COMPUTABLY TOTALLY DISCONNECTED LOCALLY COMPACT GROUPS

ALEXANDER MELNIKOV AND ANDRE NIES

Abstract. We study totally disconnected, locally compact (t.d.l.c.) groups from an algorithmic perspective. We give various approaches to defining computable presentations of t.d.l.c. groups, and show their equivalence. In the process, we obtain an algorithmic Stone-type duality between t.d.l.c. groups and certain countable ordered groupoids given by the compact open cosets. We exploit the flexibility given by these different approaches to show that several natural groups, such as $Aut(T_d)$ and $\mathrm{SL}_n(\mathbb{Q}_p)$, have computable presentations. We show that many construction leading from t.d.l.c. groups to new t.d.l.c. groups have algorithmic versions that stay within the class of computably presented t.d.l.c. groups. This leads to further examples, such as $\mathrm{PGL}_n(\mathbb{Q}_p)$. We study whether objects associated with computably t.d.l.c. groups are computable: the modular function, the scale function, and Cayley-Abels graphs in the compactly generated case. We give a criterion when computable presentations of t.d.l.c. groups are unique up to computable isomorphism, and apply it to $\mathbb{Q}_p$ as an additive group, and to the semidirect product $\mathbb{Z} \ltimes \mathbb{Q}_p$.

Contents

1. Introduction 1
2. Computably locally compact subtrees of $\mathbb{N}^*$ 8
3. Defining computable t.d.l.c. groups via closed subgroups of S_∞ 10
4. Defining computably t.d.l.c. groups via meet groupoids 12
5. Uniform equivalence of two definitions of computably t.d.l.c. 15
6. Computable functions on the set of paths of computable trees 17
7. Defining computably t.d.l.c. groups via Baire presentations 20
8. Computability for particular subclasses of t.d.l.c. groups 24
9. More on the equivalences of notions of computable presentation 26
10. Algorithmic properties of the scale function 29
11. Closure properties of the class of computably t.d.l.c. groups 30
12. Uniqueness of computable presentations 36
References 40

1. Introduction

Algorithmic aspects of t.d.l.c. groups play a considerable role in recent research such as Willis [50]; for a summary of directions being pursued at present, see the conference on computational aspects of t.d.l.c. groups in Newcastle, Oct. 2022. We develop a general algorithmic theory of totally disconnected, locally compact (t.d.l.c.) groups, hoping to establish a successful theoretical framework for these

The first author was supported by Rutherford Discovery Fellowship RDF-VUW1902 of the Royal Society Te Aparangi. The second author was supported by the Royal Society Te Aparangi under the standard Marsden grant UOA-19-346. This work received its initial impetus during a meeting of the authors at the Research Centre Coromandel in July 2020.

aspects. We assume that all topological groups in this paper have a countable basis. Our theory will address the following:

Questions 1.1.

(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) *Relative to such a presentation, are objects such as the rational valued Haar measures, the modular function, or the scale function computable?*
(c) *Which 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?*

After some background on t.d.l.c. groups and on computability we will discuss the questions raised initially. We hope that the first five sections of the paper will be accessible to readers with only basic knowledge of computability theory; as we progress, we will explain some notions from computability theory that are more advanced.

1.1. **Background on t.d.l.c. groups.** Van Dantzig [44] showed that each t.d.l.c. group has a neighbourhood basis of the identity consisting of compact open subgroups. With Question 1.1(a) in mind, we provide six examples of t.d.l.c. groups, and indicate a compact open subgroup when it is not obvious. We will return to them repeatedly during the course of the paper.

(i) All countable discrete groups are t.d.l.c.
(ii) All profinite groups are t.d.l.c.
(iii) $(\mathbb{Q}_p, +)$, the additive group of p-adic numbers for a prime p is an example of a t.d.l.c. group that is in neither of the two classes above. The additive group $\mathbb{Z}_p$ of p-adic integers forms a compact open subgroup.
(iv) The semidirect product $\mathbb{Z} \ltimes \mathbb{Q}_p$ corresponding to the automorphism $x \mapsto px$ on $\mathbb{Q}_p$, and $\mathbb{Z}_p$ is a compact open subgroup.
(v) Algebraic groups over local fields, such as $\mathrm{SL}_n(\mathbb{Q}_p)$ for $n \geq 2$, are t.d.l.c. Here $\mathrm{SL}_n(\mathbb{Z}_p)$ is a compact open subgroup.
(vi) Given a connected countable undirected graph such that each vertex has finite degree, its automorphism group is t.d.l.c. The stabiliser of any vertex forms a compact open subgroup.

By an *undirected tree* we mean a connected graph without cycles. For $d \geq 3$, by T_d one denotes the undirected tree where each vertex has degree d. The group $\mathrm{Aut}(T_d)$ was first studied by Tits [43].

Towards Question 1.1(b), we will review some objects that are associated with a locally compact group G. The left and right Haar measures on G are treated in standard textbooks such as [13]. Recall that for any left Haar measure μ on G and any $g \in G$, one obtains a further left Haar measure μ_g by defining $\mu_g(A) = \mu(Ag)$. By the uniqueness up to a multiplicative constant of the left Haar measure , there is a real $\Delta(g) > 0$ such that $\mu_g(A) = \Delta(g)\mu(A)$ for each measurable A. The function $\Delta\colon G \to \mathbb{R}^+$, called the *modular function* for G, is a group homomorphism.

Seminal work of Willis published in the 1990s, such as in [48], provided the impetus for intense research on t.d.l.c. groups in the last decades. He introduced the scale function $s\colon G \to \mathbb{N}^+$. Let $g \in G$. For a compact open subgroup V of G, let $m(g, V) = |V^g\colon V \cap V^g|$ be the displacement of V through conjugation by g (with the usual definition $V^g = g^{-1}Vg$). Let $s(g)$ be the minimum value of $m(g, V)$ over all V. It is not hard to show that $\Delta(g) = s(g)/s(g^{-1})$. If a group has a *normal* compact open subgroup, such as for the examples (i)–(iii) above, the scale function is constant of value 1. The group $\mathbb{Z} \ltimes \mathbb{Q}_p$ for a prime p is among the simplest

examples of a t.d.l.c. group with a nontrivial scale function: one has $s(t) = p$ for the generator t of $\mathbb{Z}$ such that $\alpha^t = \alpha/p$ for each $\alpha \in \mathbb{Q}_p$.

Cayley-Abels graphs form a further type of objects associated to certain t.d.l.c. groups. To motivate them, we recall that a guiding principle in the study of a t.d.l.c. group G is that it has a topological and a geometric aspect. The two aspects capture the small-scale (or local), and the large-scale (or global) behaviour, respectively. The small-scale aspect is given by a compact open subgroup U (provided by van Dantzig's theorem), which is of course profinite. Any two such subgroups are commensurable.

Next, we discuss the large-scale aspect, beginning with some background. Gromov and others initiated a geometric theory of (discrete) finitely generated groups via their Cayley graphs. Such graphs for different generating sets are quasi-isometric. This means that there is a map ϕ from one vertex set to the other that only distorts distances affinely, and for some constant c, each vertex has at most distance c from the range of ϕ. Large-scale geometric properties are invariant under quasi-isometry.

T.d.l.c. groups that are algebraically generated by a compact subset form the topological analog of discrete finitely generated groups. It is easy to see that given a compactly generated t.d.l.c. group G with a compact open subgroup U, the group G is generated by $U \cup S$ for some finite symmetric set S. The Cayley-Abels graph $\Gamma_{U,S}$ is the analog of the Cayley graph in this new setting, where the elements of the group are replaced by the left cosets of U. As before, any two such graphs are quasi-isometric. So one can think of the Cayley-Abels graphs as capturing the large-scale aspect of a compactly generated group.

For further background and references on the topics discussed above, see [47], and also [19].

1.2. **Background on computable mathematics.** A general goal of computable mathematics is to study the algorithmic content of areas such as algebra [1, 8], analysis [36, 45], or topology [39]. A first step is invariably to define what a computable presentation of a mathematical structure in that area is, such as a countable group, a complete metric space, or a separable Banach space. (Note that "computable" is the commonly accepted adjective used with presentations, rather than "algorithmic".) One also introduces and studies computability for objects related to the structure. For instance, in computable analysis one uses rational approximations to reals in order to define what it means for a function $f\colon \mathbb{R} \to \mathbb{R}$ to be computable (see this section further below). A large body of results addresses the algorithmic content of classical results. For example, consider the result that each continuous real-valued function f on the unit interval has a maximum value. In the algorithmic approach, one assumes the function is computable, and ask whether there is a *computable* at which the value is maximal. (The answer is "no" in general.)

There are also interesting new questions with no pendant in the classical setting. For instance, how difficult is it to recognize whether two computable presentations present the same structure? How hard is it to determine whether the structure presented has a certain property? (E.g., to determine whether a computably presented group is torsion-free.) The basic distinction is "decidable/undecidable". Mathematical logic provides a bevy of descriptive complexity classes, with corresponding completeness notions, for a more detailed answer in the undecidable case. For instance, torsion-freeness is of maximum complexity within the co-recursively enumerable properties.

Towards defining computable presentations, we first recall the definition of a computable function on $\mathbb{N}$, slightly adapted to our purposes in that we allow the domain to be any computable set.

Definition 1.2. 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.

A structure in the model theoretic sense consists of a nonempty set D, called the domain, with relations and functions defined on it. The following definition was first formulated in the 1960s by Mal'cev [23] and Rabin [37], independently.

Definition 1.3. 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 1.4. (a) For each $k \geq 1$, the group $GL_k(\mathbb{Q})$ is computably presenatble. To obtain a computable copy, one fixes an algorithmic encoding of the rational $k \times k$ matrices by natural numbers, and lets the domain D be the computable set of numbers that encode a matrix with nonzero determinant. Since the encoding is algorithmic, the domain and the matrix operations are computable.
(b) It is not hard to verify that an n-generated group G has a computable copy if and only if its word problem is decidable.

Next, we discuss how to define that an uncountable structure has a computable presentation. In the field of computable analysis, one represents all the elements of the structure by "names" which are directly accessible to computation, and requires that the functions and relations are computable on the names. For detail see e.g. [33, 39]. Often names are elements of the set $[T]$ of paths on some computable subtree T of $\mathbb{N}^*$ (the tree of strings with natural number entries). For instance, a standard name of a real number r is a path coding a sequence of rationals $\langle q_n \rangle_{n\in\mathbb{N}}$ such that $|q_n - q_{n+1}| \leq 2^{-n}$ and $\lim_n q_n = r$. Using so-called "oracle Turing machines", one can define computability of functions on $[T]$; we will provide detail in Section 6. This indirectly defines computability on spaces relevant to computable analysis; for instance, whether a function on $\mathbb{R}$ is computable. The example above shows that names and the object they denote can be of quite a different kind. In contrast, each totally disconnected Polish space is homeomorphic to $[T]$ for some subtree T of $\mathbb{N}^*$, which is advantageous because in principle there is no need to distinguish between names and objects in our setting.

A rather different, *ad hoc* way to define computability often works for particular classes of uncountable structures: impose algorithmic constraints on the definition of the class. For instance, a profinite group G is computable in the sense of Smith [41] and La Roche [20] if $G = \varprojlim_i (A_i, \psi_i)$ for a computable diagram $(A_i, \psi_i)_{i\in\mathbb{N}}$ of finite groups and epimorphisms $\psi_i\colon A_i \to A_{i-1}$ $(i > 0)$. Such a definition often turns out to be equivalent to the general definition of computability restricted to the class; for profinite groups this will follow by combining Proposition 7.3 and Proposition 8.6.

The aforementioned approach to profinite groups of Smith and La Roche admits equivalent formulations that work beyond this class. One such reformulation (in terms of effectively branching subtrees of $\mathbb{N}^*$) was discovered by Smith [41], and the other (in terms of effectively compact metric spaces) can be found in the very recent work [5] (see Thm. 4.33). Results of this sort indicate that the respective notion of computable presentability is robust in the class.

In the remaining sections of the introduction we discuss the questions posed at the beginning of the paper in more detail.

1.3. **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 want this class to have good algorithmic closure properties, and also ask that our definition extend the existing definitions for discrete, and for profinite groups. We provide several types of computable presentations, which will turn out to be equivalent.

Type S (for "symmetric group") is based on the fact that each t.d.l.c. group G is isomorphic to a closed subgroup of S_∞. We represent such subgroups by subtrees of $\mathbb{N}^*$ in a certain way (to be described below). We impose algorithmic conditions on the tree to define when the presentation is computable. This approach is consistent with earlier work [12] on computable subgroups of S_∞.

Type M (for "meet groupoid") is based on an algebraic structure $\mathcal{W}(G)$ on the countable set of compact open cosets in G. 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. The intersection of two compact open cosets is such a coset itself, unless it is empty. So, after adjoining $\emptyset$ as a least element (which does not interact with the groupoid structure), we obtain a meet semilattice. A Type M computable presentation of G 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. The idea to study appropriate Polish groups via an algebraic structure on their open cosets is due to Katrin Tent, and first appeared in [18]. This idea was further elaborated in a paper on the complexity of the isomorphism problem for oligomorphic groups [31]. There, 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.

Type B (where "B" stands for "Baire") of computable presentation generalizes Type S. For computable Baire presentations, one asks that the domain of G is what we call an effectively locally compact subtree of $\mathbb{N}^*$ (the tree of strings with natural number entries), and the operations are computable in the sense of oracle Turing machines. This approach generalizes one of the definitions in [41] from profinite groups to t.d.l.c. groups. It also works for t.d.l.c. structures other than groups. We postpone it until Section 7, mainly because it requires more advanced notions from computability theory. In particular, it relies on computable functions on the set of paths of a computable tree. These notions will be provided in Section 6.

Among our main results is that the various approaches to computable presentations are equivalent. This will be stated formally in Theorem 5.1 and its extension Theorem 7.6. Indeed, the approaches are equivalent in the strong sense that from a presentation of one type, one can effectively obtain a presentation of the other type for the same t.d.l.c. group. Below, we will initially say that a t.d.l.c. group is computably t.d.l.c. of a particular type, for instance via a closed subgroup of S_∞, or via a Baire presentation. Once the equivalences have been established, we will often omit this.

These results and the characterization established Theorem 5.1 suggest that our approach to computability for t.d.l.c. groups is natural and robust. We will later support this thesis with non-trivial examples showing that many widely studied t.d.l.c. groups are computably t.d.l.c., as well as further characterizations extending Theorem 5.1.

Baire presentations appear to be the simplest and most elegant notion of computable presentation for general totally disconnected Polish groups. However, computable Baire presentations are hard to study because the domain is usually uncountable (while the meet groupoids are countable), and there are no specific technical tools available (unlike the case of permutations of $\mathbb{N}$). In the proofs of several results, notably Theorem 11.10, we will work around this by replacing a Baire presentation with a more accessible one of Type M or S.

The operation that leads from a t.d.l.c. group to its meet groupoid has an inverse operation. Both operations are functorial for the categories with isomorphisms. This yields a duality between t.d.l.c. groups and a certain class of countable meet groupoids (similar to the duality in [31] between oligomorphic groups and the "coarse groups"). This class of meet groupoids can be described axiomatically. The equivalence of computable presentations of Type B and Type M can be extended to a computable version of that duality. We will elaborate on this in Remark 12.5 near the end of the paper.

1.4. **Which t.d.l.c. groups G have computable presentations?** Discrete groups, as well as profinite groups, have a computable presentation as t.d.l.c. groups if and only if they have one in the previously established sense, reviewed in Section 1.2 above. We will provide numerous examples of computable presentations for t.d.l.c. groups outside these two classes. The various equivalent approaches to computable presentations will be useful for this, because they allow us to construct a presentation of the type most appropriate for a given group. For a group of automorphisms such as $Aut(T_d)$ we will use Type S (presentations as closed subgroups of S_∞). For $(\mathbb{Q}_p, +)$ we use Type M (meet groupoids). For $\mathrm{SL}_n(\mathbb{Q}_p)$ we use Type B (computable Baire presentations). One can generalize the latter example to other algebraic groups over $\mathbb{Q}_p$ by using an effective form of σ-compactness, as mentioned in Section 1.8 below.

Some uncountable structures are equipped with a 'natural' computable structure. The Banach space $C([0,1],\mathbb{R})$ with the maximum norm has the dense set of polynomials $\mathbb{Q}[x]$; note that the distance between any two of them is computable. In other cases, it can be hard to determine whether a given uncountable structure admits a computable presentation. Some effort is needed to show that the compact metric space of probability measures on 2^ω has such a presentation; we cite [4]. Similarly, it can be difficult to determine whether a particular t.d.l.c. group has a computable presentation. Nonetheless, our broad conjecture is that *all* "natural" groups that are considered in the field of t.d.l.c. groups have computable presentations. An interesting testing ground for this conjecture is given by Neretin's groups $\mathcal{N}_d$ of almost automorphisms of T_d, for $d \geq 3$; see [16].

1.5. **Associated computable objects.** Recall that to a t.d.l.c. group G we associate its meet groupoid $\mathcal{W}(G)$, an algebraic structure on its compact open cosets. If G is given by a computable Baire presentation, then we construct a copy $\mathcal{W} = \mathcal{W}_{\mathrm{comp}}(G)$ that is computable in a strong sense, essentially including the condition that a rational valued Haar measure on G is computable as a function $\mathcal{W} \to \mathbb{R}$. We will show in Theorem 9.2 that the left, and hence also the right, action of G on $\mathcal{W}$ is computable. We conclude that the modular function on G is computable. If G is compactly generated, for each Cayley-Abels graph one can determine a computable copy, and any two copies of this type are computably quasi-isometric (Theorem 9.5). Intuitively, this means that the large-scale geometry of G is a computable invariant.

The computability of the scale function has been shown for particular t.d.l.c. groups in works such as [10] and [49, Section 6]; see the survey [50], which also considers the scale of a general endomorphism of G. In these particular cases, it

was generally clear what it means that one can compute the scale $s(g)$: provide an algorithm that shows it. One has to declare what kind input the algorithm takes; necessarily it has to be some approximation to g, as g ranges over a potentially uncountable domain. Our new framework allows us to give a precise meaning to the question whether the scale function is computable for a particular computable presentation of a t.d.l.c. group, thus also allowing for a precise negative answer. We note that this is reminiscent of the answer to Hilbert's 10th problem, which asked for an algorithm that decides whether a multivariate polynomial over $\mathbb{Z}$ has a zero. Only after a precise notion of computable function was introduced in the 1930s, it became possible to assert rigorously that no such algorithm exists; the final negative answer was given in 1970 by Y. Matyasevich in his PhD thesis submitted to the Steklov Institute. We leave the following open; also see Section 10 and Remark 11.7.

Question 1.5. *Given a computable presentation of a t.d.l.c. group G, is the scale function computable for this presentation?*

If the answer is in the negative, one can further ask whether for some computably presented G, the scale is non-computable for *each* of its computable presentations. An even stronger negative result would be that G can be chosen to have a unique computable presentation (see the discussion below).

1.6. **Algorithmic versions of constructions that lead from t.d.l.c. groups to new t.d.l.c. groups.** Section 11 shows that 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. In particular, the constructions (1), (2), (3) and (6) described in [46, Thm. 1.3] can be phrased algorithmically in such a way that they stay within the class of computably t.d.l.c. groups. This provides further evidence that our class is robust. These constructions are suitable versions, in our algorithmic topological setting, of passing to closed subgroups, taking extensions by continuous actions, forming "local" direct products, and taking quotients by closed normal subgroups (see [46, Section 2] for the detail). The algorithmic version of taking quotients (Theorem 11.10) is the most demanding; it uses extra insights from the proofs that the various forms of computable presentation are equivalent, which are provided in Theorem 9.2.

Several constructions lead to new examples of t.d.l.c. groups with computable presentations. E.g., after defining a computable presentation of $\mathrm{SL}_n(\mathbb{Q}_p)$ directly, we proceed to a computable presentation of $\mathrm{GL}_k(\mathbb{Q}_p)$ via taking a closed subgroup, and then to $\mathrm{PGL}_k(\mathbb{Q}_p)$ via taking a quotient.

1.7. **When is a computable presentation unique?** Willis [50, Section 5] writes that "it is a truism that computation in a group depends on the description of the group". In the present article, we apply our notion of computable presentability of a t.d.l.c. group to formally clarify this intuition, and also give some examples where the statement actually fails. Viewing a computable Baire presentation as a description, we are interested in the question whether such a description is unique, in the sense that between any two of them there is a computable isomorphism. Adapting terminology for countable structures going back to Mal'cev, we will call such a group *autostable*. If a t.d.l.c. group is autostable, then computation in the group can be seen as independent of its particular description.

We apply our methods to show in Section 9 that the additive group $\mathbb{Q}_p$ of the p-adic numbers is autostable, and so is $\mathbb{Z} \ltimes \mathbb{Q}_p$. Proving the autostability of these groups requires more effort than the reader would perhaps expect. For other groups, such as $\mathrm{SL}_n(\mathbb{Q}_p)$ for $n \geq 2$ and $Aut(T_d)$, we leave open whether a computable presentation is unique up to computable isomorphism.

1.8. **Some context.**
Related work on autostability. A countable structure is called autostable (or computably categorical [8, 1] if it has a computable copy, and such a copy is unique up to computable isomorphism. For example, any computable finitely generated algebraic structure is autostable. In contrast, there is a discrete 2-step nilpotent group with exactly two computable presentations up to computable isomorphism [11]. We note that in the discrete case our notion of autostability for t.d.l.c. groups reduces to the established one.

A profinite abelian group is autostable if and only if its Pontryagin dual is autostable [26]; note that this dual is a discrete, torsion abelian group. Autostability of the latter type of groups is characterized in [28]. In this way one obtains a characterization of autostability for profinite abelian groups.

Pour El and Richards [36] gave an example of a Banach space with two computable presentations without a computable linear isometry between them. Works such as [29] and then [3, 25, 14] systematically study autostability in separable spaces, using tools of computable (discrete) algebra.

Other related work. The first author has shown how to give an equivalent definition of a computable t.d.l.c. group in terms of an 'effectively σ-compact metric'. For a draft see [7, Section 4].

Our work [22] with Lupini focusses on abelian locally compact groups. We introduce two notions of computable presentation for abelian t.d.l.c. groups that take into account their special structural properties. The first is based on the fact that such groups are pro-countable, the other on the fact that such a group is an extension of a discrete group by a profinite group. Both notions can be used to provide further examples of computable abelian t.d.l.c. groups that are neither discrete nor compact. The work [22] states that in the abelian case, the notion of computable presentability given in the present paper is equivalent to these notions. We prove this in Section 8 of the present paper.

An approach to computability for Polish groups was suggested in [27] and then developed in, e.g., [26, 34]. In that approach, a Polish group is said to be computable if the underlying topological space is computably, completely metrized and the group operations are computable operators (functionals) on this space. It is known (see [26]) that there exist computably metrized profinite groups that do not possess a computable presentation in the sense of LaRoche and Smith. However, if we additionally assume that the underlying computably metrized space is 'effectively compact' (equivalently, the Haar measure is computable [34, 5]), then we can produce a computable presentation of the group in the sense of LaRoche and Smith; see [5] for a proof.

2. Computably locally compact subtrees of $\mathbb{N}^*$

Definition 2.4 in this section introduces computably locally compact trees. This purely computability theoretic concept will be of central technical importance for the whole paper. For basics on computability theory see, e.g., the first two chapters of [42], or the first chapter of [32] which also contains notation on strings and trees. Our paper is mostly consistent with the terminology of these two sources. They also serve for basic concepts such as Turing programs, computable functions, as well as partial computable (or partial recursive) functions, which will be needed from Section 6 onwards. In this section we will review some more specialized concepts related to computability.

Notation 2.1. 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. We write $\sigma \preceq \tau$ to denote that σ is an initial segment of τ, and $\sigma \prec \tau$ to denote that σ is a proper initial segment. We can also identify finite strings of length $n+1$ with partial functions $\mathbb{N} \to \mathbb{N}$ having finite support $\{0, \dots, n\}$. We then write τ_i instead of $\tau(i)$. By $\max(\tau)$ we denote $\max\{\tau_i \colon i \leq n\}$. 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 2.2 (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.

We will usually identify a finite subset of $\mathbb{N}^*$ with its strong index.

Remark 2.3. Why "strong index"? By an index, in computability theory one usually means a (code for a) Turing program that decides a set or computes a function. Such an index can be obtained from a strong index as defined above. However, a strong index tells us more about the finite set, such as its size. This is not true for an index as a Turing program (even if we are promised that the index describes a finite set).

Unless otherwise mentioned, by a (directed) tree we mean a nonempty subset of T of $\mathbb{N}^*$ such that $\sigma \in T$ and $\rho \prec \sigma$ implies $\rho \in T$. By $[T]$ one denotes the set of (infinite) paths of a tree T. It carries the topology inherited from Baire space $\mathbb{N}^{\mathbb{N}}$ with the usual product topology. Note that if T has no leaves, then $[T]$ is compact if and only if each level of T is finite; in other words, if and only if T is finitely branching. 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.

Definition 2.4 (c.l.c. trees)**.** Let T be a computable subtree of $\mathbb{N}^*$ without leaves. We say that T is *computably locally compact (c.l.c.)* if

(1) the space $[T]$ is locally compact,
(2) the set $\{\sigma \in T \colon [\sigma]_T \text{ is compact}\}$ is decidable, and
(3) the tree of extensions of such a string is uniformly computably branching. More formally, there is a computable binary function H such that, if $[\sigma]_T$ is compact and $\rho \in T$ extends σ, then $\rho(i) \leq H(\sigma, i)$ for each $i < |\rho|$.

The following will be used throughout.

Definition 2.5 (Code numbers for compact open sets)**.** Let T be a c.l.c. tree. For a finite set $u \subseteq T$, let

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

in case this set is compact. By a *code number* for a compact open set $\mathcal{K} \subseteq [T]$ we mean the strong index for a set u of strings such that $\mathcal{K} = \mathcal{K}_u$.

Such a code number is not unique (unless $\mathcal{K}$ is empty). So we will carefully distinguish between the actual compact open set, and any of its code numbers. Note that $\mathcal{K}_u$ is defined if and only if $[\eta]_T$ is compact for each $\eta \in u$. So one can decide, given $u \in \mathbb{N}$ as an input, whether u is a code number. Clearly, each compact open subset of $[T]$ is of the form $\mathcal{K}_u$ for some u.

The following lemma shows that the basic set-theoretic relations and operations are decidable for sets of the form $\mathcal{K}_u$, similar to the case of finite subsets of $\mathbb{N}$.

Lemma 2.6. Suppose that we are given a c.l.c. tree T. 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 $\mathcal{K}_{u^*} = \mathcal{K}_u$.

Proof. (i) The case of union is trivial. For the intersection operation, it suffices to consider the case that u and w are singletons. For strings $\alpha, \beta \in T$, one has $[\alpha]_T \cap [\beta]_T = \emptyset$ if α, β are incompatbile, and otherwise $[\alpha]_T \cap [\beta]_T = [\gamma]_T$ where γ is the longest common initial segment of α, β.
(ii) Let H be a computable binary function as in Definition 2.4. It suffices to consider the case that u is a singleton. Suppose that $\alpha \in T$ and $[\alpha]_T$ is compact. The algorithm to decide whether $[\alpha]_T \subseteq \mathcal{K}_w$ is as follows. Let N be the maximum length of a string in w. Answer "yes" if for each $\beta \succeq \alpha$ of length N such that $\beta(k) \leq H(\alpha, k)$ for each $k < N$, there is $\gamma \in w$ such that $\gamma \preceq \beta$. Otherwise, answer "no". □

Definition 2.7. Given a c.l.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.

3. Defining computable t.d.l.c. groups via closed subgroups of S_∞

Convention 3.1. *To avoid trivial cases, for the rest of the paper we will assume that all t.d.l.c. groups are infinite.*

This section, in particular Def. 3.4, spells out Type S of computable presentations of t.d.l.c. groups. It was informally described in Section 1.3.

3.1. Computable closed subgroups of S_∞.

By S_∞ we denote the topological group of permutations of $\mathbb{N}$. We first provide a computable presentation of S_∞ based on the set of paths of a tree. It is related to the computable presentation of S_∞ in [12, Def 1.2], where S_∞ is viewed as a topological group with a computable compatible metric. However, in our presentation, an element of S_∞ is given as a path on a tree that encodes pairs (h, h^{-1}) where h is a permutation of $\mathbb{N}$. This enables us to define a computable tree, denoted $\mathit{Tree}(S_\infty)$, each path of which corresponds to a permutation of $\mathbb{N}$.

Suppose strings $\sigma_0, \sigma_1 \in \mathbb{N}^*$ both have length N. By $\sigma_0 \oplus \sigma_1$ we denote the string of length $2N$ that alternates between σ_0 and σ_1. That is,

$$(\sigma_0 \oplus \sigma_1)(2i + b) = \sigma_b(i) \text{ for } i < N, b = 0, 1.$$

Similarly, for functions f_0, f_1 on $\mathbb{N}$, we define a function $f_0 \oplus f_1$ on $\mathbb{N}$ by

$$(f_0 \oplus f_1)(2i + b) = f_b(i).$$

Informally, $\mathit{Tree}(S_\infty)$ is the tree of strings so that it is consistent that the entries at odd positions extend to an inverse of the entries at even positions. Let $\mathit{Tree}(S_\infty) =$

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

Our concrete presentation of S_∞ is the group defined on the paths of $\mathit{Tree}(S_\infty)$. So we view S_∞ as the group of functions of the form $h \oplus h^{-1}$ where h is a permutation of $\mathbb{N}$. If $f = f_0 \oplus f_1$ and $g = g_0 \oplus g_1$ in S_∞, we define $f^{-1} = f_1 \oplus f_0$ and $gf = (g_0 \circ f_0) \oplus (f_1 \circ g_1)$. We will verify in Fact 6.4 below that these group operations are computable (in the sense of Definition 6.2).

Definition 3.2. We say that a closed subgroup C of S_∞ is *computable* if its corresponding tree, namely $\mathit{Tree}(C) = \{\eta \in \mathit{Tree}(S_\infty) \colon [\eta]_T \cap C \neq \emptyset\}$ is computable.

Remark 3.3. It is well known that the closed subgroups of S_∞ are precisely the automorphism groups of structures M with domain $\mathbb{N}$. Suppose that M is a computable structure, and there is an algorithm to decide whether a bijection between finite subsets of M (encoded by a strong index) can be extended to an automorphism. Then the automorphism group is computable. To see this, one uses that a string $\eta = \sigma \oplus \tau$ on $\mathit{Tree}(S_\infty)$ determines the finite injective map

$$\alpha_\eta = \{\langle i, k\rangle \colon\ \sigma(i) = k \vee \tau(k) = i\} \tag{1}$$

between finite subsets of M. This map is extendible to an automorphism of M if and only if $\eta \in \mathit{Tree}(C)$.

For instance, assuming a computable bijection between $\mathbb{Q}$ and $\mathbb{N}$, the group $\mathrm{Aut}(\mathbb{Q}, <)$ is computable: By Cantor's back and forth argument, a bijection between finite subsets of $\mathbb{Q}$ can be extended to an automorphism of G if and only if it preserves the ordering. There is an algorithm to decide the latter condition.

3.2. First definition of computably t.d.l.c. groups.

Definition 3.4 (Computably t.d.l.c. groups via closed subgroups of S_∞)**.** Let G be a t.d.l.c. group. We say that G is *computably t.d.l.c.* (via a closed subgroup of S_∞) if there is a closed subgroup C of S_∞ such that $G \cong C$, and the tree

$$\mathit{Tree}(C) = \{\eta \in \mathit{Tree}(S_\infty) \colon [\eta] \cap C \neq \emptyset\}$$

is c.l.c. in the sense of Def. 2.4.

In this context we will often ignore he difference between G and C. That is, we will assume that G itself is a closed subgroup of S_∞.

Recall from the introduction that $\mathrm{Aut}(T_d)$ is the group of automorphism of the undirected tree T_d where each vertex has degree d.

Example 3.5. Let $d \geq 3$. The t.d.l.c. group $G = \mathrm{Aut}(T_d)$ is computably t.d.l.c. via a closed subgroup of S_∞.

Proof. Via an effective encoding of the vertices of T_d by the natural numbers, we can view G itself as a closed subgroup of S_∞. We can decide whether a finite injection α on T_d can be extended to an automorphism by checking whether it preserves distances. Each $\eta \in \mathit{Tree}(S_\infty)$ corresponds to such an injection. 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.

To see that $\mathit{Tree}(G)$ is c.l.c. as defined in Def. 2.4, note that if $\sigma \in \mathit{Tree}(G)$ maps $x \in T_d$ to $y \in T_d$, then every extension $\eta \in \mathit{Tree}(G)$ of σ maps elements in T_d at distance n from x to elements in T_d at distance n from y, and conversely. This yields a computable bound $H(\sigma, i)$ as required in (3) of Def. 2.4. □

We supply a lemma showing that given a group G as in Definition 3.4, the group operations are algorithmic when applied to its compact open subsets.

Lemma 3.6. Suppose G is computably t.d.l.c. via a closed subgroup of S_∞ (identified with G). Write $T = \mathit{Tree}(G)$. Recall from Definitions 2.5 and 2.7 that $\mathcal{K}_u$ denotes the open subset of $[T]$ with code number u, and that $E_T \subseteq \mathbb{N}$ denotes the computable set of minimal code numbers for compact open subsets of $[T]$.

(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) There is a computable function $M \colon E_T \times E_T \to E_T$ such that for each $u, v \in E_T$, one has $\mathcal{K}_{M(u,v)} = \mathcal{K}_u \mathcal{K}_v$.

Proof. In the argument below, we will use Lemma 2.6 without mention. For (i), let $I(u)$ be the least strong index for the set $\{\sigma_1 \oplus \sigma_0 \colon \sigma_0 \oplus \sigma_1 \in u\}$. For (ii), first note that since $\mathit{Tree}(G)$ is c.l.c., we can computably replace each string σ in u by its set of extensions on $\mathit{Tree}(G)$ of a given length $N \geq |\sigma|$. So we may assume that all the strings in $u \cup v$ have the same length. Hence it suffices to define $M(u, v)$ in case that $u = \{\sigma\}$ and $v = \{\tau\}$ where $|\sigma| = |\tau| =: n$.

For such σ, τ let $m = 1 + \max(\sigma, \tau)$; that is, $m - 1$ is the maximum number occurring in any of the two strings. For strings $\gamma, \delta \in \mathbb{N}^*$ such that $|\delta| \geq 1 + \max(\gamma)$,

by $\delta \cdot \gamma$ we denote the string $\langle \delta(\gamma(i)) \rangle_{i<|\gamma|}$. We will verify that for each $f \in G$,

$$(2)\ f \in [\tau]_T[\sigma]_T \quad \Leftrightarrow \quad \exists \beta \succ \tau \exists \alpha \succ \sigma [\beta, \alpha \in T \wedge |\beta| = 2m \wedge |\alpha| = 2\max(\beta) + 2 \wedge \beta_0 \cdot \sigma_0 \prec f_0 \wedge \alpha_1 \cdot \beta_1 \prec f_1],$$

where $\alpha = \alpha_0 \oplus \alpha_1$, $\beta = \beta_0 \oplus \beta_1$, and $f = f_0 \oplus f_1$ as usual. Given this, we let $M(u, v)$ be the least strong index for the set of strings $(\beta_0 \cdot \sigma_0 \oplus \alpha_1 \cdot \tau_1)$ as above, which we can compute from u and v by the hypothesis on T. This will complete the proof of (ii) of the claim.

If the left hand side of (2) holds, then $f = hg$ for some $g, h \in G$ such that $\sigma \prec g$ and $\tau \prec h$. Then the right hand side holds via $\beta = h \upharpoonright 2m$ an $\mathrm{d}\alpha = g \upharpoonright 2\max \beta + 2$.

Now suppose that the right hand side of (2) holds. Since $\beta \in T$, there is $h \in G$ such that $h \succ \beta$. Let $g = h^{-1}f$. Then $g \in G$. Since $h \succ \tau$, it suffices to show that $g \succ \sigma$. Note that by definition $f = f_0 \oplus f_1$ where $f_1 = (f_0)^{-1}$, and similarly $g = g_0 \oplus g_1$ and $h = h_0 \oplus h_1$. We have $g_0 = h_1 \circ f_0$ and $g_1 = f_1 \circ h_0$.

We check that $g_0 \succ \sigma_0$ as follows: for each $i < n$ we have $\beta_0(\sigma_0(i)) = f_0(i)$ by hypothesis. Hence $h_0(\sigma_0(i)) = f_0(i)$, so $\sigma_0(i)) = h_1(f_0(i)) = g_0(i)$.

Next, we check that $g_1 \succ \sigma_1$: using $\alpha_1 \cdot \beta_1 \prec f_1$, for each $i < n$ we have

$$g_1(i) = f_1(h_0(i)) = f_1(\tau_0(i)) = \alpha_1(\beta_1(\tau_0(i))).$$

Since $\beta \in T$, $\beta_0 \succ \tau_0$ and $|\beta_1| > \max(\tau_0)$, we have $\beta_1(\tau_0(i)) = i$. So the value of the rightmost term is $\alpha_1(i)$, which equals $\sigma_1(i)$. □

4. Defining computably t.d.l.c. groups via meet groupoids

This section provides the detail for the second type (Type M) of computable presentations of t.d.l.c. groups described in Section 1.3.

4.1. **The meet groupoid of a t.d.l.c. group.** Intuitively, a *groupoid* generalizes the notion of a group by allowing that the binary operation is partial. A groupoid is given by a domain $\mathcal{W}$, together with a unary operation $X \to X^{-1}$ and a partial binary operation, denoted by "$\cdot$". 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 $A^{-1} \cdot A \cdot B = B$.

Equivalently, one can view a groupoid as a small category where each morphism has an inverse. The elements of the domain are the morphisms of the category. The morphisms U such that $U \cdot U = U$ correspond to the objects of the category. One has $A \colon U \to V$ where $U = A \cdot A^{-1}$ and $V = A^{-1} \cdot A$.

Definition 4.1. A *meet groupoid* is a groupoid $(\mathcal{W}, \cdot, .^{-1})$ that is also a meet semilattice with least element. The meet operation is denoted $\cap$, the least element $\emptyset$, and inclusion is defined by $A \subseteq B \Leftrightarrow A \cap B = A$. It satisfies the conditions that $\emptyset^{-1} = \emptyset$, that $\emptyset \cdot A$ and $A \cdot \emptyset$ are undefined for each A, and that the groupoid operations are monotonic:

(d) $A \subseteq B \Leftrightarrow A^{-1} \subseteq B^{-1}$, and
(e) if $A_i \cdot B_i$ are defined $(i = 0, 1)$ and $A_0 \subseteq A_1, B_0 \subseteq B_1$, then $A_0 \cdot B_0 \subseteq A_1 \cdot B_1$.

Given meet groupoids $\mathcal{W}_0, \mathcal{W}_1$, a bijection $h \colon \mathcal{W}_0 \to \mathcal{W}_1$ is an *isomorphism* if it preserves the three operations. Given a meet groupoid $\mathcal{W}$, the letters A, B, C will range over elements of $\mathcal{W}$, and U, V, W range over objects in $\mathcal{W}$, i.e., elements A such that $A \cdot A = A$.

Note that we use set theoretic notation even if the partial order is not actual inclusion of sets. This is because the main examples we have in mind are set theoretic, as follows.

Definition 4.2. 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 (i.e., cosets of compact open subgroups of G), as well as the empty set. We define $A \cdot B$ to be the usual product AB in case that A is a left coset of a subgroup V and B is a right coset of V; otherwise $A \cdot B$ is undefined.

The intersection of two cosets is empty, or again a coset. So we have

Fact 4.3. *$\mathcal{W}(G)$ is a meet groupoid with the groupoid operations $\cdot$ and $A \to A^{-1}$, and the usual intersection operation $\cap$.*

Viewing a groupoid as a small category, $A\colon U \to V$ means that A is a right coset of U and a left coset of V. So, if $A\colon U \to V$ and $B\colon V \to W$, then $A \cdot B\colon U \to W$ as required. We note that $\mathcal{W}(G)$ satisfies the axioms of "inductive groupoids" as defined in [21, page 109]. See [6, Section 4] for more on an axiomatic approach to meet groupoids.

4.2. **Second definition of computably t.d.l.c. groups.**

Definition 4.4 (Haar computable 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 the sense of Definition 1.2; 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.

Here $|U\colon U \cap V|$ is defined abstractly as the number of left, or equivalently right, cosets of $U \cap V$ contained in U; we require implicitly that this number is always finite. Note that by (b), the partial order induced by the meet semilattice structure of $\mathcal{W}$ is computable. Also, (b) implies that being a subgroup is decidable when viewed as a property of elements of the domain S; this is used in (c). The condition (c) intuitively corresponds to the computable bound $H(\sigma, i)$ required in (3) of Definition 2.4. For ease of reading will say that $n \in D$ *denotes* a coset A, rather than saying that n "is" a coset.

Definition 4.5 (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 $\mathcal{W}$. In this context, we call $\mathcal{W}$ a computable presentation of G (in the sense of meet groupoids).

Remark 4.6. In this setting, Condition (c) of Definition 4.4 is equivalent to saying that every Haar measure μ on G that assigns a rational to some compact open subgroup (and hence is rational valued) is computable on $\mathcal{W}$, in the sense that the function assigning the rational $\mu(A)$ to a compact open coset A is computable. Consider left Haar measures, say. First suppose (c) holds. Given A, compute the subgroup V such that $A = A \cdot V$, i.e., A is a left coset of V. Compute $W = U \cap V$. We have $\mu(A) = \mu(V) = \mu(U) \cdot |V : W|/|U : W|$.

Conversely, if the Haar measure is computable on $\mathcal{W}$, then (c) holds because $|U\colon V| = \mu(U)/\mu(V)$.

For discrete groups, the condition (c) can be dropped as the proof of the following shows.

Example 4.7. A discrete group G is computably t.d.l.c. via a meet groupoid $\Leftrightarrow$ G has a computable copy in the usual sense of Definition 1.3.

Proof. For the implication $\Leftarrow$, we may assume that G itself is computable; in particular, we may assume its domain is a computable subset of $\mathbb{N}$. Each compact coset in G is finite, and hence can be represented by a strong index for a finite set

of natural numbers. Since the group operations are computable on the domain, this implies that the meet groupoid of G has a computable copy. It is then trivially Haar computable.

For the implication $\Rightarrow$, let $\mathcal{W}$ be a Haar computable copy of $\mathcal{W}(G)$. Since G is discrete, $\mathcal{W}$ contains a least subgroup U. The set of left cosets of U is computable, and forms a group with the groupoid and inverse operations. This yields the required computable copy of G. □

By $\mathbb{Q}_p$ we denote the additive group of the p-adics. By the usual definition of semidirect product ([38, p. 27]), $\mathbb{Z} \ltimes \mathbb{Q}_p$ is the group defined on the Cartesian product $\mathbb{Z} \times \mathbb{Q}_p$ via the binary operation $\langle z_1, \alpha_1\rangle \cdot \langle z_2, \alpha_2\rangle = \langle z_1 + z_2, p^{z_2}\alpha_1 + \alpha_2\rangle$. This turns $\mathbb{Z} \ltimes \mathbb{Q}_p$ into a topological group with the product topology.

Example 4.8. 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. Analogous results hold when $\mathbb{Q}_p$ is replaced by $\mathbb{F}_p((t))$, the abelian group of infinite Laurent series over $\mathbb{F}_p$.

Proof. We begin with the additive group $\mathbb{Q}_p$. Note that its open proper subgroups are of the form $U_r = p^r\mathbb{Z}_p$ for some $r \in \mathbb{Z}$. Let C_{p^∞} denote the Prüfer group $\mathbb{Z}[1/p]/\mathbb{Z}$, where $\mathbb{Z}[1/p] = \{zp^{-r} \colon z \in \mathbb{Z} \wedge r \in \mathbb{N}\}$. For each r there is a canonical epimorphism $\pi_r \colon \mathbb{Q}_p \to C_{p^\infty}$ with kernel U_r: if $\alpha = \sum_{i=-n}^{\infty} s_i p^i$ where $0 \leq s_i < p$, $n \in \mathbb{N}$, we have

$$\pi_r(\alpha) = \mathbb{Z} + \textstyle\sum_{i=-n}^{r-1} s_i p^{i-r};$$

here an empty sum is interpreted as 0. (Informally, $\pi_r(\alpha)$ is the "tail" of α from the position $r-1$ onwards to the last position, and shifted to be represented by an element of C_{p^∞}.) So each compact open coset in $\mathbb{Q}_p$ can be uniquely written in the form $D_{r,a} = \pi_r^{-1}(a)$ for some $r \in \mathbb{Z}$ and $a \in C_{p^\infty}$. Formally speaking, the domain $S \subseteq \mathbb{N}$ of the Haar computable copy $\mathcal{W}$ of $\mathcal{W}(\mathbb{Q}_p)$ consists of natural numbers canonically encoding such pairs $\langle r, a\rangle$. However, they will be identified with the cosets they denote.

The groupoid operations are computable because we have $D_{r,a}^{-1} = D_{r,-a}$, and $D_{r,a} \cdot D_{s,b} = D_{r,a+b}$ if $r = s$, and undefined else. Furthermore, using the informal view of $D_{r,a}$ as the set of α with tail a from position $r-1$ on, it is easy to check that $D_{r,a} \subseteq D_{s,b}$ iff $r \geq s$ and $p^{r-s}a = b$. So the inclusion relation is decidable. We have $D_{r,a} \cap D_{s,b} = \emptyset$ unless one of the sets is contained in the other, so the meet operation is computable. Finally, for $r \leq s$, we have $|U_r : U_s| = p^{s-r}$.

Next, let $G = \mathbb{Z} \ltimes \mathbb{Q}_p$; we build a Haar computable copy $\mathcal{V}$ of $\mathcal{W}(G)$. We will extend the listing $(D_{r,a})_{r\in\mathbb{Z},a\in C_{p^\infty}}$ of compact open cosets in $\mathbb{Q}_p$ given above. For each compact open subgroup of G, the projection onto $\mathbb{Z}$ is compact open, and hence the trivial group. So the only compact open subgroups of G are of the form U_r. Let g be the generator of $\mathbb{Z}$ such that $\alpha^g = p\alpha$ for each $\alpha \in \mathbb{Q}_p$. Each compact open coset of G has a unique form $g^z D_{r,a}$ for some $z \in \mathbb{Z}$. Formally speaking, the domain of the computable copy of $\mathcal{W}(G)$ consists of natural numbers encoding the triples $\langle z, r, a\rangle$ corresponding to such cosets; as before they will be identified with the cosets they denote.

To show that the groupoid and meet operations are computable, note that we have $gD_{r,a} = D_{r-1,a}g$ for each $r \in \mathbb{Z}, a \in C_{p^\infty}$, and hence $g^z D_{r,a} = D_{r-z,a}g^z$ for each $z \in \mathbb{Z}$. Given two cosets $g^v D_{r,a}$ and $g^w D_{s,b} = D_{s-w,b}g^w$, their composition is defined iff $r = s - w$, in which case the result is $g^{v+w}D_{s,a+b}$. The inverse of $g^z D_{r,a}$ is $D_{r,-a}g^{-z} = g^{-z}D_{r-z,-a}$.

To decide the inclusion relation, note that we have $g^z D_{r,a} \subseteq g^w D_{s,b}$ iff $z = w$ and $D_{r,a} \subseteq D_{s,b}$, and otherwise, they are disjoint. Using this one can show that the meet operation is computable (by an argument that works in any computable

meet groupoid $\mathcal{V}$): if $A_0, A_1 \in \mathcal{V}$, $A_i \colon U_i \to V_i$, and A_0, A_1 are not disjoint, then $A_0 \cap A_1$ is the unique $C \in \mathcal{V}$ such that $C \colon U_0 \cap U_1 \to V_0 \cap V_1$ and $C \subseteq A_0, A_1$. Since $\mathcal{W}$ satisfies Condition (c) in Definition 4.4, and $\mathcal{V}$ has no subgroups beyond the ones present in W, we conclude that $\mathcal{V}$ is Haar computable.

In the case of $\mathbb{F}_p((t)), +$, let $U_r = t^{-r}\mathbb{F}_p[[t]]$, $r \in \mathbb{Z}$. There is a natural surjection $\pi_r \colon \mathbb{F}_p((t)) \to C_p^{(\omega)}$ with kernel U_r, where $C_p^{(\omega)}$ is the additive group of the vector space of dimension ω over $\mathbb{F}_p$: for $\alpha = \sum_{i=-n}^{\infty} s_i t^i$, $0 \leq s_i < p$,

$$\pi_r(\alpha) = \mathbb{F}_p[[t]] + \textstyle\sum_{i=-n}^{r-1} s_i t^{i-r}.$$

The rest of the proof carries over. □

Remark 4.9. By Proposition 11.3 below, the class of computably t.d.l.c. groups is closed under finite direct products. It follows that the additive group of any totally disconnected local field is computably t.d.l.c.

Remark 4.10. The profinite group $(\mathbb{Z}_p, +)$ is computably t.d.l.c., because the meet groupoid of cosets contained in U_0 is Haar computable. One can also modify the proof above in order to obtain this directly. Let now r range over $\mathbb{N}$, and instead of the π_r consider the natural projections $\eta_r \colon \mathbb{Z}_p \to C_{p^r}$. Let $D_{r,a} = \eta_r^{-1}(a)$ for $a \in C_{p^r}$. With these new notations, the computability of the operations and inclusion relation are obtained the same way as before, regarding the cyclic groups C_{p^r} as subgroups of C_{p^∞}.

5. Uniform equivalence of two definitions of computably t.d.l.c.

We show that Definitions 3.4 and 4.5 of computably t.d.l.c. groups are equivalent. This provides our first evidence for the robustness of this class of t.d.l.c. groups.

Theorem 5.1.
A group G is computably t.d.l.c. via a closed subgroup of S_∞ ⇔
G is computably t.d.l.c. via a meet groupoid.
Moreover, the equivalence is uniform*, in the sense that from a presentation of G of one type, one can effectively obtain a presentation of G of the other type.*

Proof. "⇒": Suppose that G is computably t.d.l.c. as a closed subgroup of S_∞. To save on notation, we may assume that G itself is a closed subgroup of S_∞ showing this. We will obtain a Haar computable copy of its meet groupoid $\mathcal{W}(G)$.

Recall from Def. 2.7 that $E = E_{Tree(G)} \subseteq \mathbb{N}$ is the decidable set of minimal code numbers for compact open subsets of G. For $u \in E$ we can decide whether $\mathcal{K}_u$ is a subgroup. This is the case precisely when $\mathcal{K}_{M(u,u)} = \mathcal{K}_u$ and $\mathcal{K}_{I(u)} = \mathcal{K}_u$, where the computable functions I, M were defined in Lemma 3.6. We can also decide whether $B = \mathcal{K}_v$ is a coset. This is the case precisely when BB^{-1} is a subgroup U (in which case B is its right coset). Equivalently, $\mathcal{K}_{M(v,I(v))}$ is a subgroup, which is a decidable condition.

The domain D of the computable copy of $\mathcal{W}(G)$ is the subset of E consisting of the minimal code numbers for cosets. For cosets $A = \mathcal{K}_u, B = \mathcal{K}_v \in \mathcal{W}(G)$ recall that $A \cdot B$ is defined iff A is a left coset of a subgroup V such that B is a right coset of V. Equivalently, $\mathcal{K}_{M(I(u),u)} = \mathcal{K}_{M(v,I(v))}$, which is a decidable condition. So the groupoid operations and meet operation are computable.

It remains to show that given code numbers $u, v \in D$ of subgroups U, V, one can compute $|U : U \cap V|$. To do so, one finds in D more and more distinct left cosets of $U \cap V$ contained in U, until their union equals U. (There is an algorithm to decide the latter condition by Lemma 2.6(i).) Then one outputs the number of cosets found.

"$\Leftarrow$": We begin by defining a computable operator that is dual to the operation of sending G to a computable copy of $\mathcal{W}(G)$ obtained above.

Definition 5.2. Given a Haar computable meet groupoid $\mathcal{W}$ with domain $\mathbb{N}$, let $\mathcal{G}_{\mathrm{comp}}(\mathcal{W})$ 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.

Note that these are the automorphisms of the structure obtained from $\mathcal{W}$ which instead of composition for each B has the partial unary operation $A \to A \cdot B$. Recall here that the elements of S_∞ are not actually permutations, but paths on $\mathit{Tree}(\widetilde{G})$ encoding pairs consisting of a permutation and its inverse. However, if $p \in \mathit{Tree}(\widetilde{G})$ and $A \in \mathcal{W}$ is denoted by i, we will suggestively write $p(A)$ for the element of $\mathcal{W}$ denoted by $p(2i)$. We note that for each subgroup $U \in \mathcal{W}(G)$, the set $B = p(U)$ satisfies $B \cdot U = p(U) \cdot U = p(U \cdot U) = B$, and hence is a left coset of U.

We show that $\mathit{Tree}(\widetilde{G})$ is c.l.c. as in Definition 2.4. The first claim will be used to show that $\mathit{Tree}(\widetilde{G})$ is computable.

Claim 5.3. *A finite injection α on $\mathbb{N}$ can be extended to some $p \in \widetilde{G} \Leftrightarrow$ $B \cdot A^{-1}$ is defined whenever $\alpha(A) = B$, and $\bigcap\{B \cdot A^{-1} \colon \alpha(A) = B\}$ is non-empty.*

$\Leftarrow$: Let g be an element of the intersection. Then $gA = B \cdot A^{-1} \cdot A = B = \alpha(A)$ for each $A \in \mathrm{dom}(\alpha)$.
$\Rightarrow$: Suppose $p \in \widetilde{G}$ extends α. By the foregoing claim, there is $g \in G$ such that $p = \Phi(g)$. Then $gA = p(A) = B$ for each A, B such that $\alpha(A) = B$. Such A, B are then right cosets of the same subgroup. Hence $B \cdot A^{-1}$ is defined, and clearly g is in the intersection. This establishes the claim.

Recall from Subsection 3.1 that a string $\sigma \oplus \tau \in \mathit{Tree}(S_\infty)$ gives rise to a finite injection $\alpha_{\sigma\oplus\tau}$, defined by $\alpha_{\sigma\oplus\tau}(r) = s$ iff $\sigma(r) = s \vee \tau(s) = r$. So

$$S = \{\sigma \oplus \tau \colon \alpha_{\sigma\oplus\tau} \text{ can be extended to some } p \in \widetilde{G}\}$$

is a computable subtree of $\mathit{Tree}(S_\infty)$ without leaves, and $\widetilde{G} = [S]$. Hence $S = \mathit{Tree}(\widetilde{G})$.

Claim 5.5 below will verify that $S = \mathit{Tree}(\widetilde{G})$ satisfies Condition (3) in Def. 2.4 of c.l.c. trees. The following lemma does the main work, and will also be used later on, such as the proof of Prop. 9.2 below. Informally it says that given some subgroup $U \in \mathcal{W}$, if one declares that $p \in \widetilde{G}$ has a value $L \in \mathcal{W}$ at U, then one can compute for any $F \in \mathcal{W}$ the finite set of possible values of p at F.

Lemma 5.4. 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 $\mathcal{L} = \{p(F) \colon p \in [S] \wedge p(U) = L\}$.

To see this, first one computes $V = F^{-1}F$, so that F is a right coset of the subgroup V. Next one computes $k = |U \colon U \cap V|$, the number of left cosets of $U \cap V$ in U. Note that $\mathcal{L}_0 = \{p(U \cap V) \colon p \in [S] \wedge p(U) = L\}$ is the set of left cosets of $U \cap V$ contained in L. Clearly this set has size k. By searching $\mathcal{W}$ until all of its elements have appeared, one can compute a strong index for this set. Next one computes a strong index for the set $\mathcal{L}_1$ of left cosets D of V such that $C \subseteq D$ for some $C \in \mathcal{L}_0$ (this uses that given C one can compute D). Finally one outputs a strong index for the set $\{DF \colon D \in \mathcal{L}_1\}$, which equals $\mathcal{L}$. This shows the lemma.

Claim 5.5. *There is a computable binary function H such that, if $\sigma = (a, b)$ and $\rho \in S$ extends σ, then $\rho(i) \leq H(\sigma, i)$ for each $i < |\rho|$. In particular, $[\sigma]_S$ is compact for each string $\sigma \in S$ of length 2.*

To see this, let U be the subgroup denoted by 0. Let F be the coset denoted by k. If $i = 2k$, let L be the coset denoted by a. If $i = 2k+1$, let L be the coset denoted by b. Applying Lemma 5.4 to U, L, F, one can compute $H(\sigma, i)$ as the greatest number denoting an element of $\{p(F)\colon p \in [S] \wedge p(U) = L\}$.

Now suppose that $\mathcal{W}$ is as in Definition 4.5. Recall the Convention 3.1 that all t.d.l.c. groups are infinite. So the domain of $\mathcal{W}$ equals $\mathbb{N}$, and there is an isomorphism of meet groupoids $\mathcal{W} \to \mathcal{W}(G)$, which below we will use to identify $\mathcal{W}$ and $\mathcal{W}(G)$. We note that making the assumption that 0 is a subgroup in $\mathcal{W}$ did not affect the uniformity statement of the theorem: we can search $\mathcal{W}$ for the least n such that n is a subgroup, and then work with a new copy of $\mathcal{W}$ where 0 and n are swapped.

Let $\widetilde{G} = \mathcal{G}_{\mathrm{comp}}(\mathcal{W})$ defined in 5.2. 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. $A \mapsto gA$ where $A \in \mathcal{W}(G)$. (See the comment after Definition 5.2.) Note that Φ is injective because the compact open subgroups form a neighbourhood basis of 1: if $g \neq 1$ then $g \notin U$ for some compact open subgroup U, so that $\Phi(g)(U) \neq U$.

Claim 5.6. $\Phi\colon G \cong \widetilde{G}$.

To show that Φ is onto, let $p \in \widetilde{G}$. Since $\{p(U)\colon U \in \mathcal{W}(G) \text{ is a subgroup}\}$ is a filter on $\mathcal{W}(G)$ containing a compact set, there is an element g in its intersection. Then $\Phi(g) = p$: recall that for each subgroup $U \in \mathcal{W}(G)$, the set $B = p(U)$ is a left coset of U. So, if A is a right coset of U, then $p(A) = p(U \cdot A) = B \cdot A = gA$.

To show that Φ is continuous at 1 (and hence continuous), note that a basis of neighbourhoods of the identity in $\widetilde{G}$ is given by the open sets

$$\{p \in \widetilde{G}\colon \forall i \leq n\, [p(A_i) = A_i]\},$$

where $A_1, \ldots, A_n \in \mathcal{W}(G)$. Given such a set, suppose A_i is a right coset of U_i, and let $U = \bigcap U_i$. If $g \in U$ then $gA_i = A_i$ for each i.

The open mapping theorem for Hausdorff groups says that every surjective continuous homomorphism from a σ-compact group (such as a t.d.l.c. group with a countable basis of the topology) onto a Baire group is open. So Φ is open. This verifies the claim and concludes the proof. □

6. Computable functions on the set of paths of computable trees

This section provides preliminaries on computability of functions that are defined on the set of paths of computable trees. These preliminaries will be used in Section 7 to introduce computable Baire presentations of t.d.l.c. groups, as well as in much of the rest of the paper. Most of the content of this section can either be seen as a special case of known results in abstract computable topology, or can be derived from such results. These results stem from the study of effectively compact metric spaces; some of them need to be extrapolated to the locally compact setting. They can be found in the recent surveys [15, 5]. However, with the reader in mind who has little background in computability or computable topology, we prefer to provide intuition and elementary proofs for the computability notions and results that are needed later on, rather than referring to such more general results.

Let T be a computable subtree of $\mathbb{N}^*$ without leaves. To define that a function which takes arguments from the potentially uncountable domain $[T]$ is computable, one descends to the countable domain of strings on T, where the usual computability notions work. The first definition, Def. 6.1 below, will apply when we show in Corollary 9.4 that the modular function on a computable presentation of a t.d.l.c. group is computable. As a further example, in Fact 10.1 we will show that given a

computable presentation of a t.d.l.c. group via a closed subgroup of S_∞, the function $m(g, V)$ related to the scale function, defined in the introduction, is computable.

Definition 6.1. A function $\Phi : [T] \times \mathbb{N} \to \mathbb{N}$ is computable if there is a partial computable function P_Φ defined on a subset of $T \times \mathbb{N}$, with values in $\mathbb{N}$, such that

(1) if $\sigma \prec \tau \in T$, then $P_\Phi(\sigma, n) = k$ implies $P_\Phi(\tau, n) = k$;
(2) $\Phi(f, w) = k$ iff there is an n such that $P_\Phi(f\upharpoonright_n, w) = k$.

The intuition is that the partial computable function P_Φ represents the behaviour of a so-called *oracle Turing machine*. The machine has the list of the values $f(0), f(1), f(2), \dots$ written on a special tape, and attempts to determine the value $\Phi(f, w)$ via queries of the type "what is $f(q)$?". If $\sigma \prec f$ and $P_\Phi(\sigma, w) = \eta$, then with the answers to the queries given by σ, the machine can determine this value. Clearly any extension τ of σ will then yield the same value: this is condition (1). Condition (2) says that sufficiently many queries will lead to an answer. Note that this condition implies that every computable function Φ is continuous.

The next, closely related definition is a version of a well known notion in computability theory. It will matter when we discuss computability for groups where the domain can be of the form $[T]$ for any computable tree without leaves. We need a way to say that the group operations are computable. For the setting of closed subgroups of S_∞, this was the case automatically (for the inversion operation it was due to the particular presentation of S_∞ we chose).

Definition 6.2 (Computable functions on the set of paths)**.** Let T, S be computable trees without leaves. A function $\Phi\colon [T] \to [S]$ is called *computable* if there is a partial computable monotonic function P_Φ defined on a subset of T, with values in S, such that

$$\Phi(f) = \bigcup_n \{P_\Phi(f\upharpoonright_n)\colon\ f\upharpoonright_n \in \mathrm{dom}(P_\Phi)\}$$

for each $f \in [T]$.

Similarly, a function $\Psi\colon [T] \times [T] \to [S]$ is *computable* if there is a partial computable monotonic function P_Ψ defined on pairs of strings in T of equal length, with values in S, such that

$$\Psi(f, g) = \bigcup_n \{P_\Psi(f\upharpoonright_n, g\upharpoonright_n)\colon\ (f\upharpoonright_n, g\upharpoonright_n) \in \mathrm{dom}(P_\Phi)\}.$$

It is intuitively clear, and not hard to check from the definitions, that the composition of computable unary functions is again computable. Furthermore, it is easy to verify that a function $\Phi\colon [T] \to [\mathbb{N}^*]$ is computable if and only if the function $\widetilde{\Phi} : [T] \times \mathbb{N} \to \mathbb{N}$ given by $\widetilde{\Phi}(g, n) = \Phi(g)(n)$ is computable in the sense of Def. 6.1.

Remark 6.3. The topology on the space $[T]$ is induced by a complete metric: for instance let $d(f, g) = 1/n$ where n is least such that $f(n) \neq g(n)$. Taking the usual "ε, δ" definition, one sees that $\Phi\colon [T] \to [S]$ is continuous at f if for each k there is n such that for each $g \in [T]$, if $f\upharpoonright_n = g\upharpoonright_n$ then $\Phi(f)\upharpoonright_k = \Phi(g)\upharpoonright_k$. The definitions above can be seen as algorithmic versions of continuity, where one can compute the output $\Phi(f)$ up to k from a sufficiently long part of the input f.

Recall that in Section 3.1 we introduced a special way of presenting the elements of S_∞ as paths $f = f_0 \oplus f_1$ on a directed tree $\mathit{Tree}(S_\infty)$ that keep track of both the permutation f_0 and its inverse f_1. For $f, g \in [\mathit{Tree}(S_\infty)]$, we defined the group operations of S_∞ by $f^{-1} = f_1 \oplus f_0$ and $gf = (g_0 \circ f_0) \oplus (f_1 \circ g_1)$.

Fact 6.4. *The inverse operation and the group operation of S_∞ are computable.*

Proof. We define partial computable functions P_1 and P_2 on $\mathit{Tree}(S_\infty)$, one for each operation. For the inverse, let $P_1(\sigma_0 \oplus \sigma_1) = \sigma_1 \oplus \sigma_0$, where $|\sigma_0| = |\sigma_1|$. For the group operation, given strings $\tau = \tau_0 \oplus \tau_1$ and $\sigma = \sigma_0 \oplus \sigma_1$ of the same, even length, let t be greatest such that for each $r < t$, $\rho_0(r) := \tau_0(\sigma_0(r))$ and $\rho_1(r) := \sigma_1(\tau_1(s))$

are defined. Let $P_2(\tau,\sigma) = \rho_0 \oplus \rho_1$, which is a string of length $2t$. We omit the verification that these functions work. □

The following lemma is an algorithmic version of the fact, well known in computable analysis, that each continuous function defined on a compact space is uniformly continuous. Here we restrict ourselves to the setting of paths spaces $[K]$ that are compact in an effective way (as given by the bound H below).

Lemma 6.5. Suppose that K and S are computable trees without leaves. Suppose further that there is a computable function H such that $\sigma(i) < H(i)$ for each $\sigma \in K$ and $i < |\sigma|$. Let $\Phi\colon [K] \to [S]$ be computable via a partial computable function P_Φ.

(i) There is a computable function g as follows:
$$\forall n \in \mathbb{N}\, \forall \rho \in K\, [|\rho| = g(n) \to |P_\Phi(\rho)| > n].$$
Furthermore, g is obtained uniformly in K and h.

(ii) If Φ is a bijection then Φ^{-1} is computable, via a partial computable function that is obtained uniformly in K, H, S and P_Φ.

Intuitively, the function g in (i) computes the "δ" in the definition of uniform continuity from the "ε": if $\delta = 1/n$ we have $\varepsilon = 1/g(n)$. In computability terms, this means that to obtain $n+1$ output symbols we need at most $g(n)$ input symbols. (This is a slight generalisation of the well-known fact, going back to Nerode in the 1950s, that a Turing functional defined on each infinite bit sequence can be replaced by a truth-table functional.)

Proof. (i) First we claim that for each n, a possible value r for $g(n)$ exists. Assume that n is the least counterexample to this claim. Consider the subtree of K consisting of the initial segments of strings ρ such that $|P_\Phi(\rho)| \leq n$. This subtree is infinite by assumption. Hence there is a path $f \in [K]$. Clearly $\Phi(f)(n)$ is undefined, contradiction. Now, using the hypotheses on K, given n one can search for the least such value r and output it as $g(n)$.

(ii) We define a partial computable function $P_{\Phi^{-1}}$. Clearly $[S]$ is compact and Φ^{-1} is continuous. Hence there is a computable function h with $h(s) \geq s$ such that given $s \in \mathbb{N}$, for each η of length $g(h(s))$, $P_\Phi(\eta)\restriction_{h(s)}$ determines $\eta\restriction_s$. For $\beta \in S$ of length $h(s)$, define $P_{\Phi^{-1}}(\beta) = \alpha$ if $|\alpha| = s$ and there is $\eta \succeq \alpha$ of length $g(h(s))$ with $P_\Phi(\eta) \succeq \beta$. It is easy to verify that $P_{\Phi^{-1}}$ shows that Φ^{-1} is computable, and that $P_{\Phi^{-1}}$ is obtained uniformly in the given data. □

The rest of this section discusses computable functions on the set of paths of c.l.c. trees. Given such a tree T, recall from Def. 2.5 that by $\mathcal{K}_u$ we denote the compact open subset of $[T]$ with code number $u \in \mathbb{N}$. That is, u is the strong index for a set of strings $\{\alpha_1, \dots, \alpha_r\} \subseteq T$ such that $\mathcal{K}_u = \bigcup_{i \leq r} [\alpha_i]_T$, in case this set is compact. If there is more than one tree under discussion, we will write $\mathcal{K}_u^T$ for the subset of $[T]$ with code number u. First, we establish a useful interaction between computable functions $[T] \to [S]$ and such sets. Note that this generalizes Lemma 2.6(ii) where Φ is the identity function.

Lemma 6.6. Let T, S be c.l.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 $\Phi(\mathcal{K}_u^T) \subseteq \mathcal{K}_w^S$.

Proof. Suppose that u is a strong index for the set of strings $\{\alpha_1, \dots, \alpha_r\} \subseteq T$, and w is a strong index for the set of strings $\{\beta_1, \dots, \beta_s\} \subseteq S$. Let K be the subtree of T consisting of the prefixes, or extensions, of some α_i. Clearly one can uniformly obtain a computable bound h for this K as in Lemma 6.5. Let $n = \max_i |\beta_i|$. Let N be the length computed from n through that Lemma. Then $\Phi(\mathcal{K}_u^T) \subseteq \mathcal{K}_w^S$ if

and only if for each $\alpha \in K$ of length N, there is an i such that $P_\Phi(\alpha) \succeq \beta_i$. By Lemma 6.5, this condition is decidable. □

We remark that from the viewpoint of computability theory, one can assume that in a c.l.c. tree only the root is infinitely branching. More formally:

Remark 6.7. Given a c.l.c. tree T, one can effectively obtain (1) a c.l.c. tree T' so that the only infinite branching is at the root, and (2) a bi-computable bijection $[T] \to [T']$. To see this, note that there is a computable listing $(\sigma_i)_{i\in\mathbb{N}}$ of the minimal strings $\sigma \in T$ such that $[\sigma]_T$ is compact. The partial computable function P given by $P(\sigma_i\tau) = i\tau$ induces a computable bijection $[T] \to [T']$, with computable inverse given by P^{-1}.

To prove Proposition 7.8 below, we will need a criterion on whether, given a computable subtree S of a c.l.c. tree T (where S potentially has leaves), the maximally pruned subtree of S with the same set of paths is computable.

Proposition 6.8. *Let T be a c.l.c. tree such that only the root is infinitely branching. Let S be a computable subtree of T, and 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.l.c. Of course, $[\widetilde{S}] = [S]$.)*

Proof. Given a string $\sigma \in T$, if $\sigma = \emptyset$ then $\sigma \in \widetilde{S}$. Assuming $\sigma \neq \emptyset$, we can compute the least $t \in \mathbb{N}$ such that $\sigma \prec f_t$, or $\rho \notin S$ for each $\rho \in T$ of length t such that $\rho \succeq \sigma$; the latter condition can be decided by the hypotheses on T. Clearly $\sigma \in \widetilde{S}$ iff the former condition holds. □

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

This section spells out the third type of computable presentations of t.d.l.c. groups described in Section 1.3 (Type B). We call them *computable Baire presentation*.

It is well-known that each totally disconnected Polish space X is homeomorphic to $[T]$ for some tree $T \subseteq \mathbb{N}^*$; see [17, I.7.8]. Clearly X is locally compact iff for each $f \in [T]$ there is an n such that the tree above $f\restriction_n$ is finitely branching. So in our effective setting, it is natural to work with a domain of the presentation that has the form $[T]$ for a c.l.c. tree T, and require that the group operations on $[T]$ be computable. (In fact, while the previous types of computable presentation were group-specific, in the present setting the same approach would work for other types of algebraic structure defined on $[T]$.) We will show in Thm. 7.6 that the resulting notion of computably t.d.l.c. group is equivalent to the previous ones.

Despite this equivalence, as presentations, computable Baire presentations (Type B) offer more flexibility than the first type (Type S), which relied on closed subgroups of S_∞. This will be evidenced by our proofs that some algebraic groups over local fields are computable, such as $\mathrm{SL}_n(\mathbb{Q}_p)$ and $\mathrm{SL}_n(\mathbb{F}_p((t)))$.

Definition 7.1. A *computable Baire presentation* is a topological group of the form $H = ([T], Op, Inv)$ such that

(1) T is computably locally compact as defined in 2.4;
(2) $Op\colon [T]\times[T] \to [T]$ and $Inv\colon [T] \to [T]$ 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.

One requirement on our notions of computable presentability for t.d.l.c. groups is that for profinite, as well as discrete groups, we obtain the previously accepted notions. We show here has each computable profinite group has a computable Baire presentation. First we provide the formal definition.

Definition 7.2 (LaRoche [20], Smith [41])**.** A computable profinite presentation of a profinite group G is a uniformly computable sequence of strong indices of finite groups A_i and finite surjective maps $\phi_i : \mathcal{A}_i \to \mathcal{A}_{i-1}$ such that $G = \varprojlim_i (A_i, \phi_i)$.

Proposition 7.3. *Given a computable profinite presentation of G, one can effectively obtain a computable Baire presentation of G.*

Proof. We use the notation of Definition 7.2. Let $n_i = |\mathcal{A}_i|$. By the hypotheses one can effectively identify the elements of $\mathcal{A}_i$ with $\{0, \dots, n_i - 1\}$. Let

$$T = \{\sigma \in \mathbb{N}^* : \forall i < |\sigma| [\sigma(i) < n_i \wedge \phi_i(\sigma(i)) = \sigma(i-1) \text{ if } i > 0\}.$$

It is clear from the hypotheses that $[T]$ is compact, and T is effectively (locally) compact. To show that the binary group operation Op on $[T]$ is computable, for strings σ_0, σ_1 on T of the same length $k+1$, let $P_2(\sigma_0, \sigma_1)$ be the unique string $\tau \in T$ of length $k+1$ such that in $\mathcal{A}_k$ one has $\sigma_0(k)\sigma_1(k) = \tau(k)$. The inversion operation Inv on $[T]$ is computable by a similar argument. □

We will see in Prop. 8.6 that the converse holds as well. So, as promised, for profinite groups our definition of computably t.d.l.c. group coincides with the existing one. We note that Smith [41] already obtained this equivalence, albeit using a different terminology.

Proposition 7.4. *A discrete group G has a computable presentation in the usual sense of Definition 1.3 if and only if it has computable Baire presentation.*

Proof. $\Rightarrow$: Suppose the computable presentation has as a domain a computable set $D \subseteq \mathbb{N}$. We take the c.l.c. tree

$$T_G = \{r0^k : \ r \in D \wedge k \in \mathbb{N}\},$$

and the operations canonically defined on $[T_G]$ via partial computable functions that only regard the first entry of the input strings.
$\Leftarrow$: By Theorem 7.6 proved shortly below, G has a computable presentation via a meet groupoid. Now it suffices to invoke Example 4.7. □

By Fact 6.4, each computable presentation of G as a closed subgroup of S_∞ (Definition 3.4) is a computable Baire presentation. Recall that Lemma 3.6 shows that for a computable presentation as a closed subgroup of S_∞, the group theoretic operations on compact open sets are algorithmic. In the more general setting of computable Baire presentations, a weakened version of Lemma 3.6 still holds. Recall from Def. 2.7 that given a c.l.c. tree T, by E_T we denote the set of minimal code numbers u such that $\mathcal{K}_u$ is compact.

Lemma 7.5. Let G be computably t.d.l.c. via a computable Baire presentation $([T], Op, Inv)$.
(i) There is a computable function $I : E_T \to E_T$ 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 $\mathcal{K}_u \mathcal{K}_v \subseteq \mathcal{K}_w$.

Proof. (i) By Lemma 6.6 one can decide whether $\mathcal{K}_u \subseteq (\mathcal{K}_w)^{-1}$. The equality $\mathcal{K}_u = (\mathcal{K}_w)^{-1}$ is equivalent to $\mathcal{K}_u \subseteq (\mathcal{K}_w)^{-1} \wedge \mathcal{K}_w \subseteq (\mathcal{K}_u)^{-1}$. So one lets $I(u)$ be the least index v such that this equality holds.
(ii) Let $\widetilde{T}$ be the tree of initial segments of strings of the form $\sigma_0 \oplus \sigma_1$, where $\sigma_0, \sigma_1 \in T$ have the same length. Then $\widetilde{T}$ is a c.l.c. tree, $[\widetilde{T}]$ is naturally homeomorphic to $[T] \times [T]$, and Op can be seen as a computable function $[\widetilde{T}] \to [T]$. Now one applies Lemma 6.6. □

As discussed at the beginning of this section, the following adds to Theorem 5.1 a further equivalent condition for a t.d.l.c. group to be called computable.

Theorem 7.6.
A group G is computably t.d.l.c. via a Baire presentation $\Leftrightarrow$
G is computably t.d.l.c. via a meet groupoid.
From a presentation of G of one type, one can effectively obtain a presentation of G of the other type.

Proof. $\Leftarrow$: This implication follows from the corresponding implication in Theorem 5.1.
$\Rightarrow$: We build a Haar computable copy $\mathcal{W}$ of the meet groupoid $\mathcal{W}(G)$ as in Definition 4.5. By Lemma 7.5, one can decide whether $u \in E_T$ is the code number of a subgroup (Definition 2.5). Furthermore, one can decide whether $B = \mathcal{K}_v$ is a left coset of a subgroup $U = \mathcal{K}_u$: this holds iff $BU \subseteq B$ and $BB^{-1} \subseteq U$, and the latter two conditions are decidable by Lemma 7.5. Similarly, one can decide whether B is a right coset of U.

It follows that 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 (in other words, V is recursively enumerable). Hence, by a basic fact of computability theory, there is computable 1-1 function θ defined on an initial segment of $\mathbb{N} - \{0\}$ such that the range of θ equals this set. Write $A_n = \mathcal{K}_{\theta(n)}$ for $n > 0$, and $A_0 = \emptyset$.

The domain of $\mathcal{W}$ is all of $\mathbb{N}$. By Lemma 2.6 the intersection operation on $\mathcal{W}$ is computable, i.e., there is a computable binary function c on $\mathbb{N}$ such that $A_{c(n,k)} = A_n \cap A_k$. Next, given $n, k \in \mathbb{N} - \{0\}$ one can decide whether A_n is a right coset of the same subgroup that A_k is a left coset of. In that case, one can compute the number r such that $A_r = A_n \cdot A_k$: one uses that A_r is the unique coset C such that

(a) $A_n A_k \subseteq C$, and
(b) C is a right coset of the same subgroup that A_k is a right coset of.

As in the corresponding implication in the proof of Theorem 5.1, for subgroups U, V, one can compute $|U \colon U \cap V|$ by finding in $\mathcal{W}$ further and further distinct left cosets of $U \cap V$ contained in U, until their union reaches U. The latter condition is decidable. □

Definition 7.7. Given a computable Baire presentation G, by $\mathcal{W}_{\mathrm{comp}}(G)$ we denote the computable copy of $\mathcal{W}(G)$ obtained in the proof above.

Proposition 7.8. *Let p be a prime, and let $n \geq 2$. Let $\mathbb{Q}_p$ and $F_p((t))$ denote the rings of p-adic numbers, and Laurent series over $\mathbb{F}_p$, respectively. The t.d.l.c. groups $\mathrm{SL}_n(\mathbb{Q}_p)$ and $\mathrm{SL}_n(\mathbb{F}_p((t)))$ have computable Baire presentations.*

Proof. We provide the proof for the groups $\mathrm{SL}_n(\mathbb{Q}_p)$, and then indicate the changes that are necessary for the groups $\mathrm{SL}_n(\mathbb{F}_p((t)))$. We begin by giving a computable Baire presentation of $\mathbb{Q}_p$ as a ring.

Let Q be the tree of strings $\sigma \in \mathbb{N}^*$ such that all entries, except possibly the first, are among $\{0, \ldots, p-1\}$, and $r0 \not\preceq \sigma$ for each $r > 0$. We think of a string $r\sigma \in Q$ as denoting the rational $p^{-r} n_\sigma \in \mathbb{Z}[1/p]$, where n_σ is the number which has $\sigma 1$ as its p-ary expansion, written in reverse order:

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

The condition that $r0 \not\preceq \sigma$ for each $r > 0$ says that p doesn't divide n_σ.

For strings σ, τ over $\{0, \ldots, p-1\}$ of the same length ℓ, by $\sigma + \tau$ we denote the string ρ of the same length ℓ such that $n_\rho = n_\sigma + n_\tau \mod p^\ell$. By $\sigma \cdot \tau$ we denote the strings of length 2ℓ such that $n_\rho = n_\sigma n_\tau \mod p^{2\ell}$.

We now provide partial computable functions P, P_1, P_2, P_3 showing that various operations are computable according to Def. 6.2. For a string of the form $r\sigma$ let

$$P(r\sigma) = (r-k)\tau,$$

where $k \in \mathbb{N}$ is maximal such that $k \leq r$ and $0^k \preceq \sigma$, and τ is the rest, i.e. $0^k\tau = \sigma$.
To show the computability of the function $q \to -q$, let
$P_1(r\sigma) = r\tau$ where $|\tau| = |\sigma|$ and $\sigma + \tau = 0 \mod p^{|\sigma|}$.
To show that the addition operation is computable, let
$P_2(r\sigma, s\tau) = P(s(0^{s-r}\sigma + \tau))$ in case $r \leq s$, and
$P_2(r\sigma, s\tau) = P(r(\sigma + 0^{r-s}\tau))$ otherwise.
To show that the multiplication operation is computable, let
$P_3(r\sigma, s\tau) = P((2s)(0^{s-r}\sigma \cdot \tau))$ in case $r \leq s$, and
$P_3(r\sigma, s\tau) = P((2r)(\sigma \cdot 0^{r-s}\tau))$ otherwise.
P_3 is the correct partial recursive function because, say for $r \leq s$, in $\mathbb{Z}[1/p]$ one has

$$p^{-r}n_\sigma p^{-s}n_\tau = p^{-2s}n_{0^{s-r}\sigma}n_\tau.$$

We now provide a computable Baire presentation $([T], Op, Inv)$ of $\mathrm{SL}_n(\mathbb{Q}_p)$ as in Definition 7.1. Let T be the c.l.c. tree that is an n^2-fold "power" of Q. More precisely, $T = \{\sigma \colon \forall i < n^2\, [\sigma^i \in Q]\}$, where σ^i is the string of entries of σ in positions of the form kn^2+i for some $k, i \in \mathbb{N}$. Clearly, $[T]$ can be naturally identified with the matrix algebra $M_n(\mathbb{Q}_p)$. By the computability of the ring operations on $\mathbb{Q}_p$ as verified above, the matrix product is computable as a function $[T]\times[T] \to [T]$, and the function $\det \colon [T] \to [Q]$ is computable.

Note that for any c.l.c. trees T and R, any computable path f of R, and any computable function $\Phi \colon [T] \to [R]$, there is a computable subtree S of T such that $[S]$ equals the pre-image $\Phi^{-1}(f)$. (In the language of computable analysis, the pre-image is effectively closed.) To see this, suppose that Φ is computable according to Definition 6.2 via a partial computable function L defined on strings in T. Let S consists of the strings $\sigma \in T$ of length t such that t steps of the attempted computations $L(\tau)$ for all $\tau \preceq \sigma$ don't yield a contradiction to the hypothesis that the output of the oracle Turing machine equals f. More formally,

$$S = \{\sigma \in T \colon \forall \tau \preceq \sigma\, [L_t(\tau) \text{ is defined } \to L_t(\tau) \prec f]\}.$$

In our setting, applying this to the function $\det[T] \to [Q]$ and the path $f = 01000\ldots$ that denotes $1 \in \mathbb{Q}_p$, we obtain a computable subtree S of T such that $[S]$ can be identified with $SL_n(\mathbb{Q}_p)$.

It is well-known that $SL_n(\mathbb{Z}[1/p]$ is dense in $SL_n(\mathbb{Q}_p)$. This is a special case of strong approximation for algebraic groups (see [35, Ch. 7]), but can also be seen in an elementary way using Gaussian elimination. 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) of these paths. So by Proposition 6.8 we can replace S by a c.l.c. tree $\widetilde{S}$ such that $[\widetilde{S}] = [S]$.

To obtain a computable Baire presentation based on $\widetilde{S}$, note that matrix multiplication on $[\widetilde{S}]$ is computable as the restriction of matrix multiplication on $[T]$. To define the matrix inversion operation Inv, we use the fact that the inverse of a matrix with determinant 1 equals its adjugate matrix; the latter can be obtained by computing determinants on minors.

For the case of $SL_n(\mathbb{F}_p((t)))$, to give a Baire presentation of $\mathbb{F}_p((t))$ as a ring, in the proof above we use the same tree Q but think of a string $r\sigma$ as representing the finite Laurent series $t^{-r}n_\sigma \in \mathbb{Z}[1/p]$, where $n_\sigma = \sum_{i<|\sigma|} t^i\sigma(i)$. For strings σ, τ over $\{0, \ldots, p-1\}$ and of the same length ℓ, by $\sigma + \tau$ we denote the string ρ of the same length ℓ such that $n_\rho = n_\sigma + n_\tau \mod p^\ell$. By $\sigma \cdot \tau$ we denote the string ρ of

length 2ℓ such that $n_\rho \equiv n_\sigma n_\tau \mod t^{2\ell}$. Now the ring operations are computable via the same partial computable functions as for $\mathbb{Q}_p$. The remainder of the proof didn't use any particulars about the computable Baire presentation of the ring $\mathbb{Q}_p$ besides the fact that the path denoting 1 is computable, which carries over to the present case. It is also known that the matrices of determinant 1 over finite Laurent series, i.e., $SL_n(\mathbb{F}_p(t))$ form a subgroup that is dense in $SL_n(\mathbb{F}_p((t)))$. □

8. Computability for particular subclasses of t.d.l.c. groups

Our recent work [22], joint with Lupini, studies locally compact *abelian* groups with computable presentations. In particular we investigate the algorithmic content of Pontryagin– van Kampen duality for such groups. Each abelian t.d.l.c. group is procountable, i.e., an inverse limit of countable groups. This enabled us in [22] to use a notion of computable presentation for t.d.l.c. abelian groups convenient for the given context. This notion, which in the present paper we will call a *computably procountable presentation with effectively finite kernels*, is reviewed in Definition 8.3 below. The main purpose of this short section is to show that the definition of computably t.d.l.c. abelian groups in [22] is equivalent to the one given here.

8.1. **Procountable groups.** We review some concepts mainly from [22, Section 3]. Suppose we are given a sequence of groups $(\mathcal{A}_i)_{i\in\mathbb{N}}$ such that each A_i is countable discrete. Suppose we are also given epimorphisms $\phi_i : \mathcal{A}_i \to \mathcal{A}_{i-1}$ for each $i > 0$. Then $\varprojlim(\mathcal{A}_i, \phi_i)$ can concretely defined as the closed subgroup of the topological group $\prod_{i\in\mathbb{N}} A_i$ consisting of those g such that $\phi_i(g(i)) = g(i-1)$ for each $i > 0$.

Definition 8.1. A topological group G is called *procountable* if $G \cong \varprojlim(\mathcal{A}_i, \phi_i)$ for some sequence $(\mathcal{A}_i, \phi_i)_{i\in\mathbb{N}}$ as above.

The t.d.l.c. groups that are pro-countable are precisely the SIN groups (where SIN stands for "small invariant neighbourhoods); see [46, Section 2.2].

Remark 8.2. Let G be a closed subgroup of S_∞, let $\langle N_i\rangle_{i\in\mathbb{N}}$ be a descending sequence of open normal subgroups of G with trivial intersection, and let $\mathcal{A}_i = G/N_i$. Then $G \cong \varprojlim_{i>0}(\mathcal{A}_i, \phi_i)$ where the ϕ_i are the canonical maps. This is well-known; see [24, Lemma 2].

8.2. **Computably procountable groups.** Extending Definition 7.2 of computable profinite presentations, (2) below allows the $\mathcal{A}_i$ to be discrete computable groups, while retaining the condition that the kernels of the connecting maps be finite and given by strong indices.

Definition 8.3 ([22], Def. 3.4)**.**

(1) A computable presentation of a procountable group G is a sequence $(\mathcal{A}_i, \phi_i)_{i\in\mathbb{N}}$, of discrete groups $\mathcal{A}_i$ and epimorphisms $\phi_i : \mathcal{A}_i \to \mathcal{A}_{i-1}$ (for $i > 0$) such that $G \cong \varprojlim(\mathcal{A}_i, \phi_i)$, each group $\mathcal{A}_i$ is uniformly computable as a discrete group, and the sequence of maps $(\phi_i)_{i\in\mathbb{N}^+}$ is uniformly computable.

(2) Suppose that $ker\ \phi_i$ is finite for each i, so that G is locally compact by [22, Fact 3.2]. We say that $(\mathcal{A}_i, \phi_i)_{i\in\mathbb{N}}$ is a *computably procountable presentation with effectively finite kernels* if in addition, from i one can compute a strong index for $ker\ \phi_i$ as a subset of A_i.

Fact 8.4. *If G is compact and has a computable procountable presentation with effectively finite kernels, then this presentation is a computable profinite presentation.*

Proof. Since G is compact, the $\mathcal{A}_i$ are finite. All we need is strong indices for the $\mathcal{A}_i$ as groups. Since the $\mathcal{A}_i$ are computable groups uniformly in i, it suffices

to compute the size of $\mathcal{A}_i$ uniformly in i. One can do this recursively, using that $|\mathcal{A}_i| = |ker\, \phi_i| \times |\mathcal{A}_{i-1}|$. □

Proposition 8.5. *If a t.d.l.c. group G has a procountable presentation $(\mathcal{A}_i, \phi_i)_{i\in\mathbb{N}}$ with effectively finite kernels, then G has a computable Baire presentation.*

Proof. This is a straightforward extension of the argument in the profinite setting, Proposition 7.3. For simplicity we may assume that $\mathcal{A}_0$ is infinite. So we can effectively identify the elements of $\mathcal{A}_i$ with $\mathbb{N}$, and hence view ϕ_i as a map $\mathbb{N} \to \mathbb{N}$. As before,

$$T = \{\sigma \in \mathbb{N}^* \colon \forall i < |\sigma| [\sigma(i) \wedge \phi_i(\sigma(i)) = \sigma(i-1) \text{ if } i > 0\}.$$

It is clear from the hypotheses that $[\sigma]_T$ is compact iff σ is a nonempty string, and that T is effectively locally compact. The rest is as before, using that the $\mathcal{A}_i$ are computable groups uniformly in i. □

For abelian, as well as for compact, t.d.l.c. groups, we can provide a converse to the foregoing observation. The compact case essentially restates to [41, Thm. 1] for the recursively profinite case.

Proposition 8.6. *Let G be a computably t.d.l.c. group that is abelian or compact. Then G has a computably procountable presentation with effectively finite kernels.*

Proof. Suppose that the meet groupoid $\mathcal{W}(G)$ has a Haar computable copy $\mathcal{W}$ as in Definition 4.5. We may assume that its domain is all of $\mathbb{N}$, and that 0 denotes a (compact open) subgroup. In the framework of that copy one can compute a descending sequence $\langle U_i \rangle_{i\in\mathbb{N}}$ of compact open subgroups of G, such that U_0 is the group denoted by 0 and for each compact open U, there is an i with $U_i \subseteq U$. In the abelian case trivially each U_i is normal; in the case that G is compact, we effectively shrink the U_i so that they are also normal. To do so, by the hypothesis that $\mathcal{W}$ is Haar computable, we can compute from i a strong index for the set of $\{B_1, \dots, B_k\}$ of distinct right cosets of U_i. Replace U_i by $\bigcap_{r=1}^k B_r^{-1} \cdot U_i \cdot B_r$.

Let $\mathcal{A}_i$ be the automorphism group of the object U_i in the groupoid $\mathcal{W}$ viewed as a category; that is, $\mathcal{A}_i$ the set of cosets of U_i, with the groupoid operations restricted to it. For $i > 0$ let $\phi_i \colon \mathcal{A}_i \to \mathcal{A}_{i-1}$ be the map sending $B \in \mathcal{A}_i$ to the unique coset of U_{i-1} that contains B. Since $\mathcal{W}$ is Haar computable, condition (1) of Definition 8.3 holds. Also, the kernel of ϕ_i is the set of cosets of U_i contained in U_{i-1}. By Definition 4.5(b) we can compute the number of such cosets, so we can compute a strong index for the kernel. Hence condition (2) of Definition 8.3 also holds.

We complete the proof by verifying the following claim.

Claim 8.7. $G \cong \varprojlim_{i>0} (\mathcal{A}_i, \phi_i)$.

As in the proof of the implication "⇐" of Theorem 5.1, let $\widetilde{G}$ be the closed subgroup of S_∞ of elements p that preserve the inclusion relation on $\mathcal{W}$ in both directions, and satisfy $p(A) \cdot B = p(A \cdot B)$ whenever $A \cdot B$ is defined. Recall that $\widetilde{G} \cong G$. So it suffices to show that $\widetilde{G} \cong \varprojlim (\mathcal{A}_i, \phi_i)$.

Let $\mathcal{N}_i$ be the stabilizer of U_i, which is a compact open subgroup of $\widetilde{G}$. Since each $p \in \widetilde{G}$ preserves the inclusion relation, we have $\mathcal{N}_i \subseteq \mathcal{N}_{i-1}$ for $i > 0$. By the choice of the U_i, the intersection of the $\mathcal{N}_i$ is trivial. Each $\mathcal{N}_i$ is normal: if $p \in \mathcal{N}_i$ and $q \in \widetilde{G}$, let $B = q(U_i)$, a left, and hence also right, coset of U_i. Then

$$q^{-1}pq(U_i) = q^{-1}p(U_iB) = q^{-1}(U_iB) = q^{-1}(B)U_i = U_i.$$

So $q^{-1}pq \in \mathcal{N}_i$.

For each i, the map $\widetilde{G} \to \mathcal{A}_i$ sending p to $p(U_i)$ induces a group isomorphism $\widetilde{G}/\mathcal{N}_i \to \mathcal{A}_i$. For each $i > 0$ the natural map $\alpha_i \colon \widetilde{G}/\mathcal{N}_i \to \widetilde{G}/\mathcal{N}_{i-1}$ induced by the identity on $\widetilde{G}$ corresponds to ϕ_i via these isomorphisms. Thus $\varprojlim_{i>0}(\widetilde{G}/\mathcal{N}_i, \alpha_i) \cong \varprojlim_{i>0}(\mathcal{A}_i, \phi_i)$. The claim now follows in view of Remark 8.2. □

9. More on the equivalences of notions of computable presentation

This section obtains extra information from the proofs of the equivalences of our various notions of computable presentation of a t.d.l.c. group, Theorem 5.1 and its extension Theorem 7.6. Its main result, Thm. 9.2 will show that, given a computable Baire presentation $([T], Op, Inv)$ of a t.d.l.c. group G, one can effectively obtain a computable presentation of G via a closed subgroup $\widetilde{G}$ of S_∞, with an isomorphism $\Phi \colon [T] \to \widetilde{G}$ so that both Φ and Φ^{-1} are computable. The presentation based on a closed subgroup of S_∞ can be seen as an "improved" computable Baire presentation. For instance, the group operations on compact open subsets of $\widetilde{G}$ are fully computable by Lemma 3.6, while in the general case we only have the weaker form Lemma 7.5.

The section then proceeds to two applications of Thm. 9.2: computability of the modular function, and computability of Cayley-Abels graphs. For a compactly generated t.d.l.c. group G, its Cayley-Abels graphs are useful generalisations of the usual Cayley graphs in the setting of a (discrete) finitely generated group. We will show that the meet groupoid interprets each Cayley-Abels graph via first-order formulas with parameters. The interpretation is such that if G is computably t.d.l.c., the graphs are computable in a uniform way.

Section 10 uses Thm. 9.2 to bound the computational complexity of the scale function. As a further application, in Section 12 we obtain an equivalent criterion on whether a t.d.l.c. group G has a unique computable Baire presentation: any two Haar computable copies of its meet groupoid $\mathcal{W}(G)$ are computably isomorphic. This is useful because uniqueness of a computable presentation is easier to show for countable structures. In Theorem 12.6 we will apply the criterion to show uniqueness of a computable Baire presentation for the additive groups of the p-adic integers and the p-adic numbers, as well as for $\mathbb{Z} \ltimes \mathbb{Q}_p$.

Besides the computability theoretic concepts introduced in Section 6, we will need a fact on continous, open functions on the set of paths of c.l.c. trees. It deduces the computability of such a function from the hypothesis that its action on the compact open sets is computable in terms of their code numbers. Similar to the results in Section 6, this is a special case of a result in the field of computable topology; see [5, Lemma 2.13]. However, we prefer to present a short, elementary proof.

Proposition 9.1. *Let T and S be c.l.c. trees (see Def. 2.4), and let the function $\Phi \colon [T] \to [S]$ be continuous and open. Suppose that there is a computable function $f \colon \mathbb{N} \to \mathbb{N}$ such that for each code number u one has $\Phi(\mathcal{K}^T_u) = \mathcal{K}^S_{f(u)}$. Then*

(i) Φ is computable;

(ii) if Φ is a bijection then Φ^{-1} is computable as well.

Proof. (i) We define a partial computable function P_Φ on T with values in S. Given $\sigma \in T$ such that $[\sigma]_T$ is compact, compute u such that $\mathcal{K}^T_u = [\sigma]_T$. Let $\{\beta_1, \dots, \beta_s\} \subseteq S$ be the finite set with strong index $f(u)$. Let $\eta = P_\Phi(\sigma)$ be the longest common initial segment of the strings β_i. Note that with the standard ultrametric on $[S]$, the set $[\eta]_S$ is a closed ball centred at any element of $\mathcal{K}^S_{f(u)}$,

with the same diameter as $\mathcal{K}^S_{f(u)}$. So if $\sigma \preceq \tau \in T$ then $[P_\Phi(\sigma)]_S \supseteq [P_\Phi(\tau)]_S$. This shows that P_Φ is monotonic. Since Φ is continuous, one has

$$\Phi(f) = \bigcup_n \{P_\Phi(f\restriction_n)\colon\ f\restriction_n \in \mathrm{dom}(P_\Phi)\}$$

as required.
(ii) By Lemma 2.6 applied to S, there is a computable function g that is "inverse" to f in the sense such that for each v such that $\mathcal{K}^S_v$ is compact, one has $\Phi^{-1}(\mathcal{K}^S_v) = \mathcal{K}^T_{g(v)}$. Now one applies (i) to Φ^{-1} and g. □

Towards the main result of this section, let $G = ([T], Op, Inv)$ be a computable Baire presentation of a t.d.l.c. group. Let $\mathcal{W}$ be the computable copy of $\mathcal{W}(G)$ with domain $\mathbb{N}$ given by the proof of "⇒" in Thm. 7.6. Recall (Definition 7.7) that we write $\mathcal{W} = \mathcal{W}_{\mathrm{comp}}(G)$. Let $\widetilde{G} = \mathcal{G}_{\mathrm{comp}}(\mathcal{W})$ as in Definition 5.2. Let $\Phi\colon G \to \widetilde{G}$ be given by the action $(g, A) \mapsto gA$ as near the end of the proof of the implication "⇐" of Thm. 5.1.

Theorem 9.2. *Let $G, \mathcal{W}$, $\widetilde{G}$ and the homeomorphism $\Phi\colon G \to \widetilde{G}$ be as defined above.*

(i) Φ is computable, with computable inverse.
(ii) The action $[T] \times \mathbb{N} \to \mathbb{N}$, given by $(g, A) \mapsto gA$, is computable.

Proof. (i) Write $S = \mathit{Tree}(\widetilde{G})$. By Prop 9.1 it suffices to show that there is a computable function f such that $\Phi(\mathcal{K}^T_u) = \mathcal{K}^S_{f(u)}$ for each code number u.

We may assume that $\mathcal{K}^T_u$ is of the form $A \cap B$ where A is a left coset of a subgroup U such that B is a right coset of U: trivially a group G is the union of the right cosets of any given subgroup; hence such sets form a basis of the topology of G. And by Lemma 2.6, given a general compact set $\mathcal{K}^T_w$ one can effectively write it as a finite union of sets of this form.

If $\mathcal{K}^T_u$ is of the form $A \cap B$ as above, we have

$$\Phi(\mathcal{K}^T_u) = \{p \in \widetilde{G}\colon\ p(U) = A \wedge p^{-1}(U) = B\}.$$

Given any $F \in \mathcal{W}$, by Lemma 5.4 we can compute bounds on $p(F)$ and $p^{-1}(F)$ whenever $p \in \Phi(\mathcal{K}^T_u)$, letting $L = A$ and $L = B$ respectively. Recall the set E_T from Def. 2.7, and recall from the proof of "⇒" in Thm. 7.6 that $\theta\colon \mathbb{N} \to E_T$ is a 1-1 function with range the minimal code numbers of compact open cosets in G, and that we write A_n for the coset with code number $\theta(n)$, and often identify A_n with n. Suppose that $U = A_n$. Let $f(u)$ be a strong index for the finite set of strings $\beta \in S$ of length $2n + 2$ such that

(a) $A_{\beta(2n)} = A$ and $A_{\beta(2n+1)} = B$,
(b) for $r < 2n$ of the form $2i$, $\beta(r)$ is less than the bound on $p(F)$ given by Lemma 5.4 where $L = A$ and $F = A_i$.
(c) for $r < 2n$ of the form $2i + 1$, $\beta(r)$ is less than the bound on $p^{-1}(F)$ given by Lemma 5.4 where $L = B$ and $F = A_i$.

Then $\Phi(\mathcal{K}^T_u) = \mathcal{K}^S_{f(u)}$ as required.
(ii) By the observation after Def 6.2, the left action is computable iff Φ is computable. Alternatively, informally speaking, we use an oracle Turing machine that has as an oracle a path g on $[T]$, and as an input an $A \in \mathcal{W}$. If A is a left coset of a subgroup V, it outputs the left coset B of V such that it can find a string $\sigma \prec g$ with $[\sigma]_T A \subseteq B$. □

Note (ii) implies that the right action is also computable, using that $Ag = (g^{-1}A^{-1})^{-1}$ and inversion is computable both in G and in $\mathcal{W}$. We apply the theorem to obtain a computable version of the open mapping theorem for t.d.l.c.

groups in the case of a bijection. (This shows that the inverse of Φ in (i) of the theorem is in fact automatically computable.)

Corollary 9.3. *Let G, H be t.d.l.c. groups given by computable Baire presentations, and let $\Psi\colon G \to H$ be a computable bijection that is a group homomorphism. Then Ψ^{-1} is computable.*

Proof. Fix a compact open subgroup U of G. Since Ψ is open, $V = \Psi(U)$ is a compact open subgroup of H. Let $\langle a_i\rangle_{i\in\mathbb{N}}$ be a uniformly computable sequence of left coset representatives of U in G. (To obtain this, let $\langle b_k\rangle_{k\in\mathbb{N}}$ be a uniformly computable path of $\mathit{Tree}(G)$ extending the k-th string in $\mathit{Tree}(G)$, for some effective numbering of such strings. Let $\langle a_i\rangle_{i\in\mathbb{N}}$ be the subsequence obtained by deleting b_k if $b_kU = b_\ell U$ for some $\ell < k$.) Now $\langle \Psi(a_i)\rangle_{i\in\mathbb{N}}$ is a uniformly computable sequence of left coset representatives for V in H. By (ii) of the theorem, the sequences of (code numbers for) compact open cosets $K_i := a_iU$ and $S_i := \Psi(K_i) = \Psi(a_i)V$ are uniformly computable.

By Lemma 6.5(ii), we have a "local" computable inverse $\Theta_i\colon S_i \to K_i$ of the restriction $\Psi_i \restriction_{K_i}$, given by uniformly partial computable functions Q_i with arguments in $\mathit{Tree}(H)$ and values in $\mathit{Tree}(G)$, according to Definition 6.2. To show Ψ^{-1} is computable, intuitively, using a path $h \in H$ as an oracle, compute the i such that $h \in S_i$, and output $\Theta_i(h)$. More formally, define a partial computable function Q as follows: given $\sigma \in \mathit{Tree}(H)$, search for i such that $[\sigma]_{\mathit{Tree}(H)} \subseteq S_i$. If i is found, simulate $Q_i(\sigma)$ and give the corresponding output. □

In Subsection 1.1 we discussed the modular function $\Delta\colon G \to \mathbb{R}^+$. As our second application of Theorem 9.2, we show that for any computable presentation, the modular function is computable.

Corollary 9.4. *Let G be computably t.d.l.c. via a Baire presentation $([T], \mathit{Inv}, \mathit{Op})$. Then the modular function $\Delta\colon [T] \to \mathbb{Q}^+$ is computable.*

Proof. Using the notation of the foregoing theorem, let $V \in \mathcal{W}$ be any subgroup. Given $g \in [T]$, by (ii) of the theorem compute $A = gV$, and compute $U \in \mathcal{W}$ such that $A\colon U \to V$ (i.e., $A = Ug$). For any left Haar measure μ on G, we have

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

By Remark 4.6 we can choose μ computable; so this suffices to determine $\Delta(g)$. □

Our third application of the theorem is to show computability of the Cayley-Abels graphs, discussed in the introduction. Let G be a t.d.l.c. group that is compactly generated, i.e., algebraically generated by a compact subset. Then there is a compact open subgroup U, and a set $S = \{s_1, \dots, s_k\} \subseteq G$ 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

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

Some background and original references are given in Section 5 of [50]. For more detailed background see Part 4 of [47], or [19, Section 2]. If G is discrete (and hence finitely generated), then $\Gamma_{S,\{1\}}$ is the usual Cayley graph for the generating set S. Any two Cayley-Abels graphs of G is are quasi-isometric. See [19, Def. 3] or [47] for the formal definition.

Theorem 9.5. *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 $\mathcal{L}$. Given a Haar computable copy $\mathcal{W}$ of the meet groupoid $\mathcal{W}(G)$, one can obtain $\mathcal{L}$ effectively from $U \in \mathcal{W}$ and the left cosets $C_i = s_iU$, where $\{s_1, \dots, s_k\}$ is as above.

(ii) *If $\Gamma_{T,V}$ is another Cayley-Abels graph obtained as above, then $\Gamma_{S,U}$ and $\Gamma_{T,V}$ are computably quasi-isometric.*

(iii) *Given a computable Baire presentation of G based on a tree $[T]$, let $\mathcal{W} = \mathcal{W}_{comp}(G)$ be the computable copy of its meet groupoid as in Definition 7.7. Then the left action $[T] \times \mathcal{L} \to \mathcal{L}$ is also computable.*

Proof. (i) For the domain of the computable copy $\mathcal{L}$, we take the computable set of left cosets of U. We show that the edge relation is first-order definable from the parameters in such a way that it can be shown to be computable as well.

Let $V_i = C_i \cdot C_i^{-1}$ so that C_i is a right coset of V_i. Let $V = U \cap \bigcap_{1 \leq i \leq k} V_i$. To first-order define E_Γ in $\mathcal{W}$ with the given parameters, the idea is to replace the elements g in the definition of E_Γ by left cosets P of V, since they are sufficiently accurate approximations to g. It is easy to verify that $\langle A, B\rangle \in E_\Gamma \Leftrightarrow$

$$\exists i \leq k \exists P \in L(V) \exists Q \in L(V_i)\, [P \subseteq A \wedge P \subseteq Q \wedge B = Q \cdot C_i],$$

where $L(U)$ denotes the set of left cosets of a subgroup U: For the implication "$\Leftarrow$", let $g \in P$; then we have $A = gU$ and $B = gs_iU$. For the implication "$\Rightarrow$", given $A = gU$ and $B = gs_iU$, let $P \in L(V)$ such that $g \in P$.

We verify that the edge relation E_Γ is computable. Since $\mathcal{W}$ is Haar computable, by the usual enumeration argument we can obtain a strong index for the set of left cosets of V contained in A. Given P in this set and $i \leq k$, the left coset $Q = Q_{P,i}$ of V_i in the expression above is unique and can be determined effectively. So we can test whether $\langle A, B\rangle \in E_\Gamma$ by trying all P and all $i \leq k$ and checking whether $B = Q_{P,i} \cdot C_i$.

It is clear from the argument that we obtained $\mathcal{L}$ effectively from the parameters U and C_i.

(ii) (Sketch) First suppose that $V \subseteq U$. There is a computable map $\psi\colon L(U) \to L(V)$ such that $\psi(A) \subseteq A$. The proof of [19, Thm. 2^+] shows that $\psi\colon \Gamma_{S,U} \to \Gamma_{T,V}$ is a quasi-isometry. In the general case, let $R \subseteq G$ be a finite symmetric set such that $(U \cap V) \cup R$ algebraically generates G. There are computable quasi-isometries $\phi\colon \Gamma_{S,U} \to \Gamma_{R,U\cap V}$ and $\psi\colon \Gamma_{T,V} \to \Gamma_{R,U\cap V}$ as above. There is a computable quasi-isometry $\theta : \Gamma_{R,U\cap V} \to \Gamma_{T,V}$: given a vertex $y \in L(U \cap V)$, let $x = \theta(y)$ be a vertex in $L(V)$ such that $\psi(x)$ is at distance at most c from y, where c is a constant for ψ as above. Then $\theta \circ \phi$ is a quasi-isometry as required.

(iii) This follows immediately from Theorem 9.2(ii). □

10. Algorithmic properties of the scale function

In Subsection 1.1 we discussed the scale function $s\colon G \to \mathbb{N}^+$ for a t.d.l.c. group G, introduced by Willis [48]. Recall that for a compact open subgroup V and $g \in G$ one defines $m(g, V) = |V^g\colon V \cap V^g|$, and

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

Willis proved that the scale function is continuous, where $\mathbb{N}^+$ carries the discrete topology. He introduced the relation that a compact open subgroup V is *tidy for* g, and showed that this condition is equivalent to being minimizing for g in the sense that $s(g) = m(g, V)$. Möller [30] used graph theoretic methods to show that V is minimizing for g if and only if $m(g^k, V) = m(g, V)^k$ for each $k \in \mathbb{N}$. He also derived the "spectral radius formula": for any compact open subgroup U, one has $s(g) = \lim_k m(g^k, U)^{1/k}$.

For this section, fix a computable Baire presentation $([T], Op, Inv)$ of a t.d.l.c. group G as in Def. 7.1. Let $\mathcal{W} = \mathcal{W}_{\text{comp}}(G)$ be the Haar computable copy of $\mathcal{W}(G)$ given by Definition 7.7. Recall that the domain of $\mathcal{W}$ is $\mathbb{N}$. Via $\mathcal{W}$ we can identify

compact open cosets of G with natural numbers. The following is immediate from Theorem 7.6 and Theorem 9.2.

Fact 10.1. *The function $m\colon [T]\times\mathbb{N}\to\mathbb{N}$ (defined to be 0 if the second argument is not a subgroup) is computable.*

It is of interest to study whether the scale function, seen as a function $s\colon [T]\to\mathbb{N}$, is computable in the sense of Definition 6.1. We note that neither Möller's spectral radius formula, nor the tidying procedure of Willis (see again [50]) allow to compute the scale in our sense. The scale is computable iff one can algorithmically decide whether a subgroup is mimimizing:

Fact 10.2. *The scale function on $[T]$ is computable $\Leftrightarrow$ the following function Φ is computable in the sense of Def. 6.1: if $g\in[T]$ and V is a compact open subgroup of G, then $\Phi(g,V)=1$ if V is minimizing for g; otherwise $\Phi(g,V)=0$.*

Proof. $\Rightarrow$: An oracle Turing machine with oracle g searches for the first V that is minimizing for g, and outputs $m(g,V)$.
$\Leftarrow$: For oracle g, given input V check whether $m(g,V)=s(g)$. If so output 1, otherwise 0. □

We next provide an upper bound on the complexity of the scale function. We say that a function $\Psi:[T]\to\mathbb{N}$ is *computably approximable from above* if 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

$$\Psi(f)=k \text{ iff } \lim_r\Theta(f,r)=k.$$

Fact 10.3. *The scale function is computably approximable from above.*

Proof. Let $\Theta(f,r)$ be the minimum value of $m(f,s)$ over all $s\leq r$. □

The following example is well-known ([50, Example 2]); we include it to show that our framework is adequate as a general background for case-based approaches used in earlier works.

Example 10.4 (with Stephan Tornier)**.** For $d\geq 3$, the scale function on $Aut(T_d)$ in the computable presentation of Example 3.5 is computable.

Proof. An automorphism g of T_d has exactly one of three types (see [9]):

(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 the set-wise stabilizer of the set of endpoints of this edge.
(3) g translates along a geodesic (a subset of T_d that is a homogeneous tree of degree 2): then $s(g)=(d-1)^\ell$ where ℓ is the length. To see this, for $\ell=1$ one uses as a minimizing subgroup the compact open subgroup of automorphisms that fix two given adjacent vertices on the axis. For $\ell>1$ one uses that $s(r^k)=s(r)^k$ for each k and $r\in Aut(T_d)$; see again [48].

The oracle machine with oracle a path corresponding to $g\in Aut(T_d)$ searches, in parallel, for a witness for (1), a witness for (2), or a sufficiently long piece of the axis in (3) so that the shift becomes visible. It then outputs the corresponding value of the scale. □

11. Closure properties of the class of computably t.d.l.c. groups

All computable presentations in this section will be Baire presentations (see Definition 7.1), and we will usually view a t.d.l.c. group G concretely as a computable

Baire presentation. Extending the previous notation in the setting of closed subgroups of S_∞, by $Tree(G)$ we denote the c.l.c. tree underlying this computable Baire presentation. The following is immediate.

Fact 11.1 (Closure under computable closed subgroups)**.** *Let G be a computably t.d.l.c. group. Let H be a closed subgroup of G (so that $Tree(H)$ is a subtree of $Tree(G)$). Suppose that $Tree(H)$ is c.l.c. Then H is computable t.d.l.c. via the Baire presentation based on $Tree(H)$, with the operations of G restricted to H.*

For instance, consider the closed subgroups $U(F)$ of $Aut(T_d)$, where $d \geq 3$ and F is a subgroup of S_d. They were introduced by Burger and Mozes [2]. By Example 3.5 together with the preceding fact, each group $U(F)$ is computably t.d.l.c.

For another example, consider the computable Baire presentation of $\mathrm{SL}_2(\mathbb{Q}_p)$ in Proposition 7.8. Let S be the c.l.c. subtree of T whose paths describe matrices of the form $\begin{pmatrix} r & 0 \\ 0 & s \end{pmatrix}$ (so that $s = r^{-1}$). This yields a computable Baire presentation of the group $(\mathbb{Q}_p^*, \cdot)$.

Example 11.2. For each prime p and $n \geq 2$, the group $\mathrm{GL}_n(\mathbb{Q}_p)$ is computably t.d.l.c.

Proof. We employ the embedding $F\colon GL_n(\mathbb{Q}_p) \to SL_n(\mathbb{Q}_p)$ which extends a matrix A to the matrix B where the new row and column vanish except for the diagonal element (which necessarily equals $(\det A)^{-1}$). Clearly there is a c.l.c. subtree S of the c.l.c. subtree of T in Proposition 7.8 for $n+1$ such that $[S] = \mathrm{range}(F)$. Now we apply Fact 11.1. □

A further construction staying within the class of t.d.l.c. groups is the semidirect product based on a continuous action. In the effective setting, we use actions that are computable in the sense of Section 6. For computable actions in the more general context of Polish groups see [27].

Proposition 11.3 (Closure under computable semidirect products)**.** *Let G, H be computably t.d.l.c. groups. Suppose $\Phi\colon G \times H \to H$ is a computable function (Definition 6.2) 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.*

Proof. Let T be the tree obtained by interspersing strings of the same length from the trees of G and H, i.e.

$$T = \{\sigma \oplus \tau\colon\ \sigma \in Tree(G) \wedge \tau \in Tree(H)\}.$$

It is clear that T is a c.l.c. tree. Via the natural bijection

$$[T] \to [Tree(G)] \times [Tree(H)],$$

one can write elements of L in the form $\langle g, h\rangle$ where $g \in [Tree(G)]$ and $h \in [Tree(H)]$.

By the usual definition of semidirect product ([38, p. 27]), writing the operations for G and H in the usual group theoretic way, we have

$$\begin{aligned} Op(\langle g_1, h_1\rangle, \langle g_2, h_2\rangle) &= \langle g_1 g_2, \Phi(g_2, h_1) h_2\rangle \\ Inv(\langle g, h\rangle) &= \langle g^{-1}, (\Phi(g^{-1}, h))^{-1}\rangle. \end{aligned}$$

This shows that Op and Inv are computable, and hence yields a computable Baire presentation $([T], Op, Inv)$ for L. □

Remark 11.4. The foregoing proposition leads to a different proof that $\mathrm{GL}_n(\mathbb{Q}_p)$ is computably t.d.l.c. (Example 11.2). To simplify notation we let $n = 2$; it is not hard to generalise the argument below to a general n. As mentioned after Fact 11.1 there is a computable Baire presentation of $(\mathbb{Q}_p^*, \cdot)$. We have $\mathrm{GL}_2(\mathbb{Q}_p) = \mathbb{Q}_p^* \ltimes_\Phi \mathrm{SL}_2(\mathbb{Q}_p)$

via the inclusion embedding of $\mathrm{SL}_2(\mathbb{Q}_p)$, the embedding $q \to \begin{pmatrix} q & 0 \\ 0 & 1 \end{pmatrix}$ of $\mathbb{Q}_p^*$, and the computable action $\Phi(q, \begin{pmatrix} a & b \\ c & d \end{pmatrix}) = \begin{pmatrix} a & q^{-1}b \\ qc & d \end{pmatrix}$.

Note that the two computable Baire presentations of $GL_2(\mathbb{Q}_p)$ obtained above are computably isomorphic: one maps $(q, \begin{pmatrix} a & b \\ c & d \end{pmatrix})$ to $\begin{pmatrix} qa & qb & 0 \\ c & d & 0 \\ 0 & 0 & q^{-1} \end{pmatrix}$.

Given a sequence of t.d.l.c. groups $(G_i)_{i\in\mathbb{N}^+}$, the direct product $\prod_{i\in\mathbb{N}^+} G_i$ is not t.d.l.c. in general. In [46, Def. 2.3] a local direct product is described that retains the property of being t.d.l.c. This construction depends on the choices of compact open subgroups U_i of G_i, for each i: let $G = \bigoplus_i (G_i, U_i)$ consist of the elements $f \in \prod_i G_i$ such that $f(i) \in U_i$ for sufficiently large i. We have $G = \bigcup_k H_k$ where $H_0 = \prod_i U_i$ and for $k > 0$, $H_k = G_1 \times \ldots \times G_k \times \prod_{i>k} U_i$. The H_k are equipped with the product topology. A set $W \subseteq G$ is declared open if $W \cap H_k$ is open for each k. (In particular, $\prod_i U_i$ is a compact open subgroup.)

Fix a computable bijection $\langle ., . \rangle \colon \mathbb{N} \times \mathbb{N}^+ \to \mathbb{N}$ such that $\langle a, b\rangle \geq \max(a, b)$. For a string $\sigma \in \mathbb{N}^*$ and $i > 0$, by $\sigma^{(i)}$ we denote the string τ of maximum length such that $\tau(k) = \sigma(\langle k, i\rangle)$ for each $k < |\tau|$. Similarly, for $f \colon \mathbb{N} \to \mathbb{N}$ we define $f^{(i)}$ to be the function such that $f^{(i)}(k) = f(\langle k, i\rangle)$ for each k. Given uniformly computable subtrees B_i of $\mathbb{N}^*$, by $B = \prod_i B_i$ we denote the computable tree $\{\sigma \colon \forall i[\sigma^{(i)} \in B_i]\}$. Note that $[B]$ is canonically homeomorphic to $\prod_i [B_i]$ via $f \to (f^{(i)})_{i\in\mathbb{N}}$. So we can specify a path f of B by specifying all the $f^{(i)}$.

Proposition 11.5 (Closure under local direct products)**.** *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.*

Proof. By our convention for this section, the G_i are computable Baire presentations (T_i, Op_i, Inv_i). The uniformity hypothesis on the U_i can be made explicit as follows: there is a computable function q such that $\mathcal{K}^{T_i}_{q(i)} = U_i$. We use these data to build a computable Baire presentation $([T], Op, Inv)$ of G. We aim at defining uniformly c.l.c. trees V_k such that as topological spaces, $[V_0]$ is homeomorphic to H_0 defined above, and for $k > 0$, V_k is homeomorphic to $H_k - H_{k-1}$. All these homeomorphism are canonically given, as can be seen during the construction of the V_k. The c.l.c. tree that our Baire presentation of G is based on will be

$$T = \{k\sigma \colon k \in \mathbb{N} \wedge \sigma \in V_k\}.$$

It is easy to verify that $[T]$ is homeomorphic to G, using that the H_k are clopen subgroups of G.

Towards defining the trees V_k, let R_i be the subtree of T_i such that $[R_i] = U_i$, and let S_i be the subtree of T_i such that $[S_i] = [T_i] - U_i$.

Claim 11.6. *The trees R_i, S_i are c.l.c. trees uniformly in $i \in \mathbb{N}^+$.*

To check this, it suffices to show that these trees are uniformly computable. Given i, let $F_i \subseteq \mathbb{N}^*$ be the finite set with strong index $q(i)$. Note that R_i consists of the strings compatible with a string in F_i, which is a decidable condition uniformly in i. To determine whether $\tau \in S_i$, first check whether some prefix of τ is in F; if so answer "no". Otherwise, using the conditions defining c.l.c. trees check whether $[\tau]_{T_i}$ is compact; if not answer "yes". If so, check whether τ has an extension longer that any string in F_i that is not in S_i; if so answer "yes", otherwise "no". This shows the claim.

Now let $V_0 = \prod_i R_i$, and for $k > 0$ let

$$V_k = \prod_{1\le i<k} T_i \times S_k \times \prod_{i>k} R_i,$$

interpreted as subtrees of $\prod_i T_i$ in the obvious way. (So, f is a path of V_k iff $f^{(i)}$ is a path of T_i for $1 \le i < k$, $f^{(k)}$ is *not* in U_k, but $f^{(i)}$ is in U_i for $i > k$.) It is easy to check that the V_k are uniformly c.l.c.

Uniformly in k, on $[V_k]$ we have a computable function L_k given by $L_k(f)^{(i)} = Inv_i(f^{(i)})$. So there is a computable function Inv on $[T]$ given by

$$Inv(kf) = kL_k(f).$$

Next, for $r, s \in \mathbb{N}$ and $rf, sg \in [T]$, let

$$Op(rf, sg) = th,$$

where h is the function given by $h^{(i)} = Op_i(f^{(i)}, g^{(i)})$, and $t \le \max(r, s)$ is the least number such that $t = 0$, or $t > 0$ and $h^{(t)} \restriction_{\max(F_t)} \in S_t$. That is, we compute the binary group operation componentwise, and then check which tree V_t the overall result is a path of; this can be done because from $\max(r, s)$ on, the component of the result will be in the relevant compact open subgroup.

It should be clear that via the homeomorphism of the spaces $[T]$ and G outlined above, $([T], Op, Inv)$ is a computable Baire presentation of G. □

Remark 11.7. The foregoing result might turn out to be useful for answering Question 1.5 in the negative. Suppose that uniformly in $i \in \mathbb{N}^+$ one can build a computably t.d.l.c. group G_i and computable element g_i so that $s(g_i)$ depends in some predetermined way on whether i is in the halting set $\mathcal{K}$; for instance, let $s(g_i) = 2$ if $i \notin \mathcal{K}$, and $s(g_i) = 1$ else. Let U_i be some uniformly determined compact open subgroup of G_i. Then the scale function on the t.d.l.c. group $G = \bigoplus_{i\in\mathbb{N}^+}(G_i, U_i)$, with the computable presentation given above, is non-computable.

It takes some effort to prove that being computably t.d.l.c. is preserved under taking quotients by computable normal closed subgroups. As an application we will show that the groups $PGL_n(\mathbb{Q}_p)$ are computably t.d.l.c. For $n = 2$ these groups have been the subject of much research; for instance, they are homeomorphic to closed subgroups of $Aut(T_{p+1})$ as shown by Serre [40, Section II.1].

First we need some notation and preliminaries. The variables α, β etc. will range over strings in $\mathbb{N}^*$ without repetitions. The variables P, Q, R range over permutations of $\mathbb{N}$. Recall from Section 3.1 that in our setup the elements of S_∞ have the form $P \oplus P^{-1}$ where P is a permutation of $\mathbb{N}$. It appears that a crucial "finitization" argument, Claim 11.12, is best shown in the setting of true permutations, rather than elements of S_∞. So we need two lemmas allowing us to pass back and forth between the two pictures.

We adopt the setting of Theorem 9.2. So let $G = ([T], Op, Inv)$ be a computable Baire presentation, let $\mathcal{W}$ be a Haar computable copy of $\mathcal{W}(G)$ with domain $\mathbb{N}$, and let $\widetilde{G}$ be as detailed there; in particular, $S = Tree(\widetilde{G})$ is c.l.c., and $[\sigma]_S$ is compact for each string σ of length ≥ 2. Let

$$S_0 = \{\emptyset\} \cup \{\alpha\colon \exists\beta[|\alpha| = |\beta| > 0 \wedge \alpha \oplus \beta \in S\}.$$

The first lemma verifies that S_0 is computable, and takes a nonempty string in S_0 to the set of elements of $[S]$ with first component extending it.

Lemma 11.8. (i) S_0 is a computable tree.
(ii) There is a computable function $B\colon S_0 - \{\emptyset\} \to \mathbb{N}$ such that

$$\mathcal{K}_{B(\alpha)} = \{f \in [S]\colon \alpha \prec f_0 \text{ where } f = f_0 \oplus f_1\}.$$

Proof. (i) We first show that there is a computable bound $H(k)$ on $\beta(k)$ that is uniformly obtained from α. Since α is nonempty, we have $\alpha(A) = B$ for some

$A, B \in \mathcal{W}$. Hence $f(AA^{-1}) = BA^{-1}$ for any $f \in \widetilde{G}$ such that $\alpha \prec f$. By Lemma 5.4 with $U = AA^{-1}$, $L = BA^{-1}$ and F the coset denoted by k, we obtain a bound $H(k)$ as required. This shows that from α one can compute a finite set of possible candidates for β. So S_0 is computable.

(ii) Given a nonempty $\alpha \in S_0$, by the argument above we can compute a strong index $B(\alpha)$ for the set of strings of the form $\alpha \oplus \beta$ in S. □

The second lemma takes a nonempty string in S and writes the paths of S extending it in terms of a finite set of strings in S_0.

Lemma 11.9. Given $\sigma \in S$ such that $|\sigma| \geq 2$, one can compute (a strong index for) a finite set $F \subseteq S_0$ such that

$$[\sigma]_S = \bigcup\{\mathcal{K}_{B(\alpha)} \colon \alpha \in F\}.$$

Proof. Let $n = \max(\sigma) + 1$. Let F consist of the strings $\alpha \in S_0$ of length n such that viewed as an injection, α extends the injection α_σ associated with σ as in (1). Since $|\sigma| \geq 2$, by Lemma 5.4 we can compute a strong index for F. If $\alpha \in F$ and $P \succ \alpha$, then $P \oplus P^{-1} \in [\sigma]_S$, because P extends α_σ. Conversely, if $P \oplus P^{-1} \in [\sigma]_S$, then $P \upharpoonright_n \in F$. □

Theorem 11.10 (Closure under quotients by computable closed normal subgroups)**.** *Let G be computably t.d.l.c. 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.*

Proof. We continue to adopt the setting of Theorem 9.2. Recall that $S = \mathit{Tree}(\widetilde{G})$, and $S_0 = \{\emptyset\} \cup \{\alpha \colon \exists \beta[|\alpha| = |\beta| > 0 \wedge \alpha \oplus \beta \in S\}$. Let $M = \Phi(N)$ where Φ is the bicomputable homeomorphism $G \to \widetilde{G}$ established in Theorem 9.2. Note that $\mathit{Tree}(M)$ is a computable subtree of S; given $\tau \in S$, one can search for a string $\sigma \in \mathit{Tree}(G)$ such that $P_\Phi(\sigma) \succeq \tau$; then $\tau \in \mathit{Tree}(M)$ iff $\sigma \in \mathit{Tree}(N)$.

We will build a Haar computable copy $\mathcal{V}$ of $\mathcal{W}(\widetilde{G}/M)$. We use that each compact open subset of $\widetilde{G}/M$ has the form $M\mathcal{K}$ where $\mathcal{K}$ is a compact open subset of $\widetilde{G}$. In the first step, we will show that the preordering $\mathcal{K} \subseteq M\mathcal{L}$ is decidable, where $\mathcal{K}, \mathcal{L}$ are compact open sets. In the second step, we will use as the domain of $\mathcal{V}$ the least numbers in the classes of the computable equivalence relation associated with this pre-ordering.

We may assume that $\mathcal{K} = [\sigma]_S$ for some string $\sigma \in S$. So the following suffices for the first step:

Lemma 11.11. Given strings $\sigma, \tau_1, \ldots, \tau_r \in S$ of length at least 2, one can decide whether the inclusion $[\sigma]_S \subseteq \bigcup_i M[\tau]_i$ holds.

To verify the lemma, let $M_0 = \{P \colon P \oplus P^{-1} \in M\}$. For each $\alpha \in S_0$, let

$$\underline{\alpha} = \{P \in [S_0] \colon \alpha \prec P\}$$

(recall here that P is a permutation of $\mathbb{N}$, not merely a path of S_0, which in general could fail to be onto). Let

$$T_0 = \{\alpha \colon \underline{\alpha} \cap M_0 \neq \emptyset\}.$$

By Lemma 11.9, it is sufficient to decide whether a version of the inclusion holds that only refers to the permutations, not directly to their inverses.

Claim 11.12. *For $\alpha, \beta_1, \ldots, \beta_k \in S_0$ one can decide whether $\underline{\alpha} \subseteq \bigcup_i M_0\underline{\beta_i}$.*

We may assume that $|\alpha| \geq m := 1 + \max_i \max(\beta_i)$. We show that m is the maximum height on T_0 relevant for this inclusion: for each i,

$$\exists Q \in M_0\, [\underline{\alpha} \subseteq Q \circ \underline{\beta_i}] \Leftrightarrow \exists \eta \in T_0\, [|\eta| = m \wedge \underline{\alpha} \subseteq \underline{\eta} \circ \underline{\beta_i}]. \tag{3}$$

For the implication "$\Rightarrow$", simply let $\eta = Q \restriction_m$. For the implication $\Leftarrow$, fix a permutation $Q \succ \eta$ such that $Q \in M_0$. Given $P \in \underline{\alpha}$, we can choose $R \in \underline{\eta}$ and $R' \in \underline{\beta_i}$ such that $P = R \circ R'$. Since $R, Q \succ \eta$ and $|\eta| = m > \max(\beta_i)$, we have $Q^{-1} \circ R \circ R' \succ \beta_i$. So $P \in M_0 \circ \underline{\beta_i}$ via Q. This verifies the equivalence (3).

Now, if $\underline{\alpha} \subseteq \bigcup_i M_0 \circ \underline{\beta_i}$, then because $|\alpha| > \max |\beta_i|$ for each i, we have $\underline{\alpha} \subseteq M_0 \circ \underline{\beta_i}$ for some single i. If η works on the right hand side of (3) then $\eta(\beta_i(0)) = \alpha(0)$. So by Lemma 5.4 again, there is a computable bound $H(k)$ on $\eta(k)$ that is uniformly obtained from the values $\alpha(0), \beta_1(0), \ldots, \beta_k(0)$. This shows that one can compute a strong index for a finite set containing all potential witnesses η on the right hand side of (3). For each such η, using Lemma 11.8(ii), it is equivalent to decide whether $\mathcal{K}_{B(\alpha)} \subseteq \mathcal{K}_{B(\eta)}\mathcal{K}_{B(\beta_i)}$, which can be done using Lemma 7.5(ii). This shows the claim and hence verifies Lemma 11.11.

By the Lemma (and the discussion preceding it), the equivalence relation on $\mathbb{N}$ given by

$$A \sim B \text{ if } AM = BM$$

is computable; recall here that $\mathcal{W}$ has domain $\mathbb{N}$. For the domain $\mathcal{D}$ of the computable copy $\mathcal{V}$ of $\mathcal{W}(\widetilde{G}/M)$, we use the computable set of least elements of equivalence classes.

We think of an element A of $\mathcal{D}$ as denoting the compact open coset AM of $\widetilde{G}/M$. Given $A, B \in \mathcal{W}$ we have $(AM)^{-1} = A^{-1}M$ and $(AM)(BM) = (AB)M$. In particular, one can decide whether AM, viewed as a subset of $\widetilde{G}/M$, is a left coset of a subgroup of $\widetilde{G}/M$ that BM is a right coset of. So by Lemma 3.6 the groupoid operations are computable on D.

For the computability of the meet operation, suppose $A, B \in \mathcal{W}$ are given. One has $AN \cap BN = (A \cap BN)N$. Suppose A is a left coset of the subgroup U and B is a left coset of the subgroup V. Then $A \cap BN$ is a left coset of the compact open subgroup $U \cap VN$. Note that

$$A \cap BN = \bigcup \{L\colon\ L \subseteq A \wedge L \text{ is left coset of } U \cap V \wedge L \sim B.\}$$

(The inclusion $\supseteq$ is trivial. For $\subseteq$, if $x \in A \cap BN$, then $x \in L$ for some left coset of $U \cap V$. Since $x = bn$ for some $b \in B, n \in M$ we have $LN = BN$.) So one can compute $C \in \mathcal{W}$ such that $C = A \cap BN$ using Lemma 2.6. Then one outputs the element C' of $\mathcal{D}$ such that $C' \sim C$. Finally, to show that $\mathcal{V}$ is Haar computable, note that $|UN : UN \cap VN| = |U\colon U \cap VN|$. By the above (in the special case that A and B are subgroups) one can compute $U \cap VN \in \mathcal{W}$. So one can compute the index using that $\mathcal{W}$ is Haar computable. □

Example 11.13. For each prime p and each $n \geq 2$, the group $\mathrm{PGL}_n(\mathbb{Q}_p)$ is computably t.d.l.c.

Proof. We use the computable Baire presentation (T, Op, Inv) of $\mathrm{GL}_n(\mathbb{Q}_p)$ obtained in Example 11.2. 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 $Tree(N)$ is a computable subtree of the tree S in Example 11.2. Hence we can apply Theorem 11.10. □

12. Uniqueness of computable presentations

As discussed in Subsection 1.7, in computable structure theory a countable structure is called autostable if it has a computable copy, and all its computable copies are computably isomorphic. We adapt this notion to the present setting.

Definition 12.1. A computably t.d.l.c. group G is called *autostable* if for any two computable Baire presentations of G, based on trees $T, S \subseteq \mathbb{N}^*$, there is a computable group homeomorphism $\Psi\colon [T] \to [S]$.

Note that Ψ^{-1} is also computable by Corollary 9.3. For abelian profinite groups, the notion of autostability used in [26] is equivalent to our definition. This follows from the proofs of Prop. 8.5 and Prop. 8.6, which show that in the abelian case the correspondence between Baire presentations and procountable presentations is uniform and witnessed by uniformly obtained group-isomorphisms between these presentations. The first author [26, Cor. 1.11] characterizes autostability for abelian compact pro-p groups given by computable procountable presentations with effectively finite kernels: such a group is autostable iff its Pontryagin - van Kampen dual is autostable. For instance, $(\mathbb{Z}_p, +)$ is autostable because its dual is the Prüfer group C_{p^∞}, which is easily seen to be autostable as a countable structure (see the proof of Theorem 12.6 below).

We now provide a criterion for autostability, and show its usefulness through various examples.

Criterion 12.2. *A computably t.d.l.c. group G is autostable $\Leftrightarrow$ any two Haar computable copies of its meet groupoid $\mathcal{W}(G)$ are computably isomorphic.*

We will only apply the implication "$\Leftarrow$". However, the converse implication is interesting on its own right because it shows that our notion of autostability is independent of whether we use computable Baire presentation, or computable presentations based on meet groupoids. In fact, the proof of the converse implication shows that we could also use presentations based on closed subgroups of S_∞.

Proof. We may assume that G itself is a computable Baire presentation. As before, let $\mathcal{W} = \mathcal{W}_{\mathrm{comp}}(G)$ as in Definition 7.7, and let $\widetilde{G} = \mathcal{G}_{\mathrm{comp}}(\mathcal{W})$ as in Definition 5.2. Let $\Phi\colon G \to \widetilde{G}$ be the group homeomorphism given by Theorem 9.2.

$\Leftarrow$: Let H be a computable Baire presentation such that $G \cong H$. Let $\mathcal{V} = \mathcal{W}_{\mathrm{comp}}(H)$, and let $\widetilde{H} = \mathcal{G}_{\mathrm{comp}}(\mathcal{V})$. Clearly $G \cong H$ implies $\mathcal{W} \cong \mathcal{V}$. So there is a computable isomorphism $\theta\colon \mathcal{W} \to \mathcal{V}$; note that θ is a permutation of $\mathbb{N}$, so θ^{-1} is also computable. We define a computable homeomorphism $\widetilde{\theta}\colon \widetilde{G} \to \widetilde{H}$ as follows. Given $p = f \oplus f^{-1} \in \widetilde{G}$, let

$$\widetilde{\theta}(p) = (\theta \circ f \circ \theta^{-1} \oplus \theta \circ f^{-1} \circ \theta^{-1}). \tag{4}$$

Using that θ is an isomorphism of meet groupoids, it is easy to verify that $\widetilde{\theta}$ is a homeomorphism. Clearly $\widetilde{\theta}$ is computable. The inverse of $\widetilde{\theta}$ is given by exchanging θ and θ^{-1} in the above, and hence is computable as well.

Let $\Psi\colon H \to \widetilde{H}$ be the homeomorphisms given by Theorem 9.2 with H in place of G. We have a group homeomorphism $\Psi^{-1} \circ \widetilde{\theta} \circ \Phi\colon G \to H$, which is computable as a composition of computable maps. Also, its inverse is computable because the inverse of Γ is computable.

$\Rightarrow$: Suppose $\mathcal{V}$ is a Haar computable meet groupoid such that $\mathcal{W} \cong \mathcal{V}$ via an isomorphism β. We need to show that $\mathcal{W}, \mathcal{V}$ are computably isomorphic. To this end, let $\widetilde{H} = \mathcal{G}_{\mathrm{comp}}(\mathcal{V})$. Let $\widetilde{\mathcal{W}} = \mathcal{W}_{\mathrm{comp}}(\widetilde{G})$ and $\widetilde{\mathcal{V}} = \mathcal{W}_{\mathrm{comp}}(\widetilde{H})$. Let $\alpha_{\mathcal{W}}\colon \mathcal{W} \to \widetilde{\mathcal{W}}$ and $\alpha_{\mathcal{V}}\colon \mathcal{V} \to \widetilde{\mathcal{V}}$ be the maps given by

$$\alpha_{\mathcal{W}}(A_U) = \{p \in \widetilde{G}\colon\ p_0(U) = A\} \text{ and } \alpha_{\mathcal{V}}(B_U) = \{q \in \widetilde{H}\colon\ q_0(U) = B\},$$

where the notation A_U indicates that A is a left coset of U.

Informally, $\widetilde{\mathcal{W}}$ is the double dual of $\mathcal{W}$, and $\alpha_{\mathcal{W}}$ maps $\mathcal{W}$ into its double dual. The following is not true in general, but holds because $\mathcal{W} \cong \mathcal{W}(G)$ for the t.d.l.c. group G.

Claim 12.3. *$\alpha_{\mathcal{W}}$ is a bijection, and hence a meet groupoid isomorphism.*

Since $\Phi\colon G \to \widetilde{G}$ is a group homeomorphism, its dual $\widetilde{\Phi}\colon \mathcal{W} \to \widetilde{\mathcal{W}}$ is a meet groupoid isomorphism, where $\widetilde{\Phi}(A) = \{\Phi(g)\colon\ g \in A\}$. Now note that

$$\widetilde{\Phi}(A_U) = \alpha_{\mathcal{W}}(A_U)\colon$$

if $g \in A$ then clearly $\Phi(g)(U) = gU = A$, so $\Phi(g) \in \alpha_{\mathcal{W}}(A)$. Conversely, suppose $p_0(U) = A$ for $p \in \widetilde{G}$, and let $g = \Phi^{-1}(p)$. Then $g \in A$ because $\Phi(g)(U) = p(U) = A$. This verifies the claim.

Let $\Gamma\colon \widetilde{\mathcal{W}} \to \widetilde{\mathcal{V}}$ be the "double dual" of β; that is, $\Gamma(A) = \{\widetilde{\beta}(p)\colon\ p \in A\}$, where $\widetilde{\beta}$ is the dual of β defined as in Eq. (4). Note that Γ is an isomorphism of meet groupoids.

Claim 12.4. *The diagram*

$$\begin{array}{ccc} \widetilde{\mathcal{W}} & \xrightarrow{\Gamma} & \widetilde{\mathcal{V}} \\ {\scriptstyle \alpha_{\mathcal{W}}}\uparrow & & \uparrow{\scriptstyle \alpha_{\mathcal{V}}} \\ \mathcal{W} & \xrightarrow[\beta]{} & \mathcal{V} \end{array}$$

commutes. Hence $\alpha_{\mathcal{V}}$ is an isomorphism of meet groupoids.

To see this, if $A_U \in \mathcal{W}$, then

$$\begin{aligned} \Gamma(\alpha_{\mathcal{W}}(A)) &= \{p^\beta\colon p \in \widetilde{G} \wedge\colon p_0(U) = A\} \\ &= \{q \in \widetilde{H}\colon q_0(\beta(U) = \beta(A)\} \\ &= \alpha_{\mathcal{V}}(\beta(A)), \end{aligned}$$

where $p^\beta := \beta \circ f \circ \beta^{-1} \oplus \beta \circ f^{-1} \circ \beta^{-1}$, for $p = f \oplus f^{-1}$, similar to Eq. (4).

Since G is autostable by hypothesis, and $\Phi\colon G \to \widetilde{G}$ is bicomputable, $\widetilde{G}$ is autostable. Since β is an isomorphism, we have $\widetilde{G} \cong \widetilde{H}$. Hence there is a bicomputable isomorphism $\widetilde{G} \to \widetilde{H}$. Inspecting the construction in the proof of the implication "$\Leftarrow$" of Theorem 5.1 shows that there is a computable isomorphism $\widetilde{\mathcal{W}} \to \widetilde{\mathcal{V}}$. An argument similar to the one in the proof of Lemma 11.8 shows that the maps $\alpha_{\mathcal{W}}$ and $\alpha_{\mathcal{V}}$ are computable. So there is a computable isomorphism $\mathcal{W} \to \mathcal{V}$, as required. □

Remark 12.5 (Computable duality between t.d.l.c. groups and meet groupoids)**.** [6, Section 4], to be published in a separate paper, axiomatizes the class **M** of countable meet groupoids $\mathcal{W}$ that are isomorphic to $\mathcal{W}(G)$ for some t.d.l.c. group G. Note that Definition 5.2 of $\mathcal{G}_{\mathrm{comp}}(\mathcal{W})$ makes sense for any meet groupoid $\mathcal{W}$ with domain $\mathbb{N}$. Besides some basic algebraic axioms on meet groupoids (such as saying that different left cosets of the same subgroup are disjoint), one needs an axiom ensuring local compactness of $\mathcal{G}_{\mathrm{comp}}(\mathcal{W})$: there is a subgroup K such that each subgroup $U \subseteq K$ has only finitely many left cosets contained in K. Furthermore, one needs to say that the map $\alpha_{\mathcal{W}}\colon \mathcal{W} \to \widetilde{\mathcal{W}}$ in Claim 12.3 is onto. In general, this could fail: consider the meet groupoid obtained from a computable copy of $\mathcal{W}(\mathbb{Z}_p)$ by deleting all cosets of subgroup of the form $p^{2i+1}\mathbb{Z}_p$: its double dual $\mathcal{W}$ is isomorphic to $\mathcal{W}(\mathbb{Z}_p)$. The required "completeness axiom" avoiding this situation, called CLC in [6, Section 4], states that if $N \in \mathcal{W}$ is normal (i.e., each left coset of N is also a right coset), and $\mathcal{S} \subseteq L(N)$ is finite and closed under inverse and product, then there is a subgroup $V \in \mathcal{W}$ such that $C \subseteq V \leftrightarrow C \in \mathcal{S}$. (These axioms

are sufficiently simple to imply that $\mathbf{M}$ is an arithmetical class in the sense of descriptive set theory.) Using the methods to prove Criterion 12.2 one can proceed to showing that the operators $\mathcal{W}_{\mathrm{comp}}$ and $\mathcal{G}_{\mathrm{comp}}$, restricted to the Haar computable meet groupoids in $\mathbf{M}$, are inverses of each other. That is, composing one with the other leads to a computable copy of the original structure that is computably isomorphic to it.

Theorem 12.6. *The computably t.d.l.c. groups $\mathbb{Q}_p$ and $\mathbb{Z} \ltimes \mathbb{Q}_p$ are autostable.*

Proof. In Example 4.8 we obtained a Haar computable copy $\mathcal{W}$ of the meet groupoid $\mathcal{W}(\mathbb{Q}_p)$. Recall that the elements of $\mathcal{W}$ are given as cosets $D_{r,a} = \pi_r^{-1}(a)$ where $r \in \mathbb{Z}$, $\pi_r \colon \mathbb{Z}_p \to C_{p^\infty}$ is the canonical projection with kernel $p^{-r}\mathbb{Z}_p$, and $a \in C_{p^\infty}$.

By the criterion above, it suffices to show that any Haar computable copy $\widetilde{\mathcal{W}}$ of $\mathcal{W}(\mathbb{Q}_p)$ is computably isomorphic to $\mathcal{W}$. By hypothesis on $\widetilde{\mathcal{W}}$ there is an isomorphism $\Gamma \colon \mathcal{W} \to \widetilde{\mathcal{W}}$. Let $\widetilde{U}_r = \Gamma(U_r)$ for $r \in \mathbb{Z}$. We will construct a *computable* isomorphism $\Delta \colon \mathcal{W} \to \widetilde{\mathcal{W}}$ which agrees with Γ on the set $\{U_r \colon r \in \mathbb{Z}\}$. First we show that from r one can compute the subgroup $\widetilde{U}_r \in \widetilde{\mathcal{W}}$.

(a) If $\widetilde{U}_r$ has been determined, $r \geq 0$, compute $\widetilde{U}_{r+1}$ by searching for the unique subgroup in $\widetilde{\mathcal{W}}$ that has index p in $\widetilde{U}_r$.
(b) If $\widetilde{U}_r$ has been determined, $r \leq 0$, compute $\widetilde{U}_{r-1}$ by searching for the unique subgroup in $\widetilde{\mathcal{W}}$ such that $\widetilde{U}_r$ has index p in it.

The shift homeomorphism $S \colon \mathbb{Q}_p \to \mathbb{Q}_p$ is defined by $S(x) = px$. Note that $B \to S(B)$ is an automorphism of the meet groupoid $\mathcal{W}$. Using the notation of Example 4.8 (recalled above), for each $\alpha \in \mathbb{Q}_p, r \in \mathbb{Z}$, one has $\pi_{r+1}(S(\alpha)) = \pi_r(\alpha)$, and hence for each $a \in C_{p^\infty}$,

$$S(D_{r,a}) = D_{r+1,a}. \tag{5}$$

We show that S is definable within $\mathcal{W}$ by an existential formula using subgroups U_r as parameters. Recall that given a meet groupoid $\mathcal{W}$, by $L(U)$ we denote the set of left cosets of a subgroup U. For $D \in L(U_r)$ we write D^k for $D \cdot \ldots \cdot D$ (with k factors), noting that this is defined, and in $L(U_r)$.

Claim 12.7. *Let $B \in L(U_r)$ and $C \in L(U_{r+1})$. Then*

$$C = S(B) \Leftrightarrow \exists D \in L(U_{r+1})\, [D \subseteq B \wedge D^p = C].$$

$\Leftarrow$: If $x \in C$ then $x = py$ for some $y \in B$, so $x \in S(B)$. So $C \subseteq S(B)$ and hence $C = S(B)$ given that $S(B) \in L(U_{r+1})$.
$\Rightarrow$: Let $x \in C$, so $x = S(y)$ for some $y \in B$. Let $y \in D$ where $D \in L(U_{r+1})$. Then $D \subseteq B$. Since $D^p \cap C \neq \emptyset$, these two (left) cosets of U_{r+1} coincide. This shows the claim.

We use this to show that the function $\widetilde{S} = \Gamma \circ S \circ \Gamma^{-1}$ defined on $\widetilde{\mathcal{W}}$ is computable. Since $\Gamma(U_r) = \widetilde{U}_r$, $(r \in \mathbb{Z})$, $\widetilde{S}$ satisfies the claim when replacing the U_r by the $\widetilde{U}_r$. Since the meet groupoid $\widetilde{\mathcal{W}}$ is computable, given $B \in \widetilde{\mathcal{W}}$, one can search $\widetilde{\mathcal{W}}$ for a witness $D \in L(\widetilde{U}_{r+1})$ as on the right hand side, and then output $C = \widetilde{S}(B)$. So the function $\widetilde{S}$ is computable.

We build the computable isomorphism $\Delta \colon \mathcal{W} \to \widetilde{\mathcal{W}}$ in four phases. The first three steps build a computable isomorphism $L(U_0) \to L(\widetilde{U}_0)$, where $L(\widetilde{U}_0) \subseteq \widetilde{\mathcal{W}}$ denotes the group of left cosets of $\widetilde{U}_0$. (This group is isomorphic to C_{p^∞}, so this amounts to defining a computable isomorphism between two computable copies of C_{p^∞}.) The last step extends this isomorphism to all of $\mathcal{W}$, using that $\widetilde{S}$ is an automorphism of $\widetilde{\mathcal{W}}$.

For $q \in \mathbb{Z}[1/p]$ we write $[q] = \mathbb{Z} + q \in C_{p^\infty}$. We define $\widetilde{D}_{r,[q]} = \Delta(D_{r,a})$ for $r \in \mathbb{Z}, q \in \mathbb{Z}[1/p]$

(a) Let $\widetilde{D}_{0,[p^{-1}]}$ be an element of order p in $L(\widetilde{U}_0)$.
(b) Recursively, for $m > 0$ let $\widetilde{D}_{0,[p^{-m}]}$ be an element of order p^m in $L(\widetilde{U}_0)$ such that $(\widetilde{D}_{0,[p^{-m}]})^p = \widetilde{D}_{0,[p^{-m+1}]}$.
(c) For $a = [kp^{-m}]$ where $0 \leq k < p^m$ and p does not divide k, let $\widetilde{D}_{0,a} = (\widetilde{D}_{0,[p^{-m}]})^k$.
(d) For $r \in \mathbb{Z} - \{0\}$ let $\widetilde{D}_{r,a} = \widetilde{S}^r(\widetilde{D}_{0,a})$.

One can easily verify that $\Delta\colon \mathcal{W} \to \widetilde{\mathcal{W}}$ is computable and preserves the meet groupoid operations. To verify that Δ is onto, let $B \in \widetilde{\mathcal{W}}$. We have $B \in L(\widetilde{U}_r)$ for some r. There is a least m such that $B = (\widetilde{D}_{r,[p^{-m}]})^k$ for some $k < p^m$. Then p does not divide k, so $B = \widetilde{D}_{r,[kp^{-m}]}$.

We next treat the case of $G = \mathbb{Z} \ltimes \mathbb{Q}_p$. Let $\mathcal{V}$ be the Haar computable copy of $\mathcal{W}(G)$ obtained in Example 4.8, and let $\widetilde{\mathcal{V}}$ be a further Haar computable copy of $\mathcal{W}(G)$. Using the notation of Example 4.8, let

$$E_{z,r,a} = g^z D_{r,a} \text{ for each } z, r \in \mathbb{Z}, a \in C_{p^\infty}.$$

We list some properties of these elements of $\mathcal{V}$ that will be needed shortly. Note that we can view $\mathcal{W}$ as embedded into $\mathcal{V}$ by identifying $\langle r, a\rangle$ with $\langle 0, r, a\rangle$. Also note that $E_{z,r,a}\colon U_{r-z} \to U_r$ (using the category notation discussed after Fact 4.3). Since $D_{r+1,a} \subseteq D_{r,pa}$, we have

$$E_{z,r+1,a} \subseteq E_{z,r,pa}. \tag{6}$$

Furthermore,

$$E_{z,r,0} = g^z U_r = U_{r+z} g^z = (g^{-z} U_{r-z})^{-1} = (E_{-z,r-z,0})^{-1}. \tag{7}$$

By hypothesis on $\widetilde{\mathcal{V}}$, there is a meet groupoid isomorphism $\overline{\Gamma}\colon \mathcal{V} \to \widetilde{\mathcal{V}}$. Since G has no compact open subgroups besides the ones present in $\mathcal{W}(\mathbb{Q}_p)$, the family $(\widetilde{U}_r)_{r\in\mathbb{Z}}$, where $\widetilde{U}_r = \overline{\Gamma}(U_r)$, is computable in $\widetilde{\mathcal{V}}$ by the same argument as before. The set of elements A of $\widetilde{\mathcal{V}}$ that are a left and a right coset of the same subgroup is computable by checking whether $A^{-1} \cdot A = A \cdot A^{-1}$. The operations of $\widetilde{\mathcal{V}}$ induce a Haar computable meet groupoid $\widetilde{\mathcal{W}}$ on this set. Clearly the restricted map $\Gamma = \overline{\Gamma} \mid \mathcal{W}$ is an isomorphism $\mathcal{W} \to \widetilde{\mathcal{W}}$. So by the case of $\mathbb{Q}_p$, there is a computable isomorphism $\Delta\colon \mathcal{W} \to \widetilde{\mathcal{W}}$.

We will extend Δ to a computable isomorphism $\overline{\Delta}\colon \mathcal{V} \to \widetilde{\mathcal{V}}$. The following summarizes the setting:

$$\begin{array}{ccc} \mathcal{V} & \xrightarrow{\overline{\Gamma}/\overline{\Delta}} & \widetilde{\mathcal{V}} \\ \subseteq\uparrow & & \uparrow\subseteq \\ \mathcal{W} & \xrightarrow[\Gamma/\Delta]{} & \widetilde{\mathcal{W}} \end{array}$$

In five phases we define a computable family $\widetilde{E}_{z,r,a}$ $(z, r \in \mathbb{Z}, a \in C_{p^\infty})$, and then let $\overline{\Delta}(E_{z,r,a}) = \widetilde{E}_{z,r,a}$. As before write $\widetilde{D}_{r,a} = \Delta(D_{r,a})$.

(a) Let $\widetilde{E}_{0,r,a} = \widetilde{D}_{r,a}$. Choose $F_0 := \widetilde{E}_{-1,0,0}\colon \widetilde{U}_1 \to \widetilde{U}_0$
(b) compute $F_r := \widetilde{E}_{-1,r,0}\colon U_{r+1} \to U_r$ by recursion on $|r|$, where $r \in \mathbb{Z}$, in such a way that $\widetilde{F}_{r+1} \subseteq \widetilde{F}_r$ for each $r \in \mathbb{Z}$; this is possible by (6) and since $\mathcal{V} \cong \widetilde{\mathcal{V}}$ via $\overline{\Gamma}$.
(c) For $z < -1$, compute $\widetilde{E}_{z,r,0}\colon U_{r-z} \to U_r$ as follows:

$$\widetilde{E}_{z,r,0} = F_{r-z-1} \cdot F_{r-z-2} \cdot \ldots \cdot F_r.$$

(d) For $z > 0$ let $E_{z,r,0} = (\widetilde{E}_{-z,r-z,0})^{-1}$; this is enforced by (7).

(e) Let $\widetilde{E}_{z,r,a} = \widetilde{E}_{z,r,0} \cdot \widetilde{D}_{r,a}$.

One verifies that $\overline{\Delta}$ preserves the meet groupoid operations (we omit the formal detail). To show that $\overline{\Delta}$ is onto, suppose that $\widetilde{E} \in \mathcal{V}$ is given. Then $\widetilde{E} = \Gamma(E_{z,r,a})$ for some z, r, a. By (6) we may assume that $z < 0$. Then $E_{z,r,0} = \prod_{i=1}^{-z} E_{-1,r-z-i,0}$ as above. So, writing F_s for $\widetilde{E}_{-1,s,0}$, we have $\Gamma(E_{z,r,0}) = \prod_{i=1}^{-z} F_{r-z-i}\widetilde{D}_{r-z-i,a_i}$ for some $a_i \in C_{p^\infty}$.

Note that $\widetilde{S}(D) = F \cdot D \cdot F^{-1}$ for each $D \in L(\widetilde{U}_r) \cap \widetilde{\mathcal{W}}$ and $F \colon \widetilde{U}_{r+1} \to \widetilde{U}_r$. For, the analogous statement clearly holds in $\mathcal{V}$; then one uses that $\widetilde{S} = \Gamma \circ S \circ \Gamma^{-1}$, and that $\overline{\Gamma} \colon \mathcal{V} \to \widetilde{\mathcal{V}}$ is an isomorphism. Since $\widetilde{D}_{r+1,a} = \widetilde{S}(\widetilde{D}_{r,a})$, we may conclude that $\widetilde{D}_{r+1,a} \cdot F = F \cdot D_{r,a}$ for each such F. We can use these "quasi-commutation relations" to simplify the expression $\prod_{i=1}^{-z} F_{r-z-i}\widetilde{D}_{r-z-i,a_i}$ to $\widetilde{E}_{z,r,0}\widetilde{D}_{r,b}$ for some $b \in C_{p^\infty}$. Hence $\widetilde{E} = \widetilde{E}_{z,r,0}\widetilde{D}_{r,b}\widetilde{D}_{r,a}$. This shows that $\widetilde{E}$ is in the range of $\overline{\Delta}$, as required. □

References

[1] C. Ash and J. Knight. *Computable structures and the hyperarithmetical hierarchy*, volume 144 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 2000.

[2] M. Burger and S. Mozes. Groups acting on trees: from local to global structure. *Publications Mathématiques de l'IHÉS*, 92:113–150, 2000.

[3] J. Clanin, T. McNicholl, and D. Stull. Analytic computable structure theory and L^p spaces. *Fund. Math.*, 244(3):255–285, 2019.

[4] A. Day and J. S. Miller. Randomness for non-computable measures. *Trans. Amer. Math. Soc.*, 365(7):3575–3591, 2013.

[5] R. Downey and A. Melnikov. Effectively compact spaces. Preprint.

[6] A. Nies (editor). Logic Blog 2020. Available at https://arxiv.org/pdf/2101.09508.pdf, 2020.

[7] A. Nies (editor). Logic Blog 2021. Available at https://arxiv.org/pdf/2202.13643.pdf, 2020.

[8] Y. Ershov and S. Goncharov. *Constructive models*. Siberian School of Algebra and Logic. Consultants Bureau, New York, 2000.

[9] A. Figà-Talamanca and C. Nebbia. *Harmonic Analysis and Representation Theory for Groups Acting on Homogenous Trees*, volume 162. Cambridge University Press, 1991.

[10] H. Glöckner. Scale functions on p-adic lie groups. *manuscripta mathematica*, 97(2):205–215, 1998.

[11] S. Goncharov. Groups with a finite number of constructivizations. *Dokl. Akad. Nauk SSSR*, 256(2):269–272, 1981.

[12] N. Greenberg, A. Melnikov, A. Nies, and D. Turetsky. Effectively closed subgroups of the infinite symmetric group. *Proceedings of the American Mathematical Society*, 146(12):5421–5435, 2018.

[13] E. Hewitt and K. Ross. *Abstract Harmonic Analysis: Volume I Structure of Topological Groups, Integration Theory, Group Representations*, volume 115. Springer Science & Business Media, 2012.

[14] Z. Iljazović. Isometries and computability structures. *J.UCS*, 16(18):2569–2596, 2010.

[15] Z. Iljazović and T. Kihara. Computability of subsets of metric spaces. In *Handbook of Computability and Complexity in Analysis*, pages 29–69. Springer, 2021.

[16] C. Kapoudjian. Simplicity of Neretin's group of spheromorphisms. In *Annales de l'institut Fourier*, volume 49, pages 1225–1240, 1999.

[17] A. S. Kechris. *Classical descriptive set theory*, volume 156. Springer-Verlag New York, 1995.

[18] A. S. Kechris, A. Nies, and K. Tent. The complexity of topological group isomorphism. *The Journal of Symbolic Logic*, 83(3):1190–1203, 2018.

[19] B. Krön and R. Möller. Analogues of Cayley graphs for topological groups. *Mathematische Zeitschrift*, 258(3):637–675, 2008.

[20] P. La Roche. Effective Galois theory. *The Journal of Symbolic Logic*, 46(2):385–392, 1981.

[21] M. Lawson. *Inverse semigroups: the theory of partial symmetries*. World Scientific, 1998.

[22] M. Lupini, A. Melnikov, and A. Nies. Computable topological abelian groups. *To appear in J. Algebra*, 2022. arXiv preprint arXiv:2105.12897.
[23] A. Mal′cev. Constructive algebras. I. *Uspehi Mat. Nauk*, 16(3 (99)):3–60, 1961.
[24] M. Malicki. Abelian pro-countable groups and orbit equivalence relations. *Fundamentae Mathematicae*, 233, 2014. arXiv preprint arXiv:1405.0693.
[25] T. H. McNicholl. Computable copies of ℓ^p. *Computability*, 6(4):391–408, 2017.
[26] A. Melnikov. Computable topological groups and Pontryagin duality. *Trans. Amer. Math. Soc.*, 370(12):8709–8737, 2018.
[27] A. Melnikov and A. Montalbán. Computable Polish group actions. *J. Symb. Log.*, 83(2):443–460, 2018.
[28] A. Melnikov and K. M. Ng. Computable torsion abelian groups. *Adv. Math.*, 325:864–907, 2018.
[29] A. G. Melnikov. Computably isometric spaces. *J. Symbolic Logic*, 78(4):1055–1085, 2013.
[30] R. Möller. Structure theory of totally disconnected locally compact groups via graphs and permutations. *Canadian Journal of Mathematics*, 54(4):795–827, 2002.
[31] A. Nies, P. Schlicht, and K. Tent. Coarse groups, and the isomorphism problem for oligomorphic groups. *Journal of Mathematical Logic*, page 2150029, 2021.
[32] André Nies. *Computability and randomness*, volume 51 of *Oxford Logic Guides*. Oxford University Press, Oxford, 2009.
[33] A. Pauly. On the topological aspects of the theory of represented spaces. *Computability*, 5(2):159–180, 2016.
[34] A. Pauly, D. Seon, and M. Ziegler. Computing Haar Measures. In Maribel Fernández and Anca Muscholl, editors, *28th EACSL Annual Conference on Computer Science Logic (CSL 2020)*, volume 152 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 34:1–34:17, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
[35] V. Platonov and A. Rapinchuk. *Algebraic groups and number theory*. Academic press, 1993.
[36] Marian B. Pour-El and J. Ian Richards. *Computability in analysis and physics*. Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1989.
[37] M. Rabin. Computable algebra, general theory and theory of computable fields. *Trans. Amer. Math. Soc.*, 95:341–360, 1960.
[38] D. Robinson. *A course in the theory of groups*. Springer–Verlag, 1988.
[39] M. Schröder. Admissibly represented spaces and qcb-spaces. In *Handbook of Computability and Complexity in Analysis*, pages 305–346. Springer, 2021.
[40] J.P. Serre. *Trees*. Springer–Verlag, 1980.
[41] R. Smith. Effective aspects of profinite groups. *The Journal of Symbolic Logic*, 46(04):851–863, 1981.
[42] R. I. Soare. *Recursively Enumerable Sets and Degrees*. Perspectives in Mathematical Logic, Omega Series. Springer–Verlag, Heidelberg, 1987.
[43] J. Tits. Sur le groupe des automorphismes d?un arbre. In *Essays on topology and related topics*, pages 188–211. Springer, 1970.
[44] D. Van Dantzig. Zur topologischen Algebra. iii. Brouwersche und Cantorsche Gruppen. *Compositio Mathematica*, 3:408–426, 1936.
[45] K. Weihrauch. *Computable analysis*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 2000. An introduction.
[46] P. Wesolek. Elementary totally disconnected locally compact groups. *Proceedings of the London Mathematical Society*, 110(6):1387–1434, 2015.
[47] P. Wesolek. An introduction to totally disconnected locally compact groups. *Preprint of a book, people. math. binghamton. edu/ wesolek/ mathdocs/ TDLC_ Groups. pdf*, 2018.
[48] G. Willis. The structure of totally disconnected, locally compact groups. *Mathematische Annalen*, 300(1):341–363, 1994.
[49] G. Willis. Further properties of the scale function on a totally disconnected group. *Journal of Algebra*, 237(1):142–164, 2001.
[50] G. Willis. Computing the scale of an endomorphism of a totally disconnected locally compact group. *Axioms*, 6(4):27, 2017.

School of Mathematics and Statistics, Victoria University of Wellington, New Zealand
Email address: alexander.g.melnikov@gmail.com

School of Computer Science, The University of Auckland, New Zealand
Email address: andre@cs.auckland.ac.nz