

METHOD AND SYSTEM FOR NUMBER COMPARISON DURING STREAM PROCESSING

Field of Invention

The invention relates to a method and system for number comparison during stream
5 processing, specifically, to a method and system for fast comparison of large integral or
decimal numbers directly from their ASCII character based representation without
converting them into number representation as maintained in computer.

Background

Number comparison is a very common activity in programming tasks. Typically,
10 computers keep integers in 2's complement form and decimal numbers in IEEE standard
754. In existing comparison method, each number need to be converted into integer or
decimal numbers as maintained in computer before comparison. For example, integer
conversion is done by traversing through the digits from the beginning and multiplying the
converted part by 10 and then adding the product with the next digit which then becomes
15 converted part for next iteration.

As explained above, it takes time to convert numbers, e.g. ASCII codes, into a
computer based number format in the existing comparison method. However, in certain
functional areas such as stream processing where available time to process individual
records is very less, typically a few milliseconds, e.g. 1–5 milliseconds. For example, during
20 a telecom call detail record processing, a common requirement is to know which operators
do the calling and called number belong to. Since mobile number portability is in use in
many countries, it is not possible to identify operators by checking whether the mobile
number belongs to a particular block. Typically, a large number of blocks are to be checked

for this purpose. Since the ranges are too numerous, e.g. 500,000 blocks for a country like Netherlands, it is not convenient to load these entirely in computer memory. A frequently used approach for this kind of search is to decide a coarse block from memory based search and then do a record by record search within the bigger block in hard disk.

5 Traditionally, the input number is checked with start of the range and end of the range. If it falls in between, the operator ID is picked up from this record. Since master files with ranges are generally made available by public agencies for diverse use, they encode the start and end of range using human readable ASCII character codes. Commonly, the input number, start of the range and end of the range are first converted from ASCII to numbers,
10 as maintained in computer, and then two simple number comparisons are done.

It is therefore desirable to provide a comparison method faster than the existing method, especially when the numbers are very big.

Summary of Invention

In order to provide a faster way for comparing numbers during stream processing,
15 embodiments of the invention provide a solution in which the numbers need not to be converted into integer or decimal numbers as maintained in computer before comparison such that it takes less time to obtain a comparison result during stream processing.

According to one aspect of the invention, a method for number comparison during stream processing is provided. The method comprises:

20 generating, by a state machine, a current index associated with the given number based on a current digit pair including a first current digit from the given number and a second current digit from the target number, and a current state information associated with the given number; and

looking up, by the state machine, the generated current index in a preset state transition table to identify a next state information, wherein the preset state transition table maintains a plurality of mappings, each of the mappings is between an index and a next state information;

5 if the current digit pair includes a last digit of the given number or a last digit of the target number, determining, by the state machine, a final comparison result based on the identified next state information; and

 if the current digit pair includes neither a last digit of the input number nor a last digit of the target number, taking on, by the state machine, the identified next state information
10 as the current state information associated with the input number for comparison of a next digit pair.

According to another aspect of the invention, a system for comparing a given number and a target number during stream processing is provided. The system comprises a processor and a memory communicably coupled thereto,

15 wherein the memory is configured to store data to be executed by the processor,

 wherein the processor is configured to

 generate a current index associated with the given number based on a current digit pair including a first current digit from the given number and a second current digit from the target number, and a current state information associated with the given number; and

20 look up the generated current index in a preset state transition table to identify a next state information, wherein the preset state transition table maintains a plurality of mappings, each of the mappings is between an index and a next state information;

if the current digit pair includes a last digit of the given number or a last digit of the target number, determine a final comparison result based on the identified next state information; and

if the current digit pair includes neither a last digit of the given number nor a last digit of the target number, take on the identified next state information as the current state information associated with the given number for comparison of a next digit pair.

Embodiments of the invention use successive comparison of digit pairs from a given number and a target number to decide whether the given number received by the state machine is bigger, smaller or equal to the target number. The essence of this mechanism is to assign appropriate states to the status of comparisons up to a specific instance during traversal steps and then move from one state to another state until one of the target states is reached or widths of both numbers are traversed. The machine mostly moves forward. However, sometimes, it moves back. For example, if a given number 01234 is compared with a target number 1234, after the state machine traverses over the first current digit pair 01, instead of moving to the next digit pair 12, the pointer of reader of the given number is moved back such that the next digit pair to be compared becomes 11. This effectively allows to skip leading zero in the given number. As is obvious, this rule will only be used if the current state is initial state since the purpose is to skip initial set of zeros in the given number and/or the target number. This means that next state information and movements of reader of the two numbers are dependent on current state information. Now consider 01234 and 01235. In this case, it is not necessary to move back, the initial state is maintained after traversing through 00. Essentially, the invention shows that it is possible to device a state machine for number comparison purpose and provides state details and state transition table entries. The method provided by this invention is faster than the existing number comparison methods, especially when the number is very big.

Brief Description of the Drawings

The invention will be described in detail with reference to the accompanying drawings, in which:

Figure 1 is a flow chart illustrating the method for comparing numbers during a stream processing according to one embodiment of the invention.

Figure 2 shows a table illustrating the comparison process according to a first example of the invention;

Figure 3 shows a table illustrating the comparison process according to a second example of the invention;

Figure 4 shows a table illustrating the comparison process according to a third example of the invention;

Figure 5 shows the overall state transitions according to embodiments of the invention;

Figure 6 shows a table illustrating the structure of an index generated to lookup to decide a next state according to one embodiment of the invention;

Figure 7 shows a table illustrating the encoded values of the states according to one embodiment of the invention;

Figure 8 shows a table illustrating the encoded values of the target and input number according to one embodiment of the invention;

Figure 9 shows a table illustrating a structure of state information as obtained from lookup in state transition table according to one embodiment of the invention;

Figure 10 shows an example of the preset state transition table;

Figure 11 (a) and (b) show a state transition process when handling any number with leading zeroes during comparison according to one embodiment of the invention;

Figure 12 (a) and (b) show a state transition when handling thousand or other
5 separators during comparison according to one embodiment of the invention;

Figure 13 (a) and (b) show a state transition when handling decimal place indicator during comparison according to one embodiment of the invention;

Figure 14 (a) shows a process when handling comparison of the input number and the target number having “+” or “-” symbol/sign at the beginning of the numbers according to
10 one embodiment of the invention;

Figure 14(b) shows a preset final result table according to one embodiment of the invention;

Figure 15 shows a table illustrating performance analysis of the proposed method for number comparison compared with that of a traditional method;

15 Figure 16 is a bar chart showing the performance comparison of the proposed method for number comparison with a traditional method.

Detailed Description of Embodiments of the Invention

20 In the following description, numerous specific details are set forth in order to provide a thorough understanding of various illustrative embodiments of the invention. It will be understood, however, to one skilled in the art, that embodiments of the invention may be practiced without some or all of these specific details. It is understood that the terminology used herein is for the purpose of describing particular embodiments only, and is not

intended to limit the scope of the invention. In the drawings, like reference numerals refer to same or similar functionalities or features throughout the several views.

Embodiments of the invention provide an efficient method for comparing numbers during stream processing. In this method, the two numbers to be compared need not to be converted into integer or decimal number, each pair of digits in the two numbers are directly compared and a comparison result is obtained according to a pre-set state transition table. Referring to Figure 1, according to one embodiment of the invention, the method for comparing a given number, e.g. an input number, and a target number during stream processing comprises the following steps:

In Block 101, a state machine generates a current index associated with the given number based on a current digit pair including a first current digit from the given number and a second current digit from the target number, and a current state information associated with the given number.

In one example of the embodiment, the current index associated with the given number includes 12 bits, wherein 4 bits are used for defining the current state information associated with the given number, 4 bits are used for defining the first current digit from the given number and the remaining 4 bits are used for defining the second current digit from the target number.

In Block 102, the state machine looks up the generated current index in a preset state transition table to identify a next state information from the preset state transition table. The preset state transition table maintains a plurality of mappings each between an index and a next state information.

In Block 103, the state machine determines whether the current digit pair includes a last digit of the given number or a last digit of the target number, if yes, the flow chart proceeds to block 104, if no, the flow chart proceeds to block 105.

In Block 104, the state machine determines a final comparison result based on the identified next state information.

The final comparison result has many uses in real world for ultra-low latency requirements in telecommunication systems. In one example, in 4G, calls can be completed by base station itself if both caller and callee are served by the same base station. This needs a real-time look-up and may involve number comparison. In another example, a common requirement in many telecom systems is to identify, in real-time, which

telephone number is served by which operator. Since mobile number portability is in use in many countries now, it is not possible to apply a simple number series based comparison. Rather, a detailed block based table lookup where table size is very big, typically hundreds of thousands of entries, has to be employed. The mentioned search needs fast comparison of big numbers, typically 10 digits or more in length.

In Block 105, the state machine takes on the identified next state information as the current state information associated with the given number and takes on the next digit pair as the current digit pair, and the flow chart proceeds back to block 101 to conduct the comparison of the next digit pair.

As described above, the proposed invention checks pair of digits, from target and given number, as it goes along. It keeps track of the results of this traversal using a state machine. At any time if the state machine concludes on the comparison result, the normal computation is stopped. At this stage, a secondary traversal may be done on rest of the digit strings of the numbers, both the given number and the target number, to ensure that there is no formatting error. If no formatting error is detected, then earlier conclusion by the state machine is output. Compared with the traditional method of comparison, the method disclosed in embodiments of the invention is more efficient, because the neither the target number nor the given number need to be converted into integer or decimal number. A simple comparison example is explained below.

In this example, the target number is 1234, and the given number is an input number 456. In order to determine whether the input number is greater, equal or smaller than the target number, the following process will be conducted:

Preparatory step: Pad input number with a space, i.e. "456(space)". Note that the target number is in a file with fixed width for it. If width is not filled with digits then space is expected.

First, digit 1 from the target number and digit 4 from the given number are checked. Since 4 is greater than 1, internal state is set to “possibly greater” meaning input number is possibly greater than the target number.

Next, digit 2 from the target number and digit 5 from the given number are checked.

- 5 Since 5 is greater than 2, the earlier state i.e. “possibly greater” is maintained.

Next, digit 3 from the target number and digit 6 from the given number are checked.

Since 6 is greater than 3, the earlier state i.e. “possibly greater” is maintained.

Next, digit 4 from the target number and space from the given number are checked.

- 10 As per the machine state transition rules, the state is output as “small”, based on the fact that earlier state was “possibly greater” and now space is received in input number and a regular digit between 0 to 9 is received in target number, meaning that input number is smaller than target number.

- As can be seen, this method traverses over the digits just once. Precisely speaking, only for handling leading zeros, the machine stays over same character; otherwise it traverses over the digits just once. It makes the correct decision by following the states and state transitions in a proper manner. The actual algorithm is more complex than explained above since it handles “+” or “-” signs at the beginning, decimal place indicators “.”, any number of zeros in the beginning and separator characters like comma. State transition rules are maintained in an array. The array index is made up of state (encoded as number), digit of target number and digit of input number. This arrangement alleviates the need for search and locates the correct entry in array directly. To reduce the array size, state and the digits are packed into individual nibbles. So, 12 bits containing 4 bits for state, 4 bits for digit of target number and 4 bits for digit of input number are needed to represent an array index. Consequently, an array of 4096 length is needed. Since array will hold output states and
- 15
- 20

only 7 states are there, array data type is byte. As will be apparent, this byte contains 4 bits of state information, a single bit for forward movement indicator and another single bit for backward movement indicator and two bits are kept for future use. In effect, 4096 bytes or 4KB for transition rule storage are used. For current day commodity hardware, this memory requirement is not big and no special attention is required from hardware side to use proposed method. One example of the state transition rules is shown in Figure 10 which is a preset state transition table maintaining a plurality of mappings each between an index and a result byte, i.e. a next state value or a final state value.

This invention provides a state machine based novel method for fast in-place comparison of numbers, both integer and decimal. The context of this invention is there is a need to check whether a given number falls between a series of start and end range values provided to us in a text file. Both the numbers - start and end of ranges - have their own positions and width in the file. If the number cannot fill up the width specified, then file comes with spaces in those locations. All three numbers, start and end of range and input number are coded in ASCII i.e. each digit, decimal and signs are coded using ASCII coding. Input number is part of streaming data and series of ranges in file for each input number is to be checked in real-time. All three numbers are typically big, approximately 10 digit long integers. Obviously, the major work involved here is a series of comparisons in trying to find a match. If a match is found, then a portion of the record, between given start and end character positions, is extracted as outcome and further checking for the same input is stopped.

There could be many practical scenarios where large numbers of number comparisons are required to be made in lowest possible time. One such scenario is deciding operator identifiers for calling and called numbers during telecom call detail record processing. Operator identifiers are typically used to analyze operator-wise business

volume as well as capacity decisions for points of interconnection. They can also be used for operator-based differential charging. For a country like India with approximately 885 million mobile connections, 12 mobile operators and prevalent mobile number portability, the number of ranges to search to identify operators is enormous. Though part of the
5 ranges can be loaded in RAM and a more efficient method can be used for searching, there is still a need for record by record search because of numerous ranges.

The essence of stream processing is low latency. Traditional comparison first converts ASCII based numbers into format as maintained in computer i.e. 2's complement for integer and IEEE Standard 754 for floating point number. Then comparison is made
10 between numbers. Since comparison is implemented using special purpose comparator circuits, it is fast. However, converting the number in ASCII representation to computer number format takes time. Intuitively, the conversion can be avoided if a decision can be made by comparing ASCII digits. The comparison is trivial if the input and target number has same width and doesn't have unequal number of zeroes in the beginning. However, a
15 method which takes care of both integer and decimal numbers, handles any number of leading zeroes, works fine with decimal place indicators and positive/negative signed number ('+' and '-') as well as unsigned numbers and handles format error gracefully is non-trivial.

The proposed invention uses the following approach to compare numbers. When
20 humans see numbers, humans seem to decide based on number length, digits, use of decimal place indicator etc. Humans do not always seem to mentally convert a big number as millions or billions to compare with a smaller number which is of 4 digits. Even if both the numbers are big, humans try to find dissimilarities to discern which one is bigger. Inherent in this way of analysis is that while humans see numbers, they make incremental inferences
25 continuously and stop when they feel certain.

Let's take an example to compare an input number and a target number:

Target number: 1234

Input number: 1439

Each pair of digits (1, 1), (2, 4), (3, 3) and (4, 9) are sequentially analyzed to keep
5 track of the last decision made. The comparison result byte of each step and the final
comparison result are shown in Figure 2. Referring to Figure 2, at the beginning, the current
state associated with the input number is initial state; then, the first digit pair (1,1) is
compared, the identified next state is "possibly equal" state, accordingly, the current state
associated with the input number is changed to "possibly equal" state; next, the second digit
10 pair (2, 4) is compared, since the current state is "possibly equal" state, it is inferred that the
next state is "possibly greater" state. But it is not the final comparison result since the input
number may be lengthwise smaller than the target number; then the current state is
changed to the "possibly greater" state; next, the third digit pair (3,3) is compared, the next
state is identified as the same as the current state, i.e. "possibly greater" state; next, the
15 fourth digit pair (4, 9) is compared, since the current state is "possibly greater" state and
same number of digits have been encountered in both numbers, it is inferred to remain in
the same state; finally, since the digit pair (4, 9) is the last digit pair in the two numbers, the
final comparison result is generated based on the "possibly greater" state. The final
comparison result is "greater" state, meaning that the input number is greater than the
20 target number.

Let's take another example. In this example, the target number includes one leading
zero.

Target number: 01234

Input number: 1235

Each pair of digits is analyzed as shown in Figure 3. Referring to Figure 3, before comparing any digit pair, the current state associated with the input number is initial state; then the process proceeds to compare the first digit pair (0,1), since the current state is the initial state, and a leading zero is identified in the target number, the state machine skips the leading zero digit and stay in the initial state. It means that the next step is to compare the digit pair (1,1) instead of (1,2). The initial state value may be 0.

To see how decimal numbers are handled, let's go through steps involved in a simple case with two decimal numbers.

Target number: 123.45

Input number: 12.345

Each pair of digits is analyzed as shown in Figure 4. Referring to Figure 4, before comparing any digit pair, the current state is the initial state, the current state value may be 0; then, the first digit pair (1,1) is compared, the next state associated with the input number is identified as "possibly equal" state; next, the second digit pair (2, 2) is compared, the next state associated with the input number is remained in "possibly equal" state; next, the third digit pair (3, .(dot)) is compared, the next state associated with the input number is identified as "smaller" state since a decimal place indicator is encountered in the input number and a normal digit is encountered in the target number. Then, the comparison is stopped since a final state has been identified. However, the process may be continued to traverse rest of the digit pairs in the numbers to ensure that these indeed are valid numbers, meaning that these numbers do not contain any character other than 0-9, dot etc.

There are more details involved in the proposed method e.g. handling sign characters, handling thousand separators, verifying presence of format issue in an efficient way etc. However, essentially, the proposed method deviates from converting ASCII representations to computer based number formats by having a state based method and
5 traverses over the digit strings only once. Both of these considerations improve performance drastically. Even though computer format based number comparison is implemented in hardware as a special circuit, proposed arrangement reduces average comparison time by half. Figure 5 depicts state transitions at an overall level.

“Greater”, “smaller” or “equal” are final results of comparing input number with target
10 number i.e. $\text{input} > \text{target}$, $\text{input} < \text{target}$ and $\text{input} = \text{target}$

All the states except “possibly equal with decimal place indicator” are self-explanatory. State 7 is treated separately than state 3 since transitions to other states are different from these two situations. For clarity, let us consider following two examples:

Example 1

15 Target number: 1234

Input number: 1235

While the last digit pair (4, 5) is compared, the current state value is 3, i.e. the current state is “possibly equal” state, then after comparing the last digit pair (4, 5), the next state is “possibly greater”, i.e. the state value 1.

20 Example 2

Target number: 123.4

Input number: 123.5

While the last digit pair (4, 5) is compared, the current state value is 7 instead of 3, i.e. the current state is "possible equal with decimal place indicator", then the next state is the "greater" state, i.e. the state value 4. To differentiate between these two situations, we move to state 7, instead of state 3, when the digit pair (. (dot), . (dot)) in the example 2 is reached.

The proposed method in this invention works as follows:

As part of setup activities, a state transition table is generated. The table index consists of 12 bits of information arranged as mentioned below according to one embodiment of the invention. Table 1 in Figure 6 shows Structure of index used to lookup to decide next state. States are encoded according to the Table 2 shown in Figure 7. Target and input digits are encoded according to Table 3 shown in Figure 8.

Lookup in the state transition table returns next state information in a single byte. Next state information also includes move indicator for target and input numbers as shown in Figure 8.

State transition table is generated based on the understanding of how it is inferred that the input number is greater, smaller or equal to the target number. This is the key point of the method. Majority of the transitions are straight-forward except for few special situations which are described in more detail in sections that follow.

The method traverses through the digits of target number and input number by generating an index and looking up the index in a preset state transition table. The index comprises current state (4 bits), digit from target number (4 bits) and digit from input number

(4 bits). As mentioned above, lookup in state transition table returns a byte. On receiving this result byte, the following actions are taken:

- a. Check 8th bit and if move indicator for target number is set, target number pointer is moved back by one position.
- 5 b. Check 7th bit and if move indicator for input number is set, input number pointer is moved back by one position
- c. Extract 4 LSB bits and use it as current state

If lookup in the state transition table returns a default value (for our experimentation, 10 0x0B is considered) indicating that lookup was not successful, digits in input or target or both are inappropriate and a format error is produced. Format validation can optionally be done during encoding of digits of target and input characters; however for performance reasons, format validation is performed during lookup.

15 At the end of each iteration step, the method checks whether any of the end states i.e. 4, 5 or 6, is reached. If yes, state based checking ceases and individual checks on remaining digits (until space character is encountered or number boundary as specified by width is reached) of both numbers is performed to ensure that number formats are correct. If number formats are correct, comparison result is produced; else, a format error is 20 produced.

One specific format validation namely presence of multiple dots can't be directly handled during encoding of digits as well as during lookup in state transition table. This is done by having two counters, each with initial value as 2, for target and input numbers and 25 reducing by 1 whenever a dot is encountered. After every such reduction, a check is performed: if the counter is zero and, if so, an error is output. This check is performed both

during digit encoding as well as extra digit validation at the end. To avoid an 'if' based check for performance reasons, digit encoding array is made two-dimensional; second dimension contains zero for all digits except for dot.

One example of the preset state transition table entries is shown in Figure 10. In this example, each index is generated by: state + target digit + input digit; and each result byte includes a byte containing state value and movement indicators. For example, the index 0X010 defining current state value 0 (the corresponding binary value 0000) + target digit 1 (the corresponding binary value 0001) + input digit 0 (the corresponding binary value 0000), the corresponding result byte is 0X80 defining the next state value 0 (the corresponding binary value 0000) + movement indicator of the target number is on (the corresponding binary value 1000); the index 0X19F defining current state value 1 (the corresponding binary value 0001) + target digit 9 (the corresponding binary value 1001) + input digit F (meaning there is a space character), the corresponding result byte is 0X05 defining the next state value 5 (the corresponding binary value 0101) + no movement indicator 0 (the corresponding binary value 0000). Based on the preset lookup table and the encoding rules, e.g. the encoding rules shown in Figure 5 and Figure 6, the state machine can generate an index based on the current state value and the values of each digit pair of the two numbers, i.e. the given number and the target number, and lookup the generated index to identify next state information including the next state value and movement indicators of the two numbers. Then the next state value will be considered as current state value for comparison of a next digit pair. Once all of the digit pairs of the two numbers have been traversed or a final state value is identified, the final comparison result will be determined and the comparison will be stopped.

Handling of any number of leading zeroes

In some embodiments of the invention, both the input and target number can have any number of leading zeroes. The special treatment is needed only when the state machine is in initial state and zero is encountered in either input or target number. The basic idea is that initial state is maintained only while “skipping” over the zeroes. In normal traversal, counter is increased in both numbers after each index formation and lookup. When moving forward in any number is not desired, the normal flow is not disturbed but the pointer is moved back by one position based on move indicators received on result byte (bit positions 7 and 8) from state transition table lookup. Figure 11 (a) and (b) depict the corresponding state transitions according to one example of the invention. Referring to Figure 11(a) and (b), if the current state is “0”, i.e. initial state, and any one of the first current digit in the current digit pair from the target number and the second current digit in the current digit pair from the input digit is 0, the other one is any one of “0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, F”, then the next state remains “0”, i.e. initial state.

Initial state is indicated by state code 0 at the middle of the figure. The incoming tokens are a sequence of digit from target number and input number, left to right. During the course of skipping leading zeroes, if end of digit strings is reached in both numbers, end state “6-Equal” is taken on.

Handling of thousand or similar other separators

Handling of thousand separators just involves skipping those separators. Figure 12(a) and (b) depict relevant state transitions on encountering separator either in input number or in target number or in both numbers according to one example of the invention. As is intuitive, separator can only appear while in state “1 – possibly greater”, “2 – possibly small” or “3 – possibly equal”. If separator is encountered while the current state is other states e.g. “0 – Initial” or “7 – possibly equal with decimal”, format error is generated since

no corresponding state transition is configured. Referring to Figure 12 (a) and (b), if a comma or another thousand separator appears in either of the input number or the target number, or both, the next state will be same as the current state, and the reader of the number in which the comma or thousand separator appears will be moved back one position.

Handling decimal place indicator

Decimal place indicators/decimal points may appear in any of the input and target numbers or both. In a simple situation, in the current state is “1 – Possibly greater” state and a decimal digit appears in target number and a normal digit (between 0 and 9) in input number. Then, it is easy to understand that input number is greater than target number and hence the next state moves to state “4 – Greater”. Similar such situations can easily be thought of to cover all possible tokens and corresponding state transitions. Figure 13 (a) and (b) depict graphically state transitions involving token with decimal digit(s) according to one example of the invention. Referring to Figure 13 (a) and (b), the current state may be “1 – possibly greater”, “2 – possibly small” or “3 – possibly equal”. if a decimal place indicator appears in the input number, the current digit from the target number is any one of 1, 2, 3, 4, 5, 6, 7, 8, 9, the next state moves to “5-Smaller” regardless of the current state; if a decimal place indicator appears in the target number, the current digit from the input number is any one of 1, 2, 3, 4, 5, 6, 7, 8, 9, the next state moves to “4-Greater” regardless of the current state. If a space character appears in one of the input number and the target number, and a decimal place indicator appears in the other one, the next state moves to a corresponding final/intermediate state of the current state, i.e. if the current state is “1– possibly greater”, the next state is “4-Greater”; if current state is “2 – possibly small”, the next state is “5 –Smaller”, if current state is “3 – possibly equal”, the next state is “7-Possibly

equal with decimal point” and the reader of the number in which the space character appears is moved back by one position.

Handling of + or – symbols at the beginning of input or target numbers

Steps for sign handling are illustrated in Figure 14 (a).

5 In block 1401, the state machine determines signs of the given number and the target number and encodes the signs according to a predetermined rule. For example, a negative sign is encoded as “0” and a positive sign is encoded as “1”.

In block 1402, the state machine performs comparison of digit pairs from the given number and target number according to the method disclosed in Figure 1.

10 In block 1403, the state machine determines the final comparison result based on the next state information identified in block 1302 and a preset final result table which maintains a plurality of mappings each between a final comparison result and a pair of signs of the given number and the target number.

Figure 14(b) shows an example of the preset final result table. According to this
15 preset final result table, if both the sign of the given number and the sign of the target number are positive signs, the final comparison result is same as the next state information identified in block 1302; if both the sign of the given number and the sign of the target number are negative signs, the final comparison result is contrary to the next state information identified in block 1302; if the sign of the non-zero given number is negative, the
20 sign of the target number is positive, the final comparison result is “Smaller” ; if the sign of the given number is positive, the sign of the non-zero target number is negative, the final comparison result is “Greater”.

Experimental results

In one embodiment of the invention, the method for number comparison may be used to check whether a given telephone number is same as a target telephone number. In order to determine whether the given telephone number is same as the target telephone number, the method may further comprise the following:

if the next state information identified based on the current digit pair is not "equal" or "possibly equal", determining, by the state machine, that the given telephone number is not same as the target telephone number, and

if the final comparison result is that the given telephone number is same as the target telephone number, retrieving, by the state machine, a corresponding operator name from a predetermined position in the text file.

In one example of the embodiment, the method of comparison is used to check whether a given telephone number lies between two other numbers which are mentioned in ASCII format in a text file. The two numbers are in ranges of position 5 to 24 and 25 to 44 in each line of the text file. The comparison is made for ranges as mentioned in each line of the text file sequentially. If a match is found in a line, then operator identifier is extracted from a corresponding record from position 93 to 99 in the line. Also, in such case, rest of the ranges in the text file is not checked. If a match is not found, then all the ranges in the text file are checked. The text file has 49 lines in increasing order.

First and last lines are provided below to provide clarity.

02RN0102010000	0102010009	0102010000	010 V
3020071119112750PTT	PTT	91024B325B325RT-C A	RT-C A
001000001000ABCD-XYZT.B325	024D MI		

...

...

...

	02RN0102010940		0102010949		0102010944		010 V
5	3020030702190000XYZ1	PTT	91024B325B325RT-C	A			RT-C A
	001000001000ABCD-XYZT.B325		024D MI				

In the table shown in Figure 15, time taken by the proposed method is contrasted against “conversion to number” based method for the above mentioned search for 1 million times. The tests were performed on a commodity grade machine with single Intel Pentium(R) dual-core CPU 2.0 GHz memory having 2 GB RAM, running Microsoft Windows Vista and Java 1.6. “Conversion to number” based method is already implemented in Java as Integer.parseInt() and Double.parseDouble() and the same were used for comparison purpose. Different kinds of given numbers, varying in magnitude, digits after decimal point, leading zeroes, presence of sign character etc. as mentioned in the table shown in Figure 15 have been attempted to ascertain average improvement of performance.

As apparent from the above, on the average, the proposed method reduces search time by half. Figure 16 pictorially depicts performance comparison.

It is surprising that a method of comparing numbers can be implemented in a more efficient way than currently used. This has high importance where low latency computations are required or range check in mass scale is involved. The method disclosed in the embodiments of the invention can be applied to compare numbers with thousand separator or other separator characters, numbers with leading or trailing currency symbol, very

large/small numbers and very high precision numbers. The advantages of the method disclosed in embodiments of the invention include but not limited to the following:

a) The method according to the invention is very fast. On the average (based on tests over numbers with varying number of digits before or after decimal place), the invention is 107% faster than traditional method.

b) The method according to the invention can compare two numbers directly even though the numbers are having thousand separators (e.g. comma) or leading \$ etc. Traditional method will need removal of the corresponding characters before comparison which will take extra computation time.

c) Very large numbers – integers which can't be expressed in 4 bytes and floats which can't be expressed in 8 bytes following IEEE Standard 754, can't be compared using traditional method. Proposed invention has no limit on the size of number or number of places after decimal.

It is to be understood that the embodiments and features described above should be considered exemplary and not restrictive. Many other embodiments will be apparent to those skilled in the art from consideration of the specification and practice of the invention. The scope of the invention should, therefore, be determined with reference to the appended claims, along with the full scope of equivalents to which such claims are entitled. Furthermore, certain terminology has been used for the purposes of descriptive clarity, and not to limit the disclosed embodiments of the invention.

Claims

1. A method for comparing a given number and a target number during stream processing, the method comprising:

generating, by a state machine, a current index associated with the given number
5 based on a current digit pair including a first current digit from the given number and a second current digit from the target number, and a current state information associated with the given number; and

looking up, by the state machine, the generated current index in a preset state transition table to identify a next state information, wherein the preset state transition table
10 maintains a plurality of mappings, each of the mappings is between an index and a next state information;

if the current digit pair includes a last digit of the given number or a last digit of the target number, determining, by the state machine, a final comparison result based on the identified next state information; and

15 if the current digit pair includes neither a last digit of the given number nor a last digit of the target number, taking on, by the state machine, the identified next state information as the current state information associated with the given number for comparison of a next digit pair.

2. The method according to claim 1, wherein each index in the preset state transition table
20 includes 12 bits, wherein 4 of the 12 bits are used for defining the current state information associated with the given number, 4 of the 12 bits are used for defining the first current digit from the given number, 4 of the 12 bits are used for defining the second current digit from the target number;

wherein generating a current index associated with the given number includes generating a 12 bits index having a same format as the index in the preset state transition table.

3. The method according to claim 2, wherein the current state information in each index is selected from 8 predetermined state types including: one Initial state, four intermediate state types including: "possibly greater", "possibly smaller", "possibly equal", "possibly equal with decimal place indicator", and three final state types including: "greater", "smaller" and "equal".

4. The method according to claim 3, wherein either of the first current digit and the second current digit is any one of 13 predetermined digits including: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A which defines a decimal place indicator, E which defines a thousand separator, and F which defines a space character.

5. The method according to claim 3, wherein each next state information corresponding to an index in the preset state transition table includes 8 bits wherein 4 of the 8 bits are used for defining a next state value corresponding to the index, 1 of the 8 bits is used for indicating whether a reader of the first current digit from the given number should be moved back by one position, and 1 of the 8 bits is used for indicating whether a reader of the second current digit from the target number should be moved back by one position.

6. The method according to claim 5, further comprising:

generating the preset state transition table.

7. The method according to claim 6, wherein the generating the preset state transition table includes:

if the current state information defined in the index is Initial state, and any one of the first current digit from given number and the second current digit from the target number is 0, setting a corresponding next state information in the preset state transition table to include a state value defining the Initial state and an indicator defining the reader of the number in which 0 appears will be moved back by one position.

8. The method according to claim 6, wherein the generating the preset state transition table includes:

if the first current digit from the given number and/or the second current digit from the target number is a comma, or other thousand separator, setting a corresponding next state information to include a state value defining a state same as the current state information defined in the index and an indicator defining the reader of the number in which the comma or other thousand separator appears will be moved back by one position.

9. The method according to claim 6, wherein the generating the preset state transition table includes:

if the first current digit from the given number is a decimal place indicator, the second current digit from the target number is any one of 1, 2, 3, 4, 5, 6, 7, 8, 9, setting a corresponding next state information to include a state value defining the "smaller" state regardless of the current state information defined in the index;

if the second current digit from the target number is a decimal place indicator, the first current digit from the given number is any one of 1, 2, 3, 4, 5, 6, 7, 8, 9, setting a corresponding next state information to include a value defining the "greater" state regardless of the current state information defined in the index;

if one of the first current digit from the given number and the second current digit from the target number is a space character, and the other one of the first current digit and the second current digit is a decimal place indicator, setting a corresponding next state to include a state value defining a corresponding final/intermediate state of the current state information defined by the index, and an indicator defining the reader of the number in which the space character appears will be moved back by one position.

10. The method according to claim 9, wherein

if the current state information is "possibly greater", setting the corresponding next state to include a state value defining a corresponding final state "greater",

10 if the current state information is "possibly smaller", setting the corresponding next state information to include a state value defining a corresponding final state "smaller",

if the current state information is "possibly equal", setting the corresponding next state information to include a state value defining a state "possibly equal with decimal place indicator".

15 11. The method according to claim 1, further comprising:

if the given number and the target number includes negative or positive signs, determining the final comparison result based on a preset final result table which maintains a plurality of mappings each between a final comparison result and a pair of signs of the given number and the target number.

20 12. The method according to claim 1, wherein the given number is a given telephone number, the target number is a target telephone number stored in a text file, in order to

determine whether the given telephone number is same as the target telephone number,
the method further comprising:

if the next state information identified based on the current digit pair is not "equal" or
"possibly equal", determining, by the state machine, that the given telephone number is not
5 same as the target telephone number, and

if the final comparison result is that the given telephone number is same as the
target telephone number, retrieving, by the state machine, a corresponding operator name
from a predetermined position in the text file.

13. A system for comparing a given number and a target number during stream processing,
10 the system comprising: a processor and a memory communicably coupled thereto,

wherein the memory is configured to store at least a preset state transition table to
be executed by the processor,

wherein the processor is configured to

generate a current index associated with the given number based on a current digit
15 pair including a first current digit from the given number and a second current digit from the
target number, and a current state information associated with the given number; and

look up the generated current index in a preset state transition table to identify a next
state information, wherein the preset state transition table maintains a plurality of mappings,
each of the mappings is between an index and a next state information;

20 if the current digit pair includes a last digit of the given number or a last digit of the
target number, determine a final comparison result based on the identified next state
information; and

if the current digit pair includes neither a last digit of the given number nor a last digit of the target number, take on the identified next state information as the current state information associated with the given number for comparison of a next digit pair.

14. The system according to claim 13, wherein each index in the preset state transition table includes 12 bits, wherein 4 of the 12 bits are used for defining the current state information associated with the given number, 4 of the 12 bits are used for defining the first current digit from the given number, 4 of the 12 bits are used for defining the second current digit from the target number;

wherein the processor is further configured to generate a 12 bits index having a same format as the index in the preset state transition table.

15. The system according to claim 14, wherein the processor is further configured to generate the current index based on a current state information selected from 8 predetermined state types including: one Initial state, four intermediate state types including: "possibly greater", "possibly smaller", "possibly equal", "possibly equal with decimal place indicator", and three final state types including: "greater", "smaller" and "equal".

16. The system according to claim 15, wherein each next state information corresponding to an index in the preset state transition table includes 8 bits wherein 4 of the 8 bits are used for defining a next state value corresponding to the index, 1 of the 8 bits is used for indicating whether a reader of the first current digit from the given number should be moved back by one position, and 1 of the 8 bits is used for indicating whether a reader of the second current digit from the target number should be moved back by one position.

17. The system according to claim 16, wherein the processor is further configured to:

generate the preset state transition table.

18. The system according to claim 17, wherein the processor is further configured to if the current state information defined in the index is Initial state, and any one of the first current digit from given number and the second current digit from the target number is 0, set a corresponding next state information in the preset state transition table to include a state value defining the Initial state and an indicator defining the reader of the number in which 0 appears will be moved back by one position .

19. The system according to claim 17, wherein the processor is further configured to if the first current digit from the given number and/or the second current digit from the target number is a comma, or other thousand separator, set a corresponding next state information to include a state value defining a state same as the current state information defined in the index and an indicator defining the reader of the number in which the comma or other thousand separator appears will be moved back by one position.

20. The system according to claim 17, wherein the processor is further configured to

if the first current digit from the given number is a decimal place indicator, the second current digit from the target number is any one of 1, 2, 3, 4, 5, 6, 7, 8, 9, set the corresponding next state information to include a state value defining the "smaller" state regardless of the current state information defined in the index;

if the second current digit from the target number is a decimal place indicator, the first current digit from the given number is any one of 1, 2, 3, 4, 5, 6, 7, 8, 9, set the corresponding next state information to include a value defining the "greater" state regardless of the current state information defined in the index;

if one of the first current digit from the given number and the second current digit from the target number is a space character, and the other one of the first current digit and the second current digit is a decimal place indicator, set the corresponding next state to include a state value defining a corresponding final/intermediate state of the current state information defined by the index, and an indicator defining the reader of the number in which the space character appears will be moved back by one position.

21. The system according to claim 20, wherein the processor is further configured to

if the current state information is "possibly greater", set the corresponding next state to include a state value defining a corresponding final state "greater",

if the current state information is "possibly smaller", set the corresponding next state information to include a state value defining a corresponding final state "smaller",

if the current state information is "possibly equal", set the corresponding next state information to include a state value defining a state "possibly equal with decimal place indicator".

22. The system according to claim 13, the processor is further configured to

if the given number and the target number includes negative or positive signs, determine the final comparison result based on a preset final result table which maintains a plurality of mappings each between a final comparison result and a pair of signs of the given number and the target number.

23. The system according to claim 13, wherein the given number is a given telephone number, the target number is a target telephone number stored in a text file,

in order to determine whether the given telephone number is same as the target telephone number, the processor is further configured to

if the next state information identified based on the current digit pair is not “equal” or “possibly equal”, determine that the given telephone number is not same as the target

5 telephone number, and

if the final comparison result is that the given telephone number is same as the target telephone number, retrieve a corresponding operator name from a predetermined position in the text file.