



(51) International Patent Classification:
G06F 16/61 (2019.01)

(21) International Application Number:
PCT/CN2019/081492

(22) International Filing Date:
04 April 2019 (04.04.2019)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **ALIBABA GROUP HOLDING LIMITED**
[—/CN]; Fourth Floor, One Capital Place, P.O. Box 847,
George Town, Grand Cayman (KY).

(72) Inventors: **FU, Chen**; 10322 Alpine Dr. Apt B, Cuperti-
no, California 95014 (US). **SHEN, Chunhui**; Building 8,

No.969 West Wen Yi Road, Yu Hang District, Hangzhou,
Zhejiang 311121 (CN).

(74) Agent: **BEIJING TSINGYUANHUI INTELLECTUAL
PROPERTY LAW FIRM**; Room 362, 3rd Floor, Suzhou
Street No.1, Haidian District, Beijing 100080 (CN).

(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, DJ, DK, DM, DO,
DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN,
HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP,
KR, KW, 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.

(54) Title: SPLIT-KEY ESTIMATION METHOD FOR TABLE PARTITION IN DISBTRIBUTED DATA STORAGE SYSTEMS

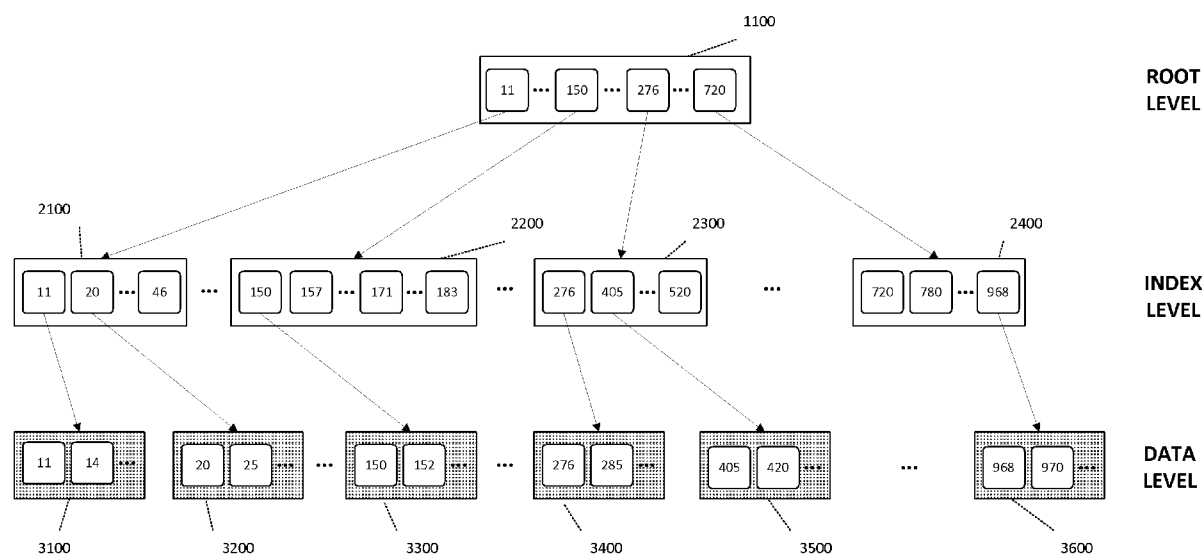


FIG. 2

(57) Abstract: A distributed data storage system incorporating a light-weight method for split key estimation is provided. The distributed data storage system comprises partitions represented by a rooted tree structure comprising a root-level index block(1100), a set of intermediate-level index blocks(2100,2200,2300,2400) organized into one or more index levels, and a set of data-level blocks(3100,3200,3300,3400,3500,3600) having row-key pairings that corresponds to indexes provided in the set of intermediate-level index blocks(2100,2200,2300,2400) and the root-level index block(1100). The method comprises obtaining a start key and an end key based on a request to split a partition, determining a start block based on a search of the rooted tree structure using the start key, determining an end block based on a search of the rooted tree structure using the end key, determining a middle block of the rooted tree structure based on the start block and the end block, determining a split key from the middle block, and splitting the partition

(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).

Published:

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

SPLIT-KEY ESTIMATION METHOD FOR TABLE PARTITION IN DISBTRIBUTED DATA STORAGE SYSTEMS

BACKGROUND

[001] In distributed data storage systems, it is often not feasible to store all the data in a
5 single table. Instead, such systems divide a data table into distinct parts called “partitions.” When
the data volume of a table grows, there is a need to split partitions in order to keep partition sizes
manageable. Since the distribution of data is often uneven in a distributed data storage system,
finding at least one split key that divides a key range evenly can be both challenging and time-
consuming. There exists a need for a light-weight estimation method to determine one or more
10 split keys.

SUMMARY

[002] Embodiments of the present disclosure provide a method for split key estimation
in distributed data storage system. The distributed data storage system comprises partitions
represented by a rooted tree structure comprising a root-level index block, a set of intermediate-
15 level index blocks organized into one or more index levels, and a set of data-level blocks having
row-key pairings that corresponds to indexes provided in the set of intermediate-level index
blocks. The method comprises obtaining a start key and an end key based on a request to split a
partition, determining a start block based on a search of the rooted tree structure using the start
key, determining an end block based on a search of the rooted tree structure using the end key,
20 determining a middle block of the rooted tree structure based on the start block and the end
block, determining a split key from the middle block, and splitting the partition based on the
determined split key.

[003] Embodiments of the present disclosure further provide a distributed data storage system comprising a plurality of partitions and a processor. The plurality of partitions are configured to be organized in the form of a rooted tree structure comprising a root-level index block, intermediate-level index blocks organized into one or more index levels, and a set of data-level blocks having row-key pairings that corresponds to indexes provided in the intermediate-level index blocks. The processor is configured to obtain a start key and an end key based on a request to split a partition, to determine a start block and an end block based on a search of the rooted tree structure using the start key and the end key respectively, to determine a middle block based on the start block and the end block, to determine a split key from the middle block, and to split the partition based on the determined split key.

[004] Embodiments of the present disclosure further provide a non-transitory computer readable medium that stores a set of instructions that are executable by one or more processors of an apparatus to perform a method. The method comprises obtaining a start key and an end key, determining a start block and an end block based on a search of a rooted tree structure using the obtained start key and the end key, determining a middle block based on the start block and the end block, determining a split key from the middle block, and splitting the partition based on the determined split key.

BRIEF DESCRIPTION OF THE DRAWINGS

[005] Embodiments and various aspects of the present disclosure are illustrated in the following detailed description and the accompanying figures. Various features shown in the figures are not drawn to scale.

[006] **FIG. 1A** is a schematic diagram illustrating an exemplary distributed data storage system comprising servers having partitions, according to some embodiments of the present disclosure.

[007] **FIG. 1B** is a schematic diagram illustrating an exemplary server of a distributed data storage system, according to some embodiments of the present disclosure.

[008] **FIG. 2** is a schematic diagram of an exemplary data organization inside a partition using a rooted tree in a distributed data storage system, according to some embodiments of the present disclosure.

[009] **FIG. 3** is a flow diagram of an exemplary method for light-weight split key estimation in a distributed data storage system, according to some embodiments of the present disclosure.

[010] **FIG. 4** is a flow diagram of an exemplary method for light-weight split key estimation incorporating an estimation threshold in a distributed data storage system, according to some embodiments of the present disclosure.

[011] **FIG. 5** is a flow diagram of an exemplary method for light-weight split key estimation incorporating a start offset and an end offset in a distributed data storage system, according to some embodiments of the present disclosure.

DETAILED DESCRIPTION

[012] The following description refers to the accompanying drawings in which the same numbers in different drawings represent the same or similar elements unless otherwise represented. The implementations set forth in the following description of exemplary embodiments do not represent all implementations consistent with the disclosure. Instead, they

are merely examples of apparatuses and methods consistent with aspects related to the disclosure as recited in the appended claims.

[013] In distributed data storage systems, a data table is divided into many partitions. A partition often includes rows whose keys are within a range. For example, in a table where the
5 keys are integers, a partition may contain rows with keys in [4096, 8191]. Several partitions are assigned to a server, where read and write requests are served from. Partitions may be re-assigned to other servers for load balancing or recovery. Keeping partition sizes under a certain threshold is important for partition re-assignment to work efficiently.

[014] When the size of a partition grows, there is a need to split large partitions into
10 smaller partitions in order to keep partition sizes manageable. Partitions having large sizes slow down the re-assignment process, which leads to slow recovery, difficulty in load balancing, and potential risks of running out of disk space for the underlying server.

[015] When a partition is split into multiple smaller partitions, the original big partition is referred to as a “parent partition,” and the resulting smaller partitions are referred to as “child
15 partitions.” Some systems divide one parent partition into two child partitions with a single “split key” that divides the parent’s key range into two smaller ranges. For example, a split key 1000 divides a range [0, 1999] into two smaller ranges [0, 999] and [1000, 1999]. Other systems use n split keys to split a parent partition into $n+1$ child partitions.

[016] When a parent partition is split, it is desirable that the child partitions are
20 approximately the same size. Some systems use a simple, arithmetic center of the parent partition’s range as the split key. For example, for a parent partition with a range of [0, 1999], some conventional systems choose to use 1000 as the split key. Unfortunately, the arithmetic center of the parent partition’s range often cannot create an even split to create child partitions of

approximately the same size because the distribution of the keys is unknown. Referring to the previous example where 1000 is used as the split key to split the parent partition with the range [0, 1999], if most of the keys are greater than 1000, the child partitions end up with highly uneven sizes. Other conventional systems try to scan all the data in the parent partition to find split keys for even splitting. This method, however, ignores the fact that when a system decides to split a parent partition, the system has already determined that the parent partition was experiencing heavy traffic and needs to be split immediately to alleviate the heavy traffic. Scanning all the data in the parent partition is, therefore, highly inefficient and creates large overhead in the system.

[017] Another problem with conventional systems is that a single mid-key is only good for splitting a partition into two. When the data in the parent partition is growing at a fast pace, the system desires to split the parent partition into more than two child partitions.

[018] Embodiments of the present disclosure mitigate this problem using a new light-weight method and system to estimate split keys in order to produce even splits. **FIG. 1A** is a schematic diagram illustrating an exemplary distributed data storage system comprising servers having partitions, according to some embodiments of the present disclosure. According to **FIG. 1A**, exemplary distributed data storage system 100 comprises a plurality of servers 1-K, with each server having 10 different partitions. For example, server 1 comprises partitions A1-A10, server 2 comprises partitions B1-B10, and server K comprises of partitions K1-K10. Distributed data storage system 100 allows each partition to be re-assigned to a different server. It is appreciated that one or more of these servers can incorporate fast partition splitting.

[019] **FIG. 1B** is a schematic diagram illustrating an exemplary server 110 of a distributed data storage system, according to some embodiments of the present disclosure.

According to **FIG. 1B**, server 110 comprises a bus 112 or other communication mechanism for communicating information, and one or more processors 116 communicatively coupled with bus 112 for processing information. Processors 116 can be, for example, one or more microprocessors.

5 [020] Server 110 further comprises storage devices 114, which may include random access memory (RAM), read only memory (ROM), and data storage systems comprised of partitions. Storage devices 114 can be communicatively coupled with processors 116 via bus 112. Storage devices 114 may include a main memory, which can be used for storing temporary variables or other intermediate information during execution of instructions to be executed by
10 processors 116. Such instructions, after being stored in non-transitory storage media accessible to processors 116, render server 110 into a special-purpose machine that is customized to perform operations specified in the instructions.

 [021] Server 110 can transmit data to or communicate with another server 130 through a network 122. Network 122 can be a local network, an internet service provider, internet, or any
15 combination thereof. Communication interface 118 of server 110 is connected to network 122. In addition, server 110 can be coupled via bus 112 to peripheral devices 140, which comprises displays (e.g., cathode ray tube (CRT), liquid crystal display (LCD), touch screen, etc.) and input devices (e.g., keyboard, mouse, soft keypad, etc.).

 [022] Server 110 can be implemented using customized hard-wired logic, one or more
20 ASICs or FPGAs, firmware, or program logic that in combination with the server causes server 110 to be a special-purpose machine.

 [023] The term “non-transitory media” as used herein refers to any non-transitory media storing data or instructions that cause a machine to operate in a specific fashion. Such non-

transitory media can comprise non-volatile media and/or volatile media. Non-transitory media include, for example, optical or magnetic disks, dynamic memory, a floppy disk, a flexible disk, hard disk, solid state drive, magnetic tape, or any other magnetic data storage medium, a CD-ROM, any other optical data storage medium, any physical medium with patterns of holes, a RAM, a PROM, and EPROM, a FLASH-EPROM, NVRAM, flash memory, register, cache, any
5 other memory chip or cartridge, and networked versions of the same.

[024] Various forms of media can be involved in carrying one or more sequences of one or more instructions to processors 116 for execution. For example, the instructions can initially be carried out on a magnetic disk or solid state drive of a remote computer. The remote computer
10 can load the instructions into its dynamic memory and send the instructions over a telephone line using a modem. A modem local to server 110 can receive the data on the telephone line and use an infra-red transmitter to convert the data to an infra-red signal. An infra-red detector can receive the data carried in the infra-red signal and appropriate circuitry can place the data on bus 112. Bus 112 carries the data to the main memory within storage devices 114, from which
15 processors 116 retrieves and executes the instructions.

[025] Inside each partition, data is organized in a rooted tree structure comprising multiple runs sorted by row keys. **FIG. 2** is a schematic diagram of an exemplary data organization using a rooted tree in a distributed data storage system, according to some embodiments of the present disclosure. According to **FIG. 2**, the rooted tree comprises three
20 levels: a root level, an index level, and a data level, although any number of levels may be used.

[026] The root level and the index level contain only index blocks, which are colored in white. The root level comprises index block 1100, which comprise index entries with entry keys 11, 150, 276 and 720. The pointer in an index entry can either point to a data block in the data

level or another index block in a lower index level of the rooted tree structure. The entry key in the index entry is the first key of the block the entry points to. For example, index entry with the entry key 11 in index block 1100 points to the index block 2100 whose first entry key is 11.

[027] The index level comprises index blocks 2100, 2200, 2300 and 2400. Each index block comprises one or more index entries, and each index entry has an entry key and a pointer. For example, index block 2100 comprises index entries with keys 11, 20, and 46. There can be multiple index levels when files are large. The pointer in an index entry can either point to a data block in the data level or another index block in a lower index level of the rooted tree structure. The entry key in an index entry is the first key of the block the entry points to. For example, index entry with the entry key 276 in index block 2300 points to a data block 3400 whose first entry key is 276. The data level contains only data blocks, which are colored in grey. The data level comprises data blocks 3100, 3200, 3300, 3400, 3500 and 3600. Each data block comprises row-key pairings, which are rows of data labeled by keys.

[028] In some embodiments, other than index block 1100 at the root level, each index block has the same size except the index block with the largest entry key at each level. In other words, if N is an integer and there are N-number of index blocks on the same level, index blocks 1 through N-1 have the same number of index entries. For example, index blocks 2100, 2200, and 2300 have the same or substantially the same size, and index block 2400 (which has the largest entry keys at the index level) may have a smaller size compared to other index blocks. In some embodiments, if the distributed data storage system has multiple index levels, all index blocks across all levels except those with the largest entry key at each level have the same size. In some embodiments, the number of index entries in each index block ranges from hundreds to

thousands. In some embodiments, the amount of data in each data block can range from 64KB to 128KB.

[029] Similarly, all data blocks have the same size except the data block with the largest entry key. In other words, if X is an integer and there are X-number of data blocks, data blocks 1 through X-1 have the same number of row-key pairings. For example, data blocks 3100, 3200, 3300, 3400 and 3500 have the same size, and data block 3600 may have a smaller size compared to other data blocks. Each data entry within a data block points to a row of data having a row key that is the same as the entry key of the data entry. For example, data entry with entry key 11 in data block 3100 points to a row of data whose row key is 11.

[030] Embodiments of the present disclosure provides a light-weight method for split-key estimation. **FIG. 3** is a flow diagram of an exemplary method for light-weight split key estimation in a distributed data storage system, according to some embodiments of the present disclosure. According to **FIG. 3**, method 10000 comprises steps 10010, 10020, 10030, 10040, 10050 and 10060. It is appreciated that method 10000 can be performed by a distributed data storage system (e.g., distributed data storage system 100 of **FIG. 1A**) or by one or more servers (e.g., exemplary server 110 of **FIG. 1B**).

[031] In step 10010, the distributed data storage system obtains a start key and an end key. The start key and the end key form a key range in which method 10000 determines an estimated mid key as the split key for partition splitting. For example, as shown in **FIG. 2**, the distributed data storage system obtains a start key 11 and an end key 405.

[032] In step 10020, the distributed data storage system searches the root level of the rooted tree structure for the index entry associated with the start key and the end key. For example, as shown in **FIG. 2**, the distributed data storage system searches root level 1100 and

finds index entry with the entry key 11, which is associated with start key 11. Since root level 1100 does not contain an index entry with entry key 405, the distributed data storage system finds the closest entry key that is smaller than the end key 405, which is 276.

[033] In step 10030, the distributed data storage system determines a start block according to the start key. The start block has a key range in which the start key is contained. A key range of a block is from the first entry in the block to the first entry of the next data block. For example, as shown in **FIG. 2**, if the index block after index block 2100 has a first index entry 51, index block 2100 has a key range of 11 to 50. The distributed data storage system determines that index block 2100 is the start block because index block 2100 contains a key range of 11 to 50, which covers the start key 11.

[034] In step 10040, the distributed data storage system determines an end block according to the end key. The end block has a key range in which the end key is contained. For example, as shown in **FIG. 2**, if the index block after index block 2300 has a first index entry 541, index block 2300 has a key range of 276 to 540. The distributed data storage system determines that index block 2300 is the end block because index block 2300 has a key range of 276 to 540, which covers the end key 405.

[035] In step 10050, the distributed data storage system determines a middle block according to the start block and the end block. In some embodiments, the middle block is the block located in the center between the start block and the end block. In a scenario where there are an even number of blocks between the start block and the end block, the middle block is the block located to the immediate left of the center between the start block and the end block (although, depending on the configuration, the immediate right of the center can also be selected as the center block). For example, as shown in **FIG. 2**, there are five index blocks between start

block 2100 and end block 2300. The index block located in the center is the third block, which is index block 2200. The distributed data storage system determines index block 2200 as the middle block.

[036] In step 10060, the distributed data storage system determines a split key from the middle block. In some embodiments, the split key is the index located in the center of the middle block. For example, as shown in **FIG. 2**, index block 2200 is the middle block and it has 13 index entries. The split key is the entry key of the index entry at the 7th location, which is 171.

[037] In some embodiments of the present disclosure, all index blocks except the index block at the root level have the same number of index entries. When method 10000 determines the block located in the center between the start block and the end block, method 10000 is taking advantage of the fact that all index blocks are of the same size or substantially the same size.

Therefore, the block located in the center is generally a good estimation of where the split key should be located to lead to an even split, resulting in partitions having similar amount of data, and the entry key located at the center of the middle block is generally a good estimation of the split key itself. If the distributed data storage system is not satisfied with the accuracy of the current estimation, the distributed data storage system may obtain a more accurate result by searching one level below. In addition to accuracy, the method makes an estimation for split key without having to traverse through all the row keys in a partition, thereby improving the efficiency of partition split.

[038] In some embodiments of the present disclosure, the distributed data storage system can achieve an M-way split where M is a power of 2. For example, if the distributed data storage system obtains a start key 11 and an end key 450 and determines that 171 is the split key using method 10000, the distributed data storage system may initiate additional estimation

processes for two more split keys where the search ranges are [11, 171] and [171, 450]. In the first estimation process, the distributed data storage system obtains a start key 11 and an end key 171. In the second estimation process, the distributed data storage system obtains a start key 171 and an end key 450. Having determined the split keys for both searches using method 10000, the distributed data storage system can perform a 4-way split based on the three split keys, including split key 171.

[039] Some embodiments of the present disclosure incorporate an estimation threshold to determine whether the current level being searched is satisfactory for accuracy. **FIG. 4** is a flow diagram of an exemplary method for light-weight split key estimation incorporating an estimation threshold in a distributed data storage system, according to some embodiments of the present disclosure. On the basis of **FIG. 3**, method for split key estimation in **FIG. 4** further comprises additional steps 10015, 10051, 10052 and 10053. It is appreciated that method 10000 can be performed by a distributed data storage system (e.g., distributed data storage system 100 of **FIG. 1A**) or by one or more servers (e.g., exemplary server 110 of **FIG. 1B**).

[040] Step 10015 is performed after step 10010 for obtaining the start and end keys of a rooted tree structure. In step 10015, the distributed data storage system obtains an estimation threshold. The estimation threshold refers to the minimum number of blocks between the start block and the end block that the system accepts before the system stops the estimation process. The estimation threshold may be a preset value, or it may be determined by the system or a user. For example, in the distributed data storage system shown in **FIG. 2**, the distributed data storage system may obtain an estimation threshold of 10.

[041] Step 10051 is performed after step 10050 in which a middle block is determined. In step 10051, the distributed data storage system determines whether the estimation process has

reached a data level. If the distributed data storage system determines that the current level being searched is the data level, the distributed data storage system executes step 10060 and determines a split key in the middle block. If the distributed data storage system determines that the current level being searched is not the data level, the distributed data storage system executes step 10052. For example, as shown **FIG. 2**, when the distributed data storage system reaches the index level and determines that the middle block is 2200, the distributed data storage system determines that the current level being searched is not a data level. As a result, the distributed data storage system moves on to step 10052.

[042] In step 10052, the distributed data storage system determines whether the number of blocks between the start block and the end block is larger than the estimation threshold. If it is larger, the distributed data storage system is satisfied with the level of accuracy in split key estimation provided in the current level. The distributed data storage system then executes step 10060 and determines a split key in the middle block. If the number of blocks is less than or equal to the estimation threshold, the distributed data storage system executes step 10053. For example, as shown in **FIG. 2**, there are five data blocks between the start block 2100 and the end block 2300. If the obtained estimation threshold is 10, the number of blocks in between is smaller than the estimation threshold, and the distributed data storage system moves on to step 10053. If the obtained estimation threshold is 3, the number of blocks in between is larger than the estimation threshold, and the distributed data storage system moves on to step 10060.

[043] In step 10053, the distributed data storage system searches the next lower level in the rooted tree structure. The distributed data storage system then moves on to step 10030 and determines a start block, an end block and a middle block in the new level. For example, as shown in **FIG. 2**, when the distributed data storage system determines that the number of blocks

between the start block 2100 and the end block 2300 is smaller than the estimation threshold, the distributed data storage system searches the level below, which is the data level. The distributed data storage system then moves on to step 10030 and determines a start block, an end block and a middle block in the data level. In the example shown in **FIG. 2**, the distributed data storage system determines that data block 3100 is the new start block for start key 11, and data block 3500 is the new end block for end key 405. If there are 50 data blocks between the start block 3100 and the end block 3500, the middle block is the data block located in the center between the start block and the end block, which is the 25th data block.

[044] The use of the estimation threshold allows the distributed data storage system to make decisions on the tradeoff between accuracy and efficiency in the estimation process. If accuracy is more critical to the system, the estimation threshold can be set to be a very large number, and the distributed data storage system would search multiple levels below the root level. If efficiency is more critical to the system, the estimation threshold can be set to be a smaller number, and the distributed data storage system would search only a few levels. The use of estimation threshold provides the distributed data storage system more flexibility in the estimation process.

[045] Some embodiments of the present disclosure incorporate a start offset and an end offset to achieve a more accurate estimation. **FIG. 5** is a flow diagram of an exemplary method for light-weight split key estimation incorporating a start offset and an end offset in a distributed data storage system, according to some embodiments of the present disclosure. On the basis of **FIG. 3** and **FIG. 4**, method 10000 for split key estimation in **FIG. 5** further comprises steps 10031, 10041 and 10060. It is appreciated that method 10000 can be performed by a distributed

data storage system (e.g., distributed data storage system 100 of **FIG. 1A**) or by one or more servers (e.g., exemplary server 110 of **FIG. 1B**).

[046] Step 10031 is performed after step 10030 in which the start block is determined based on the start key. In step 10031, the distributed data storage system determines a start offset according to the start block and the start key. In some embodiments, the start offset is the location of the index entry whose entry key is closest to the start key in the start block. For example, as shown in **FIG. 2**, the start key is 11 and the index entry with the entry key closest to the start key in the start block 2100 is 11, which is the first index entry in the start block. The start offset is the location of the entry key closest to the start key in the start block. In our example, the location of entry key 11 in the start block 2100 is 1 (e.g., first index entry), and hence the start offset is 1.

[047] Step 10041 is performed after step 10040 in which the end block is determined based on the end key. In step 10041, the distributed data storage system determines an end offset according to the end block and the end key. In some embodiments, the end offset is the location of the index entry whose entry key is closest to the end key in the end block. For example, as shown in **FIG. 2**, the end key is 405 and the index entry with the entry key closest to the end key in the end block 2300 is 405, which is the second index in the end block. The end offset is the location of the entry key closest to the end key in the end block. In our example, the location of entry key 405 in the end block 2300 is 2 (e.g., second index entry), and hence the end offset is 2.

20 In another example, if the end key is 521 and the end block is 2300, the index entry with the entry key closest to the end key in the end block 2300 is 520. If an index block has a size of 13 indices, index entry 520 is the 13th index entry in the end block 2300. Therefore, the end offset is 13.

[048] In step 10060, the distributed data storage system determines a split key in the middle block according to the start offset and the end offset. In some embodiments, the split key is the index in the middle block that has a location equal to the average of the start offset and the end offset. For example, as shown in **FIG. 2**, the start offset is 1 and the end offset is 2. The average of the start offset and the end offset is 1.5, which can be adjusted to an integer of 2. Using the integer 2, the split key can be determined as the second index within the middle block, which in this case is index 157 of middle block 2200. Therefore, the split key is determined to be 157. In another example, the start offset is 1 and the end offset is 13. The average of the start offset and the end offset can be adjusted to 7. The 7th index entry in the middle block 2200 is 171. Therefore, the split key is determined to be 171.

[049] By implementing the start offset and the end offset, the distributed data storage system is able to pinpoint a more accurate location of the split key in the middle block, thereby improving the overall accuracy of the method for split key estimation.

[050] The various example embodiments described herein are described in the general context of method steps or processes, which may be implemented in one aspect by a computer program product, embodied in a computer-readable medium, including computer-executable instructions, such as program code, executed by computers in networked environments. A computer-readable medium may include removeable and nonremoveable storage devices including, but not limited to, Read Only Memory (ROM), Random Access Memory (RAM), compact discs (CDs), digital versatile discs (DVD), etc. Generally, program modules may include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Computer-executable instructions, associated data structures, and program modules represent examples of program code for executing steps of

the methods disclosed herein. The particular sequence of such executable instructions or associated data structures represents examples of corresponding acts for implementing the functions described in such steps or processes.

[051] In the foregoing specification, embodiments have been described with reference
5 to numerous specific details that can vary from implementation to implementation. Certain adaptations and modifications of the described embodiments can be made. Other embodiments can be apparent to those skilled in the art from consideration of the specification and practice of the invention disclosed herein. It is intended that the specification and examples be considered as exemplary only, with a true scope and spirit of the invention being indicated by the following
10 claims. It is also intended that the sequence of steps shown in figures are only for illustrative purposes and are not intended to be limited to any particular sequence of steps. As such, those skilled in the art can appreciate that these steps can be performed in a different order while implementing the same method.

[052] In the drawings and specification, there have been disclosed exemplary
15 embodiments. Many variations and modifications, however, can be made to these embodiments. Accordingly, although specific terms are employed, they are used in a generic and descriptive sense only and not for purposes of limitation, the scope of the embodiments being defined by the following claims.

20

WHAT IS CLAIMED IS:

1. A method comprising:

obtaining a first key and a second key based on a request to split a partition represented by a rooted tree structure;

5 determining a start block of the rooted tree structure based on a search of the rooted tree structure using the obtained first key;

determining an end block of the rooted tree structure based on a search of the rooted tree structure using the obtained second key;

10 determining a middle block of the rooted tree structure based on the start block and the end block; and

determining a split key from the middle block.

2. The method according to claim 1, wherein the rooted tree structure comprises a root-level index block, a set of intermediate-level index blocks, and a set of data-level blocks having
15 row-key pairings that corresponds to indexes provided in the set of intermediate-level index blocks.

3. The method according to claim 2, wherein the set of intermediate-level index blocks comprises N-number of index blocks and index blocks 1 through N-1 have the same number of
20 indexes, wherein

N is an integer greater than 0.

4. The method according to of claim 2 or 3, wherein the set of data-level blocks comprises X-number of data blocks and data blocks 1 through X-1 have the same number of row-key pairings, wherein

X is an integer greater than 0.

5

5. The method according to any one of claims 2-4, further comprising:

obtaining an estimation threshold;

determining the middle block in response to the estimation threshold being smaller than a number of blocks between the start block and the end block; and

10

searching a next lower level of the rooted tree structure in response to the estimation threshold being larger than the number of blocks between the start block and the end block.

6. The method of claim 5, wherein the next level of the rooted tree structures comprises another set of intermediate-level index blocks that are lower in the rooted tree structure than the set of intermediate-level index blocks, wherein the set of data-level blocks has row-key pairings that corresponds to indexes provided in the other set of intermediate-level index blocks.

15

7. The method according to any one of claims 1-6, further comprising:

determining a start offset according to the start key and the start block; and

20

determining an end offset according to the end key and the end block.

8. The method according to claim 7, wherein determining the start offset according to the start key and the start block further comprising finding a location of an index that is closest to the start key in the start block.

5 9. The method according to claim 7 or 8, wherein determining the end offset according to the end key and the end block further comprising finding a location of an index that is closest to the end key in the end block.

10 10. The method according to any one of claims 7-9, wherein determining the split key from the middle block further comprising finding the index in the middle block whose location in the middle block is based on an average of the start offset and the end offset.

15 11. The method according to any one of claims 1-10, wherein determining the middle block of the rooted tree structure based on the start block and the end block further comprising determining the middle block as the block located at a center between the start block and the end block.

12. The method according to any one of claims 1-11, further comprising splitting the partition based on the determined split key.

20

13. A distributed data storage system, comprising:
a plurality of partitions configured to be organized in a form of a rooted tree structure;
a processor configured to:

obtain a first key and a second key based on a request to split a partition;

determine a start block of the rooted tree structure based on a search of the rooted tree structure using the obtained first key;

5 determine an end block of the rooted tree structure based on a search of the rooted tree structure using the obtained second key;

determine a middle block of the rooted tree structure based on the start block and the end block; and

determine a split key from the middle block.

10 14. The distributed data storage system according to claim 13, wherein the rooted tree structure comprises a root-level index block, a set of intermediate-level index blocks, and a set of data-level blocks having row-key pairings that corresponds to indexes provided in the set of intermediate-level index blocks.

15 15. The distributed data storage system according to claim 14, wherein the set of intermediate-level index blocks comprises N-number of index blocks and index blocks 1 through N-1 have the same number of indexes, wherein:

N is an integer greater than 0.

20 16. The distributed data storage system according to claim 14 or 15, wherein the set of data-level blocks comprises X-number of data blocks and data blocks 1 through X-1 have the same number of row-key pairings, wherein

X is an integer greater than 0.

17. The distributed data storage system according to any one of claims 14-16, wherein the processor is further configured to:

obtain an estimation threshold;

5 determine the middle block in response to the estimation threshold being smaller than a number of blocks between the start block and the end block; and

searching a next lower level of the rooted tree structure in response to the estimation threshold being larger than the number of blocks between the start block and the end block.

10 18. The distributed data storage system according to claim 17, wherein the next level of the rooted tree structures comprises another set of intermediate-level index blocks that are lower in the rooted tree structure than the set of intermediate-level index blocks, wherein the set of data-level blocks has row-key pairings that corresponds to indexes provided in the other set of intermediate-level index blocks.

15

19. The distributed data storage system according to claim 17 or 18, wherein the estimation threshold is equal to 0.

20 20. The distributed data storage system according to any one of claims 13-19, wherein the processor is further configured to:

determine a start offset according to the start key and the start block; and

determine an end offset according to the end key and the end block.

21. The distributed data storage system according to claim 20, wherein the processor is further configured to determine the start offset by finding a location of an index that is closest to the start key in the start block.

5 22. The distributed data storage system according to claim 20 or 21, wherein the processor is further configured to determine the end offset by finding a location of an index that is closest to the end key in the end block.

10 23. The distributed data storage system according to any one of claims 20-22, wherein the processor is further configured to determine the split key from the middle block by finding the index in the middle block whose location in the middle block based on an integer closest to an average of the start offset and the end offset.

15 24. The distributed data storage system according to any one of claims 13-23, wherein the processor is further configured to determine the middle block as the block located at a center between the start block and the end block.

20 25. The distributed data storage system according to any one of claims 13-24, wherein the processor is configured to split the partition based on the determined split key.

26. A non-transitory computer readable medium that stores a set of instructions that are executable by one or more processors of an apparatus to perform a method comprising:

obtaining a first key and a second key based on a request to split a partition represented by a rooted tree structure;

5 determining a start block of the rooted tree structure based on a search of the rooted tree structure using the obtained first key;

determining an end block of the rooted tree structure based on a search of the rooted tree structure using the obtained second key;

10 determining a middle block of the rooted tree structure based on the start block and the end block;

determining a split key from the middle block.

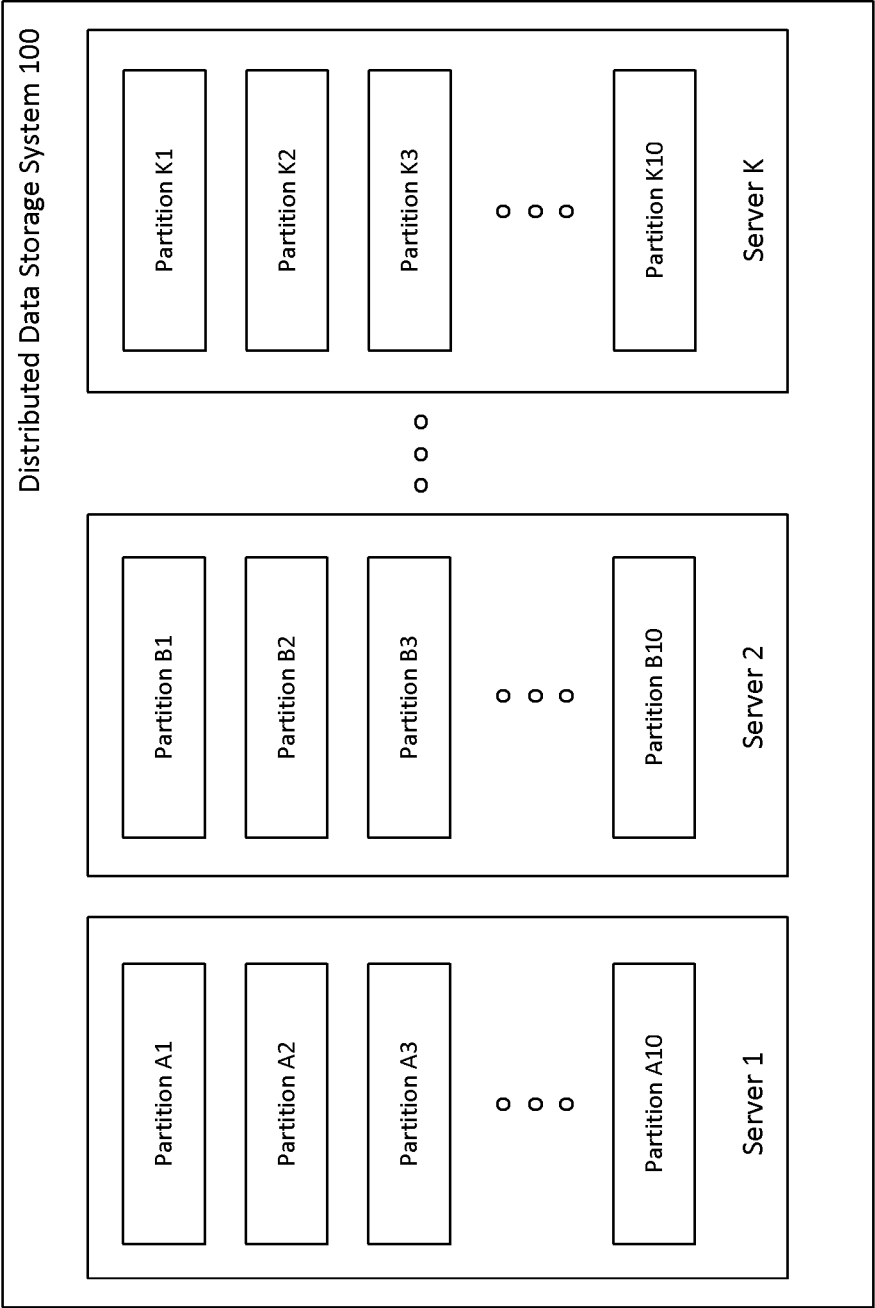


FIG. 1A

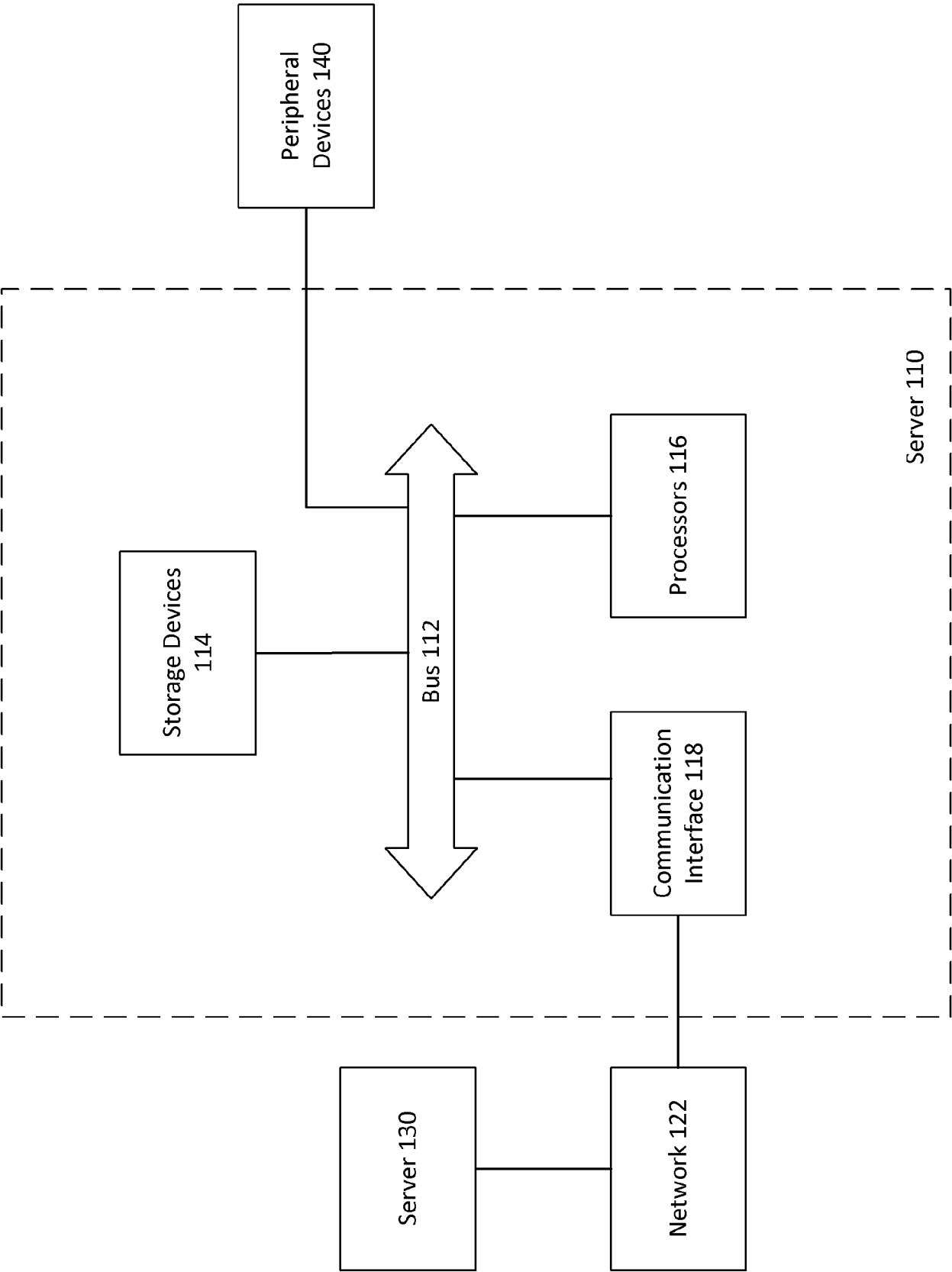


FIG. 1B

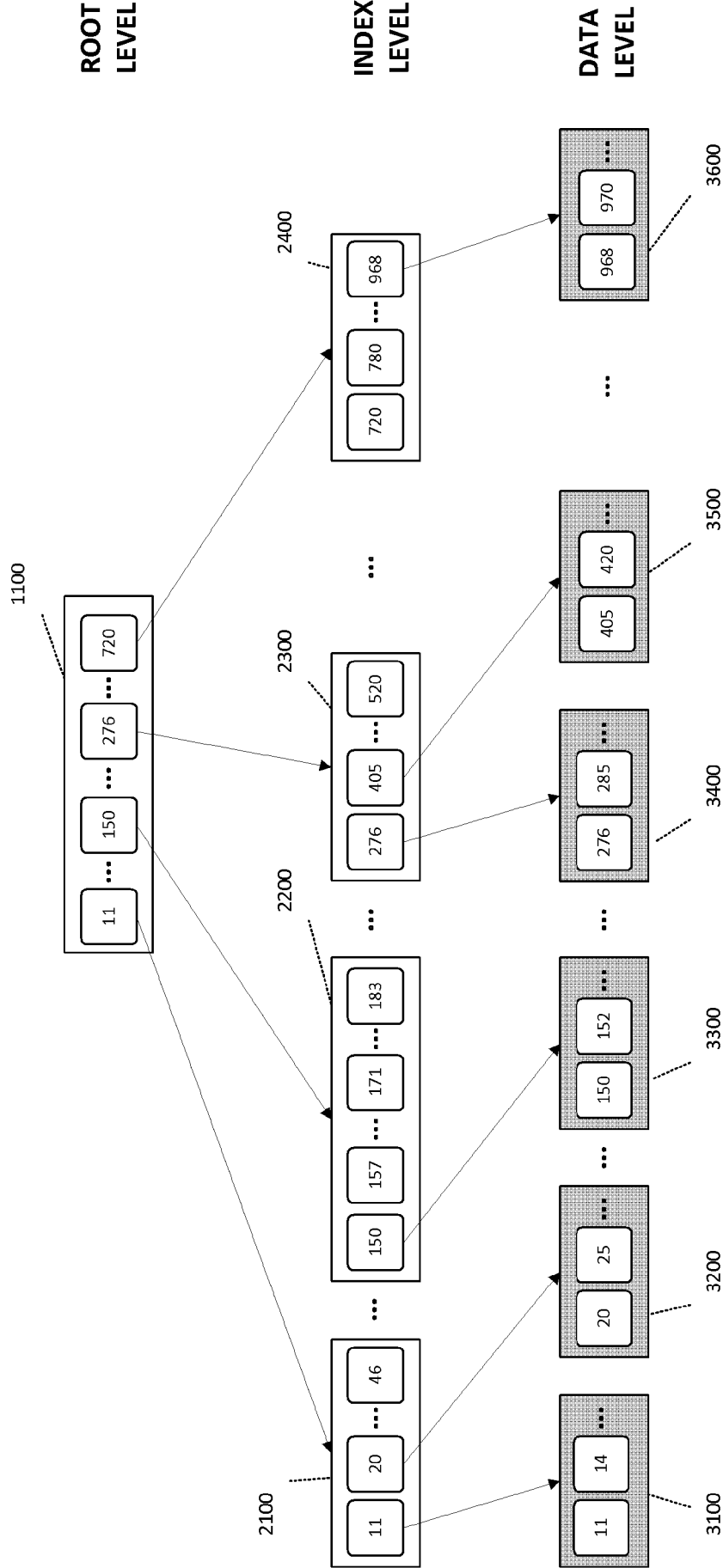


FIG. 2

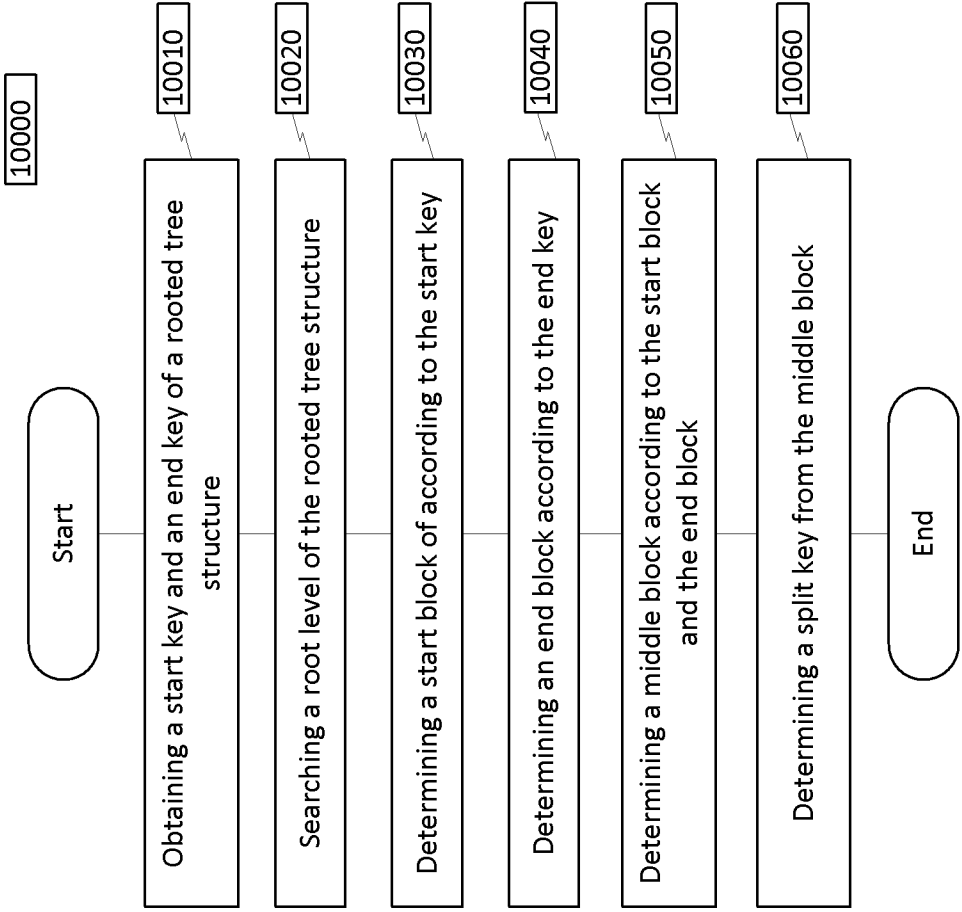


FIG. 3

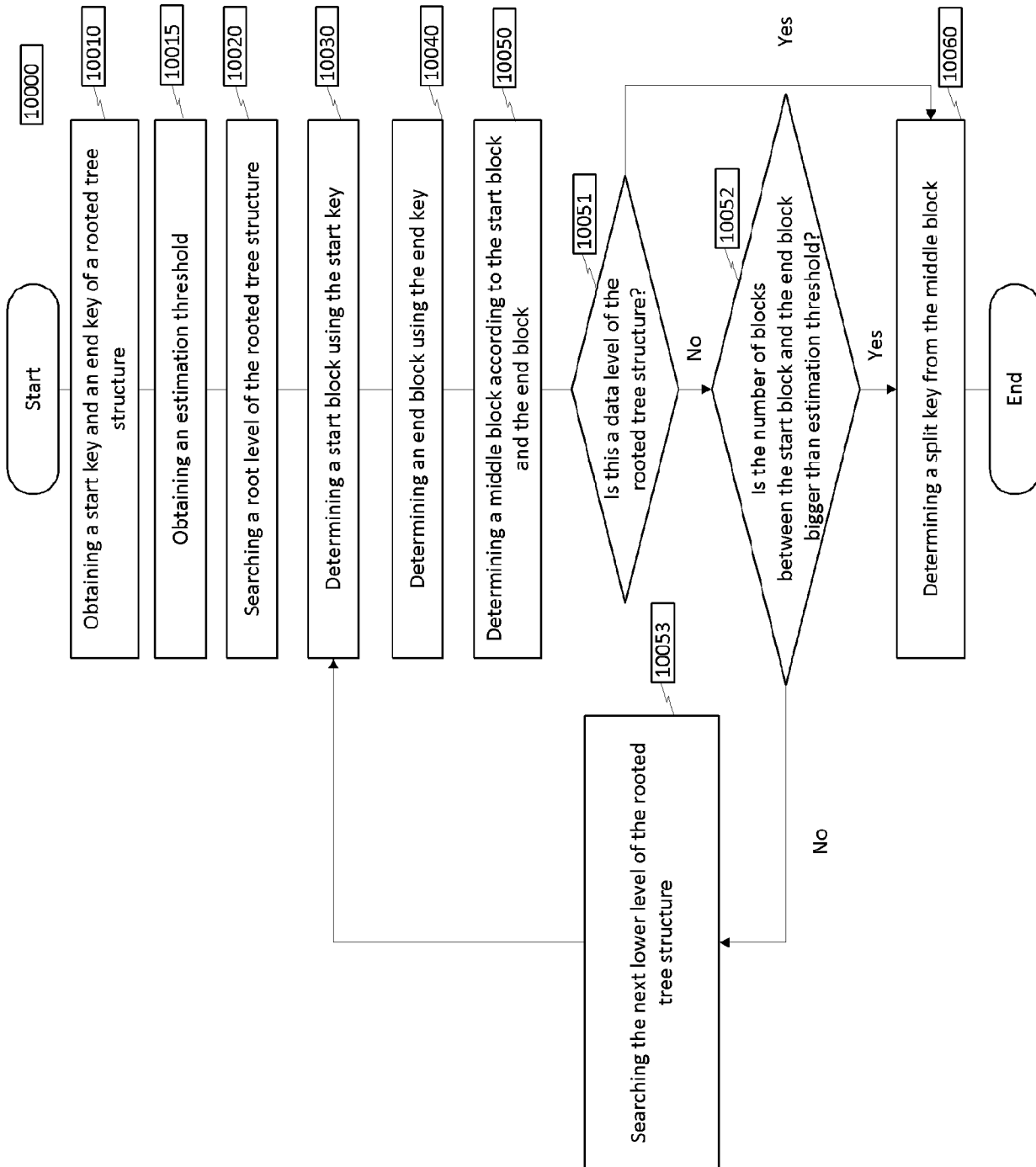


FIG. 4

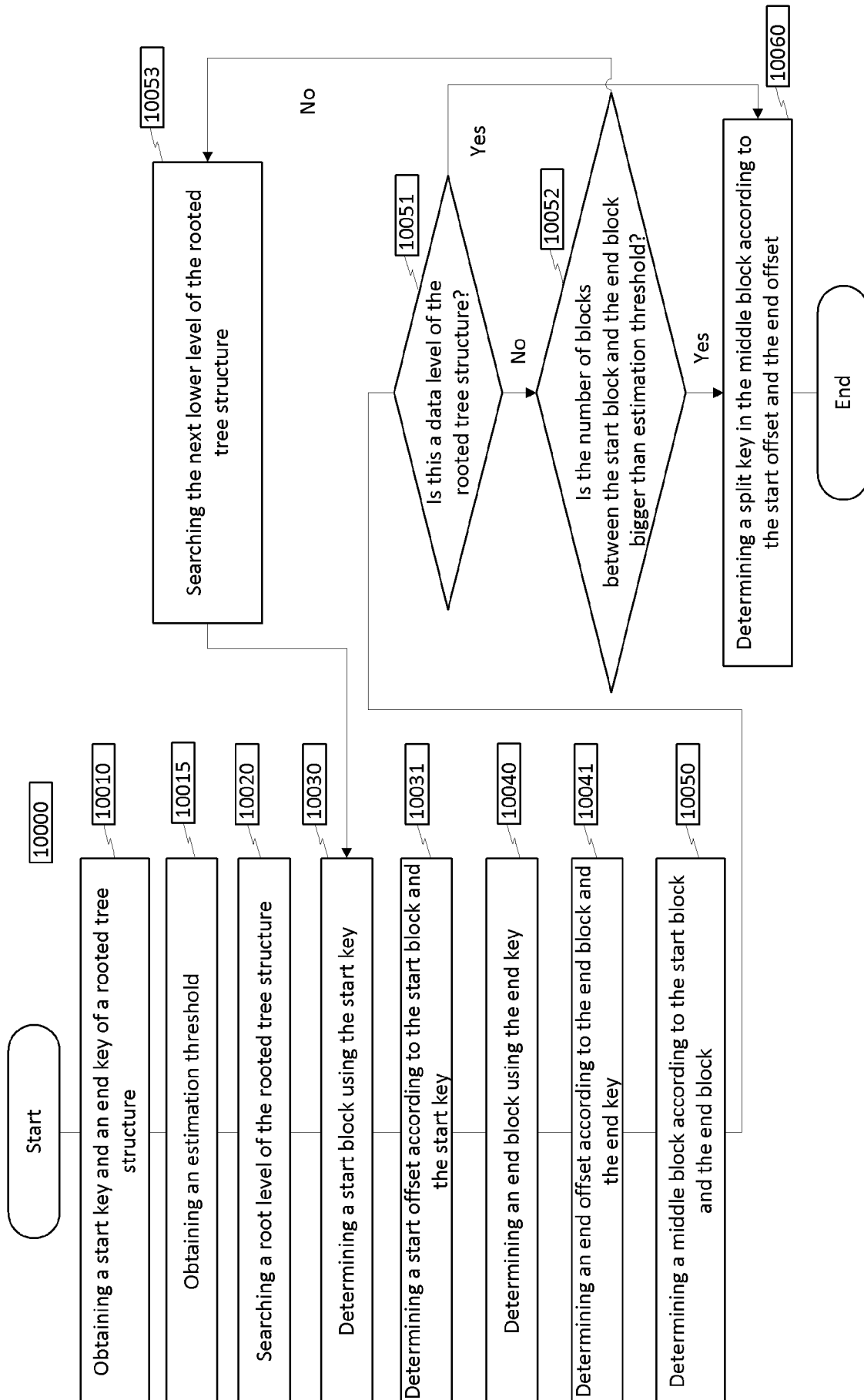


FIG. 5

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2019/081492

A. CLASSIFICATION OF SUBJECT MATTER

G06F 16/61(2019.01)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

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNPAT,WPI,EPODOC,CNKI:index, tree, split, search, query, retrieve, keyword, distributed, database, key, block, partition

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	CN 104750708 A (HUAWEI TECHNOLOGIES CO., LTD.) 01 July 2015 (2015-07-01) abstract, description, paragraphs [0063]-[0180]	1-26
A	CN 108932347 A (NORTHEASTERN UNIVERSITY) 04 December 2018 (2018-12-04) the whole document	1-26
A	WO 2014141802 A1 (SONY CORPORATION) 18 September 2014 (2014-09-18) the whole document	1-26
A	US 2012117065 A1 (MICROSOFT CORPORATION) 10 May 2012 (2012-05-10) the whole document	1-26



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

25 December 2019

Date of mailing of the international search report

14 January 2020

Name and mailing address of the ISA/CN

National Intellectual Property Administration, PRC
6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing
100088
China

Authorized officer

LIU, Changyong

Facsimile No. (86-10)62019451

Telephone No. 86-(10)-53961368

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2019/081492

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
CN	104750708	A	01 July 2015	WO 2015096582 A1	02 July 2015
CN	108932347	A	04 December 2018	None	
WO	2014141802	A1	18 September 2014	None	
US	2012117065	A1	10 May 2012	None	