CREATING A UNIVERSAL INFORMATION LANGUAGE

OSCAR BENDER-STONE

ABSTRACT. Welkin is a formalized programming language to store information. We introduce its use cases and rigorously define its syntax and semantics. From there, we introduce the bootstrap, making Welkin completely self-contained under a meta-theory based on combinators, equivalent to a provably minimal fragment of arithmetic.

CONTENTS

1. INTRODUCTION

Information Management (IM) is an open area of research as a result of the depth and breadth of disciplines. In terms of depth, many areas are often specialized, requring an immense understanding of the broader concepts involved and nomenclature used. This is evident in the sciences, as explored in [1] and [2]. Additionally, in terms of breadth, creating common representations shared accross subdisciplines can be difficult. In mathematics, for example, … [FINISH]. Moreover, creating a standardized form accross communities is challenging. In other subjects, like the social sciences, there are no standard terms, and the majority of cited references are books, which are not indexed by many databases [3]. More broadly, IM *itself* is divided from distinct approaches [CITE]. Some authors equate IM to Knowledge Management (KM) and assert that information must be true. These problems posed by broadly and faithfully capturing subjects demonstrate the enormous task of effective information management.

In attempt to address these challenges, several solutions have been proposed, but none completely fix these issues. In the sciences, a group of researchers created the Findable Accessible Interoperable Resuable (FAIR) guidelines [4]. . However, . Another proposal is . Unfortunately, this project has focused primarly on logic . In addition to …, Burgin's work on a theory information [5], [5] a comprhensive. He does

TODO: Dicuss truth here and Burgin!

include a free parameter in the theory but does not demonstrate how it can interact in a formal system. His chapters on complexity theory, where he explores the smallest size .

This thesis introduces a language to resolve these issues. I call this language **Welkin**, based on an old German word meaning cloud [6]. The core result of this thesis is proving that Welkin is a) universal, b) scalable, and c) standardized. The core idea is to generalize Burgin's free parameter and enable arbtitrary representations in the theory, controlled by a computable system. The notion of representaiton builds on Peirce's semiotics, or the study of the relationship between a symbol, the object it represents, and the intepreter or interpretation that provides it that meaning [7]. Moreover, to address queries on the validity of truth, we use a relative notion that includes a context managed by a formal system. Truth can then be determined on an individual basis, providing flexibility to any discipline. The focus then shifts to the usefulness of representaitons based on a topological notion of "collapse" or "coherency". This approach is inspired by coherentism, a philosophical position that states truth is determined in comparison to other truths. [8]. We incorporate ideas from coherentism to identify which representations identify their corresponding objects, and we define information as an invariant under these coherent representations. We include definitions on a *working* basis as what is most practical, not an epistemological stance that can be further clarified in truth systems.

1.1. Goals.

- **Goal 1: Universality.** The language must include unspecified, user created parameters to accomodate for arbitrary concepts and ideas.
- **Goal 2: Standardization.** The language needs a rigorous and formal specification. Moreover, the bootstrap must be formalized, as well as an abstract machine model. The grammar and bootstrap must be fixed to ensure complete forwards and backwards compatbility.
- **Goal 3: Efficiency.** Local queries in the database, determining if there is enough "explicit" information, must be efficient.

1.2. Organization.

- Section 2. Foundations: defines the meta-theory use and its connections to first-order logic and fragments of arithmetic.
- Section 3. Information Organization. Develops the optimal informal system (w.r.t. to a metric defined in this system) to satisfy Goal 3.
- Section 4. Defines the syntax and semantics.
- Section 5. Bootstraps Welkin by proving that there is a Welkin node that contains enough information about the standard. Fulfills Goal 2 with both the Standard AND the complete bootstrap.
- Section ?. Prototype. Time permitting, develop a prototype to showcase the language, implemented in python with a GUI frontend (Qt) and possibly a hand-made LL(1) parser.
- Section 8. Conclusion. Reviews the work done in the previous sections. Then outlines several possible applications.

2. Foundations

This section establishes the theory underlying Welkin, capturing representations, consisting of an entity, a symbol, and an intepreter. We develop representations using *units*, or arbitrary entities that are denoted through a numeric ID.[1] Units have two important properties. First, they can be be broken down further or form larger units. Second, their respective symbol can be manipulated by a partial computable function. We impose no implementations on these properties, but instead analyze their usefulness through the coherency of a unit, or how much structure it distinguishes.

2.1. Motivating Example: Maps.

We start with a motivating example that equally serves as a useful metaphor: geographic maps. Consider a traveler A exploring a new area. To track their journey, they take a piece of paper and draw a box to represent the landscape. This box is a unit. As they travel, they record landmarks and paths as their own symbols. Each of these are units, with an important property: they are denoted through *distinct* symbols. Without distinct symbols, A could confuse one landmark with another and become lost. This is a foundational kind of coherency, namely non-triviality. The map is neither empty nor represents all entities with a single symbol.

TODO: Draw several figures here! Will be easier to keep track.

Suppose, now, that A meets another traveler B, drawing a different non-trivial map. They compare maps and find they wrote down different landmarks, but agree on their overlapping paths. With this, A and B could settle on some notation and combine their maps. For instance, to identify their specific landmarks, they could apply a special prefix, say A and B, respectively. The *combination* of these maps are then coherent, because it faithfully encode the map, now with more information than before.

The scenario with A and B is a straightforward case. But we want to investigate more difficult ones. Consider a third traveler, C, with a completely divergent map.

In this specific context, the geographically distinguished landmarks may be more appropriate for travel than the random marks of the latter. However, suppose these marks distinguished an artistically relevant feature, or possibly a hidden treasure. How would the latter communicate this to the former?

To communicate between these maps, we need an appropriate notion of coherency.

2.2. Definitions.

Now we develop the formal framework to discuss information in terms of units, enabling a complete mechanization of Welkin's meta-theory.

Definition 2.2.0. The **alphabet of binary strings** is $\mathcal{A}_{\text{bit}} ::= 0 \mid 1 \mid . \mid w$. A **binary string** is defined recursively: the symbols 0 or 1 are strings, or if w is a string, then $w.0$ and $w.1$ are strings. We abbreviate $w.w'$ to ww'.

Definition 2.2.1. The **alphabet of units** is $\mathcal{A}_{\text{unit}} = u$. A **unit ID** is combination of symbols u_b, where b is a binary string.

[1]The word unit is inspired by a cloud. A cloud can be broken down further or be part of a larger group of clouds. Additionally, clouds can be transformed, which is reflected in units through operations on their symbols.

Example 2.2.2. Consider a house with a dog and a cat. We can represent the house as unit H, the dog as unit D, the cat by unit C. We can impose that H contains both C and D.

New units can be made as follows:

- Given units A and B, $\{B, A\}$ is a unit.

Theorem 2.2.3. *A unit is coherent relative to a context iff the unit and that context are coherent.*

Remark 2.2.4. This theorem is a natural generalization of consistency in first-order logic. We will frequently rely on this result throughout the thesis.

Definition 2.2.5. Information over a unit u is a unit u' such that $u \equiv u'$ iff $v_u = v_{u'}$. In other words, information is an invariant for a unit modulo $\equiv$.

Theorem 2.2.6. *A representation is preserves information modulo $\equiv$ iff the representation modulo $\equiv$ is coherent.*

Remark 2.2.7. This theorem enables truth management via specific contexts, specified as units. The task of finding core truths is then free, left open to flexibility accomodate for any truth management.

2.3. **Translations Between First Order Logic.**

- Want easy access to first order logic
  - Review literature. Notable examples:
    - SMT solvers in Rocq + Lean (via monomorphization of types)
  - Problem: abstractions are hard to convey! Lots of “bloat”
  - BUT SMT solvers are very well established, particularly with Gödel’s completeness theorem.
  - How to get best of both worlds? Solution: slates!
- First step: define extension to first order logic (let’s call it, say, FOL(Slate))
  - Add slates as a special sort, but focuses on first order terms.
    - Emphasize that there are FOL theories **weaker** than combinators.

    So, with a coherency argument, argue that FOL can be powerful **precisely because** RE is possible, WITH the combination of the completness theorem. (Not possible in all logics!).

• Second step: show that FOL(Slate) is equivalent to FOL by treating slates
as an additional sort.

• Straightforward, but emphasize rule on slates on making meanginful/useful
abstractions!

• IF time allows, provide experiments, but mostly argue why, based
on the argument for slates, this would work.

• Argue that you could AT LEAST embed the necessary abstractions via slates.
And organization will help show this is feasible with a theoretical argument (but it’s not exponential time. It is (hopefully) ACTUALLY feasible.)

- Final step: show that there is an equivalent embedding that **preserves** slates.
  - Important part: preservation up to iso!

3. Universal Systems

- Provide previous section (translation to FOL) as a major example
- Generalize from the case of a formal system from an earlier draft
    - Earlier definition: (D, R), with D a grammar and R a set of RE rules
    - Universal system: U = (D_U, R_U) is universal if, for each formal system S,

  there is a term t in D_U such that derivations in S are reflected and preserved via t in D_U. So they are faithfully encoded
    - Earlier proof: a system is universal iff it induces a comptuable, RE

  full sub-category of the category of formal systems.
    - Refine these ideas to use slates + coherency from before.

  Can involve more ambitious encodings!
    - Also develop reflection!
- Hint at topic of next section, or smooth out transition. Next

section is discussing **which** universal system to use or how to effectively translate between them

4. Information Organization

- Main question: **which** universal system to choose? Is this practical?
    - What is a suitable criterion for a base theory?
    - Recall aim: want to mechanically store systems for a database
        * What if possible performance degredation? Will we get stuck

      if we start with one architecture? Will we have to adjust later?
        * Aim is to ensure architecture is completely flexible and

      can automatically adapt
        * One key metric: ability to store as many systems coherently as possible,

      i.e., store as much information as psosible
    - Main problem: Blum's speedup theorem
        * Briefly generalize this for slate logic
        * Show that no single way to completely organize systems based on a computable metric.

      This is part of the need for new search techniques!
        * Want to separate search from storage though, but we want to improve

      stored results **with** new results. This forms the idea behind the database architecture: have a simple way to store results that automatically gets better with new techniques/results.
        * Need explicit proofs for this! Not sure how to store certificates…

4.1. **Impossible Classes.**

The reason to restrict our transformations is two-fold. First, we need to ensure we can *verify* them efficiently. Determining whether a morphism between two formal systems exist can be reduced to the Halting problem, and is therefore not practical for defining an optimal formal system. Second, if we include those tranformations that we *can* effectively check, no optimal formal system exists.

Theorem 4.1.0. *With respect to the class of all computable transformations that can be computably verified, there is no optimal formal system.*

4.2. **Efficient Querying.**

Instead of making proofs most efficient as is, we want to support finding optimal representations. But we want to do this from an efficiently queryable system, which *is* the most optimal.

5. The Welkin Language

5.1. **Syntax.**

- Want to include essential components and use slates

for the rest. May incluce lemmas to show **why** these notions are useful (but make be part of a separate section).

- Scoping: need to avoid collisions! With slates, can expand further and generalize to, e.g., nodes with sharing
- Rewrite rules: need the formal system part to

be clear that is ultimately symbolic.

5.2. **Semantics.**

- Semantics on AST
  - Terms: graphs
  - For ease of use, include a null node

  that is the root of the tree. This represents the module itself.
- For information organization: integrate with previous section
  - Emphasize how this is a useful tool and can ensure

  new information content is being created (at least, that can be distnguished from the current module). If already existing, but that doesn't match the user's expectations, they need to refine it! OR, maybe it **does** match similarly with something else! (e.g., hidden connections between math and music)
- Emphasize pragmatics as well, via slates

6. Bootstrap

- Provided in bootstrap.welkin file or similar. Main module is welkin, which needs to:
  - Provide slates, so inductive definition of binary strings + variables
  - Explain combinators
  - Explain the basics of universal systems. This is very important!
    * Can elide proofs IF there is enough information content.
    * TODO: figure out how proofs might work in this setting
  - Provides syntax and semantics of Welkin itself
- This thesis: will prove that the AST generated from this file

is correct AND that it does, with a suitable interperation bootstrap itself.

- Explain how slates expand the envelope for implementations, BUT ensures

that the final product, the syntax + semantics checkers and the information organization, can be externally seen!

7. Conclusion

- Review of thesis
    - Developed slate logic + bi-translation with FOL
    - Developed locally optimal organizational technique
  that can improve based on annotations/certificates
    - Introduced the language, with a straightforward
  graph syntax and semantics
    - Builds upon the last section
    - Bootstrapped standard + used coherency condition
- Significance
    - Backwards AND forwards compatbile standard
  that bootstraps itself. Easy for implementations!
    - Applications to any human subject
        * Sciences
        * Liberal arts
        * Economics
        * Etc.
- Future work
    - Programming language semantics + synthesis
        * Incorporate broader aspects + intent of users! ESSENTIAL for new programming languages to be able to discuss pragmatics in some way!
        * Also reproducible AND executable specifications,
      though creating an engine to execute these is far beyond the scope of the thesis
    - Organizing large corpuses of human text
    - Numerous applications to AI and improving results
        * Emphasize role of symbol grounding problem in AI

References

1. Fanelli, W., Daniele AND Glänzel: Bibliometric Evidence for a Hierarchy of the Sciences. Plos One. 8, 1–11 (2013). https://doi.org/10.1371/journal.pone.0066938
2. Casadevall, A., Fang, F.C.: Specialized Science. Infection and Immunity. 82, 1355–1360 (2014). https://doi.org/10.1128/iai.01530-13
3. Archambault, É., Vignola-Gagné, É., Côté, G., Larivière, V., Gingras, Y.: Benchmarking scientific output in the social sciences and humanities: The limits of existing databases. Scientometrics. 68, 329–342 (2006)
4. Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Silva Santos, L.O.B. da, Bourne, P.E., Bouwman, J., Brookes, A.J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S.C., Evelo, C.T.A., Finkers, R., González-Beltrán, A.N., Gray, A.J.G., Groth, P., Goble, C.A., Grethe, J.S., Heringa, J., Hoen, P.A.C. 't, Hooft, R.W.W., Kuhn, T., Kok, R.G., Kok, J.N., Lusher, S.J., Martone, M.E., Mons, A., Packer, A.L., Persson, B., Rocca-Serra, P., Roos, M., Schaik, R.C. van, Sansone, S.-A., Schultes, E.A., Sengstag, T., Slater, T., Strawn, G.O., Swertz, M.A., Thompson, M., Lei, J. van der, Mulligen, E.M. van, Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J., Mons, B.: The FAIR Guiding Principles for scientific data management and stewardship. Scientific Data. 3, (2016)
5. Burgin, M.: Theory of Information. World Scientific (2009)
6. Dictionary, O.E.: welkin, n., https://www.oed.com/dictionary/welkin_n
7. Atkin, A.: Peirce's Theory of Signs, (2023)
8. Bradley, F.H.: The Principles of Logic. Mind. 32, 352–356 (1923)

Department of Mathematics, University of Colorado at Boulder, Boulder, CO
Email address: oscar-bender-stone@protonmail.com