Uncountable computable model theory

Noam Greenberg

Victoria University of Wellington

30th June 2013

Mathematical structures

Model theory provides an abstract formalisation of the notion of a mathematical structure, or object.

Examples:

- Rings and fields (systems of numbers): integers, real and complex numbers, number fields (Gaussian integers), function fields, collections of matrices.
- Vector spaces ($\mathbb{R}^n$, $\mathbb{C}^n$), Hilbert and Banach spaces.
- Graphs (networks), orderings, trees
- Groups (symmetries)

Mathematical structures

Formally, a structure is simply a collection of elements, together with distinguished relations and operations.

In the examples above:

- Number systems are specified by the collection of numbers, and the operations of addition and multiplication.
- Graphs are specified by the collection of vertices, and the relation of being connected by an edge.
- A space is the collection of points, and the subsets which are the curves, surfaces, etc.

Information in structures

Information about a mathematical structure can be fed as input to a computer. This means we can ask:

- How much information is encoded in a structure?
- How complicated are various operations and constructions we perform on mathematical objects?

Example: bases of vector spaces

Task: find a basis for a given vector space V. Essentially, find a system of axes which tells you the cardinal directions in the space, and so find coordinates for every vector.

For finite dimensional spaces: only finitely much information is needed. Computationally trivial.

For an infinite dimensional space: the task may require consulting the halting problem for advice. We need to know, in finite time, the answer to a question which requires an infinite search.

Other examples

- Word problem for finitely presented groups.
- Can we colour a graph effectively?

Measuring complexity

Relative complexity is measured by Turing reducibility. $A \leqslant_{\mathrm{T}} B$ means that B contains at least as much information as A. Gives us Turing degrees.

Absolute complexity is given by identifying “milestones” in the Turing degrees. The least degree is that of the computable sets: no information.

The main non-computable example is the halting problem, and its iterations.

Extending computability

We can only feed countable objects to computers. Computers don’t have the time or space to comprehend larger structures.

A ‘diagonal argument’, similar to the one showing the halting problem is not computable, shows that the real line is uncountable.

$$
\begin{array}{r}
0.111111111\ldots \\
3.141592653\ldots \\
2.718281828\ldots \\
1.414213562\ldots \\
5.000000000\ldots \\
0.693147180\ldots \\
\vdots
\end{array}
$$

Higher computability

So some important objects are uncountable. We want to ask questions about their information content.

One possible solution: use idealised computers which can run for really long times and have really big storage capacity. Get “admissible computability”.

Test case: linear orderings

A linear ordering is a collection of points, ordered in a “complete hierarchy”.

Main examples:

- The real line, and subsets (natural numbers, integers, rational numbers)
- Ordinal numbers (the building blocks of admissible computability).

Countable linear orderings

- [Richter] Other than the computable ones, no countable linear order-type has a simplest presentation.
- [Dzgoev;Remmel] Characterisation of the computable linear orderings, all computable copies of whose one can “effectively see”. This is in terms of the successor relation.
- [Watnick;Denisov] Finding a subset of order-type the natural numbers cannot always be done effectively.

Uncountable behaviour

For uncountable linear orderings we get different results.

- Every piece of uncountable information can be coded into a linear order-type.
- The criterion for “computable categoricity” actually involves hidden effectiveness conditions.

We get a better understanding of the countable case by examining generalisations. Much of the new behaviour is due to the fundamental difference between finite and infinite.

Differentiation and integration

Hausdorff gave a ‘derivative’ operation for linear orderings: identify points which lie finitely far apart.
There are many possible reversals. A canonical one is replacing every point by a copy of the integers.

Watnick measured the complexity of these operations. They require precisely two iterations of the Turing jump.

Uncountable differentiation and integration

Watnick’s theorem fails badly for uncountable linear orderings. It turns out that there is a more fundamental derivative operation on linear orderings, given by Cantor and Bendixon: erase left-isolated points.

The corresponding version of Watnick’s theorem holds for both countable and uncountable linear orderings.

Thank you.