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, requiring an immense understanding of the broader concepts involved and nomenclature used. This is evident in the sciences, as explored in [Fan13], [CF14]. Additionally, in terms of breadth, creating common representations shared across sub-disciplines can be difficult. In mathematics, for example, has extremely diverse disciplines, and connecting these areas is an open problem in scalability [Car+21]. Moreover, creating a standardized form across 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 [Arc+06]. More broadly, IM *itself* is divided from distinct approaches that lack interoperability [AJ23]. Some authors equate IM to Knowledge Management (KM) and assert that information must be true [Edw22]. These problems posed by broadly and faithfully capturing subjects demonstrate the enormous task of effective IM.

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 [Wil+16]. Instead of providing a concrete specification or implementation, FAIR provides suggestions to encourage better storage of scientific information. However, multiple papers have

outlined problems with these overarching principles, including missing checks on data quality [Gui+25], missing expressiveness for ethics frameworks [Car+21], and severe ambiguities that affect implementations [Jac+20]. Along with the sciences, there are several proposals in mathematics, including QED project and the OpenLogic Project. ... While they have advanced storage for mathematical information, OpenLogic these project has focused primarily on logic ... These proposals, even focused on specific fields, fail to accommodate for all of the mentioned challenges.

In addition to these proposals, Burgin's work on a theory of information [Bur09], [Bur09] comprehensively includes many separate areas for IM as a whole. He provides flexible definitions through a free parameter, an "infological system" that encompasses domain specific terminology and concepts. He then proceeds to mention many areas in the natural sciences, and connects his theory back to related mathematical studies, including Algorithmic Information Theory. Despite the large coverage of fields, Burgin does not closely tie the free parameter with his formal analysis, making it unclear how to use this in a practical implementation. Each of these proposals has severe shortcomings and highlights major obstacles for IM.

This thesis introduces a language to resolve these issues. I call this language **Welkin**, based on an old German word meaning cloud [Dic25]. 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 arbitrary representations in the theory, controlled by a computable system. The notion of representation builds on Peirce's semiotics, or the study of the relationship between a symbol, the object it represents, and the interpreter or interpretation that provides it that meaning [Atk23]. 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. [Bra23]. 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.

TODO: Determine if this is now unnecessary.

- 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, centered around representations. Loosely, a **representation** is a mapping from a **sign** to a **referrant** via an **interpreter**. Each of these components are defined as *units*, which are entities referred to by a numeric ID.[1] Units have two important properties. First, they can be broken down and combined. Second, units have a sign that can be manipulated by any partial computable function. Both of these properties are used to prove universality, in that any free parameter describlae by a partial computble function can be described as a unit, see Theorem 2.3.3.

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.

Now, suppose A mistook some shadows for a ravine, and adds a blockade on the map.[2] The map still accurately represents *some* of the landmarks, but not the open path where the symbolic ravine is added. How can this mistake be identified? A direct approach would be to add *both* assertions together, that the map is blocked and not. This makes the combination of a new item incoherent, as the new representation fails to be faithful. There are multiple ways to then *fix* the issue. In one way, the original map, without any checks, can be recorded as a revision, and then another revision can be made that removes the ravine. This is addressed more in Section 3.

> **TODO:** Draw several figures here.

---

[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.

[2] Credit to Professor Kearnes for inspiring this example.

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. To keep this section self-contained, we explicitly provide all recursive definitions.

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

For simplicity, we extend the alphabet to include decimal and hexadecimal.

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

We now define representations using unit IDs as a base notion.

Definition 2.2.2. Units are recursively defined:
- **Base case:** binary strings are units.
- **Recursive step:**
    - **Parts:**: if $u_1, .., u_n$ are finitely many units, then so is their combination.
    - **Representations:** If u, w, v are units, so is $u \leftarrow w - v$. We say v

    refers to u **via** w,

or conversely, u is **referred to** by v **via** w.

Parts of units are denoted as $u.w$. Scoping is included to provide namespces.

Example 2.2.3. 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. We can consider an abstract entity A as well, and could say that A represents C and D.

New units can be made as follows:
- Given units A and B, $\{B, A\}$ is a unit.

Example 2.2.4. Consider the recursive definition of a binary tree: either it is a leaf node, or it contains two distinct nodes, left and right. We can model this as follows. We consider a unit T (tree), as well as symbols for L (left), R (right), and E (end/leaf node). We could also add a symbol for C, child. T then contains a recursive definition: E represents T, and for nodes A, B, if A represents T and B represents T, then so does their combination.

An important idea in this example is that the abstraction could be defined *first*, or a concrete model could. For this reason, the choice of how entities are represented is flexible.

Theorem 2.2.5. *A unit is coherent relative to a context iff the unit and that context are coherent.*

Remark 2.2.6. This theorem is a natural generalization of consistency in first-order logic. We will frequently rely on this result throughout the thesis.

Definition 2.2.7. Information over a unit u is a unit u' such that $u \equiv u'$ iff $I_u = I_{u'}$. In other words, information is an invariant for a unit modulo $\equiv$.

Theorem 2.2.8. *A representation is preserves information modulo $\equiv$ iff the representation modulo $\equiv$ is coherent.*

Remark 2.2.9. This theorem enables truth management via specific contexts, specified as units. The task of finding core truths is then free, left open to flexibility accommodate for any truth management.

Example 2.2.10. First Order Logic

2.3. **Universal Systems.**

Theorem 2.3.11. *Any computable function can be processed as a representation.*

Note that there are multiple ways to prove Theorem 2.3.3, infinitely in fact. This motivates the following definition.

Definition 2.3.12. A universal representation system is a unit that can represent any representation.

Theorem 2.3.13. *A unit is a universal representation system if and only if it can represent any partial computable function. Moreover, any universal representation system can represent any universal representation system. In particular, representing itself is called* **reflection**.

The term *universal* is specifically for expressing *representations* symbolically. The free parameter still needs to be included and is an additional feature on top of partial comptuable functions. However, the *management* of these symbols is done entirely with partial computable functions.

The next section discusses the issue of *managing* these infinite choices.

3. 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...

3.1. **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 3.1.0. *With respect to the class of all computable transformations that can be computably verified, there is no optimal formal system.*

3.2. **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.

4. THE WELKIN LANGUAGE

4.1. **Syntax.**

- Want to include essential components and use slates

for the rest. May inclue 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.

4.2. **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

5. 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 organi-
zation, can be externally seen!

6. 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

[Arc+06] Archambault, Éric, Vignola-Gagné, Étienne, Côté, Grégoire, Larivière, Vincent, and Gingras, Yves, Benchmarking scientific output in the social sciences and humanities: The limits of existing databases, Scientometrics **68** (2006) 329–342.
[Atk23] Atkin, Albert, Peirce's Theory of Signs, Spring2023 ed., Metaphysics Research Lab, Stanford University, 2023.

[AJ23]   Auth, Gunnar, and Jokisch, Oliver, A systematic mapping study of standards and frameworks for information management in the digital era, Online Journal of Applied Knowledge Management **11** (2023) 1–13.

[Bra23]   Bradley, F. H., The Principles of Logic, Mind **32** no. 127 (1923) 352–356.

[Bur09]   Burgin, Mark, *Theory of Information*, World Scientific, 2009.

[Car+21]   Carette, Jacques, Farmer, William M., Kohlhase, Michael, and Rabe, Florian, Big Math and the One-Brain Barrier: The Tetrapod Model of Mathematical Knowledge, The Mathematical Intelligencer **43** no. 1 (2021) 78–87.

[Car+21]   Carroll, Stephanie Russo, Herczog, Edit, Hudson, Maui, Russell, Keith, and Stall, Shelley, Operationalizing the CARE and FAIR Principles for Indigenous Data Futures, Scientific Data **8** no. 1 (2021) 108.

[CF14]   Casadevall, Arturo, and Fang, Ferric C., Specialized Science, Infection and Immunity **82** no. 4 (2014) 1355–1360.

[Dic25]   Dictionary, Oxford English, welkin, n., Oxford University Press, 2025.

[Edw22]   Edwards, John S., Where knowledge management and information management meet: Research directions, International Journal of Information Management **63** (2022) 102458.

[Fan13]   Fanelli, Wolfgang, Daniele AND Glänzel, Bibliometric Evidence for a Hierarchy of the Sciences, Plos One **8** no. 6 (2013) 1–11.

[Gui+25]   Guillen-Aguinaga, Miriam, Aguinaga-Ontoso, Enrique, Guillen-Aguinaga, Laura, Guillen-Grima, Francisco, and Aguinaga-Ontoso, Ines, Data Quality in the Age of AI: A Review of Governance, Ethics, and the FAIR Principles, Data **10** no. 12 (2025).

[Jac+20]   Jacobsen, Annika, Miranda Azevedo, Ricardo de, Juty, Nick, Batista, Dominique, Coles, Simon, Cornet, Ronald, Courtot, Mélanie, Crosas, Mercè, Dumontier, Michel, Evelo, Chris T., Goble, Carole, Guizzardi, Giancarlo, Hansen, Karsten Kryger, Hasnain, Ali, Hettne, Kristina, Heringa, Jaap, Hooft, Rob W.W., Imming, Melanie, Jeffery, Keith G., Kaliyaperumal, Rajaram, Kersloot, Martijn G., Kirkpatrick, Christine R., Kuhn, Tobias, Labastida, Ignasi, Magagna, Barbara, McQuilton, Peter, Meyers, Natalie, Montesanti, Annalisa, Reisen, Mirjam van, Rocca-Serra, Philippe, Pergl, Robert, Sansone, Susanna-Assunta, Silva Santos, Luiz Olavo Bonino da, Schneider, Juliane, Strawn, George, Thompson, Mark, Waagmeester, Andra, Weigel, Tobias, Wilkinson, Mark D., Willighagen, Egon L., Wittenburg, Peter, Roos, Marco, Mons, Barend, and Schultes, Erik, FAIR Principles: Interpretations and Implementation Considerations, Data Intelligence **2** nos. 1–2 (2020) 10–29.

[Wil+16]   Wilkinson, Mark D., Dumontier, Michel, Aalbersberg, IJsbrand Jan, Appleton, Gabrielle, Axton, Myles, Baak, Arie, Blomberg, Niklas, Boiten, Jan-Willem, Silva Santos, Luiz Olavo Bonino da, Bourne, Philip E., Bouwman, Jildau, Brookes, Anthony J., Clark, Tim, Crosas, Mercè, Dillo, Ingrid, Dumon, Olivier, Edmunds, Scott C., Evelo, Chris T. A., Finkers, Richard, González-Beltrán, Alejandra N., Gray, Alasdair J. G., Groth, Paul, Goble, Carole A., Grethe, Jeffrey S., Heringa, Jaap, Hoen, Peter A. C. 't, Hooft, Rob W. W., Kuhn, Tobias, Kok, Ruben G., Kok, Joost N., Lusher, Scott J., Martone, Maryann E., Mons, Albert, Packer, Abel Laerte, Persson, Bengt, Rocca-Serra, Philippe, Roos, Marco, Schaik, Rene C. van, Sansone, Susanna-Assunta, Schultes, Erik Anthony, Sengstag, Thierry, Slater, Ted, Strawn, George O., Swertz, Morris A., Thompson, Mark, Lei, Johan van der, Mulligen, Erik M. van, Velterop, Jan, Waagmeester, Andra, Wittenburg, Peter, Wolstencroft, Katy, Zhao, Jun, and Mons, Barend, The FAIR Guiding Principles for scientific data management and stewardship, Scientific Data **3** (2016).

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF COLORADO AT BOULDER, BOULDER, CO
Email address: oscar-bender-stone@protonmail.com