(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 June 2017 (22.06.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/105508 A1

(51) International Patent Classification:
G06F 17/30 (2006.01)

(21) International Application Number:
PCT/US2015/066830

(22) International Filing Date:
18 December 2015 (18.12.2015)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP** [US/US]; 11445 Compaq Center Drive W., Houston, Texas 77070 (US).

(72) Inventors: **ESHGHI, Kave**; 1501 Page Mill Rd., Palo Alto, California 94304-1100 (US). **KAFAI, Mehran**; 1501 Page Mill Rd., Palo Alto, California 94304-1100 (US).

(74) Agents: **KIRCHEV, Ivan T.** et al.; Hewlett Packard Enterprise, 3404 E. Harmony Road, Mail Stop 79, Fort Collins, Colorado 80528 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

— *with international search report (Art. 21(3))*

WO 2017/105508 A1

(54) Title: CLUSTERING

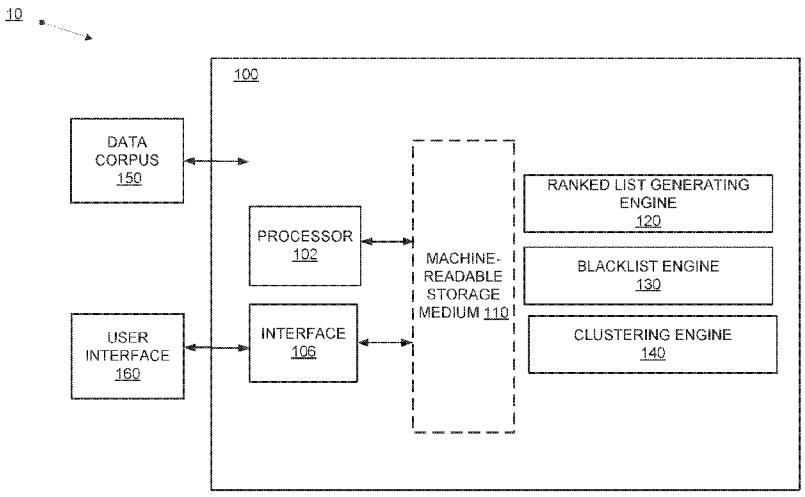

FIG. 1

(57) Abstract: An example method is provided in according with one implementation of the present disclosure. The method comprises computing, via a processor, a ranked elements list for each of a plurality of objects. The method also comprises iteratively computing, via the processor, a blacklist of elements for the objects. The method further comprises determining, via the processor, cluster centers that include top ranked non-blacklisted elements, and assigning, via the processor, each object to at least one cluster center.

CLUSTERING

[0001] A variety of analytic tasks may be performed on data (e.g., big data that generally exceeds the processing capacity of conventional systems), and the results may be provided to a user. The analytics tasks may include creating and running queries, indexing, retrieval, clustering, pattern detection, classification, and others. Clustering or partitioning of data is typically the task of grouping a set of items or objects in such a way that objects in the same group (e.g., cluster) are more similar to each other than to those in other groups (e.g., clusters).

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] Figure 1 is a schematic illustration of an example system for clustering data objects by computing a blacklist in accordance with an implementation of the present disclosure.

[0003] Figure 2 illustrates a flowchart showing an example of a method for clustering data objects by computing a blacklist in accordance with an implementation of the present disclosure.

[0004] Figure 3 illustrates a flowchart showing an example of a method for computing a ranked elements list for data objects in accordance with an implementation of the present disclosure.

[0005] Figure 4 illustrates a flowchart showing an example of a method for computing a blacklist for the objects in accordance with an implementation of the present disclosure.

[0006] Figure 5 is an example block diagram illustrating a computer-readable medium in accordance with an implementation of the present disclosure.

DETAILED DESCRIPTION OF SPECIFIC EXAMPLES

[0007] Many entities (e.g., enterprises, organizations) utilize databases for storage of data relating to the entities. For example, a business may maintain a database of customer information, and the customer information may be accessed by querying the database. Further, entities may generally store vast amounts of data originating from their business, including operations data, customer feedback data, financial data, Human Resource data, and so forth. Data stored in these databases may be accessed and updated for various purposes.

[0008] As described above, data stored in a database may be accessed and analyzed in real-time for various purposes. For example, clustering or partitioning of data has become increasingly popular in recent years. Many organizations use various data clustering methods and techniques to help them analyze and cluster different types of data (e.g., customer surveys, customer support logs, engineer repair notes, system logs, etc.). As used herein, the terms "group" and "cluster" are to be used interchangeably and refer to a set of items that are grouped in a way such that items in the same group are more similar to each other than to those in other groups. As used herein, the terms "data object" and "object" are to be used interchangeably and refer to a data element (e.g., vector of numbers) that, for example, may be stored in a database.

[0009] In many situations, exiting clustering techniques may be slow, inaccurate, and inefficient. Therefore, there is always a need for an improved clustering techniques that provide faster and more accurate analysis of different data.

[0010] In this regard, according to examples, techniques for clustering data objects by computing a blacklist of elements for the data objects are disclosed herein. In the proposed techniques, a blacklist of elements may be used to improve clustering of different data. When data objects are similar, they tend to have common elements (e.g., when an object is represented as a vector). Therefore, the techniques described herein propose using the elements of the different data object to determine similarity between the objects and to cluster the objects accordingly. The techniques described herein propose computing a ranked list for each data object, iteratively determining a blacklist of elements (i.e., elements that cannot be cluster centers for the objects), and assigning each of the objects to cluster centers (i.e., cluster identifiers that form the individual clusters) that include top ranked non-blacklisted elements for the objects. The proposed techniques enhance the efficiency and the accuracy of clustering.

[0011] In the following detailed description, reference is made to the accompanying drawings, which form a part hereof, and in which is shown by way of illustration specific examples in which the disclosed subject matter may be practiced. It is to be understood that other examples may be utilized and structural

or logical changes may be made without departing from the scope of the present disclosure. The following detailed description, therefore, is not to be taken in a limiting sense, and the scope of the present disclosure is defined by the appended claims. Also, it is to be understood that the phraseology and terminology used herein is for the purpose of description and should not be regarded as limiting. The use of "including," "comprising" or "having" and variations thereof herein is meant to encompass the items listed thereafter and equivalents thereof as well as additional items. Furthermore, the term "based on," as used herein, means "based at least in part on." It should also be noted that a plurality of hardware and software based devices, as well as a plurality of different structural components may be used to implement the disclosed methods and devices.

[0012] Referring now to the figures, Figure 1 is a schematic illustration of an example system 10 for clustering data objects by computing a blacklist. The illustrated system 10 is capable of carrying out the techniques described below. As shown in Figure 1, the system 10 is depicted as including at least one computing device 100 (e.g., application server, compute node, desktop or laptop computer, smart phone, etc.). In the embodiment of Figure 1, computing device 100 includes a processor 102, an interface 106, and a machine-readable storage medium 110. Although only computing device 100 is described in details below, the techniques described herein may be performed by several computing devices or by engines distributed on different devices. Thus, the computing device 100 may or may not be an independent computing device. The computing device 100 may include additional components and some of the components depicted therein may be removed and/or modified without departing from a scope of the system that allows for carrying out the functionality described herein.

[0013] In one example, the computing device 100 (or another computing device) may communicate with a data corpus 150 and with an interactive user interface 160 (e.g., graphical user interface). The data corpus 150 may include different types of data objects. The data in the data corpus 150 may include text-like data, categorical data, numerical data, structured data, unstructured data, or any other type of data. The device 100 may receive an incoming data stream of data objects from the data corpus 150.

[0014] The computing device 100 may implement engines 120-140 (and components thereof) in various ways, for example as hardware and programming. Each of the engines 120-140 may include, for example, a hardware device including electronic circuitry for implementing the functionality described below, such as control logic and/or memory. In addition or as an alternative, the engines 120-140 may be implemented as any combination of hardware and software to implement the functionalities of the engines. The programming for the engines 120-140 may take the form of processor-executable instructions stored on a non-transitory machine-readable storage medium and the hardware for the engines 120-140 may include a processing resource to execute those instructions. A processing resource may include a number of processors and may be implemented through a single processor or multi-processor architecture. In an alternative example, engines 120-140 may be distributed between the computing device 100 and other computing devices. It is to be understood that the operations described as being performed by the engines 120-140 of the computing device 100 that are related to this description may, in some implementations, be performed by external engines (not shown) or distributed between the engines of the computing device 100 and other electronic/computing devices.

[0015] Processor 102 may be central processing unit(s) (CPUs), microprocessor(s), and/or other hardware device(s) suitable for retrieval and execution of instructions (not shown) stored in machine-readable storage medium 110. Processor 102 may fetch, decode, and execute instructions to identify different groups in a dataset. As an alternative or in addition to retrieving and executing instructions, processor 102 may include electronic circuits comprising a number of electronic components for performing the functionality of instructions.

[0016] Interface 106 may include a number of electronic components for communicating with various devices. For example, interface 106 may be an Ethernet interface, a Universal Serial Bus (USB) interface, an IEEE 1394 (Firewire) interface, an external Serial Advanced Technology Attachment (eSATA) interface, or any other physical connection interface suitable for communication with the computing device. Alternatively, interface 106 may be a wireless interface, such as a wireless local area network (WLAN) interface or a near-field communication

(NFC) interface that is used to connect with other devices/systems and/or to a network. The user interface 160 and the computing device 100 may be connected via a network. In one example, the network may be a mesh sensor network (not shown). The network may include any suitable type or configuration of network to allow for communication between the computing device 100, the user interface 160, and any other devices/systems (e.g., other computing devices, displays), for example, to send and receive data to and from a corresponding interface of another device.

[0017] In one example, the ranked list generating engine 120 may compute a ranked elements list for each of a plurality of objects (e.g., received from the data corpus 150). As explained in additional details below, the ranked list may include an index of elements of each data object (e.g., index of elements that refer to the position of a vector of the object) and may be computed by using an orthogonal transform. As noted above, the data corpus may include various data objects. Various techniques may be used to compute the ranked elements list for each object. In one implementation, the ranked list generating engine 120 may: compute a replicated vector for each object from an input vector associated with the object; apply a random permutation to the replicated vector for each object; perform orthogonal transform to the replicated vector for each object to generate an index vector; and generate a ranked elements list for each object from the index vector.

[0018] The blacklist engine 130 may iteratively compute a blacklist of elements for the objects. In one example, the blacklist engine 130 may iteratively: select a top ranked element from the ranked list of elements, iteratively determine, for the top ranked element, the count of objects that have the same top ranked element. The blacklist engine 130 may further identify the top ranked element with a highest count of objects, and place the element with the highest count of objects on the blacklist of elements. That way, the blacklist engine 130 may calculate the blacklist of elements.

[0019] The blacklist engine 130 may further iteratively update the blacklist of elements by including another element having the highest count of objects from the top ranked elements for the plurality of objects, where the ranked list of elements excludes elements that are already on the blacklist of elements. The

blacklist engine 130 may identify a plurality of elements having the highest count of objects from the elements for the plurality of objects, and may place the plurality of element having the highest count of objects on the blacklist of elements. That way, multiple element may be simultaneously added to the blacklist.

[0020] The clustering engine 140 may determine cluster centers that include top ranked non-blacklisted elements. In other words, the engine 130 may identify different elements that can be used to cluster the plurality of objects since these elements are common for many objects. The clustering engine 140 may then assign each object from to at least one cluster center. That way, the clustering engine 140 may cluster the object by grouping objects with similar elements together.

[0021] Figure 2 illustrates a flowchart showing an example of a method 200 for clustering data objects by computing a blacklist. Although execution of the method 200 is described below with reference to the system 10, the components for executing the method 200 may be spread among multiple devices/systems. The method 200 may be implemented in the form of executable instructions stored on a machine-readable storage medium, and/or in the form of electronic circuitry.

[0022] In one example, the method 200 can be executed by at least one processor (e.g., processor 102 of device 100). In other examples, the method may be executed by another processor in communication with the system 10. Various elements or blocks described herein with respect to the method 200 are capable of being executed simultaneously, in parallel, or in an order that differs from the illustrated serial manner of execution. The method 200 is also capable of being executed using additional or fewer elements than are shown in the illustrated examples.

[0023] The method 200 begins at 210, where a processor may compute a ranked elements list for each of a plurality of objects. In one example, the ranked elements list includes an index of elements of each data object (e.g., index of elements that refer to the position of a vector of the object). Various techniques may be used to compute the ranked list for each object. An example technique is described below in relation to Figure 3.

[0024] At 220, the processor may iteratively compute a blacklist of elements for the objects. An example technique for computing the blacklist of elements is described below in relation to Figure 4. In one example, the processor may evaluate the ranked list of elements (i.e., the list may sorted based on the value of the elements). In other words, the processor may analyze a prioritized list of the elements for all objects. For the top ranked element (i.e., the element with the highest value on the list), the processor may determine the count of objects that have the same top ranked element. In other words, the processor may count how many objects are associated with the same top ranked element. The element with the highest count is usually not a very good element because the cluster that corresponds to that element will be a very big cluster and this dilutes the difference between the objects. The processor may then place the element with the highest count on the blacklist.

[0025] At 230, the processor may determine cluster centers that include top ranked non-blacklisted elements. In other words, the processor may identify top ranked elements not included in the blacklist of elements for the object and label them as cluster identifiers (i.e., cluster centers) that will help to form the individual clusters of objects. Each object may then be assigned to each of these cluster centers. In one example, the processor may select each top ranked, non-blacklisted element for each object as a potential cluster center.

[0026] Next, the processor may assign each object to at least one cluster center (at 240). In one example, each object may be assigned to its top non-blacklisted element that was identified as a cluster center. Similar object would share the same elements (i.e., cluster centers) and, therefore, will be assigned to the same cluster centers to form a cluster of objects. The generated clusters are very well representative of the received similar data objects. The clusters take into account not only the similarity between the objects but also their differences. Therefore, mathematically, the generated clusters are more useful for future computational and analytical purposes.

[0027] Figure 3 illustrate a flowchart showing an example of a method 300 for computing a ranked elements list for each of the plurality of data objects. Although execution of the method 300 is described below with reference to the

system 10, the components for executing the method 300 may be spread among multiple devices/systems. The method 300 may be implemented in the form of executable instructions stored on a machine-readable storage medium, and/or in the form of electronic circuitry. In one example, the method 300 can be executed by at least one processor of a computing device (e.g., processor 102 of device 100).

[0028] The method 300 begins at 320, where a processor may compute a replicated vector R for each object from an input vector associated with the object. In some example, each data object may be represented as an input vector (e.g., a feature vector from the object). In other words, an object may be represented as an n-dimensional real value numerical input vector. Various techniques could be used to compute a replicated vector R from the input vector.

[0029] At 330, the processor may apply a random permutation to the replicated vector R for each object. Next, the processor may perform an orthogonal transform to the replicated vector R for each object to generate an index vector *IX* (at 340). In one example, the transform may be discrete cosine transform ("DCT"). As described below, various techniques may be used to generate the index vector *IX*.

[0030] In one example, the orthogonal transform to the replicated vector R may generate a transformed vector R'. The processor may sort the elements of the transformed vector R' (e.g., in descending order) to generate the index vector *IX*. If, for example, the largest element of the transformed vector R' is in position/coordinate 5, the index vector *IX* may have 5 as its first element. If, for example, the second largest element of the transformed vector R' is in position 17, the index vector *IX* may have 17 as its second element, etc. In other words, the index vector may record the positions of the vector R' when the positions are transformed. In another example, the processor may not use sorting but may generate the index vector by selecting the first N elements of the transformed vector R', where N may be predetermined or flexible number.

[0031] At 350, the processor may generate a ranked elements list for each object from the index vector *IX*. The ranked list includes an index of the elements of the index vector *IX*, which refer to the positions of the transformed vector R'. In other words, the ranked element includes an index of elements of each data object. Thus, based on the index vector *IX*, the processor may generate the ranked list for each

object. In one implementation, the ranked list may be the index vector *IX* itself. In another implementation, the ranked list may be a reversal of the index vector *IX*. In yet another implementation, the ranked list may be the first *k* elements of the index vector.

[0032] Figure 4 illustrates a flowchart showing an example of a method 400 for computing a blacklist of elements for the objects. Although execution of the method 400 is described below with reference to the system 10, the components for executing the method 400 may be spread among multiple devices/systems. The method 400 may be implemented in the form of executable instructions stored on a machine-readable storage medium, and/or in the form of electronic circuitry. In one example, the method 400 can be executed by at least one processor of a computing device (e.g., processor 102 of device 100).

[0033] Computing the blacklist of elements that cannot cluster centers may be performed iteratively. In one example, the processor may begin with computing a first element to be added to the blacklist. Then, the process may iterate a number of times, where each iteration may add another element to the blacklist. The number of iterations (i.e., the size of the blacklist) must be smaller than the size of the ranked elements list for each object.

[0034] The method 400 begins at 420, where the processor may select a top ranked element from the ranked list of elements for an object. For example, the processor may evaluate the ranked list of elements for each object to determine the element with the highest value on the list (i.e., the top ranked element). The processor may perform this for all existing objects and may, therefore, ultimately determine a list with top ranked elements from all objects. Then, the processor may iteratively determine, for the top ranked element, the count of objects that have the same top ranked element (at 430). In other words, the processor may determine how many objects from the plurality of objects are associated with the same top ranked element. The reasoning behind is that the element with the highest count is usually not a very good element to use for clustering, because the cluster that corresponds to that element will include a large number of objects.

[0035] At 440, the processor may identify the top ranked element (i.e., from the top ranked element for all objects) with the highest count of objects. In other

words, considering all evaluated objects in blocks 420-430, the processor may determine which is the top ranked element having the highest count of objects.

[0036] Next, the processor may place the element with the highest count of objects on the blacklist of elements (at 450). In other words, the processor may identify one element per iteration, and that element is to be included in the blacklist of elements. The process 400 may iterate a number of times, where the number of iterations must be smaller than the size of the ranked list for each object. For example, the processor may determine if any more iterations are required to add additional element to the blacklist (at 460). If no more iterations are required, the processor may end the process 400. If additional iterations are required, the process may return to 420 in order to add a new add a new element to the blacklist. The number of iterations may be based on a threshold, may be predetermined, or adaptive/flexible. The process 400 may end when the identified number of iterations is completed and the size of blacklist reaches a specific level. The proposed technique removes the most common elements for the objects and helps to distribute the objects to other elements, making clusters evenly distributed and better representing similarities between objects.

[0037] In one example, the first time the processor implements the method 400, there may be only one element on the blacklist of elements. In the next iteration, the processor may add another element to the black list, using the techniques described above. Thus, the processor may iteratively update the blacklist of elements by including another element having the highest count of objects from the top ranked elements for the plurality of objects. When another element is being added, the processor may exclude elements that are already on the blacklist of elements from the top ranked list of elements. In other words, while updating the blacklist, the processor may not consider elements that are already on the list.

[0038] In addition, during the implementation of the method 400 the processor may identify a plurality of elements having the highest count of objects from the elements for the plurality of objects. In other words, the processor may simultaneously identify several elements to be included on the blacklist. Then, the processor may place the plurality of elements having the highest count of objects on the blacklist of elements. Thus, the processor may update the blacklist by

adding multiple elements at once. The number of multiple elements added to the blacklist may be predetermined or flexible. After the blacklist of elements is updated, the processor may reassign an object to a different cluster center. Thus, is some examples, the objects may be reevaluated and added to a different cluster based on the updated blacklist.

[0039] Figure 5 illustrates a computer 501 and a non-transitory machine-readable storage medium 505 according to an example. In one example, the computer 501 maybe similar to the computing device 100 of the system 10 or may include a plurality of computers. For example, the computer may be a server computer, a workstation computer, a desktop computer, a laptop, a mobile device, or the like, and may be part of a distributed system. The computer may include one or more processors and one or more machine-readable storage media. In one example, the computer may include a user interface (e.g., touch interface, mouse, keyboard, or gesture input device).

[0040] Computer 501 may perform methods 200-400 and variations thereof. Additionally, the functionality implemented by computer 501 may be part of a larger software platform, system, application, or the like. Computer 501 may be connected to a database (not shown) via a network. The network may be any type of communications network, including, but not limited to, wire-based networks (e.g., cable), wireless networks (e.g., cellular, satellite), cellular telecommunications network(s), and IP-based telecommunications network(s) (e.g., Voice over Internet Protocol networks). The network may also include traditional landline or a public switched telephone network (PSTN), or combinations of the foregoing.

[0041] The computer 501 may include a processor 503 and non-transitory machine-readable storage medium 505. The processor 503 (e.g., a central processing unit, a group of distributed processors, a microprocessor, a microcontroller, an application-specific integrated circuit (ASIC), a graphics processor, a multiprocessor, a virtual processor, a cloud processing system, or another suitable controller or programmable device) and the storage medium 505 may be operatively coupled to a bus. Processor 503 can include single or multiple cores on a chip, multiple cores across multiple chips, multiple cores across multiple devices, or combinations thereof.

[0042] The storage medium 505 may include any suitable type, number, and configuration of volatile or non- volatile machine-readable storage media to store instructions and data. Examples of machine-readable storage media include read-only memory ("ROM"), random access memory ("RAM") (e.g., dynamic RAM ["DRAM"], synchronous DRAM ["SDRAM"]), electrically erasable programmable read-only memory ("EEPROM"), magnetoresistive random access memory (MRAM), memristor, flash memory, SD card, floppy disk, compact disc read only memory (CD-ROM), digital video disc read only memory (DVD-ROM), and other suitable magnetic, optical, physical, or electronic memory on which software may be stored.

[0043] Software stored on the non-transitory machine-readable storage medium 505 and executed by the processor 503 includes, for example, firmware, applications, program data, filters, rules, program modules, and other executable instructions. The processor 503 retrieves from the machine-readable storage medium 505 and executes, among other things, instructions related to the control processes and methods described herein.

[0044] The processor 503 may fetch, decode, and execute instructions 507-511 among others, to implement various processing. As an alternative or in addition to retrieving and executing instructions, processor 503 may include at least one integrated circuit (IC), other control logic, other electronic circuits, or combinations thereof that include a number of electronic components for performing the functionality of instructions 507-511. Accordingly, processor 503 may be implemented across multiple processing units and instructions 507-511 may be implemented by different processing units in different areas of computer 501.

[0045] The instructions 507-511 when executed by processor 503 (e.g., via one processing element or multiple processing elements of the processor) can cause processor 503 to perform processes, for example, methods 200-400, and/or variations and portions thereof. In other examples, the execution of these and other methods may be distributed between the processor 503 and other processors in communication with the processor 503.

[0046] For example, ranked list generating instructions 507 may cause processor 503 to compute a ranked elements list for each of a plurality of objects. These instructions may function similarly to the techniques described in block 210 of method 200 and the techniques described in method 300. In one example, the ranked list generating instructions 507 may cause processor 503 to: compute a replicated vector for each object from an input vector associated with the object; apply a random permutation to the replicated vector for each object; perform an orthogonal transform to the replicated vector for each object to generate an index vector; and generate a ranked elements list for each object from the index vector.

[0047] Blacklist generating instructions 509 may cause the processor 503 to iteratively compute a blacklist of elements for the objects and to iteratively update the blacklist of elements by including at least one new element having highest count of objects. These instructions may function similarly to the techniques described in block 220 of method 200 and the techniques described in relation to the method 400. For example, blacklist generating instructions 509 may cause the processor 503 to select a top ranked element from the ranked list of element; iteratively determine, for the top ranked element, the count of objects that have the same top ranked element; identify the top ranked element with a highest count of objects; and place the element with the highest count of objects on the blacklist of elements. Further, the blacklist generating instructions 509 may cause the processor 503 to reassign an object to a different cluster center after updating the blacklist of elements.

[0048] Cluster instructions may cause the processor 503 to determine cluster centers that include top ranked non-blacklisted elements, and to assign each object to at least one cluster center. These instructions may function similarly to the techniques described blocks 230-240 of method 200.

[0049] In the foregoing description, numerous details are set forth to provide an understanding of the subject matter disclosed herein. However, implementations may be practiced without some or all of these details. Other implementations may include modifications and variations from the details discussed above. It is intended that the appended claims cover such modifications and variations.

What is claimed is:

1. A method comprising:

computing, via a processor, a ranked elements list for each of a plurality of objects;

iteratively computing, via the processor, a blacklist of elements for the objects;

determining, via the processor, cluster centers that include top ranked non-blacklisted elements; and

assigning, via the processor, each object to at least one cluster center.

2. The method of claim 1, wherein computing a raked elements list comprises:

computing, via the processor, a replicated vector for each object from an input vector associated with the object;

applying, via the processor, a random permutation to the replicated vector for each object;

performing, via the processor, an orthogonal transform to the replicated vector for each object to generate an index vector; and

generating, via the processor, a ranked elements list for each object from the index vector.

3. The method of claim 1, wherein computing the blacklist of elements comprises:

iteratively performing:

selecting, via the processor, a top ranked element from the ranked list of elements for an object,

iteratively determining, via the processor, for the top ranked element, the count of objects that have the same top ranked element,

identifying, via the processor, the top ranked element with a highest count of objects, and

placing, via the processor, the element with the highest count of objects on the blacklist of elements.

4. The method of claim 3, further comprising:

iteratively updating, via the processor, the blacklist of elements by including another element having the highest count of objects from the top ranked elements for the plurality of objects, wherein the ranked list of elements excludes elements that are already on the blacklist of elements.

5. The method of claim 3, further comprising:

identifying, via the processor, a plurality of elements having the highest count of objects from the elements for the plurality of objects;

placing, via the processor, the plurality of elements having the highest count of objects on the blacklist of elements.

6. The method of claim 1, wherein the ranked elements list includes an index of elements of each of the objects.

7. A system comprising:

a ranked list generating engine to compute a ranked elements list for each of a plurality of objects, wherein the ranked elements list is computed by using an orthogonal transform;

a blacklist engine to iteratively compute a blacklist of elements for the objects; and

a clustering engine to:

determine cluster centers that include top ranked non-blacklisted elements, and

assign each object from to at least one cluster center.

8. The system of claim 7, wherein the ranked list generating engine is further to:

compute a replicated vector for each object from an input vector associated with the object;

apply a random permutation to the replicated vector for each object;

perform orthogonal transform to the replicated vector for each object to generate an index vector; and

generate a ranked elements list for each object from the index vector.

9. The system of claim 7, wherein the blacklist engine is further to:

iteratively perform:

select a top ranked element from the ranked elements list for an object,

iteratively determine, for the top ranked element, the count of objects that have the same top ranked element,

identify the top ranked element with a highest count of objects, and

place the element with the highest count of objects on the blacklist of elements.

10. The system of claim 7, wherein the blacklist engine is further to:

iteratively update the blacklist of elements by including another element having the highest count of objects from the top ranked elements for the plurality of objects, wherein the ranked list of elements excludes elements that are already on the blacklist of elements.

11. The system of claim 7, wherein the blacklist engine is further to:

identify a plurality of elements having the highest count of objects from the elements for the plurality of objects; and

place the plurality of elements having the highest count of objects on the blacklist of elements.

12. A non-transitory machine-readable storage medium encoded with instructions executable by at least one processor, the machine-readable storage medium comprising instructions to:

compute a ranked elements list for each of a plurality of objects;

iteratively compute a blacklist of elements for the objects;

iteratively update the blacklist of elements by including at least one new element having highest count of objects;

determine cluster centers that include top ranked non-blacklisted elements; and

assign each object to at least one cluster center.

13. The non-transitory machine-readable storage medium of claim 12, further comprising instructions to: reassign an object to a different cluster center after updating the blacklist of elements.

14. The non-transitory machine-readable storage medium of claim 12, further comprising instructions to:

compute a replicated vector for each object from an input vector associated with the object;

apply a random permutation to the replicated vector for each object;

perform an orthogonal transform to the replicated vector for each object to generate an index vector; and

generate a ranked list for each object from the index vector.

15. The non-transitory machine-readable storage medium of claim 12, further comprising instructions to:

iteratively perform:

select a top ranked element from the ranked list of elements for an object,

iteratively determine, for the top ranked element, the count of objects that have the same top ranked element,

identify the top ranked element with a highest count of objects, and

place the element with the highest count of objects on the blacklist of elements.

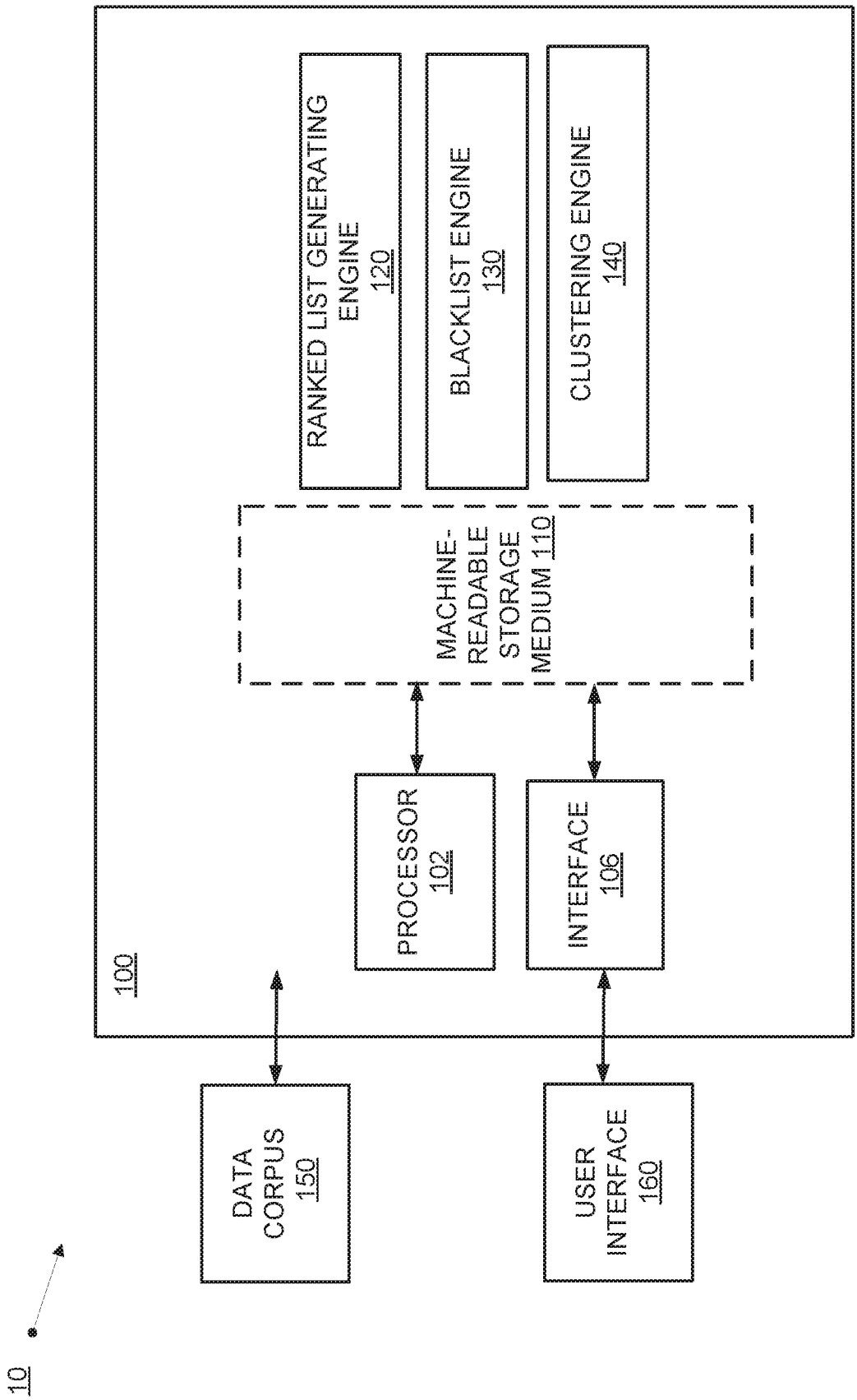
10
100
RANKED LIST GENERATING ENGINE 120
BLACKLIST ENGINE 130
CLUSTERING ENGINE 140
MACHINE-READABLE STORAGE MEDIUM 110
PROCESSOR 102
INTERFACE 106
DATA CORPUS 150
USER INTERFACE 160

FIG. 1

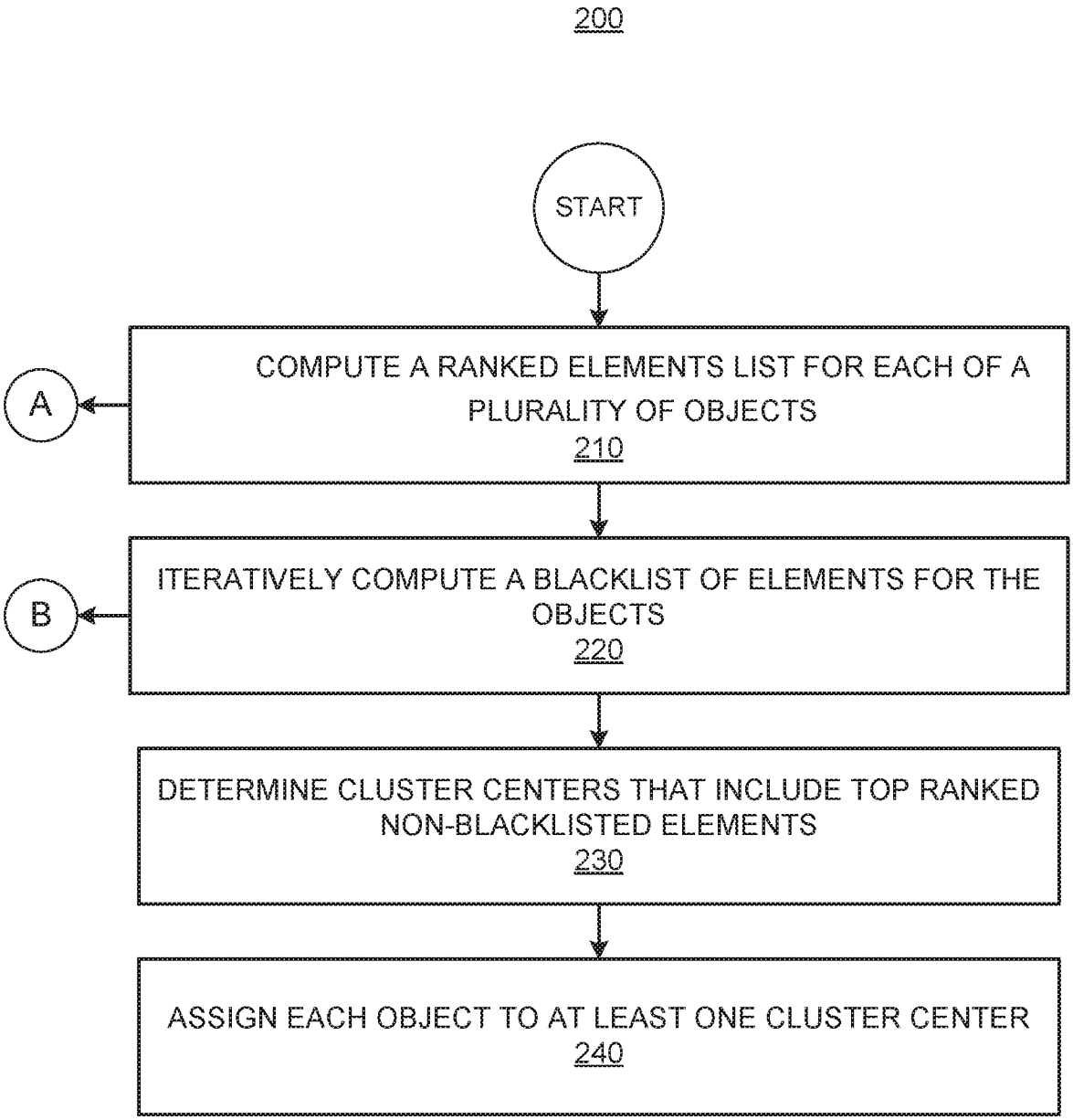
200
START
A
COMPUTE A RANKED ELEMENTS LIST FOR EACH OF A PLURALITY OF OBJECTS
210
B
ITERATIVELY COMPUTE A BLACKLIST OF ELEMENTS FOR THE OBJECTS
220
DETERMINE CLUSTER CENTERS THAT INCLUDE TOP RANKED NON-BLACKLISTED ELEMENTS
230
ASSIGN EACH OBJECT TO AT LEAST ONE CLUSTER CENTER
240

Fig. 2

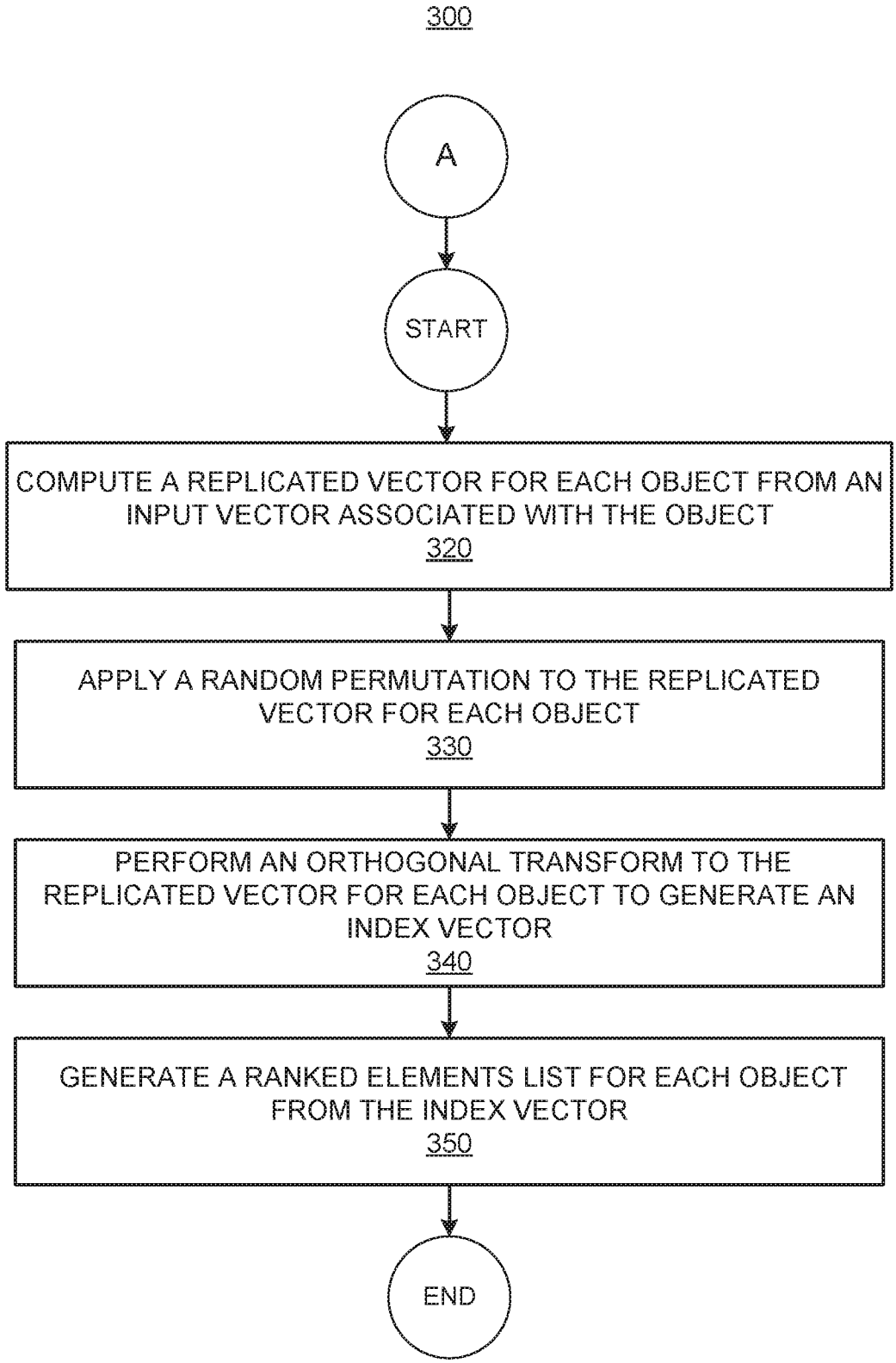
300
A
START
COMPUTE A REPLICATED VECTOR FOR EACH OBJECT FROM AN INPUT VECTOR ASSOCIATED WITH THE OBJECT
320
APPLY A RANDOM PERMUTATION TO THE REPLICATED VECTOR FOR EACH OBJECT
330
PERFORM AN ORTHOGONAL TRANSFORM TO THE REPLICATED VECTOR FOR EACH OBJECT TO GENERATE AN INDEX VECTOR
340
GENERATE A RANKED ELEMENTS LIST FOR EACH OBJECT FROM THE INDEX VECTOR
350
END

Fig. 3

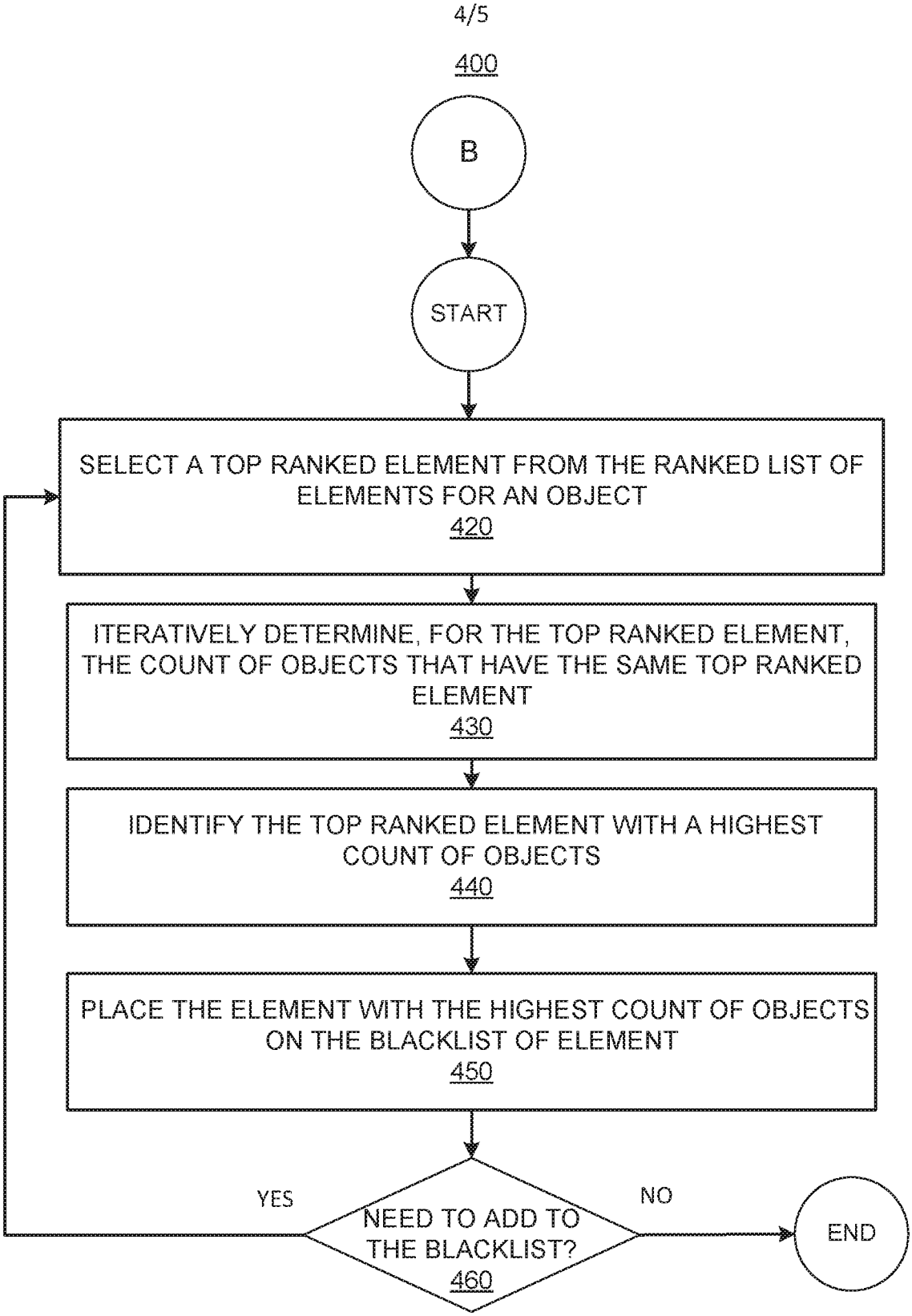
400
B
START
SELECT A TOP RANKED ELEMENT FROM THE RANKED LIST OF ELEMENTS FOR AN OBJECT
420
ITERATIVELY DETERMINE, FOR THE TOP RANKED ELEMENT, THE COUNT OF OBJECTS THAT HAVE THE SAME TOP RANKED ELEMENT
430
IDENTIFY THE TOP RANKED ELEMENT WITH A HIGHEST COUNT OF OBJECTS
440
PLACE THE ELEMENT WITH THE HIGHEST COUNT OF OBJECTS ON THE BLACKLIST OF ELEMENT
450
YES
NEED TO ADD TO THE BLACKLIST?
460
NO
END

Fig. 4

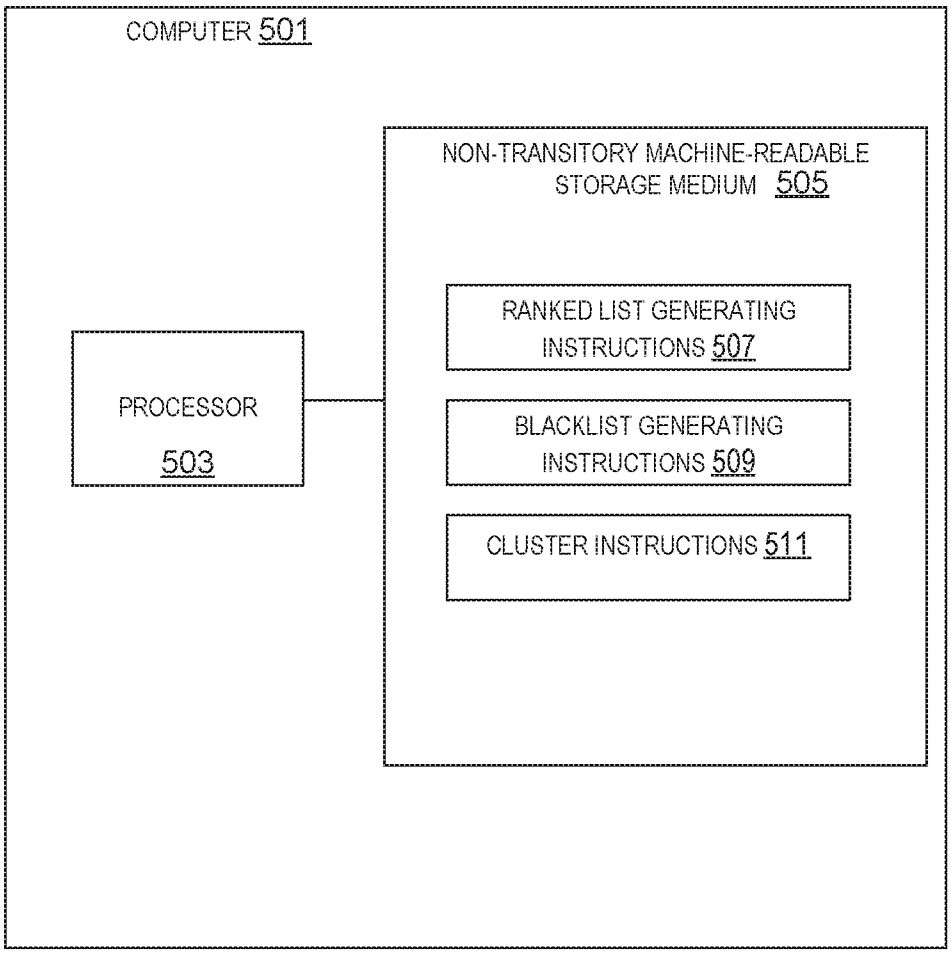
COMPUTER 501
NON-TRANSITORY MACHINE-READABLE STORAGE MEDIUM 505
RANKED LIST GENERATING INSTRUCTIONS 507
BLACKLIST GENERATING INSTRUCTIONS 509
CLUSTER INSTRUCTIONS 511
PROCESSOR 503

Fig. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2015/066830

A. CLASSIFICATION OF SUBJECT MATTER

G06F 17/30(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 17/30; G06K 9/62

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: rank, object, blacklist, cluster, and similar terms.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2008-0085055 A1 (CATHLEEN D. CEROSALETTI et al.) 10 April 2008 See paragraphs [0028], [0075], [0079]-[0081], and [0087]; claims 1 and 24; and figures 1-2.	1,3-6,12-13,15
Y		7,9-11
A		2,8,14
Y	US 2014-0310314 A1 (SAMSUNG ELECTRONICS CO., LTD.) 16 October 2014 See paragraphs [0030]-[0033]; and figure 1.	7,9-11
A	WO 2007-059216 A2 (CLAIRVOYANCE CORPORATION) 24 May 2007 See page 5, lines 5-20; page 14, lines 9-13; and figures 1-2.	1-15
A	US 2015-0012540 A1 (HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.) 08 January 2015 See paragraphs [0014]-[0016] and [0070]-[0071]; and figures 1-3.	1-15
A	US 2012-0294540 A1 (JIAN SUN et al.) 22 November 2012 See paragraphs [0037]-[0048]; and figures 3-4.	1-15

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
07 September 2016 (07.09.2016)	**07 September 2016 (07.09.2016)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea
Facsimile No. +82-42-481-8578

Authorized officer
LEE, Dong Yun
Telephone No. +82-42-481-8734

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2015/066830

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2008-0085055 A1	10/04/2008	EP 2069974 A2	17/06/2009
		EP 2087441 A1	12/08/2009
		EP 2122498 A1	25/11/2009
		JP 2010-506296 A	25/02/2010
		JP 2010-506297 A	25/02/2010
		JP 2010-521024 A	17/06/2010
		JP 5261724 B2	14/08/2013
		US 2008-0085032 A1	10/04/2008
		US 2008-0205772 A1	28/08/2008
		US 7869658 B2	11/01/2011
		WO 2008-045233 A2	17/04/2008
		WO 2008-045233 A3	05/06/2008
		WO 2008-045236 A1	17/04/2008
		WO 2008-103412 A1	28/08/2008
US 2014-0310314 A1	16/10/2014	WO 2014-171735 A1	23/10/2014
WO 2007-059216 A2	24/05/2007	JP 2009-516307 A	16/04/2009
		US 2007-0112867 A1	17/05/2007
		WO 2007-059216 A3	04/12/2008
US 2015-0012540 A1	08/01/2015	None	
US 2012-0294540 A1	22/11/2012	None	