GeoHash and UUID identifier for Multi agent systems

Conference Paper · June 2012

DOI: 10.1007/978-3-642-30947-2_33

CITATIONS
32

READS
831

3 authors:

Zoran Balkić
Base58
9 PUBLICATIONS **60** CITATIONS
SEE PROFILE

Damir Sostaric
Base58 d.o.o
31 PUBLICATIONS **176** CITATIONS
SEE PROFILE

Goran Horvat
Base58 d.o.o.
41 PUBLICATIONS **199** CITATIONS
SEE PROFILE

GeoHash and UUID identifier for Multi agent systems

Zoran Balkić[1], Damir Šoštarić[2] and Goran Horvat[2]

Department of Automation and Process Computing[1], Department of Communications[2]
Faculty of Electrical Engineering, J. J. Strossmayer University of Osijek
Kneza Trpimira 2b, 31000 Osijek, Croatia
{zoran.balkic,damir.sostaric,goran.horvat}@etfos.hr

Abstract: This paper documents the case to address the distributed data storage and widely used unique identifiers which are in use from the beginning of databases and backend systems. The importance of uniqueness for object system identifiers is crucial for efficient data retrieval, storage and comparison. We are representing the new method for objects tagging using synergy mechanism between well known Geohash algorithm and Universally Unique Identifiers we call GHUUID (Geohash+UUID) gaining the agent system spatio-temporal identification of all data available. Multi-agent systems in their nature strongly rely upon distributed databases across vast amount of data. Some of the interoperability issues are solved with usage of semantic web principles which can be cumbersome to implement and maintain. Most systems need the basic space-time correlation which can be the corner stone for the efficient analysis and data aligning regardless of objects stored and their nature.

Keywords: Multi-agent mechanism design, distributed unique identifiers, geo location awareness, data filtering and analysis

1 Introduction

Since the beginning of the information technology era there has been a necessity to implement unique identifiers over stored data. Using identifiers has become the base for successful database design as well as different programmed solutions in all programing languages and data formats. Without them it would be practically impossible to address the specific object in the local data space including, relational, document oriented or hierarchical databases in use today. Identifiers are the basic structure in every modern database implementation and the core of every relational database ensuring the consistence of data and relational paradigm using primary and foreign keys for schemas and object reference. Multi-agent systems are the one that benefit from this methodology the most since identifiers are essential for successful data retrieval over different database tables as well as data update and deletion. Graphical user interfaces of all kinds (web, fat clients, embedded systems etc.) strongly rely on identifiers uniqueness and consistency. All modern programming languages and data formats use them heavily during creation and annotation process for automatic programming models generation. The most important characteristic of the ID's is their

uniqueness and the fact that once identifier is generated can not be duplicated at any time, regardless of object deletion or move in the data instances space. All database solutions today implement this mechanism to ensure that ID's, often called the "key", should not be duplicated which would lead to data corruption and inconsistency. Database relations have ID's in their core allowing classic relational database to operate.

Usually the ID's of the objects in the database table are used for this purpose, not offering any global identifier in the global information space. With the introduction of UUID's this goal is implemented and conducted in the way that globally unique identifier is satisfied, offering the true uniqueness across different systems without need for centralized authority for issuing those identifiers.

UUID is time-based generated identification mechanism and, as such, highly precise and collision free.

With the rise of Geo location aware systems lately, the importance of change in agent's infrastructure design is evident.

2 Motivation

Multi agent system (MAS) is the system that consists of a different number of standalone agents which should interact with each other. The main requirement for their operation is cooperation, coordination and negotiation like people do in their everyday life. As people are mainly interested in space and time information of the object, this information is the main goal for implementation into multi agent systems on the global identifier level.

One among the first MAS applications was distributed vehicle monitoring (DVMT) [1,2] where a set of geographically distributed agents monitor vehicles that pass through their respective areas, attempt to come up with interpretations of what vehicles are passing through the global area, and track vehicle movements. The DVMT has been used as an MAS test bed.

To enable greater decision making process simpler we propose Geohash+UUID (GHUUID) mechanism which would take location based data, obtained from the network or explicitly from GPS data, with UUID functions. Both GeoHash and UUID functions are implemented in most modern database systems and widely available as standalone functions in modern programing languages for both, client and server side usage.

Searching the Internet and distributed databases for a specific query can be a long and tedious process thus assembling the right tool for searching and retrieval of data between higher correlated data based on GHUUID is of high importance.

Internet or network enabled agents, typically needs to gather information that would require synthesizing pieces of information from various available information sources and GHUUID is a perfect tool for simplifying that task.

According to [3] multi agent based approach to clustering, that harnesses the processing power of a collection of "clustering" agents, to produce a "best" set of clusters given a particular clustering problem does not exist. General purpose, best clustering algorithm that is suited for all data is still unavailable, therefore GHUUID is one of

the best methods providing instant alignment tool that could enhance multi agent data mining environments. The accuracy of a set of multi agent clusters requires pre-labeled data which needs to be manually added to the system that entails an undesirable overhead. In the [3] two measures are introduced for multi agent systems clustering, Within Group Average Distance (WGAD) and total Between Group Distance (BGD) which needs data mining ontologies for efficient multi agent Data mining (MADM).

The main contributions of this paper are as follows:

- Introduction of new identifier that consists of Geohash encoding and UUID unique identifiers which eliminates the usage of any external labeling of data or data mining ontologies.
- A generic approach to multi agent based clustering techniques to identify a best set of clusters using a GHUUID which relates agents according to space and time measurements
- Development of the test environment based on Java Enterprise Edition (JEE) with Spatial enabled MySQL database server for GHUUID showcase

3 Geohash

Geohash is a latitude/longitude geocode system invented by Gustavo Niemeyer for encoding/decoding (lat,lon) pairs in a compact form.

The Geohash algorithm [4,5] can be used to divide geographic regions into a hierarchical structure. A Geohash is derived by interleaving bits obtained from latitude and longitude pairs and then converting the bits to a string using a base-32 character map. A Geohash string represents a fixed spatial bounding box. For example, the latitude and longitude coordinates of *45.557, 18.675* falls within the Geohash bounding box of "*u2j70vx29gfu*". Appending characters to the string would make it refer to more precise geographical subsets of the original string.

Encoding the position takes only (lat, lon) pairs neglecting the third coordinate (altitude) and as such it is unaware of real 3D position of the object in space.

As a consequence of the gradual precision degradation, nearby places will often (not in some edge cases, equator or a meridian) present similar prefixes. The longer a shared prefix is, the closer the two places are.

To obtain the latitude and longitude bits from an initial pair of coordinates representing a target point in space, the algorithm is applied recursively across successively more precise geographical regions bounding the coordinates. The remaining geographical area is reduced by selecting a halfway pivot point that alternates between longitude and latitude at each step. If the target coordinate value is greater than the pivot, a 1 bit is appended to the overall set of bits; otherwise, a 0 bit is appended. The remaining geographic area that contains the original point is then used in the next iteration of the algorithm. Successive iterations increase the accuracy of the final Geohash string. An appealing property of the Geohash algorithm is that nearby points will generally share similar Geohash strings. The longer the sequence of matching bits is, the closer two points are.

This property is exploited in GHUUID to support simple range-based spatial queries that return data in a given Geohash region, allowing agents to specify more or less precise hashes to select smaller or larger areas. It is also possible to use Geohashes for quick proximity searches. In addition to queries, Geohashes can also be used to group similar data. If a collection of data gets too large, simply using more precise Geohashes allows the system to create more specific, and therefore smaller, groupings of data. This property also allows quick retrieval of similar data blocks for use in computations.

In this paper database GeoHash encoding function is created for the MySQL database since it is not part of the database distribution. Standard database stored procedure, in this case, is deployed into the database which is triggered upon any data creation or update and stored in the separate table for faster data retrieval, search and comparison.

4 Universal Unique Identifier (UUID)

Universal Unique Identifier (UUID), also known as GUID (Globally Unique IDentifier) is generated according to [6]. A UUID is designed as a number that is globally unique in space and time. Two calls to UUID() function are expected to generate two different values, even if these calls are performed on two separate computers that are not connected to each other.

Stated in [7] UUID is a 128-bit number represented as an utf-8 string of five hexadecimal numbers in aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee format and requires no central registration process:

- The first three numbers are generated from a system timestamp.
- The fourth number preserves temporal uniqueness in case the timestamp value loses monotonicity (for example, due to daylight saving time).
- The fifth number is an IEEE 802 node number that provides spatial uniqueness. A random number is substituted if the latter is not available (for example, because the host computer has no Ethernet card, or we do not know how to find the hardware address of an interface on your operating system). In this case, spatial uniqueness cannot be guaranteed. Nevertheless, a collision should have very low probability.

Several variants and versions of UUID generated strings exist today:

- Version 1 - MAC address based with the time at 100 nanoseconds intervals since the adoption of Gregorian Calendar
- Version 2 - DCE security with the upper byte of the clock sequence replaced by the identifier for a "local domain"
- Version 3 - MD5 hash version uses scheme deriving a UUID via MD5 from URL
- Version 4 - relies on the generation of random numbers
- Version 5 - SHA-1 hashing instead of MD5 like Version 3

Currently, the MAC address of an interface is taken into account only on Linux Operating systems. On other operating systems, most systems use a randomly generated 48-bit number.

Since we are interested only in "time" part of the UUID strings for the spatio temporal analysis we are using the Version 1 for the construction of GHUUID identifiers.

Standard Java library does not contain UUID generator which can generate timestamp dependable UUID strings, only Versions 3 and 4 due to lack of means to access MAC addresses using pure Java before version 6.

Even though UUID string could be generated server side using third party Java libraries, in the test process we have used built in UUID function of the MySQL database server which eliminates the external generation of UUID strings thus speeding up the system. Extraction of the "time of creation" from the encoded UUID string is implemented in Java as standalone utility class which is used in the developed JEE application.

5 Geohash UUID (GHUUID)

In this paper we propose the new system for generating system wide global identifiers with all characteristics essential for the high quality identifier that is spatio-temporal aware. GHUUID consists of two parts, first one id GeoHashed longitude/latitude pair of data which can be obtained from either agent, server side system or directly from databases and the other, standard UUID that could be generated on the backend side of the multi-agent system. In the case UUID is generated on the backend side, it contains the IEEE 802 node number that provides additional spatial data, even though this information could be of less importance to the agent itself.

GHUUID's in essence are variable length data depending on Geohash precision encoding. In most cases the significant precision of uniqueness is 192-bits (Geohash – 64-bits and UUID – 128-bits) long.

Example of GHUUID looks like:
(u2j70vx29gfu-2d004620-3fc7-11e1-b86c-0800200c9a66)

- First part: *u2j70vx29gfu* represent GeoHashed lat/lon information that corresponds to the initial geo coordinates of *45.557, 18.675*.
- The second part: *2d004620-3fc7-11e1-b86c-0800200c9a6*6 is generated UUID that contains a timestamp taken at Sunday, January 15, 2012 10:20:48 PM GMT.

Multi agent systems operating in the way of using GHUUID could simply from those identifiers extract information needed for spatio-temporal awareness over position in the real world and the timestamp for the data either creation or replacement.

GHUUID data can be directly shown in the client Geo visualization software through standard mapping graphical user interfaces with direct links to the object data that are stored on the agents or central system of communication.

Further on, as GHUUID contains timestamp data it could also be presented in the timemap clients giving us the spatio-temporal presentation straight from the system

identifiers (Fig. 1.) regardless of the nature, context or data information of the object visualized.

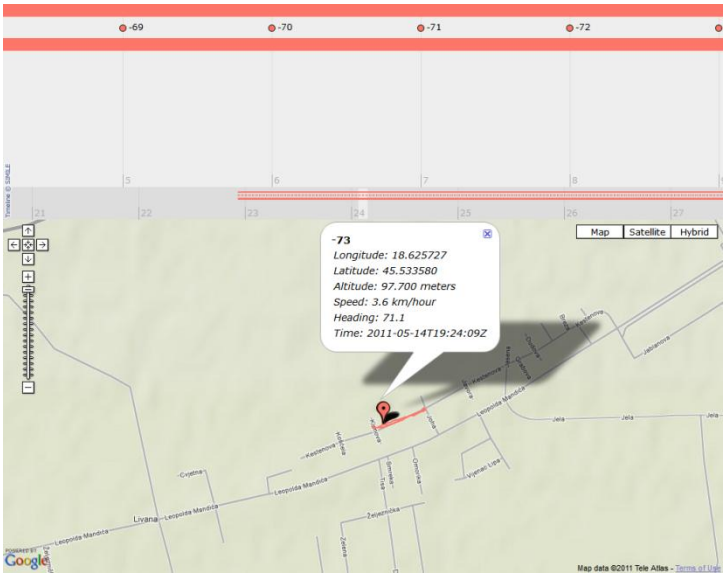

Fig. 1. Timemap view of the system GHUUID identifiers for mobile Agent with corresponding trajectory

5.1 GHUUID prototype testing application

Prototype application is developed using Java Enterprise Edition using Servlets and Java Server pages, deployed into standard Tomcat Java server coupled with MySQL database server containing spatial extensions.

We have chosen Java for the compatibility with the JADE [8] a software framework that can be considered one of the most known and used software frameworks for the MAS development and possible future integration with it.

Full system test was made on the data set that consists of 6988 mobile Multi Agents and 52 static Multi Agents which are static in space thus changing only their time component. All mobile Agents have their data collected in the time period of 2 minutes with the pseudo random pattern across the space.

GeoHash is calculated based upon current location of the Agent (Fig 2.) with coordinates (*45.534, 18.628*) and passed to the application in full GeoHash encoded form (*u2j70vx29gfu*) constrained by the number of characters used to match the neighboring Agents. This process is iterative in the manner of shortening the Geohash from six to one character (*u2j70v, u2j70, u2j7, u2j, u2, u*) giving greater number of matching Agents in space and speeding up the query execution.

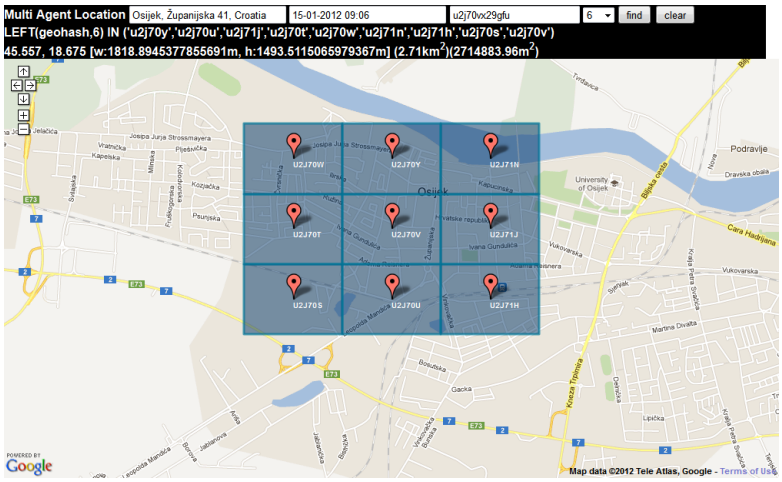

Fig. 2. Map view of the searched GHUUID identifiers

GHUUID is consisted of two identifiers and their relation in the process of determining the closest Agents in space and time can vary from 0-50% in both directions from space to time and vice versa as shown in (Fig 3).

Calculated GHUUID weight ratio (%) visualized in (Fig. 4.)

- 20% in the favor of Location
- 40% in the favor of Location
- 50% in the favor of Location or Time (used in this test case)
- 40% in the favor of Time
- 20% in the favor of Time

Multi agents that are more related in space and time are colored red and ones that are less related are colored green (Fig. 3.).

Regardless of the type and nature of stored data per Agent efficient relation is calculated in real time and metrics of queries sent to the database system is shown in Table 1. The results of the metrics brings interesting conclusion that searching the wider area, which gives more matching results, takes less time than more specific and narrowed queries which is not the case in the classic Geo spatial queries which usually take more time with the larger data sets.

GeoHash Agents	1	2	3	4	5	6
1	0.0101	0.0104	0.0106	0.0386	1.3634	1.3736
2	0.0094	0.0097	0.0102	0.0394	1.3705	1.3722
3	0.0088	0.0094	0.0102	0.0373	1.3621	1.3664
4	0.0086	0.0091	0.0101	0.0373	1.3568	1.3697

5	0.0083	0.0089	0.0102	0.0371	1.3555	1.3705
10	0.008	0.0092	0.0100	0.037	1.3501	1.3711
100	0.0098	0.0103	0.0110	0.0409	1.4626	1.4705
1000	0.0144	0.0151	0.0161	0.0599	2.1418	2.1533

Table 1. Metrics of the system performance in Agent number per GeoHash lenght (characters) and the query results in seconds

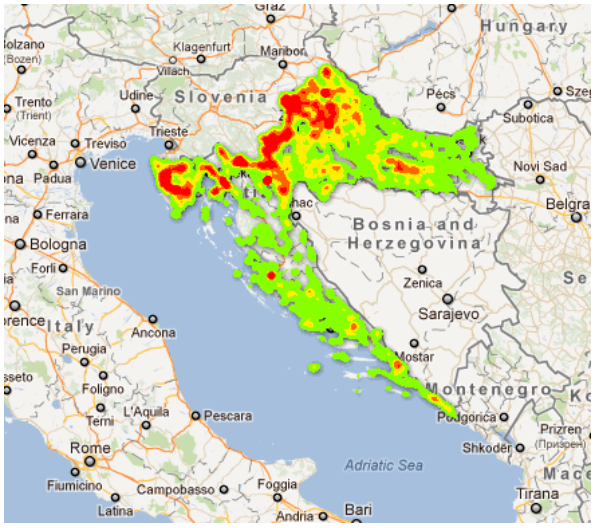

Fig. 3. GHUUID relation heat map

Fig. 4. GHUUID ratio visualization

6 Conclusion

Design and implementation of agent systems is difficult task. All the standard problems that exist with building traditional distributed, concurrent systems are emphasized with additional requirements like smart interactions between autonomous components. The effectiveness of the implemented MAS is therefore more important than before.

Implementing GHUUD generation, sorting and comparison methods into Multi Agent systems leads to simplification of the Urban Search and Rescue (USAR) domains and Geospatial Multi agent Systems (GMAS) which rely upon contributing agents for Multi agent Path Planning (MAPP) [9].

Standard algorithms such as A* algorithm, E-GAP, Potential Field method (PFM) and CPAD algorithm as stated in [10] could be simplified with the introduction of GHUUID identifier.

The GHUUID method is not limited only to Geo tagged objects, static or dynamic ones; it can be used in the systems without particular Geo referential system with conversion of referential spatial systems, such as two dimensional Cartesian coordinate system, into the Geo datum enabled WGS84 or country specific geo datum.

Using our truly global identifier for data/object Meta tagging can simplify the MAS structure, give the industrial-strength to agent interaction, enhance the visualization development and predict the other agent's actions under different circumstances.

GHUUID enabled agents could perform autonomous data identifiers generation, filtering and enable smarter data flow over the Internet and private networks, obtaining only the data that are closer thus more connected.

7 References

1. Durfee, E. 1996. Planning in Distributed Artificial Intelligence. In Foundations of distributed Artificial Intelligence, eds. G. M. P. O'Hare and N. R. Jennings, 231–246. New York: Wiley.
2. Durfee, E. H., and Lesser, V. 1989. Negotiating Task Decomposition and Allocation Using Partial Global Planning. In Distributed Artificial Intelligence, Volume 2, eds. L. Gasser and M. Huhns, 229–244. San Francisco, Calif.: Morgan Kaufmann.
3. Chaimontree S., Atkinson K. and Coenen F. 2010. Clustering in a Multi-Agent Data Mining Environment
4. W. Contributors, "Geohash," Wikipedia.org, 2011
5. http://geohash.org/site/tips.html
6. http://www.ietf.org/rfc/rfc4122.txt - A Universally Unique IDentifier (UUID) URN Namespace
7. http://dev.mysql.com/doc/refman/5.0/en/miscellaneous-functions.html#function_uuid – MySQL UUID support function

8. F. Bellifemine, G. Caire, A. Poggi and G. Rimassa. JADE: a Software Framework for Developing Multi-Agent Applications. Lessons Learned. Information and Software Technology Journal, 50:10-21, 2008
9. Gaber H., Amin S., Salem A-B. M. 2011. Geospatial Multi-agent System for Urban Search and Rescue
10. Gaber H., Amin S., Salem A-B. M., Proc of ISDA, 2010. A Combined Coordination Technique for Multi-Agent Path Planning