,\_$$

where $\circ$ is the state of the corresponding cell, indicating a unit of the representation of n, and $\_$ is the quiescent state.

We shall split the marking process in two parts : first, the migration, one by one of the units constituting n from their initial position to the central cell. The transition table, below, on the left hand, rules that process, an example of which is given on the right side of the table:

_	□	∘	→	■	■	S	∘	→	•		□	∘	∘	∘	∘	_
□	∘	∘	→	S	S	∘	∘	→	S		■	S	∘	∘	∘	_
∘	∘	∘	→	∘	_	■	ξ	→	□		□	•	S	∘	∘	_
∘	∘	_	→	∘	_	□	•	→	■		■	_	•	S	∘	_
∘	_	_	→	_	□	•	S	→	_		□	•	_	•	_	_
•	S	∘	→	•	■	_	•	→	•		■	_	•	_	_	_
_	•	S	→	_	S	∘	_	→	_		□	•	_	_	_	_
□	•	_	→	_	•	_	•	→	•		■	_	_	_	_	_
_	•	_	→	_	•	_	_	→	_							

where ξ indicates any state.

The second process is controlled by the central cell according to the following pattern, reading from up to down, left column first:

■	_	_	_	_	_		□	•	∘	**□**	_	_
□	**□**	_	_	_	_		•	∘	•	**□**	_	_
•	**□**	_	_	_	_		□	•	∘	■	_	_
□	■	_	_	_	_		•	∘	•	∘	**□**	_
•	∘	**□**	_	_	_		□	•	∘	•	□	_
□	•	**□**	_	_	_		•	∘	•	∘	■	_
•	∘	■	_	_	_							

Finding the corresponding table is an exercise left to the reader, see [Margenstern et al. 2000].

That second process indicates what happens on each branch of $\mathcal{B}$. When a unit arrives in the central cell, it is sent on each subtree of $\mathcal{B}$. It reaches the current leaves, making the leaves go by one step higher and the new leaves wait until the next unit reaches them for repeating that process which is stopped when the central cell sends the last unit, also for signalizing the end of the process.

Evaluating clauses

The same process can be used for evaluating a clause : it is enough to encode the clause, say $y_\alpha \vee y_\beta \vee y_\gamma$, by at most n units with, at positions α, β and γ, a mark of that index, also indicating whether literal y is to be read x or $\neg x$:

_ • ∘ . . . ∘ • . . . • . . . □

where *bullet*'s are placed at positions α, β and γ.

The algorithm for marking an exponential area allows each above index, say ι, to find all nodes of height ι in the binary tree. Starting from each node of that height, the index signal sends value 0 on its left hand and value 1 on its right hand or conversely if the considered literal is a negative occurrence of the variable. And so, three values arrive at the leaves from which each leaf may compute the value of the clause under the assignation it represents. As the current value of the already computed part of the conjunction can also be stored by the leaf, it can also update the current value of the conjunction. When the last clause is computed, leaves send back their value to their fathers who compute the *or* of the sent values, until the central cell is reached, receiving the result.

It can be noticed that the *feed back* process can be a bit changed for giving an assignation providing value *true* if such an assignation exists.

For the conversion from binary to unary, notice that the number of nodes with height k in the binary tree is *embedded* in the Fibonacci tree where the number of the nodes of that height is greater: it is bounded by $M.2^{(3k)/2}$, M a positive constant. See [Margenstern et al. 2000] for more details.

3.2 Tractability of NP-problems on the pentagrid

We have only to check that the computation of a Turing machine checking in polynomial time that the guessed candidate is a solution of the considered problem can be simulated in our setting in a polynomial time with respect to the size of the initial configuration. It is the case, as shown in [Margenstern et al. 2000].

4 Further investigations

It seems to us that the main result of the paper draws our attention on the space environment in which we consider complexity problems, which, up to now, was not taken into account.

Considering the general definition of P and NP classes, let us add subscripts indicating m, the computational model in use, s, the considered space and k its dimension : (N)$\mathrm{P}_{m,s,k}$. Standard computations take place in (N)$\mathrm{P}_{t,e,1}$, t for *Turing machines*, $e, 1$ for *euclidean* line. It is not difficult to see that $\mathrm{P}_{t,e,1} = \mathrm{P}_{t,e,k}$ for all k's as well as $\mathrm{NP}_{t,e,1} = \mathrm{NP}_{t,e,k}$. Moreover, $\mathrm{P}_{ca,e,1} = \mathrm{P}_{ca,e,k} = \mathrm{P}_{t,e,1}$ and $\mathrm{NP}_{ca,e,1} = \mathrm{NP}_{ca,e,k} = \mathrm{NP}_{t,e,1} = \mathrm{NP}_{t,e,k}$. This justifies the notation $\mathrm{P} = \mathrm{P}_e$ and $\mathrm{NP} = \mathrm{NP}_e$, indicating a strong robustness of P and NP classes with respect to models and dimensions in the euclidean case.

Our result states that $\mathrm{NP}_e \subseteq \mathrm{P}_{ca,h,2}$, where h denotes the reference to hyperbolic geometry. Is that inclusion a strict one? Notice that $\mathrm{P}_{t,h,1} = \mathrm{P}_{ca,h,1} = \mathrm{P}_e$, and also $\mathrm{NP}_{t,h,1} = \mathrm{NP}_{ca,h,1} = \mathrm{NP}_e$. We may also ask whether the inclusions $\mathrm{P}_{ca,h,1} \subseteq \mathrm{P}_{ca,h,2}$, $\mathrm{NP}_{ca,h,1} \subseteq \mathrm{NP}_{ca,h,2}$ and $\mathrm{P}_{ca,h,2} \subseteq \mathrm{NP}_{ca,h,2}$ are strict or not. Our opinion is that all those three inclusions are strict. Another interesting question is the following : are inclusions $\mathrm{P}_e \subseteq \mathrm{NP}_e \subseteq \mathrm{P}_{ca,h,2}$ strict or not? We may have $\mathrm{P}_e \neq \mathrm{P}_{ca,h,2}$ and still $\mathrm{P}_e = \mathrm{NP}_e$. We think that it could be easier to prove $\mathrm{P}_e \neq \mathrm{P}_{ca,h,2}$ which could give a new starting point for trying to prove $\mathrm{P}_e \neq \mathrm{NP}_e$. Another interesting point is the difference between $(\mathrm{N})\mathrm{P}_{t,h,2}$ and $(\mathrm{N})\mathrm{P}_{ca,h,2}$. At first glance, our constructions does not work for $\mathrm{P}_{t,h,2}$.

Other directions of researches could also be investigated. First, it would be worth to construct an explicit cellular automaton for 3-SAT in our pentagrid. This could be done with a rather small number of states, possibly less than fifty ones. Other grids could also be investigated as far as our construction involving the Fibonacci tree can directly be generalized to the regular n-polygon with right angles, $n \geq 5$, as well as dual grids obtained by considering the graph joining the centers of the n-polygons. In the case of the pentagrid, we obtain a tiling with the square with $(2\pi)/5$ as the interior angle between consecutive sides. Also higher dimensions could be investigated.

This new setting could also provide a polynomial classification of NP_e: 3-SAT is at most quadratic (a better result is surely available), but it is possible that other NP-complete problems have higher h-polynomial complexity, or the same with another grid. This could also be an interesting field of investigations.

Acknowledgement

Both authors wishes to thank the Japanese Ministry of Education and the University of Metz for making possible their meetings without which that work would certainly have never been possible.

The first author thanks *INTAS* project 97-1259 that gave him the possibility to present the content of the paper at FWLA'99 workshop under FCT'99 conference in Iaşi, Romania, august, 24 - september, 3, 1999.

References

[Carathéodory 54] C. Carathéodory. Theory of functions of a complex variable, vol.II, 177–184, Chelsea, New-York, 1954.

[Garey and Johnsohn 79] M.R. Garey, D.S. Johnson. Computers and intractability: a guide to the theory of NP-completeness. W.H. Freeman and Co, San Francisco, 1979.

[Magnus 74] W. Magnus, Noneuclidean tessellations and their groups. Academic Press, New-York, 1974.

[Margenstern et al. 2000] Margenstern M., Morita K., NP problems are tractable in the space of cellular automata in the hyperbolic plane, accepted for publication in *TCS*, 37p.

[Maskit 71] Maskit. B. On Poincaré's theorem for fundamental polygons. *Advances in Math.*, **7**, 219–230, (1971).

[Meshkowski 64] H. Meschkowski, Noneuclidean geometry, translated by A. Shenitzer. Academic Press, New-York, 1964.

[Millman et al. 81] R.S. Millman, G.D. Parker, A metric approach with models. Springer-Verlag, New-York, 1981.

[Poincaré 1882] Poincaré H., Théorie des groupes fuchsiens. Acta Mathematica, **1**, 1–62, (1882).

[Robinson 78] Robinson R.M. Undecidable tiling problems in the hyperbolic plane. Inventiones Mathematicae, **44**, 259-264, (1978).