Notes on a Paper by Frolov

Duarte Maia

October 30, 2024

1 Introduction

I have been reading a paper by Andrey Frolov [1] which has been proving challenging to read. I thought it would be beneficial to write myself an account of a proof I was interested in understanding. I also make some extra observations at the end.

2 Notation

I follow Frolov's notation, for the most part. Here are the relevant pieces of notation. Some divergence may occur; I will attempt to mark these points of divergence with footnotes. Assume L is a linearly ordered set, ordered by $<$.

- The successor relation $S(x, y)$ is defined as meaning that y is the immediate successor of x,
- We define interval notation $[x, y]$ as the set $\{z \in L \mid x \leq z \leq y\}$,[1]
- We define the *block relation* $F(x, y)$ as meaning that there are finitely many elements between x and y,
- The block of $x \in L$, denoted $[x]$, is the equivalence class of x under F,
- Given $k \in \mathbb{N}$, we say that L is *k-quasidiscrete* if every block $[x]$ is either infinite or has size at most k.

3 Computable Representations

Since I personally am not interested in isomorphism complexity at the moment, I state and prove Frolov's results in a simplified manner.

Theorem 1. Suppose that $f : L \to R$ is an order embedding satisfying

(1) If $S(x, x')$ then $F(f(x), f(x'))$, and

(2) If $y \in R \setminus \operatorname{im} f$ then $[y]$ is infinite and there exist x, x' L-successors so that $y \in [f(x), f(x')]$.

Then, L and R are isomorphic orders.

Proof: First, we investigate how f acts on blocks of L. Indeed, by condition (1) any two x and x' in the same L-block will be taken to the same R-block. On the other hand, the fact that f is an order embedding implies that any two elements in distinct blocks will be taken to distinct blocks.

Then, we note that each block (in either side) is either finite, ω, ω^*, or $\omega^* + \omega$. Condition (2) is sufficient to ensure that each block is taken to a block of the same type, and moreover that a finite block is taken to a finite block of the same size. ■

[1] Frolov defines it as the set of all points nonstrictly *between* x and y, which differs from our definition if $x > y$.

Theorem 2. Let L be a k-quasidiscrete linear order such that both L and its successor relation are $0'$-computable. Then, L is isomorphic to a computable order R.

Proof: This is done by constructing the (computable) linear order R, together with an ($0'$-computable) embedding $f\colon L \to R$, satisfying the conditions of Theorem 1.

Since $(L, <, S)$ is $0'$-computable, we may assume that we are given a computable sequence $(L_s, <_s, S_s)$ whose limit is our original order. We can easily modify this sequence to assume, without loss of generality, that at each step one of the following operations is performed:

- An element is added to L, i.e. $L_{s+1} = L_s \cup \{q(s)\}$, with its relation to other elements being immediately set, or
- An element of L is removed, with $L_{s+1} = L_s \setminus \{q(s)\}$, with everyone else's relation to each other remaining the same.

We assume also that no natural number is "recycled". In other words, once an element is removed, it is never added back. We moreover demand that at each step $(L_s, <_s, S_s)$ is "plausible", in the sense that $<_s$ is a linear order, and that $S_s(x, y)$ implies $x <_s y$ and that there is no one else $<_s$-inbetween them. Also, the result is obvious if L is finite, so we assume that L is infinite.

Under these assumptions, we may see L_s structurally as disjoint and ordered union of "successor chains". That is, we decompose $L_s = T_0 + \ldots + T_n$ with each $T_j = \{a_1^j S a_1^j S \ldots S a_{\lambda(j)}^j\}$ and no elements of T_j and T_{j+1} being successors. Moreover, at each step we are performing one of the following operations:

- Removal of an element (possibly splitting a chain into two),
- Addition of an isolated element,
- Addition of an element that connects to the start or end of a chain,
- Addition of an element that connects two chains.

We add four more assumptions to our construction of L:

1. If an element is ever removed, we also remove all elements that have been added after it. This can be acheived by, if an element is ever removed, we subsequently remove and re-add all elements that were added after it. (It requires minor verification to see that every element of L is removed and added a finite number of times.)
2. If an element ν is added that connects two chains T_j and T_{j+1}, we must have $\lambda(j) + 1 + \lambda(j+1) > k$. This is because, if an element was ever about to be added that would connect two chains with total length k or less, we instead remove all elements of the youngest chain, and subsequently add ν and the elements of this chain one by one, so that at each step we are only adding an element that connects to the start or end of a chain.

   It requires verification that every element of L is removed and added a finite number of times in this manner. As such, let x be an element of L. If x is part of a large ($> k$) block, eventually x will be (forevermore) part of a large enough chain that it will never be removed by this provision. On the other hand, suppose that x is part of a small ($\leq k$) block. Enumerate L until all elements of this block have been enumerated. Moreover, for any other element y of L enumerated by this stage, there are infinitely many – and so, at least $k + 1$ – elements between this chain and y. Wait until at least $k + 1$ elements have been permanently added by the pre-modification enumeration between the chain containing x and every other element y that was added to L before x was. Now, if the chain containing x is ever connected to an older chain, it will have had to pass through at least $k + 1$ elements on the way, and so the chain containing x will never be erased.

3. We assume that L has a maximal element $+\infty$ and a minimal element $-\infty$, that these are the first two elements enumerated into L, and that no one is ever added above $+\infty$ or below $-\infty$. Our construction only works for linear orders with such a maximal and minimal element. Now, given any linear order L, we can ask whether it has a maximal element and add one if not, and ask whether it has a minimal element and add one if not, and in either case change our enumeration of L as to satisfy our assumption. It should be noted, however, that this is not a universal process, and indeed Proposition 5 below shows that it could not be so.

4. We also never allow $-\infty$ or $+\infty$ to be removed. In light of assumption 2, this means that we need to ensure that $-\infty$ and $+\infty$ are never connected by less than k successive elements. A way to ensure this is by, if about to add an element that connects $-\infty$ and $+\infty$, not actually adding it. This is inconsequential: we're assuming that L is infinite, and so in light of assumption 1 the element that we just refused to add was going to be removed later in any case.

We now provide an algorithm that reacts to each of the above moves to create a computable linear order by stages, $R = \cup_s R_s$, and a computable sequence of maps $f_s\colon L_s \to R_s$. We will show that the sequence f_s stabilizes pointwise and satisfies the conditions of Theorem 1.

We start by handling the cases where an element has been added. Suppose $L_{s+1} = L_s \cup \{q\}$.

- Suppose that q is not related to anyone by S_s, i.e. it is an isolated element (for now). By assumptions 3 and 4 however, it definitely has immediate successors and predecessors in L_s (meaning not by the S_s relation, but instead by the successor relation induced by $<_s$). Let $a <_s q <_s b$ be this predecessor and successor. We seek to define $f(q)$. By virtue of f being (intended to be) an order-preserving embedding, there is some restriction on what $f_{s+1}(q)$ may be defined as; namely, it has to be in the open interval $Q =]f(a), f(b)[$. If Q is nonempty, we define $f(q)$ as the element of Q that has been added to R the earliest. Otherwise, add a new element r to R satisfying $f(a) < r < f(b)$ and set $f(q) = r$.

- Suppose now that q connects to the start of a chain T_j (though not to the end of another). This paragraph may be suitably modified if q connects to the end of a chain but not the start of another.

  Recall that $\lambda(j)$ is the size of the chain T_j. The construction separates into cases, depending on whether $\lambda(j) + 1$ is less than or equal to k, or greater than k.

  – If $\lambda(j) + 1 \leq k$: Add a new element to R just before $f(a_1^j)$, and set $f(q)$ to be this element;

  – If $\lambda(j) + 1 > k$: Let Q be the set of elements of R_s that are less than $f(a_1^j)$, but greater than $f(x)$ for all $x <_s a_1^j$. If this set is nonempty, set $f(q)$ to be the element of Q that was first added to R. Otherwise, add a new element to R as to make Q nonempty, and set $f(q)$ to be this element.

- Finally, suppose that q connects two chains T_j and T_{j+1}. By assumption 2, we must have $\lambda(j) + 1 + \lambda(j+1) > k$. Now, let Q be the set of elements of R_s that are inbetween $f(a^j_{\lambda(j)})$ and $f(a_1^{j+1})$. If Q is nonempty, set $f(q)$ to be the earliest added element of Q. Otherwise, add a new element to R as to make Q nonempty, and set $f(q)$ to be this new element.

Finally, if an element has been removed from L, we simply remove it and leave everything else the way it is. In particular, this is a source of – and indeed the only source of – elements of R that are not given by $f(x)$ for some $x \in L$.

The above algorithm produces a function $f\colon L \to R$. Indeed, any element of L corresponds to an element x that was added and never removed, and none of the rules above ever change the value of $f(x)$. Moreover, f is clearly an order embedding $L \hookrightarrow R$, so it remains to verify the two conditions of Theorem 1.

To verify condition (1), suppose that $xSx' \in L$. Run the simulation until both x and x' have been added to L. By the way we set up the presentation of L, we know that no element will ever be added into L between x and x'. Inspecting the construction of R, we see that whenever a new element is added between $f(x)$ and $f(x')$ it is because an element was added to L between x and x'; thus, the number of elements of

R between $f(x)$ and $f(x')$ will never increase from its current (finite value). In other words, we've shown that $S(x,x')$ implies $F(f(x), f(x'))$.

Finally, we verify condition (2). Let $y \in R \setminus \operatorname{im} f$. This means that y was at some point added into R as $f(x)$, but meanwhile x was erased from L. Now, run the simulation until this point, then continue running it until every element that is now present, that will ever be removed, has already been removed. Note that at that stage, every remaining element of L is now "immortal". By this point, there must be some x and x' that have been enumerated into L, so that there is no one between them (at this moment), and $f(x) < y < f(x')$.

Remark 3. If, by luck, it happens that $S(x,x')$, we see that condition (2) is verified. Indeed, y is between the images of two successor elements of L, and by inspection of the construction we see the following fact: If $S(x,x')$ but $f(x)$ and $f(x')$ are not successors, it must be the case that x and x' are part of an L-block that is of size larger than k, and by k-quasidiscreteness must be infinite. As such, by condition (1), the R-block that they – and consequently y – are contained in must also be infinite.

The argument in Remark 3 extends to the scenario where x and x' are in the same L-block. Thus, let us assume that they are in distinct blocks, and so that infinitely many (immortal) elements will be added to L between x and x'.

Suppose, for the sake of argument, that y is the oldest element of R between $f(x)$ and $f(x')$. Wait until the next immortal element, say z, is added to L between x and x'. By inspection of the construction we see that, in almost every case, $f(z)$ is defined as y, and since z is assumed to be immortal, this contradicts the assumption that y is not in the image of f. The only remaining cases are those where z is defined as a successor of x or as a predecessor of x', and even then it must be the case that the corresponding element is in a chain shorter than length k. As a consequence, these remaining cases happen a maximum of $2k$ times, and since infinitely many elements are added between x and x', we have a contradiction. Thus, y *cannot* be the oldest element between $f(x)$ and $f(x')$.

Suppose now instead that y is the *second* oldest element between $f(x)$ and $f(x')$, with y_0 being the oldest. Working still under the assumption that x and x' are in distinct blocks, we see by the previous paragraph that there shall eventually be an element z such that $f(z) = y_0$. Now, we perform a similar argument recursively: If (say) y is between $f(x)$ and $f(z)$ (with a similar argument holding if instead y is between $f(z)$ and $f(x')$), by the reasoning of the previous paragraph we see that x and z must be in the same block, and so condition (2) will hold.

Supposing now that y is the *third* oldest element between $f(x)$ and $f(x')$ (say again that y_0 is the oldest), with x and x' being in distinct blocks, we wait until an element z is added so that $f(z) = y_0$. Then, assuming that y is between $f(x)$ and $f(z)$ (with a similar argument holding if it is between $f(z)$ and $f(x)$), we are now in a condition to apply the argument of the previous paragraph, and so on. Inductively, we see that no matter how far from the oldest element y is from being, we will eventually find two immortal successor elements of L between which we find y, which by the argument outlined in Remark 3 is enough to guarantee condition (2) of Theorem 1. This completes the proof. ∎

Corollary 4. Any low k-quasidiscrete order is isomorphic to a computable order.

Proof: The successor relation for an order L is computable in its jump, which in this case is $0'$. ∎

Proposition 5. Theorem 2 is not effective, and indeed not even $0'$ effective. More precisely, any degree d that computes a function that turns an index for a $0'$-computable (infinite) k-quasidiscrete order with successor relation into an index for an isomorphic computable order must necessarily satisfy $d \oplus 0' \geq 0''$.

Proof: There is a uniform way to, given an oracle program P, build a $0'$-computable order (with successor relation) that is isomorphic to ω if $P^{0'}\uparrow$, and to ω^* if $P^{0'}\downarrow$. Roughly speaking: Until the program halts, add a new element on the right; once it does, add it to the left. Now, if Theorem 2 was d-uniform for some degree d, we could turn this into a *computable* order satisfying these properties, and $0'$ is able to confirm that this order has a minimal or a maximal element. Depending on the case, this can be used to deduce whether $P^{0'}$ halts, and so $d \oplus 0'$ computes $0''$. ∎

Proposition 6. Corollary 4 is effective, in the following sense: Given a low degree d, there is a computable function that turns an index for an infinite d-computable k-quasidiscrete order into an index for an isomorphic computable order.

Proof: Let L be a d-computable order. Then, $\bar{L} = L \cup \{-\infty, \infty\}$ is also d-computable, and so we can apply Theorem 2 to $\bar{L}$ effectively to obtain a computable isomorphic copy $\bar{R}$. Moreover, by inspection of the proof of Theorem 2, we see that the value of $f(-\infty)$ and $f(\infty)$ is determined at the start and never changed. Thus, we can simply remove these two elements from $\bar{R}$ to obtain the desired isomorphic computable order $R \cong L$. ■

References

[1] A. N. Frolov. Linear orderings of low degree. *Sibirsk. Mat. Zh.*, 51(5):1147–1162, 2010.