YOUR BRAIN IS COGNITIVE, NOT A DATABASE
A DISTRIBUTED ASSOCIATIVE MEMORY BASE FOR NATURAL INTELLIGENCE

Intel® Saffron™ Cognitive Solutions are the most universal, unified, and commercially mature approach to brain-like thinking. Its platform closely mimics the human brain’s cognitive skills for computing, learning and reasoning in real time — but at a scale that far exceeds the capabilities of human brains. Representing a confluence of non-traditional approaches, Saffron Cognitive Solutions are a key:value, incremental learning, fast-query, graph-oriented, matrix implemented, semantic, and statistical knowledge store inspired by the associative structure and function of real neural systems. Saffron Cognitive Solutions automatically identify connections and their context to find trends in big data, making it possible to not only understand what is happening, but why it is happening and how it can be acted upon. This technical paper describes Saffron Cognitive Solutions' methods of matrix partitioning and distribution, attribute sorting and grouping for co-locality, and hyper-sparse matrix compression as well as streambased aggregation for real-time analytics.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th></th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Executive Summary</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>The Big Data Revolution</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>Best And Brightest Trends</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>Making The Difference For Decision-Making</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>Matrix Representation</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>Quintuple Name Space</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>Attribute Grouping</td>
<td>12</td>
</tr>
<tr>
<td>8</td>
<td>Large Matrix Distribution</td>
<td>14</td>
</tr>
<tr>
<td>9</td>
<td>Hyper-Sparse Compression</td>
<td>16</td>
</tr>
<tr>
<td>10</td>
<td>Observing And Imagining</td>
<td>18</td>
</tr>
<tr>
<td>11</td>
<td>Customer Results</td>
<td>20</td>
</tr>
<tr>
<td>12</td>
<td>Conclusion</td>
<td>22</td>
</tr>
</tbody>
</table>
Complex data analytics is quickly becoming a source of enormous competitive advantage for modern enterprises. To rapidly move and stay ahead of competitors, businesses must better unify, understand, learn from, and act upon information — at big data scale. To do so, competitive organizations will move beyond traditional, rigid databases and latency batch processing, and towards new technologies that automatically identify and count relationships in data. Such technologies include key:value data stores and knowledge-based graphs which enable an operational environment that can leverage all the data available to make real time, informed decisions. These trends include matrix implementations of graphs and their unification of semantic and statistical methods that underpin the emerging industry of brain-like reasoning by computers.

The Intel® Saffron™ Cognitive Solutions are based on human-like connectivity, association, and reasoning. Saffron Cognitive Solutions’ “network of associative memory networks” represents all these innovative trends in an effective and efficient implementation. This paper will provide an overview of the key elements that make these networks powerfully different, including Intel® Saffron Cognitive Solutions’ methods of matrix partitioning and distribution, attribute sorting and grouping for co-locality, and hyper-sparse matrix compression as well as streambased aggregation for large-scale real-time analytics.

CTOs, CIOs, innovators, futurists, and other curious persons familiar with NOSQL systems will appreciate the design behind Saffron Cognitive Solutions’ cognitive computing platform in this transparent detail.

This paper describes how Saffron Cognitive Solutions identify context and trends in big data, making it possible to not only understand what is happening, but why it is happening and how it can be acted upon. The offered use cases further demonstrate and prove how the implementation of an associative memory answers new business’ demand for making sense of relentless, ever-increasing volumes of data. Finally, this paper details and shows how cognitive systems can better inform decisions by thinking more like the way humans do.

**THE BIG DATA REVOLUTION**

Enterprises are challenged by the notion of a big data revolution and the ability to exploit it for business value. However, this revolution, contrary to popular belief, is not about data volume. According to Gartner, many IT leaders still fall short of solving big data challenges because they focus on managing the high volumes of information at the exclusion of other dimensions such as variety and velocity[^1]. Increasing storage capabilities to multi-petabyte data stores is pointless and irrelevant if the stores are not able to exploit knowledge in the data with ad-hoc tools efficiently. Forrester further elaborates on data dimensions to include cost, complexity, and scalability as other elements any practical solution must also address.

Traditional analytics models require batch-oriented construction that becomes “stale”
without reconstruction, and batch-oriented queries, which take more compute time when time is the most critical aspect. Given the velocity of data and the competitive business advantage of fast analysis and response, enterprises must find ways to exploit data ever faster, toward the goal of real-time learning and inference. Businesses must also be data-agnostic as "data" is not limited to structured data.

Saffron Cognitive Solutions are designed to efficiently support the variety, velocity, and volatility challenges presented by today's enterprise data sources. The associative approach for unifying diverse data and quickly responding to fast changing requirements, allows Saffron Cognitive Solutions to support hybrid data by fusing structured data with unstructured content. Saffron Cognitive Solutions further unifies analytics itself across descriptive, diagnostic, predictive, and prescriptive applications. Designed for scalability, Saffron Cognitive Solutions can easily meet the demands of increasing data without increased costs or complexity.

Saffron Cognitive Solutions' implementations have culminated in its most recent products and have been described, in part, in the recently issued patent “Methods, Systems And Computer Program Products For Providing A Distributed Associative Memory Base” (US8352488). Saffron Cognitive Solutions’ patented technology development led to the commercial release of Intel® SaffronMemoryBase™.

This paper describes elements of Saffron Cognitive Solutions' technology to prove how a distributed "memory base" unifies data and identifies patterns and future outcomes, and does so with real-time speed at low cost and complexity.

BEST AND BRIGHTEST TRENDS

The best and brightest experts continue to debate how to derive value from the varied dimensions of big data analytics, revealing an important time of transformation. Many approaches focus on extending traditional architectures while others advocate new ideas, new methods. Saffron Cognitive Solutions' methods, systems and products are leading the market forward in new ways to capture value from any and all data. Below is a review of traditional and newer methods to help inform readers and to begin to compare/contrast to Saffron Cognitive Solutions’ approach to applying cognitive methods to big data analytics.

Relational or Key:value Stores?

Relational Database Management Systems were designed for transactional integrity for enterprise financial and supply chain systems. They were not designed to address the unification of unstructured and structured data or to scale to the levels demonstrated by Google's Bigtable and the plethora of distributed key:value stores such as Hadoop. The key:value approach to distributed parallel computing includes map-reduced design pattern, distributed hash coding, and other methods of the Not Only SQL (NOSQL) movement, promising much greater data scale, lower cost, and greater analytic value than traditional data storage.
Row or Column Orientation?

Row orientation found in transactional databases has been the tradition of data storage. However, providers of column oriented stores argue the many limitations of row-based access. For example, column orientation is efficient for dashboard analytics, usually accessing one or few columns — such as when showing sales distributions in a pie chart — without needing to access entire rows. Usually, each column contains only one value type, which also allows type-specific forms of compression. On the other hand, the industry recognizes that column stores are weak when many or all columns need assessing for more advanced analytics. For example, vector-pattern computations over many field types will touch many columns.

Batch or Real-time Speed?

Reasoning can be performed either when the data is loaded into the knowledge base or when a query is issued. The former class of knowledge bases, which perform reasoning when data is loaded, are called materialized knowledge stores. “Compute power over raw data” is a common-approach analytics applied to raw data stores, most notably applying MapReduce computations over a key:value store. In-memory and hardware-based acceleration is becoming common.

Beyond document indexes and hyperlinks, graph orientation of the Semantic Web envisions the linking of “things” and ideas within documents. If more intelligent computing is to operate on these things and their connections, then the natural storage representation of things and connections is through graphs. From a mathematical perspective, the node-link-node elements of a graph are universally expressive for many forms of knowledge, such as the subject-verb-object structure of all human language when addressing unstructured text, and are applicable to the links between persons, places, things, situations, actions, and outcomes within both structured and unstructured data.

Data or Graph Representation?

The NOSQL movement is largely composed of approaches to data storage with the analytic method “on top.” However, several initiatives aligned with the Semantic Web movement specifically focus on the storage of graphs. Graph stores are an example of materialized knowledge.

Graph or Matrix Representation?

Graphs are natural forms of knowledge, but should graphs be implemented in the form of graphs? In other words, should graph-links be implemented as pointers in random access
memory and “chase” pointers from one node to another node? A chorus of researchers argue that matrices are formally equivalent to graphs, but provide the better implementation. Graph Algorithms in the Language of Linear Algebra\(^5\) describes how matrices better address syntactic complexity, ease-of-implementation, and performance. The “spaghetti” nature of graph-oriented “pointer chasing” has hampered Artificial Intelligence (AI) throughout its history. In contrast, matrix organization tames complexity and offers a number of physical properties for efficient and scalable implementation.

**Semantic or Statistical Reasoning?**

Advanced analytic methods tend to split AI-type predicate semantics and statistical numerical analysis as two separate worlds. For example, the standard query language of graph stores (SPARQL) does not admit statistics and the standard of predictive modeling (PMML) does not admit logic. The types of data they address also often separate these forms into two different and fractured solutions. For instance, machine learning of numerical values in structured data can be radically different from learning about discrete relationships in unstructured (and structured) data. In addition to taming complexity, easing implementation, and speeding performance, matrix representation unifies semantics and statistics. Matrices are equivalent to graphs in order to address semantics and they have a rich history in numerical methods to address statistics.

**Computer or Brain Science?**

Any software implementation should adopt methods of distribution, hashing, and compression which are matters of computer science. However, neural structures, real-world intelligence, and real-time learning — rather than data structures — should inspire building machine intelligence. Traditional AI and machine learning, following the logical models of computer science, bear high costs, high latencies, and high levels of inaccuracy. Saffron Cognitive Solutions reflect the fundamental shift to more brain-like thinking in associative memory-based representation and reasoning, but many recent vendors also claim “brainlike” and cognitive approaches. The focus of these endeavors range from lower-level perceptual processing (images and video for example) to conceptual mapping (topic mapping for example), all of which should include the most important element of brain-like thinking: the ability to adapt and respond in real time.

**Saffron Cognitive Solutions’ Answer? Natural Intelligence**

Saffron Cognitive Solutions remain the most universal, unified, and commercially-mature approach to brain-like thinking. Representing a confluence of the non-traditional approaches described above from NOSQL to brainlike reasoning, Intel® Saffron Memory Base is a key:value, incremental learning, fast-query, graph-oriented, matrix-implemented, semantic, and statistical knowledge store inspired by the associative structure and function of real neural systems.

This unique combination of approaches represents the best way forward to real intelligence. Given realtime and real-world requirements, whether for animal or business survival, the laws of physics constrain the implementation of knowledge engines. Whether these laws address or violate physics will directly impact speed, effectiveness, efficiency, and accuracy. The brain has already discovered these laws and, with this inspiration, the group calls its own implementation the Saffron Cognitive Solutions.

The Saffron Cognitive Solutions group believes that a natural intelligence approach is fundamental to the future of computing and will impact every industry’s ability to use the past to understand the present in order to anticipate and prepare for the future.
In order to effectively assist efforts to implement a natural intelligence approach to computing, computers should be designed as intelligent machines with the human brain abilities for sense-making and decisionmaking in mind. Sense-making is often described as “connecting the dots” between specific pieces of information. When analyzing a situation, the brain remembers the connections and similarities between people, places, and things. “Entity Analytics” answers both of these fundamental questions: Who (or what) connects to whom? Who (or what) is similar to whom? For decision-making applications, a new situation will trigger a memory of similar situations and their prior connections to actions and outcomes. In addition to recalling similar people, places, and things, brains reason by similarity to make predictions. The Saffron Cognitive Solutions group bases its approach to “Anticipatory Analytics” on such recall of similar experience. Given acquired experience, is the situation good or bad? What did we do in past situations? How did it turn out? Do we do it again? What do we do differently?

Machine memories should be more brain-like to “think” the way humans do. In the same way that relational database management system (RDBMS) tables support SQL queries, new kinds of queries require new structures. Semantic queries require a graph store. Statistical queries require a frequency store. Moreover, semantic queries can use statistics and vice versa. For example, when asking about relationships, the strength or the “oomph” of the link is also informative beyond the connection's existence. As described above, there is a growing need for a matrix store that supports both.
The Saffron Cognitive Solutions group believes that knowledge is best represented in matrix form. In Intel® Saffron Memory Base matrices were designed to represent the connections, context, and counts of every person, place, situation, action, event, or outcome. The real neuroscience of brains is much more complicated, but thinking each neuron as an associative matrix provides a simple yet powerful principle of organization. The brain is a massive composition of such neurons, built from simple to complex, based on the elements of synaptic connections and the strengths – the synaptic weights – of these connections.

Connections
A matrix is equivalent to a graph of connections. Given the index of a row and index of a column, an “adjacency matrix” of 1s and 0s in the matrix often defines whether or not the row and column indices are connected, which is equivalent to a graph. Connectivity is one meaning of the word “associative”: whether two things do link to each other. A single matrix represents a network that is highly organized in a regular form for writing and reading connections.

Counts
Rather than a binary matrix of 1’s and 0’s, the cells at the intersections of rows and columns in a matrix can represent weights resulting from an increasing number of occurrences of the observed connections. Beyond 1’s and 0’s, such a count within a matrix represents the joint frequency distributions between things (their rows and columns). These counts support the statistical meaning of an “association”: i.e., the degree of dependence between two things. Partial correlation or, more generally speaking, the three-way interaction entropy, captures information between things.

Context
If one matrix can represent a network (a graph), then a “network of associative networks” can represent semantic context as there is a “triple” association in every matrix-row-column. Each matrix represents the conditional perspective of a person, place, event, action, outcome, or whatever. A collection of 2-D matrices represents a 3-D cube, forming a hypermatrix or hypergraph. One underlying network of networks representation unifies context semantics, as in graph databases (node-link-node as three elements), as well as higher-order entropies (interaction information).

Combinations
Many regions comprise the brain for different functions, including semantic memory, episodic memory, procedural memory, and motor memory. These different “memory design patterns” define different graph configurations which come together into larger applications. Called “named graphs” or 4-level “quad stores” in the semantic graph world, the Saffron Cognitive Solutions group describes such larger systems as “networks of networks of associative memory networks” including both semantics and statistics as the core elements of all memory designs.

Networks of matrices represent a semantic graph. The storage and recall of frequency distributions also supports statistical methods. Queries are real-time due to a quick lookup of matrices, rows, and columns rather than table scanning to compute the same.
Conditional labeling is applied to each matrix, with at least one matrix for learning about each person, place, or thing. Each matrix represents the associated pairs of other things. When asked a simple query of any two things, such as John Smith and London, one matrix and one row contains all the “triples”, both semantically and statistically. Asking for specific columns, such as aviation carriers, returns specific column labels and their associative frequencies contained in the row-column cells. A set of matrices represents an entire semantic graph — a network of associative matrices. Many graphs can be composed into larger combinations as networks of networks.
Intel® Saffron Memory Base defines a distributed key:value store, where the frequency count is the value under a hierarchical semantic key. Beyond a “quad store”, Saffron Cognitive Solutions define a five-level “quint store” according to the following hierarchy:

**Space**
Each top-level partition, or memory space, contains a large set of memories. Spaces partition different security levels of knowledge access in the same way that “named graphs” separate/combine the provenance of data sources. Spaces can also partition different data types and data sources, such as one space for learning about data and another for learning about users' behavior with data for end-user personalization and expertise lookup. Spaces can also be applied to address different use cases as defined by functional groups and lines of businesses.

**Memory**
A memory is a conceptual entity, representing a person, an event, or any “thing” to be learned and remembered. Each instance of an observed entity constitutes a memory. Each memory grows as related data arrives, unifying its knowledge across both structured and unstructured sources.

**Matrix**
Memories are conceptual objects while matrices are their physical underpinning. Matrices provide an additional dimension within each conceptual object. For example, the memory of an online consumer can contain one matrix for what is “liked” and another for what is “not liked” along with the context of social influences. Matrices can also manage time dimensions as “time slices” within each memory, whether used to define temporal patterns or used for retention policies to delete older slices or move them to tertiary or cold storage.

**Row**
Matrix-orientation includes both row and column indexing. Rows and columns can be identical in a triangular associative matrix but, in physical implementation, Saffron Cognitive Solutions are row-dominant. Given a matrix name, its rows are the next element of lookup to an input query. Each row represents a distribution of column connections and cell counts, conditional to the memory matrix.

**Column**
Rows are dominant for input, while columns and their counts are the output. With at least one matrix as the physical implementation to every memory, a memory name and row name provide the pair of semantic elements needed to look up the associated columns as a semantic “triple.” Saffron Cognitive Solutions also return the strength of each triple. Given the entire space-memory-matrix-row-column key, the count of the matrix cell is its value. In other words, the entire name-space hierarchy provides a key to access a matrix cell.

As next described, category:value tuples for naming the memories, rows, and columns provide even greater resolution of the hierarchy.
Categories can be hierarchical themselves, such as to represent a taxonomic tree by dot notation of the category name. These five levels represent the physical structure of Intel® Saffron Memory Base, but the logical structure can be even richer. Below is a description of how the cell value can be any byte array, but the use of frequency is key to the definition of associative memory networks, which unify both semantics and statistics as a universal knowledge store for advanced analytics.

Figure 2. Quintuple name space for an associative memory base

The Saffron Solutions Group defines a memory base as a hierarchical name space. Each space contains many memories. Each memory is implemented by at least one matrix, defined by its rows and columns, which are indices to cells of count (or other statistical) values. Typical queries provide input keys for the return of column semantics and cell statistics. However, any subset of keys can query the next level of the hierarchy, such as listing spaces, listing memories in a space, listing matrices in a memory, etc.
ATTRIBUTE GROUPING

The name space includes additional organizational structures for data tuples, called name:value, attribute:value, or field:value pairs. Whether from field:values in structured data or category:values from unstructured data, the entity, topic, or sentiment markup tuples are naturally found in data as the kinds of things to learn about. Category:values define labels of memories such as for Person: John Smith. Organized together by their categories, the same is true for row and column labels.

Tuples are elemental to the “language” of data; all values are logically grouped within a category together. For example, when asking the semantic question, “What are John Smith's flight preferences when traveling to London?” or “Who did John Smith meet in London?”, everything about John Smith and London is organized within the memory for Person: John Smith and the row for City: London (as well as in the memory of City: London and row Person: John Smith). Queries tend to be more specific, such as when asking only for persons or only for flights, as in these examples. Category grouping provides another level of organization for fast performance by co-local storage and retrieval of all the relevant answers.

All values are logically grouped by mapping each category:value to a 64-bit index. The high bits encode each category while the low bits encode each value. In other words, each value is indexed “within” the range of the category.

With 64 bits, a 10-bit allocation for category names allows over 1000 kinds of things and leaves 254 possible values per category. The use of 10 locator bits (and other reserved bits) still leaves billions of possible values per category. The balance of category, locator, and value bits is also configurable, of course. Logical grouping places all values of a category “together” in bit space, and physical co-locality leverages this encoding, as described below for compression. A number of other benefits derive from such bit encoding:

Atom Indexing

Neurons follow a “line code” representation: They do not know the specific semantic labels of their inputs and outputs. In similar fashion, the core of Saffron Cognitive Solutions act as an index to each memory, matrix, row, and column but does not include its external string labels within its store. Saffron Cognitive Solutions organize and stores data by internal line IDs. Especially important for long string values, external data values are stored only once in an atom table, which converts external values to internal IDs and vice versa. Removing the repeated storage of string values is one form of basic compression.

Global Sorting

When all memories, all rows, and all columns use the same atoms, everything across the distributed system is ordered the same way. As each memory returns its local responses, these responses can merge with those from other memories. In other words, semantic “joins” between memories require only a merge join, not a sort-merge join. For even greater performance, global ordering allows the streaming of merged result sets. Merging by integer comparisons, rather than long string comparisons, further improves performance.
**Schema Changing**

A 264-bit space can store a practically unlimited number of categories and typical designs start with fewer than 100 categories. Therefore, allocating only 10 bits for 1000 category types leaves plenty of room for the dynamic addition of new category types and specializations. Unlike many key:value stores, the addition of new category groups can occur on-the-fly without reconfiguration of the name space and physical re-distribution.

In summary, memories, rows, and columns represent category:values but are stored and addressed internally by 64-bit IDs. This internal encoding supports physical co-locality, efficient representation, and global sorting. The dimensions of the memory base can grow dynamically.

**Figure 3. Group-by category:value encoding of atoms**

Memories, rows, and columns include the additional structure of category:values in the name-space hierarchy. Whether field:values in structured data or taxonomic category:values in unstructured text, data values are grouped by category type. Rather than storing strings in the physical matrix store, an atom table more efficiently stores strings once, translating strings to atoms that encode the grouping of values in low bits “under” the category in high bits. The atom also reserves bits to store the distributed locations of “things.” 64-bit atoms are used in 64-bit machines. 8-bit examples are shown here for simplicity.
LARGE MATRIX DISTRIBUTION

Placing one matrix on one machine and another matrix on another is an obvious but naïve approach to hypermatrix distribution over the nodes of a machine cluster. This approach might be adequate for simple storage to support offline access to any matrices, such as for offline data mining. However, for real-time machine learning and real-time query, some matrices can become exceptionally large, creating performance bottlenecks, while low-demand matrices can starve for lack of work.

In real neurons, it seems that different data densities and performance requirements create different neuron shapes. Some neurons exhibit more fan-in than others, while some are more nonlinear. Similarly, one matrix representation is not optimal for all cases. Different data loads create matrices of different sizes, which Saffron Cognitive Solutions implement in different ways.

Small Matrices
Matrix sizes tend to follow a Poisson distribution in that most matrices are relatively small. Small matrices are typically stored locally on one machine. Unless expressly declared a priori, all matrices start as small matrices in their first observations of a few data records or text sentences. The location of each small matrix is “pinned” to one machine by a distributed hash coding (DHC) of the memory-matrix name. For example, modulo 10 of the name’s hash code over a 10-machine cluster defines which machine stores and services a particular matrix.

Large Matrices
Large matrices with millions of rows and columns are fewer in number but are common given high data volume. If co-localized to one machine, large matrices create bottlenecks when frequently-mentioned memory labels occur during data ingestion or when frequently needed during user queries. Distributing large matrices is a must. As a large matrix by definition and pre-declaration, Saffron Cognitive Solutions include a global Directory Memory to store pairwise associations across everything — every atom-atom pair within each memory space. In such cases, Intel® Saffron Memory Base partitions large matrices by row. The DHC of the memory-matrix — as well as a B+-tree — defines the machine location of each partition to locate a specific row. Following dominance of rows for inputs (columns for output), the physical distribution of large matrices is also rowdominant.

Large Rows
At more extreme data scales, some matrix-rows become exceptionally large and need further partitioning and distribution. This is sometimes called the “large row” problem and is a limit to many databases and many key:value stores. It is not a problem for Saffron Cognitive Solutions’ scaling and performance of extreme large associative memories.

Of course, the size of matrices and rows changes over time. Saffron has the ability to detect the crossover point and dynamically promote a small matrix to a large matrix, or to partition a large row. Some “small” matrices can remain small in some row dimensions while growing large in others, which should also be stored as large rows. No matter the case, Saffron Cognitive Solutions apply the most appropriate partitioning, compression, and other methods to increase concurrency as needed.
Unless pre-specified to be a large matrix, all matrices begin life as small matrices, compressed by standard sparse encoding. As matrices learn and grow beyond a performance crossover size, Saffron Cognitive Solutions partition the row and switches to a hyper-sparse encoding method. Distributed across machines for load balance, if a row grows beyond a specific size, further partitioning and distribution occurs. Codices for the various location and compression methods are automatic and dynamic.
Different matrix sizes suggest different methods of compression. Very small matrices with few vector observations tend to be dense but, even after moderate loading of more records and/or sentences, the row and column dimensionality tends to grow much faster than the non-zero cell density. Standard compression methods such as Compressed Row Storage (CRS) or Compressed Column Storage (CCS) avoid the storage of zero-value cells. CRS, for example, lists non-zero rows, each pointing to nonzero columns, each pointing to the cell value for the row-column intersection.

However, with a name space of up to a quadrillion possible dimensions, large matrices and large rows can become hyper-sparse. Even when there are near-astronomical distances between one non-zero cell and another, matrices contain a massive amount of information across all the cells that do exist. Using 64 bits for a row index, 64 bits for a column index, and storing the value can become expensive even for CRS. Furthermore, for matrices to represent a semantic “triple” store, each triple must be inserted at least three times (three taken two at a time) to provide every entry point to efficiently query triples. The CRS indexing cost must be multiplied by three.

Saffron Cognitive Solutions insert every triple 6 times: 3 times for each memory-matrix perspective as well as 2 times for each memory matrix and its transpose. The storage of both the associative matrix and its transpose ensures that every row contains all its associations without needing to scan other rows. Said another way, the row for A connects to all its columns, such as to any column B. The row for B also connects to all its columns, such as to column A. Storing each triple six times is optimal for fast queries, but can become expensive without further compression.

Hyper-sparse matrix compression does more than avoid the storage of zero cells. It combines two methods, one for atom IDs and one for ID counts.

**Relative Index Encoding**

Because Saffron Cognitive Solutions transform all categories and values to 64-bit integer IDs and manages them in a globally ascending sort order, each ID down the row is higher than the one before it. Rather than store an explicit ID, the offset of one ID to another is a smaller number, more efficiently stored in fewer bytes. Because the IDs are in category:value bit space, this relative encoding is equivalent to a zero-run length encoding. The distance between one non-zero cell and other nonzero cell represents the number of intermediate zero cells. To efficiently store the astronomically large range of ID distances, from a distance of 1 bit to 2^64 for example, a variable length byte array uses as few bytes as required or as many bytes as needed.

**Variable-size Counter Encoding**

The counters for connection strengths can range from one to an unlimited size over a system’s lifetime of observations. A fixed counter size would either underutilize a small counter or would overflow at some size limit. Therefore, the counter also encodes a variable length byte array, often called a BigNum encoding. One byte or even one “nibble” suffices for small counts, but no matter the counter size that might occur over long hauls, this representation dynamically grows to cover it.
These two encodings, one that defines the column ID and one that defines its connection strength, combine to form one linear array within each row. Both encodings are of variable length so Saffron Cognitive Solutions reserve one bit in each byte as a continuation bit; does the current byte conclude the index or counter or are more bytes required? The two encodings switch between a byte run to define the column ID and another byte run to define its count. Therefore, another bit defines whether the current byte encoding is an index code or counter code.

The higher order bits for the category type halt the byte scan when values for one category roll into the values for another. The relative encoding defines the distance from one cell to another. As long as the added distances from one ID to another remain within the range of the requested category, such as to lookup all associated values to the category person, the decoding of IDs and their counts continue from one ID to the next. However, when the addition of the next distance causes the category bits to increment the ID into the next category range, all the answers for the requested category are read already and the decoder is terminated.

Again, the use of various encoders/decoders optimizes the size and density of each matrix, but this particular method addresses hyper-sparse matrices. This compression is a writable compression: Saffron Cognitive Solutions insert new connections at will into the byte array. Counters for given connections grow only logarithmically, but a new counter byte can be inserted whenever needed.

**Figure 5. Combined relative column ID and variable-size integer encoding**

Hyper-sparse matrix-rows encode a byte array so that it contains both semantic and statistical information. Zero run length defines the number of empty cells between column labels with non-zero cell values — the relative encoding distance between atom IDs. Counter frequency follows each column ID in the row. One bit determines whether the following bits represent a zero run length or a frequency counter. The distances between column IDs and the frequencies of observation can be greater than 256 limits of one byte. Therefore, a continue-bit allows variable byte length encoding of large distances and big counters.
Saffron Cognitive Solutions increment its cell counts automatically. An intrinsic, server-side operator observes new vector patterns and increments the respective association encounters. Dynamic growth of connections and counts is fundamental to a memory base form of machine learning. This form of learning is nonparametric and non-functional in that a memory does not “fit” data to a specific mathematical model. A memory learns and remembers everything as it observes and recollects anything it knows when asked. Rather than joining and scanning over raw data, a memory base has all the semantic connections and all the statistical frequencies at its fingertips, which is how a memory can rapidly respond.

Adaptive compression supports real-time learning and real-time query. While much of the focus of this white paper is on the structure of a memory base, incremental machine learning is another important property of an associative memory system. This memory base structure operates on the fly, whether for data in or knowledge out. Saffron “observes” and “imagines” in real-time.

**Observing**

Observing data is the process of learning. Whether it is a transaction vector from structured data or an unstructured sentence and its markup, Saffron Cognitive Solutions route the vector to the memories for each element of the vector. For example, a sentence may have two persons, one place, one vehicle, and a sentiment mark. Each “thing” in this sentence represents a memory, and the data vector of the sentence can be routed to each memory, enabling each item to learn independently of the others. Each element in the vector may form a new memory, add a new row/column, insert a new cell or increment a counter. All these growth operations are dynamic.

**Imagining**

When presented a query, imagining is Saffron Cognitive Solutions’ process of recollecting what is learned. Querying tends to be in real time, in the sense of synchronous query-and-return as opposed to asynchronous or “batch” computation. As with observing, different memories represent different elements in the query, each with its own perspective knowledge of the answer, depending on the context of the query vector.

Saffron Cognitive Solutions route the query to the memories for each element of the query. Each memory applies the other query terms as context to its rows, returning its answers on the columns. The more complex queries, such as for the nearest neighbor lookup or for predictive classification, rely on the same kind of lookup. When looking up similar persons to one person, for example, Saffron Cognitive Solutions recall associated features and relationships to find other persons with such features and relationships. Shannon entropy or Kolmogorov complexity\(^6\) describes how much information weight is given to each feature or relationship link. Fundamental to all queries, the recollection of stored semantics and statistics is the basis for the answers.

Imagining is a process of streaming aggregation. Saffron Cognitive Solutions analyze large and complex vector-oriented queries and establishes where the relevant memories, matrices, and rows are located. Saffron Cognitive Solutions then send requests to various machines, also establishing answer queues. Each answer queue connects to one or more rows across a cluster and aggregates the perspective answers from the rows into one ultimate return. These queues are streaming in the sense that each machine immediately returns
its co-local answer sets from storage. Due to the global sort order of IDs, the queues release the earliest IDs it receives as it continues to process remaining IDs.

This paper describes most of the physical machinery at the core of Saffron Cognitive Solutions. The memory-based approach is known as “just-in-time” or Lazy Learning\(^7\), based on the principle of minimum commitment in order to maximize flexibility. Rather than trying to fit data to a parametric, functional, “eager” model, a memory simply remembers. In this sense, a memory constantly learns as data arrives and then recollects its connections, counts, and the context to fit the context of the question. The power of memory-based reasoning rests in an efficient physical layer and the ability to dynamically map this physical layer to a logical layer through streaming aggregations, whether the aggregation method is semantic, statistical, or both.

Continual and real-time data assimilation is critical for incremental learning. However, speed of imagination is more critical. When a competitive advantage requires fast answers to changing questions, the time to exploit what is known can mean the difference between life or death, winning or losing.

Figure 6. Streaming queue aggregation from distributed matrix servers

A single queue can attach to more than one row across more than one matrix server. For example, a product taxonomy can have different rows for different product types, which a query can aggregate together. The queue fetches one or more ID:Count streams from across the matrix servers. Saffron Cognitive Solutions sort the IDs globally; in these streams, merging and count accumulation are in ascending ID order. The queue records the “MIN ID”, the minimum ID from across sources, as results are aggregated. In other words, if a source reports a higher ID, the sort order guarantees that it has already “weighed in” on everything of earlier order. Therefore, all sources have passed the MIN ID and the queue can release the merged results to return as it continues to request and merge more results. Such aggregation is useful across taxonomies, time frames, name variants/aliases, and more. Aggregation is also possible over returned column IDs mapped to the same “thing.” This example of counter addition is only one form or aggregation. Any semantic or statistical operator applies to the vectors returned over any number of matrices. Complex queries may engage hundreds of such queues from thousands of vectors in any number of aggregations.
Saffron supports operations across a range of enterprise customers. Actual use cases extend from sense-making to decision-making, using Saffron as the unifying platform, in the same way that memory-based reasoning underlies all human thinking. The things represented in memory are the “dots” of knowledge, connected in many different ways for different purposes.

Connecting the Dots

Saffron Cognitive Solutions describe sense-making as “connecting the dots.” Given a variety of structured and unstructured data sources — inside and outside the organization — Saffron unifies knowledge of data for customers ranging from national security intelligence to competitive market intelligence. The memory reads and remembers everything, connecting the dots so users do not have to. A memory base unifies dozens of data sources in one place and provides knowledge of sources of critical importance that analysts simply do not have time to read. Moreover, Saffron Cognitive Solutions address security by vector-oriented “bit masking” of access, made possible with matrices.

Finding Similar Dots

Connecting the dots is only one side of the Entity Analytic “coin.” Finding similar dots is the other. Saffron Cognitive Solutions solve the alias detection problem in national security and helps with part harmonization in inventory management. Knowing what Saffron Cognitive Solutions have accomplished for alias detection, one creative customer understood how the same solution for finding similar people could be used to find similar parts. In fact, without needing specific match rules or models, a memory can recall similar things in any category. Nearest-neighbor, similarity-based reasoning is fundamental to memory-based reasoning, using both the semantic connections of a dot’s features and relationships as well as statistical measures of information.

Illuminating the Dots that Matter

Classification is the most elemental form of Anticipatory Analytics. Saffron Cognitive Solutions’ memory-based reasoning allows a form of non-parametric, non-functional “modeling” which is used for threat scoring in risk intelligence, among other applications. Matrix-orientation is nonlinear, enabling unparalleled accuracy for these applications. Rather than assuming linear independence, one dot depends on other dots as context, their statistics captured in associative pairs and triples. This type of classification has shown outstanding results: A customer using Saffron Cognitive Solutions for its cyber security personalization product was named “The World’s Best Spam Filter.”[8]
Taking Action on the Dots that Matter

Saffron Cognitive Solutions provide superb decision support. Connecting the dots for Entity Analytics can be extended to connecting the dots between situations, actions, and outcomes. Reasoning is again by similarity, but this time it's related to past experience: Has this been seen before, what was the response to it, and how did it turn out? Previously applied to Intelligence Surveillance and Reconnaissance tasking, Saffron Cognitive Solutions currently support the nuclear power industry to associate and recall the associations between conditions, assignments, actions, and resolutions that are captured in vast documents of human experience.

Anticipating the Dots not (yet) in the Data

Saffron Cognitive Solutions’ product focus is shifting to the most profound use of memory-based reasoning: Predict what is not in the data. Connecting the dots is basic to sensemaking, but recent industry trends are advancing the need for anticipatory sense-making. Seeing past and current trend-lines as simple frequencies is looking in the rear-view mirror, and “head-light” forecasting by projecting the trend-line forward is generally inaccurate.

A memory base defines the patterns from data using statistical dependencies between the dots over time to anticipate what is not yet in data, similar to how the brain uses past experiences to anticipate future outcomes.

Near Linear Scalability of Observation

Scalability has been the historical bugaboo of largescale graph stores. As a graph grows larger, it becomes increasingly difficult to insert new nodes and links. In contrast, Saffron Cognitive Solutions are stable and its hardware throughput remains constant over time. Performance and scaling tests of SaffronMemoryBase® showed no hotspots or starvations on any machines, indicating a balanced distribution. Most importantly, the addition of machines to larger clusters shows near-linear scalability: Each additional machine (64-bit, 8-core, 16G DRAM) added over 20,000 triples/sec throughput during data ingestion. A linear scaling fit of \( R^2 = 0.99 \) indicates near-linear scalability due to the near shared nothing distribution for billions and more matrices.

Near Constant Response Time of Imagination

Given the breadth of semantic and statistical queries, it is impossible to report results for all possible query forms. However, a “simple query” of triples is the standard unit of measurement of graph stores. A simple query with two terms such as a node and link returns all associated other nodes; it is what we use to test Saffron Cognitive Solutions' query time.

It is reasonable to think that query time is a function of data size but, with a store of almost 30 billion triples across 2 billion matrices, the query time of Saffron Cognitive Solutions are largely independent of the result set size. Near independence of data size and the result set size is due to the materialized co-locality of answers in a stream-based architecture. Whether for 1, 10, or 10,000 results, the slope of query time for Saffron Cognitive Solutions are only 0.0002 seconds. In other words, each addition of 1000 query returns adds only 0.2 seconds through REST APIs over a 1Gbit network.
World Record Knowledge Compression

The physical size and hardware scale of very large graph stores is notorious and, therefore, not generally reported. However, the CRS standard is a good basis of comparison. In a 64-bit machine, assume a row index of 8 bytes, column index of 8 bytes, and a cell counter of 8 bytes. Effective triple stores require 3 inserts, which would cost 72 bytes. Supporting high performance of more advanced vector queries also requires storing the transpose of each matrix which would cost 144 bytes; this does not include the required overhead for B+ tree indexing, additional name space management, and many other details needed for large and numerous matrices. Intel® Saffron Memory Base has been consistently measured as requiring only 20-30 bytes/triple, including both connections as well as connection counts. As data size grows, the need for new matrices shifts to adding only new associations in given matrices and then to only the counter increments of given associations. Even larger memory bases lead to even greater efficiency.

These are the properties for an effective and efficient intelligence that the Saffron Cognitive Solutions group calls Natural Intelligence. Businesses must utilize the power of Natural Intelligence to survive in the real world, the same way human brains use it to survive on earth.
The word “associations” has a long history, dating back to Aristotle and the basis of his observations and belief that people learn and reason from memory, by what he called “recollection.” Associationism also underpinned modern neurology and psychology at the turn of the 20th century. Neurons connect to each other. Ideas connect to each other.

During the birth of computer science, Vannevar Bush, the Father of Modern American Science and considered to be the greatest engineer of the 20th century, hoped to build computers based on the idea of “association rather than indexing.” Bush's article “As We May Think”[^9] is credited as the original idea for hypertext, the Web's interconnection of documents, now becoming the Semantic Web of knowledge as the network of ideas within documents.

The right representation is required to address the scale of all “things” and the complexity of connections between these things. Brains represent networks of networks of associative networks because this also reflects the real world. The scale and complexity of things drive the growing need of big data analytics as well. Google's Bigtable was a boon for the massive indexing and retrieval of documents, but the last years have seen the rise of more dynamic systems like Google's BigQuery for interactive analysis, along with other initiatives for real-time streaming, graph stores, and matrices as the best and unifying implementation.

Imagine a memory for every customer, with matrices to remember what they looked at, liked, not liked, commented on, purchased, returns, shipping locations, and the patterns associated with these actions. Imagine that these customers share a connection, and influence each other, not as a simple friendship graph but as contextual links on many levels, depending on product specifics and other conditions of time, location, and/or weather. Imagine a memory for every vehicle part, remembering all its history and events, such as failures and fixes, also knowing its connections and conditional connections to other parts.

Data-oriented attempts to represent and exploit such scale and complexity become massively inefficient at big data scale. This is why brains are a memory base – not a database. As brains become overwhelmed by the volume and velocity of growing data, Saffron Cognitive Solutions provide the right representation for Cognitive Computing, “thinking” like people do, helping make sense and make decisions to provide assistance as Vannevar Bush first imagined.


[6] To learn more about the Kolmogorov Complexity and our methods for identifying the strength of a relationship link, we invite you to read our “Reasoning by Cognitive Distance on an Associative Memory” Technical White Paper.

[7] Lazy learning - David Aha, 1997
