

White Paper

An Efficient System and Method for Organizing and Analyzing Data

Version 1.0

Contents

THE CHALLENGE 3

THE SOLUTION, RELEASE THE CPU TO DO WHAT IT DOES BEST 3

RE-INVENT WORDS 3

NUMERIC WORDS 4

WHAT CAN BE DONE WITH NUMERIC WORDS? 5

COMPARISONS 5

CLUSTERS, CLUMPS, PATTERN 5

TRANSPOSITIONS 6

COMBINE RECORDS 6

ORGANIZE DATA FOR BEST USE 7

VIEW & SUB DATA STORE 7

THE DOWNSIZE OF INDEXES IN TRADITIONAL DATABASE ENVIRONMENTS 8

THE ADVANTAGE OF THE BINARY SEARCH 8

FINANCIAL PROCESSING EXAMPLE 9

BACKGROUND 9

CREATE A DISTILLED REFERENCE BASE 9

ADD NEW FILES AND GROW REFERENCE BASE 10

CONCLUSIONS 11

ABOUT HILBERT 12

The Challenge

The sudden emergence of computers and the subsequent explosion in their numbers, on both the business and the home, seems to have caught everyone by surprise. We can generate and store more data and information that we can even begin to process.

Both miniature word processors and database applications sprouted up at almost the same time to match the capabilities of the micro CPUs. Database engines resembled word processors and data processing became more like a committee debate than analysis. Such hobby-sized applications flooded the market. Competitive forces caused them to accumulate features and to grow in size and importance. By sheer force of numbers, microprocessor applications overtook the mainframe business.

There was a quantum leap in hardware capability. Oxymoronic, giga-microprocessors evolved. Coincidentally relational databases became popular, although they were still more like word processors than computers, both convoluted and complex inside and out.

Not many people were thinking of documents (prepared on word processors) as quantifiable data stores. Historically, documents were created by people, read by people, filed and retrieved by people and thus analyzed by people. Automated document processing began by mimicking what people do, resulting in massive applications founded on overly complex and often contradictory assumptions and rules.

Today, the enormous quantity of accumulated data can no longer be processed by anthropocentric algorithms, which are mainly based on human perception. There just isn't enough processing time available, even on modern super-micros.

A new paradigm is required: one in which numbers, mathematics, axiomatic logic and scientific reasoning rule. By way of comparison, it requires millions of times less time to reach an analytical result than to stage a trial, make arguments and pray for a judgment.

The Solution, Release the CPU to do what it does best

At the core of a computer there lies one or more Central Processing Units (CPUs). CPUs process numbers. Nothing beats them for grinding out results from numbers. The obvious first step is not to express the problem in social or journalistic term but to express the problem in numerical, mathematical terms.

Re-invent Words

Change from the narrow world of word processing to the infinite universe of mathematics.

To re-invent words, just think of words as numbers and use them as such. After all, no one has trouble thinking of "12345," a string of characters, as a number. One does not compare "123456" with "654321" character by character to determine whether they are the same number. The following sections explain how all words (including "123456") are actually numbers.

Numeric Words

For example, consider this **binary** number:

11110000011010011110010001111010100111
10010010001011₍₂₎

Which, in **octal** representation, would be:

360323621724744427₍₈₎

And, converted to a Decimal representation would read:

8458803055151383₍₁₀₎

Whereas, in a Hexadecimal form the number would be:

1E0D3C8F53C917₍₁₆₎

But behold, in TetraDecimal (base 40 or "tet") this same number would be:

WASHINGTON₍₄₀₎

Everyone immediately recognizes this string of characters as a Word although it is really a number.

Any arbitrary number of symbols can be used to represent digits in a number. We selected the following forty characters for this example:

0123456789ABCDEFGHIJKLMN O P Q R S T U V W X Y Z # ! @ ^

This particular selection of the base-40 digit symbols allows processing of the usual decimal digits in context with 26 upper-case alphabetic characters and four extra symbols. Any one the symbols can be set by the user, according to the needs of the problem.

If an application needs both upper and lower case symbols, then base-64 numbers can be used. Fundamentally, one must consider the nature of the problem environment and make an appropriate selection of the number base and the symbols used to represent the digits.

In HILBERT, one can conveniently switch between number bases and symbol sets during processing, i.e. on-the-fly, as dictated by the analysis strategy.

With 40 for a base, one can store six characters in a single precision, 32-bit computer word. Double precision, or 64-bit, computer words can hold 12 tet characters. Using 64 as the base, a single 32-bit single precision, computer word holds five characters and, naturally double precision computer words hold 10 characters. Base 85 can be applied if support for more complex symbols is required for Chinese, for example. The ability to deal with Chinese characters and hence the language itself is no different.

The Hilbert Engine uses, for example, a BASE-forty System:

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39
0 1 2 3 4 5 6 7 8 9 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z # ! @ ^

M A R Y

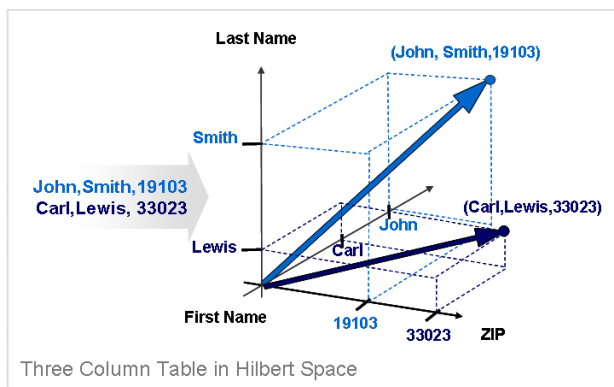
MARY₄₀ = 22 x 40³ + 10 x 40² + 27 x 40¹ + 34 x 40⁰ = 1,425,114₁₀
= 1,425,114₁₀ = 10101101 11110110 11010₂ ← apx. 50% reduction →

MARY = 4D 41 52 59 00₁₆ = 01001101 01000001 01010010 01011001 00000000₂

Quantification: Text to Integers

What Can Be Done With Numeric Words?

Regarding words as numbers has many important implications, but go far beyond merely converting words to large decimal bases. Words are simply placed in a coordinate system. More importantly, they become elements in a multi-dimensional hyper-geometric space. All of more than 400 years of mathematics is now available



to solve problems based on words. Vector and tensor analysis can be used. The transforms, as used in relativistic and quantum physics, that brought us the phenomenal developments of the 20th century are now within reach for processing textual data.

With words-as-quantities, language, words and meaning enter the universe of, solid, reasoned mathematics, physics and engineering.

Recent papers by David Horn and Assaf Gottlieb ^[1] ^[2] illustrate the use of quantum mechanical mathematics to locate clusters in

[1] David Horn and Assaf Gottlieb. Algorithm for Data Clustering in Pattern Recognition Problems Based on Quantum Mechanics. Phys. Rev. Lett. 88 (2002) 018702.

[2] David Horn and Assaf Gottlieb. The Method of Quantum Clustering. School of Physics and Astronomy, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv 69978, Israel.

number-based data. These papers and others like them make it clear that many new advances are in store, now that words can also be numbers.

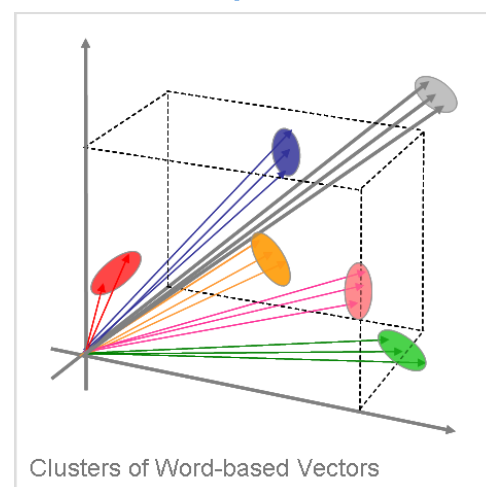
Having all data (numbers and text) as numbers a data record will become a vector in Hilbert Space wherein each field is a element. Once there are vectors, the computational horizons expand enormously.

All of the computational Hilbert files are numeric and normal arithmetic operations obviously work. Each of the fields (single files) is a vector element value and so normal vector addition, dot- and cross-product computations also work in the usual way.

Comparisons

Comparisons are now very quick. Because every value is a number and because CPUs only deal with numbers, comparing two words is a nearly instantaneous operation—only a computer clock cycle or two. Contrast this to the world of word processing and text where comparisons take millions of clock cycles.

Clusters, Clumps, Pattern



Clumps or clusters can easily be detected. The more dimensions (fields) there are in a particular vector comparison, the greater the distance there between the clumps. This is both an opportunity and a responsibility. The responsibility part is for the analyst to select the most relevant subset of dimensions for the clumping. If noisy or naturally un-related fields are included, they will have the effect of blowing apart data that should otherwise lie together.

Transpositions

The age-old numeric algorithm to detect transpositions is one example of what is very convenient with numbers but inconvenient with characters. It works like this:

Suppose you have two names of the same length and you want to know if they are the same name. Take, for example, "JOHNSON" and "JHONSON."

1. Subtract the two numbers

```

JOHNSON
-JHONSON
-----
 6X0000
    
```

2. A non-zero result, so the two names are not identical.
3. Divide by "40" which, in decimal is 39 or base - 1, ($40 - 1 = 39$) and get a remainder of 0. This result is suggestive of a transposition. Now check the digits. There are many ways to do this without resorting to characters.

Number theory, of which the above example makes use, is a rich and fertile ground for further research.

Combine Records

The Hilbert system can quickly produce all of the data from the database related to a particular question. This quantity of related data can be thought of as a vector also—a vector of a variable number of dimensions.

For example:

HUSBAND NAME1	JOHN
HUSBAND NAME2	SMITH
HUSBAND NAME3	JR
SSN	777-46-7777
WIFE NAME1	MARY
WIFE NAME2	ALICE
WIFE NAME3	SMITH

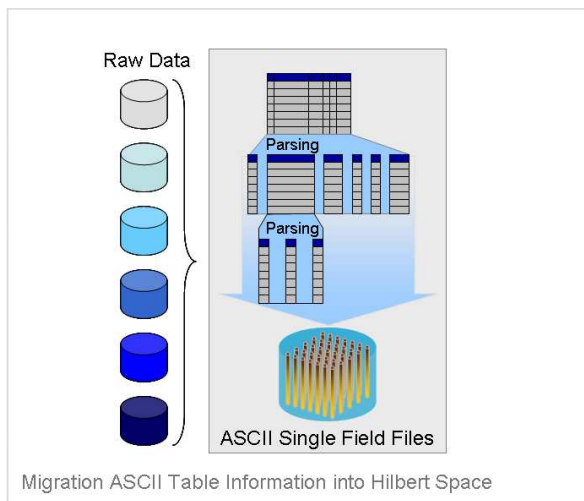
Is clearly a vector and that has seven relatively well-understood elements. However, if we add the various credit card accounts related to this married couple, we get a variable number of new vector components:

ACCT NO	123987R
BANK	CHASE OF OHIO
BAL	10.35
ACCT NO	43T71NN
BANK	FIRST BANK
BAL	125.99
Etc.	

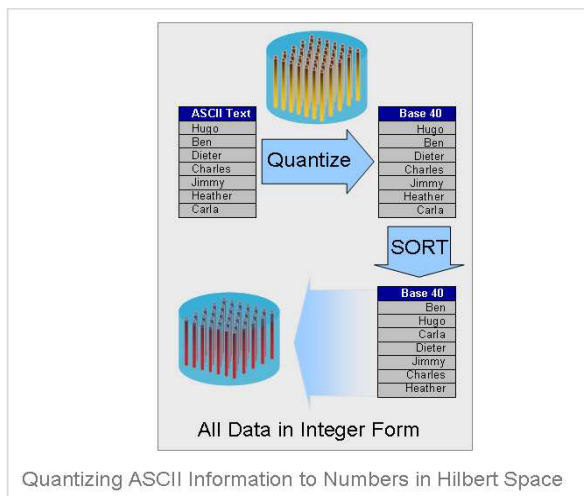
These related data, together with the name and spouse data make up another kind of joined vector.

Organize Data For Best Use

Hilbert is designed to make maximum and efficient use of the opportunities offered by the numeric representation environment.

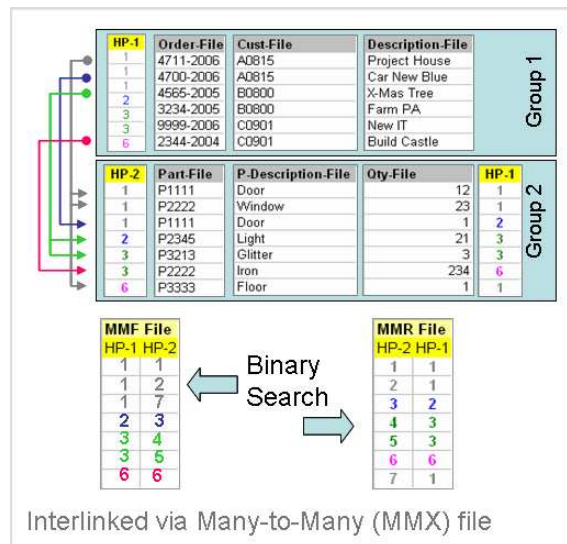


All data are held by the Hilbert system in numeric form. This promotes quick access and fast analytical processing. Optionally, Hilbert can also hold a text version of the same files if the analyst needs text for convenient report genera-



tion.

Data are stored in single word field files or organized in groups so that all members of the group, with one exception, share a common implied row number. The exception is for the sorted (*.srt) file type, and that type carries the row number explicitly. Groups are analogous to tables of a traditional database. Relationships between file groups are ex-



pressed as many-to-many (.mmx) reference files of two types: forward (.mmf) and reverse (.mmr). These reference files have two columns. For the forward reference .mmx file the first column contains the parent row number and the second column contains the corresponding child row number. It is sorted by the contents of the first column for easy searching. The reverse many-to-many reference file has the same two columns but they are interchanged. The child group row number is in the first column and the parent numbers are in the second. As before the first column is sorted for quick searching. All relations among groups are many-to-many and the Hilbert system is therefore equipped for the most complex relational situations.

View & Sub Data Store

As a direct consequence of its unique many-to-many relational structure the Hilbert system can quickly extract a complete sub-data store from the main structure.

An example will help clarify this: Visualize making a query for all records pertaining to "WYOMING" and from those results ask for all records pertaining to "ORANGE GOLF BALLS." At that point Hilbert can, in seconds, and without further programming, create a completely relational sub data store containing absolutely all data that relate in any way at all to "WYOMING" and "ORANGE GOLF BALLS." This sub database can be sent to a PDA, next generation cell phone or other small computer and be used by field personnel as they visit customers.

The Hilbert system makes the most use of the aforementioned sub data store capability to create a user specified vector view of all the records captured by a set of queries. These are called view vectors. Using the view vector system an analyst can quickly explore the data landscape from a variety of points of view and rapidly develop an understanding of the problem at hand.

The Downsize of Indexes in Traditional Database Environments

Traditional indexes are not needed in the Hilbert environment. Traditional indexes contain a column of words and a column of row numbers. It is the list of words that must be sorted and all searching is done in that same list of character-based words. A traditional relational database with a few indexes of this type can easily require an entire night work-shift to maintain the indexes. This is such a problem that Data Base Administrators (DBAs) of the usual relational database systems frequently restrict the number of indexes that developers are allowed to create.

In addition to such maintenance headaches, when an index is used in an analytical procedure, the list of words must be searched and text searches are very costly.

The Advantage of the Binary Search

Sorting lists of numbers, on the other hand, is inherently very fast. Millions of rows can be sorted in less than a second. Hilbert's sorted numeric data files serve not only a quick referencing function but many other useful purposes and at hugely faster speeds. A binary search in a billion record file requires less than thirty jumps, each of which is executed at the full numeric speed

prcinfo_prc_arrit	
11	2497300
5	185385
7	79030
10	66611
20	47100
21	44200
6	33950
9	33302
16	28015
12	7250
17	5850
8	3000
18	2380
22	2300
3	2274
4	2274
19	2200
14	2190
15	1455
24	963
27	862
26	619
23	554
25	453
13	420
1	100
2	100

? Find 79030
→ Binary Search 5 Jumps

1. Start in the Middle
2. If smaller go up half
If greater go down half
3. If not found

Binary Search in Hilbert Engine

of the processor.

Hilbert also makes use of its sorted files in a patented, feedback-control scheme to produce very fast NxN comparisons in very much less than N squared time. This is one of the features of Hilbert processing that makes the following commercial example execute in minutes rather than weeks.

FINANCIAL PROCESSING EXAMPLE

Background

As an illustration of the power of words-as-numbers and of the Hilbert environment, we present this example from the financial arena.

The client, in this case, was a law firm that specializes in processing the creditor side of bankruptcy cases. Every large credit organization, such as a bank that has issued credit cards, periodically receives large numbers of legal notices that some of their customer-debtors have filed for relief under the US bankruptcy laws. Many such creditors send the resulting legal correspondence to this law firm, our client.

The law firm's objective is to respond to the various court processes according to required protocol, with the appropriate documents and representations so as to qualify their client creditors to receive whatever payments the court eventually mandates.

This is a time-urgent problem, because each court has a precise and short schedule upon which the entire process hangs. Moreover, our client receives a great many new notice documents each day. Remember that the original notice was sent to the creditor-bank who must first recognize that it has been received, decide to send it to our client and then actually deliver it to them. This takes a certain variable amount of time and this extra time often leaves the law firm with only days to respond to the court.

Any given petitioner probably reports many debts for different creditors to the court. Many creditors use the services of our client. Thus any given petitioner will probably be reported to our client several times. Each creditor knows any given debtor by its own customer record and in its own peculiar format, and including its own errors. Our client has to gather all these versions into a single picture of petitioner and attach that picture to a particular bankruptcy case.

Several years ago, our client set up the required data operations on an XBase engine running on several mini-computers and using Clipper as the query environment. The data were held in related file structures according to its source and nature (e.g., debtor, debtor accounts, courts, lawyers, etc.). All went well until the system became overloaded. New errors and duplications began to develop in the database. The Clipper based system was becoming sluggish and the DBA management problems were reaching crisis proportions. Our client hired one of the big-5 consultancies to design and stand up a new relational database system with SQL as the query language. After several years and at the expense of several million dollars, the new system was ready. The consultancy devised what they considered to be an appropriate rule-based translation/migration system and tested it on simulated data. The test results showed that, by running day and night on several mini-computers, seven days a week, they would be able to "clean" and port all the data into the new system in something like 50 weeks. This was unacceptable. It was at this point that we were asked to help solve the problem.

Create a Distilled Reference Base

Our Client's idea had been to group all the data together in clusters centered on the court docket number. That is to say, identify all the person/account records that belong to a particular court case. Our first step was to change the focus from bankruptcies to persons and cluster the data into groups, each group referring to a particular person. Each such group was assigned **Person Identification Number**, a PIDN.

As an aside, when we focused on individuals, we were able to point out several instances of potential fraud, wherein one person seemed to have started more than one bankruptcy proceeding further indicating perhaps more than one identity. Working from the point of view of court docket numbers obscures the evidence of this type of fraudulent behavior.

Before beginning the real processing, we culled out all the records that had no meaningful data, the “orphans” that could not be connected to any other information. Obvious examples of this were records that had only the words “test test” in one field and no other data in the record.

The basic strategy is, after the extraneous records have been culled, to build a solid reference base of incontrovertibly identified persons. Such a group would have, for instance the same SSN, the same Last Name, the same First Name, the same address and the same Co-debtor name, etc. This is equivalent to performing a vector dot product and requiring a zero angle between vectors.

Add New Files and Grow Reference Base

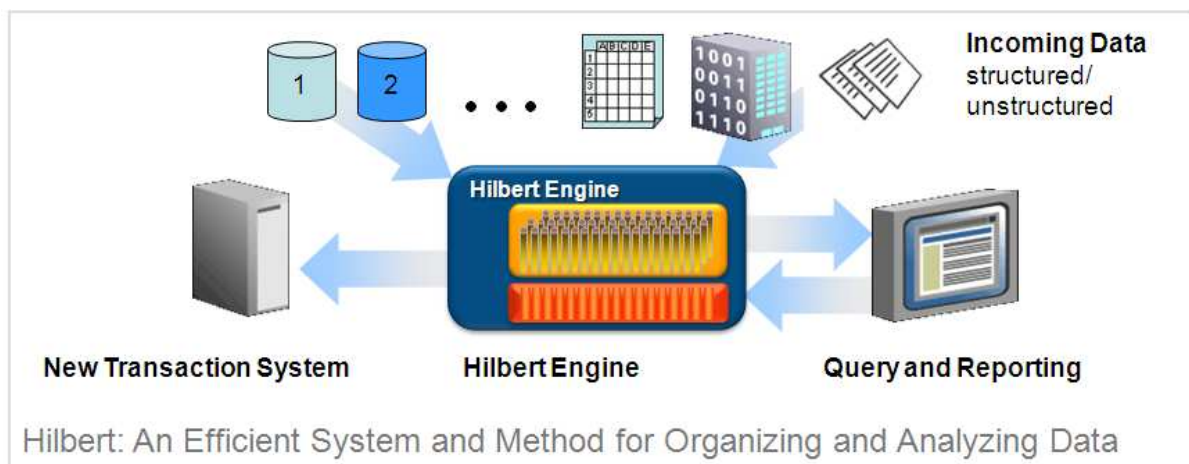
The plan is to grow the reference base in stages. With this small group of highly reliable IDs in hand, relax the criteria, one at a time, and re-compare the reference set to the unidentified pool in repeated passes.

One relaxation considers the possibility of transposed characters in the SSN or name fields, as explained above. If all the other chosen fields are identical in a particular record and there is a single transposition in a name or SSN, then that record is accepted into the reference-base PIDN.

There is also the possibility that the customer’s input parsing routine mistakenly reversed first name and surname. Thus another step checks to see if name-01 is actually the first name and name-02 is the surname. Such reversals are accepted into the growing reference-base as long as all the other fields match.

Operations of this kind are equivalent to allowing the angle between two vectors to be non-zero, but requiring it to be small.

At the completion of this process, we found that it is possible to produce all the needed PIDN groupings of this client’s data in only a few minutes on a relatively slow (by today’s standards) laptop. Consequently client analysts can experiment with many associative schemes, one after another in quick succession.



CONCLUSIONS

Hilbert has mathematical, axiomatic foundation, vs traditional database/word-processing naive logic.

Conclusions are based on rigorous, mathematical reasoning rather than relying on the limitations of SQL. The very existence of this capability encourages analysts to change the way they look at problems and build new and more accurate solution strategies.

Single-number field files create an efficient file structure. It is simple, easy to understand and it makes maximum use of the various over-the-network sharing schemes in common use today. Working with the problems of several clients, Hilbert has demonstrated that it can distribute several databases over as many PCs as are necessary to hold the data in a serverless server environment. Furthermore, the customer example clarifies that simultaneously five separate analyses on five separate machines using data from all of the individual databases with less than 10% degradation due to network delays have been successfully performed.

The inherent many-to-many relational structure of Hilbert and its view vector capability make it possible for an analyst to develop an extremely robust understanding of the problem space or businesses to handle massive data sets without the complexity of the current database technologies.

About Hilbert

Pennsylvania based Hilbert Technology Inc. is an international provider of data management and analysis solutions for large and medium size enterprises worldwide. The offering is based on the revolutionary, patented Hilbert Engine technology for the ultra high-speed access, manipulation, storage and analysis of large volumes of structured and unstructured data. The Hilbert solutions are offered as industry or process specific solutions. Organizations in public services, law, government, finance, communications, whole- and retail sales, transportation & tourism and chemical & pharmaceutical can employ Hilbert solutions to gain unparalleled speed in access and analysis over large data volumes.

This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor is it subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Hilbert Technology Inc.
P.O. Box 330
Newtown, PA 18940
USA
Web: www.hilbertcompany.com
E-Mail: info@hilbertcompany.com
Tel: +1 (212) 252 1600
Fax: +1 (212) 252 1615