



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

(21) International Application Number:
PCT/US2016/015848

(22) International Filing Date:
30 January 2016 (30.01.2016)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: ENTIT SOFTWARE LLC [US/US]; 1140 Enterprise Way, Building G, Sunnyvale, CA 94089 (US).

(72) Inventor: BYRD, Christopher; 2 Carlson Road, New Milford, Connecticut 06776 (US).

(74) Agent: HEWLETT PACKARD ENTERPRISE; 3404 E. Harmony Road, Mail Stop 79, Fort Collins CO 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))

[Continued on next page]

(54) Title: LOG EVENT CLUSTER ANALYTICS MANAGEMENT

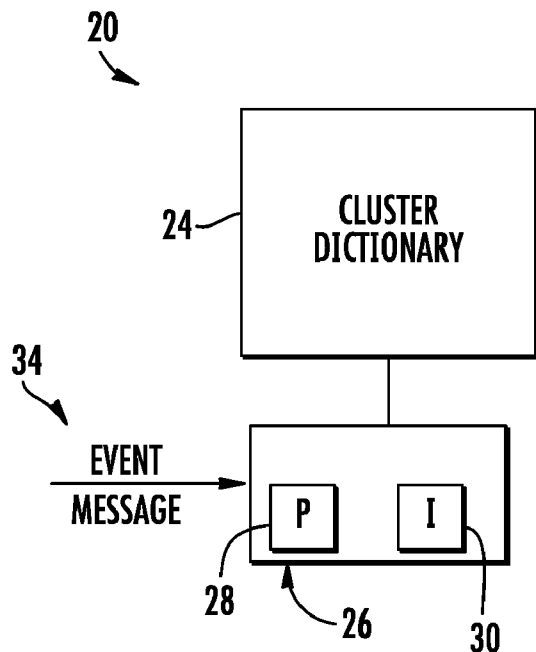


FIG. 1

(57) Abstract: A log event cluster analytics management method may involve storing a first portion of an entire cluster dictionary in a transient memory, storing at least a second portion of the entire cluster dictionary in a persistent database and comparing a new log event message to the first portion of the overall cluster dictionary. In response to not assigning the new log event message to any cluster in the first portion of the entire cluster dictionary in the transient memory, selecting a subset of clusters of the at least second portion of the cluster dictionary in the persistent database, comparing the new log event message to a cluster of the selected subset of clusters and assigning the new log event message to the cluster of the selected subset of clusters based upon the comparison.



— *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))*

LOG EVENT CLUSTER ANALYTICS MANAGEMENT

BACKGROUND

[0001] Computer systems are formed from multiple components, such as hardware components and programming components. Such components may generate and store human readable, textual event messages into system event log files or logs. These logs may be stored on network servers and may be usable in system development, debugging and understanding behavior of a system. Automated analysis of such logs of event messages may be used to diagnose computer system components.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] Figure 1 is a schematic diagram of an example log event cluster analytics management system.

[0003] Figure 2 is a flow diagram of an example method for managing analytics of log event message clusters.

[0004] Figure 3 is a table of example log event messages of a portion of an example computer system log file.

[0005] Figure 4 is a table of the example log event messages of Figure 3 after rearrangement.

[0006] Figure 5 is a schematic diagram of example clusters.

[0007] Figure 6 is a table of the example log event messages of Figure 3 assigned into the example clusters of Figure 5.

[0008] Figure 7 is an example representation of a reduced log generated from the example message cluster assignments illustrated in Figure 6.

[0009] Figure 8 is an example diagram schematically illustrating a cluster dictionary comprising cluster families of message clusters.

[00010] Figure 9 is a flow diagram of an example method for managing analytics of log event clusters using cluster families.

[00011] Figure 10 is a schematic diagram illustrating an example log event cluster analytics management system.

[00012] Figure 11 is a flow diagram of an example method for managing analytics of log event clusters using multi-level cluster analysis.

[00013] Figure 12 is a schematic diagram illustrating an example log event cluster analytics management system.

[00014] Figure 13 is a flow diagram of an example method for managing analytics of log event clusters using distributed analysis and cluster merging.

[00015] Figure 14 is a schematic diagram illustrating an example log event cluster analytics management system.

[00016] Figure 15A is a flow diagram of a first portion of an example method for managing analytics of log event clusters.

[00017] Figure 15B is a flow diagram of a second portion of the example method for managing analytics of log event clusters.

[00018] Figure 15C is a flow diagram of a third portion of the example method for managing analytics of log event clusters.

[00019] Figure 16 is a flow diagram of an example method for merging clusters.

DETAILED DESCRIPTION OF EXAMPLES

[00020] Traditional methods for automated analysis of such computer-generated event message logs may not be well suited for large-scale systems and networks, wherein the number of event message clusters may overwhelm available resources. Such traditional methods may consume large amounts of memory and processing time. For example, with some traditional methods, event messages may be arranged in clusters of related messages, wherein the entire dictionary of clusters is stored entirely in main memory of a single system and wherein a new event message is compared to each and every cluster, in an age-based sequence, until a match is found.

[00021] In contrast to such traditional systems, examples consistent with the present disclosure may facilitate analytics of computer-generated event message logs for large-scale systems and networks. As will be described hereafter, examples consistent with the present disclosure may implement the grouping of clusters into families, wherein the families themselves are analyzed in a predefined order so as to more quickly identify a match between existing clusters and an incoming event message.

[00022] Examples consistent with the present disclosure may additionally or alternatively store a portion of the total cluster dictionary in the transient memory of a computer node for initial faster analysis while maintaining a larger persistent storage database of the entire cluster dictionary, wherein a text search engine is used to identify those clusters in the persistent storage database for analysis should an incoming event message not match any of the clusters in the transient memory. As a result, the generally smaller transient memory may not contain the entire cluster dictionary.

[00023] In some implementations, examples consistent with the present disclosure distribute the analysis of incoming event messages amongst multiple computer nodes, each of the computer nodes having an associated

transient memory storing a portion of the overall cluster dictionary, wherein the overall cluster dictionary is stored in the larger persistent storage. In such an example implementation, the system may merge sufficiently similar clusters in the persistent storage database, pursuant to a predefined similarity threshold, to eliminate redundant clusters in the database that are created by the different compute nodes. The merged clusters of the database are propagated to the transient memories. In one implementation, a text search engine is used to identify those clusters in the persistent storage database for analysis should an incoming event message not match any of the clusters in the transient memory of a particular computer node.

[00024] Figure 1 schematically illustrates an example automated log event cluster analytics management system 20. As shown by Figure 1, system 20 comprises cluster dictionary 24 and log event cluster analytics manager 26. Cluster dictionary 24 comprises clusters of log event messages (also known as logs) against which newly incoming log event messages are compared. In one implementation, cluster dictionary 24 comprises a master set or the entire set of clusters to be compared to a new event message for assignment. In one implementation, cluster dictionary 24 is stored in a transient storage device, a volatile storage device, a device that maintains stored values while being powered, but may lose stored values when not powered, such as a random access memory. For example, in one implementation, cluster dictionary 24 is stored in the main memory, the transient memory, of a commodity server or the main memory of a desktop.

[00025] In another implementation, cluster dictionary 24 is stored in a persistent storage device, a nonvolatile device that maintains values in the absence of being powered, such as a disk storage. In contrast to a transient storage device which may have a maximum size on the order of gigabytes, a persistent storage device may have a maximum size on the order of terabytes or larger. In some implementations, the transient storage device may be distributed amongst multiple persistent storage device units which are

sequentially accessed. In some implementations, the persistent storage device may serve as a backup storage for multiple individual transient storage devices or main memories associated with multiple computer nodes.

[00026] Log event cluster analytics manager 26 comprises a computer node comprising a processor 28 and stored instructions 30. Processor 28 comprises a processing unit (processing hardware) that follows instructions 30 provided in a non-transitory computer-readable medium or memory. For purposes of this application, computer node comprises both a processing unit and stored instructions. The term “processing unit” shall mean a presently developed or future developed processing unit or processing hardware that executes sequences of instructions contained in a memory. Execution of the sequences of instructions causes the processing unit to perform steps such as generating control signals. The instructions may be loaded in a random access memory (RAM) for execution by the processing unit from a read only memory (ROM), a mass storage device, or some other persistent storage. In other examples, hard wired circuitry may be used in place of or in combination with instructions to implement the functions described. For example, log event cluster analytics manager 26 may be embodied as part of one or more application-specific integrated circuits (ASICs).

[00027] Figure 2 is a flow diagram illustrating an example of how log event cluster analytics manager 26 may create message clusters from incoming event messages and one example of how log event cluster analytics manager 26 may manage the analytics of a large number of message clusters that may accumulate over time from a large-scale system or network. As indicated by block 40, a new event message from an event log file or log is received. Figure 3 illustrates an extract from an example log file 100 that may be received by log event cluster analytics manager 26. Logs comprise semi-structured events generated automatically when programming or hardware components output messages describing actions, warnings or errors during their operation. The illustrative portion from the example log file 100 includes

including twelve individual events 104. An event 104 may have a timestamp 105, representing the date and time at which the programming generated the event, and at least a textual narrative 106 describing the event. In some events additional fields might appear (not shown), for example describing severity level, source method/function, etc.

[00028] Even though there are twelve events shown in the log file in the example of Figure 3, four message templates have been used to generate the messages with ten distinct (i.e. completely different) messages. The four templates are illustrated in the re-arranged table shown in Figure 4. The ten distinct messages result from variables 109 (i.e. numbers, words or and symbols) in the narrative 106 that have been inserted into the templates. In this example, the variables 109 in Figure 4 are IP addresses, user session #, project names and conststr identifiers.

[00029] For the following purposes, each log event, e , is denoted by a tuple (t, msg) , where t is the timestamp 105 of the message and msg is the message text 106, represented by the word vector $\text{msg} = w_1, w_2, \dots, w_n$, where w_i is the word in the i 'th position, while n is the number of words in a message. Each w_i represents a word from all the words present in the logs, and the set of log entries is E .

[00030] As shown in Figure 2, as each new event message is received, the event message is either assigned to an existing message cluster (block 42) or serves as a basis or message event template for defining or forming a new message cluster (block 44). Figure 5 schematically illustrates an example set of clusters 602, 604, 606, 608, 610 and 612 formed from the example portion of the log file 100 shown in Figures 3 and 4. Clusters 610 and 612 are illustrated as being split from cluster 606.

[00031] Clusters 602, 604, 606, 608, 610 and 612 may be part of an overall cluster dictionary 450. In generating a cluster dictionary, such as the example cluster dictionary 450, system 20 begins with zero or more clusters

defined in the cluster dictionary. Although schematically illustrated, cluster dictionary 450, may in practice, may be arranged as any appropriate form of data structure, for example a text file, table, database, xml file, etc. Each cluster 602 in the cluster dictionary 450 includes at least an event template, comprising the text 106 (or some other appropriate representation of the text, such as, for example, an encoded or hashed variant or a pointer to the text in a template database or the like) of a representative log event message, and a message count 603, indicating the number of times a log event message has been assigned to the cluster. In effect, each cluster represents a prototypical feature message according to a representative message. New event messages may be compared against the message event template text to determine whether the new event message is sufficiently similar to the cluster to be assigned to the cluster.

[00032] In one implementation, message clusters, such as clusters 602, 604, 606 comes 608, 610 and 612, are formed by clustering event messages that are sufficiently similar to one another based upon an order-sensitive cosine similarity function. One example of such an order-sensitive cosine similarity function that may be applied to provide a measure of similarity (i.e. a 'distance') of two messages may take the form:

$$\langle msg_1, msg_2 \rangle = \frac{n_{12}}{\sqrt{n_1 \cdot n_2}}, \quad \text{<Equation 1>}$$

where n_{12} is the number of identical words comparing *each word position* of msg_1 and msg_2 , and n_1 , n_2 are the numbers of words in each message. In the example, a resulting cosine distance is a number between 0 and 1. When the result is 1, the two messages are identical, and when the result is 0, the two messages are completely different. Values between 1 and 0 represent a measure or degree of similarity.

[00033] The determined similarity value may then be compared against a predefined threshold to determine whether a new event message should be

assigned to an existing cluster. The pre-defined threshold may be determined heuristically by applying the process to a real data set – for which a higher threshold would lead to more clusters and a lower threshold would lead to fewer clusters – to establish what threshold provides the most accurate clustering. The threshold applied may vary between different systems, depending on the nature of the events that are generated.

[00034] Figure 6 illustrates an example application of the order sensitive cosine similarity function (Equation 1) to the example 12 log event messages shown in Figure 3. As shown by Figure 6, text in the second event in the log file extract is very different from the text in the first event, even though each event comprises 14 words. As shown by the dotted arrows, the first nine words in each event are not the same; however the final five words in each event are the same, as represented by solid arrows. Applying Equation 1 to these two events provides the similarity measure of $5/14 = 0.357$, which is below a predefined or selected threshold of 0.85 and means that the second event forms CLUSTER B 604, and the second event is assigned cluster B 440 in the cluster assignment 440. In contrast, the text in the fifth event in the log is the same as the text in the first event, apart from the 9th words in each, which are 55555 and 44444 respectively. Applying Equation 1 to the first and fifth events provides the similarity measure of $13/14 = 0.929$, which is above the threshold of 0.85. Hence the fifth event does not form a new cluster. Instead, it is assigned to cluster A in the cluster assignment 440 and the message count 603 for cluster A is incremented by one.

[00035] In other implementations, an edit distance, or variations of it, allowing for word insertions and deletions may be used to define a degree of similarity between two log event messages for purposes of clustering.

[00036] According to some examples, each new event is compared to the template in each of the existing clusters in the order in which the clusters were created, and is assigned to the first cluster to which the similarity threshold is exceeded. In some circumstances, an event may be assigned to

the first cluster with which it exceeds the threshold, even though a later cluster might be a closer match. If the similarity threshold is not surpassed for any of the existing clusters, a new cluster is created and the event text is used as the event template of the new cluster.

[00037] As indicated by block 46, system 20 groups the clusters formed in blocks 42 and 44 into different families. Once the clusters have been initially grouped into families pursuant to block 48, analysis of subsequently received event messages 40 in blocks 42 and 44 is carried out in a predefined family-by-family order that may facilitate faster matching or assignment of the incoming event message to an event message cluster. In one implementation, within each family, the clusters are compared to the incoming event message based on the relative age of each cluster in the family, in an order from oldest to youngest.

[00038] Block 46 illustrates the triggering of the grouping of clusters into families. In one implementation, the grouping of clusters into families is automatically triggered in response to the creation of a new message cluster, per block 44. In such an implementation, the newly created cluster may be compared to the existing clusters to determine whether the newly created cluster should join an existing family of clusters or should serve as a foundation for a new family of clusters. In another implementation, the grouping of clusters into families is automatically triggered in response to a predefined threshold being satisfied or not being satisfied, such as the total number of individual clusters, not part of any multi-cluster family, exceeding a threshold or in response to performance of system 20 failing to satisfy a predefined performance threshold. In another implementation, the grouping of clusters into families is automatically triggered on a periodic basis. For example, such grouping of clusters may occur “off-line” at predefined time intervals. In yet another implementation, the grouping of clusters may be triggered in response to a user inputted command to initiate grouping.

[00039] As indicated by block 48, once such cluster grouping has been triggered, manager 26 proceeds with the grouping of clusters into families. During such grouping, manager 28 compares the relationship between existing clusters in cluster dictionary 24 to determine whether clusters are sufficiently similar so as to overlap one another. Figure 8 is a diagram schematically illustrating an example cluster dictionary 850 illustrating various cluster families 852A, 852B, 852C, 852D and 852E (collectively referred to as cluster families 852) formed within memory 26 by manager 26. As shown by Figure 8, the different cluster families 852 are each composed of different log event message clusters 854. The log event clusters 854 of each of cluster families 852 are sufficiently proximate or near one another so as to be grouped in a single cluster family. As shown by Figure 8, cluster families 852 may have various forms. Cluster 852A illustrates an example cluster family in which three clusters each concurrently overlap one another, each cluster overlapping both of the other two clusters. Cluster family 852B illustrates a cluster family in which two outer clusters overlap a central cluster, forming a chain. Cluster family 852C illustrates a cluster family formed from two overlapping clusters, wherein the two clusters have different cosine distances with respect to their centers or representative messages. Cluster family 852E comprises a single cluster to which additional clusters may be subsequently appended.

[00040] In one implementation, the determination of whether to group two clusters is in a fashion similar to the determination of the nearness between two log event messages when determining whether the two event messages should be clustered together. In one implementation, the representation message serves as a center of each cluster while the outer bounds of the cluster is defined by the threshold order sensitive cosine distance (described above) from the representation message. The order sensitive cosine distance of the representative message serves as a radius. Two clusters are grouped together as part of a single cluster family when the

outer bounds of a first cluster intersects or overlaps the outer bounds of a second cluster.

[00041] For example, in one implementation, for each event message cluster being examined, cluster manager 28 may determine whether the representative event message of the cluster being examined falls within an order sensitive cosine threshold of any other cluster of cluster dictionary 24. In response to the cluster being examined having a representative message falling within the order sensitive cosine threshold of another cluster (whether the other cluster is an individual cluster or a cluster family), the cluster is merged with the other cluster to form a multi-cluster family. In other implementations, other criteria may be applied to determine the relationship or proximity of a cluster and another cluster or multi-cluster family and to determine whether the cluster should be merged with the other cluster or multi-cluster family.

[00042] Figure 9 is a flow diagram schematically illustrating one example method 900 of how manager 26 utilizes cluster dictionary 850, comprising families of clusters, to facilitate quicker matches such that those clusters more likely to match (i.e., fall within the order sensitive cosine threshold of the representative message of a cluster) the new event message are processed before other less relevant clusters. As indicated by block 902, manager 26 receives the new log event message (similar block 40).

[00043] As indicated by block 906, manager 26 may compare the new event message to each cluster 852 in cluster dictionary 850 according to a predefined order of cluster families. The predefined order of the cluster families 852 may not necessarily be based upon age of the cluster family and may not static. For example, in one implementation, the order in which manager 26 compares cluster families 852 to the new event message is based upon a historical frequency of event messages being added to a member cluster of the particular cluster family. In one implementation, those cluster families that are the most active with respect to the addition of event

messages, are examined first, prior to other cluster families that may be less active with respect to the addition of event messages. For example, in one particular circumstance where cluster family 852B was being assigned event messages over a predetermined period at a higher rate as compared to the rate at which event messages, during the same predetermined period of time, were assigned to cluster family 852C, manager 26 may compare a new event message to the clusters of cluster family 852B prior to comparison to the clusters of cluster family 852C. In this fashion, those clusters that are more relevant, that may be more likely to be a match with the new event message (where the new event message is more likely to fall within the threshold cosine distance or the outer bounds of a particular cluster) are examined before other less relevant clusters.

[00044] In one implementation, the predetermined order at which the cluster families 852 are examined with respect to a new event message is based upon activity or frequency of event message additions over multiple periods of time, wherein the different periods of time may be differently weighted. For example, the rate at which prior event messages have been added to a particular cluster family for a more recent period of time may be given a greater weight or weighting as compared to the rate at which prior event messages have been added to the particular cluster family during an older period of time.

[00045] In one implementation, different weightings may be applied to the historical activity values for the different cluster families 852 for different periods of time based upon the period of time or current universal time from which the new event message has arisen. For example, in circumstances where the new event message was generated or stored in the log at a particular time of the day, day of the week, day of the month or the like, those activity rates for each of the different cluster families for the same particular time the day, day of the week, day of the month or the like may be given a higher weighting as compared to other different times of day, days of the

week, days of the month or the like. In one implementation, the closer a particular time period is to the period of time during which the new event message originated, the greater the weighting may be applied to the activity value for the particular period of time for each of the cluster families 852. In other words, the activity (the rate at which event matches are being added a member cluster of a cluster family) for a particular period of time will have a larger impact on the order in which the different cluster families are examined with respect to a new event message depending upon how similar the particular period of time is to the particular period of time during which the new event message was received or logged. If a new event message is received on Wednesday afternoon, the historical activity values for the different cluster families 852 for historical periods of time encompassing Wednesday afternoons will be given a greater weighting as compared to the activity values for other periods of time, say Friday mornings, for the different cluster families 852.

[00046] In one implementation, the order in which cluster families 852 are compared to the new event message is not based upon activity (the rate at which event messages have been previously added to a member cluster of the cluster group), but is based upon timing similarities. Those cluster families 852 having clusters having a larger percentage of assigned event messages that were logged or received closer in time or during the same period of time as the new event message being examined will be examined prior to those cluster families having clusters having a smaller percentage of assigned event messages that were logged or received farther away in time or more remote with respect to the period of time during which the new event message being examined was received or logged. For example, if a new event messages received or logged at 2 PM, manager 26 may first examine those cluster families 852 having a large percentage of event messages assigned to them that were also received or logged at 2 PM as compared to other cluster families having a smaller percentage of event messages assigned to them that were also received or logged at 2 PM. By identifying

those particular cluster families that are more likely to contain a cluster that matches the new event message, either based upon historical activity or based upon seasonality or timing, the time to match or assign the new event message to an appropriate cluster may be reduced, shortening the automated processing time for the new event message.

[00047] As indicated by block 908, for each cluster family 852 that is being examined with respect to the new event message received by manager 26, manager 26 automatically compares the new event message to each message cluster of the cluster family 852 in an order based upon an age of each message cluster. For example, when examining cluster family 852B, manager 26 compares the ages (when each of the clusters were created) of each of the member event clusters 854, and proceeds through the member clusters of the cluster family in an order based upon the comparison. In one implementation, manager 26 compares a new event message to each of the clusters of the cluster family, in an order from oldest to youngest. By examining each cluster within a particular cluster family based upon the age of the individual clusters, uniformity or local determination is achieved in that a new event message that falls within the scope of multiple clusters within the same cluster group will be assigned to the same cluster each time, the oldest cluster.

[00048] Upon identifying a matching cluster and assigning the new event message to the matching cluster, manager 26 proceeds to the next received event message and repeats the above described method 900. In the event that the new event message does not match any of the existing clusters of any of the existing cluster families, manager 26 creates a new cluster and a new corresponding cluster family (the new cluster family initially comprising a single cluster), wherein the new event message serves as the initial representative message for the cluster and corresponding cluster family.

[00049] Figure 10 schematically illustrates another example automated log event log event cluster analytics management system 1120, an example of system 20. System 1120 facilitates the analysis of log event messages received from across large-scale systems and networks by storing a portion (less than the entirety) of the total cluster dictionary in a transient memory of a computer node for initial faster analysis while maintaining a larger persistent storage database of the entire cluster dictionary, wherein a text search engine is used to identify those clusters in the persistent storage database for analysis should an incoming event message not match any of the clusters in the transient memory. As a result, the generally smaller transient memory may facilitate faster matching of an incoming log event message to a cluster.

[00050] In the illustrated example, the entire cluster dictionary 24 of clusters 1150 is stored in a nonvolatile persistent storage device or database 1154. In one implementation, database 1154 comprises a disk-based storage system. In other implementations, database 1154 may comprise other forms of non-volatile persistent storage devices. In one implementation, database 1154 comprises a single persistent storage unit. In other implementations, database 1154 may be distributed across multiple individual persistent storage units. Although possibly slower to access information as compared to a transient memory, such as random access memory, database 1154 offers a storage platform having a much larger capacity that may be more suited for storing the large number of clusters that may result from a large-scale system or network. In some implementations, database 1154 may have a maximum size on the order of terabytes or larger.

[00051] As further shown by Figure 10, system 1120 allocates or apportions a subset 1160 of the total number of clusters 1150 stored in database 1154 to a main memory or transient memory 1156. Transient memory 1156 comprises a volatile memory, such as a random access memory. In contrast to the persistent storage device of database 1154, the transient memory 1156 may offer faster searching and faster retrieval of data,

facilitating faster matching of an incoming log event message to an event cluster, if the event cluster is contained in transient memory 1156. Transient memory 1156 may have a much smaller capacity as compared to database 1154. In one implementation, transient memory 1156 may have a maximum size on the order of gigabytes.

[00052] In one implementation, manager 26 selects the portion or subset 1160 of the total number of clusters 1150 of dictionary 24 that is contained within transient memory 1156 based upon the age of the individual message clusters 1150. For example, in one implementation, manager 26 automatically maintains those message clusters of dictionary 24 in transient memory 1156 that have an age less than a predetermined threshold age. In another implementation, manager 26 automatically maintains a predetermined percentage of the youngest message clusters of dictionary 24 within transient memory 1156. In yet another implementation, manager 26 automatically maintains a predetermined number of the most recently created message clusters within memory 1156, wherein older clusters are expelled on a first in first out basis. In yet other implementations, manager 26 automatically maintains in memory 1156 those message clusters having an activity level (based upon the number of event messages assigned to the particular message cluster during a predetermined period of time) that is greater than a predetermined threshold. In yet other implementations, manager ranks or orders the message clusters within dictionary 24 and automatically maintains a predetermined number or percentage of the message clusters in memory 1156, based upon the relative activity ranking. In such a manner, those clusters most likely to satisfactorily match the new event message and be assigned the new event message are maintained in transient memory 1156 while those less relevant clusters, the clusters that are older, or that have lower activity levels or the like, are maintained as part of the overall cluster dictionary 24 maintained in persistent database 1154.

[00053] Figure 11 is a flow diagram illustrating an example method 1200 for managing the analytics of clusters by manager 26 in system 1120. As indicated by block 1202, a first portion of the entire overall cluster dictionary is stored in transient memory 1156. As indicated by block 1204, at least a second portion of the entire overall cluster dictionary is stored in persistent database 1154.

[00054] As indicated by block 1206, manager 26 initially carries out a level I analysis by comparing the new log event message to each of the clusters in the transient memory 1156. In one implementation, manager 26 compares the new event message to the clusters 1160 in transient memory 56 in order based upon the age of each of clusters 1160, from oldest to youngest. Should a match be found, should the new event message be satisfactorily near one of clusters 1160 (the new event message falling within the order sensitive cosine distance of the particular cluster 1160), manager 26 assigns the new event message to the particular cluster 1150 and proceeds with analysis of the next succeeding new event message.

[00055] As indicated by block 1208, in response to the new event message not satisfactorily matching any of the clusters 1150 within transient memory 1156, manager 26 automatically carries out a level II analysis by selecting a subset 1166 of the clusters contained in persistent database 1154 for examination with respect to the new event message. In one implementation, the subset 1166 is selected by manager 26 using term vector analysis. In one implementation, the persistent database 1154 is built on a Lucene index or similar indexing foundation, wherein Lucene and its term vector analysis is used to efficiently recall or extract subset 1166 from persistent database 1154 based upon a proximity of each of the clusters of subset 1166 to the new event message.

[00056] As indicated by block 1212, manager 26 compares the new event message to each of the clusters of the subset 1166 from the persistent

database 1154 in the same fashion as was performed with the comparison of the new event message to each of the clusters in the transient memory 1156. In one implementation, the new event message is compared to each of the clusters in subset 1166 until a sufficient match is found, wherein the manager 26 compares the new event message to the subset 1166 of clusters 1150 in order based upon the age of each of clusters 1150 of subset 1166, from oldest to youngest. Should a match be found, should the new event message be satisfactorily near one of clusters 1150 of subset 1166 (the new event message falling within the order sensitive cosine distance of the particular cluster 1162), manager 26 assigns the new event message to the particular cluster 1150 and proceeds with analysis of the next succeeding new event message. Should a match not be found, log event cluster analytics manager 26 creates a new cluster based upon the new event message, the new cluster being added to the overall cluster dictionary 24, and in some implementations, to the subset of clusters stored in transient memory 1156.

[00057] In the above described implementation of system 1120, persistent database 1154 stores the entire cluster dictionary 24, wherein a portion or subset of the entire dictionary 24 is duplicated and maintained in the transient memory 1156. In an alternative implementation, persistent database 1154 may not store the entire cluster dictionary 24, but the entire cluster dictionary 24 may be distributed between persistent database 1154 and transient memory 1156. For example, in one implementation, more frequently matched clusters 1150 may be stored in transient memory 1156 while the less frequently matched or less active clusters 1150 may be stored in persistent database 1154. In such an implementation, transient memory 1156 may not contain the entire cluster dictionary 24. At the same time, persistent database 1154 also may not contain the entire cluster dictionary 24. Moreover, the entire cluster dictionary 24 may not be indexed and searched pursuant to block 1208 and 1212 of method 1200.

[00058] In some implementations, manager 26 automatically maintains those clusters 1150 in persistent database 1154 (and not in transient memory 1156) that have attained an age greater than a predetermined threshold age. In another implementation, manager 26 automatically maintains a predetermined percentage of the oldest message clusters within persistent database 1154. In yet another implementation, manager 26 automatically maintains a predetermined number of message clusters within transient memory 1156, wherein older clusters are expelled or transferred from transient memory 1156 to distributed database 1154 on a first in first out basis. In yet other implementations, manager 26 automatically maintains in distributed database 1154 (and not in transient memory 1156) those message clusters having an activity level (based upon the number of event messages assigned to the particular message cluster during a predetermined period of time) that falls below a predetermined threshold. In yet other implementations, manager ranks or orders the message clusters within transient memory 1156 and automatically transfers a predetermined number or percentage of the message clusters to persistent database 1154, based upon the relative activity ranking. In such a manner, those clusters most likely to satisfactorily match the new event message and be assigned the new cup event message are maintained in transient memory 1156 while those less relevant clusters, the clusters that are older, have lower activity levels or the like, are maintained in persistent database 1154.

[00059] Figure 12 schematically illustrates another example automated log event log event cluster analytics management system 1320, an example of system 20. System 1320 facilitates the analysis of log event messages received from across large-scale systems and networks by distributing the analysis of incoming event messages amongst multiple computer nodes, each of the computer nodes having an associated transient memory storing a portion or subset of the overall cluster dictionary, wherein the overall cluster dictionary is stored in the larger persistent storage database. Each computer

node operates in a fashion similar to system 1120 and method 1200 described above.

[00060] In the example illustrated in Figure 12, manager 26 apportions the comparison of newly received event messages with respect to the cluster dictionary 24 amongst different computer nodes 1302, 1304, 1306, each compute node comprising a processor and instructions for directing operation of processor, wherein the instructions are stored on a non-transitory computer-readable medium. As shown by Figure 12, computer nodes 1302, 1304 1306 compare their received log event messages to corresponding assigned subsets 1312, 1314 and 1316 of clusters 1318 stored in transient memories 1322, 1324 and 1326, respectively.

[00061] In the example illustrated, computer node 1302 compares its assigned log event messages against clusters 1-100 stored in its associated transient memory 1322; computer node 1304 compares its assigned log event messages against clusters 50-150 stored in its associated transient memory 1324; and computer node 1306 compares its assigned log event messages against clusters 100-200 stored in its associated transient memory 1326. Log event cluster analytics manager 26 maintains the entire dictionary in persistent database 1154, clusters 1-200 in the illustrated example. Although Figure 12 illustrates three compute nodes, in other implementations, greater or fewer number of such compute nodes may be used to collectively process received event messages.

[00062] Although Figure 12 illustrates the cluster dictionary 24 being allocated in an overlapping fashion amongst the memories 1322, 1324, 1326, of the different computer nodes, in other implementations, the cluster dictionary 1315 may be allocated in other overlapping fashions or non-overlapping fashions amongst the different memories, such as random access memories, of the different computer nodes 1302, 1304, 1306. For example, in other implementations, computer node 1302 may compare new event

messages to clusters 1-66, computer node 1304 may compare new event messages to clusters 67-133 and computer node 1306 may compare new event clusters to clusters 134-200. By allocating the analysis or matching of new event messages amongst multiple computer nodes, processing time and backlogs for handling large numbers of event messages are reduced, and message processing throughput can be scaled higher within the framework of the system.

[00063] Figure 13 is a flow diagram illustrating an example method 1400 for managing clusters by manager 26 in system 1320. As indicated by block 1402, manager 26 divides or allocates new event messages that are received amongst the different computer nodes, such as computer nodes 1302, 1304, 1306, for comparison to the different associated subsets of clusters of the overall set of clusters forming the cluster dictionary 1350. As discussed above with respect to blocks 42 and 44 in Figure 2, each computer node 1302, 1304, 1306 compares its assigned log event messages to its associated subset of clusters. If a sufficient match has been found, the particular event message is assigned to a particular cluster in the subset of clusters associated with a particular computer node. If the new event message being processed by particular computer node does not satisfactorily match any of the clusters in the subset of clusters in the particular transient memory for the particular computer node (i.e., the new event message does not fall within the order sensitive threshold cosine distance of the representative event message for any of the clusters in the transient memory assigned to the particular computer node), the computer node proceeds by selecting a subset of the clusters stored in distributed database 1154 for examination as described above with respect to block 208. The individual compute node 1302, 1304, 1306 compares the new log event message to the clusters of the selected subset and assigns the new log event message to one of the clusters if a match is identified. If no match is identified, the compute node 1302, 1304, 1306 creates a new message cluster containing the new

event message in its associated transient memory 1322, 1324, 1326 and in persistent database 1154.

[00064] Due to the divided or distributed analysis of event messages by manager 26, different computer nodes 1302, 1304, 1306 may, at times, create message clusters in their respective memories and in persistent database 1154 that are substantially similar or identical to one another. As indicated by block 1404 in Figure 13, log event cluster analytics manager 26 periodically or in an off-line fashion, compares the clusters in the persistent database 1154 to one another to determine if a sufficient amount of similarity between any of the clusters exists to trigger merging of different newly created clusters. In one implementation, log event cluster analytics manager 26 determines whether the center (the representative message) of a particular cluster lies within the outer bounds of another cluster (within the order sensitive cosine distance threshold from the representative message of the other cluster) to determine whether the two clusters should be merged to achieve a stable eventually consistent model of log clusters or cluster dictionary. In one implementation, for each of the clusters within persistent database 1154, manager 26 determines whether the representative message of a first cluster being examined falls within a predetermined order sensitive cosine threshold with respect to a second cluster to determine whether the first cluster and the second cluster should be merged into a single cluster. In one implementation, the predetermined order sensitive cosine threshold used to determine whether two clusters should be merged into a single cluster is less than the predetermined order sensitive cosine threshold described above with respect to method 900 that is used to determine whether two clusters should be grouped as part of a multi-cluster family.

[00065] As indicated by block 1406 in Figure 13, if two clusters are found to be the same log cluster or substantially similar to one another (satisfying a predetermined similarity threshold) so as to support merging, log event cluster analytics manager 26 merges the two clusters and persists the single cluster

resulting from the merger to persistent database 1154. The single cluster formed from the merging of the two previous clusters in database 1154 may then be apportioned to the appropriate transient memories 1322, 1324, 1326 for use by the different computer nodes 1302, 1304, 1306.

[00066] Although each of methods 1200 and 1400 are described in terms of individual clusters 1150, in other implementations, methods 1200 and 1400 may be carried out with respect to families comprising an individual cluster and multi-cluster families as described above with respect to Figures 2, 8 and 9. In such an implementation, clusters 1150 are grouped into families and wherein an incoming message is analyzed with respect to the formed cluster families in a manner described above with respect to method 900.

[00067] Figure 14 is a diagram schematically illustrating another example automated log event log event cluster analytics management system 1520, an example of system 20. System 1520 facilitates the analysis of log event messages received from across large-scale systems and networks by carrying out each of methods 900, 1200 and 1400 described above. In the example illustrated, the individual clusters in each of memories 1322, 1324, 1326 are grouped into cluster families in a fashion similar to the cluster families as described above with respect to Figures 2 and 8-9. In one implementation, all of the clusters of the overall cluster dictionary 24 are grouped into cluster families, wherein the formed cluster families are allocated or apportioned amongst the different transient memories 1322, 1324, 1326 of the different computer nodes 1302, 1304, 1306 in an overlapping fashion (similar to the overlapping of the clusters as described above with respect to Figure 12) or in a non-overlapping fashion where memories 1322, 1324, 1326 do not contain any identical cluster family. In another implementation, all of the clusters of the overall cluster dictionary 1350 are first allocated or apportioned amongst the transient memories 1322, 1324, 1326 of the different computer nodes 1302, 1304, 1306, respectively, wherein the individual

clusters assigned to the different memories 1322, 1324, 1326 are then examined and grouped into cluster families within each memory. The clusters of the overall cluster dictionary 1524 contained in persistent database 1154 are provided with the same cluster families as those contained in memories 1322, 1324, 1326.

[00068] Figures 15A-15C are flow diagrams of an example method 1600 that involves three general operations: determining whether an incoming log message 1602 falls within an existing cluster in main or transient memory 1152 (operation 1622 illustrated in Figure 15A), if not, determining whether the incoming log message 1602 falls within an existing cluster in a distributed may database 1154 (operation 1623 illustrated in Figure 15B), and, if not, creating a new cluster based upon the incoming log message 1602 (operation 1624 illustrated in Figure 15C). In the example illustrated, the operation 1623 is concurrently carried out by each compute node 1322, 1324, 1326 as each compute node 1322, 1324, 1326 receives its allocated incoming log message 1602 for analysis. In one implementation, as illustrated by Figure 14, the clusters 1150 in both the individual transient memories 1322, 1324 and 1326 as well as persistent database 1154 have been previously grouped into families 852. Each family 852 comprises a parent cluster N that is the oldest cluster of the family and may additionally comprise younger clusters X which are children of the parent cluster N. Some of families 852 may have a parent cluster N without any child clusters X.

[00069] For purposes of discussion, operation 1622 is described as being carried out with respect to an incoming log message 1602 received by compute node 1302 with associated transient memory 1322. Each of the steps described with respect to compute node 1302 as part of operation 1622 or as part of other operations of method 1600 may likewise be similarly carried out by each of the other compute nodes, compute nodes 1304 and 1306. As compute node 1302 ask upon its transient memory 1322, compute nodes 1304 and 1306 act upon their respective transient memories 1324 and

1326, respectively. As indicated by blocks 1626 and 1628, for each cluster family 852 contained in the particular transient memory 1322, compute node 1302 determines whether the particular incoming log message 1602 falls within an order sensitive cosine similarity threshold of a particular cluster of the family. The analysis to determine whether the incoming log event message 1602 falls within the order sensitive cosine similarity threshold is pursuant to the order described above respect to method 900. As described above with respect to block 906 and method 900, the families are analyzed in a predefined order of cluster families to facilitate faster matching of the incoming log message 602 to a cluster.

[00070] As indicated by block 1628, for each family 852 being examined, the incoming log event message 1602 is first compared to the parent cluster N of the cluster family 852. As indicated by arrow 1630, if the incoming log message 1602 does fall within the cosine similarity threshold for a particular parent cluster N, node 1302 proceeds by assigning the incoming log message to the particular parent cluster N and by continuing with the next incoming log message. As indicated by block 1634, if the incoming log message 1602 does not match the parent cluster N of the particular family being examined by compute node 1302, the computer node 1302 determines whether the parent cluster N has any children clusters X. As indicated by arrow 1636, if the particular cluster N does not have any children (i.e., the cluster family 852 contained a single cluster), compute node 1302 returns to block 1226 and examines the next cluster family 852 in transient memory 1322. As indicated by blocks 1638 and 1640, if the parent cluster N has children clusters X, for each child cluster X of cluster N, compute node 1302 determines whether the incoming log message 1602 falls within the order sensitive cosine similarity threshold of cluster X. The compute node 1322 compares the incoming log event message 1602 to each of the children clusters X of the same family 852 in an order based upon the age of each child cluster X, oldest to youngest, until a match is found or until all of the children clusters X of the particular parent cluster N in the particular family 852 have been examined. As

indicated by arrow 1641, if the incoming log message 1602 does fall within the cosine similarity threshold for a particular child cluster X, manager 1232 assigns the incoming log message to the particular cluster X and by continues with the next incoming log message.

[00071] As indicated by arrow 1642, if the incoming log message 1602 does not fall within the cosine similarity threshold of any of the child clusters X of the particular cluster N in the particular family 852, manager 26 returns to block 1626 to examine the next cluster family 852, beginning with the parent cluster N of the particular family 852. As indicated by arrow 1644, if the incoming log message does not fall within the order sensitive cosine similarity threshold of any cluster N or any child clusters X of any cluster families 852 residing in the particular transient memory 1322 contained in transient memory 1322, method 1600 proceeds to operation 1623.

[00072] In operation 1623, manager 26 and/or compute node 1302 determines whether the incoming log message 1602 falls within any of the parent clusters N of the families 852 of the overall cluster dictionary 1524 as stored in persistent database 1154. As indicated by block 1650, manager 26 and/or compute node 1302 searches for similar log messages or logs using a text search engine 1154, returning the top most similar representative message or messages along with their associated clusters 1150 and associated cluster families 852. In one implementation, the text search engine comprises Lucene, a high performance text searching library. In such an implementation, the Lucene index is built over the representative messages of each cluster as it is created, and linked to cluster dictionary 1524 in persistent database 1154. In other implementations, other text search engines may be utilized to identify the most similar representative log messages and their associated clusters contained in secondary memory 1154.

[00073] As indicated by blocks 1652 and 1654, for each cluster N in the text search engine found clusters 1150, manager 26 and/or compute node 1302 determines whether the incoming log message 1602 falls within the order sensitive cosine similarity threshold of the particular cluster N. As indicated by block 1656, if the incoming log message 1602 falls within the cosine similarity threshold of the particular cluster N (cluster 1150 of subset 1166) found in persistent database 1154, manager 26 and/or compute node 1302 loads particular matching cluster N (cluster 1150 of subset 1166) and its children or parents, if any, from the persistent database 1154 into the transient memory 1322 associated with compute node 1302. As indicated by block 1658, if the transient memory 1322 is full, manager 26 and/or compute node 1302 first evicts at least one infrequently used cluster 1150 from transient memory 1322 to make room for the cluster (cluster 1150 of subset 1166) (along with its children X). As indicated by block 1660, the incoming log message 1602 is assigned to the particular cluster N (cluster 1150 of subset 1166) and compute node 1302 proceeds with operation 1222 for the next incoming log message 1602.

[00074] As indicated by arrow 1662, if the search carried out in block 1650 does not identify any similar logs and corresponding clusters, manager 26 and/or compute node 1302 proceeds with operation 1224 with respect to the incoming log message 1602. Likewise, as indicated by arrow 1664, if the incoming log message 1602 does not fall within the order sensitive cosine similarity threshold for any of the clusters N identified by the text search engine (clusters 1162 of subset 1166), manager 26 and/or computer node 1302 proceeds with operation 1624 with respect to the incoming log message 1602.

[00075] In operation 1224, manager 26 and/or compute node 1302 creates a new cluster for the incoming log message 1602. As indicated by block 1670, manager 26 and/or compute node 1302 creates a new cluster with a representative message equal to the incoming log message 1602. As

indicated by block 1672, manager 26 and/or compute node 1302 persists, promulgates or adds the newly created cluster to the distributed persistent database 1154. As indicated by block 1674, manager 26 and/or computer node 1302 further indexes the incoming log message 1602 and the identifier of the newly created cluster as a document in the text search engine, such as Lucene, within persistent database 1154.

[00076] As indicated by block 1676, manager 26 and/or compute node 1302 further adds the newly created cluster to transient memory 1322. As indicated by block 1678, if the transient memory 1322 is full, manager 26 and/or compute node 1302 first evicts at least one infrequently used cluster family 852 from transient memory 1322 to make room for the newly created cluster. As indicated by block 1680, the incoming log message 1602 is assigned to the new cluster and compute node 1302 proceeds with operation 1222 for the next incoming log message 1602.

[00077] As the number of newly created clusters in persistent database 1154 increases, the grouping or merging of clusters in persistent database 1154 may be triggered. Examples of criteria which may trigger the grouping, merging and reconciliation of clusters are described above with respect to block 46. For example, in one implementation, such merging or grouping may be automatically periodically triggered, wherein such group clustering is carried out off-line, while compute nodes 1302, 1304 and 1306 continue to receive and analyze incoming log event messages. In another implementation, the trigger event 46 may be based upon a threshold number of clusters existing across different transient. In still other implementations, other events or circumstances may be used as a basis for triggering method 1700.

[00078] As indicated by blocks 1724 and 1726, for each or parent cluster N of each family 852 in distributed persistent database 1154, manager 26 compares the representative message of the particular cluster N to the

other clusters N of the other families 852 according to an order sensitive cosine similarity function. In one implementation, the order in which the other parent clusters and of the different families are analyzed is based upon age, oldest to youngest. As indicated by block 1728, for each of the other parent clusters N of the other families 852, manager 26 first determines whether the particular cluster N being examined is the “same” cluster as the other cluster N. If the particular cluster N and the other parent cluster of the other families are the “same” (with the same representative message or separated a distance within a first order sensitive cosine threshold distance for belonging to the same cluster), manager 26 proceeds with block 1730.

[00079] As indicated by block 1730, manager 26 merges the two clusters and determines which of the two parent clusters is the oldest. The oldest parent cluster becomes a cluster while the younger cluster is deleted. The younger cluster’s current count is merged with the older cluster and the younger clusters children (if any) become children of the older cluster. Thereafter, manager 26 returns to block 1724, comparing the next cluster N in persistent database 1154.

[00080] As indicated by block 1732, if the particular cluster N is not the “same” as any of the other parent clusters N of the different families 852, manager 26 determines whether the particular cluster N being examined falls within a second order sensitive cosine threshold, larger than the first order sensitive cosine threshold, indicating that the two clusters are similar to one another. As indicated by block 1734, in response to determining that the particular cluster N being examined is similar to one of the other clusters N in the distributed database (falling within a threshold distance indicating similarity), manager 26 merges the two clusters and determines which the two clusters is oldest. The oldest cluster of the two clusters becomes a parent cluster. The younger cluster of the two clusters becomes a child of the older cluster. Any younger cluster’s ancestors or descendents (parents and children), if any, become children of the older cluster. Thereafter, manager

returns to block 1724, comparing the next cluster N in the persistent database 1154. As indicated by block 1740, after all of the clusters N have been compared to one another and merged as appropriate, manager 26 persists for promulgates the changes in merged clusters to database 1154.

[00081] Although the present disclosure has been described with reference to example implementations, workers skilled in the art will recognize that changes may be made in form and detail without departing from the spirit and scope of the claimed subject matter. For example, although different example implementations may have been described as including features providing benefits, it is contemplated that the described features may be interchanged with one another or alternatively be combined with one another in the described example implementations or in other alternative implementations. Because the technology of the present disclosure is relatively complex, not all changes in the technology are foreseeable. The present disclosure described with reference to the example implementations and set forth in the following claims is manifestly intended to be as broad as possible. For example, unless specifically otherwise noted, the claims reciting a single particular element also encompass a plurality of such particular elements. The terms “first”, “second”, “third” and so on in the claims merely distinguish different elements and, unless otherwise stated, are not to be specifically associated with a particular order or particular numbering of elements in the disclosure.

WHAT IS CLAIMED IS:

- 1 1. A log event cluster analytics system comprising:
2 a cluster dictionary of log event message clusters; and
3 at least on processing unit to:
4 group the log event message clusters into cluster families;
5 receive a new log event message associated with an event
6 log of a tenant of a network;
7 compare the new log event message to each of the cluster
8 families in a predefined order of the cluster families;
9 for each cluster family, compare the new log event message
10 to each message cluster of the cluster family in an order based on an age
11 of each message cluster of the cluster family; and
12 assign the new log event message to one of the clusters
13 based on the comparison.
- 1 2. The log event cluster analytics management system of claim 2,
2 wherein the predefined order of cluster families is based upon a frequency of
3 event messages being added to each of the cluster families.
- 1 3. The log event cluster analytics management system of claim 2,
2 wherein the predefined order of cluster families varies based upon a current
3 universal time.

1 4. The log event cluster analytics management system of claim 2,
2 wherein the computer node is associated with a first memory storing cluster
3 families of a first portion of the cluster dictionary, the system further comprising:

4 a second memory storing a second portion of the cluster
5 dictionary, the second portion comprising second cluster families
6 different than the cluster families of the first portion;

7 a persistent database storing the cluster dictionary,
8 comprising the cluster families and the second cluster families; and

9 a second computer node to:

10 receive a second new event message associated with
11 a second computer system event log;

12 compare the second new event message to each of
13 the second cluster families in a predefined order of the second
14 cluster families;

15 for each of the second cluster families, comparing the
16 second new event message to each message cluster of the second
17 cluster family in an order based on an age of each message cluster
18 of the second cluster family; and

19 assigning the second new log event message to a cluster of
20 the second cluster families based on the comparison.

1 5. The log event cluster analytics management system of claim 4
2 cluster families and the second cluster families, the first computer node to:

3 receive a third new event message associated with a computer
4 system event log;

5 compare the third new event message to clusters of the cluster
6 families stored in the first memory;

7 in response to the third new event message not being assigned to a
8 cluster of the cluster families stored in the first memory, select a subset of
9 the clusters stored in the persistent database;

10 compare the third new event message to clusters of the subset of
11 the clusters of the persistent database; and

12 determine whether to assign the third new log event message to a
13 cluster of the subset of clusters based on the comparison.

1 6. The log event cluster analytics management system of claim 5,
2 wherein the log event cluster analytics manager, in response to the third new
3 event message not being assigned to a cluster of the subset of the clusters of the
4 third cluster families stored in the persistent database, is to create a new cluster
5 in the persistent database based upon the third new log event message and
6 containing the third new event message.

1 7. The log event cluster analytics management system of claim 5,
2 wherein the subset is selected from the clusters stored in the persistent database
3 using a text search engine.

1 8. The log event cluster analytics management system of claim 1,
2 wherein the log event cluster analytics manager is to:

3 compare a first cluster in the cluster dictionary to a second cluster
4 in the cluster dictionary; and

5 merge the first cluster and the second cluster into a single
6 combined cluster based upon the comparison.

- 1 9. A log event cluster analytics management method comprising:
- 2 storing a first portion of an entire cluster dictionary in a transient
- 3 memory;
- 4 storing at least a second portion of the entire cluster dictionary in a
- 5 persistent database;
- 6 comparing a new log event message to the first portion of the
- 7 overall cluster dictionary;
- 8 in response to not assigning the new log event message to any
- 9 cluster in the first portion of the entire cluster dictionary in the transient
- 10 memory, selecting a subset of clusters of the at least second portion of the
- 11 cluster dictionary in the persistent database;
- 12 comparing the new log event message to a cluster of the selected
- 13 subset of clusters; and
- 14 assigning the new log event message to the cluster of the selected
- 15 subset of clusters based upon the comparison.
- 1 10. The log event cluster analytics management method of claim 11,
- 2 wherein the at least second portion of the overall cluster dictionary comprises the
- 3 entire cluster.
- 1 11. The log event cluster analytics management method of claim 11,
- 2 wherein the at least second portion of the overall cluster dictionary comprises
- 3 less than the overall cluster dictionary.
- 1 12. The log event cluster analytics management method of claim 11,
- 2 wherein the first portion of the entire cluster dictionary in the transient memory
- 3 comprises families of group clusters, wherein the comparing of the new log event

4 message to the first portion of the overall cluster dictionary comprises comparing
5 the new log event message to each cluster family in a predefined order of cluster
6 families and for each cluster family, comparing the new log event message to
7 each message cluster of the cluster family in an order based on an age of each
8 message cluster of the cluster family.

1 13. The log event cluster analytics management method of claim 11
2 further comprising:

3 storing a third portion of the entire cluster dictionary in a second
4 transient memory;

5 comparing a second new log event message to the third portion of
6 the overall cluster dictionary;

7 in response to not assigning the second new log event message to
8 any cluster in the third portion of the entire cluster dictionary in the second
9 transient memory, selecting a second subset of clusters of the at least
10 second portion of the cluster dictionary in the persistent database;

11 comparing the second new log event message to a cluster of the
12 selected subset of clusters; and

13 assigning the second new log event message to the cluster of the
14 second selected subset of clusters based upon the comparison.

1 14. A non-transitory computer-readable medium containing
2 instructions to manage analytics of log event clusters by directing a processing
3 unit to:

4 store a first portion of a cluster dictionary in a transient memory;

5 store at least a second portion of the cluster dictionary in a persistent
6 database;

7 compare a new log event message to the first portion of the overall cluster
8 dictionary;

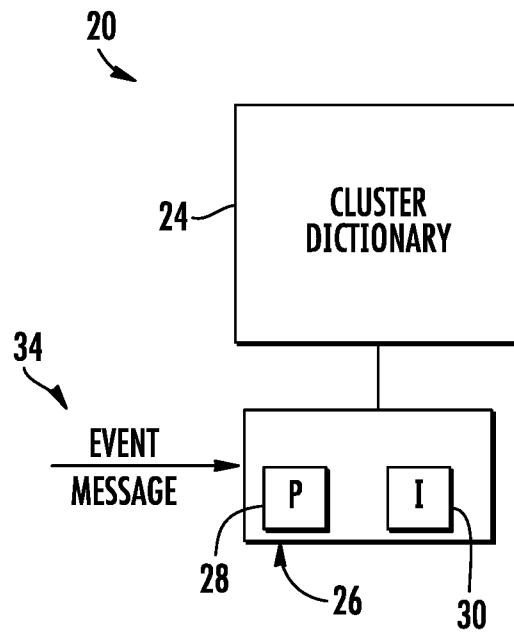
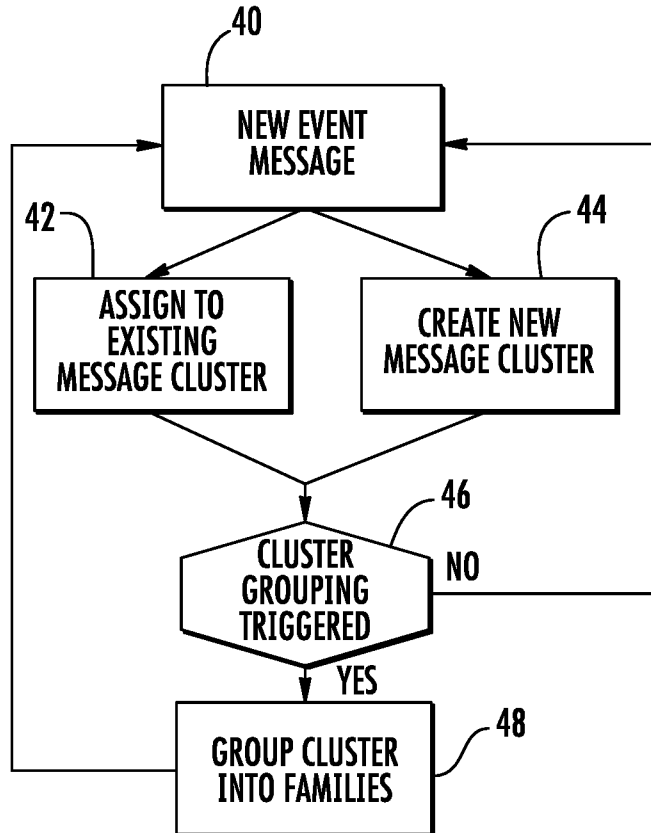
9 in response to not assigning the new log event message to any cluster in
10 the first portion of the entire cluster dictionary in the transient memory, select a
11 subset of clusters of the at least second portion of the cluster dictionary in the
12 persistent database;

13 compare the new log event message to a cluster of the selected subset of
14 clusters; and

15 assign the new log event message to the cluster of the selected subset of
16 clusters based upon the comparison.

1 15. The non-transitory computer-readable medium of claim 14, wherein
2 the instructions are further to direct the processor to form cluster families by
3 grouping message clusters based upon a similarity of the message clusters,
4 wherein the comparing of the new log event message to the first portion of the
5 overall cluster dictionary comprises comparing the new log event message to
6 each cluster family in a predefined order of cluster families and for each cluster
7 family, comparing the new log event message to each message cluster of the
8 cluster family in an order based on an age of each message cluster of the cluster
9 family..

1/11

**FIG. 1****FIG. 2**

100	105	106	104
1	2008-02-06 14:35:16	UNEXPECTED FAILURE WHILE TRYING TO PING USER SESSION #55555 THE SESSION AUTHENTICATION HAS FAILED	
1	2008-02-06 14:35:17	FAILED TO RETRIEVE THE META DATA OF PROJECT 'NULL0' THE SESSION AUTHENTICATION HAS FAILED	
1	2008-02-06 14:35:19	FAILED TO GET LICENSES FOR PROJECT SESSION THE SESSION AUTHENTICATION HAS FAILED	
1	2008-02-06 14:35:19	ERROR PROCESSING REQUEST FROM 192.111.22.33 DATA STARTS WITH 0000000230 CONSTSTR DOWNLOAD	
2	2008-02-06 14:35:39	UNEXPECTED FAILURE WHILE TRYING TO PING USER SESSION #44444 THE SESSION AUTHENTICATION HAS FAILED	
2	2008-02-06 14:35:40	FAILED TO RETRIEVE THE META DATA OF PROJECT 'NULL1' THE SESSION AUTHENTICATION HAS FAILED	
3	2008-02-06 14:35:41	UNEXPECTED FAILURE WHILE TRYING TO PING USER SESSION #33333 THE SESSION AUTHENTICATION HAS FAILED	
3	2008-02-06 14:35:41	FAILED TO GET LICENSES FOR PROJECT SESSION THE SESSION AUTHENTICATION HAS FAILED	
2	2008-02-06 14:50:08	FAILED TO GET LICENSES FOR PROJECT SESSION THE SESSION AUTHENTICATION HAS FAILED	
2	2008-02-06 14:50:09	ERROR PROCESSING REQUEST FROM 192.111.22.33 DATA STARTS WITH 0000000140 CONSTSTR DOWNLOAD	
3	2008-02-06 14:50:11	FAILED TO RETRIEVE THE META DATA OF PROJECT 'NULL3' THE SESSION AUTHENTICATION HAS FAILED	
3	2008-02-06 14:50:14	ERROR PROCESSING REQUEST FROM 192.111.22.33 DATA STARTS WITH 00000005120 CONSTSTR DOWNLOAD	
107			

FIG. 3

107	109		
1	2008-02-06 14:35:16	UNEXPECTED FAILURE WHILE TRYING TO PING USER SESSION #55555 THE SESSION AUTHENTICATION HAS FAILED	
2	2008-02-06 14:35:39	UNEXPECTED FAILURE WHILE TRYING TO PING USER SESSION #44444 THE SESSION AUTHENTICATION HAS FAILED	
3	2008-02-06 14:35:41	UNEXPECTED FAILURE WHILE TRYING TO PING USER SESSION #33333 THE SESSION AUTHENTICATION HAS FAILED	
1	2008-02-06 14:35:17	FAILED TO RETRIEVE THE META DATA OF PROJECT 'NULL0' THE SESSION AUTHENTICATION HAS FAILED	
2	2008-02-06 14:35:40	FAILED TO RETRIEVE THE META DATA OF PROJECT 'NULL1' THE SESSION AUTHENTICATION HAS FAILED	
3	2008-02-06 14:50:11	FAILED TO RETRIEVE THE META DATA OF PROJECT 'NULL3' THE SESSION AUTHENTICATION HAS FAILED	
1	2008-02-06 14:35:19	FAILED TO GET LICENSES FOR PROJECT SESSION THE SESSION AUTHENTICATION HAS FAILED	
2	2008-02-06 14:35:41	FAILED TO GET LICENSES FOR PROJECT SESSION THE SESSION AUTHENTICATION HAS FAILED	
3	2008-02-06 14:50:08	FAILED TO GET LICENSES FOR PROJECT SESSION THE SESSION AUTHENTICATION HAS FAILED	
1	2008-02-06 14:35:19	ERROR PROCESSING REQUEST FROM 192.111.22.33 DATA STARTS WITH 0000000230 CONSTSTR DOWNLOAD	
2	2008-02-06 14:50:09	ERROR PROCESSING REQUEST FROM 192.111.22.33 DATA STARTS WITH 0000000140 CONSTSTR DOWNLOAD	
3	2008-02-06 14:50:14	ERROR PROCESSING REQUEST FROM 192.111.22.33 DATA STARTS WITH 00000005120 CONSTSTR DOWNLOAD	

FIG. 4

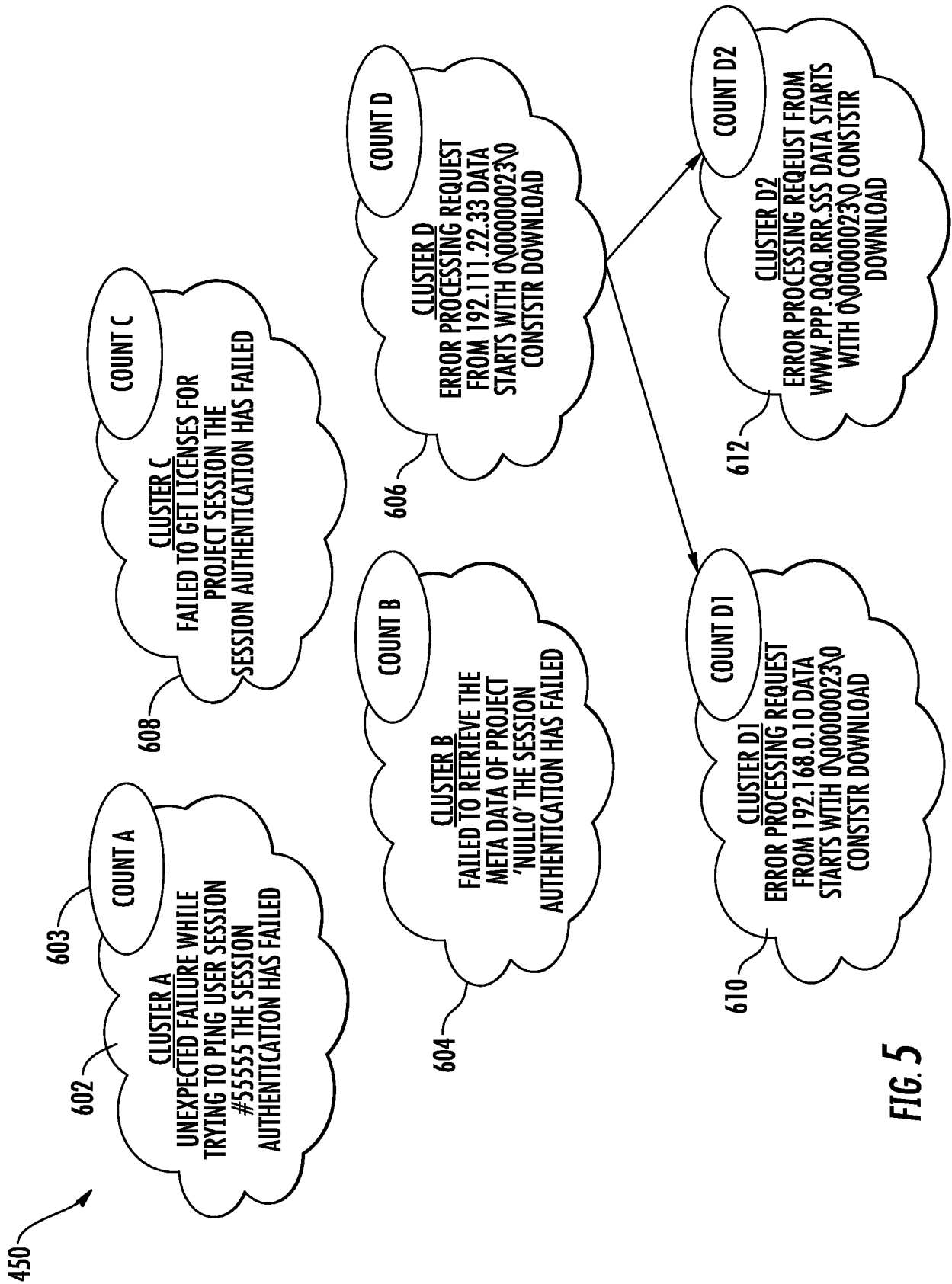


FIG. 5

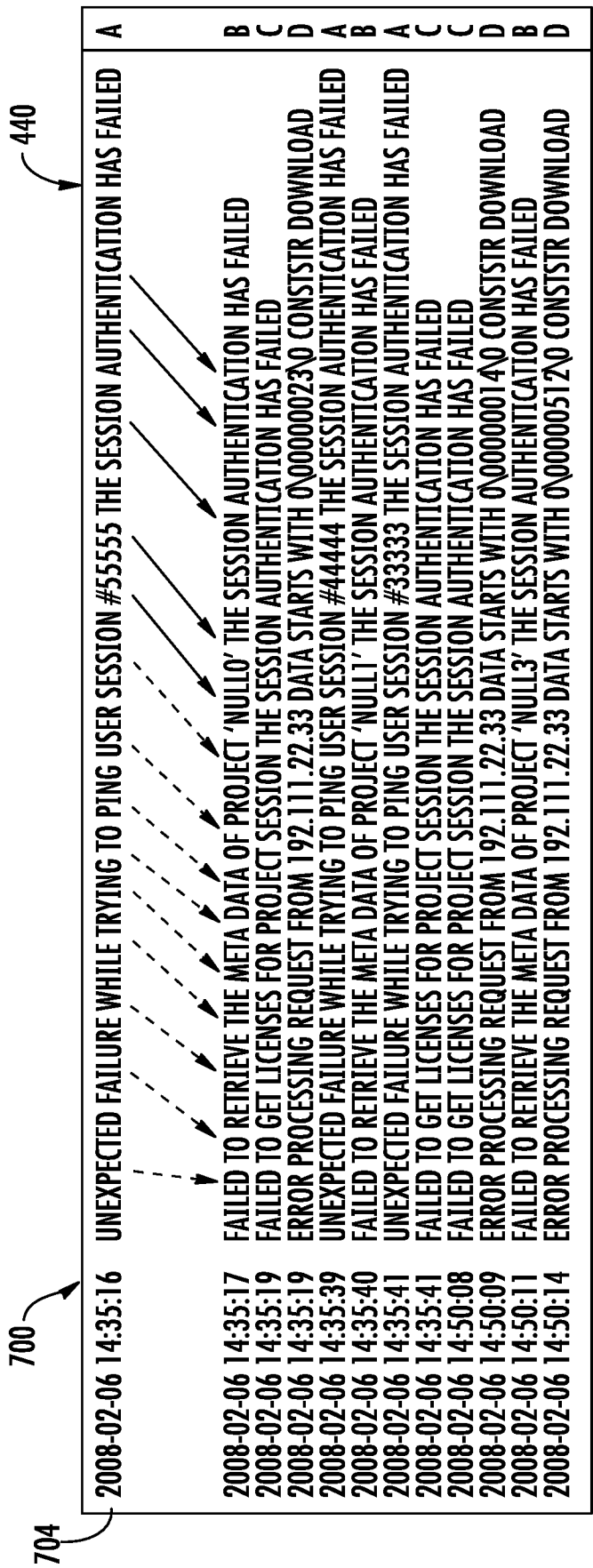


FIG. 6

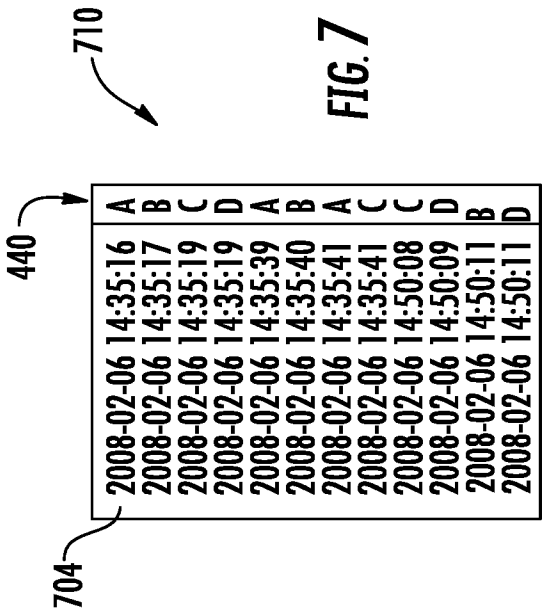
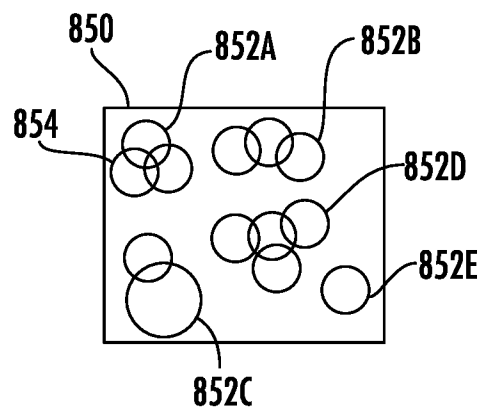
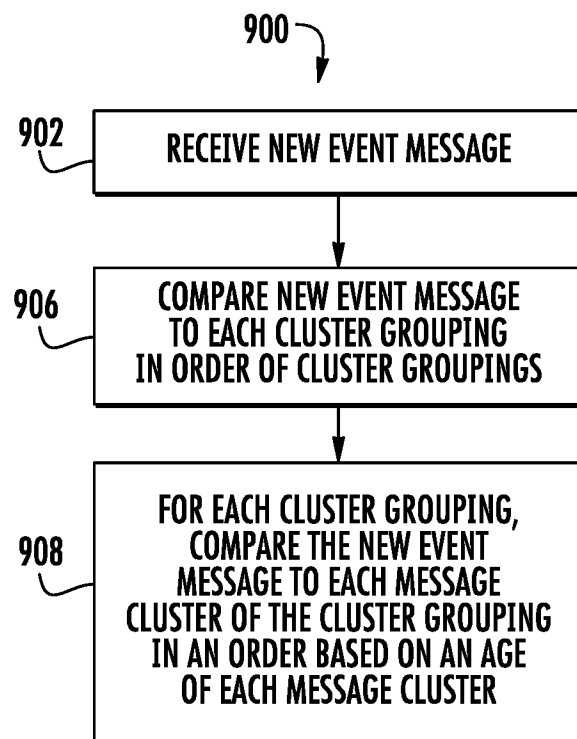


FIG. 7

5/11

**FIG. 8****FIG. 9**

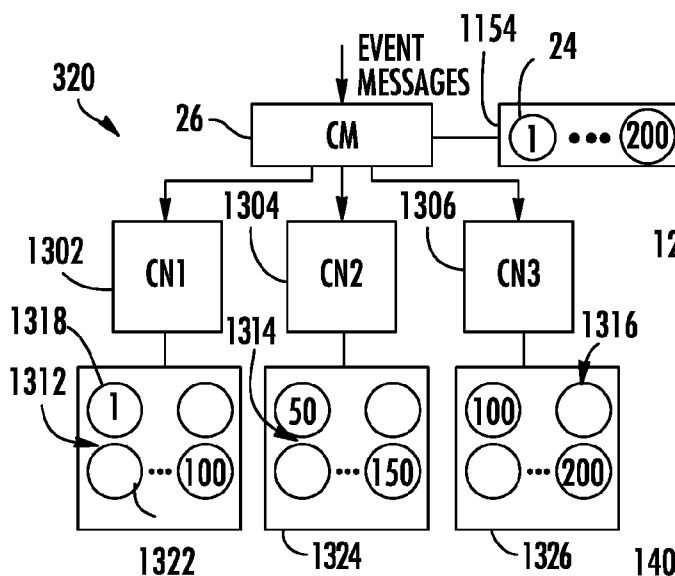
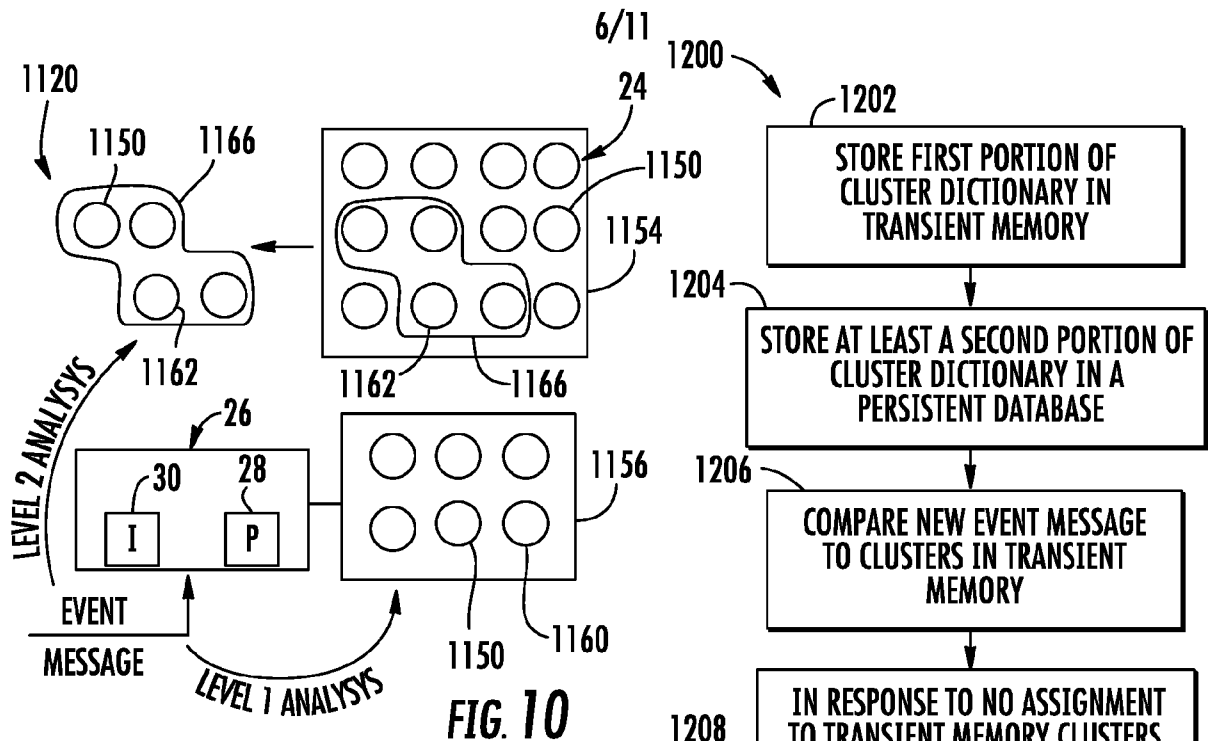
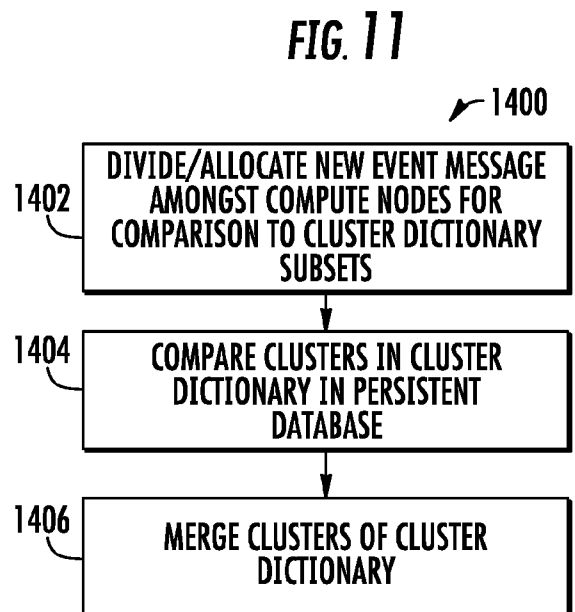
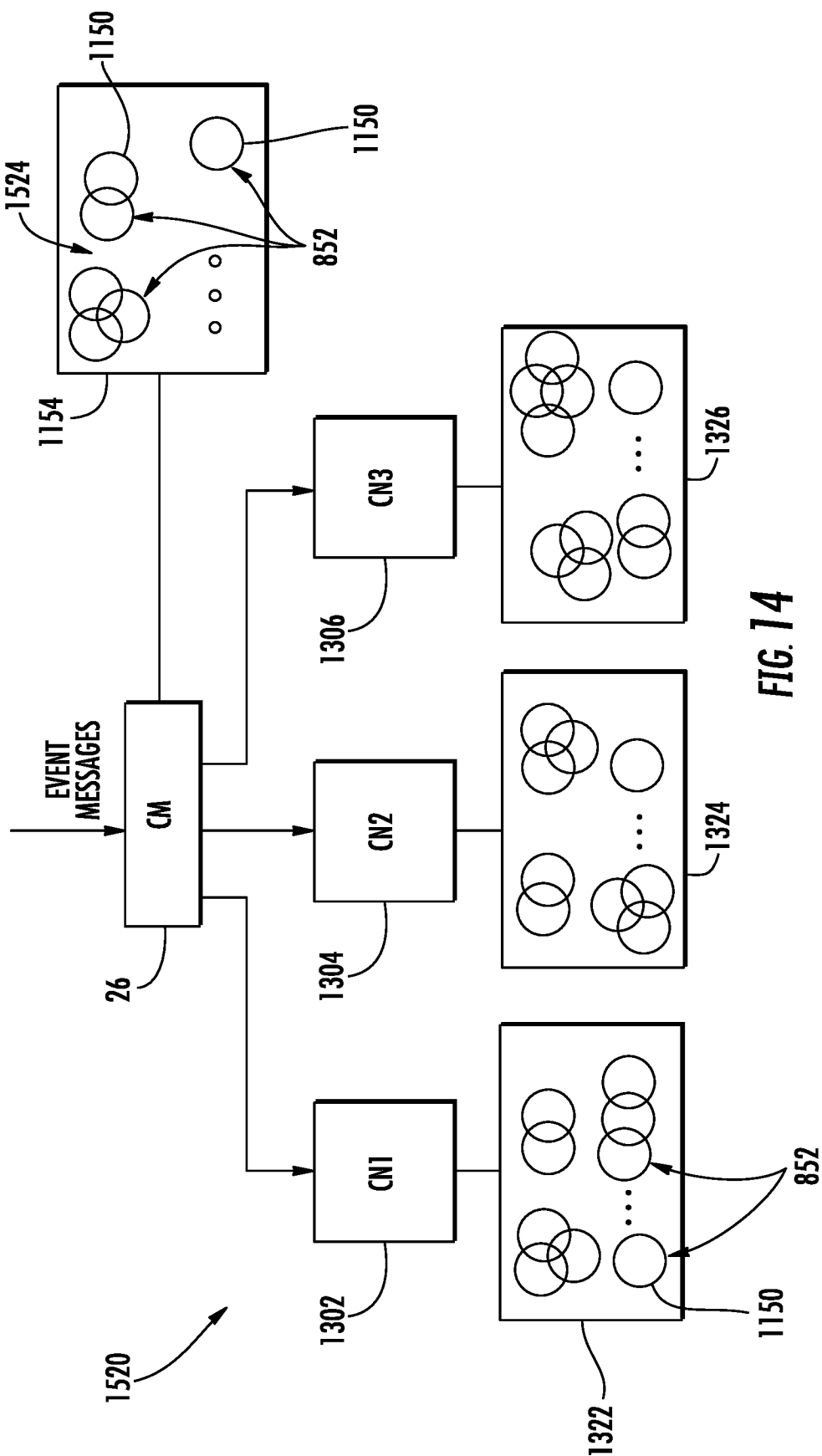


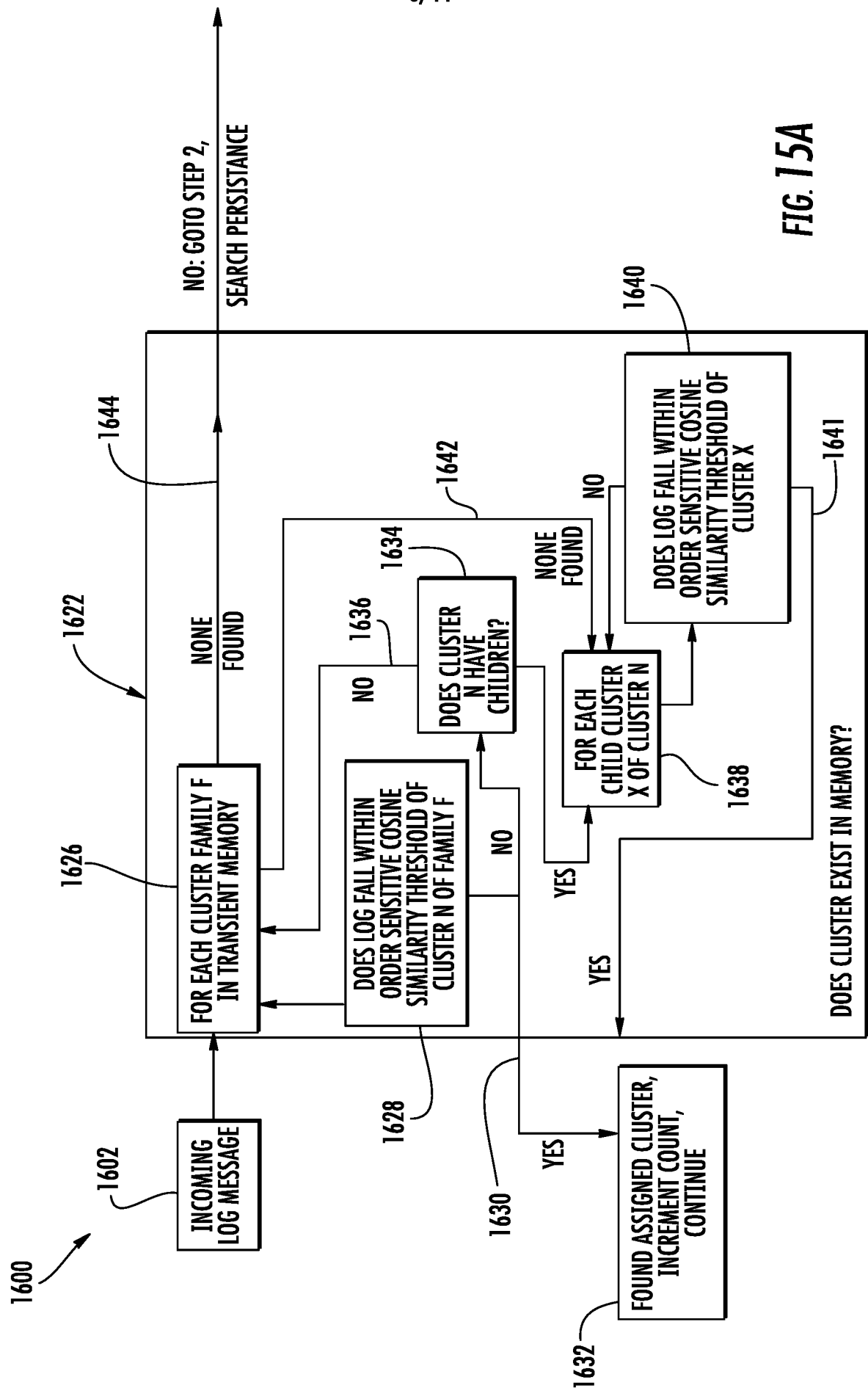
FIG. 12



7/11



8/11



9/11

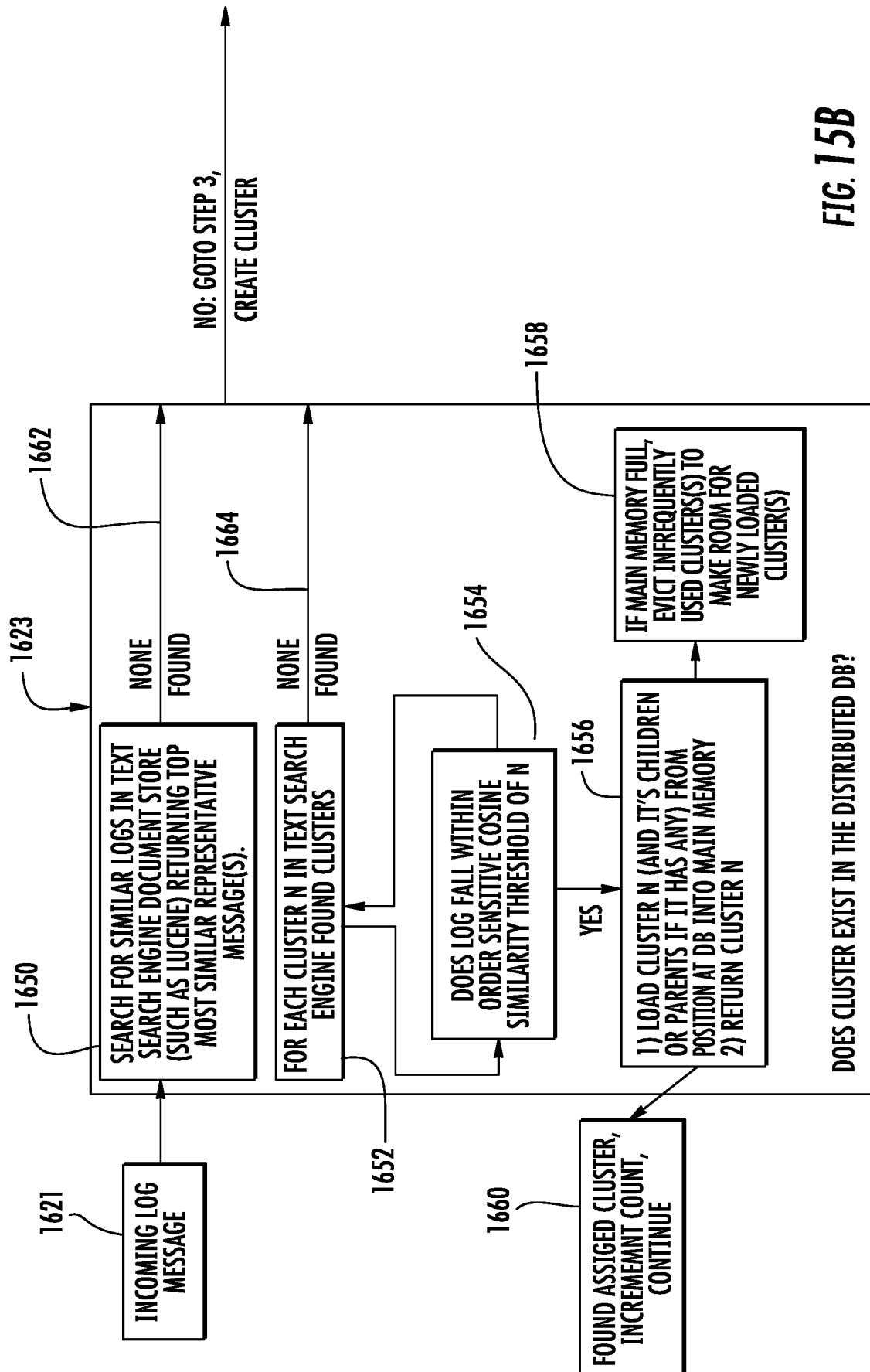


FIG. 15B

10/11

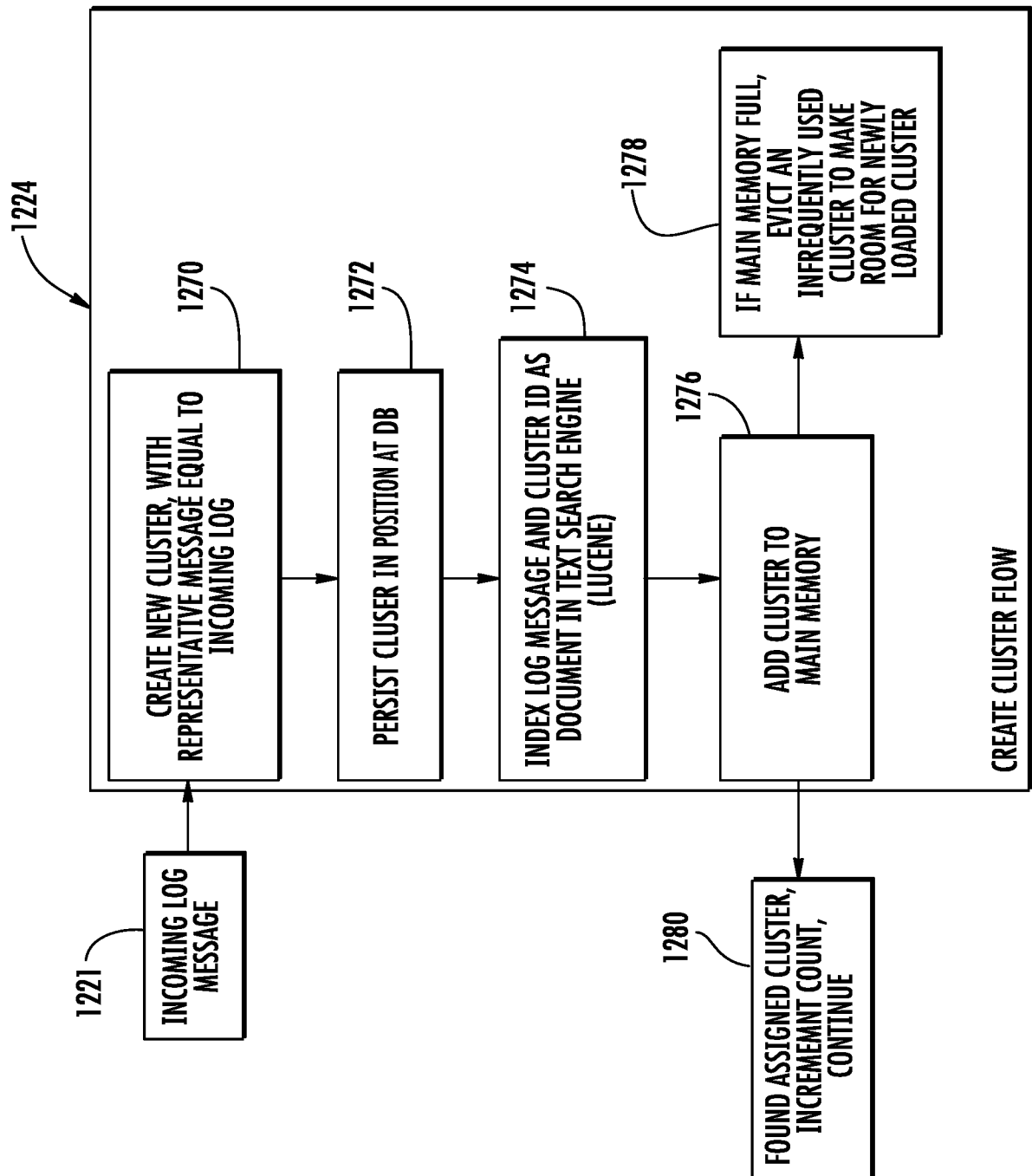


FIG. 15C

11/11

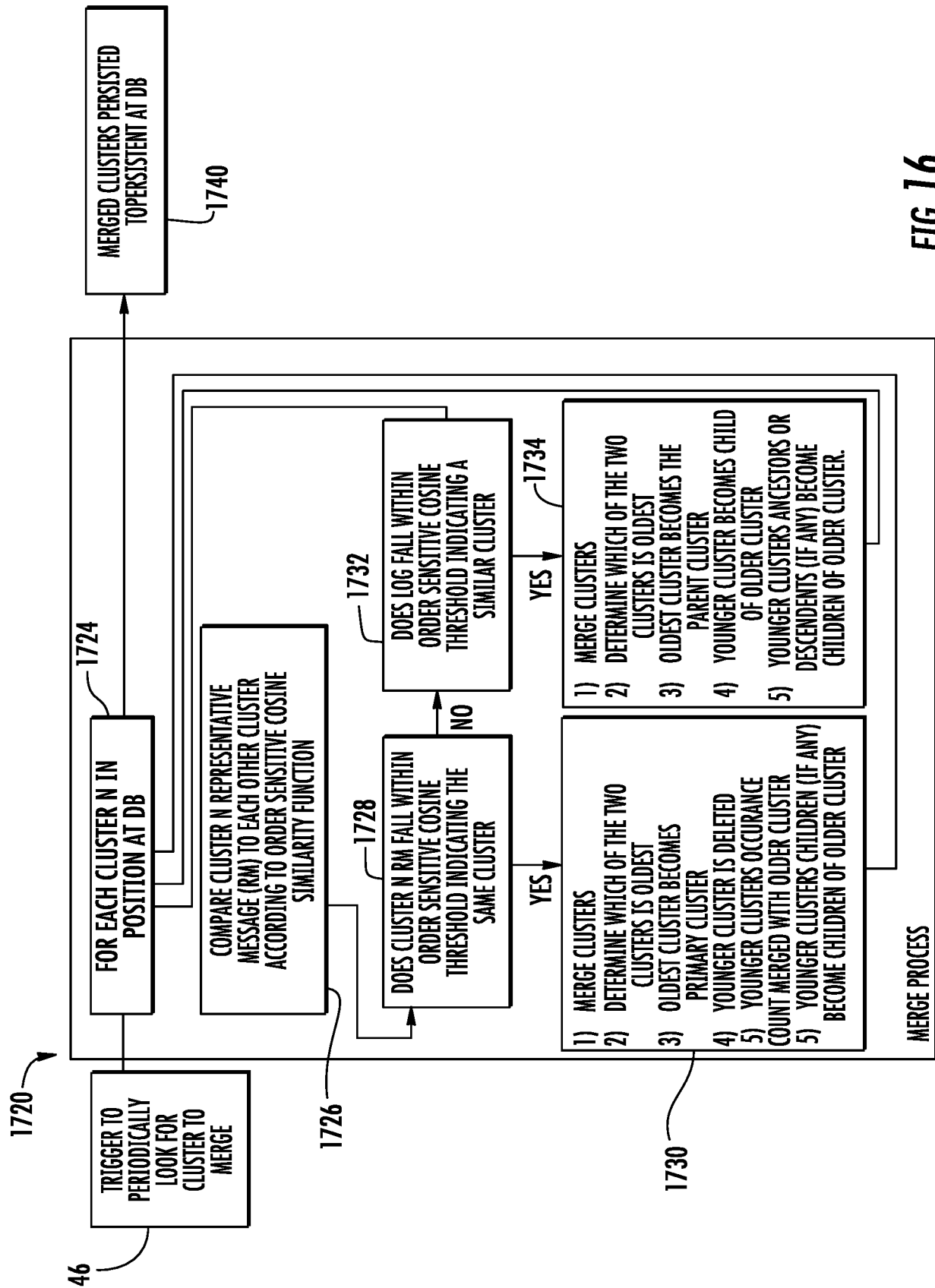


FIG. 16

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2016/015848**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; H04L 12/26; H04L 12/66; G06F 11/08; G06F 9/46; G06K 9/00

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: event, message, cluster, group, merge, log, memory, compare, 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 2013-0042147 A1 (TOSHIO TONOUCHI) 14 February 2013 See paragraphs [0038]-[0039], [0050], [0062]-[0064], [0066] and [0077]; claims 5 and 7-8; and figures 1-3 and 8.	1-5, 8-15
Y		6-7
Y	US 2015-0370799 A1 (VMWARE, INC.) 24 December 2015 See paragraphs [0062] and [0069]; and figure 14C.	6-7
A	US 2014-0334739 A1 (XYRATEX TECHNOLOGY LIMITED) 13 November 2014 See paragraphs [0229]-[0232]; and figure 5.	1-15
A	US 2010-0153397 A1 (ALBERT B. BARABAS et al.) 17 June 2010 See paragraphs [0039]-[0040]; claim 40; and figure 2.	1-15
A	WO 2007-016412 A2 (PORTAL SOFTWARE, INC.) 08 February 2007 See paragraphs [0034]-[0036]; and figure 3.	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

28 October 2016 (28.10.2016)

Date of mailing of the international search report

28 October 2016 (28.10.2016)

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

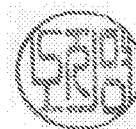


Facsimile No. +82-42-481-8578

Authorized officer

NHO, Ji Myong

Telephone No. +82-42-481-8528



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2016/015848

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2013-0042147 A1	14/02/2013	US 9021304 B2 WO 2011-111599 A1	28/04/2015 15/09/2011
US 2015-0370799 A1	24/12/2015	None	
US 2014-0334739 A1	13/11/2014	GB 2513885 A	12/11/2014
US 2010-0153397 A1	17/06/2010	EP 1197876 A2 EP 1197876 A3 JP 2002-169718 A JP 2012-138110 A JP 2014-038657 A JP 5292489 B2 JP 5425355 B2 JP 5844333 B2 US 2003-0233370 A1 US 2004-0267807 A1 US 2010-0191705 A1 US 7587428 B2 US 7689560 B2 US 8489567 B2 US 9189536 B2	17/04/2002 16/04/2003 14/06/2002 19/07/2012 27/02/2014 18/09/2013 26/02/2014 13/01/2016 18/12/2003 30/12/2004 29/07/2010 08/09/2009 30/03/2010 16/07/2013 17/11/2015
WO 2007-016412 A2	08/02/2007	AU 2006-263656 A1 AU 2006-275665 A1 CA 2613701 A1 CA 2616194 A1 CA 2616194 C CN 101233497 A CN 101233497 B CN 102100027 A EP 1935152 A2 EP 1938193 A2 EP 1938193 A4 JP 2008-547361 A JP 2009-504030 A JP 4664410 B2 US 2007-0198283 A1 US 8117358 B2 WO 2007-002841 A2 WO 2007-002841 A3 WO 2007-016412 A3	04/01/2007 08/02/2007 04/01/2007 08/02/2007 17/02/2015 30/07/2008 30/01/2013 15/06/2011 25/06/2008 02/07/2008 04/08/2010 25/12/2008 29/01/2009 06/04/2011 23/08/2007 14/02/2012 04/01/2007 23/04/2009 12/07/2007