# *Online structures*

Keng Meng Ng

Nanyang Technological University, Singapore

25 March 2019

# Introduction

# Motivating questions

* Study how computation interacts with various mathematical concepts.
* Complexity of constructions and objects we use in mathematics (how to calibrate?)
* Can formalize this more syntactically (reverse math, etc).
* Or more model theoretically...

# Motivating questions

* Study how computation interacts with various mathematical concepts.
* Complexity of constructions and objects we use in mathematics (how to calibrate?)
* Can formalize this more syntactically (reverse math, etc).
* Or more model theoretically...

# Motivating questions I: Presentations

- In computable model / structure theory, can different effective concepts
  - presentations of a structure,
  - complexity of isomorphisms within an isomorphism type,
  - investigations can descend into a more degree-theoretic approach.
- Classically $\mathcal{A}$ and $\mathcal{B}$ are considered the same if $\mathcal{A} \cong \mathcal{B}$.
- However, from an effective point of view, even if $\mathcal{A} \cong \mathcal{B}$ are computable, they may have very different “hidden" effective properties.
- Standard example: $(\omega, <) \cong \mathcal{A}$ where you arrange for $2n$ and $2n + 2$ to be adjacent in $\mathcal{A}$ iff $n \notin \emptyset'$.

# Motivating questions I: Presentations

* In computable model / structure theory, can different effective concepts
  * presentations of a structure,
  * complexity of isomorphisms within an isomorphism type,
  * investigations can descend into a more degree-theoretic approach.
* Classically $\mathcal{A}$ and $\mathcal{B}$ are considered the same if $\mathcal{A} \cong \mathcal{B}$.
* However, from an effective point of view, even if $\mathcal{A} \cong \mathcal{B}$ are computable, they may have very different "hidden" effective properties.
* Standard example: $(\omega, <) \cong \mathcal{A}$ where you arrange for $2n$ and $2n + 2$ to be adjacent in $\mathcal{A}$ iff $n \notin \emptyset'$.

# Motivating questions I: Presentations

* In computable model / structure theory, can different effective concepts
  * presentations of a structure,
  * complexity of isomorphisms within an isomorphism type,
  * investigations can descend into a more degree-theoretic approach.
* Classically $\mathcal{A}$ and $\mathcal{B}$ are considered the same if $\mathcal{A} \cong \mathcal{B}$.
* However, from an effective point of view, even if $\mathcal{A} \cong \mathcal{B}$ are computable, they may have very different "hidden" effective properties.
* Standard example: $(\omega, <) \cong \mathcal{A}$ where you arrange for $2n$ and $2n+2$ to be adjacent in $\mathcal{A}$ iff $n \notin \emptyset'$.

# Motivating questions I: Presentations

* In computable model / structure theory, can different effective concepts
  * presentations of a structure,
  * complexity of isomorphisms within an isomorphism type,
  * investigations can descend into a more degree-theoretic approach.
* Classically $\mathcal{A}$ and $\mathcal{B}$ are considered the same if $\mathcal{A} \cong \mathcal{B}$.
* However, from an effective point of view, even if $\mathcal{A} \cong \mathcal{B}$ are computable, they may have very different “hidden" effective properties.
* Standard example: $(\omega, <) \cong \mathcal{A}$ where you arrange for $2n$ and $2n+2$ to be adjacent in $\mathcal{A}$ iff $n \notin \emptyset'$.

# Motivating questions II: Complexity of Isomorphisms

* In the standard example $(\omega, <) \cong \mathcal{A}$, “successivity" was the hidden property. Any isomorphism must transfer all definable properties, so this says that...

**Definition**

A computable structure $\mathcal{A}$ is computably categorical if for every computable $\mathcal{B} \cong \mathcal{A}$, there is a computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

* Aim of the project: Systematic approach to all these considerations, with even stricter / finer effective restrictions.

# Motivating questions II: Complexity of Isomorphisms

- In the standard example $(\omega, <) \cong \mathcal{A}$, “successivity" was the hidden property. Any isomorphism must transfer all definable properties, so this says that...

### Definition

A computable structure $\mathcal{A}$ is computably categorical if for every computable $\mathcal{B} \cong \mathcal{A}$, there is a computable isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

- Aim of the project: Systematic approach to all these considerations, with even stricter / finer effective restrictions.

# Computable structure theory

## Definition (Mal'cev, Rabin, 60's)

A structure is computable if it's domain and all operations and relations are uniformly computable.

* Equivalent variations (allow domain to be computable or c.e.).
* Seen to unify all earlier effective algebraic concepts, e.g. explicitly presented fields, recursively presented group with solvable word problem, etc.
* This has grown since into a large body of research;

  groups, fields, Boolean algebras, linear orders, model theory, reverse mathematics.

# Computable structure theory

* Our investigation is to place even finer restrictions:

## Question

*When does a computable structure have a feasible presentation?*

* One obvious way: structure presented by a *finite automaton* (we won't discuss here).
* This talk will be centered around the notion of online computability (1960's).
* *Online situation*: Input arrives one bit at a time, but decision has to be made instantly.
* *Offline situation*: Decision made only after seeing the entire (but finite) input.

# Computable structure theory

* Our investigation is to place even finer restrictions:

## Question

*When does a computable structure have a feasible presentation?*

* One obvious way: structure presented by a *finite automaton* (we won't discuss here).
* This talk will be centered around the notion of online computability (1960's).
* *Online situation*: Input arrives one bit at a time, but decision has to be made instantly.
* *Offline situation*: Decision made only after seeing the entire (but finite) input.

# Computable structure theory

- Our investigation is to place even finer restrictions:

## Question

*When does a computable structure have a feasible presentation?*

- One obvious way: structure presented by a *finite automaton* (we won't discuss here).
- This talk will be centered around the notion of online computability (1960's).
- *Online situation*: Input arrives one bit at a time, but decision has to be made instantly.
- *Offline situation*: Decision made only after seeing the entire (but finite) input.

# Practical online algorithms

Scheduling problem: Given *k* identical machines, and a sequence of jobs arriving. We must schedule each arrived job immediately without knowledge of future jobs.

Bin packing: Given *k* bins and a sequence of objects of different sizes arriving, pack each item immediately while minimizing number of bins used. Greedy algorithm is good, but not optimal. Decision problem is *NP*-complete.

Ski rental problem: Go skiing for an unknown number of days, each day we must decide to rent or buy the skis. Optimal (deterministic) online strategy: Break even strategy.

# Practical online algorithms

Scheduling problem: Given $k$ identical machines, and a sequence of jobs arriving. We must schedule each arrived job immediately without knowledge of future jobs.

Bin packing: Given $k$ bins and a sequence of objects of different sizes arriving, pack each item immediately while minimizing number of bins used. Greedy algorithm is good, but not optimal. Decision problem is *NP*-complete.

Ski rental problem: Go skiing for an unknown number of days, each day we must decide to rent or buy the skis. Optimal (deterministic) online strategy: Break even strategy.

# Practical online algorithms

Secretary problem: Interview a number of candidates for a job, must immediately decide to hire or reject after each interview. Optimal online strategy: Reject the first $\frac{n}{e}$ candidates.

Bandit problem: A gambler at a row of slot machines, decide to continue playing the current machine (exploitation) or try a different machine (exploration). Example of stochastic scheduling, considered by Allied scientists.

# Practical online algorithms

Secretary problem: Interview a number of candidates for a job, must immediately decide to hire or reject after each interview. Optimal online strategy: Reject the first $\frac{n}{e}$ candidates.

Bandit problem: A gambler at a row of slot machines, decide to continue playing the current machine (exploitation) or try a different machine (exploration). Example of stochastic scheduling, considered by Allied scientists.

# Practical online algorithms

Online graph colouring:

Vertices of a finite (or infinite) graph arrives one at a time, and the induced subgraph is shown to us immediately.

A colour has to be assigned immediately, and cannot be changed.

Minimize the number of colours used.

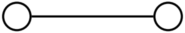
For every $k$ there is a tree with $2^k$ vertices that cannot be online-coloured in $< k$ colours.

# Practical online algorithms

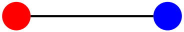
## Online graph colouring:

- Vertices of a finite (or infinite) graph arrives one at a time, and the induced subgraph is shown to us immediately.
- A colour has to be assigned immediately, and cannot be changed.
- Minimize the number of colours used.
- For every $k$ there is a tree with $2^k$ vertices that cannot be online-coloured in $< k$ colours.

# No online 2-colouring



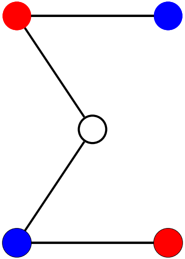
# No online 2-colouring



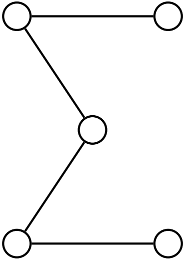
# No online 2-colouring

# No online 2-colouring

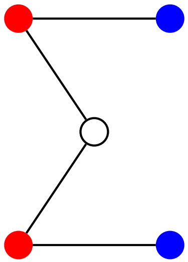
# No online 2-colouring



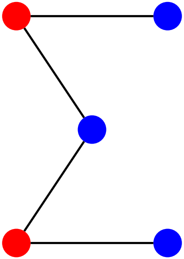
# Can be 2-coloured offline



# Can be 2-coloured offline



# Can be 2-coloured offline



# What does “online” mean for an infinite structure?

# Capturing online nature of infinite structures

* In the examples mentioned above, we had to make a decision *immediately*.
* It is of course, perfectly fine to wait for 100 more steps. But how much more?
* An obvious formalization: polynomial time structures (Cenzer, Remmel, Downey).
  * This depends on how the domain is represented (as $\mathbb{N}$ or $2^{<\omega}$).
  * This leads to an entire hierarchy of different notions of being online.

# Capturing online nature of infinite structures

- In the examples mentioned above, we had to make a decision *immediately*.
- It is of course, perfectly fine to wait for 100 more steps. But how much more?
- An obvious formalization: polynomial time structures (Cenzer, Remmel, Downey).
  - This depends on how the domain is represented (as $\mathbb{N}$ or $2^{<\omega}$).
  - This leads to an entire hierarchy of different notions of being online.

# Capturing online nature of infinite structures

- What is the most general notion of online computability? Obviously, Turing computability is too weak.
- A computable infinite tree has a computable 2-colouring. Wait for a node to be connected to the root.
- The "unbounded search" nature of a general recursive operation is what allows this.
- The general model we adopt for online computation is based on being primitive recursive.

# Capturing online nature of infinite structures

- What is the most general notion of online computability? Obviously, Turing computability is too weak.
- A computable infinite tree has a computable 2-colouring. Wait for a node to be connected to the root.
- The “unbounded search” nature of a general recursive operation is what allows this.
- The general model we adopt for online computation is based on being primitive recursive.

# Poly-time versus primitive recursive

* Again, there's a large body of work (80's) done on polynomial time (mostly) algebras.
* Our starting point is a series of papers of Cenzer, Remmel (and other co-authors), on various classes of "feasible" structures.
* In computable structures we allow algorithms to be extremely inefficient.
* Sometimes, every computable structure has a polynomial-time copy:

  Linear orders, certain kinds of BAs, some commutative groups.

# Poly-time versus primitive recursive

* Again, there's a large body of work (80's) done on polynomial time (mostly) algebras.
* Our starting point is a series of papers of Cenzer, Remmel (and other co-authors), on various classes of "feasible" structures.
* In computable structures we allow algorithms to be extremely inefficient.
* Sometimes, every computable structure has a polynomial-time copy:

  Linear orders, certain kinds of BAs, some commutative groups.

# Poly-time versus primitive recursive

* A problematic version using primitive recursion:

**Definition (Mal'cev, Rabin, 60's)**

A structure is computable if it's domain and all operations and relations are uniformly computable.

**Definition (Cenzer, Remmel)**

A structure is primitive recursive if it's domain and all operations and relations are primitive recursive.

* *Does not capture online nature*: In a primitive recursive structure, new elements can be enumerated very slowly.
* (Alaev) Every computable locally finite structure has a primitive recursive copy.

# Poly-time versus primitive recursive

* A problematic version using primitive recursion:

**Definition (Mal'cev, Rabin, 60's)**

A structure is computable if it's domain and all operations and relations are uniformly computable.

**Definition (Cenzer, Remmel)**

A structure is primitive recursive if it's domain and all operations and relations are primitive recursive.

* *Does not capture online nature*: In a primitive recursive structure, new elements can be enumerated very slowly.
* (Alaev) Every computable locally finite structure has a primitive recursive copy.

# Poly-time versus primitive recursive

* A problematic version using primitive recursion:

**Definition (Mal'cev, Rabin, 60's)**

A structure is computable if it's domain and all operations and relations are uniformly computable.

**Definition (Cenzer, Remmel)**

A structure is primitive recursive if it's domain and all operations and relations are primitive recursive.

* *Does not capture online nature*: In a primitive recursive structure, new elements can be enumerated very slowly.
* (Alaev) Every computable locally finite structure has a primitive recursive copy.

# Capturing online nature of infinite structures

* We want the definition of an "online structure" to have no possible way to delay revealing the structure:

**Definition (Kalimullin, Melnikov, N)**

A structure is punctual if it has domain $\mathbb{N}$, and all operations and relations are primitive recursive.

* Intuition: Punctual structures have to decide right away what to do with the next element.
* We only consider finite languages.
* Already used by Cenzer and Remmel as a technical tool.
* The goal is to initiate a systematic study of punctuality (online) versus computable (offline).

# Capturing online nature of infinite structures

* We want the definition of an "online structure" to have no possible way to delay revealing the structure:

**Definition (Kalimullin, Melnikov, N)**

A structure is punctual if it has domain $\mathbb{N}$, and all operations and relations are primitive recursive.

* Intuition: Punctual structures have to decide right away what to do with the next element.
* We only consider finite languages.
* Already used by Cenzer and Remmel as a technical tool.
* The goal is to initiate a systematic study of punctuality (online) versus computable (offline).

# Considerations

* We can place effectivity on math structures in several ways. In the same vein, we can ask:

**Question (1)**

*When does a computable structure have a punctual copy?*

**Question (2)**

*How many punctual copies does a punctual structure have, up to punctual isomorphisms?*

* We contrast to the computable case; often different, sometimes even unclear.
* Measures the “online" nature of a computable structure.

# Considerations

* We can place effectivity on math structures in several ways. In the same vein, we can ask:

### Question (1)

*When does a computable structure have a punctual copy?*

### Question (2)

*How many punctual copies does a punctual structure have, up to punctual isomorphisms?*

* We contrast to the computable case; often different, sometimes even unclear.
* Measures the “online" nature of a computable structure.

# Considerations

* We can place effectivity on math structures in several ways. In the same vein, we can ask:

### Question (1)

*When does a computable structure have a punctual copy?*

### Question (2)

*How many punctual copies does a punctual structure have, up to punctual isomorphisms?*

* We contrast to the computable case; often different, sometimes even unclear.
* Measures the “online" nature of a computable structure.

Which structure has a punctual presention?

# When does a structure have a punctual copy?

Theorem (Kalimullin, Melnikov, N)

*Each computable structure in the following classes has a punctual copy:*

- *Equivalence structures,*
- *linear orders,*
- *torsion-free abelian groups,*
- *boolean algebras,*
- *abelian p-groups.*

Proof.

Each of these structures $\mathcal{A}$ has an infinite local part $\mathcal{B} \subset \mathcal{A}$ that is very simple, and trivially related to the elements of $\mathcal{A} - \mathcal{B}$. Allows us to simulate arbitrary finite delay. □

# When does a structure have a punctual copy?

## Theorem (Kalimullin, Melnikov, N)

*Each computable structure in the following classes has a punctual copy:*

* *Equivalence structures,*
* *linear orders,*
* *torsion-free abelian groups,*
* *boolean algebras,*
* *abelian p-groups.*

## Proof.

Each of these structures $\mathcal{A}$ has an infinite local part $\mathcal{B} \subset \mathcal{A}$ that is very simple, and trivially related to the elements of $\mathcal{A} - \mathcal{B}$.

Allows us to simulate arbitrary finite delay. □

# When does a structure have a punctual copy?

* The classes above have a “online" basis of some sort, used for simulating arbitrary finite delay. However, merely having a basis is insufficient for having a punctual copy:

Theorem (Cenzer, Remmel, KMN)

*There is a computable torsion abelian group with no punctual copy.*

Question

* *Find a reasonable sufficient condition for a computable structure to have a punctual copy.*
* *E.g. formalize the notion of a punctual basis.*

# When does a structure have a punctual copy?

- The classes above have a “online" basis of some sort, used for simulating arbitrary finite delay. However, merely having a basis is insufficient for having a punctual copy:

Theorem (Cenzer, Remmel, KMN)

*There is a computable torsion abelian group with no punctual copy.*

Question

- *Find a reasonable sufficient condition for a computable structure to have a punctual copy.*
- *E.g. formalize the notion of a punctual basis.*

# When does a structure have a punctual copy?

* We turn to pure relational languages.

**Fact**

*Every computable locally finite graph has a punctual copy.*

* Converse is not true, for example the random graph and the infinite star have punctual copies.
* Perhaps every computable graph has a punctual copy.

**Theorem (Kalimullin, Melnikov, N)**

*There is a computable graph with no punctual copy.*

# When does a structure have a punctual copy?

* We turn to pure relational languages.

### Fact

*Every computable locally finite graph has a punctual copy.*

* Converse is not true, for example the random graph and the infinite star have punctual copies.
* Perhaps every computable graph has a punctual copy.

### Theorem (Kalimullin, Melnikov, N)

*There is a computable graph with no punctual copy.*

# When does a structure have a punctual copy?

Is there a natural description of which computable structures have punctual copies? Unfortunately,

Theorem (Bazhenov, Harrison-Trainor, Kalimullin, Melnikov, N)

*The following index sets are $\Sigma^1_1$-complete:*

$\{e : M_e$ *is computable and has a punctual copy*$\}$.

$\{e : M_e$ *is computable and has an automatic copy*$\}$.

$\{e : M_e$ *is computable and has an polynomial-time copy*$\}$.

# When does a structure have a punctual copy?

Is there a natural description of which computable structures have punctual copies? Unfortunately,

Theorem (Bazhenov, Harrison-Trainor, Kalimullin, Melnikov, N)

*The following index sets are $\Sigma^1_1$-complete:*

$\{e : M_e$ *is computable and has a punctual copy*$\}$.

$\{e : M_e$ *is computable and has an automatic copy*$\}$.

$\{e : M_e$ *is computable and has an polynomial-time copy*$\}$.

# The number of punctual presentations

# Punctual categoricity

* Recall that the complexity of a computable structure can be measured by the minimal complexity of isomorphisms between computable copies.

## Definition

A punctual structure $\mathcal{A}$ is punctually categorical if for every punctual $\mathcal{B} \cong \mathcal{A}$ there is a punctual isomorphism $f : \mathcal{A} \mapsto \mathcal{B}$.

* What does a "punctual isomorphism" mean?

  "$f$ and $f^{-1}$ are both primitive recursive."

* Warning: This is different from saying that "$f : \mathcal{A} \mapsto \mathcal{B}$ and $g : \mathcal{B} \mapsto \mathcal{A}$ for some primitive recursive $f, g$", or saying that "$\mathrm{Graph}(f)$ is primitive recursive"..
* For computable isomorphisms, these are all equivalent.

# Punctual categoricity

* Recall that the complexity of a computable structure can be measured by the minimal complexity of isomorphisms between computable copies.

## Definition

A punctual structure $\mathcal{A}$ is punctually categorical if for every punctual $\mathcal{B} \cong \mathcal{A}$ there is a punctual isomorphism $f : \mathcal{A} \mapsto \mathcal{B}$.

* What does a "punctual isomorphism" mean?

  "$f$ and $f^{-1}$ are both primitive recursive."

* Warning: This is different from saying that "$f : \mathcal{A} \mapsto \mathcal{B}$ and $g : \mathcal{B} \mapsto \mathcal{A}$ for some primitive recursive $f, g$", or saying that "$\mathrm{Graph}(f)$ is primitive recursive"..
* For computable isomorphisms, these are all equivalent.

# Punctual categoricity: Examples

1. The additive group $\bigoplus_{i\in\omega} \mathbb{Z}_p$ is punctually categorical.
   - Given a punctual copy $\mathcal{A}$, some $a \in \mathcal{A}$, and some $S \subseteq \mathcal{A}$, it is primitive recursive to check if $a$ is linearly independent over $S$.
   - An online back-and-forth argument works.
2. The dense linear order $(\mathbb{Q}, <)$ is surprisingly not punctually categorical.
   - An online back-and-forth argument does not work.
   - Given $p < q$ an element $r \in (p, q)$ might not arrive quickly.
3. The structure $(\omega, \mathrm{Succ})$ is also not punctually categorical.
   - Given an element $n$, its distance to 0 might not be primitive recursive.

# Punctual categoricity: Examples

1. The additive group $\bigoplus_{i\in\omega} \mathbb{Z}_p$ is punctually categorical.
   - Given a punctual copy $\mathcal{A}$, some $a \in \mathcal{A}$, and some $S \subseteq \mathcal{A}$, it is primitive recursive to check if $a$ is linearly independent over $S$.
   - An online back-and-forth argument works.
2. The dense linear order $(\mathbb{Q}, <)$ is surprisingly not punctually categorical.
   - An online back-and-forth argument does not work.
   - Given $p < q$ an element $r \in (p, q)$ might not arrive quickly.
3. The structure $(\omega, \mathrm{Succ})$ is also not punctually categorical.
   - Given an element $n$, its distance to 0 might not be primitive recursive.

# Punctual categoricity: Examples

1. The additive group $\bigoplus_{i\in\omega} \mathbb{Z}_p$ is punctually categorical.
   * Given a punctual copy $\mathcal{A}$, some $a \in \mathcal{A}$, and some $S \subseteq \mathcal{A}$, it is primitive recursive to check if $a$ is linearly independent over $S$.
   * An online back-and-forth argument works.
2. The dense linear order $(\mathbb{Q}, <)$ is surprisingly not punctually categorical.
   * An online back-and-forth argument does not work.
   * Given $p < q$ an element $r \in (p, q)$ might not arrive quickly.
3. The structure $(\omega, \mathrm{Succ})$ is also not punctually categorical.
   * Given an element $n$, its distance to 0 might not be primitive recursive.

# Punctual categoricity: Examples

## Theorem (KMN)

*In each of the following classes, a structure is punctually categorical if and only if it is "trivial".*

- *Equivalence structures: only classes of size 1, or finitely many classes at most one of which is infinite.*
- *Linear orders: finite.*
- *Boolean algebras: finite.*
- *Abelian p-groups: $pG = 0$.*
- *Torsion-free abelian groups: trivial group $\{0\}$.*

# Punctual categoricity and rigidity

* The examples of punctually categorical structures we've seen so far were far from rigid ($\oplus\mathbb{Z}_p$, equivalence structures). What about rigid structures?

## Theorem (KMN)

* *There is a rigid functional structure which is not punctually categorical* $(\omega, \mathrm{Succ})$*.*
* *There is a rigid functional structure which is punctually categorical.*
* *However, rigid relational structures are never punctually categorical.*

# Comparing punctual and computable categoricity

* We saw that $(\omega, \mathrm{Succ})$ is an example of a computably categorical but not punctually categorical structure.
* A very natural conjecture would be that every punctually categorical structure is computably categorical.
* This is true for many natural classes (equivalence structures, linear orders, Boolean algebras, abelian *p*-groups, TFAGs).

**Theorem (KMN)**

*There is a punctually categorical structure which is not computably categorical.*

**Theorem (In progress)**

*There is a punctually categorical structure $\mathcal{A}$ where every isomorphism between computable copies of $\mathcal{A}$ compute $\emptyset''$.*

# Comparing punctual and computable categoricity

* We saw that $(\omega, \mathrm{Succ})$ is an example of a computably categorical but not punctually categorical structure.
* A very natural conjecture would be that every punctually categorical structure is computably categorical.
* This is true for many natural classes (equivalence structures, linear orders, Boolean algebras, abelian $p$-groups, TFAGs).

### Theorem (KMN)

*There is a punctually categorical structure which is not computably categorical.*

### Theorem (In progress)

*There is a punctually categorical structure $\mathcal{A}$ where every isomorphism between computable copies of $\mathcal{A}$ compute $\emptyset''$.*

# Comparing punctual and computable categoricity

* We saw that $(\omega, \mathrm{Succ})$ is an example of a computably categorical but not punctually categorical structure.
* A very natural conjecture would be that every punctually categorical structure is computably categorical.
* This is true for many natural classes (equivalence structures, linear orders, Boolean algebras, abelian *p*-groups, TFAGs).

## Theorem (KMN)

*There is a punctually categorical structure which is not computably categorical.*

## Theorem (In progress)

*There is a punctually categorical structure $\mathcal{A}$ where every isomorphism between computable copies of $\mathcal{A}$ compute $\emptyset''$.*

# Graphs and universality

It is well-known that graphs are universal for computable structures.

Theorem (Downey, Harrison-Trainor, Kalimullin, Melnikov, Turetsky)

*Graphs are not universal for punctual structures.*
*Indeed, a graph $\mathcal{G}$ is punctually categorical if and only if there are $v_0, \cdots, v_n$ such that $\mathcal{G} - \{v_0, \cdots, v_n\}$ is a clique or an anti-clique and each $v_i$ is adjacent to all or none of $\mathcal{G} - \{v_0, \cdots, v_n\}$.*

# Comparing the online content between two punctual structures

# Comparing online content

- If $\mathcal{A}$ and $\mathcal{B}$ are punctual copies of the same structure, what should $\mathcal{A} \leq_{pr} \mathcal{B}$ mean?
- $\mathcal{B}$ has more online content than $\mathcal{A}$.
- We say that $\mathcal{A} \leq_{pr} \mathcal{B}$ if there is a primitive recursive isomorphism $f : \mathcal{A} \xrightarrow{\text{onto}} \mathcal{B}$.
- This is merely a preordering (since $f^{-1}$ is not always p.r.)
- Let **FPR**($\mathcal{A}$) denote $\{$all punctual copies of $\mathcal{A}\}/\equiv_{pr}$.
- The standard copy of $(\mathbb{Q}, <)$ is the greatest element of **FPR**$(\mathbb{Q}, <)$
- The standard copy of $(\mathbb{N}, \mathrm{Succ})$ is the least element of **FPR**$(\mathbb{N}, \mathrm{Succ})$.

# Comparing online content

- If $\mathcal{A}$ and $\mathcal{B}$ are punctual copies of the same structure, what should $\mathcal{A} \leq_{pr} \mathcal{B}$ mean?
- $\mathcal{B}$ has more online content than $\mathcal{A}$.
- We say that $\mathcal{A} \leq_{pr} \mathcal{B}$ if there is a primitive recursive isomorphism $f : \mathcal{A} \xrightarrow{\text{onto}} \mathcal{B}$.
- This is merely a preordering (since $f^{-1}$ is not always p.r.)
- Let **FPR**$(\mathcal{A})$ denote $\{$all punctual copies of $\mathcal{A}\}/\equiv_{pr}$.
- The standard copy of $(\mathbb{Q}, <)$ is the greatest element of **FPR**$(\mathbb{Q}, <)$
- The standard copy of $(\mathbb{N}, \mathrm{Succ})$ is the least element of **FPR**$(\mathbb{N}, \mathrm{Succ})$.

# Comparing online content

- If $\mathcal{A}$ and $\mathcal{B}$ are punctual copies of the same structure, what should $\mathcal{A} \leq_{pr} \mathcal{B}$ mean?
- $\mathcal{B}$ has more online content than $\mathcal{A}$.
- We say that $\mathcal{A} \leq_{pr} \mathcal{B}$ if there is a primitive recursive isomorphism $f : \mathcal{A} \xrightarrow{\text{onto}} \mathcal{B}$.
- This is merely a preordering (since $f^{-1}$ is not always p.r.)
- Let **FPR**($\mathcal{A}$) denote $\{$all punctual copies of $\mathcal{A}\}/\equiv_{pr}$.
- The standard copy of $(\mathbb{Q}, <)$ is the greatest element of **FPR**$(\mathbb{Q}, <)$
- The standard copy of $(\mathbb{N}, \mathrm{Succ})$ is the least element of **FPR**$(\mathbb{N}, \mathrm{Succ})$.

# Online back-and-forth

* If $|\mathbf{FPR}(\mathcal{A})| = 1$ then all punctual copies of $\mathcal{A}$ have the same online content. Is this enough to carry out an online back-and-forth argument?

## Theorem (Melnikov,N)

*A graph $\mathcal{G}$ is punctually categorical if and only if $|\boldsymbol{FPR}(\mathcal{G})| = 1$.*

## Question

*Is $|\boldsymbol{FPR}(\mathcal{A})| = 1$ equivalent to saying that $\mathcal{A}$ is punctually categorical?*

# A degree-theoretic approach

* One could potentially approach this degree-theoretically:

**Theorem (In progress)**

*For every finite n, there is a structure $\mathcal{A}$ such that $|\textbf{FPR}(\mathcal{A})| = n$.*

**Question**

*What other partial orders can be realized as $\textbf{FPR}(\mathcal{A})$ for some $\mathcal{A}$? For instance, infinite linear orders? All countable distributive lattices?*

# Online content of homogeneous structures

* Consider the following homogeneous structures:
  * $(\mathbb{Q}, <)$,
  * The random graph $\mathcal{R}$,
  * The universal countable abelian $p$-group $\mathcal{P} \cong \bigoplus_{i \in \omega} \mathbb{Z}_{p^\infty}$,
  * The countable atomless Boolean algebra $\mathcal{B}$.
* In the computable setting, they are all the same, in that they share the same back-and-forth proof, and they are the Fraisse limit of all finite structures.
* Strangely, their online contents are quite different.

Theorem (Melnikov, N)

***FPR***$(\mathbb{Q}, <)$, ***FPR***$(\mathcal{R})$ *and* ***FPR***$(\mathcal{P})$ *are pairwise non-isomorphic.*

# Online content of homogeneous structures

* Consider the following homogeneous structures:
  * $(\mathbb{Q}, <)$,
  * The random graph $\mathcal{R}$,
  * The universal countable abelian $p$-group $\mathcal{P} \cong \bigoplus_{i \in \omega} \mathbb{Z}_{p^\infty}$,
  * The countable atomless Boolean algebra $\mathcal{B}$.
* In the computable setting, they are all the same, in that they share the same back-and-forth proof, and they are the Fraisse limit of all finite structures.
* Strangely, their online contents are quite different.

### Theorem (Melnikov, N)

***FPR***$(\mathbb{Q}, <)$*,* ***FPR***$(\mathcal{R})$ *and* ***FPR***$(\mathcal{P})$ *are pairwise non-isomorphic.*

# Finitely generated structures

**Question**

*Is **FPR**$(\mathbb{Q}, <)$ and **FPR**$(\mathcal{B})$ isomorphic (as partial orders)?*

**Question**

*Study the local structure of, say, **FPR**$(\mathbb{Q}, <)$.*

- Recall that $(\omega, \mathit{Succ})$ is not punctually categorical. The generalization of this is to consider finitely generated structures in a finite functional language.

**Theorem (Bazhenov, Kalimullin, Melnikov, N)**

*Suppose $\mathcal{A}$ is finitely generated. Then **FPR**$(\mathcal{A})$ is dense.*

# Finitely generated structures

### Question

*Is **FPR**$(\mathbb{Q}, <)$ and **FPR**$(\mathcal{B})$ isomorphic (as partial orders)?*

### Question

*Study the local structure of, say, **FPR**$(\mathbb{Q}, <)$.*

* Recall that $(\omega, Succ)$ is not punctually categorical. The generalization of this is to consider finitely generated structures in a finite functional language.

### Theorem (Bazhenov, Kalimullin, Melnikov, N)

*Suppose $\mathcal{A}$ is finitely generated. Then **FPR**$(\mathcal{A})$ is dense.*

# Finitely generated structures

### Question

*Is **FPR**$(\mathbb{Q}, <)$ and **FPR**$(\mathcal{B})$ isomorphic (as partial orders)?*

### Question

*Study the local structure of, say, **FPR**$(\mathbb{Q}, <)$.*

- Recall that $(\omega, \mathit{Succ})$ is not punctually categorical. The generalization of this is to consider finitely generated structures in a finite functional language.

### Theorem (Bazhenov, Kalimullin, Melnikov, N)

*Suppose $\mathcal{A}$ is finitely generated. Then **FPR**$(\mathcal{A})$ is dense.*

# Finitely generated structures

## Question

*Is **FPR**$(\mathbb{Q}, <)$ and **FPR**$(\mathcal{B})$ isomorphic (as partial orders)?*

## Question

*Study the local structure of, say, **FPR**$(\mathbb{Q}, <)$.*

- Recall that $(\omega, \mathit{Succ})$ is not punctually categorical. The generalization of this is to consider finitely generated structures in a finite functional language.

## Theorem (Bazhenov, Kalimullin, Melnikov, N)

*Suppose $\mathcal{A}$ is finitely generated. Then **FPR**$(\mathcal{A})$ is dense.*

# Questions

* Connection with definability, Scott sentences. Note: online back-and-forth works differently.
* How can we define being relatively punctually categorical?
* Develop online model theory.
* Measure the complexity of the index set $\{e : M_e$ is punctually categorical$\}$.
* More work to be done on relativization, which will lead to investigations like spectra questions, degrees of categoricity, etc.
* Thank you.

# Questions

* Connection with definability, Scott sentences. Note: online back-and-forth works differently.
* How can we define being relatively punctually categorical?
* Develop online model theory.
* Measure the complexity of the index set $\{e : M_e \text{ is punctually categorical}\}$.
* More work to be done on relativization, which will lead to investigations like spectra questions, degrees of categoricity, etc.
* Thank you.