Theory of Uninventing

Andrew Gleibman

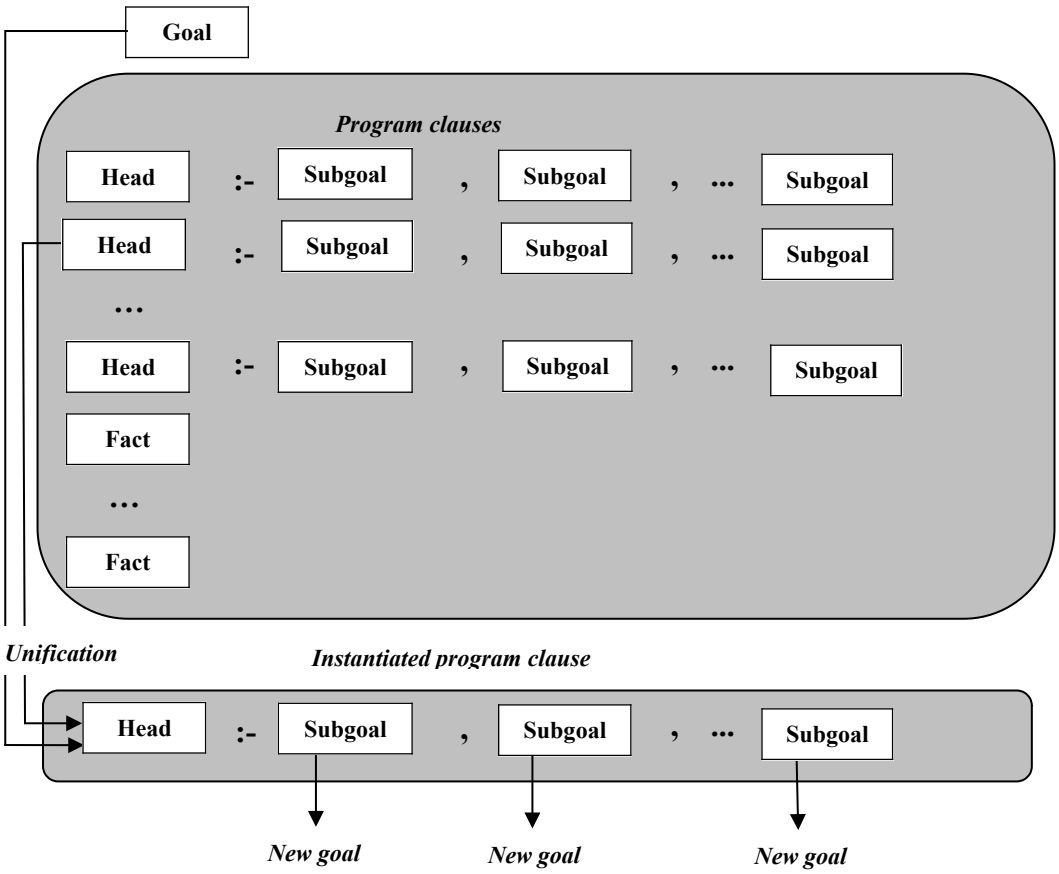
Sampletalk Research
sampletalk (at) gmail . com

Abstract

In this paper, the overuse of several popular mathematical formalisms is analyzed in view of algorithm design. We investigate situations where simplification, generalization, or even entire elimination (uninvention) of some formalism may lead to a more natural, more transparent, and more efficient algorithm design.

1. Introduction: Uninventing a Logical Predicate

Consider the following classical scheme of a logic program:



Here **Goal** is the initial program goal, which is analogous to a query in database theory terms. The goal may contain variables that should be instantiated. In order to resolve the goal, the classical scheme, proposed by J.A. Robinson in his Resolution Principle in 1965 [1], suggests trying to **unify** the goal with a **head** of some **program clause**. Program clauses present the data and the logical environment of the designed algorithm. In database terms they are analogous to the defined data and its usage logic. What is important here, the unification of a goal with the head of a program

clause being instantiated is done in the entire clause, including its **subgoals**. In this way new goals are created to be similarly resolved. **Facts** are program clauses without subgoals. The main program goal is considered *resolved* if all the involved subgoals are resolved.

Let us note how this scheme is applied in Logic Programming, mainly in Prolog language and its variants. All **goals** and **subgoals** typically correspond to the previously defined **predicates** of a specific first order predicate calculus (FOPC), which needs to be carefully designed. *This is a very expensive scheme*. Design of logical predicates is not simpler than the design of procedures and functions of a project in traditional programming. Attempts to create machine reasoning systems in 1970-2000 and later contain numerous examples of such logical predicates; see, e.g., [2].

Now, in paper [3] we suggest an alternative scheme, where instead of FOPC predicates, unrestricted texts, which may contain some variable symbols, are applied. This leads to the first observation of the present paper: **uninventing a logical predicate**, in the following sense. Rather than entirely disclaiming the usage of logical predicates we find an industrial case where they are overused, possibly leading the development of a large and expensive system to an absurd.

Here we preserve the main goal of a logic programming system: *do first order inference and find logical consequences of the main goal.* However, we avoid the concept of a logical predicate. Furthermore, the unrestricted texts, which are applied instead of the logical predicates (we call them ***samples***), can be restricted for presenting logical predicates in a text form. So, in this way we generalize the concept of a logical predicate.

We can extract *samples* from unrestricted natural sources such as natural language texts or biological sentences. *Unification* is, actually, matching, or alignment, of *samples* that have possible similarities. Remind that *samples* may contain variable symbols to be instantiated in this alignment.

2. Uninventing a Computer Operator

In paper [4] we further develop the usage of *samples* into a general definition of algorithm. S*amples* are considered as unrestricted strings of symbols in some alphabet and may include special symbols for designating variables. We define *unification*, *generalization* and *composition* of *samples* in detail and prove that any algorithm, in the Turing-completeness sense, can be composed of *samples*. This leads to the following reformulation of Church-Turing thesis:

THESIS 1: Every effective computation can be carried out using *samples* and some universal set of rules for combining such *samples*.

The universal set of rules can be fixed once forever. It may be implemented in any way, involving computer operators of any programming language. In [4] this set of rules is described in detail. We claim that this is enough for building any other algorithm using *samples* only.

Now, *samples* can be produced from various text data examples using the method of alignment, described in [4], which we call *data generalization*. This leads us to the following surprising reformulation of the thesis:

THESIS 2: Every possible computation can be carried out by alignment and matching of some data examples.

This is essentially an uninventing of a computer operator in its traditional sense. Just like in the Logic Programming example above, we do not entirely disclaim the usage of a computer operator. We only find a way to do without it, extracting all the necessary computation behavior immediately from data.

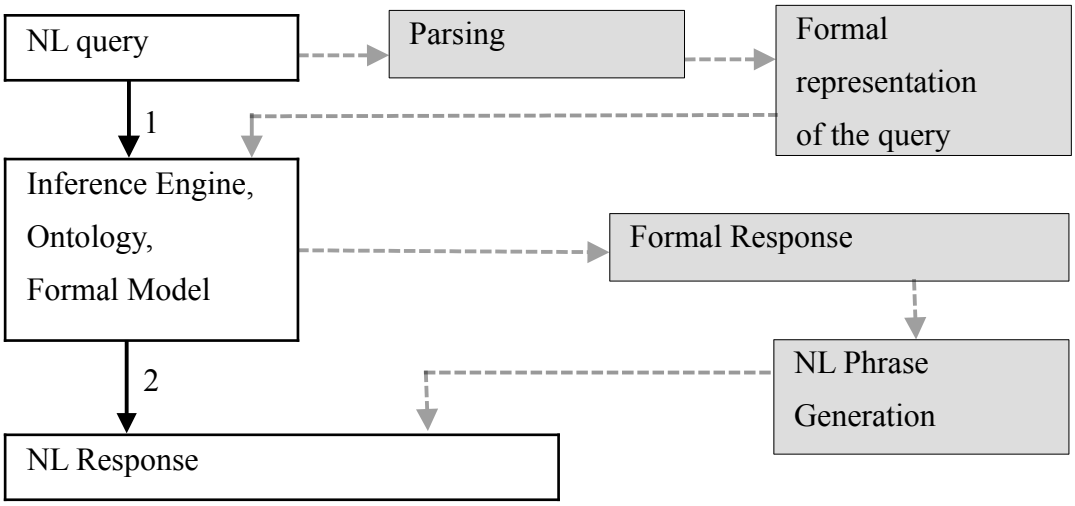
3. Uninventing Other Mechanisms

Again, by *uninventing* we mean finding a way to do without the mechanism in question rather than entirely disclaim the usage of this mechanism. We call this *uninvention case*, meaning a situation, which is prone to overusing some artificial construction. Our examples so far describe the overuse of some artificial constructions that are popular in math logic and computer science.

Papers [3] and [4] provide examples of avoiding additional formal constructions. One of them is a formal grammar:

Uninventing a Formal Grammar. We designed a natural language question-answering system based on *samples* that are extracted immediately from examples of natural language phrases. Here we explicitly avoid formal grammars.

The following scheme depicts this approach and its advantages:



Here is the legend of this scheme. Our implementation path, which is marked by continuous black arrows 1 and 2, assumes using only *samples* of data and their matching. Everything, including inference engine, ontology, first order logic reasoning, phrase analysis and synthesis, and even the formal logic of the modeled world, is implemented using *matching samples* only.

The traditional implementation path, marked in the scheme by gray boxes and dashed arrows, would assume the application of formal grammars and lead to the need to design predicates, grammars, a formal FOPC semantic model, and to implement a parser, a formal logic reasoning engine, and a NL phrase generator. This path is much more complex and expensive than our purely data-driven path. Details are provided in [3] and [4].

Proceeding in this way of defining *uninventions*, we can study many other existing and prospective *uninvention cases*. Below are some examples not related to our previous research.

Uninventing Vector Space Geometry. Pattern recognition and object classification methods, developed in the last decades, are characterized by a heavy usage of vector space geometry (SVM, clustering methods, dimensionality reduction, kernel methods etc). Today, Convolution Neural Network methods, based on a much simpler math model, successfully replace this in many applications.

Uninventing Physical Laws. Attempts are made [6] to replace formal representation of physical laws, which are applied for predicting the behavior of the Universe, by a neural network trained on the observational data. Furthermore, the entire universe on its most fundamental level is considered as a neural network.

Uinventing the Wheel. Using modern technology, we can build a vehicle that does not need

a smooth road and can move using multiple flexible legs, which dynamically analyze the terrain structure and apply it for motion and for dynamic support. In many specific situations this can be used for preserving the natural terrain environment.

The question, raised here, is how many *uninvention cases* exist around us. Our hypothesis is that the more popular a formalism, mechanism or invention, the larger is the probability that there exists its *uninvention case*.

4. Uninvention Cases Recapitulated

Uninvention case is a situation where some sub-mechanism **A (**typically quite popular) of some another mechanism **B** can be simplified, generalized, or even entirely replaced with some another mechanism **A',** without compromising its function in the mechanism **B**. Here we intentionally apply the term *mechanism* instead of the term *formalism*, hinting that other kind of devices, such as mechanical devices or processes, can be analyzed in the same fashion.

We described *uninvention cases* that provide new insights related to the cost, applicability, and further development of the whole mechanism **B**. We may consider *uninventing* as a means to replace too artificial mechanisms by nature-inspired ones. Indeed, replacing FOPC predicates by *samples* of NL phrases, as we did in Section 1, demonstrates the possibility to extract the reasoning patterns immediately from natural texts.

5. Discussion: Minimization of a Priori Knowledge Applied in Reasoning Systems

Research, presented here, is related to a minimization of a-priori knowledge, applied in reasoning systems. This framework is characterized by the following paradox, which is visible in some inductive reasoning systems:

PARADOX: The less formal knowledge we apply, the more powerful reasoning we can extract immediately from observational data.

Informally, this means that when we involve less initial formal knowledge into our inference, our treatment of data is less biased and can lead to a wider spectrum of generalizations. For the case of symbolic logic we came to an *asymptotic limit* of such minimization, which is called First Order String Calculus (FOSC) [4], as opposed to First Order Predicate Calculus (FOPC), and to the concept of Generalization Universe, which contains all possible generalizations of a set of symbolic data in form of *samples* (see [4], p.104, Definition 2.5).

This framework provides an unexpected way to handle biological sequences that we cannot understand today, to build natural language processing applications from examples without the concept of a formal grammar, to decipher unknown codes, and to extract formal theories and algorithms immediately from unrestricted text data in the fashion of *inductive logic programming*, but without the expensive need to invent logical predicates.

In a way, this is a formal framework for modeling human intuition. This minimization also provides a special treatment of the Church-Turing thesis, where only data-driven calculations are applied.

This framework can be expanded for non-text data. In the past we did such minimization for extracting astronomical equations immediately from observational data [5].

6. Conclusion

We analyzed situations where some artificial mechanism, or process, which typically is quite popular, is applied without careful justification in another mechanism or process. We call such a situation *an uninvention case*, meaning that said mechanism is overused and needs to be replaced.

Analysis of *uninvention cases* may lead to new insights related to the involved technologies. Most *uninventon cases,* which we analyzed, can be resolved either by a generalization of the mechanism in question or by replacing it with some nature-inspired mechanism or technique.

In a way, artificial mechanisms contradict, or even challenge, nature. We analyze examples, where nature-inspired techniques may successfully replace such mechanisms and provide new insights about the whole process or device. This is an additional reason why we call such situations *uninvention cases*.

Investigation of *uninvention cases* in various science or technology fields may reveal overuse of artificial mechanisms and help better understand the limitations of existing technologies.

References

[1] Robinson J. A. A Machine-Oriented Logic Based on the Resolution Principle. *J. Assoc. Comput. Mach.* **12**, 23-41, 1965.

[2] Lenat, D.B. CYC: A Large-Scale Investment in Knowledge Infrastructure. Communications of the ACM 38(11), December 1998, DOI:10.1145/219717.219745.

[3] Gleibman, A.H., Knowledge Representation via Verbal Description Generalization: Alternative Programming in Sampletalk Language. In: Workshop on Inference for Textual Question Answering, July 9, 2005 – Pittsburgh, Pennsylvania, pp. 59–68, AAAI 2005 - the Twentieth National Conference on Artificial Intelligence, https://www.aaai.org/Library/Workshops/2005/ws05-05-010.php

[4] Gleibman, A.H., Intelligent Processing of an Unrestricted Text in First Order String Calculus, M.L. Gavrilova et al. (Eds.): Trans. On Comput. Sci. V, LNCS 5540, pp. 99–127, 2009. © Springer-Verlag Berlin Heidelberg 2009, https://link.springer.com/chapter/10.1007/978-3-642-02097-1_6

[5] Gleibman, A.H., Reasoning About Equations: Towards Physical Discovery. The Issue of the Institute of Theoretical Astronomy of the Russian Academy of Sciences No.18, 1992, 37 pp. In Russian, http://www.biblus.ru/Default.aspx?book=7b36s38i8.

[6] Vitaly Vanchurin. The world as a neural network, https://arxiv.org/abs/2008.01540.