Roman V. Yampolskiy - From Seed AI to Technological Singularity via Recursively Self-Improving Software (2015)

Tom Rochette <tom.rochette@coreteks.org>

November 2, 2024 — 36c8eb68

0.1 Context

0.2 Learned in this study

0.3 Things to explore

- RSI Convergence Theory
- Is it possible that the "best" intelligence we can produce be limited to a human intelligence?

1 Overview

2 Notes

- Need a definition of improvement
  - Solving the same problem faster or with less need for computational resources
  - Improvement in error rates or finding closer approximations to optimal solutions
- The major improvement we want from self-improving intelligent software is higher degree of intelligence which can be approximated via machine friendly IQ tests with a significant G-factor correlation
- Three levels of improvement
  - Modification
  - Improvement (weak self-improvement)
  - Recursive improvement (strong self-improvement)
- In the field of complex dynamic systems, also known as chaos theory, positive feedback systems are well known to always end up in what is known as an attractor - a region within the system's state space that the system can't escape from (similar to the idea that trying to compress a compressed file multiple times will not lead to any compression at some point)
- Yudkowsky terminology
  - Cascades: When one development leads to another
  - Cycles: Repeatable cascade in which one optimization leads to another which in turn benefits the original optimization
  - Insight: New information which greatly increases one's optimization ability
  - Optimization slope: The goodness and number of opportunities in the space of solutions
  - Optimization resources: How much computational resources an agent has access to
  - Optimization efficiency: How efficiently the agent utilizes these computational resources
- The bootstrap fallacy: The notion of an unbounded self-improvement machine is a fallacy
- Two approaches to improvements discovery:
  - Brute force based approach using Levin (Universal) Search

  - The system has a certain level of scientific competence and uses it to engineer and test its own replacement
- A hybrid RSI system which includes both an artificially intelligent program and a human scientist
- We can also evaluate systems as to certain binary properties. FOr example: We may be interested in only systems which are guaranteed not to decrease in intelligence, even temporarily, during the improvement process
- We're interested in understanding the necessity of unchanging source code segments, that is, if there are certain parts of the source code of the system that needs to remain unchanged from generation to generation
- We're interested in understanding if the RSI process can take place in an isolated system or if interacting with the external environment is necessary
- Rice theorem: For any arbitrarily chosen program it is impossible to test if it has any non-trivial property such as being very intelligent
- Other difficulties related to testing remain even if we are not talking about arbitrarily chosen programs but about those we have designed with a specific goal in mind and which consequently avoid problems with Rice's theorem
  - One such difficulty is determining if something is an improvement
  - We can call this obstacle - "multidimensionality of optimization"
  - No change is strictly an improvement; it is always a tradeoff between gain in some areas and loss in others
  - [. . .] the system would have to figure out how to test intelligence at levels above its own (a problem which remains unsolved)
- Löb's theorem states that a mathematical system can't assert its own soundness without becoming inconsistent
- Procrastination paradox: Prevent the system from making modifications to its code since the system will find itself in a state in which a change made immediately is as desirable and likely as the same change made later. Since postponing making the change carries no negative implications and may actually be safe this may result in an infinite delay of actual implementation of provably desirable changes
- It may be the case that the minimum complexity necessary to become RSI is higher than what the system itself is able to understand. Paradoxically, as the system becomes more complex it may take exponentially more intelligence to understand itself and so a system which starts capable of complete self-analysis may lose that ability as it self-improves. Informally we can call it the Munchausen obstacle, the inability of a system to lift itself by its own bootstraps
- The system in question may be computationally irreducible and so it can't simulate running its own source code
  - An agent cannot predict what it will think without thinking it first
  - A system needs 100% of its memory to model itself, which leaves no memory to record the output of the simulation
- We can even attempt to formally prove the impossibility of intentional RSI process via a proof by contradiction:
  - An RSI R1 cannot algorithmically solve a problem of difficulty X, say Xi
  - If R1 modifies its source code after which it is capable of solving Xi, it violates our original assumption that R1 is not capable of solving Xi since any introduced solution could be a part of the solution process
  - Informally, if an agent can produce a more intelligent agent it would already be as capable as that new agent
  - A re-optimization problem is proven to be as difficult as optimization itself
- RSI Convergence Theorem: Regardless of the specifics behind the design of the seed AI used to start an RSI process all such systems, attempting to achieve superintelligence, will converge to the same software architecture
  - If an upper limit to intelligence exists multiple systems will eventually reach that level, probably by taking different trajectories, and in order to increase their speed will attempt to minimize the size of their source code eventually discovering smallest program with such level of ability

    - If intelligence turns out to be an unbounded property RSIs may not converge
- RSI software is really the challenge of creating a program capable of writing other programs

3 See also

4 References

- From Seed AI to Technological Singularity via Recursively Self-Improving Software
- The Singularity: A Philosophical Analysis