Computably t.d.l.c. groups

André Nies

Joint work with Alexander Melnikov

Conference on computational aspects of t.d.l.c. groups,
Newcastle, Australia, October 2022

THE UNIVERSITY OF AUCKLAND
NEW ZEALAND

The main questions

(A) How can one define a computable presentation of a t.d.l.c. group? Which t.d.l.c. groups have such a presentation?

(B) Given a computable presentation of a t.d.l.c. group, are objects such as the rational valued Haar measures, the modular function, or the scale function computable?

(C) Do constructions that lead from t.d.l.c. groups to new t.d.l.c. groups have algorithmic versions?

(D) When is a computable presentation of a t.d.l.c. group unique up to computable isomorphism?

Talk 1 mainly addresses Question A

- We will introduce two notions of computable presentation of a t.d.l.c. group G, and show their equivalence.
- The first notion relies on standard notions of computability in the uncountable setting.
- The second notion restricts computation to a countable structure of approximations of the elements, the “meet groupoid” of compact open cosets.
- Based on these, we obtain various examples of computably t.d.l.c. groups.
- The first talk also outlines some computability theoretic notions needed.

Talk 2 mainly addresses Question B-D

- We show that given a computable presentation of a t.d.l.c. group G, the modular function and the Cayley-Abels graphs (in the compactly generated case) are computable.
- We discuss algorithmic properties of the scale function on G, but leave open whether it can be non-computable.
- We explain why the class of computably t.d.l.c. groups is closed under most of the constructions studied by Wesolek (2015).
- We will give a criterion based on meet groupoids when the computable presentation is unique up to computable isomorphism.

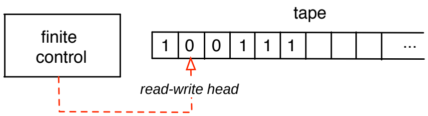
Examples of computably t.d.l.c. groups

- All computable profinite groups and all computable discrete groups
- $(\mathbb{Q}_p, +)$, the additive group of p-adic numbers for a prime p
- The semidirect product $\mathbb{Z} \ltimes \mathbb{Q}_p$ where $g \in \mathbb{Z}$ acts as $x \mapsto xp$ on $\mathbb{Q}_p$
- The groups $\mathrm{SL}_n(\mathbb{Q}_p)$ for $n \geq 2$
- $Aut(T_d)$, the automorphisms of a homogeneous undirected tree of degree d.

By convention, all t.d.l.c. groups will be separable and infinite.

Computable functions

Picture of a Turing machine with one tape:



Definition

Given a set $S \subseteq \mathbb{N}^k$, where $k \geq 1$, a function $f\colon S \to \mathbb{N}$ is called computable if there is a Turing machine that on inputs $n_1, \ldots, n_k$ decides whether the tuple of inputs $(n_1, \ldots, n_k)$ is in S, and if so outputs $f(n_1, \ldots, n_k)$.

One version of the Church-Turing thesis states that computability in this sense is the same as being computable by some algorithm.

Computable structures: the countable case

A structure in the model theoretic sense consists of a nonempty set D, called the domain, with relations and functions defined on it.

Definition (Mal'cev and Rabin independently, 1960s)

- A computable structure is a structure such that the domain is a computable set $D \subseteq \mathbb{N}$, and the functions and relations of the structure are computable.
- A countable structure S is called computably presentable if some computable structure W is isomorphic to it. In this context we call W a computable copy of S.

Example

For each $k \geq 1$, the group $GL_k(\mathbb{Q})$ is computably presentable.

To obtain a computable copy:

- fix an algorithmic encoding of the rational $k \times k$ matrices by natural numbers
- let the domain D be the computable set of numbers that encode a matrix with nonzero determinant.
- The matrix operations are computable.

Computable structures: the uncountable case

- computable analysis represent all the elements by "names"
- Let $\mathbb{N}^*$ denote the tree of strings with natural number entries. Names usually are elements of the set $[T]$ of paths on some computable subtree T of $\mathbb{N}^*$.
- For instance, as a name for a real number r, take a path coding a sequence of rationals $\langle q_n \rangle_{n \in \mathbb{N}}$ such that
$$\forall n\, |q_n - q_{n+1}| \leq 2^{-n} \text{ and } \lim_n q_n = r.$$
- Such names are directly accessible to computation of Turing machines with tapes that hold the infinite inputs.
- One can now define computability of functions and relations on $[T]$: one requires that they are computable on the names.
- E.g. $\exp\colon \mathbb{R} \to \mathbb{R}$ is computable.

Ad hoc ways to define computability for some uncountable structures

The following sometimes works for particular classes of uncountable structures: impose algorithmic constraints on the definition of the class.

An example is the definition of when a profinite group G is computable, due to Smith and la Roche (two papers from 1981):

$$G = \varprojlim_i (A_i, \psi_i)$$

for a computable diagram $(A_i, \psi_i)_{i \in \mathbb{N}}$ of finite groups and epimorphisms $\psi_i \colon A_i \to A_{i-1}$ $(i > 0)$.

Computable presentations of t.d.l.c. groups

- We aim at a robust definition of the class of t.d.l.c. groups with a computable presentation.
- We ask that our definition extend the existing definitions for discrete, and for profinite groups.
- We want this class to have good algorithmic closure properties.

We provide two types of computable presentations, which will turn out to be equivalent in the sense that from a presentation of one type one can construct a presentation of the other type.

1. Computable Baire presentations

Each totally disconnected Polish space is homeomorphic to the set of paths $[T]$ for some subtree T of $\mathbb{N}^*$ (the tree of strings with natural number entries). So there is no need to distinguish between names and objects in our setting.

For a computable Baire presentation,

- the domain of G equals $[T]$ for what we call an effectively σ-compact subtree of $\mathbb{N}^*$
- the operations are computable in the sense of Turing machines holding the infinite inputs on tapes of which the content keeps unchanged during the computation.

2. Computable presentations via a meet groupoid

- We introduce an algebraic structure $\mathcal{W}(G)$ on the countable set of compact open cosets in G, together with $\emptyset$.
- This structure is a partially ordered groupoid, with the usual set inclusion, and multiplication of a left coset of a subgroup U with a right coset of U (which is a coset).
- The intersection of two compact open cosets is such a coset itself, unless it is empty, so we have a meet semilattice.

A computable presentation of G via meet groupoids is a computable copy of the meet groupoid of G such that the index function on compact open subgroups, namely $U, V \mapsto |U : U \cap V|$, is also computable.

Section 2:

Computability on paths of rooted trees

Strings of natural numbers

- Let $\mathbb{N}^*$ denote the set of strings with natural numbers as entries. We use letters σ, τ, ρ etc. for elements of $\mathbb{N}^*$.

- The set $\mathbb{N}^*$ can be seen as a directed tree:
  - the empty string is the root, and
  - the successor relation is given by appending a number at the end of a string.

- $\sigma \preceq \tau$ denotes that σ is an initial segment of τ.

- can identify finite strings of length n with partial functions on $\mathbb{N}$ having domain $\{0, \ldots, n-1\}$.

Some notation for rooted trees

- By a (directed) tree we mean a nonempty subset T of $\mathbb{N}^*$ such that $\sigma \in T$ and $\rho \prec \sigma$ implies $\rho \in T$.

- By $[T]$ one denotes the set of paths of a tree T.

- If T has no leaves, $[T]$ is a closed set in Baire space $\mathbb{N}^{\mathbb{N}}$ equipped with the usual product topology.

- For $\sigma \in T$ let

$$[\sigma]_T = \{X \in [T] \colon \sigma \prec X\}.$$

  That is, $[\sigma]_T$ is the cone of paths on T that extend σ.

Computable functions on $[T]$

Let $T \subseteq \mathbb{N}^*$ be a computable rooted tree without leaves.

Definition

- A function $\Phi : [T] \times \mathbb{N} \to \mathbb{N}$ is computable if there is a Turing machine that when it has the list of the values $f(0), f(1), f(2), \ldots$ written on the read-only tape and w on some other tape, it can determine the value $\Phi(f, w)$.
- A function $\Phi \colon [T] \to [\mathbb{N}^*]$ is computable if the function $\widetilde{\Phi} : [T] \times \mathbb{N} \to \mathbb{N}$ given by $\widetilde{\Phi}(g, n) = \Phi(g)(n)$ is computable.
- In a similar way, define that $\Phi \colon [T] \times [S] \to [\mathbb{N}^*]$ is computable, using a TM with two read-only tapes.

Example

Let $T = \mathbb{N}^*$. The function $\Phi : [T] \times \mathbb{N} \to \mathbb{N}$ with $\Phi(f, w) = \sum_{i=0}^{w} f(i)$ is computable.

Proof.

The oracle TM with f written on the read-only tape queries the values of $f(i)$ for $i = 0, \dots, w$ one by one and adds them up. $\square$

Fact

Φ is computable $\Rightarrow$ Φ is continuous.

So e.g. the function $f \mapsto \min(\mathrm{range} f)$ is not computable.

(2.3) Computably σ-compact (c.s.c.) trees

Definition (computably σ-compact trees)

- Let T be a computable subtree of $\mathbb{N}^*$ without leaves such that only the root can have infinitely many successors.

- We say that T is computably σ-compact, or c.s.c. for short, if there is a computable binary function H such that, if $\rho \in T$ is a nonempty string, then
$$\rho(i) \leq H(\rho(0), i) \text{ for each } i < |\rho|.$$

Section 3:

Defining computably t.d.l.c. groups via Baire presentations

Computable Baire presentation

Definition (3.1)

A computable Baire presentation is a t.d.l.c. group of the form $H = ([T], \mathrm{Mult}, \mathrm{Inv})$ where

1. T is computably σ-compact as defined in 2.3;
2. $\mathrm{Mult}\colon [T] \times [T] \to [T]$ and $\mathrm{Inv}\colon [T] \to [T]$ are computable.

We say that a t.d.l.c. group G is computably t.d.l.c. (via a Baire presentation) if $G \cong H$ for such a group H.

Such a definitions works for t.d.l.c. algebraic structures in general.

Dependency of definitions

$$\begin{tikzcd}
 & \text{Computable tree } T \subseteq \mathbb{N}^* \text{ with no leaves} \arrow[dl] \arrow[dr] & \\
\boxed{\text{c.s.c. tree T}} \arrow[dr] & & \boxed{\text{comp. functions on [T]}} \arrow[dl] \\
 & \boxed{\text{computable Baire presentation of } G} &
\end{tikzcd}$$

The ring $\mathbb{Q}_p$ has a computable Baire presentation

Let Q be the tree of strings $\sigma \in \mathbb{N}^*$ such that

- all entries, except possibly the first, are among $\{0, \ldots, p-1\}$,
- $r0 \not\preceq \sigma$ for each $r > 0$.

A string $r\sigma \in Q$ denotes the rational $p^{-r} n_\sigma \in \mathbb{Z}[1/p]$, where n_σ is the number which has σ as a p-ary expansion, written in reverse order:

$$n_\sigma = \textstyle\sum_{i<|\sigma|} p^i \sigma(i).$$

For instance, let $p = 3$; then

$$(3,1,0,2) \text{ denotes the rational } 3^{-3} \cdot (1 + 2 \cdot 9) = 19/27.$$

We allow the case that σ ends in 0. The condition that $r0 \not\preceq \sigma$ for each $r > 0$ says that p does not divide n_σ.

Addition and multiplication on $\mathbb{Q}_p$ are computable

- For the addition operation, consider an oracle Turing machine with two oracle tapes starting with notations $r\sigma$ and $s\tau$ of numbers $p^{-r}m$ and $p^{-s}n$, where $|\sigma| + r - s = |\tau|$.
- Then $p^{-r}m + p^{-s}n = p^{-s}(p^{s-r}m + n)$.
- The machine can output a string denoting $p^{-r}m + p^{-s}n$.
- A similar argument works for multiplication.

Given a computable subtree S of a c.s.c. tree T, when can we get rid of the leaves but keep the same paths?

Proposition (3.2)

- Let T be a c.s.c. tree. Let S be a computable subtree of T.
- Suppose that there is a uniformly computable dense sequence $(f_i)_{i\in\mathbb{N}}$ in $[S]$.

Then the tree $\widetilde{S} = \{\sigma \colon [\sigma]_S \neq \emptyset\}$ is decidable.
It follows that $\widetilde{S}$ is c.s.c. Of course, $[\widetilde{S}] = [S]$.

Computable Baire presentation of $\mathrm{SL}_n(\mathbb{Q}_p)$

The presentation is based on the computable tree

$$T = \{\sigma \colon \forall i < n^2\, [\sigma^i \in Q]\},$$

where σ^i is the string of entries of σ in positions of the form $kn^2 + i$ for some $k \in \mathbb{N}$.

- The determinant function $[T] \to \mathbb{Q}_p$ is computable.
- It is easy to show that there is a computable subtree S of T such that paths on S denote matrices with determinant 1.
- $SL_n(\mathbb{Z}[1/p])$ is dense in $SL_n(\mathbb{Q}_p)$.
- The paths on S corresponding to matrices in $SL_n(\mathbb{Z}[1/p])$ are precisely the ones that are 0 from some point on. Clearly there is a computable listing $(f_i)_{i\in\mathbb{N}}$ of these paths.

So we can use Prop 3.2 to get rid of the leaves, and so to turn S into a computably σ-compact tree $\widetilde{S}$ with $[\widetilde{S}] = [S]$.

Section 4:

Defining computably t.d.l.c. groups

via meet groupoids

Axioms for groupoids

Intuitively, the notion of a groupoid generalizes the notion of a group by allowing that the binary operation is partial. A groupoid is given by a domain $\mathcal{W}$ on which a unary operation $(.)^{-1}$ and a partial binary operation, denoted by "$\cdot$", are defined. These operations satisfy the following conditions:

(a) associativity in the sense that $(A \cdot B) \cdot C = A \cdot (B \cdot C)$, with either both sides or no side defined (and so the parentheses can be omitted);

(b) $A \cdot A^{-1}$ and $A^{-1} \cdot A$ are always defined;

(c) if $A \cdot B$ is defined then $A \cdot B \cdot B^{-1} = A$ and $A^{-1} \cdot A \cdot B = B$.

Category view: a groupoid is a small category in which each morphism has an inverse.

$A \colon U \to V$ means that U, V are idempotent, and $A = UA = AV$.

Axioms for meet groupoids

Definition

A meet groupoid is a groupoid $(\mathcal{W}, \cdot, (.)^{-1})$ that is also a meet semilattice $(\mathcal{W}, \cap, \emptyset)$ of which $\emptyset$ is the least element.
Writing $A \subseteq B \iff A \cap B = A$,it satisfies the conditions

(d) $\emptyset^{-1} = \emptyset = \emptyset \cdot \emptyset$, and $\emptyset \cdot A$ and $A \cdot \emptyset$ are undefined for each $A \neq \emptyset$,

(e) if U, V are idempotents such that $U, V \neq \emptyset$, then $U \cap V \neq \emptyset$,

(f) $A \subseteq B \iff A^{-1} \subseteq B^{-1}$, and

(g) if $A_i \cdot B_i$ are defined $(i = 0, 1)$ and $A_0 \cap A_1 \neq \emptyset \neq B_0 \cap B_1$, then

$$(A_0 \cap A_1) \cdot (B_0 \cap B_1) = A_0 \cdot B_0 \cap A_1 \cdot B_1.$$

(g) implies that if U and V are idempotent, then so is $U \cap V$.

The meet groupoid of a t.d.l.c. group

Definition

Let G be a t.d.l.c. group. We define a meet groupoid $\mathcal{W}(G)$.

- Its domain consists of the compact open cosets in G, as well as the empty set.
- We define $A \cdot B$ to be the usual product AB in case that $A = B = \emptyset$, or A is a left coset of a subgroup V and B is a right coset of V;
- otherwise $A \cdot B$ is undefined.

We will use the usual group theoretic terminology for elements of an abstract meet groupoid $\mathcal{W}$.

- If U is an idempotent of $\mathcal{W}$ we call U a subgroup
- if $AU = A$ we call A a left coset of U
- if $UB = B$ we call B a right coset of U.

Where the approximation structures come from

The idea to study appropriate Polish groups via an algebraic structure on their open cosets is due to Katrin Tent, and first appeared in Kechris, N. and Tent, 2018. This idea was further elaborated in a 2021 paper by Nies, Schlicht and Tent on the complexity of the isomorphism problem for oligomorphic groups. In both works, approximation structures are used that are given by the ternary relation "$AB \subseteq C$", where A, B, C are certain open cosets. They are called "coarse groups".

In the present work, it will be important that we have explicit access to the combination of the groupoid and the meet semilattice structures (which coarse groups don't provide). Coarse groups are too "coarse" an algebraic structure to analyse algorithmic aspects of t.d.l.c. groups.

Computably t.d.l.c. groups via meet groupoids

A meet groupoid $\mathcal{W}$ is called Haar computable if

(a) its domain is a computable subset D of $\mathbb{N}$;

(b) the groupoid and meet operations are computable; in particular, the relation $\{\langle x, y\rangle \colon\ x, y \in S \wedge x \cdot y \text{ is defined}\}$ is computable;

(c) the partial function with domain contained in $D \times D$ sending a pair of subgroups $U, V \in \mathcal{W}$ to $|U : U \cap V|$ is computable.

Definition (Computably t.d.l.c. groups via meet groupoids)

Let G be a t.d.l.c. group. We say that G is computably t.d.l.c. via a meet groupoid if $\mathcal{W}(G)$ has a Haar computable copy $\mathcal{W}$.

Condition (c) is equivalent to saying that every Haar measure μ on G that assigns a rational to some (and hence every) compact open coset is computable on $\mathcal{W}$.

Computably t.d.l.c. via meet groupoid

Example (4.9)

For any prime p, the additive group $\mathbb{Q}_p$ and the group $\mathbb{Z} \ltimes \mathbb{Q}_p$ are computably t.d.l.c. via a meet groupoid.

$\boxed{\mathbb{Q}_p}$: proper open subgroups are of the form $U_r := p^r\mathbb{Z}_p$ for some $r \in \mathbb{Z}$, all compact. For each r there is a canonical epimorphism $\pi_r \colon \mathbb{Q}_p \to C_{p^\infty}$ with kernel U_r. So each compact open coset of U_r can be uniquely written in the form $D_{r,a} = \pi_r^{-1}(a)$ for some $a \in C_{p^\infty}$.

$\boxed{\mathbb{Z} \ltimes \mathbb{Q}_p}$: it has the same compact open subgroups as $\mathbb{Q}_p$.
We have $D_{r,a} = g^{-z} D_{r-z,a} g^z$ for each $z \in \mathbb{Z}$. So we have $A_{r,z} \colon U_{r-z} \to U_r$ where $A_{r,z} = g^z U_r$. In particular $A_{1,1} \colon U_0 \to U_1$.
For both groups, the groupoid operation and index function are computable.

Section 5:

Equivalence

of the two types of computable presentations

Equivalence of computable presentations in the sense of Def. 3.1 and Def. 4.6

Theorem

A group G is computably t.d.l.c. via a Baire presentation $\iff$ G is computably t.d.l.c. via a meet groupoid.

From a presentation of G of one type, one can uniformly obtain a presentation of G of the other type.

First discuss $\Leftarrow$, then $\Rightarrow$.
Each time we need a few preliminaries.

For "$\Leftarrow$": a computable presentation of the group of permutations of $\mathbb{N}$

For strings $\sigma_0, \sigma_1 \in \mathbb{N}^*$ of the same length n, let $\sigma_0 \otimes \sigma_1$ denote the string τ of that length such that $\tau(k) = \langle \sigma_0(k), \sigma_1(k) \rangle$ for each $k < n$ (where $\langle ., . \rangle$ is a computable pairing function, such as Cantor's). Define a computable tree without leaves by $\mathit{Tree}(S_\infty) = \{\sigma \otimes \tau \colon \sigma, \tau \in \mathbb{N}^* \wedge$

$$\sigma, \tau \text{ are 1-1} \wedge \sigma(\tau(k)) = k \wedge \tau(\sigma(i)) = i \text{ whenever defined}\}.$$

The paths of $\mathrm{Tree}(S_\infty)$ can be viewed as the permutations of $\mathbb{N}$, paired with their inverses:

$$[\mathit{Tree}(S_\infty)] = \{f \otimes f^{-1} \colon\ f \text{ is permutation of } \mathbb{N}\}.$$

The group operations on $\mathit{Tree}(S_\infty)$ are computable.

$\Leftarrow$: From meet groupoid to Baire presentation

Given: a Haar computable meet groupoid $\mathcal{W}$ with domain $\mathbb{N}$ such that $\mathcal{W}(G) \cong \mathcal{W}$. We can use this isomorphism to identify $\mathcal{W}(G)$ and $\mathcal{W}$.

Definition

Let $\widetilde{G} = \mathcal{G}_{\mathrm{comp}}(\mathcal{W})$ be the closed subgroup of S_∞ consisting of elements p that preserve the meet operation of $\mathcal{W}$, and satisfy $p(A) \cdot B = p(A \cdot B)$ whenever $A \cdot B$ is defined.

The following diagram displays the condition in the definition above in category terms as a commutative diagram.

$$\begin{tikzcd}
U \arrow[rr, "A"] & & V \arrow[rr, "B"] & & W \\
U' \arrow[urr, "p(A)"] \arrow[urrrr, "p(A \cdot B)"'] & & & &
\end{tikzcd}$$

1. $\widetilde{G} = \mathcal{G}_{\text{comp}}(\mathcal{W})$ is isomorphic to G

Define a group homomorphism $\Phi \colon G \to \widetilde{G}$ by letting $\Phi(g)$ be the element of S_∞ corresponding to the left action of g, i.e.

$$\Phi(g)(A) = gA$$

where $A \in \mathcal{W}(G)$ (identified with the computable copy $\mathcal{W}$).

Claim

$\Phi \colon G \cong \widetilde{G}$.

2: $\mathit{Tree}(\widetilde{G})$ is c.s.c.

Using the assumption that $\mathcal{W}$ is Haar computable, we show that $\mathit{Tree}(\widetilde{G})$ is c.s.c. To show that $\mathit{Tree}(\widetilde{G})$ is computable we use:

Claim

A finite injection α on $\mathbb{N}$ can be extended to some $p \in \widetilde{G}$
$\iff B \cdot A^{-1}$ is defined whenever $\alpha(A) = B$, and

$$\bigcap\{B \cdot A^{-1} \colon \alpha(A) = B\} \neq \emptyset.$$

To show that $\mathit{Tree}(\widetilde{G})$ has the computable bound on branching:

Lemma (Effectively finite suborbits)

Suppose that $U \in \mathcal{W}$ is a subgroup and L is a left coset of U. Let $F \in \mathcal{W}$. One can uniformly in U, L and F compute a strong index for the finite set $\mathcal{L} = \{p(F) \colon p \in [S] \wedge p(U) = L\}$.

Theorem (Recall)

A group G is computably t.d.l.c. via a Baire presentation $\Longleftrightarrow$ G is computably t.d.l.c. via a meet groupoid.

From a presentation of G of one type, one can uniformly obtain a presentation of G of the other type.

Now discuss the implication $\Rightarrow$.
We also need a few preliminaries.

Computing with compact open sets (1)

Let $h\colon \mathbb{N}^* \to \mathbb{N}$ be the canonical encoding given by $h(w) = \prod_{i<|w|} p_i^{w_i+1}$, where p_i is the i-th prime number.

Definition (Strong indices for finite sets of strings)

For a finite set $u \subseteq \mathbb{N}^*$ let $n_u = \sum_{\eta \in u} 2^{h(\eta)}$; one says that n_u is the strong index for u.

Definition (Code numbers for compact open sets)

Let T be a c.s.c. tree. For a finite set $u \subseteq T - \{\emptyset\}$, let

$$\mathcal{K}_u = \bigcup_{\eta \in u} [\eta]_T,$$

(note that this set is compact). By a code number for a compact open set $\mathcal{K} \subseteq [T]$ we mean the strong index for a finite set $u \subseteq \mathbb{N}^*$ such that $\mathcal{K} = \mathcal{K}_u$.

Computing with compact open sets (2)

Inclusion, Boolean operations are decidable for the $\mathcal{K}_u$

Lemma

Let T be a c.s.c. tree. Given code numbers u, w,

(i) one can compute code numbers for $\mathcal{K}_u \cup \mathcal{K}_w$ and $\mathcal{K}_u \cap \mathcal{K}_w$;

(ii) one can decide whether $\mathcal{K}_u \subseteq \mathcal{K}_w$. In particular, one can, given a code number $u \in \mathbb{N}$, compute the minimal code number $u^* \in \mathbb{N}$ such that $\mathcal{K}_{u^*} = \mathcal{K}_u$.

Definition

Given a c.s.c. tree T, let E_T denote the set of minimal code numbers for compact open subsets of $[T]$. By the foregoing lemma, E_T is decidable.

Computing with compact open sets (3)

Lemma

Let T, S be c.s.c. trees. Suppose a function $\Phi\colon [T] \to [S]$ is computable via a partial computable function P_Φ. Given code numbers u, w, one can decide whether $\Phi(\mathcal{K}_u^T) \subseteq \mathcal{K}_w^S$.

This is used to prove the following.

Lemma

Let G be computably t.d.l.c. via a computable Baire presentation $([T], \mathrm{Mult}, \mathrm{Inv})$. Recall the set E_T of minimal code numbers for compact open sets from def:E.

(i) There is a computable function $I\colon E_T \to E_T$ such that for each $u \in E_T$, one has $\mathcal{K}_{I(u)} = (\mathcal{K}_u)^{-1}$.

(ii) For $u, v, w \in E_T$ one can decide whether $\mathcal{K}_u\mathcal{K}_v \subseteq \mathcal{K}_w$.

“$\Rightarrow$”: From Baire presentation to meet groupoid

- The set $\{u \in E_T \colon \mathcal{K}_u \text{ is a coset}\}$ can be obtained via an existential quantification over a computable binary relation.

- So there is a computable 1-1 function θ defined on $\mathbb{N} - \{0\}$ such that the range of θ equals this set.

- Define a meet groupoid $\mathcal{W}$ with domain $\mathbb{N}$, thinking of n as coding $A_n = \mathcal{K}_{\theta(n)}$ for $n > 0$, and $A_0 = \emptyset$.

- Show the meet groupoid operations are computable.

- For subgroups $U, V \in \mathcal{W}$ compute $|U : U \cap V|$ by finding in $\mathcal{W}$ the distinct left cosets of $U \cap V$ contained in U with union U.

The actions are computable

Corollary (5.9, a corollary to "$\Rightarrow$" of the proof above)

- As before, let $\mathcal{W}$ be a Haar computable copy of $\mathcal{W}(G)$ (with domain $\mathbb{N}$).

- The left and right actions $[T] \times \mathbb{N} \to \mathbb{N}$, given by
$$(g, A) \mapsto gA \text{ and } (g, A) \mapsto Ag,$$
are computable.

Section 6:

Algorithmic properties of objects associated with a t.d.l.c. group

The modular function is computable

Throughout, let G be computably t.d.l.c. via a Baire presentation $([T], \mathrm{Mult}, \mathrm{Inv})$, and let $\mathcal{W}$ be the Haar computable copy of $\mathcal{W}(G)$.

Proposition

The modular function $\Delta\colon [T] \to \mathbb{Q}^+$ is computable.

Proof.

Let $V \in \mathcal{W}$ be any subgroup. Given $g \in [T]$, using Cor 5.9 compute $A = gV$. Compute $U \in \mathcal{W}$ such that A is a right coset of U, and hence $A = Ug$. For any left Haar measure μ on G, we have

$$\Delta(g) = \mu(A)/\mu(U) = \mu(V)/\mu(U).$$

We can choose μ computable; so this yields $\Delta(g)$. $\square$

Cayley-Abels graphs are computable

If G is compactly generated, there is a compact open subgroup U, and a set $S = \{s_1, \ldots, s_k\} \subseteq G$ such that $S = S^{-1}$ and $U \cup S$ algebraically generates G. The Cayley-Abels graph

$$\Gamma_{S,U} = (V_{S,U}, E_{S,U})$$

of G is given as follows. The vertex set $V_{S,U}$ is the set $L(U)$ of left cosets of U, and the edge relation is

$$E_{S,U} = \{\langle gU, gsU\rangle \colon\ g \in G, s \in S\}.$$

Theorem

Suppose that G is computably t.d.l.c. and compactly generated.

(i) Each Cayley-Abels graph $\Gamma_{S,U}$ of G has a computable copy $\mathcal{L}$.

(ii) Let $\Gamma_{T,V}$ be another Cayley-Abels graph. Any two computable copies of $\Gamma_{S,U}$ and $\Gamma_{T,V}$ are computably quasi-isometric.

$G = \mathrm{Aut}(T_d)$ has computable Baire presentation

- May assume the domain of T_d is $\mathbb{N}$; so we can view $\mathrm{Aut}(T_d)$ as a closed subgroup of S_∞.
- A finite injection α on T_d can be extended to an automorphism of T_d iff it preserves distances; this is a decidable condition.
- Each $\eta \in \mathit{Tree}(S_\infty)$ corresponds to an injection on T_d. So we can decide whether $[\eta]_{\mathit{Tree}(G)} = [\eta]_{\mathit{Tree}(S_\infty)} \cap G \neq \emptyset$.
- Clearly $[\eta]_{\mathit{Tree}(G)}$ is compact for every such nonempty string η.
- $\mathit{Tree}(G)$ is c.s.c.: if $\eta \in \mathit{Tree}(G)$, $\eta(0) = \langle a, b\rangle$ then for every $v \in \mathrm{dom}(\eta)$, letting $\eta(v) = \langle r, s\rangle$,

  $$\mathrm{dist}(0, v) = \mathrm{dist}(a, r) = \mathrm{dist}(b, s).$$

  This yields a computable function H with $\eta(v) \leq H(\eta(0), v)$ as required.

Algorithmic properties of the scale function

For a compact open subgroup V of G and an element $g \in G$ let $m(g, V) = |V^g \colon V \cap V^g|$.
Define the scale function $[T] \to \mathbb{N}$ by

$$s(g) = \min\{m(g, V) \colon V \text{ is a compact open subgroup}\}.$$

E.g., in $\mathbb{Z} \ltimes \mathbb{Q}_p$, where $g \in \mathbb{Z}$ acts as $x \to xp$, we have $s(g) = 1$, $s(g^{-1}) = p$.

Example (with Stephan Tornier)

For $d \geq 3$, the scale function on $Aut(T_d)$ in the computable presentation of Ex. 5.10 is computable.

The scale function on $Aut(T_d)$ is computable

An automorphism g of T_d has exactly one of three types:

1. g fixes a vertex v: then $s(g) = 1$ because g preserves the stabilizer of v, which is a compact open subgroup.
2. g inverts an edge: then $s(g) = 1$ because g preserves the set-wise stabilizer of the set of endpoints of this edge.
3. g translates along an axis: then $s(g) = (d-1)^{\ell}$ where ℓ is the length of the translation.

The Turing machine, with g as an oracle, searches in parallel for

- a witness to (1),
- a witness to (2),
- a sufficiently long piece of an axis as in (3) so that the shift becomes "visible".

It then outputs the corresponding value of the scale.

Basic algorithmic properties of the scale function

The following are immediate from Cor 5.9 (that the left and right actions on the Haar computable meet groupoid $\mathcal{W}$ are computable):

(i) The function $m\colon [T] \times \mathbb{N} \to \mathbb{N}$, $m(g, V) = |V^g\colon V \cap V^g|$ (defined to be 0 if the second argument is not a subgroup) is computable.

(ii) The scale is computable if and only if one can algorithmically decide whether a compact open subgroup V is mimimizing for g (Fact 6.5).

(iii) The scale function is computably approximable from above. That is, there is a computable function $\Theta : [T] \times \mathbb{N} \to \mathbb{N}$ such that $\Theta(f, r) \geq \Theta(f, r + 1)$ for each $f \in [T], r \in \mathbb{N}$, and $s(f) = \lim_r \Theta(f, r)$ (Fact 6.6).

An open question

Question

Given a computable presentation of a t.d.l.c. group G based on a tree T, is the scale function $s\colon [T] \to \mathbb{N}$ computable ?

- Willis' tidying procedure, and Möller's spectral radius formula don't answer this in the affirmative.
- If the answer is in the negative, one can further ask whether there is a computably presented G such that the scale is non-computable for each of its computable presentations.

Section 7:

Closure properties

of the class of computably t.d.l.c. groups

The class of computably t.d.l.c. groups is closed under suitable algorithmic versions of many constructions that have been studied in the theory of t.d.l.c. groups: (1), (2), (3) and (6) described in Thm 1.3. of Wesolek, Elementary t.d.l.c. groups (2015).

- passing to closed subgroups,
- taking group extensions via continuous actions,
- forming "local" direct products, and
- taking quotients by closed normal subgroups

The first three are reasonably straightforward. All computable presentations will be Baire. By $Tree(G)$ we denote the c.s.c. tree underlying such a presentation of G.

Computable closed subgroups

Fact

Let G be a computably t.d.l.c. group. Let H be a closed subgroup of G (so that $\mathit{Tree}(H)$ is a subtree of $\mathit{Tree}(G)$). Then H is computably t.d.l.c. via

- the Baire presentation based on the $\mathit{Tree}(H)$ (which is c.s.c.)
- the operations of G restricted to H.

Example

Burger and Mozes studied the closed subgroups $U(F)$ of $\mathit{Aut}(T_d)$, where $d \geq 3$ and F is a subgroup of S_d.
Using the computable presentation of $\mathit{Aut}(T_d)$ together with the preceding fact, each group $U(F)$ is computably t.d.l.c.

Computable semidirect products

Proposition

- Let G, H be computably t.d.l.c. groups.
- Suppose $\Phi\colon G \times H \to H$ is a computable function that specifies an action of G on H via topological automorphisms.

Then the topological semidirect product $L = G \ltimes_\Phi H$ is computably t.d.l.c.

Local direct products

For local direct products see Wesolek 2015, Def. 2.3.

Proposition (Prop 11.5 in the paper with Melnikov)

- Let $(G_i)_{i\in\mathbb{N}^+}$ be computably t.d.l.c. groups uniformly in i, and
- for each i let U_i be a compact open subgroup of G_i, uniformly in i.

Then $G = \bigoplus_{i\in\mathbb{N}^+}(G_i, U_i)$ is computably t.d.l.c.

Here the domain of $G = \bigoplus_{i\in\mathbb{N}^+}(G_i, U_i)$ consists of the functions in $f \in \prod_i G_i$ such that $f(i) \in U_i$ for sufficiently large i, and $\prod_i U_i$ is a compact open subgroup.

This result might be useful to show that the scale function can be noncomputable.

Quotients by computable closed normal subgroups

Theorem (Thm. 11.11 in paper with Melnikov)

Let N be a closed normal subgroup of G such that $\mathit{Tree}(N)$ is a computable subtree of $\mathit{Tree}(G)$. Then G/N is computably t.d.l.c.

An application of the theorem

Example (7.6)

For each prime p and each $n \geq 2$, the group $\mathrm{PGL}_n(\mathbb{Q}_p)$ is computably t.d.l.c.

Proof.

In Example 7.2 we obtain a computable Baire presentation $(T, \mathrm{Mult}, \mathrm{Inv})$ of $\mathrm{GL}_n(\mathbb{Q}_p)$. We employ the closed embedding $F\colon GL_n(\mathbb{Q}_p) \to SL_{n+1}(\mathbb{Q}_p)$ which extends a matrix A to the matrix B where the new row and new column vanish except for the diagonal element (which necessarily equals $(\det A)^{-1}$).

In this presentation, the centre N of $\mathrm{GL}_n(\mathbb{Q}_p)$ is given by the diagonal $(n+1) \times (n+1)$ matrices such that the first n entries of the diagonal agree. So clearly $\mathit{Tree}(N)$ is a computable subtree of the tree in Ex. 7.2.

References:

Kechris, N. and Tent, The complexity of topological group isomorphism, The Journal of Symbolic Logic, 83(3), 1190-1203. Arxiv: 1705.08081

Melnikov and N. Approximation groupoids. In preparation.

Lupini, Melnikov and N. Computable topological abelian groups. Submitted. Arxiv: 2105.12897

N., Schlicht and Tent, Coarse groups, and the isomorphism problem for oligomorphic groups. J. Math Logic, in press, Arxiv: 1903.08436.