1 Algorithms of FO - First order logic

2 Model-checking

```
def model-checking:
        if \phi = R(x1,x2...,xk): //R(x1)-> symbol itself,
                if(x1^s,...xk^s) element of R^s:\\ R^s(x1^s) -> meaning of the 
symbol
                        return True
                else
                        return False
                elif phi=phi1 or phi2:
                        return Model-check(phi1,S) or Model-check(phi2,S)
                elif phi = phi1 and phi2:
                        ...

                elif phi= exists(x)\phi':
                for u elementof(U^s) do
                        if model-check(phi',S[x:=u]):
                                return True
                        else
                                return False
```

What is the complexity:
$|U^s| \cdot |\phi|$
PSPACE-complete

3 Satisfiability

Theorem (Trakhtenbrot 50') Satisfiability of FO - First order logic is undecidable
Proof: by **reduction** from **Domino** to FO - First order logic

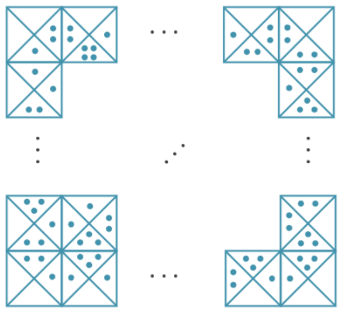
Idea: first we show that the **Halting-problem** is easier than **Domino** and **Domino** is easier that FO Satisfiability. Then we show that Satisfiability is undecidable in the **Halting-problem** from there follows that also in **Domino** and FO Satisfiability is undecidable.

reduction:
An algorithm F that solves Domino using an oracle that returns solutions to FO-Satisfiability

Domino-Problem
Also known as tiling problem
input:
finite set of 4-sided dominos:

The dominos can either have fields with numbers (dominos with the same number can be connected) or a white field that is used for the border of the rectangle we want to lay.

Goal: to lay a complete rectangle without holes with the dominos of any size with a **white border**



Special rules:

- one does not need to use all the dominos of the set
- One can reuse dominos of the set

This problem is undecidable! No computer can solve it **unless** we fix the size of the rectangle.

Fun fact: this problem occurs often in literature as one can define a lot of different variants that cover (almost) all complexity classes.

1. NP: One has to tile a square with fixed size n
2. PSPACE: If one has to tile a corridor fixed width n but arbitrary height m
3. Undecidable class: Arbitrary height and length

Lemma 7:
The Domino-Problem/tiling problem is undecidable (by reduction from halting)

Proof: By reduction: we need to find a function that transforms the input of the halting problem into an input

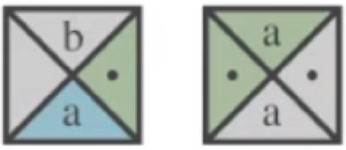
Given Turing machine T with 3 states:
What do the colors mean:
The colors denote the control states of the machine:
blue : initial state (for instance q_0)
green : second state (for instance q_1)
red : holding state (for instance q_h)

What do the dots mean:
The dot means that the tile represents a writable cell. If there is a white field it means that the tape ends there.

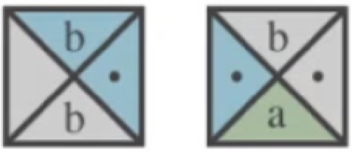
What do the tiles mean:

- The head is else where the tiles are copied into the next like this into the next row.

- The read/write head is over the left tile. The state of the square of tape is changed from a to b and the read/write head is moved to the right. The state changes from blue (q_0) to green(q_1)



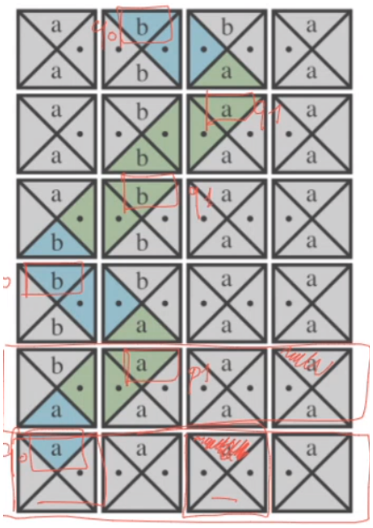
- The read/write head is over the right tile. The state of the square of tape is changed from a to b and the read/write head is moved to the left. The state changes from green (q_1) to blue (q_0).



- Initial state

- halting state

A full Turing machine reduction looks then like this:



After converting from Turing machine to Domino-Problem we now need to transform the Domino problem to A FO - First order logic formula.

i.e.
Given a set of dominos so that:
$\phi \text{satisfiable} \iff \text{dominos can be arranged in a white bordered rectangle}$

The formula should describe the correct tiling of the rectangle.

$$S: \begin{cases} U^S & = \{\text{All possible positions where a domino could lie}\} \\ H(-,-)^S & = \{\text{Describes if two tiles are next to each other horizontally}\} \\ V(-,-)^S & = \{\text{Describes if two tiles are next to each other verrtically}\} \\ N_d(-)^S & = \{\text{Describes the } d^{th} \text{ position of a tile}\} \end{cases}$$

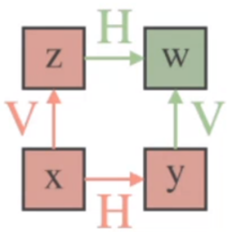
Special sets:

MH the set of all tiles that can match together horizontally
MV the set of all tiles that can match together vertically

1. ϕ_1=*There is a grid"*
   Relations $H(-,-)$ and $V(-,-)$ are partial bijections such that:

$$\forall x, y, z \quad (H(x,y) \wedge V(x,z)) \implies \exists w \quad (H(z,w) \wedge (V(y,w)))$$

Meaning: If there is a tile at the position x and a tile z above it and a tile y to the right of it. There must be another tile w that is to the right of z and above y



2. ϕ_2 =*There is exactly one domino at each node*
   For every domino d we have a relation N_d describing its position for which holds

$$\forall x (\bigvee_d N_d(x)) \wedge \ldots$$

meaning: Every domino d has a position with $N_d(x)$ describing the position.

$$\ldots \wedge (\bigwedge_{d \neq d'} \neg(N_d(x) \wedge N_{d'}(x)))$$

meaning: For each domino it is not possible that it lies at two positions N_d and $N_{d'}$

3. ϕ_3=*Sides of adjacent dominos need to match (have the same numbers)*

$$\forall x, y (H(x,y) \implies \bigvee_{(d,d') \in MH} N_d(x) \wedge N_{d'}(y) \wedge \quad \ldots$$

meaning: When two tiles x and y are next to each other horizontally (H(-,-) relation) then the dominos need to be part of the set MH which contains all dominos which can fit together horizontally.

$$\ldots \quad (V(x,y) \implies \bigvee_{(d,d') \in VH} N_d(x) \wedge N_{d'}(y)$$

meaning: When two tiles x and y are next to each other horizontally (H(-,-) relation) then the dominos need to be part of the set MH which contains all dominos which can fit together horizontally.

Important : MH and MV are not *relationship symbols* but sets of dominos.

4. ϕ_4 =*Borders are white*
   exercise!
   WH -> dominos containing white fields on their horizontal parts
   WV -> dominos containing white fields on their vertical parts

   1. **Horizontal case:**
      $\forall x((\exists y H(x,y) \land (\forall z H(x,z) \implies z = y)) \implies \bigvee_{d \in WH} N_d(x))$
      meaning: if a domino is at a position and has only one neighbor $(y = z)$ it means that this domino must have white fields at the horizontal edges.

   2. **Vertical case**
      $\forall x((\exists y V(x,y) \land (\forall z V(x,z) \implies z = y)) \implies \bigvee_{d \in WH} N_d(x))$
      meaning: if a domino is at a position and has only one neighbor $(y = z)$ it means that this domino must have white fields at the horizontal edges.

Now we have to combine the 4 subformulas to one big one:
If there exists a Domino that fulfills all the requirements stated in the formulas

$\exists x(\phi_1 \land \phi_2 \land \phi_3 \land \phi_4)$

Summary : We know that the halting problem in the Turing machine is undecidable. Further we know that the turing machine can be reduced to the domino problem and the domino problem can be reduced to a satisfiability problem in FO - First order logic.

Recap:

	Propositional Logic	QBF	FO - First order logic
Model-checking	**P**	PSPACE	PSPACE
Satisfiability	**NP**	PSPACE	Undecidable
Validity	**coNP**	PSPACE	Undecidable
logical equivalence	**coNP**	PSPACE	undecidable

Why does the Undecidability of Satisfiability imply that Validity is Undecidable.

Validity can be reduced to Satisfiability i.e.

Satisfiability has this definition $\exists S \quad S \models \phi$
Validity has this definition $\forall S \quad S \models \phi$

$\forall S \quad S \models \phi$ is logically equivalent to:
meaning: all S model ϕ
$\forall S \quad S \not\models \neg\phi$ which is logically equivalent to:
meaning: All S don't models $\neg\phi$
$\neg(\exists S \quad S \models \neg\phi)$ which is a satisfiability problem
meaning: There does not exist a single S that models $\neg\phi$

why is equivalence also undecidable?

We can reduce it from Validity
We have two formulas ϕ_1 and ϕ_2
then we define a formula $\phi_3 = \phi_1 \iff \phi_2$
Then we do a validity check on ϕ_3
$\forall S \quad S \models \phi_3$
Therefore as Validity is undecidable also logical equivalence is undecidable.

4 FO-theories

Change of perspective:
Before: we were given a formula and had to check if it holds on a structure.
Now : We have a given structure what are the formulas that hold on that Structure

We are given a Structure
Theory $FO[U^S, R^S, \ldots, x^S, ..] =$ set of formulas ϕ that hold on the structure $S = (U^S, R^S, \ldots, x^S)$

$U^S =$ Universe
$R^S, \ldots =$ all relational Symbols
$x^S, \ldots =$ definition of free variables

4.1 Examples

Note: $' =' $ is always in the signature

4.1.1

$$FO[\mathbb{N}, <]$$

Usecase: is the structure with which one describes discrete,linear time.
Contains the formulas:

$$\exists x(x = x)$$

meaning: tautology, maybe there is equality?

$$\forall x \exists y(x < y)$$

meaning: There follows a y after every x. Way to define infinity

$$\exists y \forall x \neg(x < y)$$

meaning: workaround to not define $>$ -> there is not just always a element bigger but also a element smaller than x

$$\forall xy(x = y \lor x < y \lor y < x) \ldots$$

meaning: the element is either equal,greater or smaller than x

4.1.2

$$\text{Presburg arithmetic: } FO[\mathbb{N}, +]$$

4.1.3

$$\text{Peano arithmetic: } FO[\mathbb{N}, +, \cdot]$$

4.1.4

$$\text{Tarski arithmetic: } FO[\mathbb{R}, +, \cdot]$$

4.1.5

$$FO[\text{RadoGraph}]$$