Foundations of Online Model Theory

Rod Downey
Victoria University
Wellington, New Zealand
(Joint with Melnikov and Ng)

May, 2019

Online

- Your job is to put objects $\{a_0, a_1, \dots\}$ into bins $\{b_j \mid j \in \omega\}$ of a certain maximum size and they are given one at a time. You must put a_i into some b_j at stage i before I give you a_{i+1} and try to minimize the number of bins used as you go along. '
- I give you a graph G one point at a time, giving the induced subgraph $\{a_0, a_1, \dots, a_i\}$ at step i and you must decide a colour before given a_{i+1}. Minimize the number of colours.
- You are in a vast graph and need to build, for example, an object incrementally, but don't have time to see more than a local neighbourhood.
- You are a triage nurse and patients arrive and you must order them in some priority ordering to be seen dynamically.
- You are a finitely branching tree T which I am giving you up to height n at step n and you must build a path of height n.

Online

- In all of the above you are in an online situation, though some are different than others.
- There are hundreds of algorithms for such problems both in the finite case, and in the case where e.g. a scheduler needs to works “forever”.
- There are books with taxonomies of such algorithms.
- The goal is to give a theoretical basis for the above.

Two approaches

- There seem two basic criteria needed for "online-ness".
- And there are two non-independent approaches based on them.
- The first is based on the punctuality of the online algorithm. We must do something immediately before the next item arrives. (Or leaves, this could be a Δ^0_2 process.) This has a growing and rich theory.
- The second is based on the uniformity of the operators against a hostile universe. This has almost no theory, yet!

Punctuality

- This especially enriches computable structure theory.
- Consider the proof that computable dense linear orderings without endpoints are computably categorical.
- Suppose that A and B are quick copies, could you build a quick isomorphism? No...
- No, but you could for the analogous problem is A and B are giant cliques.

Punctual structures

- The chosen model

Definition (Kalimullin, Melnikov, Ng)

A structure is punctual (fully primitive recursive) if it has domain $\mathbb{N}$ and all relations and functions are (uniformly) primitive recursive.

- You might say the use of primitive recursive seems quite arbitrary. What about automatic (Nerode, Khoussainov, Stephan, etc) or polynomial time (Cenzer, Remmel, Downey, others)....
- Primitive recursion is representative of the general total without delay set up, it is natural, and moreover no special coding or combinatorics/complexity classes etc are needed.
- The restricted Church-Turing thesis for primitive recursive functions says that a function is primitive recursive iff it can be described by an algorithm that uses only bounded loops.

Further logician's motivation

- A fundamental result in computable model theory is that a decidable theory has a decidable model.
- Almost any normal decidable theory is actually primitive recursively decidable.
- Almost any normal decision problem arising in algebra will be primitive recursive. For example, if a field has a splitting algorithm then that will naturally be primitive recursive.

Formalizing this

- Tempting: If T has a primitive recursive decision procedure then it has a punctually decidable model.
- But what do we mean by this? As observed in work by Cenzer and Remmel, for example, we need primitive recursive Skolem function

Theorem (Bazhenov,D, Kalimullin, M)

If T has a primitive recursive decision procedure then it has a punctually decidable model.

- Proof Henkin. Contrast with:

Theorem (KMN)

There is a punctually 1-decidable theory with no punctually 1-decidable model.

- Several known results lift.

Theorem (BDKM)

Suppose that T is a complete theory with primitive recursive decision procedure. TFAE

1. *T has a punctually decidable prime model*
2. *T has a prime model and an increasing uniformly primitive recursive sequence of principal types with quick witnesses.*

- D, Harrison-Trainor, Greenberg and Turetsky observed that a complete primitive recursive theory has a punctual model that omits a given primitive recursive non-principle type.
- Question Develop punctual pure model theory.

Natural Punctual Structures

Theorem (KMN)

The following structures have punctual presentations iff they have computable ones.

1. *Linear orderings (Gregorieff)*
2. *Boolean algebras*
3. *Equivalence structures*
4. *Torsion-free abelian groups.*
5. *Abelian p-groups*
6. *Locally finite graphs.*

Theorem (KMN)

The following have computable structures which have no punctual presentation.

1. *Torsion abelian groups.*
2. *(undirected) Graphs*
3. *Archimedean ordered abelian groups*

Theorem (Bazhenov, Harrison-Trainor, KMN)

$\{e \mid M_e$ *has a punctual (automatic, poly) presentation*$\}$ *is* Σ^1_1 *complete.*

Universality

- We usually regard graphs as universal structures. What about online?
- That is, we can punctually code any other punctual structure into one of the given type.

Theorem (D, Harrison-Trainor, K, M, Turetsky)

1. *The class of structures with only one binary function symbol is punctually universal.*
2. *Graphs are not universal for punctual structures.*

Theorem (K,M,Montalbán)

There is no punctually universal structure in a predicate language.

Categoricity

Definition

We say that A is punctually categorical iff for all punctual $B \cong A$, there are primitive recursive f and g $f : A \to B$ and $g : B \to A$. That is f and f^{-1} are punctual.

Theorem (KMN)

1. *An equivalence structure is punctually categorical iff it is either of the form $F \cup E$ where F is finite and each class in E has size 1, of it has finitely many classes at most one of which is infinite.*
2. *Linear orders are pc iff they are finite.*
3. *Same for boolean algebras.*
4. *Torsion free abelian groups are pc iff they are trivial.*
5. *Abelian p-group iff it has the form $F \oplus V$ where F is finite and $pV = \mathbf{0}$.*

Theorem (D, Harrison-Trainor, K, M, Turetsky)

1. *If G is an undirected graph. Then G is pc iff G becomes a clique or an anti-clique after the removal of finitely many points either adjacent to all, or disjoint from all vertices of G.*
2. *In fact any punctually categorical structure with at most binary relational symbols is automorphically trivial.*

A Punctual Monster

Theorem (KMN)

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

- Uses a special functional "pressing argument" (board).
- Greenberg, Downey, Melnikov, Turetsky and Ng has announced this can be iterated through the computable ordinals.

Reductions

- (MN) define $A \leq_{pr} B$ to mean that there is a primitive recursive $f : A \to_{\text{onto}} B$.

Theorem (MN)

If G is a punctual graph the pr-degree of G is 1 iff G is punctually categorical.

- (MN) have a number of results about the strange structure of the pr-degrees.
- There are lots of other results, but it is time to move to a new topic....
- A higher type topic....

The Operator Approach

- A criticism of the work above is that there is another aspect of online algorithms.
- Consider online colouring of a graph with the simple monotone model.
- The online algorithm A acts on G_{s+1} to (irrevocably) colour $v = s + 1$.
- For simplicity, we do not allow the algorithm to see $f(s+1)$ many new points, where f would be primitive recursive, before making its decision.
- The crucial insight is that A must act *uniformly* on any sequence $G_0, \ldots, G_{s+1}, \ldots$. The offline algorithm can be considered as a sequence of algorithms $\hat{A}_s$ acting on G_s for each s.

- The key observation: whilst there are only a primitive recursive number of graphs of size s, there is no reason that the graph the opponent builds is even remotely primitive recursive.
- There are $2^{\aleph_0}$ many possible graphs.
- We are thinking of the algorithm acting on objects represented as paths in a computable tree.

Definitions

Definition

A class $\mathcal{C}$ of relational structures is called inductive if $A \in \mathcal{C}$ implies A has a filtration $A = \cup_s A_s$ where each A_n is finite and has universe $\{1, \ldots, n\}$ and for all $n' > n$ the substructure induced by $\{1, \ldots, n\}$ in $A_{n'}$ is A_n. (Similarly g-filtration for a computable g with universe $\{1, \ldots, g(n)\}$, etc.) n ($g(n)$) is the height of A_n.

Definition

A *representation* of a class $\mathcal{C}$ of structures is a subjective function $F : \omega^{<\omega} \to \mathcal{C}^{<\omega}$, which acts computably in the sense that $F(\sigma) = C_n$ for $|\sigma| = n$ and $|C_n| = n$, and if $\sigma \preceq \tau$ then $F(\sigma)$ is an induced substructure of $F(\tau)$. (Later this might be partial, and objects might have several names.)

Online

Definition

A on-line problem is a triple (I, S, s) where I is the space inputs viewed as strings in a finite or infinite computable alphabet, S is the space of outputs (solutions) viewed as strings in (perhaps, some other) alphabet, and $s : I \to S^{<\omega}$ is a function which maps I to the set of admissible solutions of σ of S.

Intuitively, to solve a problem (I, S, s) we need to find an online computable function f which, on input i, chooses an admissible solution from the finite set $s(i)$.

Definition

A solution to an online problem (I, S, s) is a function $f : I \to S$ with the properties:

(O1) f is computable without delay (to be clarified);

(O2) $f(\sigma) \in s(\sigma)$ for every $\sigma \in I$;

(O3) $f(\sigma)$ uses only σ in its computation.

Remark: the difference

- The key difference from the punctual view of online-ness of viewed earlier, there is now no reason that the structures we are dealing with, even in the punctual case, need to be primitive recursive structures.
- Suppose that the representation is 2^ω. Then there will be uncountably many structures represented as paths through the tree. It is the algorithm acting on the paths which is uniform, and the primitive recursiveness would be relative to the path as we see later.

Finite=Infinite

Proposition

- *Suppose that A acts in an online fashion uniformly on all finite strings. Then A acts uniformly online on all computable paths through the representing space.*
- *Suppose that the algorithm A is total and acts uniformly online on all computable paths. Then A acts uniformly on all paths.*

Proof.

(i) If A fails on some computable path α it must fail on some finite initial segment. (ii) Computable paths are dense. $\square$

More precisely

- Suppose f is a solution to an online problem (I, S, s).
- The space of inputs carries a natural totally disconnected topology, and the completion of I forms the space of paths or infinite words in the language of I.
- The solution f induces a solution for the completion of the initial problem (I, S, s), in the sense that f can be uniquely extended to a functional $\overline{f} : [I] \to [S]$ between completions.
- Then $\overline{f}$ is a primitive recursive ibT operator (which means that its oracle use is bounded by the identity) with the property that, for every $n, f(p \upharpoonright n) \in s(p \upharpoonright n)$.
- In this case we say that $\overline{f}$ is a solution to the completion of (I, S, s).

Lifting things

- We can see what lefts from the previous punctual setting.
- It is possible to define online categoricity without too much difficulty.

Definition

A structure G is online categorical if there is an online computable f which, on input α and β and arbitrary representations of G outputs an isomorphism from α to β.

- Notice that we are using (representations of) partial maps in that they only need to work on copies of the structure.

Proposition

A relational structure is online categorical if, and only if, it is totally automorphically trivial.

- Here this means that for all $\overline{x}, y, z$, y and z are in to same orbit over $\overline{x}$.
- We don't know what happens if we add functions; and this is tricky to define anyway.

- I should prove something.
- If G not automorphically trivial, let $\overline{x}$ be shortest (length n) such that for some z is not in the same automorphism orbit as y over $\overline{x}$.
- To make α, β, copy $\overline{x}$ into both and calculate $f : \alpha \restriction n \to \beta \restriction n$.
- If we identify these with $\overline{x}$, then f induces a permutation o$\beta \restriction n$.
- As n is least, any permutation of $\overline{x}$ can be extended to an automorphism of the whole structure.
- Adjoin z to α and find y' playing the role of y over $f(\alpha \restriction n)$.
- Then necessarily $f(z) = f(y')$, because f has already shown its computation on the first n bits.
- However, by the choice of z and y', f cannot be extended to an isomorphism

- We comment that the theme the theory tends to be smoother (and surely involves definability and forcing) than the corresponding punctual theory.
- This is akin to computable structure theory vs uniform computable structure theory.
- Question: Also connections between uniform compuatble structure theory and (type-2) computable analysis?

Graphs; incremental computation

- There is a notion of incremental computation due to Milterson et. al. and we can show that this aligns to an online version of Weihrauch reduction.
- We can also have ratio preserving Weihrauch reductions.
- The performance ratio of a minimization problem (e.g. coloring here) is
$$\frac{|\{f_\chi(G \upharpoonright n)\}|}{|\{\chi(G \upharpoonright n\}|}.$$
- E.g. Famously First Fit Bin Packs with Performance ratio 2.
- One example as above is colouring. E.g. a graph of pathwidth k can be online coloured by $3k-2$ many colours reduces to chain covering of interval orderings.
- (Question?) If the paths correlate to path decomposition, online Courcelle Theorem
- Analog of Irani's Theorem?

Computable analysis

- So $f : 2^\omega \to 2^\omega$ is online computable if for all α, $f(\alpha \upharpoonright n) = f(\alpha) \upharpoonright n$.
- There are obvious extensions of this. For a fixed function g, f is g-online computable if $f(\alpha \upharpoonright g(n)) = f(\alpha) \upharpoonright n$. An obvious case is when $g(n) = n + k$, which would be online with delay k.
- E.g. $+$ on the reals (below) is online computable with constant 2.

Representations

- Suppose we want to look at computable relationships between a totally disconnected space like 2^ω, and, for example, $\mathbb{R}$. Topological considerations rule out "computable" injective functions from $\mathbb{R}$ to 2^ω, since we have see such functions must be continuous. In the finite case, no such topological considerations occur.
- So let X be our (topological) space, with $\mathbb{R}$ as a canonical model. As above we could define a representation of a space X as a partial function $\delta : \omega^\omega \to X$, so that elements $x \in X$ have δ-*names* p_x (strictly a set $\{p_x \mid \delta(p_x) = x\}$). Note that x can have many names p_x; consider the case of names being Cauchy sequences and the space being the reals.

Proposition

If f is online computable on $[a, b] \subseteq [0, 1]$ then $\int_a^b f(x)dx$ is online Lipschitz computable with constant 2.

- Represent a function $f : X \to Y$ is an exactly analogous manner to 2^ω, but taking into account non-uniqueness of representation. That is, $f(x) = y$ is represented (realized) by some $F : \omega^\omega \to \omega^\omega$ taking each p_x to some p_y. (The first is a δ_X-name and the second δ_Y, but will will suppress this explicitness in the pursuit of clarity.)
- Let f, g be as above. Then $f \leq_W g$, Weihrauch reducibility, is defined to mean that there are computable A and B defined now on ω^ω, such that for any p_x, and any representation (realizer) G of g,

  $$A(p_x, G(B(p_x))$$

  realizes f (i.e. is a name for $f(x)$). (Henceforth, we will suppress the coding when the context is clear; particularly in the case that we are dealing with a metric space.

- Lots other applications of this setting.
- E.g. EX-learning, Distributed computing, Büchi automata, etc.
- The idea is to somehow tye these together.
- Here is one example, from proof theory:

Δ_2^0 Processes

- Imagine you are in a situation where the data you are dealing with is so large that you cannot see it all. At each stage s your goal is to build a solution f to some problem.
- Imagine you are in a situation where the data you are dealing with is so large that you cannot see it all. At each stage s your goal is to build a solution f to some problem.

Definition

- A *limiting online algorithm* on 2^ω is a computable function A such that for each s, $A(\alpha \upharpoonright s)$ computes a string $\{f_A(n,s) \mid n \leq s\}$ such that $\lim_s f_A(n,s)$ exists for each n.
- As usual we would have $A(\alpha \upharpoonright g(s))$ for the g-online version.

Reductions

- We can then compare combinatorial problems by how fast their limits converge.
- We say that algorithm $A \leq_{O,lim} B$ if there is an online Weihrauch reduction reducing A to B such that $f_B(n,s) = f_B(n,t)$ for all $t \geq s$ implies $f_A(n,s) = f_A(n,t)$ for all $t \geq s$.
- This gives a fine grained measure of the complexity of combinatorial problems.
- Finitary Reverse Mathematics

Example

- A binary tree T of height n is called separating if for each $j \leq n-1$, for any node σ on T of height j, and $i \in \{0,1\}$, if $\sigma * i$ does not have an extension in T of height n, then for all τ of length j, neither does $\tau * i$.
- Let X_0 denote the space whose paths are separating trees, and X the paths of trees.

Proposition

There is a 2^{n+1}-limiting online reduction which finds limiting online paths in X from those in X_0.

- Also unexplored is the situation for online algorithms acting on Δ_2^0 inputs. E.g. modelling users in a network.

References

- Foundations of Online Structure Theory, BSL, in press.
- Foundations of Online Structure Theory, II, in preparation. (available on request)
- Lots of papers on Melnikov's and Ng's home pages.

Thank You