

Symbolic Information In Computing Devices

By H. Paul Zellweger

Introduction

Intuitively, 0's and 1's on paper are clearly different from their electronic counterparts on the computer. On paper, their representations are flat and motionless. On the computer, zeroes and ones travel at blinding speeds in 3-dimensional space. Yet, STEM research treats all zeroes and ones the same, regardless of their media. The author considers this oversight not only a critical loss of information but the reason why the actual limitations of mechanical computing remain unanswered. He attributes this conflation of digital symbols in different media on two critical articles that go off in two different directions. They are: 1) Claude Shannon's theory of communications that introduces the bit as a unit of information [p. 380, 1], and 2) Newell and Simon's Physical Symbol System Hypothesis (PSSH) that promotes AI by arguing that digital symbols create new expressions by symbolic manipulation [2]. The author believes that the mathematics in Shannon's article casts a long shadow over the Newell and Simon's hypothesis, in spite of the fact that their article focuses on the interaction of 0's and 1's on the computer.

In his 1948 article, Shannon makes it clear that communication signals represent physical values. He identifies them as 'bits', a portmanteau of Binary Digits. His description of bits is quite similar to 0's and 1's printed on paper, flat and 2-dimensional. He goes on to point out that bits form larger units of information to represent symbols. In the 1960's, Newell and Simon argue that physical symbols "operate on expressions to produce other expressions" [p. 116, 2]. They credit dynamic memory structures for making this possible. However, their description of symbolic manipulation is limited to a high-level discussion of programming languages like LISP. Regrettably, fifty years later this fundamental understanding of digital symbols remains unchanged [3].

To investigate the deep structure of physical symbols, the author turns to a newly discovered uniform pattern of data in the RDBMS. It is called the Aleph data relation. Like Borger's namesake short story, the Aleph and its models are a portal into digital computation. In this mechanical setting, the Aleph is a parent/child data relation that is ubiquitous throughout the RDBMS. In architectural terms, it is a design pattern. It is self-similar to the tree structure and, like a fractal, it has scaling symmetry. These features enable decision trees to emerge from relational data by recursive algorithms. From an engineering perspective, the Aleph is a data object that is analogous to objects in object-oriented programming languages. In a file or data stream, the spatial proximity of parent and child data translates its physical symbols into an IF-THEN method. Child data always comes after a parent. And finally, in terms of mathematical generalizations, the Aleph is an abstract object whose spatial embodiment of logical implication explains why a hardware address always implies its content.

Mark Burgin's named set theory led to the discovery of the Aleph data relation []. In fact, his mathematical theory resulted in an extensive system of nested models that represent tree structures. These models program relational data to self-reference data symbols at multiple levels of detail. They transform relation data into menu data for "The Database Taxonomy" decision tree interface [4]. End-users navigate down its data topics to pinpoint information managed by the database. In effect, Burgin's models turn the database system inside out. They also reduce its complexity and detail. Accordingly, they align two notoriously complex systems, Codd's relational model and the von Neumann machine, to unify their details. This system of models creates a cross section of the database on a computer that penetrates deep down into the hardware details. One model highlights how the database system relies on logic gates for pattern matching on input and indirect addressing for output. Another model in the system shows how the same data symbol alternates between these two I/O operations. Subsequently, Burgin's models reveal how symbolic manipulation occurs at a deeper level of 0's and 1's. The author identifies this more in-depth view of digital symbols as meta-symbols: physical-values and constructed-types []. The paper focuses on the relationship between the Aleph data relation its physical symbols to explore the profound nature of zeroes and ones on the computer.

To study the relationship, the author draws on three cognitive tools. First, he relies extensively on Mark Burgin's theory of named sets because he believes, it brings about a radical simplification of ordered complexity¹. Second, he employs Herbert A. Simon's *The Science of the Artificial*, and its use of the *interface* metaphor, to analyze the exterior/interior views of artifacts [p. 6-12, 7]. In the paper, the author extends this interpretation of objects to include mathematical expressions. And finally, he deploys the *container* metaphor used throughout database research to

¹ According to Warren Weaver, there are two types of complexity: ordered and chaotic.

bridge the widely accepted view of physical symbols, as values, with an overarching view of their addressability, as the content of 3-dimensional containers.

Burgin's Mathematical Theory

Burgin's named set theory applies the rigor and precision of mathematics to the study of names. Research in this area goes back to Frege's distinction between reference and meaning. To adapt this idea for structures today, Burgin adds a third component, a connection between a reference and its meaning. He calls this new triadic structure the fundamental triad [p. 40, 5].

This formal study of names centers on structures, in general, and on mathematical structures, in particular. All mathematical structures conform to a well-defined triplet expression. When a mathematician discovers something new, he or she intuitively treats this triplet as a reference. Drawing on the fundamental triad, Burgin makes the connection between this triplet and its mathematical name explicit. And now, the analysis of any combination of names, references, and meanings is possible. With this new understanding in place, Burgin has undertaken a decades-long investigation of mathematical structures to study their patterns for new levels of abstraction.

In advanced algebraic systems, for example, Burgin notes a prevalence of mathematical structures with chaining patterns [p. 445-412, 5]. In this study of physical symbols, named set theory predicts chaining patterns in the relational database management systems. The B-Z chain presented at the end of the next section confirms his conjecture.

The Aleph Data Relation

This mathematical investigation of physical symbols begins with the discovery of Aleph data relation. It is based on implementing Burgin's *algorithmic* named set [p. 42, 5]. This theoretical structure consists of an algorithm, input and output sets. In notation, $\mathbf{A} = (\mathbf{X}, \mathbf{A}, \mathbf{Y})$ where \mathbf{A} is an algorithm, \mathbf{X} is an input set, and \mathbf{Y} is an output set. In the database, the implementation of the algorithm named set allows the author to study the mapping between input and out attributes at the data level. He calls this mathematical structure, listed below, a binary attribute relation or BAR,

$$(A_{input}, rules, B_{output}),$$

where A_{input} and B_{output} are attributes in table R , and the *rules* are a wff SQL SELECT.

The author calls this SQL SELECT a **BAR query**. Its details, listed below, explicate how data condition v , *known* input data, self-references A_{input} 's domain to fetch *unknown* output from B_{output} . The algebra in this expression allows output data t to include one or more values, regardless of the database application. The Aleph takes shape by the data mapping between v and t . It is mechanically constructed parent/child data relation.

$$(\text{SELECT } B_{output} \text{ FROM } R \text{ WHERE } A_{input} = v) \rightarrow t$$

$$\text{where } v \ni A_{input} ; t \subset B_{output} ; \text{ and } ((A_{input} \subset R) \wedge (B_{output} \subset R))$$

The overall simplicity of BAR query creates our first opportunity to view the deeper structure of physical symbols. Its straightforward layout highlights a division of labor between input and output. By viewing these two channels as containers, we can create a conceptual metaphor that allows us to telescope down to the hardware and bounce back up with output data. For instance, with $input = v$, data signal v flows down to logic gates in the CPU to conduct pattern matching on A_{input} components in the current dataset. When v is equivalent to a component value, the database selects the record for the new dataset. On output, data t rises upward from the newly selected dataset in line with address locations assigned to B_{output} 's record components. Thus, the computer relies on both the value of each data symbol as well as on its attribute container.

In the next Burgin model, the B-Z chain aggregates the Aleph to generate menu data for a decision tree. The recursive algorithm that generates this tree structure can be found in []. In the database, the B-Z chain models a data network. Each link in the chain depicts a BAR that models the Aleph. The B-Z chain aggregates the each Aleph data relation by having links that overlap or self-reference the next link. At the data level, output from one link represents input for the next link in the chain,

(A, A) (A, B) (B, C).

In a complex chain, the data flow includes keyed attributes in adjacent links that connect one table to another. At the data level, it reveals a second Aleph hidden between two related tables.

The Aleph's Upward Ascent

The Aleph data relation exists naturally throughout the database. In the decision tree interface, it is a design pattern that ascends from the database table upward to data topics displayed in its GUI. To investigate this upward ascent, the author created a conceptual model in Table 1 below to identify each transition point.





Burgin's Models	Digital Constructions	Interface Perspective	Aleph's Graphic Representation	Physical Symbols
(COLOR, COLOR) (COLOR, SIZE) (SIZE, ID)	Database Taxonomy Interface	Exterior HCI view		{white (medium, large)}
	k2h	Interior HCI view		{white (medium, large)}
(COLOR, COLOR) (COLOR, SIZE) (SIZE, ID)	OHDS	menu data file		{white (medium, large)}
SELECT COLOR FROM WIDGETS WHERE COLOR = 'white'	SQL	Exterior SELECT interface view	white → medium, large	{white (medium, large)}
		Interior SELECT interface view: inside Widgets table		{white (medium, large)}

Table 1

In Table 1 above, the ascent of relational data from the table to the decision tree passes through four states:

1. Data in the database table, the bottom row.
2. The construction of the Aleph data relation, one row up from the bottom.
3. Nested Aleph data objects in menu data files, the third row of Table 1.
4. Aleph files displayed in the decision tree GUI, the top two rows.

In these four states, two interfaces are presented: the decision tree HCI and the SQL SELECT. In keeping with Simon's study of artifacts, the table depicts the exterior and interior views of each interface. The transition from one view to other is particularly noteworthy in the SELECT because here Simon's and Burgin's theoretical approaches overlap. The SELECT's exterior is a **black box** process. In Burgin's theory, it is an algorithmic named set consisting of input and output. And in both theories, the SELECT's interior view is a **white box** process. In column four, the *Aleph's Graphic Representation* shows how the SELECT on physical symbols clearly operates in a set-theoretic manner. It demonstrates the enduring power of Codd's set metaphor.

To investigate the role of physical symbols in these transitions the analysis in Table 1 goes bottom-up, left-to-right, starting with *Burgin's Models* in column one. All of *Burgin's models* implicitly represent the Aleph in terms of physical symbols. His models serve as blueprints for the *Digital Constructions* in column two that transport the Aleph data

object in digital space. Before the Aleph, the database table contains only application data. With Burgin's SELECT statement model, the Aleph emerges as a new figure rising from background data. In the next table row, the B-Z chain predicts the open hierarchical data structure (OHDS), and both outline the format for storing Aleph objects in files. When these menu data files are displayed in the decision tree, the B-Z chain is general enough to predict their new structure in the k2h hyper-graph. And finally, its general nature even predicts the exterior structure of the Aleph in the decision tree display.

While Burgin's models are general predictors, the *Aleph's Graphic Representation* in column three depicts the Aleph data object in a more detailed fashion. Consequently, each depiction in this column is different because each one reflects its digital setting and/or state. In column five, however, a more profound pattern of information unfolds. When each graphic representation is modeled by its *Physical Symbols* and — according to their hierarchical containment — a uniform models emerges. This model highlights the invariant layout of its physical symbols. The models in this column are held together by a deeper, underlying mathematical bond that transcends time and space.

Physical Symbols and Meta-Symbols

In Table 1, there is a deep mathematical connection that holds all of the Aleph models together. The author believes this relationship is grounded in the logical structure of physical symbols. In column five, the invariant pattern of models makes this point clear. Therefore, the author reasons that all 0's and 1's on the computer possess a deeper logical structure.

However, today's understanding of physical symbols is limited to a single dimension: each bit or string of bits represent real numeric values. In database research, these values are widely recognized as the source for linking tables together [p. 21, 8]. Recent discoveries in physics add support to the physicality of signal/symbols on the computer [9]. But the author believes that this single dimension of digital symbols is not a complete or balanced view. He contends that these symbols populate a more detailed system that is precise and order, like any distinct branch of mathematics.

And so, the author argues that physical symbols on a computer are composed of deeper, sources of information. They include mechanical sources of information that shed light on how the computer system manages its digital symbols in a uniform fashion. He believes that the logical relationship between its signals and symbols is bi-conditional. The system treats both on a casual, mechanical basis. For any possible signal, the system always determines the same symbol. The same holds true when converting from symbols to signals. But the speed of these transformations makes direct observation of this mechanical phenomena virtually impossible. The only we can investigate these details is by applying pure mathematics, such as Burgin's named set theory, to model these systems and to draw inferences from these abstractions. Subsequently, the author relies on the integrity of this mathematical system to establish new levels of abstraction that permit us to investigate these details as sources of new information.

Based on the predictive power of Burgin's system of models, and on the mathematical certainty of mechanical computing, the author holds that all physical symbols on the computer are composed of two **meta-symbols**: 1) physical-values and 2) constructed-types. The author believes that these two aspects of digital symbols enable computer systems to treat its 0's and 1's, in a linguistic fashion, as types and tokens.

The physical-value of each symbol enables the computer to treat all strings of 0's or 1's as a "type." This meta-symbol allows the system to differentiate one form of signal/symbol from another by pattern matching.

In contrast, a symbol's constructed-type allows the computer to treat each token as the content of a container. Some of these containers are constructed by hand, like table attributes, while others emerge at runtime, such as dynamic addressing. Both types of containers scale up and down the computer system in a hierarchical fashion that is unbounded. At the bottom of this virtual system, the hardware surface serves as a container for all containers. Each virtual address establishes a container, one that differentiates one bit from every other bit on the machine. And so, this simple address/content relationship makes each string of 0's and 1's unique. It also underscores the logical power of indirect addressing: a *known* address always implies knowledge of its content, even when its actual value is *unknown*.

At higher levels of the system, a constructed-type could be a table attribute in a database, a database table, or even the database itself. With these nested containers, one can easily see how a database differentiates “NY” the *CITY* from “NY” the *STATE* as two different “NY” tokens based on their attribute membership or constructed-types.

Discussion

Mathematic’s ability to distal new details by abstraction is crucial to the discovery of new information. The paper presents a system of mathematical models based on Mark Burgin’s named set theory. These models penetrate deep into the logical structure of physical symbols on the computer to reveal new information about their properties. These models not only uncover a hidden parent/child data pattern in the database called the Aleph they also show how decision trees emerge from its physical symbols. With this scientific evidence, and with the certainty of mechanical computing, the author proposes a theory about digital symbols on the computer. These symbols consist of two meta-symbols: 1) physical values, and 2) constructed-types. He contends that mechanical computing employs these two properties to differentiate one symbol from another, even when two such symbols are visibly identical. He argues that these meta-symbols enable computing systems to manage their digital content in a linguistic fashion that differentiates types from tokens. He believes this new understanding is a balanced view of physical symbols that unifies Shannon’s understanding of their numeric values with PSSH’s focus on their dynamic structures and addressing.

To further develop a theory of physical symbols, the author considers Marshall McLuhan’s advice to study the media of mechanical computation. The aim here is to undercover the nature of its dimensions. Next, he intends to use mathematical modeling to investigate symbol grounding on these machines. The objective of this research is not only to survey and chart the logical boundaries of physical symbols but to differentiate them from the purely abstract symbols in the mind’s eye. To make this distinction, the author intends to revisit the philosophic issues raised by Ernst Cassirer and his student Susan Langer regarding the difference between signals and symbols. He also plans to investigate how symbolic content emerges from an artifact compared to its emergence in natural phenomena as explained by [10].

References

- [1] Shannon, Claude E. A Mathematical Theory of Communication, Bell System Technical Journal, Vol. 27, pp. 379–423. 1948.
- [2] Newell, Alan and Herbert A. Simon. Computer science as empirical inquiry: symbols and search. Communications of the ACM: Volume 19 Issue 3, Sept. 1995, pp.113-126.
- [3] Nilsson, Nils J. The Physical Symbol System Hypothesis: Status and Prospects. in 50 Years of Artificial Intelligence, Essays Dedicated to the 50th Anniversary of Artificial Intelligence Eds: Lungarella, M., Iida, F., Bongard, J., Pfeifer, R. Berlin, Germany: Springer-Verlag, 2007. pp. 9-17.
- [4] Zellweger, H. Paul. A Knowledge Visualization of Database Content Created By A Database Taxonomy. The 15th International Conference on Information Visualization. (IV2011). London, UK. (July 2011), 323-328.
- [5] Burgin, Mark. Theory of Named Sets. Nova Science Publishers (January 1, 2011).
- [6] Zellweger, Paul. Tree Visualizations in Structured Data Recursively Defined by the Aleph Data Relation. In the 20th International Conference Information Visualization (IV’16). Lisbon, Portugal July 2016, pp. 21-26.
- [7] Simon, Herbert A. The Sciences of the Artificial. Cambridge, Massachusetts: MIT Press, Third Edition, 1998.
- [8] Atzeni, Paolo, Stefano Ceri, Stefano Paraboschi, and Riccardo Torlone. Database Systems, Concepts, Languages and Architectures. Maidenhead, Berkshire, England: McGraw-Hill Publishing Company. 2000.
- [9] Be’rut, Antonone, Artak Arakelyan, Artyom Petrosyan, Sergio Ciliberto, Raoul Dillenschneider & Eric Lutz. Experimental verification of Landauer’s principle linking information and thermodynamics. Nature, 8 March 2012, Vol. 483. pp. 187-190.
- [10] Touretsky, Davis and Dean Pomerleau. Reconstructing Physical Symbol Systems. Cognitive Science, 18, Vol. 2. 1994. pp. 345-353.