Computable Linear Orders

UW Graduate Logic Seminar (Math 975), Spring 2023

Hongyu Zhu

University of Wisconsin-Madison

April 17, 2023

Table of Contents

Definitions

Definition (Computable Linear Order)

A linear order $A = (\omega, \leq_A)$ with $\leq_A$ computable.
Equivalently, an (infinite) suborder of $(\mathbb{Q}, \leq_{\mathbb{Q}})$ with computable domain.

Definition (Finitely Apart, Block)

For a linear order $(X, \leq)$, say $a, b \in X$ are finitely apart, $a \sim b$, if there are only finitely many elements between a and b.

A block is a $\sim$-equivalence class.

Definition

ζ, η are the order types of $\mathbb{Z}, \mathbb{Q}$, respectively.

Suborders of Linear Orders

Theorem

Every infinite linear order has a suborder of order type ω or ω^*.

Suborders of Linear Orders

Theorem

Every infinite linear order has a suborder of order type ω or ω^*.

Proof. Just do it! And clearly both ω and ω^* are necessary.

Suborders of Linear Orders

Theorem

Every infinite linear order has a suborder of order type ω or ω^*.

Proof. Just do it! And clearly both ω and ω^* are necessary.
Analogue in the computable world:

Theorem (Rosenstein)

Every computable linear order A has a computable suborder of one of the following order types: ω, ω^*, $\omega + \omega^*$, $\omega + \zeta\eta + \omega^*$.

Suborders of Linear Orders

Theorem

Every infinite linear order has a suborder of order type ω or ω^*.

Proof. Just do it! And clearly both ω and ω^* are necessary.
Analogue in the computable world:

Theorem (Rosenstein)

Every computable linear order A has a computable suborder of one of the following order types: ω, ω^*, $\omega + \omega^*$, $\omega + \zeta\eta + \omega^*$.

Proof. (Sketch) Suppose A has no computable ω or ω^* suborder. Then there are a first element and a last element, and every other element has a successor and a predecessor. Hence $A = \omega + \zeta\alpha + \omega^*$ for some α ($A/\sim = 1 + \alpha + 1$). If α is not dense, A has a computable $\omega + \omega^*$ subinterval. Otherwise, A must be $\omega + \zeta\eta + \omega^*$.

$\omega + \omega^*$ Is Necessary

Theorem (Tennenbaum, Denisov)

There is a computable linear order A of order type $\omega + \omega^*$ with no computable suborder of order type ω or ω^* (certainly not $\omega + \zeta\eta + \omega^*$).

$\omega + \omega^*$ Is Necessary

Theorem (Tennenbaum, Denisov)

There is a computable linear order A of order type $\omega + \omega^*$ with no computable suborder of order type ω or ω^* (certainly not $\omega + \zeta\eta + \omega^*$).

Idea: Every element needs to be pushed to one side or the other. Push to the wrong side when permitted.

Proof. Finite injury. Build A in initial and final segments. Start with a minimum element and a maximum.

Requirements:

- $R_e : W_e$ infinite $\to W_e$ has an element with only finitely many successors in A. (Similarly R_e^*)
- N_a: a either has only finitely many predecessors, or has only finitely many successors in A.

Priority: arrange the requirements in order type ω. Each requirement restrains a finite initial segment and a final one.

Strategy for R_e (R_e^* is dual; N_a can use either ideas):

1. Wait until some unrestrained (by higher priority strategies) x enters W_e.
2. Let x be the predecessor of the current last element. Restrain this modified final segment.
3. Restart strategy if injured by higher priority strategy.

Verification: easy.

$\omega + \zeta\eta + \omega^*$ Is Necessary

Theorem (Lerman)

There is a computable linear order A with no computable suborder of order type ω, ω^*, or $\omega + \omega^*$.

$\omega + \zeta\eta + \omega^*$ Is Necessary

Theorem (Lerman)

There is a computable linear order A with no computable suborder of order type ω, ω^*, or $\omega + \omega^*$.

Proof. Π_2 argument.

Requirement $R_e : W_e$ infinite $\to W_e$ has an element with infinitely many successors and infinitely many predecessors in W_e. (This clearly guarantees W_e does not have any of the specified order types.)

Start by fixing a global min 0 and a max 1.

$R_e : W_e$ infinite $\to W_e$ has an element with infinitely many successors and infinitely many predecessors in W_e.

Strategy for single R_e: maintains a *working interval* $I_e = (0, 1)$ (fixed), an *anchor* x_e (fixed once initialized), and a variable *current permitting interval* $P_e \in \{(0, 1), (0, x_e), (x_e, 1)\}$ starting as $(0, 1)$.

1. Wait until some $x \in (0, 1)$ enters W_e. Let $x_e = x, P_e = (0, x_e)$.
2. Restrain $I_e \backslash P_e$; keep building in P_e, until some $y \in P_e$ enters W_e.
3. *Flip* P_e (from $(0, x_e)$ to $(x_e, 1)$ or vice versa). Go back to Step 2.

$R_e : W_e$ infinite $\to W_e$ has an element with infinitely many successors and infinitely many predecessors in W_e.

Strategy for single R_e: maintains a *working interval* $I_e = (0, 1)$ (fixed), an *anchor* x_e (fixed once initialized), and a variable *current permitting interval* $P_e \in \{(0, 1), (0, x_e), (x_e, 1)\}$ starting as $(0, 1)$.

1. Wait until some $x \in (0, 1)$ enters W_e. Let $x_e = x, P_e = (0, x_e)$.
2. Restrain $I_e \backslash P_e$; keep building in P_e, until some $y \in P_e$ enters W_e.
3. *Flip* P_e (from $(0, x_e)$ to $(x_e, 1)$ or vice versa). Go back to Step 2.

There are 3 outcomes: If we wait forever at Step 1, W_e is finite. If we go back to Step 2 finitely many times, W_e is again finite. If we go back infinitely many times, the anchor has infinitely many predecessors and successors. Hence R_e is satisfied.

Tree of strategies: $T = 3^{<\omega}$. Each node σ works on requirement $R_{|\sigma|}$.

Data for R_e at σ: $I_\sigma = P_\tau = (a_\sigma, b_\sigma)$ (where τ is parent of σ), or $(0, 1)$ for $I_\varnothing$; $x_\sigma \in (a_\sigma, b_\sigma)$; $P_\sigma \in \{(a_\sigma, b_\sigma), (a_\sigma, x_\sigma), (x_\sigma, b_\sigma)\}$ starting as (a_σ, b_σ).

At stage s we visit levels $R_0, R_1, R_2, \ldots, R_s$, activating nodes along current path. Inactivated nodes keep their data.

Strategy at stage s for visiting level R_e: (say current node is σ)

1. If x_σ not initialized, check if some $x \in I_\sigma$ enters W_e. If so, let $x_\sigma = x, P_\sigma = (a_\sigma, x_\sigma)$.
2. If some $y \in P_\sigma$ enters W_e, flip P_σ.
3. Choose the next node to visit depending on the current value of P_σ.

At the end of stage s, put $s+2$ into the smallest working interval seen in stage s.

Verification: Look at the "true path." (Key: Activating infinitely many times is not very different from being always active.)

Π_n Linear Orders

We might want to say a Σ_n/Π_n linear order is some $(\omega, \leq_A)$ with $\leq_A$ having that complexity, but then Σ_1, Π_1 collapse to computable for trivial reasons.

Π_n Linear Orders

We might want to say a Σ_n/Π_n linear order is some $(\omega, \leq_A)$ with $\leq_A$ having that complexity, but then Σ_1, Π_1 collapse to computable for trivial reasons. We can keep generalizing this definition to consider interpretations of a linear order in ω (so $\leq_A$ is only a linear pre-order), giving rise to a "Σ_n-presented" linear order. Or we can generalize the second definition and work with suborders of $\mathbb{Q}$. We use the latter convention here.

Definition ($\Sigma_n/\Pi_n/\Delta_n$ Linear Order)

A suborder of $(\mathbb{Q}, \leq_{\mathbb{Q}})$ whose domain has the corresponding complexity.

Π_n Linear Orders

We might want to say a Σ_n/Π_n linear order is some $(\omega, \leq_A)$ with $\leq_A$ having that complexity, but then Σ_1, Π_1 collapse to computable for trivial reasons. We can keep generalizing this definition to consider interpretations of a linear order in ω (so $\leq_A$ is only a linear pre-order), giving rise to a "Σ_n-presented" linear order. Or we can generalize the second definition and work with suborders of $\mathbb{Q}$. We use the latter convention here.

Definition ($\Sigma_n/\Pi_n/\Delta_n$ Linear Order)

A suborder of $(\mathbb{Q}, \leq_{\mathbb{Q}})$ whose domain has the corresponding complexity.

It turns out that there is no loss of generality.

Theorem (Feiner)

Every Σ_n-presented linear order is isomorphic to a Π_n linear order.

Theorem (Feiner)

Any c.e. suborder of $\mathbb{Q}$ (i.e. the domain is c.e.) is isomorphic to a computable linear order.

Proof. (Rosenstein) (sketch) Use $\eta \cong \eta \cdot \eta$ and Rosser's trick.

This relativizes, so we have the following hierarchy:

$$\Delta_1 \subsetneq \Pi_1 \subsetneq \Pi_2 \subsetneq \Pi_3 \subsetneq \dots$$

Watnick's Generalization of Tennenbaum-Denisov

Theorem (Watnick)

If $S \subseteq \mathbb{Q}$ is a Π_2 linear order with order type τ, then there is a computable linear order with order type $\omega + \zeta\tau + \omega^*$ with no computable ω, ω^* suborder.

Watnick's Generalization of Tennenbaum-Denisov

Theorem (Watnick)

If $S \subseteq \mathbb{Q}$ is a Π_2 linear order with order type τ, then there is a computable linear order with order type $\omega + \zeta\tau + \omega^*$ with no computable ω, ω^* suborder.

Proof. We do the construction in blocks: we construct $1 + \tau + 1$ blocks, where the first block only grows to the right, the last only to the left, and all others in both directions.

When selecting members of A from $\mathbb{Q}$ we also decide the boundary of each block. If we decide to merge two blocks, we just stop adding elements in between two representatives; and start adding elements in between when we need to separate them.

If we merge the blocks of $x < y$, we say that y is *shifted* to the left (or x is shifted to the right, depending on which one we consider "moving").

Construction When A Is Computable

At stage s, we will build the first s blocks of S, along with the terminal blocks. Requirements:

- Each block has the correct order type (ζ for non-terminal blocks).
- $R_e : W_e \cap A$ infinite $\to W_e \cap A$ has both first and last element.

The first requirement can be satisfied easily by growing each block in allowed directions to have length $\geq s$ at (the end of) stage s.

Strategy for R_e at stage $s > e$: If we see new element enter $W_e \cap A$ and $|W_e \cap A| \geq 2$, then shift its min to the left and max to the right as much as possible; i.e. do not shift any element shifted by some R_i, or the current max/min of some $W_i \cap A$, whenever $i < e$.

This is more or less a finite injury strategy.

Π_2 Strategy

Priority tree: $T = 2^{<\omega}$, $0 < 1$. Each node σ guesses $i \in S \iff \sigma(i) = 0$. Using Tot being Π^0_2-complete, we can guess in a way such that $i \in S \iff$ we guess $i \in S$ infinitely many times. Hence, true path $f = \liminf_s f_s$. At stage $s+1$, visit levels $0, 1, \ldots, s$ and arrange the visit according to our current guesses.

When we change our guesses: If the guess changed from 0 to 1, merge the block with an adjacent block (i.e. shift min to the right/max to the left). If the guess changed from 1 to 0, we separate it from the previously merged block.

Notice: (1) New elements always grow at the end of each block, so merged blocks can always be recovered. (2) A block is separated in the final A iff we decide to separate it in infinitely many stages. (3) A block is merged into another block iff we separated it at only finitely many stages.

Bibliography

Downey, R. (1998). Computability theory and linear orders. In Yury Leonidovich Ershov (ed.), *Handbook of Recursive Mathematics*. Elsevier. pp. 138–823.

Watnick, R. (1984). A Generalization of Tennenbaum's Theorem on Effectively Finite Recursive Linear Orderings. *The Journal of Symbolic Logic*, 49(2), 563-569.

Thank you for listening!