DataSea Summary

The goal of DataSea is to greatly simplify the use of computers.

'DataSea' is a name which refers to a fluid merging of data. It is in fact an architecture and processing model which fuses information sources into a neural-like, self-organizing network. It responds to queries, commands and new input. It deals with long chains of relationships, crossing boundaries between disparate sources.

This is a radical change from current methods of computing which are dominated by keyword-search and forced-choice menu navigation.

'Query' today is essentially finding 'hits' by calculating the intersection of sets, where all the terms must be found in one source. There are attempts to exploit parts of speech today, but they fail to allow inferencing between different sources. Relational databases do allow long chains of relationships but require complex query construction and rigid, preconceived relationships.

DataSea finds, and explains, chains of relationships between multiple sources. It can give specific answers to inquiries, which allows inferencing, and also flesh out information in the 'neighborhood' of the chains of relationships it finds. Most interactions are one step, from a natural-language interface.

Motivation

Computers are hard to use: data is isolated and incompatible; multiple steps are required and <u>we</u> need to do the inferencing; usage models are pre-conceived by the programmers and require the user to have *a priori* knowledge of data structures; we can not simply say "Y*es! More {or less} of that"*, although we would certainly like to.

In short, we would like to simply tell the machine what we want, have <u>it</u> fuse data sources and present chains of inference to us, and then teach it if it's not correct.

Progress is blocked by early 20th century concepts where the programming mimics the machine construction (the so-called von Neumann machine). We should be able to simply ask for what we want. But today <u>we</u> must tell them *where* to look for information, or exactly which terms to find, and then reassemble the pieces of the answer into a coherent chain of reasoning.

Three substantial hurdles that have blocked progress in making sense of large quantities of data are (1) information fusion from disparate sources of different structures, (2) the straightforward extraction of information by non-experts and (3) teaching the machine to do better by interactive feedback, improving future results to both the given query *and other queries not yet asked*.

These three problems are exactly addressed by the existing DataSea architecture and processing model.
For instance, given two or more sources of information, to answer the simple question "<u>what is Bob's cousin's phone number</u>" using today's normal tools, we would need to translate it into a series of clicks or arcane programming steps. But the above question is exactly the sort of query that DataSea currently answers.

The DataSea system fuses multiple sources, both structured and natural language, allowing inferences within and between sources. It supports natural language user interface. It learns from user feedback.
The base technology is demonstrated today in DataSea's plug-in for Salesforce.com which fuses a relational database with user files.

Intellectual Merit

Most computer software today is still rooted in mid-20th processing models. Future computer architectures should improve human-computer interfaces by adapting to us, plausibly modeling certain processes of biological organisms (bio-mimicry). The neural-like technology of DataSea represents early-stage cognitive computing and applies to real-world tasks.

Broader Impact

This proposal gives direction to the long-term goal of biologically-inspired models in computer science. Being able to blend and infer between different sources, and get information out easily, has broad application indeed, being applicable to most every user interface, for example: giving web-site users direct paths to URLs of interest, to fusing medical records with simple access to physicians, to voice-activated home-assistants, or 'bots, for the infirm.

Objectives and Expected Significance

Three important challenges to the extraction and visualization of information from disparate sources today are:

(1) information fusion – how to represent information and draw specific inferences within the fused data
(2) information extraction by non-experts – ideally using natural language
(3) teaching the machine to do better next time – by giving feedback on the quality of results

We have developed a working software system which solves the above challenges and is embodied in a beta product “DataFusion”.

The objective of this project is to show the viability of this architecture and processing model to large data sets of broad interest, and the graphical tools to visualize and tune the results.

Ideally an information system would let non-experts ask questions using simple commands, or even natural language, to run queries and commands, as well as provide a native language to process complex tasks which are not easily expressed in English.

Such a system would be capable of:

- intuitively visualizing the results, and interactively 'teaching' the machine which were good and bad results
- fusing data from multiple sources (databases, HTML, text, spreadsheets, ad-hoc notes)
- inferencing and associative processing (neural-like processing and discovery of proximate information and discovery of deep linkages)
- explaining its answers by showing precisely the relationships leading to the answer
- being robust in the face of different source nomenclature
- having a simple user interface (natural language or simple native commands)
- complex command execution (eg., "timeline messages coincident with the system crash")

The approach taken by DataSea is to translate information, it's structure and meta-data, into a neural-like architecture. This design is based on real neural networks studied during the author's PhD research, and merged with what he learned in his subsequent involvement in computing at the Palo Alto IBM Scientific Center and MicroUnity Systems Engineering.

The underlying technology is demonstrated by the existing commercial (beta) product, DataFusion.

Background and Technical Need

Processing of data today is still largely based on the classic Von Neumann architecture, developed in the mid-20th century, and reflects traditional rule-based calculations, or, in the case of neural network models, processing "layers" that transform input to an output which are especially useful for pattern recognition but have difficulty 'explaining' why they give the answer they do. These methods do not do enough to address advances in the understanding of neural science and 100,000 years of language development.

Most databases require *a priori* knowledge of the database layout and of the user interface to get what we want. We need to specify *exactly* where the content is stored, e.g. which column from which table from which source. Query expansion is difficult (finding information 'near' specified data), and their internal structures are incompatible with other systems.

Further, putting together pieces of information to form a coherent whole, or chain of reasoning, relies on the user to guess, and piece it all together.

The typical user interface (UI), whether it is menu-driven or SQL, often requires expert operation. The commonly used relational database (RDB) make the above goals extremely difficult, requiring expert, hand-crafting of the queries, ontologies, and the internal structure itself. Extracting information from data sets usually involves expert training and the anticipation of use cases, which must be built into the system before it can be utilized.

Adding new information to the system usually requires restructuring of the database and reconfiguration of the application's anticipated use cases.

In summary, the basic design of computers today leads to these problems:

- users need *a priori* knowledge of both the applications and how the data is stored: this knowledge changes from system to system, and source to source
- search engines are incapable of automatically discovering complex relationships
- fusing databases is very hard, as is direct data sharing between systems
- inferencing automatically is not done

To 'fix' (or finesse) the basic design flaws of computer data storage and their processing models requires ad hoc, application-specific, expensive and complex programming.

The capabilities demonstrated in DataSea include:

- **fusing data** from multiple-sources (databases, HTML, natural language text ...),
- **inferencing and associative processing** (normalized, content-addressable storage and neural-like processing)
- **learning** (learn to avoid repeating the same false positives)
- **explaining** its answers showing the critical relationships

and the features include:

- **robustness** in the face of different source nomenclature and poor data (addressing false-negatives by fusing thesauri and word-stem lists)
- **simple user interface** (natural language interface or simple native commands)
- **launching of actions** (e.g. mapping, time-lining, phone-calling) from natural-language such as "*timeline messages coincident with the system crash*"

Attempts to standardize information on the web typically (RDF) requires extensive labeling and tuning.

When assimilated by DataSea, non-standard terms and special meaning are handled not by modifying the data, but by assimilating information which ties it together; for instance, to handle non-standard terms, the synonyms themselves are assimilated, providing the needed link to tie it together. The context is kept, and 'query explosion' is avoided by the network structure and processing algorithms.

Technology Introduction

DataSea technology is all about managing long chains of relationships. DataSea 'solves' the whole problem at once.

It is fundamentally, in fact ***essentially***, different from the common programs of today.

Traditional programs require breaking up a problem into short chains of relationships, executing the parts and reassembling the results. But for a computer system to 'connect the dots' to find non-trivial answers, it must be able to trace deeply through many relationships, finding those of importance. Yet the typical system operates on segregated chunks of data, unable to trace deeply inside the data. On the other hand, DSea nodes are highly connected and in principle any data node can 'see' all the other data, it can communicate, or connect itself up to, other nodes in the system.

The trick is how to assimilate real-world data, how to fuse it, how to process it. DataSea has solved these problems by virtue of its novel architecture and processing model. It is memory intensive but amenable to multi-threading certainly, and might benefit from porting to neural-chips, and is thus scalable.

Simple Example:

Content addressable search is usually keyword based. A Google search of "`what is Bob's phone`" gives 485 million results. Oddly, `Bob phone` gives only 81 million. Exploiting the information available by parsing the parts of speech can help with such a simple query, as long as it is in one source.

In general, today, if we want to search for an answer on a computer we 'triangulate',
or 'zero-in' on the answer.

If we are using a spread sheet, then we zero-in using the correct row/column,

Name	Address	Phone	Birthday	Email	Company	...
...		|				
Bob	------------	**111-2222**	------------			
...		|				

Ex.1 Spread-Sheet requires us to navigate to (x,y)

For a database we use a special language and specify the database, table, row and column to use, eg.

```
"use Personal_Database;  select Phone_Number from MainTable where Name like 'Bob %'"
```

Ex.2 An SQL query requires *a priori* knowledge of the database scheme

The simple example above takes more or less one 'step', albeit a rather complicated one.

More Complicated Example:

But what if we don't know enough yet?

What if we know the guy is Gina's brother but don't know his name?
Today this is solved by manually running multiple queries, either by hand or by custom code.

```
"use Personal_Database;  select HomePhone, CellPhoner from MainTable where Name=(select
SiblingName from MainTable where Name like “Gina %'")
```

Ex.3 SQL queries can be complex, for what we'd regard as a simple question

But DataSea thrives on these sorts of things.
It solves multi-step problems with one command, and that command can be exactly what the user had in mind in the first place:

"what is Gina's brother's phone?"

Ex.4 DataSea query

DataSea gives the answer specifically:

Answer: "111-2222, is Bob's phone"

Ex.5 DataSea: direct answer

It can do this because it can manage tremendously complex networks of fused data.

Gina − brother − Bob − phone − 111-2222

Ex.6 DataSea manages long chains

instead of the traditional need break it into two steps:

#1: Gina − brother − **Bob**

#2: **Bob** − phone − 111-2222

Ex.7 Traditionally, 2 steps are needed to get answer: cumbersome

DataSea can just as easily add the adjective 'elder', and use the synonym 'number':

"what is Gina's elder brother's number?"

Gina − brother − elder − Bob − number − phone − 111-2222

Answer: "111-2222, is Bob's phone"

Ex.8 DataSea: more complex query, very long chains, exact answer

Although the exact nodal architecture, translation and processing algorithms are proprietary, the above captures important concepts.

From this essential design emerges a surprising number of 'natural' results, like
- drawing inferences between disparate data sources,
- 'connecting the dots' and finding 'hidden' relationships,
- using natural language for input and queries,
- using voice-recognition for direct communication with the machine.

Biographical Sketch

Dr. Rocky H.W. Nevin, III (CEO, CTO DataSea, Inc.), the inventor of DataSea's technology, earned his Ph.D. in Biophysics at the University of California at Berkeley exploring the importance of connectivity and structure of sensory neurons and identifiable interneurons to complex behavior. Following two years at IBM's Palo Alto Scientific Center, he created and managed the Computer Integrated Manufacturing Group at MicroUnity Systems Engineering, where he wrote the visualization and control software for the company's $100 Million wafer fabrication plant (3D-Ops).

Email: Rnevin@DataSea.com 510.981.1084
Web: www.DataSea.