



US 20070198597A1

(19) **United States**

(12) **Patent Application Publication**

**Betz et al.**

(10) **Pub. No.: US 2007/0198597 A1**

(43) **Pub. Date: Aug. 23, 2007**

(54) **ATTRIBUTE ENTROPY AS A SIGNAL IN OBJECT NORMALIZATION**

**Publication Classification**

(76) Inventors: **Jonathan T. Betz**, Summit, NJ (US);  
**Vivek Menezes**, Jersey City, NJ (US)

(51) **Int. Cl.**  
**G06F 17/30** (2006.01)  
(52) **U.S. Cl.** ..... **707/201**

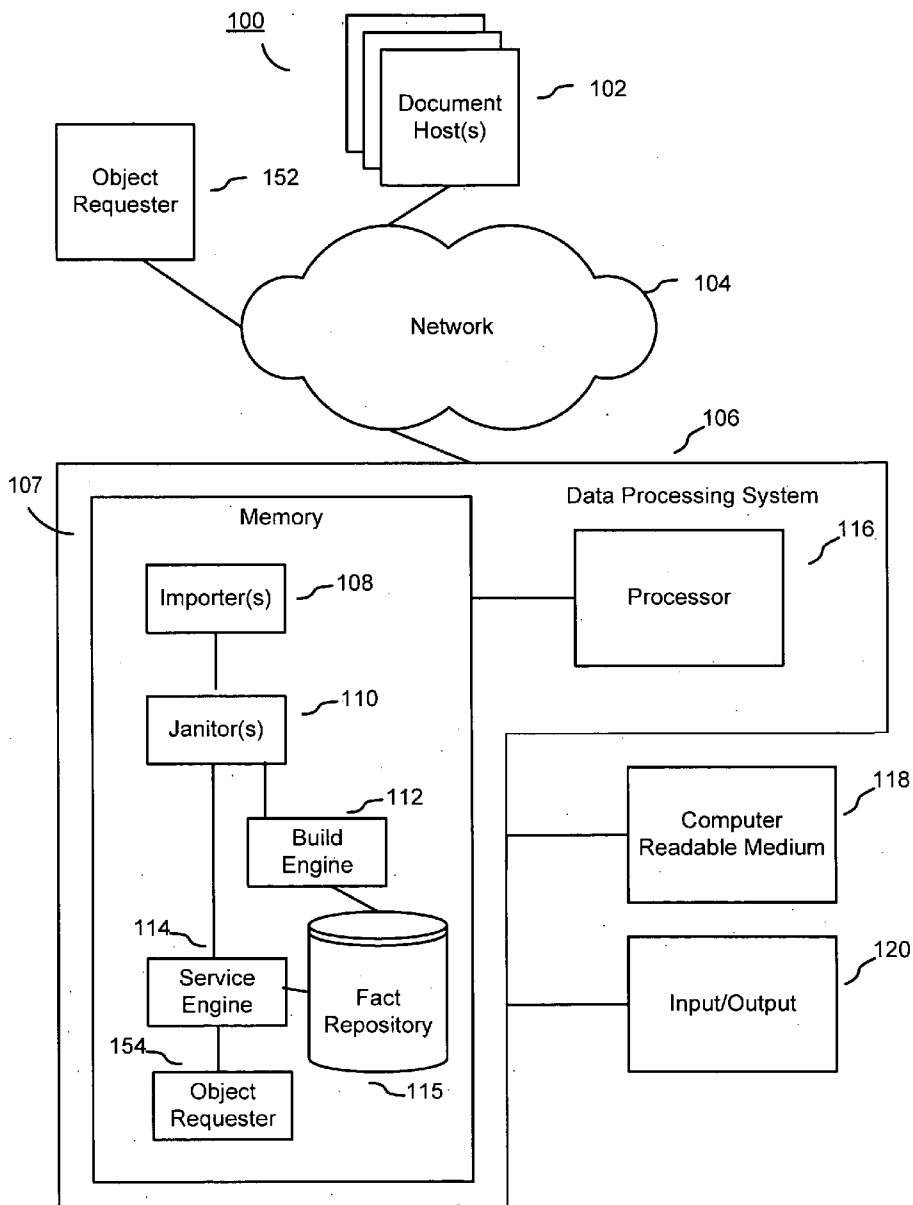
Correspondence Address:  
**GOOGLE / FENWICK**  
**SILICON VALLEY CENTER**  
**801 CALIFORNIA ST.**  
**MOUNTAIN VIEW, CA 94041 (US)**

(57) **ABSTRACT**

A system and method determines whether two objects are duplicate objects. The system and method matches common facts of the two objects based on a match measure, combines the entropies of the matching common facts, and determines whether the two objects are duplicate objects based on the sum of entropies.

(21) Appl. No.: **11/356,765**

(22) Filed: **Feb. 17, 2006**



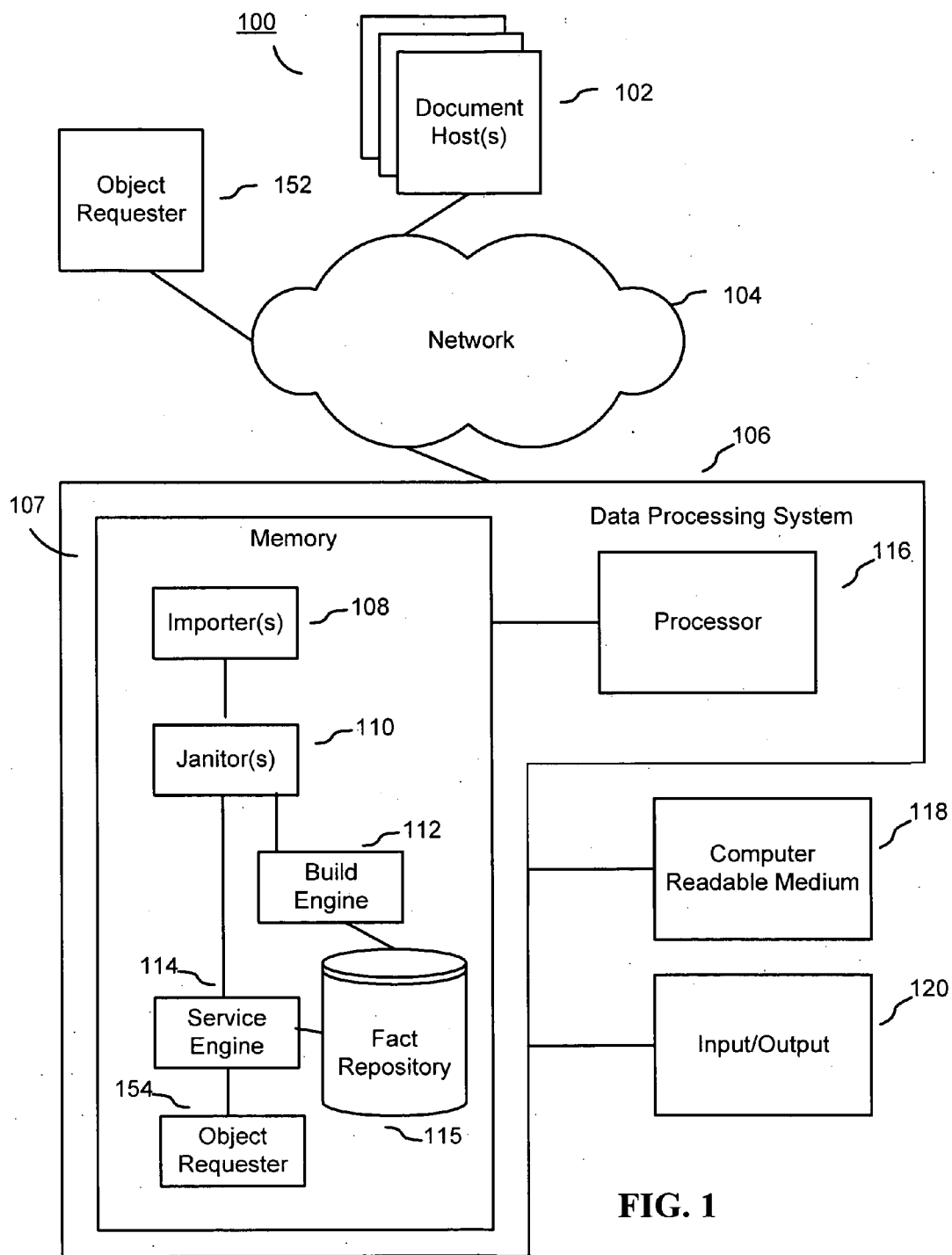
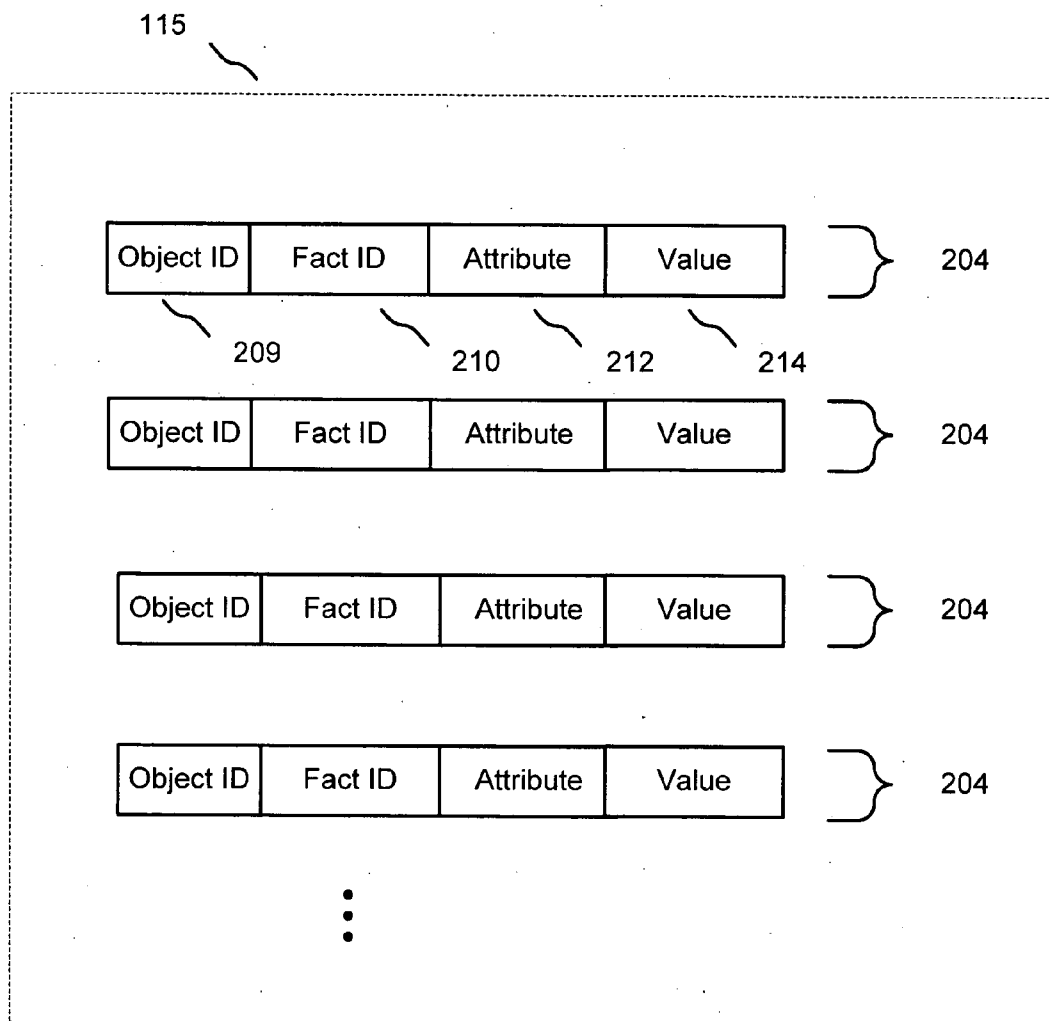
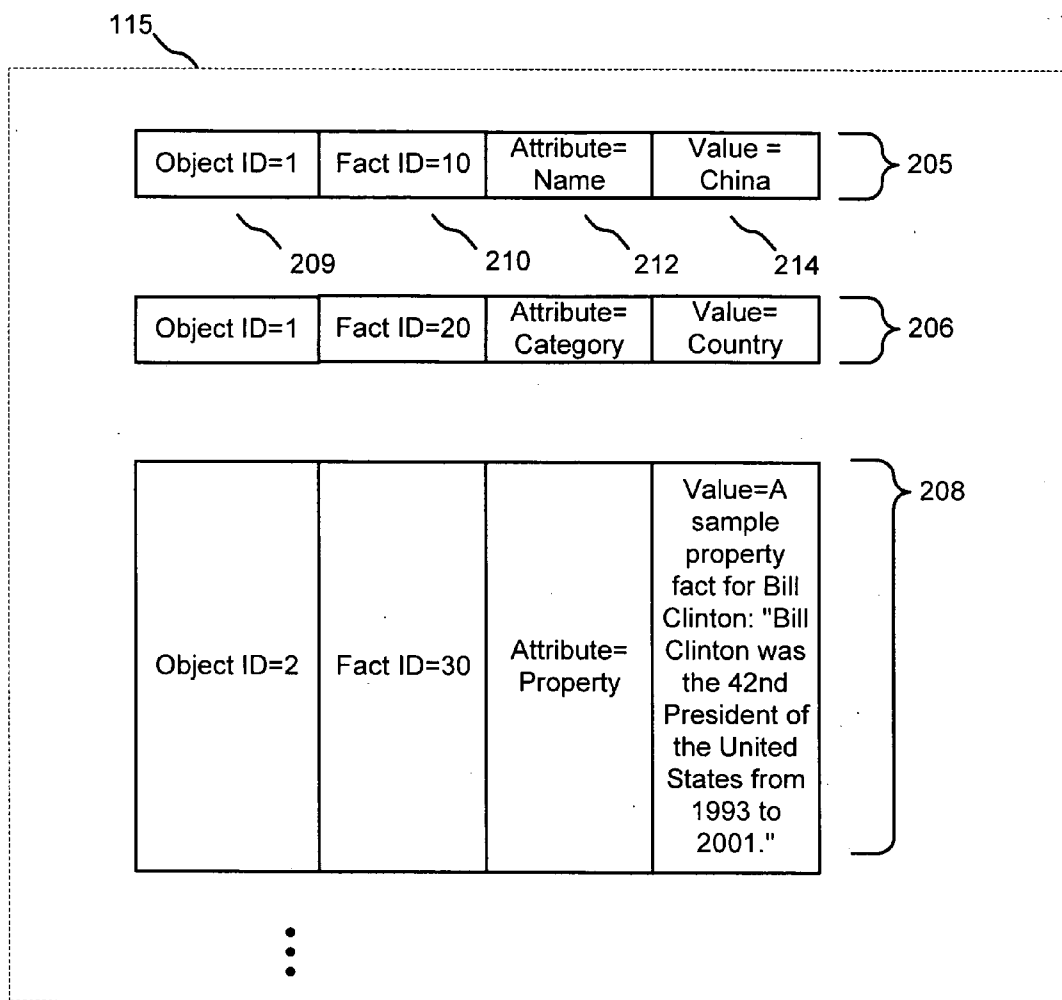


FIG. 1



**FIG. 2(a)**  
**Example Format of Facts in Repository (each fact is associated with an object ID)**



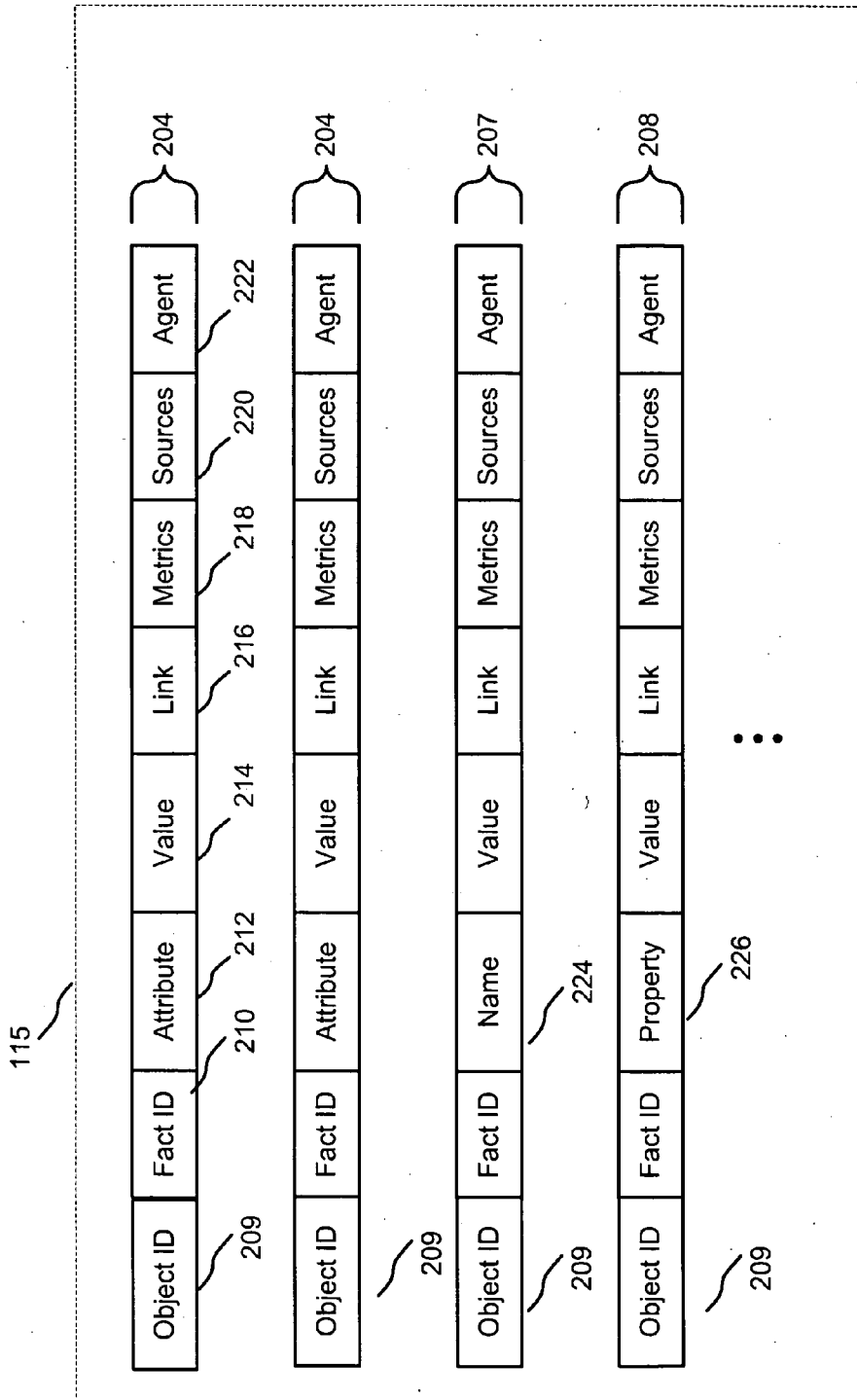
**FIG. 2(b)**  
**Example Facts in Repository (each fact is associated with an object ID)**

Object ID=1	Fact ID=10
Object ID=1	Fact ID=20
Object ID=1	Fact ID=30
Object ID=2	Fact ID=40

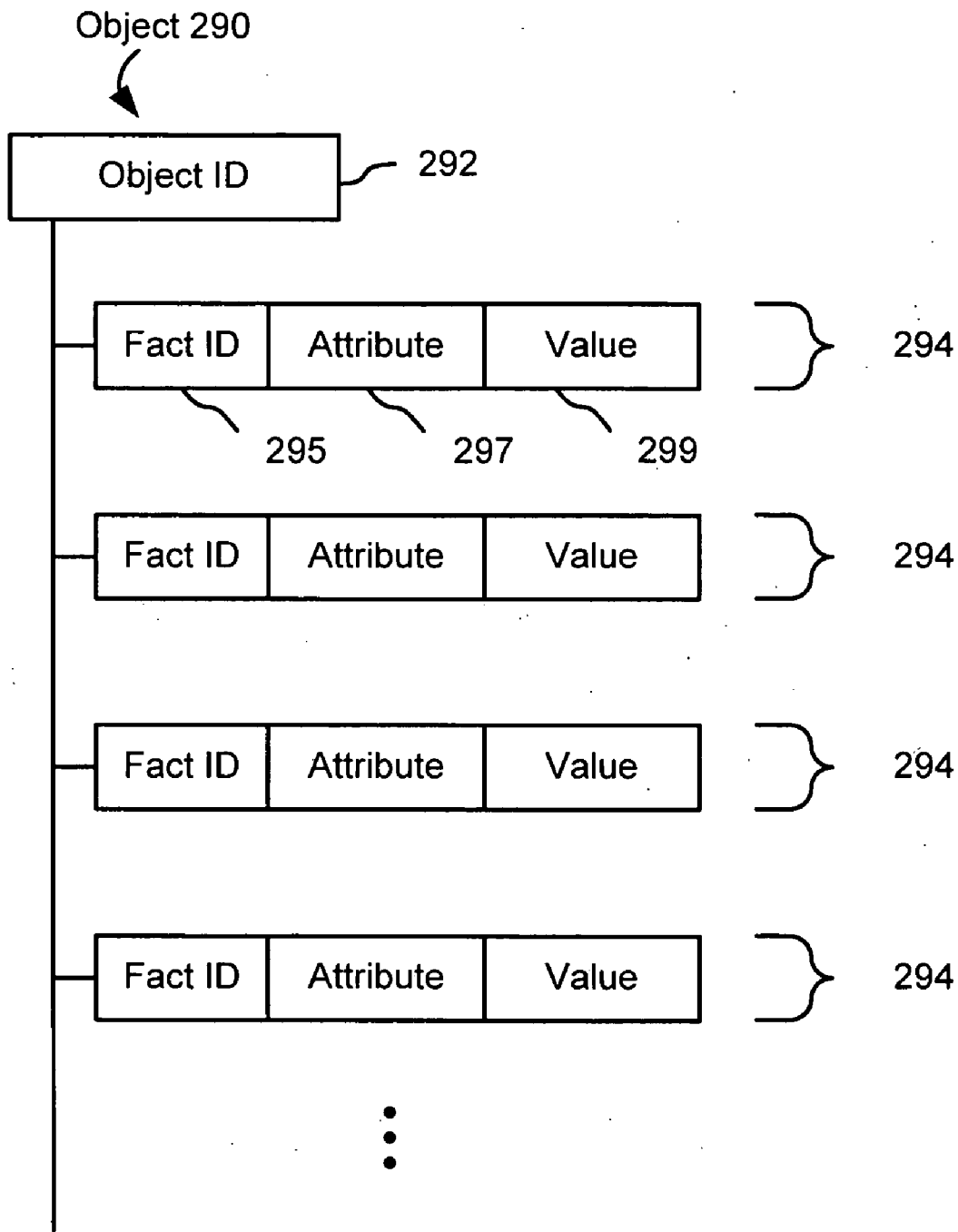
⋮

210

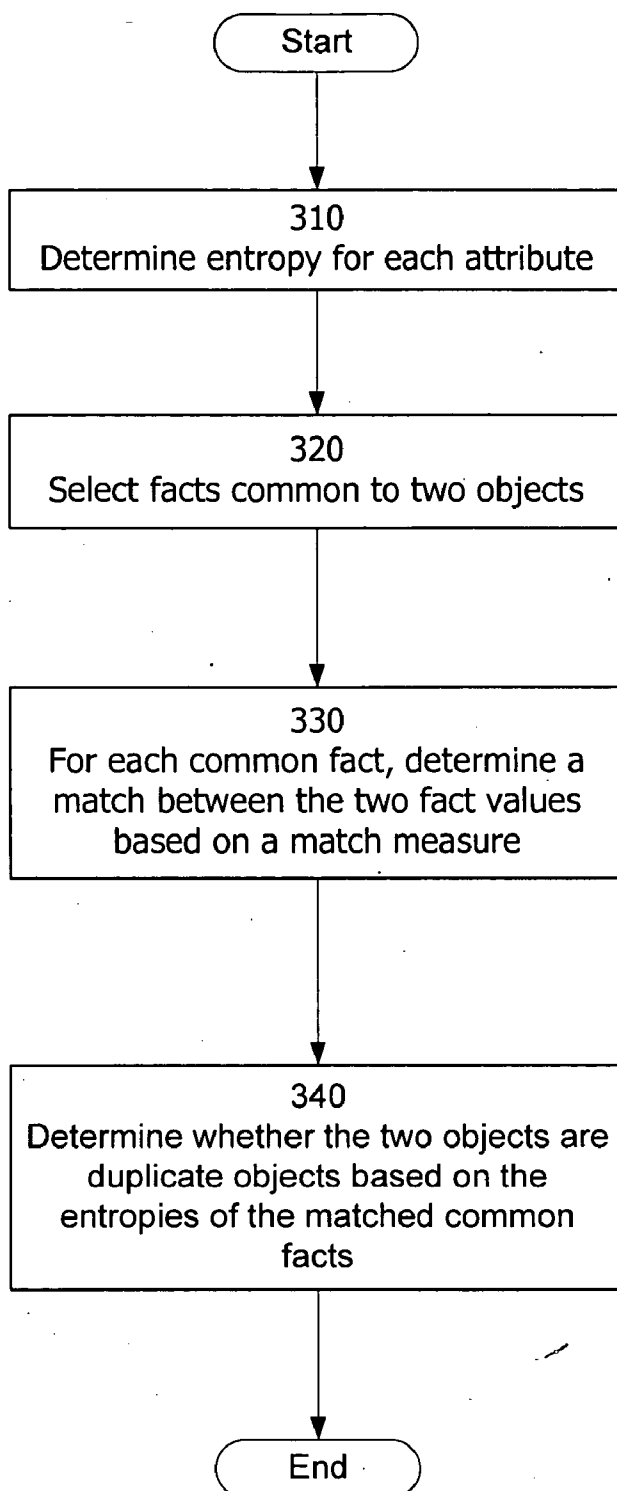
**FIG. 2(c)**  
**Example Object**  
**Reference Table**



**FIG. 2(d)**  
**Example Format of Facts in**  
**Repository (each fact is associated**  
**with an object ID)**



**FIG. 2(e)**  
**Example Objects**



**FIG. 3**

Attribute Object ID	Name	Phone Number	Type	Date of Birth	Height	Gender
O1	Joe Henry	(703) 123-4567	Human	05/18/1974	5'10"	Male
O2	J.H.	(703) 123-4567		05/18/1974	5'9.5"	

*FIG. 4(a)*

Attribute Object ID	Name	Phone Number	Type	Date of Birth	Height	Gender
O1	Joe Henry	(703) 123-4567	Human	05/18/1974	5'10"	Male
O3	John Henry		Horse	03/09/1976	5'4"	Male

*FIG. 4(b)*

Attribute	Name	Phone Number	Type	Date of Birth	Height	Gender
Entropy	5.26	6.32	2.17	5.81	3.04	0.81

*FIG. 4(c)*

**ATTRIBUTE ENTROPY AS A SIGNAL IN OBJECT NORMALIZATION**

**CROSS-REFERENCE TO RELATED PATENT APPLICATIONS**

[0001] This application is related to the following U.S. Applications all of which are incorporated by reference herein:

[0002] U.S. application Ser. No. \_\_\_\_\_, entitled "Support for Object Search", filed concurrently herewith, by Alex Kehlenbeck, Andrew W. Hogue, Jonathan T. Betz, Attorney Docket No. 24207-10945;

[0003] U.S. application Ser. No. \_\_\_\_\_, entitled "Data Object Visualization", filed on Jan. 27, 2006, by Andrew W. Hogue, David Vespe, Alex Kehlenbeck, Mike Gordon, Jeffrey C. Reynar, David Alpert, Attorney Docket No. 24207-10946;

[0004] U.S. application Ser. No. \_\_\_\_\_, entitled "Data Object Visualization Using Maps", filed on Jan. 27, 2006, by Andrew W. Hogue, David Vespe, Alex Kehlenbeck, Mike Gordon, Jeffrey C. Reynar, David Alpert, Attorney Docket No. 24207-10947;

[0005] U.S. application Ser. No. \_\_\_\_\_, entitled "Query Language", filed concurrently herewith, by Andrew W. Hogue, Doug Rhode, Attorney Docket No. 24207-10948;

[0006] U.S. application Ser. No. \_\_\_\_\_, entitled "Automatic Object Reference Identification and Linking in a Browseable Fact Repository", filed concurrently herewith, by Andrew W. Hogue, Attorney Docket No. 24207-10961;

[0007] U.S. application Ser. No. \_\_\_\_\_, entitled "Browseable Fact Repository", filed concurrently herewith, by Andrew W. Hogue, Jonathan T. Betz, Attorney Docket No. 24207-10949;

[0008] U.S. application Ser. No. \_\_\_\_\_, entitled "ID Persistence Through Normalization", filed concurrently herewith, by Jonathan T. Betz, Andrew W. Hogue, Attorney Docket No. 24207-10950;

[0009] U.S. application Ser. No. \_\_\_\_\_, entitled "Annotation Framework", filed concurrently herewith, by Tom Richford, Jonathan T. Betz, Attorney Docket No. 24207-10951;

[0010] U.S. application Ser. No. \_\_\_\_\_, entitled "Object Categorization for Information Extraction", filed on Jan. 27, 2006, by Jonathan T. Betz, Attorney Docket No. 24207-10952;

[0011] U.S. application Ser. No. \_\_\_\_\_, entitled "Modular Architecture for Entity Normalization", filed concurrently herewith, by Jonathan T. Betz, Farhan Shamsi, Attorney Docket No. 24207-10953;

[0012] U.S. application Ser. No. \_\_\_\_\_, entitled "Designating Data Objects for Analysis", filed on Jan. 27, 2006, by Andrew W. Hogue, David Vespe, Alex Kehlenbeck, Mike Gordon, Jeffrey C. Reynar, David Alpert, Attorney Docket No. 24207-11126;

[0013] U.S. application Ser. No. \_\_\_\_\_, entitled "Data Object Visualization Using Graphs", filed on Jan. 27, 2006,

by Andrew W. Hogue, David Vespe, Alex Kehlenbeck, Mike Gordon, Jeffrey C. Reynar, David Alpert, Attorney Docket No. 24207-11125.

**TECHNICAL FIELD**

[0014] The disclosed embodiments relate generally to fact databases. More particularly, the disclosed embodiments relate to determining whether two objects are duplicate objects.

**BACKGROUND**

[0015] Data is often organized as large collections of objects. When the objects are added over time, there are often problems with data duplication. For example, a collection may include multiple objects that represent the same entity. Objects are duplicate objects if they represent the same entity, even if the information about the entity contained in the objects is different. Duplicate objects increase storage cost, take longer time to process, and confuse the display of information to the user. Duplicate objects can also lead to inaccurate results, such as an inaccurate count of distinct objects.

[0016] Some applications determine whether two objects are duplicate objects by comparing the value of a specific fact, such as the Social Security Number (SSN), the International Standard Book Number (ISBN), or the Universal Product Code (UPC). This approach is effective when all objects contain one of such facts. The specific facts used for comparison are analogous to the primary keys of database tables in relational databases. But for objects built on incomplete information, some objects may not have any of these facts. Also, when values of such facts associated with either of the two objects are inaccurate, this approach treats the two objects as distinct objects even if other facts associated with the two objects indicate that they are duplicate objects. Thus, this approach only determines whether two objects are duplicate objects when both objects include accurate and complete information.

[0017] Some other applications identify whether two objects are duplicate objects by comparing all common facts of the two objects. The two objects are determined to be duplicate objects when the number of matching common facts exceeds a threshold. This approach is problematic because it does not always give accurate results. For example, the chance that two objects sharing the same gender being duplicates are much lower than that of two objects sharing the same date of birth. By treating all facts equally, this approach is both over-inclusive by identifying distinct objects sharing many facts with little indicating value, and under-inclusive by excluding duplicate objects sharing few facts with great indicating value.

[0018] For these reasons, what is needed is a method and system that determines whether two objects built from imperfect information are duplicate objects.

**SUMMARY**

[0019] One method for determining whether two objects are duplicate objects is as follows. Common facts of the two objects are identified. Common facts of the two objects are the facts with the same attribute that are associated with the two objects. Values of the common facts are compared to

identify matching common facts. Two objects have a matching common fact if they have a fact with the same attribute and the same value. Entropy of the matching common facts are combined (e.g., added) and compared with an entropy threshold. Objects with a sum of entropies exceeding the entropy threshold are determined to be duplicate objects.

[0020] These features are not the only features of the invention. In view of the drawings, specification, and claims, many additional features and advantages will be apparent.

#### BRIEF DESCRIPTION OF THE DRAWINGS

[0021] FIG. 1 shows a network, in accordance with a preferred embodiment of the invention.

[0022] FIGS. 2(a)-2(d) are block diagrams illustrating a data structure for facts within a repository of FIG. 1 in accordance with preferred embodiments of the invention.

[0023] FIG. 2(e) is a block diagram illustrating an alternate data structure for facts and objects in accordance with preferred embodiments of the invention.

[0024] FIG. 3 is a flowchart of an exemplary method for determining whether two objects are duplicate objects, in accordance with a preferred embodiment of the invention.

[0025] FIGS. 4(a)-(c) illustrate examples of determining whether two objects are duplicate objects, in accordance with a preferred embodiment of the invention.

[0026] The figures depict various embodiments of the present invention for purposes of illustration only. One skilled in the art will readily recognize from the following discussion that alternative embodiments of the structures and methods illustrated herein may be employed without departing from the principles of the invention described herein.

#### DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

##### System Architecture

[0027] FIG. 1 shows a system architecture 100 adapted to support one embodiment of the invention. FIG. 1 shows components used to add facts into, and retrieve facts from a repository 115. The system architecture 100 includes a network 104, through which any number of document hosts 102 communicate with a data processing system 106, along with any number of object requesters 152, 154.

[0028] Document hosts 102 store documents and provide access to documents. A document is comprised of any machine-readable data including any combination of text, graphics, multimedia content, etc. A document may be encoded in a markup language, such as Hypertext Markup Language (HTML), i.e., a web page, in an interpreted language (e.g., JavaScript) or in any other computer readable or executable format. A document can include one or more hyperlinks to other documents. A typical document will include one or more facts within its content. A document stored in a document host 102 may be located and/or identified by a Uniform Resource Locator (URL), or Web address, or any other appropriate form of identification and/or location. A document host 102 is implemented by a computer system, and typically includes a server adapted to communicate over the network 104 via networking proto-

cols (e.g., TCP/IP), as well as application and presentation protocols (e.g., HTTP, HTML, SOAP, D-HTML, Java). The documents stored by a host 102 are typically held in a file directory, a database, or other data repository. A host 102 can be implemented in any computing device (e.g., from a PDA or personal computer, a workstation, mini-computer, or mainframe, to a cluster or grid of computers), as well as in any processor architecture or operating system.

[0029] FIG. 1 shows components used to manage facts in a fact repository 115. Data processing system 106 includes one or more importers 108, one or more janitors 110, a build engine 112, a service engine 114, and a fact repository 115 (also called simply a "repository"). Each of the foregoing are implemented, in one embodiment, as software modules (or programs) executed by processor 116. Importers 108 operate to process documents received from the document hosts, read the data content of documents, and extract facts (as operationally and programmatically defined within the data processing system 106) from such documents. The importers 108 also determine the subject or subjects with which the facts are associated, and extract such facts into individual items of data, for storage in the fact repository 115. In one embodiment, there are different types of importers 108 for different types of documents, for example, dependent on the format or document type.

[0030] Janitors 110 operate to process facts extracted by importer 108. This processing can include but is not limited to, data cleansing, object merging, and fact induction. In one embodiment, there are a number of different janitors 110 that perform different types of data management operations on the facts. For example, one janitor 110 may traverse some set of facts in the repository 115 to find duplicate facts (that is, facts that convey the same factual information) and merge them. Another janitor 110 may also normalize facts into standard formats. Another janitor 110 may also remove unwanted facts from repository 115, such as facts related to pornographic content. Other types of janitors 110 may be implemented, depending on the types of data management functions desired, such as translation, compression, spelling or grammar correction, and the like.

[0031] Various janitors 110 act on facts to normalize attribute names, and values and delete duplicate and near-duplicate facts so an object does not have redundant information. For example, we might find on one page that Britney Spears' birthday is "12/2/1981" while on another page that her date of birth is "Dec. 2, 1981." Birthday and Date of Birth might both be rewritten as Birthdate by one janitor and then another janitor might notice that 12/2/1981 and Dec. 2, 1981 are different forms of the same date. It would choose the preferred form, remove the other fact and combine the source lists for the two facts. As a result when you look at the source pages for this fact, on some you'll find an exact match of the fact and on others text that is considered to be synonymous with the fact.

[0032] Build engine 112 builds and manages the repository 115. Service engine 114 is an interface for querying the repository 115. Service engine 114's main function is to process queries, score matching objects, and return them to the caller but it is also used by janitor 110.

[0033] Repository 115 stores factual information extracted from a plurality of documents that are located on document hosts 102. A document from which a particular fact may be

extracted is a source document (or “source”) of that particular fact. In other words, a source of a fact includes that fact (or a synonymous fact) within its contents.

[0034] Repository **115** contains one or more facts. In one embodiment, each fact is associated with exactly one object. One implementation for this association includes in each fact an object ID that uniquely identifies the object of the association. In this manner, any number of facts may be associated with an individual object, by including the object ID for that object in the facts. In one embodiment, objects themselves are not physically stored in the repository **115**, but rather are defined by the set or group of facts with the same associated object ID, as described below. Further details about facts in repository **115** are described below, in relation to FIGS. 2(a)-2(d).

[0035] It should be appreciated that in practice at least some of the components of the data processing system **106** will be distributed over multiple computers, communicating over a network. For example, repository **115** may be deployed over multiple servers. As another example, the janitors **110** may be located on any number of different computers. For convenience of explanation, however, the components of the data processing system **106** are discussed as though they were implemented on a single computer.

[0036] In another embodiment, some or all of document hosts **102** are located on data processing system **106** instead of being coupled to data processing system **106** by a network. For example, importer **108** may import facts from a database that is a part of or associated with data processing system **106**.

[0037] FIG. 1 also includes components to access repository **115** on behalf of one or more object requesters **152**, **154**. Object requesters are entities that request objects from repository **115**. Object requesters **152**, **154** may be understood as clients of the system **106**, and can be implemented in any computer device or architecture. As shown in FIG. 1, a first object requester **152** is located remotely from system **106**, while a second object requester **154** is located in data processing system **106**. For example, in a computer system hosting a blog, the blog may include a reference to an object whose facts are in repository **115**. An object requester **152**, such as a browser displaying the blog will access data processing system **106** so that the information of the facts associated with the object can be displayed as part of the blog web page. As a second example, janitor **110** or other entity considered to be part of data processing system **106** can function as object requester **154**, requesting the facts of objects from repository **115**.

[0038] FIG. 1 shows that data processing system **106** includes a memory **107** and one or more processors **116**. Memory **107** includes importers **108**, janitors **110**, build engine **112**, service engine **114**, and requester **154**, each of which are preferably implemented as instructions stored in memory **107** and executable by processor **116**. Memory **107** also includes repository **115**. Repository **115** can be stored in a memory of one or more computer systems or in a type of memory such as a disk. FIG. 1 also includes a computer readable medium **118** containing, for example, at least one of importers **108**, janitors **110**, build engine **112**, service engine **114**, requester **154**, and at least some portions of repository **115**. FIG. 1 also includes one or more input/output devices **120** that allow data to be input and output to

and from data processing system **106**. It will be understood that data processing system **106** preferably also includes standard software components such as operating systems and the like and further preferably includes standard hardware components not shown in the figure for clarity of example.

#### Data Structure

[0039] FIG. 2(a) shows an example format of a data structure for facts within repository **115**, according to some embodiments of the invention. As described above, the repository **115** includes facts **204**. Each fact **204** includes a unique identifier for that fact, such as a fact ID **210**. Each fact **204** includes at least an attribute **212** and a value **214**. For example, a fact associated with an object representing George Washington may include an attribute of “date of birth” and a value of “Feb. 22, 1732.” In one embodiment, all facts are stored as alphanumeric characters since they are extracted from web pages. In another embodiment, facts also can store binary data values. Other embodiments, however, may store fact values as mixed types, or in encoded formats.

[0040] As described above, each fact is associated with an object ID **209** that identifies the object that the fact describes. Thus, each fact that is associated with a same entity (such as George Washington), will have the same object ID **209**. In one embodiment, objects are not stored as separate data entities in memory. In this embodiment, the facts associated with an object contain the same object ID, but no physical object exists. In another embodiment, objects are stored as data entities in memory, and include references (for example, pointers or IDs) to the facts associated with the object. The logical data structure of a fact can take various forms; in general, a fact is represented by a tuple that includes a fact ID, an attribute, a value, and an object ID. The storage implementation of a fact can be in any underlying physical data structure.

[0041] FIG. 2(b) shows an example of facts having respective fact IDs of 10, 20, and 30 in repository **115**. Facts **10** and **20** are associated with an object identified by object ID “1.” Fact **10** has an attribute of “Name” and a value of “China.” Fact **20** has an attribute of “Category” and a value of “Country.” Thus, the object identified by object ID “1” has a name fact **205** with a value of “China” and a category fact **206** with a value of “Country.” Fact **30208** has an attribute of “Property” and a value of “Bill Clinton was the 42nd President of the United States from 1993 to 2001.” Thus, the object identified by object ID “2” has a property fact with a fact ID of 30 and a value of “Bill Clinton was the 42nd President of the United States from 1993 to 2001.” In the illustrated embodiment, each fact has one attribute and one value. The number of facts associated with an object is not limited; thus while only two facts are shown for the “China” object, in practice there may be dozens, even hundreds of facts associated with a given object. Also, the value fields of a fact need not be limited in size or content. For example, a fact about the economy of “China” with an attribute of “Economy” would have a value including several paragraphs of text, numbers, perhaps even tables of figures. This content can be formatted, for example, in a markup language. For example, a fact having an attribute “original html” might have a value of the original html text taken from the source web page.

[0042] Also, while the illustration of FIG. 2(b) shows the explicit coding of object ID, fact ID, attribute, and value, in

practice the content of the fact can be implicitly coded as well (e.g., the first field being the object ID, the second field being the fact ID, the third field being the attribute, and the fourth field being the value). Other fields include but are not limited to: the language used to state the fact (English, etc.), how important the fact is, the source of the fact, a confidence value for the fact, and so on.

[0043] FIG. 2(c) shows an example object reference table 210 that is used in some embodiments. Not all embodiments include an object reference table. The object reference table 210 functions to efficiently maintain the associations between object IDs and fact IDs. In the absence of an object reference table 210, it is also possible to find all facts for a given object ID by querying the repository to find all facts with a particular object ID. While FIGS. 2(b) and 2(c) illustrate the object reference table 210 with explicit coding of object and fact IDs, the table also may contain just the ID values themselves in column or pair-wise arrangements.

[0044] FIG. 2(d) shows an example of a data structure for facts within repository 115, according to some embodiments of the invention showing an extended format of facts. In this example, the fields include an object reference link 216 to another object. The object reference link 216 can be an object ID of another object in the repository 115, or a reference to the location (e.g., table row) for the object in the object reference table 210. The object reference link 216 allows facts to have as values other objects. For example, for an object "United States," there may be a fact with the attribute of "president" and the value of "George W. Bush," with "George W. Bush" being an object having its own facts in repository 115. In some embodiments, the value field 214 stores the name of the linked object and the link 216 stores the object identifier of the linked object. Thus, this "president" fact would include the value 214 of "George W. Bush", and object reference link 216 that contains the object ID for the "George W. Bush" object. In some other embodiments, facts 204 do not include a link field 216 because the value 214 of a fact 204 may store a link to another object.

[0045] Each fact 204 also may include one or more metrics 218. A metric provides an indication of the some quality of the fact. In some embodiments, the metrics include a confidence level and an importance level. The confidence level indicates the likelihood that the fact is correct. The importance level indicates the relevance of the fact to the object, compared to other facts for the same object. The importance level may optionally be viewed as a measure of how vital a fact is to an understanding of the entity or concept represented by the object.

[0046] Each fact 204 includes a list of one or more sources 220 that include the fact and from which the fact was extracted. Each source may be identified by a Uniform Resource Locator (URL), or Web address, or any other appropriate form of identification and/or location, such as a unique document identifier.

[0047] The facts illustrated in FIG. 2(d) include an agent field 222 that identifies the importer 108 that extracted the fact. For example, the importer 108 may be a specialized importer that extracts facts from a specific source (e.g., the pages of a particular web site, or family of web sites) or type of source (e.g., web pages that present factual information in tabular form), or an importer 108 that extracts facts from free text in documents throughout the Web, and so forth.

[0048] Some embodiments include one or more specialized facts, such as a name fact 207 and a property fact 208. A name fact 207 is a fact that conveys a name for the entity or concept represented by the object ID. A name fact 207 includes an attribute 224 of "name" and a value, which is the name of the object. For example, for an object representing the country Spain, a name fact would have the value "Spain." A name fact 207, being a special instance of a general fact 204, includes the same fields as any other fact 204; it has an attribute, a value, a fact ID, metrics, sources, etc. The attribute 224 of a name fact 207 indicates that the fact is a name fact, and the value is the actual name. The name may be a string of characters. An object ID may have one or more associated name facts, as many entities or concepts can have more than one name. For example, an object ID representing Spain may have associated name facts conveying the country's common name "Spain" and the official name "Kingdom of Spain." As another example, an object ID representing the U.S. Patent and Trademark Office may have associated name facts conveying the agency's acronyms "PTO" and "USPTO" as well as the official name "United States Patent and Trademark Office." If an object does have more than one associated name fact, one of the name facts may be designated as a primary name and other name facts may be designated as secondary names, either implicitly or explicitly.

[0049] A property fact 208 is a fact that conveys a statement about the entity or concept represented by the object ID. Property facts are generally used for summary information about an object. A property fact 208, being a special instance of a general fact 204, also includes the same parameters (such as attribute, value, fact ID, etc.) as other facts 204. The attribute field 226 of a property fact 208 indicates that the fact is a property fact (e.g., attribute is "property") and the value is a string of text that conveys the statement of interest. For example, for the object ID representing Bill Clinton, the value of a property fact may be the text string "Bill Clinton was the 42nd President of the United States from 1993 to 2001." Some object IDs may have one or more associated property facts while other objects may have no associated property facts. It should be appreciated that the data structures shown in FIGS. 2(a)-2(d) and described above are merely exemplary. The data structure of the repository 115 may take on other forms. Other fields may be included in facts and some of the fields described above may be omitted. Additionally, each object ID may have additional special facts aside from name facts and property facts, such as facts conveying a type or category (for example, person, place, movie, actor, organization, etc.) for categorizing the entity or concept represented by the object ID. In some embodiments, an object's name(s) and/or properties may be represented by special records that have a different format than the general facts records 204.

[0050] As described previously, a collection of facts is associated with an object ID of an object. An object may become a null or empty object when facts are disassociated from the object. A null object can arise in a number of different ways. One type of null object is an object that has had all of its facts (including name facts) removed, leaving no facts associated with its object ID. Another type of null object is an object that has all of its associated facts other than name facts removed, leaving only its name fact(s). Alternatively, the object may be a null object only if all of its associated name facts are removed. A null object repre-

sents an entity or concept for which the data processing system 106 has no factual information and, as far as the data processing system 106 is concerned, does not exist. In some embodiments, facts of a null object may be left in the repository 115, but have their object ID values cleared (or have their importance to a negative value). However, the facts of the null object are treated as if they were removed from the repository 115. In some other embodiments, facts of null objects are physically removed from repository 115.

[0051] FIG. 2(e) is a block diagram illustrating an alternate data structure 290 for facts and objects in accordance with preferred embodiments of the invention. In this data structure, an object 290 contains an object ID 292 and references or points to facts 294. Each fact includes a fact ID 295, an attribute 297, and a value 299. In this embodiment, an object 290 actually exists in memory 107.

Overview of Methodology

[0052] An entropy is a numerical measure of the amount of information carried by a variable; the more information the variable carries, the more random the variable, and the greater the entropy. Variables with a narrower range of possible values tend to have lower entropies. Variables whose values are not distributed evenly within the possible values tend to have lower entropies.

[0053] An attribute with low entropy will be one which has a low number of possible values or has many possible values but only a few them occur with any frequency. If an attribute has low entropy, the fact that two objects have the same value for that attribute provides little information about whether the two objects are the same entity. An attribute with a higher entropy is more informative than one with a lower entropy because the value of the attribute is more likely to distinguish an object. In other words, a high entropy means the associated attribute is highly useful for distinguishing among objects. As a result, attribute entropy (also called simply entropy) is a good indicator of the importance of matching common facts when determining whether two objects are duplicate objects.

[0054] For example, the chance that two objects of type human are identical if they share the same gender attribute is much less likely than if they share the same date of birth attribute. Also note that the entropy might be refined to be calculated based on the type of object—in other words, the weight attribute may not be useful for distinguishing among human objects but it may be useful for distinguishing among planet objects.

[0055] Claude E. Shannon defines entropy in terms of a discrete random event x, with possible states 1 . . . n as:

$$H(x) = \sum_{i=1}^n p(i) \log_2 \left( \frac{1}{p(i)} \right) = - \sum_{i=1}^n p(i) \log_2 p(i).$$

[0056] That is, the entropy of the event x is the sum, over all possible outcomes i of x, of the product of the probability of outcome i times the log of the probability of i.

[0057] In one embodiment, the janitor 110 uses entropy to identify duplicate objects so that the duplicate objects can be merged together.

[0058] Referring to FIG. 3, there is shown a flowchart of an exemplary method for determining whether two objects are duplicates according to one embodiment of the present invention. The method illustrated in FIG. 3 may be implemented in software, hardware, or a combination of hardware and software.

[0059] The flowchart shown in FIG. 3 will now be described in detail, illustrated by the example in FIGS. 4(a)-(c). The process commences with a set of objects, each having a set of facts including an attribute and a value (also called fact value). The set of objects are retrieved by the janitor 110 from the repository 115. In one embodiment, the janitor 110 retrieves the set of objects by asking the service engine 114 for the information stored in the repository 115. Depending how object information is stored in the repository 115, the janitor 110 needs to reconstruct the objects based on the facts and object information retrieved.

[0060] The janitor 110 determines 310 the entropy for each attribute, and saves the entropy in one of the metrics 218 of each fact with the attribute. In one embodiment, the janitor 110 asks the service engine 114 for all facts stored in the repository 115, and saves the attribute and value of each fact in a table. For each attribute, the janitor 110 calculates the associated entropy by applying the Shannon formula illustrated above to all values associated with the attribute in the table. For example, assuming there are four facts with attribute gender, three of them have value male, and one has value female, the entropy of the attribute gender will be:  $H(\text{gender}) = p(\text{male}) \log_2(1/p(\text{male})) + p(\text{female}) \log_2(1/p(\text{female})) = 75\% * \log_2(4/3) + 25\% * \log_2 4 \approx 0.81$ . The janitor 110 then stores the entropy 0.81 in one of the metrics 218 of each of the four facts with attribute gender. Alternatively, the entropy can be calculated using logarithm with bases other than two. In some embodiments, the janitor 110 recomputes the entropy for each attribute every time the repository 115 is modified.

[0061] Other embodiments may use other measures of entropy. In some embodiments, entropy is calculated based on the number of possible values that the associated attribute could hypothetically have, while in some other embodiments, entropy is calculated based on the number of actual fact values of the associated attribute. In some embodiments, certain attributes are not given an entropy. In some other embodiments, certain attributes are given a predetermined weight in stead of a calculated entropy.

[0062] For two of the set of objects, the janitor 110 selects 320 common facts of the two objects. Common facts are facts sharing the same attribute. Objects having matching fact values for common fact might be duplicates. An example of the two objects is shown in FIG. 4(a), another example is shown in FIG. 4(b).

[0063] As shown in FIG. 4(a), objects O1 and O2 are duplicate objects representing the same entity, a Mr. Joe M. Henry with nickname J. H. O1 is associated with six facts with the following attributes: name, phone number, type, date of birth, height, and gender. O2 is associated with four facts: name, phone number, date of birth, and height. Object O1 in FIG. 4(b) is the same object as the object O1 shown in FIG. 4(a). Object O3 in FIG. 4(b) represents a race horse named John Henry. O3 is associated with five facts: name, type, date of birth, height, and gender. Among the facts associated with the objects, there are considerable variations

in the associated attributes and values. One example of the entropy of each attribute in FIGS. 4(a)-(b) is illustrated in FIG. 4(c).

[0064] The common facts of objects O1 and O2 as illustrated in FIG. 4(a) are the facts with the following attributes: name, phone number, date of birth, and height. The common facts of objects O1 and O3 in FIG. 4(b) are the facts with the following attributes: name, type, date of birth, height, and gender.

[0065] For each common fact associated with the two objects, the janitor 110 determines 330 a match of the fact value based on a match measure. The match measure is designed to distinguish fact values that can be treated as equivalent from fact values that are essentially different. In one example, the match measure requires the fact values to be identical in order to be a match. In another example, two fact values can be determined to match by the match measure when they are lexically equivalent, such as "U.S.A." and "United States." Alternatively, the match measure can be a fuzzy match based on string or tuple similarity measures (e.g., edit distance, Hamming Distance, Levenshtein Distance, Smith-Waterman Distance, Gotoh Distance, Jaro Distance Metric, Dice's Coefficient, Jaccard Coefficient to name a few).

[0066] One example of a fuzzy match measure is based on the edit distance between the fact values of the common fact. The edit distance is the minimum number of character insertion, deletion, or substitution needed to transform one fact value into the other. This approach is advantageous because typographical errors can be filtered out by the fuzzy match.

[0067] For example, values of a common fact are deemed to match when the edit distance between the two values is no more than a threshold value of 3. Fact with attribute name is a common fact of objects O1-O3 as illustrated in FIGS. 4(a)-(b). The value of fact with attribute name associated with objects O1 ("Joe Henry") and that of O3 ("John Henry") have an edit distance of two, thus the janitor 110 determines 330 values of common fact name of O1 and O3 to be matched. The value of fact with attribute name associated with object O1 ("Joe Henry") and that of O2 ("J. H.") have an edit distance of seven, and the janitor 110 determines 330 values of common fact name of O1 and O2 do not match.

[0068] In another example, the match measure treats numeric fact values within a range as matched. One such match measure deems two numeric fact values to match if the difference between either one of the two fact values and the arithmetic mean of the two fact values is no more than 1% of the arithmetic mean. Fact with attribute height is a common fact of objects O1-O3 as illustrated in FIGS. 4(a)-(b). The arithmetic mean of values of fact with attribute height associated with objects O1 and O2 is 5'9.75". Because the difference between the value of fact with attribute height associated with either objects and the mean equals 0.36% of the mean ( $(5'10"-5'9.75")/5'9.75"=(5'9.75"-5'9.5")/5'9.75"=0.25"/5'9.75"≈0.36%$ ), less than the 1% threshold, the janitor 110 determines 330 values of common fact heights of O1 and O2 to be matched. On the other hand, the arithmetic mean of values of fact with attribute height associated with objects O1 and O3 is 5'7". Because the difference between the value of fact with attribute height associated with either

objects and the mean equals 4.48% of the mean ( $((5'10"-5'7")/5'7"=(5'7"-5.4")/5'7"=3"/5'7"≈4.48%$ ), more than the 1% threshold, the janitor 110 determines 330 values of common fact heights of O1 and O3 to be not matched. Instead of using arithmetic mean, the match measure can use other measures, such as geometric mean, harmonic mean, and the like.

[0069] The janitor 110 determines 340 whether the two objects are duplicate objects by combining the entropies of matching common facts into one sum of entropies and comparing the sum of entropies with an entropy threshold measure. The entropy threshold measure is designed to separate duplicate objects from distinct objects.

[0070] For example, one entropy threshold measure deems two objects as duplicate objects if the sum of entropies is over a threshold value of 7. Assuming the matching common facts of objects O1 and O2 are facts with attributes phone number, date of birth, and height, as illustrated in FIG. 4(a), the sum of entropies would be  $6.32+5.81+3.04=15.17$ . See FIG. 4(c). Because 15.17 exceeds the entropy threshold value 7, the janitor 110 correctly determines 340 O1 and O2 to be duplicate objects. Assuming the matching common facts of objects O1 and O2 are facts with attributes name and gender, as illustrated in FIG. 4(b), the sum of entropies would be  $5.26+0.81=6.07$ , smaller than the entropy threshold value 7. See FIG. 4(c). Consequently, the janitor 110 correctly determines 340 O1 and O3 to be distinct objects.

[0071] In other embodiments, the entropy threshold measure takes into consideration the entropies of common facts that do not match. The entropy threshold measure can deem two objects as duplicate objects if the sum of entropies of matching common facts exceeds the sum of entropies of common facts that do not match for a certain threshold value, for example, a threshold value of 5. Using the example illustrated above, the sum of matching entropies of objects O1 and O2 is 15.17, and the sum of entropies of common facts that do not match is 5.26, the entropy of attribute name, as shown in FIG. 4(c). Because the difference is 9.91, exceeding the threshold value of 5, the janitor 110 correctly determines 340 O1 and O2 to be duplicate objects. The sum of matching entropies of objects O1 and O3 is 6.07, and the sum of entropies of common facts that do not match is 11.02, the sum of entropies of attributes type, date of birth, and height, as shown in FIG. 4(c). Consequently, the janitor 110 correctly determines 340 O1 and O3 to be distinct objects.

[0072] To maintain the integrity of entropies, after the janitor 110 identifies duplicate objects, the entropies of common facts of the duplicate objects need to be recomputed. When re-computing the entropies, only one fact value of the common fact associated with the duplicate objects is used. In one embodiment, the janitor 110 has access to a copy of the table created when the janitor 110 determines 310 the entropy for each attribute. For each common fact of the duplicate objects, the janitor 110 removes the entry corresponding to the common fact associated with one of the two duplicate objects from the table, calculates a new entropy by applying the Shannon formula to all remaining values associated with the attribute of the common fact in the table, and updates the entropy stored in each fact of the attribute to the new entropy.

[0073] Finally, it should be noted that the language used in the specification has been principally selected for readability

and instructional purposes, and may not have been selected to delineate or circumscribe the inventive subject matter. Accordingly, the disclosure of the present invention is intended to be illustrative, but not limiting, of the scope of the invention, which is set forth in the following claims.

What is claimed is:

1. A computer-implemented method of determining if two objects are the same using one or more facts associated with each of the objects, the method comprising:

identifying facts that are substantially the same for both of the objects;

determining the entropy of facts that are substantially the same; and

determining whether the objects are the same based at least in part on the entropies of the facts that are substantially the same.

2. The method of claim 1 wherein the facts comprise attributes and values, and wherein facts that are substantially the same are identified by comparing the attributes and values of the facts.

3. The method of claim 2, wherein the facts that are substantially the same are the facts that have the same attributes and the same value.

4. The method of claim 2, wherein the facts that are substantially the same are the facts that have the same attribute and matching values based on a match measure.

5. The method of claim 2, wherein the facts further comprise entropies, each of the entropies comprising a numeric value, the numeric value being the same among facts that have the same attributes.

6. The method of claim 1, further comprising:

calculating the entropies of the facts associated with the objects.

7. The method of claim 1, further comprising:

responsive to the objects being duplicate objects, re-calculating the entropies of the facts associated with the objects that have the same attributes, the facts associated with one of the objects being excluded when doing the re-calculation.

8. The method of claim 1, wherein determining whether the objects are the same comprising:

determining whether the objects are the same based at least in part on a combination of the entropies of the facts that are substantially the same and an entropy threshold measure.

9. The method of claim 1, wherein determining whether the objects are the same comprising:

determining whether the objects are the same based at least in part on a combination of the entropies of the facts that are substantially the same and a combination of the entropies of other facts.

10. A computer-implemented method of determining if two objects are different using one or more associated facts comprising values and attributes, the method comprising:

identifying facts associated with each of the two objects that have the same attributes but substantially different values;

determining the entropy of each of such facts; and

determining whether the objects are different based at least in part on the entropies of the facts that have the same attributes and substantially different values.

11. The method of claim 10, wherein if the facts have substantially different values are determined based on a match measure.

12. The method of claim 10, wherein the facts further comprise entropies, each of the entropies comprising a numeric value, the numeric value being the same among facts that have the same attributes.

13. The method of claim 10, further comprising:

calculating the entropies of the facts associated with the objects.

14. The method of claim 10, wherein determining whether the objects are different comprising:

determining whether the objects are different based at least in part on a combination of the entropies of the facts that have the same attributes and substantially different values and an entropy threshold measure.

15. The method of claim 10, wherein determining whether the objects are different comprising:

determining whether the objects are different based at least in part on a combination of the entropies of the facts that have the same attributes and substantially different values and a combination of the entropies of other facts.

16. A system for determining if two objects are the same using one or more facts associated with each of the objects, comprising:

a processor for executing programs; and

a subsystem executable by the processor, the engine including:

instructions for identifying facts that are substantially the same for both of the objects;

instructions for determining the entropy of facts that are substantially the same; and

instructions for determining whether the objects are the same based at least in part on the entropies of the facts that are substantially the same.

17. A computer program product for use in conjunction with a computer system, the computer program product comprising a computer readable storage medium and a computer program mechanism embedded therein, the computer program mechanism including:

instructions for identifying facts that are substantially the same for both of the objects;

instructions for determining the entropy of facts that are substantially the same; and

instructions for determining whether the objects are the same based at least in part on the entropies of the facts that are substantially the same.