Algorithms and Complexity of Constraint Satisfaction Problems (course number 2)

Nicolas (Miki) Hermann

LIX, École Polytechnique

hermann@lix.polytechnique.fr
Goal

Given a relation $M \subseteq \{0, 1\}^k$, construct a formula $\varphi(x_1, \ldots, x_k)$ in CNF, such that $[\varphi] = M$.

Construction of a clause for a vector

Let $m = (m[1], \ldots, m[k])$ be a boolean vector. We construct the clause $c_m = (l_1 \lor \cdots \lor l_k)$ in the following way. For each $i = 1, \ldots, k$

$$l_i = \begin{cases} x_i & \text{si } m[i] = 0, \\ \neg x_i & \text{si } m[i] = 1. \end{cases}$$

It is easy to see that $[c_m] = \{0, 1\}^k \setminus \{m\}$. 

Construction of a formula $\varphi_m$ in CNF such that $[\varphi_m] = M$

1. Let $\bar{M} = \{0, 1\}^k \setminus M$.
2. For each $m \in \bar{M}$ we construct the clause $c_m$.
3. The formula $\varphi_M$ is the conjunction of clauses $c_m$ for each $m \in \bar{M}$, i.e.,

$$\varphi_M = \bigwedge_{m \in \bar{M}} c_m$$

Analysis

Each clause $c_m$ is falsified by the corresponding vector $m \in \bar{M}$. The formula $\varphi_M$ is falsified by the set of vectors $\bar{M}$. If a vector $m' \in M$ falsifies $\varphi_M$, there must be a falsified clause $c_{m'}$. This implies $m' \in \bar{M}$, but $M \cap \bar{M} = \emptyset$. Hence, each vector from $M$ satisfies $\varphi_M$. Therefore $[\varphi_M] = M$. 

Miki Hermann  
Algorithms and Complexity of CSP (2)
Exercise 3

The aforementioned algorithm is exponential with respect to the arity $k$. Construct another algorithm, running in time polynomial in $|M|$ and $k$. Good solution: algorithm running in $O(|M|k^2)$

Research problem 1
Are you able to construct an algorithm with an inferior asymptotic complexity?
If YES, Come to see me!

Research problem 2
Are you able to prove that the asymptotic lower bound is $\Omega(|M|k^2)$?
If YES, come to see me immediately!
Special CNF formulas

Special clauses

We say that a clause is
- **Horn** if it contains at most one positive literal;
- **dual Horn** if it contains at most one negative literal;
- **bijunctive** if it contains at most two literals;
- **affine** if it can be written as a linear equation modulo 2.

Special formulas

A formula $\varphi = c_1 \land \cdots \land c_p$ is called **Horn**, **dual Horn**, **bijunctive** or **affine** if each clause $c_i$ is **Horn**, **dual Horn**, **bijunctive** or **affine**, respectively.

Attention

The Horn, dual Horn, bijunctive and affine formulas are always in **CNF**.
### Different types of clauses

- $c_1 = (\neg x \lor \neg y \lor z)$ is a **Horn** clause, but it is neither dual Horn, nor bijunctive, nor affine;
- $c_2 = (\neg x \lor y \lor z)$ is **dual Horn**;
- $c_3 = (\neg x \lor y)$ is **bijunctive**, **Horn** and **dual Horn**;
- $c_4 = (x \lor y)$ is **bijunctive** and **dual Horn**, but not **Horn**;
- $c_5 = (x + y + z = 1) \mod 2$ is **affine**;
- $c_6 = (x \equiv y)$ is **affine**, **bijunctive**, **Horn** and **dual Horn**;
- $c_7 = (x \not\equiv y)$ is **affine** and **bijunctive**
Problem \textbf{HornSat}

\textit{Input:} A set of variables $V$ and a Horn formula $\varphi$ over $V$.

\textit{Question:} Is the formula $\varphi$ satisfiable?

Theorem

\textbf{HornSat} \textit{is decidable in polynomial time}.
**Algorithm for HornSat**

1. While the Horn formula $\varphi$ contains unit clauses.
   1. Let $\varphi = l \land \varphi'$ where $l$ is a unit clause.
   2. If $l = x$ then set $m(x) := 1$ else set $m(x) := 0$.
   3. Set $\varphi := \varphi'[x \leftarrow m(x)]$ and simplify $\varphi$, i.e., apply
      
      
      $0 \lor x \rightarrow x$,  
      $1 \lor x \rightarrow 1$,  
      $0 \land x \rightarrow 0$  
      and  
      $1 \land x \rightarrow x$.
   4. If $\varphi = 0$ then return $0$.

2. For each variable $x$ of $\varphi$, set $m(x) := 0$ and return $1$.

**Remark 1**

Note that at the end of the while-loop the formula $\varphi$ does not contain unit clauses any more and therefore each clause $c$ of $\varphi$ contains at least one negative literal.

**Remark 2**

This method is called unit resolution + propagation.
Algorithm analysis for **HORN Sat**

- Each **Horn** formula is hereditary. After substitution of a value 0 or 1 for a variable $x$, followed by simplification, the resulting formula is still Horn.
- A unary clause $c = l$, composed of one literal $l = x$ or $l = \neg x$, determines the satifying assignment $m(x)$ for the variable $x$.
- If a Horn formula $\varphi = c_1 \land \cdots \land c_p$ does not contain unary clauses, the assignment $m(x) = 0$ for each variable $x$ satisfies $\varphi$, because each clause $c_i$ contains at least one negative literal.
**Problem** \textsc{DualHornSat} \\

\textbf{Input:} A set of variables \( V \) and a dual Horn formula \( \varphi \) over \( V \).

\textbf{Question:} Is the formula \( \varphi \) satisfiable?

**Lemma**

The formula \( \varphi(x_1, \ldots, x_k) \) is \textit{Horn} if and only if the formula \( \varphi(\neg x_1, \ldots, \neg x_k) \) is \textit{dual Horn}.

**Theorem**

\textsc{DualHornSat} is decidable in polynomial time.

**Proof.**

The vector \( m = (m[1], \ldots, m[k]) \) satisfies the formula \( \varphi(x_1, \ldots, x_k) \) if and only if \( \neg m = (\neg m[1], \ldots, \neg m[k]) \) satisfies \( \varphi(\neg x_1, \ldots, \neg x_k) \). \( \square \)
### Problem 2SAT

**Input:** A set of variables $V$ and a bijunctive formula $\varphi$ over $V$.

**Question:** Is the formula $\varphi$ satisfiable?

### Theorem

2SAT is decidable in polynomial time.
Algorithm for 2SAT (binary resolution method)

1. Let $C$ be the set of clauses of the formula $\varphi$.
2. While the set $C$ is not saturated, apply the following rules:

   - $[\text{res}]$ $(l \lor x) \land (\neg x \lor l') \rightarrow (l \lor l')$
   - $[\text{fct}]$ $(l \lor l) \rightarrow l$
   - $[\text{rft}]$ $x \land \neg x \rightarrow \bot$

   and add the new clauses to the set $C$.
3. If $\bot \notin C$ then $\varphi$ is satisfiable, else $\varphi$ is invalid.
   ($\bot \in C$ means refutation of $\varphi$).
For a (finite) set of variables $V$ there is only $4|V|^2$ bijunctive clauses. Hence there are only $O(|V|^2)$ possible repetitions of the saturation loop.

The resolution (rule $res$) applied to bijunctive clauses produces only bijunctive clauses.
Satisfiability of affine formulas

**Problem** \textsc{AffineSat}

*Input:* A set of variables $V$ and a affine formula $\varphi$ over $V$.

*Question:* Is the formula $\varphi$ satisfiable?

**Reformulation of the Problem** \textsc{AffineSat}

*Input:* A system of affines linear equations $A\vec{x} = b$ over $\mathbb{Z}_2$ (also denoted as $\mathbb{Z}/2\mathbb{Z}$ or $\mathbb{F}_2$).

*Question:* Does the system $A\vec{x} = b$ have a solution?

**Theorem**

\textsc{AffineSat} is decidable in polynomial time.

**Proof.**

The solution of the affine system $A\vec{x} = b$ is determined by Gaussian elimination.
Problem $2\text{COL}$

**Input:** A graph $G = (V, E)$ with vertices $V$ and edges $E$.

**Question:** Is there a function $f : V \to \{0, 1\}$ such that $f(x) \neq f(y)$ if $(x, y) \in E$?

Analysis

Relation $2\text{col} = \{01, 10\}$ produces the affine constraint

$$2\text{col}(x, y) = (x + y = 1) \mod 2$$

Hence for each edge $(x, y) \in E$ we have the constraint $(x + y = 1)$. This produces the affine system of binary linear equations $A\vec{x} = b$ over $\mathbb{Z}_2$, whose solution is determined by Gaussian elimination in polynomial time.
### Complexity of boolean satisfiability

3SAT is **NP**-complete.

**HornSat, DualHornSat, 2sat, and AffineSat** are in **P**.

### Question 1

Are there other polynomial cases?

### Answer

**YES**, but they are hard to discover on the formula level.

### Question 2

Are there in **NP** cases which are neither polynomial, nor **NP**-complete, provided that **P** \( \neq **NP**)?

### Answer

**NO**, but we do not have the tools yet to discover it.
We will not distinguish between the relation $R \subseteq \{0, 1\}^k$ and the corresponding constraint $R(x_1, \ldots, x_k)$, because they are just two aspects of the same structure.
New constraints from old ones

Construction of relations (1)

We saw that

\[
\begin{align*}
or_0 &= [x \lor y \lor z] = \{0, 1\}^3 \setminus \{000\} \\
or_3 &= [\neg x \lor \neg y \lor \neg z] = \{0, 1\}^3 \setminus \{111\} \\
ae &= [x \lor y \lor z] \cap [\neg x \lor \neg y \lor \neg z] = \{0, 1\}^3 \setminus \{000, 111\}
\end{align*}
\]  

Construction of relations (2)

Therefore we can write

\[
nae(x, y, z) = or_0(x, y, z) \land or_3(x, y, z)
\]

on the level of constraints or

\[
nae = [x \lor y \lor z] \cap [\neg x \lor \neg y \lor \neg z]
\]

on the level of relations.
### Question

If we have a set of constraints $S = \{or_0, or_3\}$, how to construct the constraints $or_1$ and $or_2$?

### Knowledge

We have the identities

\[
\begin{align*}
or_1(x, y, z) &= or_0(x, y, \neg z) \\
or_2(x, y, z) &= or_3(\neg x, y, z)
\end{align*}
\]

but we have neither the negation available in $S$, nor the possibility to simply place a constraint in the argument of another one, i.e. we cannot compose constraints.
Intermediate constructions

First we construct the following binary constraints:

\[ \text{bor}_0(x, y) = \text{or}_0(x, y, y) = (x \lor y) \]
\[ \text{bor}_2(x, y) = \text{or}_3(x, y, y) = (\neg x \lor \neg y) \]

Inequality

We can now construct the inequality constraint

\[ \text{neq}(x, y) = \text{bor}_0(x, y) \land \text{bor}_2(x, y) \]
\[ = (x \lor y) \land (\neg x \lor \neg y) = (x \neq y) \]

Combination

We can combine now the constraints \text{neq}_2 \text{ or } \text{or}_0, \text{ or } \text{neq}_2 \text{ and } \text{or}_3, \text{ to construct } \text{or}_1 \text{ or } \text{or}_2:

\[ \text{or}_1(x, y, z) = \text{or}_0(x, y, \neg z) = \exists v \text{ neq}(z, v) \land \text{or}_0(x, y, v) \]
\[ \text{or}_2(x, y, z) = \text{or}_3(\neg x, y, z) = \exists v \text{ neq}(x, v) \land \text{or}_3(v, y, z) \]
The two previous slides prove the following proposition.

**Theorem**

\[ \text{CSP}(or_0, or_3) = \text{CSP}(or_0, or_1, or_2, or_3) = 3\text{SAT} \]

**Corollary**

\[ \text{CSP}(or_0, or_3) \text{ is NP-complete.} \]
How to construct negation (2)

Intermediate construction

Suppose that we have the relation

\[ nae = \{001, 010, 011, 100, 101, 110\}. \]

We can easily produce the constraint

\[ neq(x, y) = nae(x, y, y) \]

New construction

We can now produce the following constraints

\[
\begin{align*}
nae_1(x, y, z) &= nae(x, y, \neg z) &= \exists v \ neq(z, v) \land nae(x, y, v) \\
nae_2(x, y, z) &= nae(x, \neg y, \neg z) &= \exists v \ neq(y, v) \land nae_1(x, v, z) \\
nae_3(x, y, z) &= nae(\neg x, \neg y, \neg z) &= \exists v \ neq(x, v) \land nae_2(v, y, z)
\end{align*}
\]
Complexity of new CSPs

Problem \textbf{NAE3SAT}

\textit{Input:} A set of variables $V$ and a formula $\varphi = c_1 \land \cdots \land c_p$ in CNF over $V$ with exactly three literals per clause.

\textit{Question:} Is there an assignment $m$ to $\varphi$ interpreting in each clause $c_i$ at least one literal to 1 and at least one literal to 0?

Theorem (Schaefer 1978)

\textbf{NAE3SAT} is \textbf{NP-complete}.

Corollary

\textbf{CSP}(nae) is \textbf{NP-complete}.
Let \( S \) a finite set of relations not necessarily of same arity. The set \( \langle S \rangle \) contains all constraints which can be constructed from the relations in \( S \). The set \( \langle S \rangle \) is constructed by saturation:

### Saturation rules for \( \langle S \rangle \)

**Introduction:** If \( R \in S \) and \( ar(R) = k \) then \( R(x_1, \ldots, x_k) \in \langle S \rangle \).

**Permutation:** If \( R(x_1, \ldots, x_k) \in \langle S \rangle \) and \( \pi \) is a permutation of \( \{1, \ldots, k\} \) then \( R(x_{\pi(1)}, \ldots, x_{\pi(k)}) \in \langle S \rangle \).

**Diagonalisation:** If \( R(x_1, \ldots, x_{k-1}, x_k) \in \langle S \rangle \) and
\[
R'(x_1, \ldots, x_{k-1}) = R(x_1, \ldots, x_{k-1}, x_k) \quad \text{then} \quad R'(x_1, \ldots, x_{k-1}) \in \langle S \rangle.
\]

**Conjunction:** If \( R_1(x), R_2(y) \in \langle S \rangle \) then \( R_1(x) \wedge R_2(y) \in \langle S \rangle \).

**Quantification:** If \( R(x_1, \ldots, x_{k-1}, x_k) \in \langle S \rangle \) and \( R'(x_1, \ldots, x_{k-1}) = \exists x_k \ R(x_1, \ldots, x_{k-1}, x_k) \) then \( R'(x_1, \ldots, x_{k-1}) \in \langle S \rangle \).
Primitive positive formulas

Constraints are constructed from a set of relations $S$ by conjunction and existential quantification. This construction has a name.

**Definition (Primitive positive formula)**

A formula $\exists y_1, \ldots, y_n \varphi(x_1, \ldots, x_k, y_1, \ldots, y_n)$ constructed from a set of relations $S$, conjunction $\wedge$, and existential quantification $\exists$ is called **primitive positive**, also denoted as **pp-formula**.
Co-clones and complexity

Implementation

If $R \in \langle S \rangle$ we say that the set of relations $S$ implements the constraint $R$.

Definition

Let $S$ be a set of relations. The saturated set of constraints $\langle S \rangle$ is called a co-clone.

Theorem

The problems $\text{CSP}(S)$ and $\text{CSP}(\langle S \rangle)$ are polynomially equivalent, i.e., there exists a polynomial-time reduction from $\text{CSP}(S)$ to $\text{CSP}(\langle S \rangle)$, as well as from $\text{CSP}(\langle S \rangle)$ to $\text{CSP}(S)$.

Proof.

The reduction from $\text{CSP}(S)$ to $\text{CSP}(\langle S \rangle)$ is trivial (injection). The reduction from $\text{CSP}(\langle S \rangle)$ to $\text{CSP}(S)$ is the consequence of the construction of the co-clone $\langle S \rangle$. 

□
Co-clones and complexity

**Theorem**

Let $S_1$ and $S_2$ be two sets of relations, such that $S_1 \subseteq \langle S_2 \rangle$. Then there exists a polynomial reduction from $\text{csp}(S_1)$ to $\text{csp}(S_2)$.

**Corollary**

Let $S_1$ and $S_2$ be two sets of relations, such that $S_1 \subseteq \langle S_2 \rangle$.

- If $\text{csp}(S_1)$ is NP-complete then $\text{csp}(S_2)$ is also NP-complete.
- If $\text{csp}(S_2)$ is in P then $\text{csp}(S_1)$ is also in P.
Exercise 4

Known relations

We have the relations

\[
\text{or}_0 = \left[ x \lor y \lor z \right], \quad \text{or}_3 = \left[ \neg x \lor \neg y \lor \neg z \right], \\
\text{bor}_0 = \left[ x \lor y \right], \quad \text{bor}_2 = \left[ \neg x \lor \neg y \right], \\
\text{1-in-3} = \{100, 010, 001\}, \quad \text{nae} = \{0, 1\}^3 \setminus \{000, 111\}.
\]

Question

Which of the following implementations are correct? Construct them if they are correct.

1. \(\text{1-in-3} \in \langle \text{or}_0, \text{or}_3 \rangle\)
2. \(\text{or}_0 \in \langle \text{1-in-3} \rangle \) and \(\text{or}_3 \in \langle \text{1-in-3} \rangle\)
3. \(\text{or}_0 \in \langle \text{bor}_0, \text{bor}_2 \rangle \) and \(\text{or}_3 \in \langle \text{bor}_0, \text{bor}_2 \rangle\)
4. \(\text{nae} \in \langle \text{1-in-3} \rangle\)
5. \(\text{1-in-3} \in \langle \text{nae} \rangle\)
0-valid constraints

**Definition**

A relation \( R \) is called **0-valid** if it contains the vector \( 0 \cdots 0 \). A set of relations \( S \) is called **0-valid** if each relation \( R \in S \) is 0-valid. A constraint \( R(x_1, \ldots, x_k) \) is **0-valid** if the corresponding relation \( R \) is 0-valid.

**0-validity is hereditary**

If \( S \) is 0-valid then each constraint \( R \in \langle S \rangle \) is 0-valid, i.e., the 0-validity is a **hereditary** property.

**Theorem**

*If \( S \) is a set of **0-valid** relations then \( \text{CSP}(S) \) is decidable in polynomial time.*

**Proof.**

The all-zero vector \( 0 \cdots 0 \) is always a solution.
1-valid contraints

Duality with 0-valid constraints.

Definition
A relation $R$ is called **1-valid** if it contains the vector $1 \cdots 1$. A set of relations $S$ is called **1-valid** if each relation $R \in S$ is 1-valid. A constraint $R(x_1, \ldots, x_k)$ is **1-valid** if the corresponding relation $R$ is 1-valid.

1-validity is hereditary
If $S$ is 1-valid then each constraint $R \in \langle S \rangle$ is 1-valid, i.e., the 1-validity is a **hereditary** property.

Theorem
*If $S$ is a set of 1-valid relations then $\text{CSP}(S)$ is decidable in polynomial time.*

Proof.
The all-one vector $1 \cdots 1$ is always a solution.
**Definition**

A relation $R$ is called **Horn** if there exists a Horn formula $\varphi$ such that $[\varphi] = R$. A set of relations $S$ is called **Horn** if each relation $R \in S$ is Horn. A constraint $R(x_1, \ldots, x_k)$ is Horn if the corresponding relation $R$ is Horn.

**Horn is hereditary**

If $S$ is Horn then each constraint $R \in \langle S \rangle$ is Horn, i.e., the property of being Horn is hereditary.
Theorem

If $S$ is a set of Horn relations then $\text{CSP}(S)$ is decidable in polynomial time.

Proof.

If $S$ is Horn the set of constraints $C = \{R_1(\vec{x}_1), \ldots R_p(\vec{x}_p)\}$ corresponds to a Horn formula $\varphi_C = \varphi_1 \land \cdots \land \varphi_p$, where $[\varphi_i] = R_i$ for each $i$. There exists a polynomial-time algorithm to decide $\text{HORN$_\text{SAT}$}$. 

\hfill \Box
Solution of Exercise 2, 3rd part

Answer

The question was if $3SAT = CSP(or_2, or_3)$. The answer is NO, since

$$or_2(x, y, z) = (x \lor \neg y \lor \neg z),$$
$$or_3(x, y, z) = (\neg x \lor \neg y \lor \neg z).$$

Given that the set of relations $\{or_2, or_3\}$ is Horn, but for instance the relations $nae$ or $1$-in-$3$ are not Horn, we have that $3SAT \neq CSP(or_2, or_3)$.

There is a small problem

How do we know that the relations $nae$ or $1$-in-$3$ are not Horn?

Answer

We do not have yet the necessary tools to prove it.
Duality with Horn constraints.

**Definition**

A relation $R$ is called dual Horn if there exists a dual Horn formula $\varphi$ such that $[\varphi] = R$. A set of relations $S$ is called dual Horn if each relation $R \in S$ is dual Horn. A constraint $R(x_1, \ldots, x_k)$ is dual Horn if the corresponding relation $R$ is dual Horn.

**Dual Horn is hereditary**

If $S$ is dual Horn then each constraint $R \in \langle S \rangle$ is dual Horn, i.e., the property of being dual Horn is hereditary.
**Theorem**

If $S$ is a set of dual Horn relations then $\text{CSP}(S)$ is decidable in polynomial time.

**Proof.**

If $S$ is dual Horn then the set $C = \{R_1(\vec{x}_1), \ldots R_p(\vec{x}_p)\}$ corresponds to a dual Horn formula $\varphi_C = \varphi_1 \land \cdots \land \varphi_p$, where $[\varphi_i] = R_i$ for each $i$. There exists a polynomial-time algorithm to decide $\text{DualHornSat}$. \qed
Solution of Exercise 2, 1st part

Answer

The question was if $3\text{SAT} = \text{CSP}(or_0, or_1)$. The answer is NO, since

$$or_0(x, y, z) = (x \lor y \lor z),$$

$$or_1(x, y, z) = (x \lor y \lor \neg z).$$

Given that the set of relations $\{or_0, or_1\}$ is dual Horn, but for example the relations $nae$ or 1-in-3 are not dual Horn, we have that $3\text{SAT} \neq \text{CSP}(or_0, or_1)$.

There is a small problem

How do we know that the relations $nae$ or 1-in-3 are not dual Horn?

Answer

We do not have yet the necessary tools to prove it.
Definition

A relation \( R \) is **bijunctive** if there exists a bijunctive formula \( \varphi \) such that \([\varphi] = R\). A set of relations \( S \) is called **bijunctive** if each relation \( R \in S \) is bijunctive. A constraint \( R(x_1, \ldots, x_k) \) is **bijunctive** if the corresponding relation is bijunctive.

Bijunctive is hereditary

Sif \( S \) is **bijunctif** then each constraint \( R \in \langle S \rangle \) is **bijunctive**, i.e., the property of being bijunctive is **hereditary**.
2SAT as CSP

**Theorem**

If $S$ is a set of bijunctive relations then $\text{CSP}(S)$ is decidable in polynomial time.

**Proof.**

If $S$ is bijunctive then the set $C = \{R_1(x_1), \ldots R_p(x_p)\}$ corresponds to a bijunctive formula $\varphi_C = \varphi_1 \land \cdots \land \varphi_p$, where $[\varphi_i] = R_i$ for each $i$. There exists a polynomial-time algorithm to decide 2SAT.
**Definition**

A relation $R$ is **affine** if there exists an affine formula $\varphi$ such that $[\varphi] = R$. A set of relations $S$ is called **affine** if each relation $R \in S$ is affine. A constraint $R(x_1, \ldots, x_k)$ is **affine** if the corresponding relation is affine.

**Affinity is hereditary**

If $S$ is affine then each constraint $R \in \langle S \rangle$ is affine, i.e., the property of being affine is **hereditary**.
Theorem

If $S$ is a set of affines relations then $\text{CSP}(S)$ is decidable in polynomial time.

Proof.

If $S$ is affine then the set of contraintes $C = \{R_1(\vec{x}_1), \ldots R_p(\vec{x}_p)\}$ corresponds to a affine formula $\varphi_C = \varphi_1 \land \cdots \land \varphi_p$, where $[\varphi_i] = R_i$ for each $i$. There exists a polynomial algorithm to decide $\text{AFFINESAT}$.  \qed
Recapitulation

The problems $\text{CSP}(S)$ where the set of relations $S$ has the property of being 0-valid, 1-valid, Horn, dual Horn, bijunctive, or affine respectively, are in $P$.

Convention

Instead of writing $\text{CSP}(S)$ for each set of relations $S$ with the property of being $X$, we will write $\text{CSP}(X)$. For example: $\text{CSP}(\text{Horn})$.

Question

Are there other polynomial cases of $\text{CSP}$?

Answer

NO, but we do not have yet the tools to prove it.
It’s all for today.
Do you have questions?