Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2650571 C 2015/08/18

(11)(21) **2 650 571**

(12) BREVET CANADIEN / CANADIAN PATENT

(13) **C**

(86) **Date de dépôt PCT/PCT Filing Date:** 2007/04/24
(87) **Date publication PCT/PCT Publication Date:** 2007/11/15
(45) **Date de délivrance/Issue Date:** 2015/08/18
(85) **Entrée phase nationale/National Entry:** 2008/10/23
(86) **N° demande PCT/PCT Application No.:** US 2007/067319
(87) **N° publication PCT/PCT Publication No.:** 2007/130818
(30) **Priorité/Priority:** 2006/05/02 (US11/381,214)

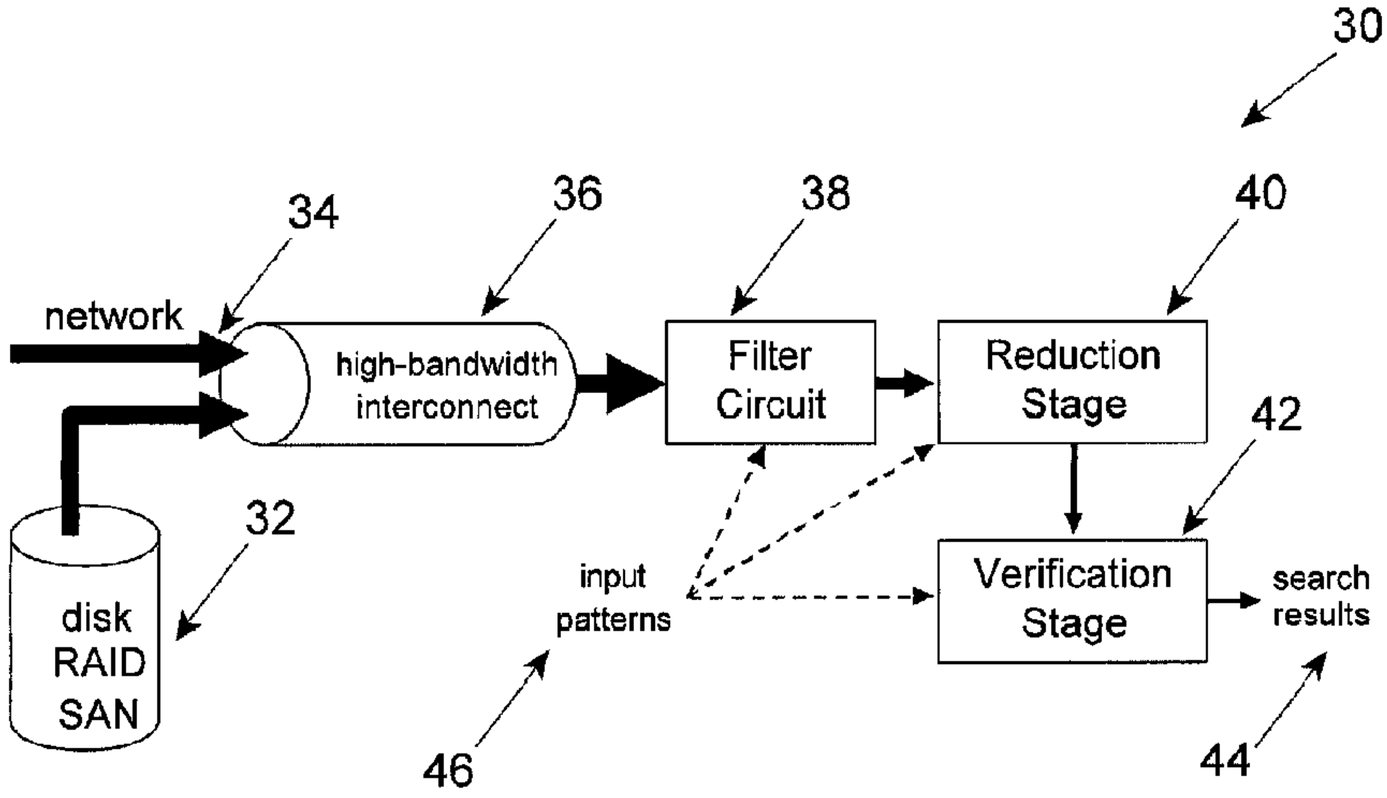
(51) **Cl.Int./Int.Cl.** ***H04L 29/06*** (2006.01)

(72) **Inventeur/Inventor:**
TAYLOR, DAVID EDWARD, US

(73) **Propriétaire/Owner:**
IP RESERVOIR, LLC, US

(74) **Agent:** OSLER, HOSKIN & HARCOURT LLP

(54) **Titre : PROCEDE ET APPAREIL POUR L'APPARIEMENT APPROXIMATIF DE MOTIFS**
(54) **Title: METHOD AND APPARATUS FOR APPROXIMATE PATTERN MATCHING**



(57) **Abrégé/Abstract:**
A system and method for inspecting a data stream for data segments matching one or more patterns each having a predetermined allowable error, which includes filtering a data stream for a plurality of patterns of symbol combinations with a plurality of parallel filter mechanisms, detecting a plurality of potential pattern piece matches, identifying a plurality of potentially matching patterns, reducing the identified plurality of potentially matching patterns to a set of potentially matching patterns with a reduction stage, providing associated data and the reduced set of potentially matching patterns, each having an associated allowable error, to a verification stage, and verifying presence of a pattern match in the data stream from the plurality of patterns of symbol combinations and associated allowable errors with the verification stage.

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
15 November 2007 (15.11.2007)

PCT

(10) International Publication Number
WO 2007/130818 A3

(51) International Patent Classification:
H04L 29/06 (2006.01)

(21) International Application Number:
PCT/US2007/067319

(22) International Filing Date: 24 April 2007 (24.04.2007)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/381,214 2 May 2006 (02.05.2006) US

(71) Applicant *(for all designated States except US)*: **EXEGY INCORPORATED** [US/US]; Suite 300, 3668 South Geyer Road, St. Louis, MO 63127 (US).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **TAYLOR, David Edward** [US/US]; 3448 Missouri Avenue, St. Louis, MO 63118 (US).

(74) Agents: **KERCHER, Kevin M.** et al.; Thompson Coburn LLP, One US Bank Plaza, St. Louis, Missouri 63101 (US).

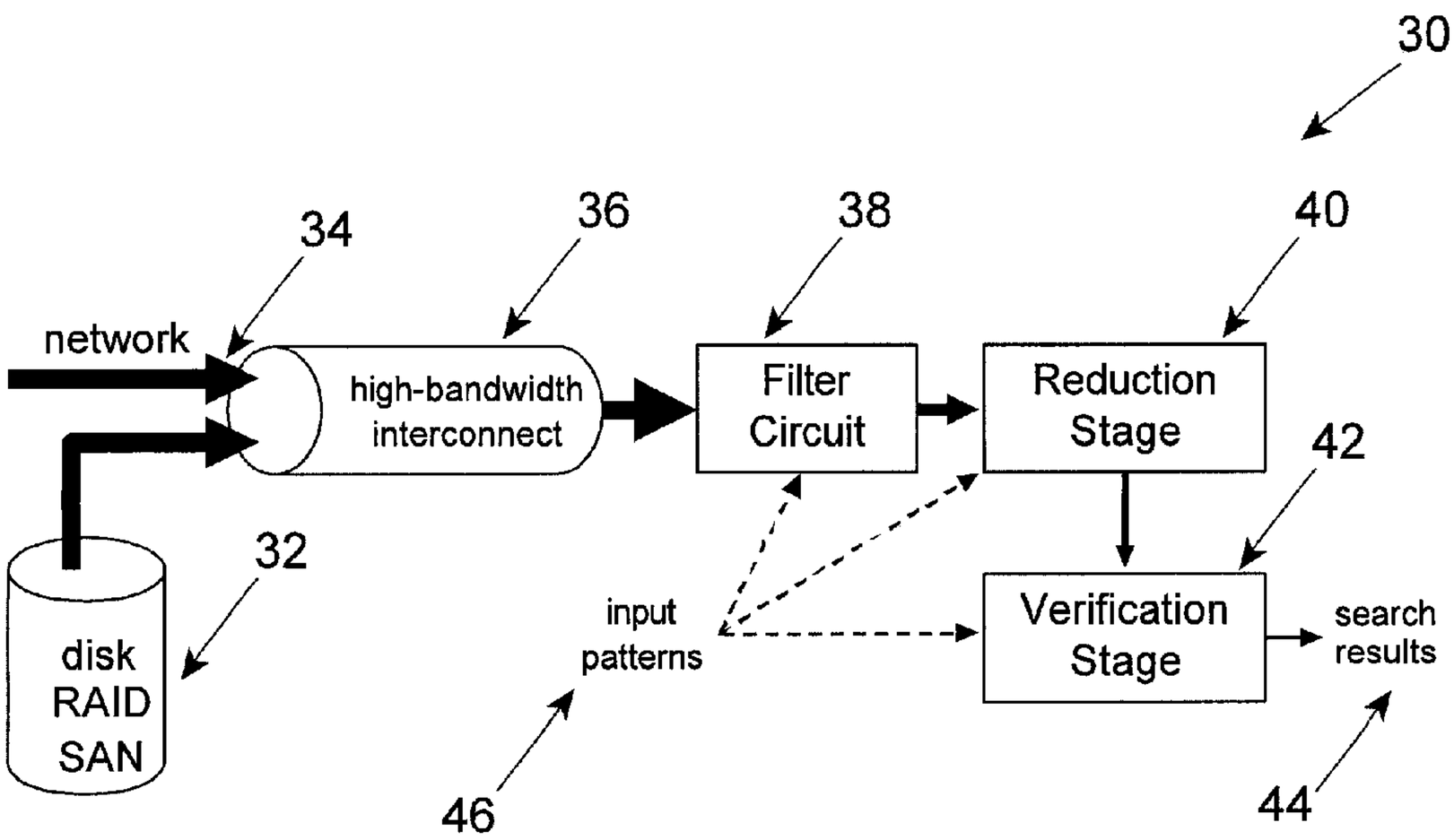
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP,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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

(88) Date of publication of the international search report:
28 February 2008

(54) Title: METHOD AND APPARATUS FOR APPROXIMATE PATTERN MATCHING



(57) Abstract: A system and method for inspecting a data stream for data segments matching one or more patterns each having a predetermined allowable error, which includes filtering a data stream for a plurality of patterns of symbol combinations with a plurality of parallel filter mechanisms, detecting a plurality of potential pattern piece matches, identifying a plurality of potentially matching patterns, reducing the identified plurality of potentially matching patterns to a set of potentially matching patterns with a reduction stage, providing associated data and the reduced set of potentially matching patterns, each having an associated allowable error, to a verification stage, and verifying presence of a pattern match in the data stream from the plurality of patterns of symbol combinations and associated allowable errors with the verification stage.

WO 2007/130818 A3

METHOD AND APPARATUS FOR APPROXIMATE PATTERN MATCHING

FIELD OF THE INVENTION

[0001] The present invention relates to the field of approximate pattern matching with a large set of patterns. In particular, the present invention relates to a scalable filtering circuit and reduction stage for approximate pattern matching with a large group of patterns.

BACKGROUND OF THE INVENTION

[0002] Approximate pattern or string matching is a significant problem that arises in many important applications. These can include, but are not limited to, computational biology, databases and computer communications. This task includes searching for matches between the specified pattern or set of patterns while typically permitting a specified number of errors. As an example, one may desire to search for the word "queuing" while allowing for two errors. This could return results such as the word "queueing" with one character insertion and "cueing" with one character substitution and one character deletion. By allowing a specified number of errors, this allows the search to catch typical spelling variations or errors and still find the desired pattern. Approximate pattern matching is not only a complex task but requires a tremendous amount of computer resources.

[0003] Typically, there is a fast filtering step that is followed by the verification step that performs the full approximate matching function. An example of this prior art filtering technique is shown by referring to FIG. 1 and is generally indicated by numeral 10. This typical approach is to slice a pattern "P", as indicated by numeral 12, into k + 1 pattern pieces, which are a sequence of non-overlapping sub-patterns, and search for exact matches between the text and the pattern pieces. In this case, "k" is equal to the number of allowable errors, which is the maximum edit distance $ed(T_{i...j},P)$, which is indicated in this nonlimiting example by the numeral two (2) as indicated by numeral 14.

[0004] A data string $T_{i...j}$ 16 is then analyzed for an occurrence of at least one substring of the data string 16 that matches at least one of the non-overlapping sub-

patterns associated with pattern "P" 12. This approach relies on the following properties:

a. If string $S = T_{a...b}$ matches pattern P with at most k errors, and $P = p_1...p_j$ (a sequence of non-overlapping sub-patterns), then some sub-string of S matches at least one of the p_i's with at most $\lfloor k/j \rfloor$ errors

b. If there are character positions $i \leq j$ such that $ed(T_{i...j},P) \leq k$, then $T_{j-m+1...j}$ includes at least $m-k$ characters of P where m is the size of the pattern (in characters)

c. Therefore, if we slice P into $k+1$ pieces (non-overlapping sub-patterns), then at least one of the pieces must match exactly

[0005] Therefore, if we slice "P" 12 by the total number of errors "k" 14 plus one (1) into non-overlapping sub-pattern pieces then at least one of the non-overlapping sub-pattern pieces must match exactly. As shown in the Example of FIG. 1, the data string $T_{i...j}$ 16 is divided into k+1 or three (3) pieces of non-overlapping sub-patterns. Therefore the three (3) pieces are "abra" indicated by numeral 18, "cada" indicated by numeral 20, and "bra" indicated by numeral 22. In this example, "cada" indicated by numeral 20 is an exact match with two errors where the letters "br" are replaced and the letter "b" is deleted.

[0006] There is a significant need for a fast and cost effective mechanism for pattern matching utilizing a substantial amount of input data with a considerable set of potentially matching patterns.

SUMMARY OF INVENTION

[0007] In one aspect of this invention, a method for inspecting a data stream for data segments matching one or more patterns each having a predetermined allowable error with at least one search engine is disclosed. This method includes filtering a data stream for a plurality of patterns of symbol combinations with a plurality of parallel filter mechanisms each configured to detect one or more patterns each with an associated allowable error, detecting a plurality of potential pattern piece matches with the plurality of parallel filter mechanisms, identifying a plurality of potentially matching patterns, each having an associated allowable error, from the plurality of parallel filter mechanisms, reducing the identified plurality of potentially matching patterns to a set of potentially matching patterns, each having an associated allowable

error with a reduction stage, providing associated data and the reduced set of potentially matching patterns, each having an associated allowable error, to a verification stage, and verifying presence of a pattern match in the data stream from the plurality of patterns of symbol combinations and associated allowable errors with the verification stage that includes an approximate match engine utilizing the associated data and the reduced set of potentially matching patterns.

[0008] In another aspect of this invention, a method for inspecting a data stream for data segments matching one or more patterns each having a predetermined allowable error with at least one search engine is disclosed. This method includes filtering a data stream for a plurality of patterns of symbol combinations with a plurality of parallel filter mechanisms each configured to detect one or more patterns each with an associated allowable error, wherein the plurality of parallel filter mechanisms is a group consisting of a set of parallel Bloom filters, a set of parallel Bloom filter arrays or a set of parallel Bloom filter arrays that utilize a single hash key generator, detecting a plurality of potential pattern piece matches with the plurality of parallel filter mechanisms, identifying a plurality of potentially matching patterns, each having an associated allowable error, from the plurality of parallel filter mechanisms, reducing the identified plurality of potentially matching patterns to a set of potentially matching patterns, each having an associated allowable error, providing associated data and the reduced set of potentially matching patterns, each having an associated allowable error, to a verification stage, and verifying presence of a pattern match in the data stream from the plurality of patterns of symbol combinations and associated allowable errors with the verification stage that includes an approximate match engine utilizing the associated data and the reduced set of potentially matching patterns.

[0009] In still another aspect of this invention, a method and system for inspecting a data stream for data segments matching one or more patterns each having a predetermined allowable error with at least one search engine is disclosed. This method includes utilizing a single hash key generator for extracting a plurality of hash values from a single hash value for inspecting the data stream for data segments matching one or more pattern pieces with false positive errors with at least one search engine, and utilizing the plurality of hash values with a plurality of parallel Bloom filter arrays.

[0010] In yet another aspect of this invention, a system for inspecting a data stream for data segments matching one or more patterns each having a predetermined allowable error with at least one search engine is disclosed. This system includes a filter stage, which utilizes a plurality of parallel filter mechanisms each configured to detect one or more patterns, each with an associated allowable error, that filter a data stream for a plurality of patterns of symbol combinations and detect a plurality of potential pattern piece matches, and identify a plurality of potentially matching patterns, each having an associated allowable error, a reduction stage, which reduces the identified plurality of potentially matching patterns to a set of potentially matching patterns, each having an associated allowable error, and a verification stage, which includes an approximate match engine, that receives and utilizes associated data and the reduced set of potentially matching patterns and associated allowable errors to verify a presence of a pattern match in the data stream from the plurality of patterns of symbol combinations.

[0011] In yet another aspect of this invention, a system for inspecting a data stream for data segments matching one or more patterns each having a predetermined allowable error with at least one search engine is disclosed. The system includes a plurality of parallel filter mechanisms, wherein the plurality of parallel filter mechanisms is a group consisting of a set of parallel Bloom filters, a set of parallel Bloom filter arrays or a set of parallel Bloom filter arrays that utilize a single hash key generator, each configured to detect one or more patterns, each with an associated allowable error, that filter a data stream for a plurality of patterns of symbol combinations and detect a plurality of potential pattern piece matches and identify a plurality of potentially matching patterns, each having an associated allowable error, a reduction stage that reduces the identified plurality of potentially matching patterns to a set of potentially matching patterns, each having an associated allowable error, and a verification stage, which includes an approximate match engine, that receives and utilizes associated data and the reduced set of potentially matching patterns and associated allowable errors to verify a presence of a pattern match in the data stream from the plurality of patterns of symbol combinations.

[0012] Illustrative, but nonlimiting, examples of potential application of the present invention include: an intrusion detection system (IDS) for computer communication

networks; computational biology and genetics; text searches for structured and unstructured text; and text searches from optical character scans (OCS).

[0013] Additional aspects of the present invention include, but are not limited to: a filtering technique for approximate matching with multiple patterns where each pattern may specify its allowable errors that can include a large number of pattern pieces, e.g., tens of thousands of patterns or more; utilizing a parallel set of exact match engines, one for each pattern piece length, to perform parallel match operations and to support a wide variety of (pattern length, allowable error) combinations; allowing each pattern to have a specified number of errors; amenability to parallel hardware search implementation and such implementation can provide fast search results; simplifying a verification stage by limiting the number of potentially matching patterns for a region of text, allowing the verification engine to process additional potential search results in a shorter period of time, which allows the total system to scale in capacity while operating at very high speeds; and utilizing a Bloom filter array for each exact match engine; and efficiently implementing each Bloom filter array by using only one hash function generator.

[0014] Still another aspect of present invention is the reduction stage wherein the scope of the search in the verification stage is reduced with a smaller set of possibly matching patterns. These techniques use a layer of indirection between pieces and patterns which allows each pattern and its allowable errors to be stored only once. There is a first illustrative technique that simplifies the data structures, making them amenable to hardware implementation. This technique includes a lookup using a bin index to retrieve the piece identifiers for the potentially matching pieces. A second lookup uses the piece identifiers to retrieve the pattern identifiers for the patterns that include the pieces. A third lookup uses the pattern identifiers to retrieve the pattern and associated allowable error pairs to be considered by the verification engine. There is a second illustrative, but nonlimiting technique that utilizes the text pieces that produced matches in the exact match engines to resolve the pattern identifiers for the patterns that include the piece. The pattern identifiers are used to retrieve the pairs of patterns and associated allowable errors to be considered by the verification engine.

[0015] These are merely some of the innumerable aspects of the present invention and should not be deemed an all-inclusive listing of the innumerable aspects associated with the present invention.

BRIEF DESCRIPTION OF DRAWINGS

[0016] For a better understanding of the present invention, reference may be made to the accompanying drawings in which:

[0017] FIG. 1 provides an illustrative overview of a prior art approximate pattern matching technique;

[0018] FIG. 2 provides an exemplary block diagram of the present invention including a data source, a filter circuit, a reduction stage and a verification stage;

[0019] FIG. 3 is an illustrative, but nonlimiting, block diagram of the present invention including Bloom filters with a match detection function; a reduction stage and a verification stage having approximate pattern matching;

[0020] FIG. 4 provides an exemplary block diagram of a first filtering stage processing technique using a Bloom filter array;

[0021] FIG. 5 provides an exemplary block diagram of a second filtering stage processing technique using a Bloom filter array with a single hash function generator;

[0022] FIG. 6 provides an exemplary block diagram of a first reduction stage processing technique; and

[0023] FIG. 7 provides an exemplary block diagram of a second reduction stage processing technique.

DETAILED DESCRIPTION OF THE INVENTION

[0024] In the following detailed description, numerous specific details are set forth in order to provide a thorough understanding of the invention. However, it will be understood by those skilled in the art that the present invention may be practiced without these specific details. In other instances, well-known methods, procedures, and components have not been described in detail so as to obscure the present invention.

[0025] The present invention is a scalable filtering circuit for approximate pattern matching with a large set of patterns. The filtering circuit checks for potential matches between a set of stored patterns, where each pattern specifies the number of allowable errors, and an input stream of characters. The number of allowable errors is predetermined or specified using the general edit distance measure that counts the number of single character additions, deletions, and substitutions. When a potential

match is detected, the location or locations in the input data stream is identified as well as the matching pattern or plurality of potentially matching patterns. The present invention is designed to operate in concert with a verification stage that looks for an approximate match in the data segment(s) of the input data stream utilizing the previously identified potentially matching pattern(s) from the total number of potentially matching patterns.

[0026] The methodology for searching for a single pattern can be stated as follows: identifying instances of pattern "P" in text "T" with "k" allowable errors, where "k" is the maximum edit distance. The edit distance is defined as the number of single character insertions, deletions, and substitutions with all errors typically, but not necessarily share the same weighting. In general, other types of errors such as transpositions may be included in the distance measure and each type of error may be assigned a unique weight.

[0027] Assuming that "m" is the size of the pattern in characters and "n" is the size of the text in characters, then the error level can be defined for a particular pattern as the ratio of the number of allowable errors and the size of the pattern, which is "$\alpha = k/m$". The error level and the size of the alphabet from which the pattern and text are constructed, "σ," affect the probability that matches will be found. An expression for the match probability, "f(m,k)" assuming a randomly constructed text and a randomly constructed pattern is provided by the following equation:

$$f(m,k) = \left(\frac{e^2}{\sigma(1-\alpha)^2} \right)^{m(1-\alpha)}$$

This is where "e" is the base of the natural logarithm.

[0028] It is believed that filtering algorithms achieve better performance for a variety of approximate pattern matching problems. The general approach being to perform a simple search on a small section of text to identify potential matches. When a potential match is found, the region of text is examined to see if it is, in fact, an approximate match for a specified pattern. In general, verification is performed by any approximate pattern matching algorithm and may be tightly or loosely coupled to a filtering operation.

[0029] A general schematic of the system of the present invention is shown in FIG. 2 and is indicated by numeral 30. Large amounts of data can be provided as input

through either a communication link, a disk, a redundant array of independent disks (RAID), or storage area network (SAN), as well as a wide variety of other data sources capable of feeding a filter circuit with high data speed. This data input is indicated by numeral 32 can also be provided through a network input that is indicated by numeral 34. High speed data can be provided through a high-bandwidth interconnect indicated by numeral 36 at high speeds. This high speed data is then passed through a filter circuit 38 that scans the input data for potential matches for a set of input patterns. There is then a reduction stage 40 between the filter circuit 38 and a verification stage 42 that narrows the set of potentially matching patterns that must be considered by the verification stage 42 when processing data segments that produce a match in the filter circuit 38. The verification stage 42 performs a full approximate operation to verify whether or not there is a match for a set of input patterns 46. Search results are then provided as indicated by numeral 44. Predetermined input patterns 46 are provided to the filter circuit 38, the reduction stage 40 and the verification stage 42.

[0030] In a particular window of text, it is possible to search for an exact match and any of the pattern pieces specified by a predetermined number "r" patterns. If a specific pattern "i" allows "k_i" errors, then the total number of pattern pieces is indicated by the equation:

$$p = \sum_{i=1}^{r} (k_i + 1)$$

[0031] This filtering approach is utilized with parallel filter mechanisms which can include a parallel set of Bloom filters, a set of parallel Bloom filter arrays, or a set of Bloom filter arrays that utilize a single hash function generator. As shown in FIG. 3, which is the schematic of the basic hardware implementation of the present invention is indicated by numeral 50 and includes a number of Bloom filters indicated by numeral 54. In a classical Bloom filter 54, elements are inserted into a set using "b" hash values were the element is utilized as the key and where each hash value identifies a bit position in a B-bit vector. The bits at each of the b bit positions are preferably set to one (1). If a bit is already set to one (1), then no change will be made. In order to test whether or not a particular element is a member of the set represented by a Bloom filter 54, the element and the same b hash functions are

utilized to compute b hash values. If all the b bits in the vector are set to one (1), then the element is declared to be a member of the set.

[0032] A Bloom filter 54 will not produce a false negative. If an element is a member of a set, then the b bit positions in the B-bit vector are set to one (1) when the element is inserted into the set. The insertion of additional elements in the set does not reset any of the bits in the vector. However, Bloom filters 54 do produce false positives with a determined probability. This probability can be computed by the equation:

$$f = \left(1 - e^{\frac{-pb}{B}}\right)^b$$

If the following relationship holds:

$$b = \frac{B}{p} \ln 2 \text{ then: } f = (1/2)^b$$

[0033] Approximate match filtering on multiple patterns and allowing each pattern to specify its allowable errors produces sets of pattern pieces of various lengths. Preferably, but not necessarily, the bloom filters 54 store fixed-length elements with one bloom filter circuit 54 for each possible pattern piece length. Therefore, the range of possible pattern piece lengths are constrained within a range.

[0034] If l_{min} is the minimum pattern piece length then l_{min} is less than or equal to the value of the size of the pattern m divided by the maximum edit distance k plus one (1):

$$l_{\min} \leq \left\lfloor \frac{m}{k+1} \right\rfloor$$

[0035] If l_{max} is the maximum piece length, then l_{max} is greater than or equal to the value of the size of the pattern m divided by the maximum edit distance k plus one (1):

$$\left\lceil \frac{m}{k+1} \right\rceil \leq l_{\max}$$

[0036] The total number of Bloom filters 54 that are required when each Bloom filter of the Bloom filters 54 corresponds to a pattern piece length is:

$l_{max} - l_{min}$ + one (1).

[0037] A preferred approach is to query each of the Bloom filters 54 in parallel, as shown by the schematic provided by numeral 50 in FIG. 3. Each one of the Bloom

filters 54 correspond to a pattern piece length so that various strides of the text window can be selected as an input key to each Bloom filter 54.

[0038] If any of the Bloom filters 54 result in a detected match 56, then the segment of data or text window 58, the location of the segment of data in the input stream 59 and additional match meta data 60 are sent to a reduction stage 40. The techniques utilized in the reduction stage 40 can narrow the potentially matching patterns significantly, e.g., over 10,000 to less than 10.

[0039] The result passing from the reduction stage 40 goes to the verification stage 42, which includes the approximate match search engine. By reducing the number of candidate patterns to be considered by the verification stage 42, allowing the verification stage to process more potential search results in a given amount of time, and thus allowing the total system to scale in capacity while operating at high speeds.

[0040] A Bloom filter array 54 typically minimizes the number of memory accesses, e.g., random access memory (RAM), required for a set membership query. Moreover, the Bloom filter array 54 partitions the B-bit vector into "W" vectors of size "q = B/W" where "q" is the word size of the memory. There can preferably be an even distribution of stored elements over the "W" vectors (memory words) using a pre-filter hash function. This creates an array of "W" "q"-bit Bloom filters.

[0041] The bits in the "q"-bit Bloom filters 54 are set, during programming, and then checked, during queries, using "b" hash functions. Querying a Bloom filter array 54 requires one (1) memory read to fetch the "q"-bit vector. Using a register 80 and bit-select circuitry 82, checking the bit locations specified by "b" hash functions may be performed on-chip, in a pipelined fashion, as shown in FIG. 4 and generally indicated by numeral 70.

[0042] In this application, a key (i.e. pattern piece) is indicated by numeral 72. The key is used by the pre-filter hash function 73 to identify a particular "q"-bit vector in the listing of "w" vectors. The particular "q"-bit vector in the listing of "w" vectors is indicated by column 74 in memory, e.g., RAM, wherein a particular and illustrative vector is identified by numeral 76. These queries are checked with series of "b" hash functions indicated by numeral 78 identifying bit positions within a register 80, which is then provided to a match detection function 82. If all "b" bit positions are set to a one (1), then the key is a pattern piece for a potentially matching pattern.

[0043] Preferably, the amount of logic required to implement a Bloom filter array 54 can be minimized, as shown in FIG. 5. This logic is generally indicated by numeral 90. There is a single hash function indicated by numeral 92. There is the generation of a single random value. A subset of the bits from this random value are utilized to construct the pre-filter hash address and "b" filter bit-positions that is indicated by numeral 94. The particular "q"-bit vector in the listing of "w" vectors is indicated by column 96 in memory, e.g., RAM, wherein a particular and illustrative vector is identified by numeral 98. This particular vector 98 is passed to a register 100 and then on to a match pattern detection function 102. This Bit select value 94 must be at least $\log_2 (W) + (b * \log_2 (q))$ bits in size. The H_3 class of hash functions 92 is an illustrative, but nonlimiting, example of hash functions that can produce wide enough values for this application.

[0044] An illustrated, but nonlimiting example of reconfigurable hardware that could be utilized includes FPGAs, i.e., field programmable gate arrays, which includes a Xilinx® VirtexII® 4000 series FPGA. Xilinx, Inc., is a Delaware corporation, having a place of business at 2100 Logic Drive, San Jose, California 95124-3400. An illustrated, but nonlimiting, example of the embedded memory in the VirtexII® series of devices include One Hundred and Twenty (120) of the eighteen (18) kilobyte block random access memories (BlockRAMs). These BlockRAMs can be configured to various size words with an illustrative, but nonlimiting, maximum word length of thirty-six (36) bits by Five Hundred and Twelve (512) words.

[0045] Utilizing the previous expression for probability of a false positive and assuming uniform hashing performance, the Bloom filter array 54 implemented with an eighteen (18) kilobyte BlockRAM can represent a set of 3,194 elements with a false probability of 0.063 when the number of b bit positions equals four (4). When number of b bit positions equals three (3), the capacity increases to 4,259 elements but the false positive probability increases to 0.125.

[0046] As previously stated, one Bloom filter array 54 is required for each unique pattern piece length. There is also consideration of the number of parallel circuits that are required to keep pace with the data input rate. In an illustrative, but nonlimiting example, a system that accepts eight (8) new ASCII characters per cycle (64-bit interface) requires eight (8) instances of the circuit operating in parallel. For the VirtexII 4000® FPGA, there are at most fifteen (15) BlockRAMs available for each

circuit instance. In order to have BlockRAM resources available for interface buffers, this results in limiting the BlockRAMs to a lower number, e.g., fourteen BlockRAMs. This illustrative, but nonlimiting, resource allocation can place a constraint on the length of pattern pieces and the combination of pattern piece and allowable error. This in turn places a limit on the maximum error level. When "m" is equal to the size of the pattern, "p" is equal to number of pattern pieces and "k" is the number of allowable errors or edit distance then the following equations are applicable:

$m_{min} = p_{min} (k+1)$

$m_{max} = p_{max} (k+1)$

$$\alpha_{max} = \frac{k}{m_{min}} = \frac{k}{p_{min}(k+1)}$$

where α = ratio of the number of errors divided by the size of the pattern.

[0047] The following Table 1 is when "α ", i.e., ratio of the number of errors divided by the size of the pattern is less than or equal to one (1):

Table 1

k	m_{min}	m_{max}
0	1	14
1	2	28
2	3	42
3	4	56
…	…	…

[0048] The following Table 2 is when "α ", i.e., ratio of the number of errors divided by the size of the pattern is less than or equal to one-half (1/2):

Table 2

k	m_{min}	m_{max}
0	2	15
1	4	30
2	6	45
3	8	60
…	…	…

[0049] The following Table 3 is when "α" , i.e., ratio of the number of errors divided by the size of the pattern is less than or equal to one-third (1/3):

Table 3		
k	m_{min}	m_{max}
0	3	16
1	6	32
2	9	48
3	12	64
...	...	...

[0050] Therefore in the second example and Table 2, when "α" , i.e., ratio of the number of errors divided by the size of the pattern is less than or equal to one-half (1/2) and when the pattern allows no errors, i.e., "k = 0", there must be at least two (2) characters and no more than fifteen (15) characters. A pattern that allows one error, i.e., "k = 1", must contain at least four (4) characters and no more than thirty (30) characters. Although a wide variety of admissible pattern sizes and allowable errors can be utilized, it is believed that a pattern is at least two (2) characters and no more than fifteen (15) characters will be a workable constraint for most text searches in English, however, this should not be construed as a limit.

[0051] A rough capacity estimate can be developed by assuming that pieces are uniformly distributed over a range of allowable lengths. In an illustrative, but nonlimiting example, if each Bloom filter array 54 has a capacity of approximately Three Thousand (3,000) pattern pieces then the system has an aggregate capacity of Forty-Two Thousand (42,000) pattern pieces. If it is assumed that each pattern can be divided into three (3) pattern pieces, then the system has a capacity of Fourteen Thousand (14,000) patterns.

[0052] Once a potential match has been detected one or more pattern piece lengths 56, as shown in FIG. 3, then the region of text must be examined by the verification stage 42 in order to determine whether or not there is an approximate match for one of the "r" patterns. Since "r" may be on the order of 10,000 patterns or more, there is a need to narrow the scope of the search that the verification stage 42 must perform. This is where the reduction stage 40 provides a valuable role of reducing the set of

possible matching patterns (pattern set) for the verification stage 42 to consider. As long as the parameters fall within the constraints, there is an assumption that the number of allowable errors may be specified for each pattern.

[0053] Throughout this patent application as shown in FIG. 3, the filter stage 38, the reduction stage 40 and/or the verification stage 42, as shown in FIG. 3, can use at least one reconfigurable logic device, e.g., Field Programmable Gate Array ("FPGA") or at least one integrated circuit, e.g., Application-Specific Integrated Circuit ("ASIC").

[0054] There are two illustrative, but nonlimiting, approaches to perform the reduction stage 40. The first approach is to simplify data searches utilized to resolve the pattern set indicated by numeral 120 in FIG. 6. This allows a data string to come into a shift register 52, which then passes into a filter stage or circuit 38 that preferably includes a Bloom filter array 54. With this approach, the objective is to utilize some, if not all, of the hash values computed by the Bloom filter array 54 in the filter stage or circuit 38 as an index into a table 128, e.g., BinIndex 124.

[0055] For example a pattern piece 121 can be received by the filter stage or circuit 38 and is received as hash values 126 comprising the BinIndex 124. The entries in this first table 128 contain identifiers 127 for the pattern pieces, e.g., PieceIDs, which map to the hash values 126 comprising the BinIndex 124. For example, there is an illustrative identifier, e.g., $PieceID_1$ and $PieceID_4$, which is associated with the example pattern piece 121. The identifiers 127 for the pattern pieces, e.g., PieceIDs, are a unique binary tag assigned to each pattern piece.

[0056] There is a second table 132 that utilizes the identifiers 128 for the pattern pieces, e.g., PieceIDs, to index one or more pattern identifiers for the set of potentially matching patterns. Since one or more patterns may specify a particular pattern piece, the entries in the second table 132 contain one or more pattern identifiers, e.g., PIDs. Pattern identifiers, e.g., PIDs, are unique binary tags associated with each pattern. For example, with the illustrative identifier, 131, relates to two patterns 137 and 138.

[0057] There is a third table 134 that utilizes the pattern identifiers, e.g. PIDs, to index one or more (pattern, allowable error) pairs. The identified set or plurality of potentially matching patterns, each with predetermined allowable errors, e.g. (pattern, allowable error) pairs 137 and 138, is then created as indicated by numeral 136.

[0058] This pattern set of potential matches 136 is passed onto the verification stage 42, which includes evaluation by an approximate match engine 142 for matching patterns and associated predetermined allowable errors. There only needs to be one copy of tables for the pattern piece identifiers 128, e.g., PieceIDs, pattern identifiers 132, e.g., PIDs and patterns 134 so long as the number of lookups per cycle does not exceed the amount of lookups supported by the memories.

[0059] There is a second and preferred methodology for the reduction stage 40, which is generally indicated by numeral 150 in FIG. 7. This allows a data string to pass into a filter stage or circuit 38 that preferably includes a Bloom filter array 54. With this approach, actual data segments or pieces that produce a match in the Bloom filter array 54 are utilized to resolve the pattern identifiers for the patterns 162 that specify those pattern pieces. The data segments or pieces that produce a match are used to identify one or more entries in data structures, indicated by numerals 156, 158 and 160, which store pattern identifiers, e.g., PIDs, for the patterns that specify the associated pattern pieces. Illustrative, but nonlimiting examples of suitable data structures are a hash table and a balanced search tree. The methodology may include one or more data structures. In the illustrative, but nonlimiting example 150 in FIG. 7, one data structure is allocated for each pattern piece length.

[0060] In an illustrative, but nonlimiting example, two data segments or pieces that produce a match in the Bloom filter array 54 are identified by numerals 121 and 123. Data piece 121 identifies the entry 157 in the data structure as part of the reduction stage 40. These data structures 156, 158 and 160 can include a wide variety of different structures, e.g., decision tree, hash table, and so forth. The result of these lookup(s) is a set of pattern identifiers for the patterns in the pattern set. As with the prior approach, these pattern identifiers, e.g., PIDs, 156, 158 and 160 are utilized to retrieve patterns and associated allowable errors from a table 40 prior to a step of verification. The previously referenced entry 157 identify patterns 163 and 165 and associated allowable errors to produce a set of potential matches 164. The set of potential matches and associated predetermined allowable errors 164 is then evaluated by an approximate match engine 166 in the verification stage 42.

[0061] This approach resolves the set of pattern identifiers in a single step instead of two steps and also eliminates false positive errors produced by the Bloom filter arrays 54. Also, since the actual data segment is utilized to locate entries in the pattern

identifier structure(s), an explicit match is performed. If there is no entry in the table, then a false positive is detected and no pattern identifiers, e.g., PIDs, are passed onto the verification stage 42 for that particular pattern piece length. The tradeoff is that the data structures can be more complex and the implementation more resource intensive depending on the implementation.

[0062] Therefore, this is a scalable design for a filtering circuit 38 and reduction stage 40 for approximate pattern matching on multiple patterns that are amenable to hardware implementation. In addition to the thousands of patterns, multiple filter circuits can support multiple input symbols per cycle. Utilizing the high performance filtering circuit 38 and the reduction stage 40, the performance requirements placed on a verification stage 40 can be analyzed. For the purpose of this analysis, we assume that all patterns specify the same number of allowable errors, "k". Effective load on the verification stage is determined as the probability of a match and the expected size of the set of potentially matching patterns. The probability of match is simply the sum of the probability that any of the "r(k+1)" pieces produce a match in a text window and the false positive probability of the Bloom filter arrays 54 where "r" is the predetermined number of "r" pattern pieces and "k" is the number of predetermined errors.

[0063] If "L" is the number of Bloom filter arrays 54 and assuming random text on an alphabet of "σ" characters, the probability of any of these pieces matching would be:

$$E[match] = \frac{r(k+1)}{\sigma^{\left\lfloor \frac{m}{k+1} \right\rfloor}}$$

[0064] The addition of the false positive probability of the Bloom filter arrays provides:

$$E[match] = \frac{r(k+1)}{\sigma^{\left\lfloor \frac{m}{k+1} \right\rfloor}} + Lf$$

[0065] By utilizing illustrative, but nonlimiting, example values L = 14, σ = 40, r = 14,000 and f = 0.0034, the match probability is highly sensitive to the minimum piece length. For example, when m = 5 and k = 1 (a minimum piece length of two (2) characters), then the match probability is one (1). If the pattern size is increased to six (6) (a minimum pattern piece length of three (3) characters), then the match

probability is 0.079. This result suggests that the minimum pattern piece size should be three (3) characters or more. In this situation, there will be an expectation of one match every twelve (12) cycles.

[0066] Utilizing the reduction stage 40 methodology shown in FIG. 6 with a series of index lookups, then given that at least one Bloom filter array 54 produces a match, the expected number of Bloom filter arrays 54 that will produce a match, i.e., the expected number of bin index lookups, is:

$$E[bins] \le 1 + \frac{L}{\sigma^{\left\lfloor \frac{m}{k+1} \right\rfloor}} + Lf$$

[0067] Under the assumption that the pattern pieces are uniformly distributed over "L" Bloom filter arrays 54 (pattern piece lengths) and uniformly distributed over the bins, the expected number of pattern piece identifiers 128, as shown in FIG. 6, per bin is:

$$E[PieceIds / bin] \le \frac{r(k+1)}{LW\left(\frac{B}{W}\right)^{b}}$$

where "B" is the size of memory used to implement the Bloom filter array 54, "W" is the number of words in the Bloom filter array 54 and "b" is the number of hash functions used in each Bloom filter in the Bloom filter array 54.

[0068] Finally, the expected number of patterns that specify a given pattern piece 131, as shown in FIG. 6, include:

$$E[patterns / PieceIDs] \le 1 + \frac{r}{\sigma^{\left\lfloor \frac{m}{k+1} \right\rfloor}}$$

[0069] Therefore, provided that at least one Bloom filter array 54 produces a match, the expected pattern size is:

$$E[patterns] \le \left(1 + \frac{L}{\sigma^{\left\lfloor m/k+1 \right\rfloor}} + Lf\right)\left(\frac{r(k+1)}{LW\left(\frac{B}{W}\right)^{b}}\right)\left(1 + \frac{r}{\sigma^{\left\lfloor \frac{m}{k+1} \right\rfloor}}\right)$$

[0070] Assuming uniform text, uniform distribution of piece lengths, uniform distributions (good hash functions) and with L = 14, σ = 40, r = 14,000, b = 3 and W

= 512, then the expected pattern set size is less than ten (10) for practical values of m and k (pattern length and allowable errors). The expected pattern set size quickly approaches one (1) as the alphabet size and/or pattern size increases.

[0071] The expression for the expected pattern set size when using other reduction stage approach indicated by FIG. 7 with the text segment as an index are similar and produce a slightly smaller pattern set size. In combination with the previous result, which is one pattern match every twelve (12) cycles, a conservative constraint for the average throughput of the verification stage 42 is approximately one (1) match on one pattern per every cycle.

[0072] Thus, there has been shown and described several embodiments of a novel invention. As is evident from the foregoing description, certain aspects of the present invention are not limited by the particular details of the examples illustrated herein, and it is therefore contemplated that other modifications and applications, or equivalents thereof, will occur to those skilled in the art. The terms "have," "having," "includes" and "including" and similar terms as used in the foregoing specification are used in the sense of "optional" or "may include" and not as "required." Many changes, modifications, variations and other uses and applications of the present construction will, however, become apparent to those skilled in the art after considering the specification and the accompanying drawings. All such changes, modifications, variations and other uses and applications which do not depart from the spirit and scope of the invention are deemed to be covered by the invention which is limited only by the claims that follow.

The embodiments of the present invention for which an exclusive property or privilege is claimed are defined as follows:

1. A method for inspecting a data stream for data segments approximately matching any of a plurality of patterns, the method comprising:

filtering a plurality of data substrings within the data stream with a plurality of parallel filter mechanisms to thereby detect

a plurality of potential matches between the data substrings and a plurality of pattern pieces, each pattern piece corresponding to at least one pattern, each data substring comprising a plurality of symbols;

reducing the detected potential matches to a plurality of pattern sets; each pattern set comprising (1) data representative of at least one pattern corresponding to a pattern piece which was a potential match to a data substring, and (2) data representative of an allowable error associated with the at least one pattern;

providing the pattern sets to a verification stage; and

verifying with the verification stage whether a data segment within the data stream is an approximate match to a pattern within a provided pattern set on the basis of the allowable error data within that provided pattern set.

2. The method according to Claim 1, further comprising, prior to the filtering step, receiving the data stream from the group consisting of a communication link, a redundant array of independent disks ("RAID"), and a storage area network. ("SAN").

3. The method according to Claim 1, wherein the plurality of parallel filter mechanisms is a group consisting of a set of parallel Bloom filters, a set of parallel Bloom filter arrays and a set of parallel Bloom filter arrays that utilize a single hash key generator.

4. The method according to Claim 1, wherein the plurality of parallel filter mechanisms are implemented on a member of the group consisting of at least one reconfigurable logic device and at least one integrated circuit, and wherein the reduction stage is implemented on a member of the group consisting of at least one reconfigurable logic device and at least one integrated circuit.

5. The method according to Claim 1, wherein the plurality of parallel filter mechanisms are implemented on a member of the group consisting of at least one Field Programmable

Gate Array ("FPGA") and at least one Application-Specific Integrated Circuit ("AS1C"), and wherein the reduction stage is implemented on a member of the group consisting of at least one Field Programmable Gate Array ("FPGA") and at least one Application-Specific Integrated Circuit ("ASIC").

6. The method according to Claim 1, wherein the plurality of parallel filter mechanisms include at least one Bloom filter array that is programmed with the pattern pieces, wherein the at least one Bloom filter array utilizes a single hash key generator for computing hash values that correspond to a bit vector and positions in the bit vector that identify whether a data substring is a potential match with one of the pattern pieces, the bit vector being stored in the Bloom filter array.

7. The method according to Claim 1, wherein the reducing step further includes:

utilizing data from the plurality of parallel filter mechanisms to lookup one or more pattern piece identifiers for at least one pattern piece that produced a potential match in the plurality of parallel filter mechanisms;

utilizing the one or more pattern piece identifiers for retrieving at least one pattern identifier, and

utilizing the at least one retrieved pattern identifier for retrieving the data representative of at least one pattern and the data representative of the allowable error associated with the at least one pattern to thereby determine at least a portion of a pattern set.

8. The method according to Claim 7, wherein the reducing step further includes utilizing a first lookup function with a bin index.

9. The method according to Claim 8, wherein the bin index is selected from the group consisting of different hash keys and portions of hash keys.

10. The method according to Claim 1, wherein the reducing step further includes:

utilizing each data substring that produced a potential match for retrieving at least one pattern identifier, and

utilizing the at least one retrieved pattern identifier for retrieving the data representative of at least one pattern and the data representative of the allowable error associated with the at least one pattern to thereby determine at least a portion of a pattern set.

11. The method according to Claim 3, wherein the reducing step further includes:

utilizing data from the plurality of parallel filter mechanisms to lookup one or more pattern piece identifiers for at least one pattern piece that produced a potential match in the plurality of parallel filter mechanisms;

utilizing the one or more pattern piece identifiers for retrieving at least one pattern identifier;

utilizing the at least one retrieved pattern identifier for retrieving the data representative of at least one pattern and the data representative of the allowable error associated with the at least one pattern to thereby determine at least a portion of a pattern set.

12. The method according to Claim 3, wherein the reducing step further includes:

utilizing each data substring that produced a potential match for retrieving at least one pattern identifier, and

utilizing the at least one retrieved pattern identifier for retrieving the data representative of at least one pattern and the data representative of the allowable error associated with the at least one pattern to thereby determine at least a portion of a pattern set.

13. A system for inspecting a data stream for data segments approximately matching any of a plurality of patterns, the system comprising:

a plurality of parallel filter mechanisms each configured to filter a plurality of data substrings within the data stream to thereby detect a plurality of potential matches between the data substrings and a plurality of pattern pieces, each pattern piece corresponding to at least one pattern, each data substring comprising a plurality of symbols;

a reduction stage configured to reduce the detected potential matches to a plurality of pattern sets, each pattern set comprising (1) data representative of at least one pattern corresponding to a pattern piece which was a potential match to a data substring, and (2) data representative of allowable error associated with the at least one pattern; and

a verification stage configured to receive the pattern sets and verify whether a data segment within the data stream is an approximate match to a pattern within a received pattern set on the basis of the allowable error data within that received pattern set.

14. The system according to Claim 13, further comprising a data stream provider selected as at least one member from the group consisting of a communication link, a redundant array of independent disks ("RAID"), and a storage area network ("SAN").

15. The system according to Claim 13, wherein the plurality of parallel filter mechanisms is a group consisting of a set of parallel Bloom filters, a set of parallel Bloom filter arrays or a set of parallel Bloom filter arrays that utilize a single hash key generator.

16. The system according to Claim 13, wherein the plurality of parallel filter mechanisms are implemented on a member of the group consisting of at least one reconfigurable logic device and at least one integrated circuit, and wherein the reduction stage is implemented on a member of the group consisting of at least one reconfigurable logic device and at least one integrated circuit.

17. The system according to Claim 13, wherein the plurality of parallel filter mechanisms are implemented on a member of the group consisting of at least one Field Programmable Gate Array ("FPGA") and at least one Application-Specific Integrated Circuit ("ASIC"), and wherein the reduction stage is implemented on a member of the group consisting of at least one Field Programmable Gate Array ("FPGA") and at least one Application-Specific Integrated Circuit ("ASIC").

18. The system according to Claim 13, wherein the plurality of parallel filter mechanisms includes at least one Bloom filter array that is programmed with the pattern pieces, wherein the at least one Bloom filter array is configured to utilize a single hash key generator for computing hash values that correspond to (1) a position of a particular Bloom filter in the at least one Bloom filter array and (2) a bit vector stored by the particular Bloom filter, and (3) positions of bits in the bit vector that identify whether a data substring is a potential match with at least one of the pattern pieces.

19. The system according to Claim 13, wherein the reduction stage is further configured to:

look up, on the basis of data from the plurality of parallel filter mechanisms, one or more pattern piece identifiers for at least one pattern piece that produced a potential match in the plurality of parallel filter mechanisms;

retrieve, on the basis of the one or more looked up pattern piece identifiers, at least one pattern identifier; and

retrieve, on the basis of the at least one retrieved pattern identifier, the data representative of at least one pattern and the data representative of the allowable error associated with the at least one pattern to thereby determine at least a portion of a pattern set.

20. The system according to Claim 19, wherein the reduction stage is further configured to utilize a first lookup function with a bin index.

21. The system according to Claim 20, wherein the bin index is selected as a member from the group consisting of different hash keys and portions of hash keys.

22. The system according to Claim 13, wherein the reduction stage is further configured to:

retrieve, on the basis of each data substring that produced a potential match, at least one pattern identifier, and

retrieve, on the basis of the at least one retrieved pattern identifier, the data representative of at least one pattern and the data representative of the allowable error associated with the at least one pattern to thereby determine at least a portion of a pattern set.

23. The system according to Claim 15, wherein the reduction stage is further configured to:

look up, on the basis of data from the plurality of parallel filter mechanisms, one or more pattern piece identifiers for at least one pattern piece that produced a potential match in the plurality of parallel filter mechanisms;

retrieve, on the basis of the one or more looked up pattern piece identifiers, at least one pattern identifier; and

retrieve, on the basis of the at least one retrieved pattern identifier, the data representative of at least one pattern and the data representative of the allowable error associated with the at least one pattern to thereby determine at least a portion of a pattern set.

24. The system according to Claim 15, wherein the reduction stage is further configured to:

retrieve, on the basis of each data substring that produced a potential match, at least one pattern identifier; and

retrieve, on the basis of the at least one retrieved pattern identifier, the data representative of at least one pattern and the data representative of the allowable error associated with the at least one pattern to thereby determine at least a portion of a pattern set.

25. A method for processing a plurality of data substrings within a data string for facilitating a determination as to whether the data string contains an approximate match to

any of a plurality of patterns, the data string comprising a plurality of data symbols, the method comprising:

querying a filter circuit with the data substrings to detect a plurality of potential matches, each potential match representing a potential match between a data substring and a pattern piece, wherein each pattern of a plurality of the patterns comprises a plurality of corresponding pattern pieces, the filter circuit being programmed with the pattern pieces;

applying the detected potential matches to a reduction stage to determine a plurality of pattern sets, each pattern set corresponding to a detected potential match and comprising (1) data representative of a pattern which corresponds to the pattern piece which produced the corresponding potential match, and (2) data representative of an allowable error associated with that pattern; and

delivering the determined pattern sets to an approximate match engine for a determination as to whether at least a portion of the data string is an approximate match to any of the patterns within the delivered pattern sets taking into consideration the allowable error data within the delivered pattern sets.

26. The method according to Claim 25 wherein the filter circuit comprises a Bloom filter circuit.

27. The method according to Claim 26 further comprising:

determining with the approximate match engine whether any approximate matches exist between the data string portion and the patterns within the delivered pattern sets taking into consideration the allowable error data within the delivered pattern sets.

28. The method according to Claim 27 further comprising:

slicing the patterns of at least a plurality of the patterns into a plurality of the pattern pieces; and

programming the Bloom filter circuit with the pattern pieces.

29. The method according to Claim 27 further comprising:

performing the querying step and the applying step in a pipelined manner.

30. The method according to Claim 29 further comprising:

performing the querying step and the applying step with reconfigurable hardware.

31. The method according to Claim 30 further comprising:

also performing the determining step with reconfigurable hardware.

32. The method according to Claim 29 further comprising:
performing the querying step and the applying step with at least one integrated circuit.

33. The method according to Claim 29 wherein the pattern pieces comprise a plurality of pattern pieces of different lengths.

34. The method according to Claim 33 wherein the pattern pieces corresponding to the same pattern are non-overlapping with each other.

35. The method according to Claim 34 wherein the Bloom filter circuit comprises a plurality of parallel Bloom filter circuits, each of the parallel Bloom filter circuits being programmed with pattern pieces of the same length such that each parallel Bloom filter circuit corresponds to a different pattern piece length than the other parallel Bloom filter circuits, and wherein the querying step comprises simultaneously providing the parallel Bloom filter circuits with a plurality of the data substrings in parallel.

36. The method according to Claim 35 wherein the Bloom filter circuit further comprises a shift register for storing at least a portion of the data string, the method further comprising:
streaming the data symbols of the data string through the shift register; and
reading the data substrings for the querying step out of the shift register.

37. The method according to Claim 36 wherein the streaming step comprises streaming a plurality of new symbols of the data string into the shift register per cycle, and wherein the reading step comprises reading a plurality of different substrings for the querying step out of the shift register per cycle.

38. The method according to Claim 35 wherein each parallel Bloom filter circuit is configured to store a plurality of bit vectors, each bit vector comprising a plurality of bits, each bit having a value, and wherein the querying step further comprises:
applying each data substring to at least one hash function to generate at least one hash key;
retrieving a bit vector from a parallel Bloom filter circuit based on the at least one generated hash key;

selecting a plurality of bit positions in the retrieved bit vector based on the at least one generated hash key; and

determining whether a potential match exists between that data substring and a pattern piece based on the values of the bits at the selected bit positions.

39. The method according to Claim 38 wherein the step of applying each data substring to at least one bash function comprises applying each data substring to a plurality of hash functions to generate a plurality of hash keys, wherein the bit vector retrieving step comprises retrieving a bit vector from a parallel Bloom filter circuit based on a generated hash key, and wherein the bit position selecting step comprises selecting a plurality of bit positions in the retrieved bit vector based on a plurality of the other generated hash keys.

40. The method according to Claim 38 wherein the step of applying each data substring to at least one hash function comprises applying each data substring to a single hash function to generate a single hash key, wherein the bit vector retrieving step comprises retrieving a bit vector from a parallel Bloom filter circuit based on a portion of the generated hash key, and wherein the bit position selecting step comprises selecting a plurality of bit positions in the retrieved bit vector based on another portion of the generated hash key.

41. The method according to Claim 38 wherein the reduction stage is configured to (1) store a plurality of pattern piece identifiers, each pattern piece identifier being indexed by data produced as a result of the querying step, (2) store a plurality of pattern identifiers, each pattern identifier being indexed by a pattern piece identifier, and (3) store a plurality of pattern set pairs, each pattern set pair being indexed by a pattern identifier, each pattern set pair comprising (a) data representative of a pattern, and (b) data representative of an allowable error associated with that pattern, and wherein the potential match applying step comprises:

retrieving at least one of the stored pattern piece identifiers based on data produced as a result of the querying step;

retrieving at least one of the stored pattern identifiers based on each retrieved pattern piece identifier; and

retrieving at least one of the stored pattern set pairs based on each retrieved pattern identifier, the retrieved at least one pattern set pair defining at least a portion of a pattern set.

42. The method according to Claim 41 wherein the pattern piece identifier retrieving step comprises retrieving at least one of the stored pattern piece identifiers based on data corresponding to the at least one generated hash key.

43. The method according to Claim 38 wherein the reduction stage is configured to (1) store a plurality of pattern identifiers, each pattern identifier being indexed by a pattern piece, and (2) store a plurality of pattern set pairs, each pattern set pair being indexed by a pattern identifier, each pattern set pair comprising (a) data representative of a pattern, and (b) data representative of an allowable error associated with that pattern, and wherein the potential match applying step comprises:

retrieving at least one of the stored pattern identifiers based on the data substring that produced a potential match; and

retrieving at least one of the stored pattern set pairs based on each retrieved pattern identifier, the retrieved at least one pattern set pair defining at least a portion of a pattern set.

44. The method according to Claim 43 wherein the reduction stage comprises a plurality of data structures for storing the pattern identifiers, each data structure corresponding to pattern pieces of the same length such that each data structure also corresponds to a different pattern piece length than the other data structures.

45. The method according to Claim 44 wherein the data structures comprise a plurality of hash tables.

46. The method according to Claim 44 wherein the data structures comprise a plurality of balanced search trees.

47. The method according to Claim 29 wherein the applying step comprises eliminating any potential matches that are false positives.

48. The method according to Claim 27 wherein the data representative of an allowable error comprises a predetermined number of allowable errors.

49. The method according to Claim 25 wherein the reduction stage is configured to (1) store a plurality of pattern piece identifiers, each pattern piece identifier being indexed by data produced as a result of the querying step, (2) store a plurality of pattern identifiers, each pattern identifier being indexed by a pattern piece identifier, and (3) store a plurality of

pattern set pairs, each pattern set pair being indexed by a pattern identifier, each pattern set pair comprising (a) data representative of a pattern, and (b) data representative of an allowable error associated with that pattern, and wherein the potential match applying step comprises:

retrieving at least one of the stored pattern piece identifiers based on data produced as a result of the querying step;

retrieving at least one of the stored pattern identifiers based on each retrieved pattern piece identifier; and

retrieving at least one of the stored pattern set pairs based on each retrieved pattern identifier, the retrieved at least one pattern set pair defining at least a portion of a pattern set.

50. The method according to Claim 49 wherein the pattern piece identifier retrieving step comprises retrieving at least one of the stored pattern piece identifiers based on data corresponding to the at least one generated hash key.

51. The method according to Claim 25 wherein the reduction stage is configured to (1) store a plurality of pattern identifiers, each pattern identifier being indexed by a pattern piece, and (2) store a plurality of pattern set pairs, each pattern set pair being indexed by a pattern identifier, each pattern set pair comprising (a) data representative of a pattern, and (b) data representative of an allowable error associated with that pattern, and wherein the potential match applying step comprises:

retrieving at least one of the stored pattern identifiers based on the data substring that produced a potential match; and

retrieving at least one of the stored pattern set pairs based on each retrieved pattern identifier, the retrieved at least one pattern set pair defining at least a portion of a pattern set.

52. The method according to Claim 51 wherein the reduction stage comprises a plurality of data structures for storing the pattern identifiers, each data structure corresponding to pattern pieces of the same length such that each data structure also corresponds to a different pattern piece length than the other data structures.

53. The method according to Claim 52 wherein the data structures comprise a plurality of hash tables.

54. The method according to Claim 52 wherein the data structures comprise a plurality of balanced search trees.

55. A system for processing a plurality of data substrings within a data string for facilitating a determination as to whether the data string contains an approximate match to any of a plurality of patterns, the data string comprising a plurality of data symbols, the system comprising:

a filter circuit configured to be queried by the data substrings to detect a plurality of potential matches, each potential match representing a potential match between a data substring and a pattern piece, wherein each pattern of a plurality of the patterns comprises a plurality of corresponding pattern pieces, the filter circuit being programmed with the pattern pieces; and

a reduction stage in communication with the filter circuit, the reduction stage being configured to (1) process the detected potential matches, (2) determine a plurality of pattern sets in response to the processing, each pattern set corresponding to a potential match and comprising (a) data representative of a pattern which corresponds to the pattern piece which produced the corresponding potential match; and (b) data representative of an allowable error associated with that pattern, and (3) output the determined pattern sets to an approximate match engine for a determination as to whether at least a portion of the data string is an approximate match to any of the patterns within the determined pattern sets taking into consideration the allowable error data within the determined pattern sets.

56. The system according to Claim 55 wherein the filter circuit comprises a Bloom filter circuit.

57. The system according to Claim 56 further comprising:

an approximate match engine in communication with the reduction stage, the approximate match engine configured to determine whether any approximate matches exist between the data string portion and the patterns within the determined pattern sets taking into consideration the allowable error data within the determined pattern sets.

58. The system according to Claim 57 wherein the Bloom filter circuit and the reduction stage are configured to operate in a pipelined manner.

59. The system according to Claim 58 wherein the Bloom filter circuit and the reduction stage are implemented with reconfigurable hardware.

60. The system according to claim 59 wherein the approximate match engine is implemented with reconfigurable hardware.

61. The system according to Claim 59 wherein the reconfigurable hardware comprises at least one Field Programmable Gate Array ("FPGA").

62. The system according to Claim 58 wherein the Bloom filter circuit and the reduction stage are implemented with at least one integrated circuit.

63. The system according to Claim 62 wherein the at least one integrated circuit comprises at least one Application Specific Integrated Circuit ("AS1C").

64. The system according to Claim 58 wherein the pattern pieces comprise a plurality of pattern pieces of different lengths.

65. The system according to Claim 64 wherein the pattern pieces corresponding to the same pattern are non-overlapping with each other.

66. The system according to Claim 65 wherein the Bloom filter circuit comprises a plurality of parallel Bloom filter circuits, each of the parallel Bloom filter circuits being programmed with pattern pieces of the same length such that each parallel Bloom filter circuit corresponds to a different pattern piece length than the other parallel Bloom filter circuits, and wherein the Bloom filter circuit is further configured to simultaneously provide the parallel Bloom filter circuits with a plurality of the data substrings in parallel.

67. The system according to Claim 66 wherein the Bloom filter circuit further comprises a shift register for storing at least a portion of the data string, and wherein the Bloom filter circuit is further configured to (1) stream the data symbols of the data string through the shift register and (2) read the data substrings out of the shift register for delivery to the parallel Bloom filter circuits.

68. The system according to Claim 67 wherein the Bloom filter circuit is further configured to (1) stream a plurality of new symbols of the data string into the shift register

per cycle, and (2) read a plurality of different substrings out of the shift register per cycle for delivery to the parallel Bloom filter circuits.

69. The system according to Claim 66 wherein each parallel Bloom filter circuit is configured to store a plurality of bit vectors, each bit vector comprising a plurality of bits, each bit having a value, and wherein the Bloom filter circuit is further configured to (1) apply each data substring to at least one hash function to generate at least one hash key, (2) retrieve a bit vector from a parallel Bloom filter circuit based on the at least one generated hash key, (3) select a plurality of bit positions in the retrieved bit vector based on the at least one generated hash key, and (4) determine whether a potential match exists between that data substring and a pattern piece based on the values of the bits at the selected bit positions.

70. The system according to Claim 69 wherein each parallel Bloom filter circuit is further configured to (1) apply each data substring to a plurality of hash functions to generate a plurality of hash keys, (2) retrieve a bit vector from a parallel Bloom filter circuit based on a generated hash key, and (3) select a plurality of bit positions in the retrieved bit vector based on a plurality of the other generated hash keys.

71. The system according to Claim 69 wherein each parallel Bloom filter circuit is further configured to (1) apply each data substring to a single hash function to generate a single hash key, (2) retrieve a bit vector from a parallel Bloom filter circuit based on a portion of the generated hash key, and (3) select a plurality of bit positions in the retrieved bit vector based on another portion of the generated hash key.

72. The system according to Claim 69 wherein the reduction stage comprises a first table, a second table, and a third table, wherein the first table is configured to store a plurality of pattern piece identifiers, each pattern piece identifier being indexed by data produced by the Bloom filter circuit, wherein the second table is configured to store a plurality of pattern identifiers, each pattern identifier being indexed by a pattern piece identifier, and wherein the third table is further configured to store a plurality of pattern set pairs, each pattern set pair being indexed by a pattern identifier, each pattern set pair comprising (a) data representative of a pattern, and (b) data representative of an allowable error associated with that pattern, and wherein the reduction stage is further configured to (1) retrieve at least one of the stored pattern piece identifiers from the first table based on data produced by the Bloom filter circuit, (2) retrieve at least one of the stored pattern identifiers from the second table based on

each retrieved pattern piece identifier, and (3) retrieve at least one of the stored pattern set pairs from the third table based on each retrieved pattern identifier, the at least one retrieved pattern set pair defining at least a portion of a pattern set.

73. The system according to Claim 72 wherein the reduction stage is further configured to retrieve at least one of the stored pattern piece identifiers from the first table based on data corresponding to the at least one generated hash key.

74. The system according to Claim 69 wherein the reduction stage comprises a first data structure and a second data structure, wherein the first data structure is configured to store a plurality of pattern identifiers, each pattern identifier being indexed by a pattern piece, and wherein the second data structure is configured to store a plurality of pattern set pairs, each pattern set pair being indexed by a pattern identifier, each pattern set pair comprising (a) data representative of a pattern, and (b) data representative of an allowable error associated with that pattern, and wherein the reduction stage is further configured to (1) retrieve at least one of the stored pattern identifiers from the first data structure based on the data substring that produced a potential match, and (2) retrieve at least one of the stored pattern set pairs from the second data structure based on each retrieved pattern identifier, the at least one retrieved pattern set pair defining at least a portion of a pattern set.

75. The system according to Claim 74 wherein the first data structure comprises a plurality of the first data structures, each first data structure corresponding to pattern pieces of a different length such that each first data structure also corresponds to a different pattern piece length than the other first data structures.

76. The system according to Claim 75 wherein the first and second data structures comprise a plurality of hash tables.

77. The system according to Claim 75 wherein the first and second data structures comprise a plurality of balanced search trees.

78. The system according to Claim 58 wherein the reduction stage is further configured to eliminate any potential matches that are false positives.

79. The system according to Claim 57 wherein the data representative of an allowable error comprises a predetermined number of allowable errors.

80. The system according to Claim 55 wherein the reduction stage comprises a first table, a second table, and a third table, wherein the first table is configured to store a plurality of pattern piece identifiers, each pattern piece identifier being indexed by data produced by the filter circuit, wherein the second table is configured to store a plurality of pattern identifiers, each pattern identifier being indexed by a pattern piece identifier, and wherein the third table is further configured to store a plurality of pattern set pairs, each pattern set pair being indexed by a pattern identifier, each pattern set pair comprising (a) data representative of a pattern, and (b) data representative of an allowable error associated with that pattern, and wherein the reduction stage is further configured to (1) retrieve at least one of the stored pattern piece identifiers from the first table based on data produced by the Bloom filter circuit, (2) retrieve at least one of the stored pattern identifiers from the second table based on each retrieved pattern piece identifier, and (3) retrieve at least one of the stored pattern set pairs from the third table based on each retrieved pattern identifier, the at least one retrieved pattern set pair defining at least a portion of a pattern set.

81. The system according to Claim 80 wherein the reduction stage is further configured to retrieve at least one of the stored pattern piece identifiers from the first table based on data corresponding to the at least one generated hash key.

82. The system according to Claim 56 wherein the reduction stage comprises a first data structure and a second data structure, wherein the first data structure is configured to store a plurality of pattern identifiers, each pattern identifier being indexed by a pattern piece, and wherein the second data structure is configured to store a plurality of pattern set pairs, each pattern set pair being indexed by a pattern identifier, each pattern set pair comprising (a) data representative of a pattern, and (b) data representative of an allowable error associated with that pattern, and wherein the reduction stage is further configured to (1) retrieve at least one of the stored pattern identifiers from the first data structure based on the data substring that produced a potential match, and (2) retrieve at least one of the stored pattern set pairs from the second data structure based on each retrieved pattern identifier, the at least one retrieved pattern set pair defining at least a portion of a pattern set.

83. The system according to Claim 82 wherein the first data structure comprises a plurality of the first data structures, each first data structure c.orrespnnding to pattern pieces of a different length such that each first data structure also corresponds to a different pattern piece length than the other first data structures.

84. The system according to Claim 83 wherein the first and second data structures comprise a plurality of hash tables.

85. The system according to Claim 83 wherein the first and second data structures comprise a plurality of balanced search trees.

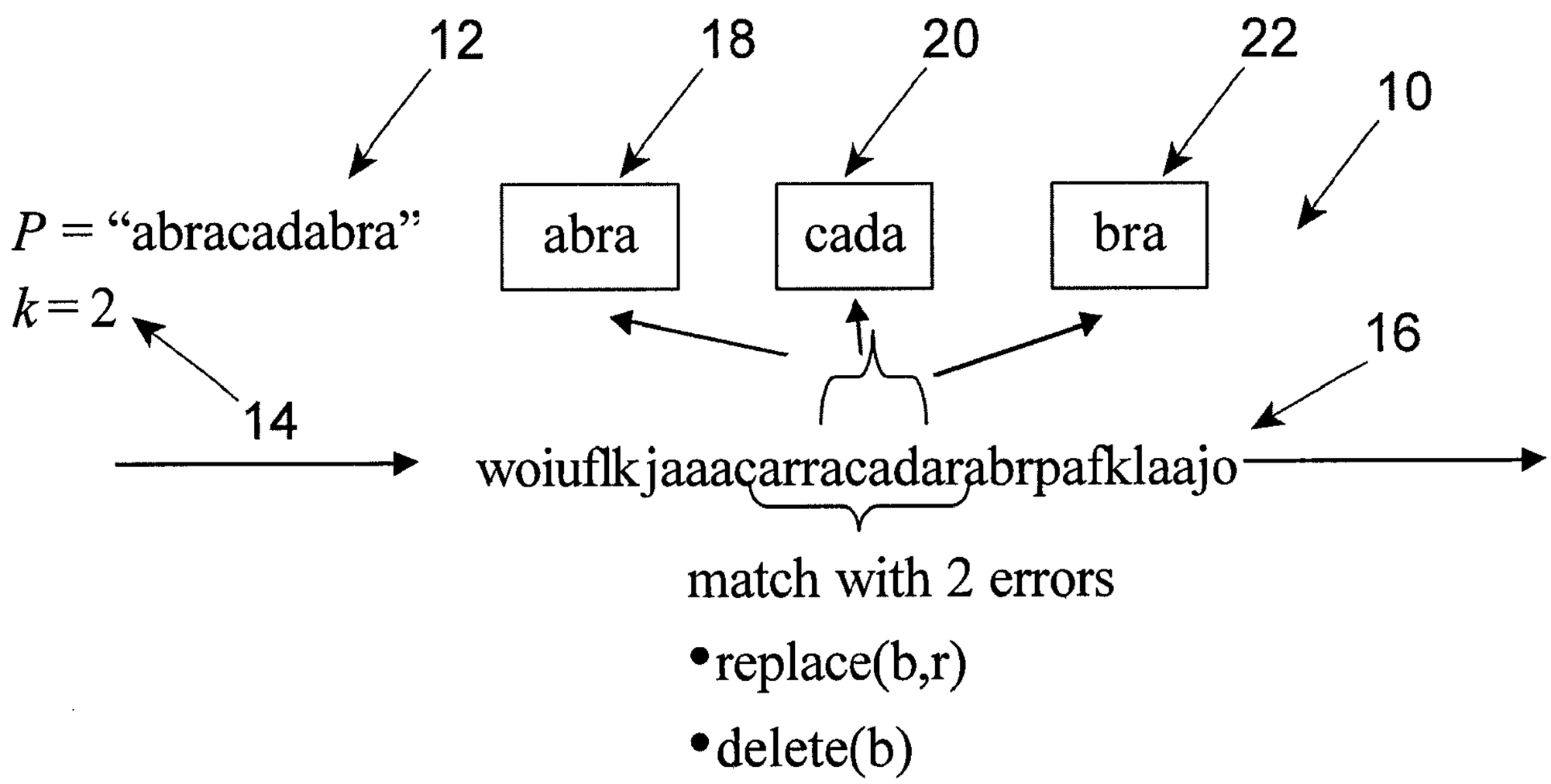


FIG. 1
Prior Art

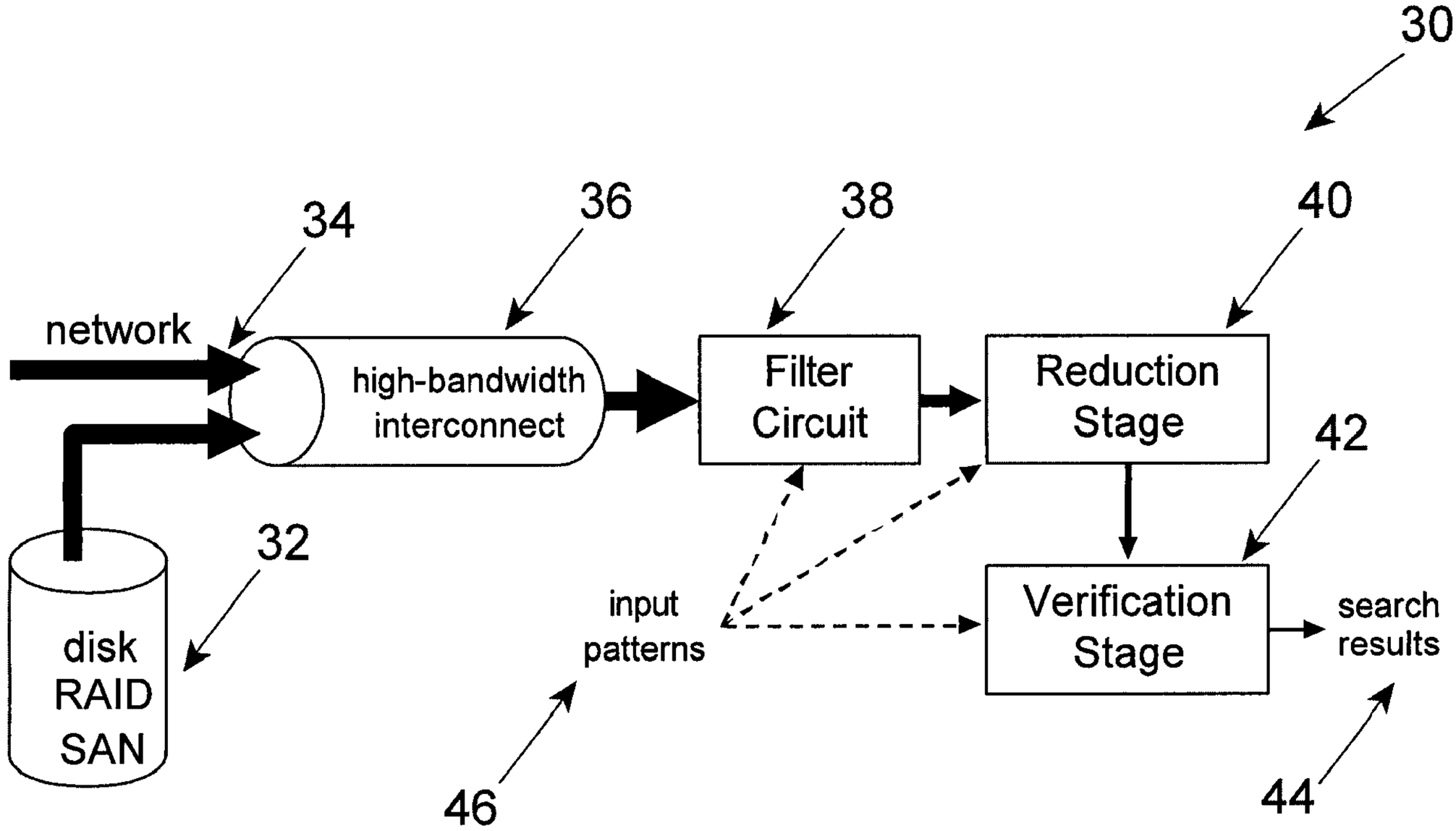

30
34
36
38
40
network
high-bandwidth interconnect
Filter Circuit
Reduction Stage
42
32
disk RAID SAN
input patterns
Verification Stage
search results
46
44

FIG. 2

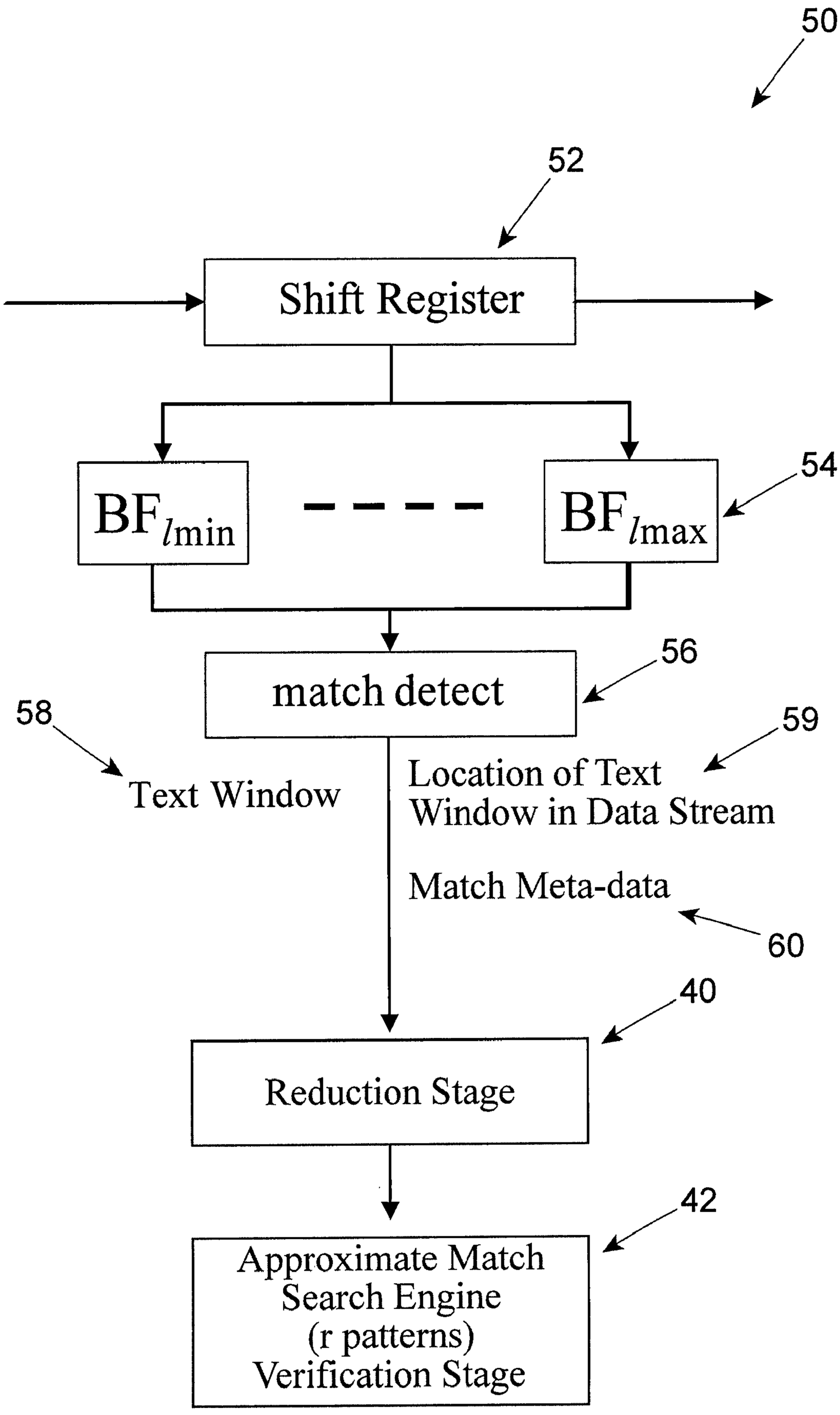

50
52
Shift Register
BF_{lmin}
BF_{lmax}
54
56
match detect
58
Text Window
59
Location of Text Window in Data Stream
Match Meta-data
60
40
Reduction Stage
42
Approximate Match Search Engine (r patterns) Verification Stage

FIG. 3

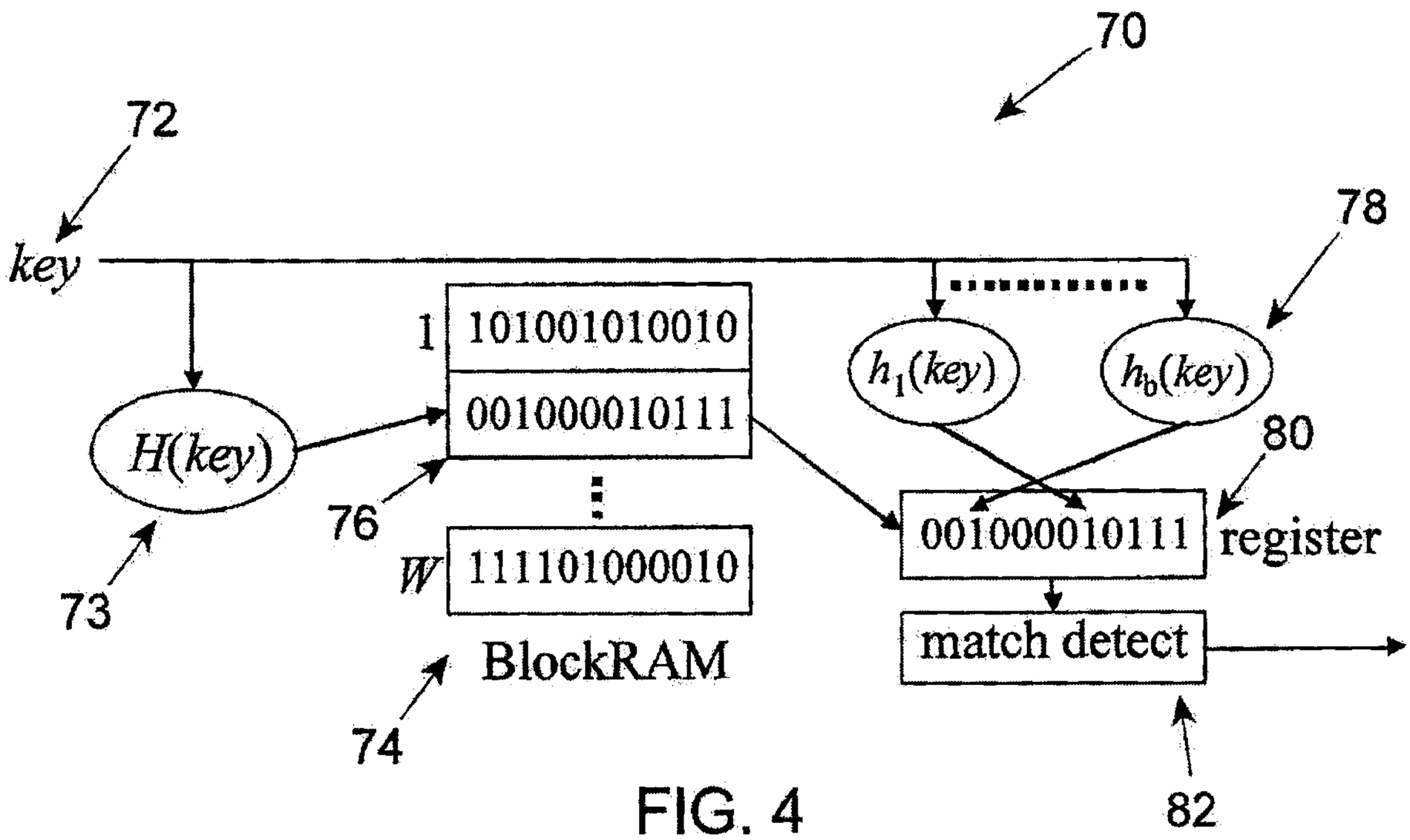

70
72
78
key
1
101001010010
001000010111
$h_1(key)$
$h_b(key)$
$H(key)$
80
76
001000010111
register
73
W
111101000010
BlockRAM
match detect
74
82

FIG. 4

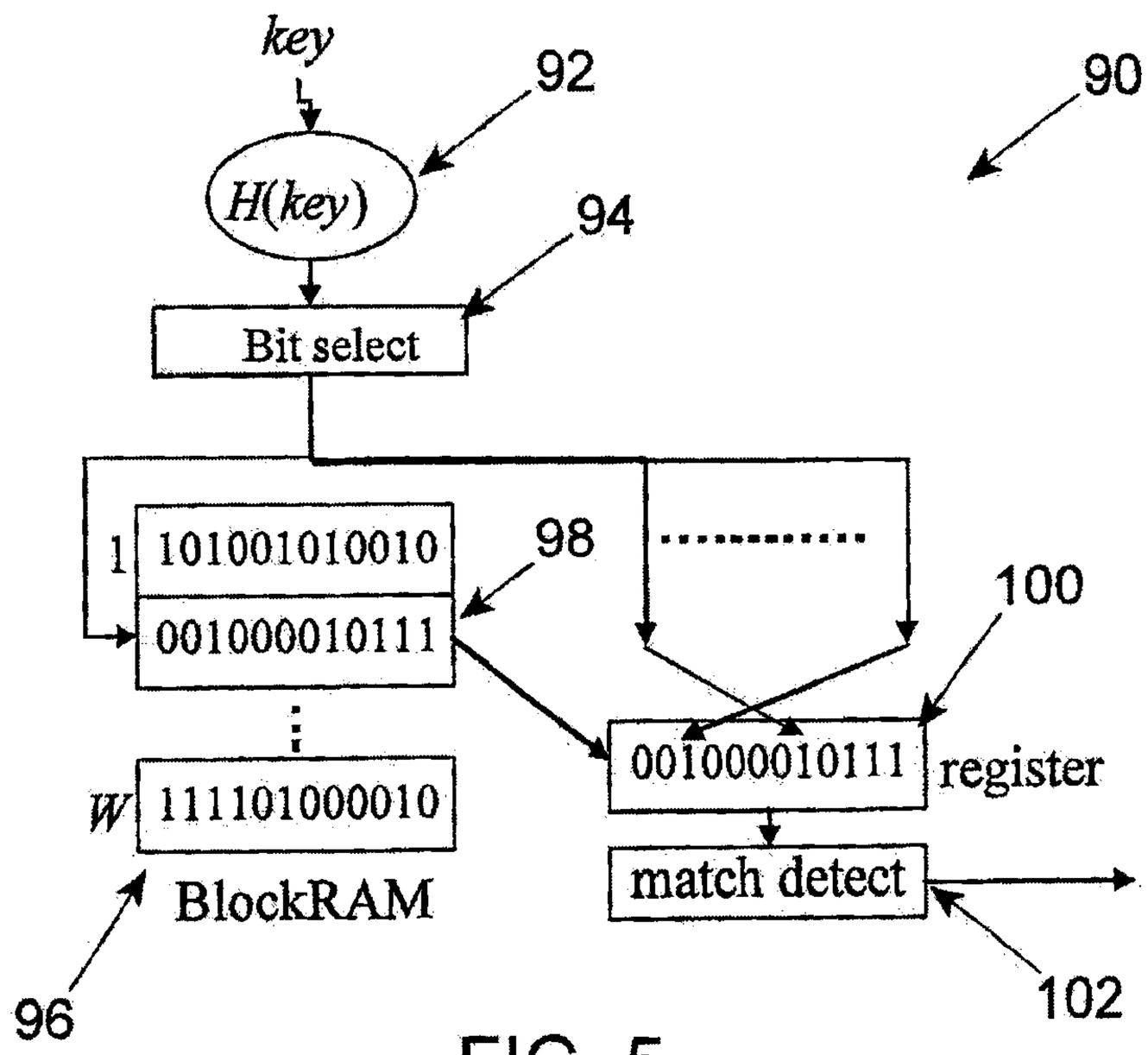

key
92
90
$H(key)$
94
Bit select
1
101001010010
98
100
001000010111
001000010111
register
W
111101000010
BlockRAM
match detect
96
102

FIG. 5

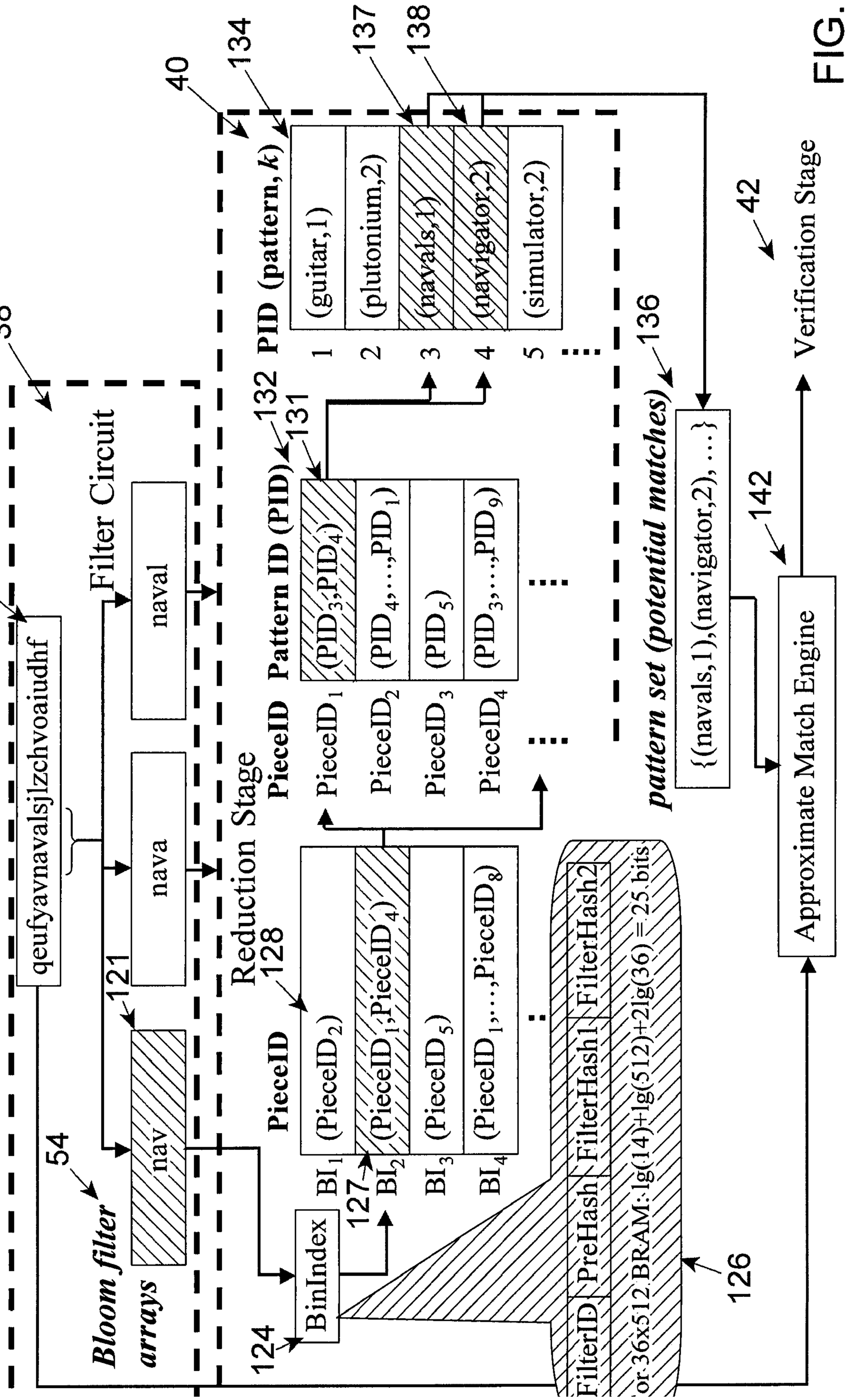

40
134
137
138
120
38
52
qeufyavnavalsjlzchvoaiudhf
Filter Circuit
naval
nava
nav
121
Reduction Stage
128
PieceID Pattern ID (PID)
PieceID
PieceID1
PieceID2
PieceID3
PieceID4
(PieceID3,PieceID4)
(PieceID4,...,PieceID1)
(PieceID5)
(PieceID3,...,PieceID9)
(PieceID2)
(PieceID1,PieceID4)
(PieceID5)
(PieceID1,...,PieceID8)
132
131
PID (pattern, k)
1 (guitar,1)
2 (plutonium,2)
3 (navals,1)
4 (navigator,2)
5 (simulator,2)
136
pattern set (potential matches)
{(navals,1),(navigator,2), ...}
142
42
Verification Stage
Approximate Match Engine
54
Bloom filter arrays
124
BinIndex
127
BI1
BI2
BI3
BI4
FilterID
PreHash
FilterHash1
FilterHash2
for 36x512 BRAM: lg(14)+lg(512)+2lg(36) = 25 bits
126

FIG. 6

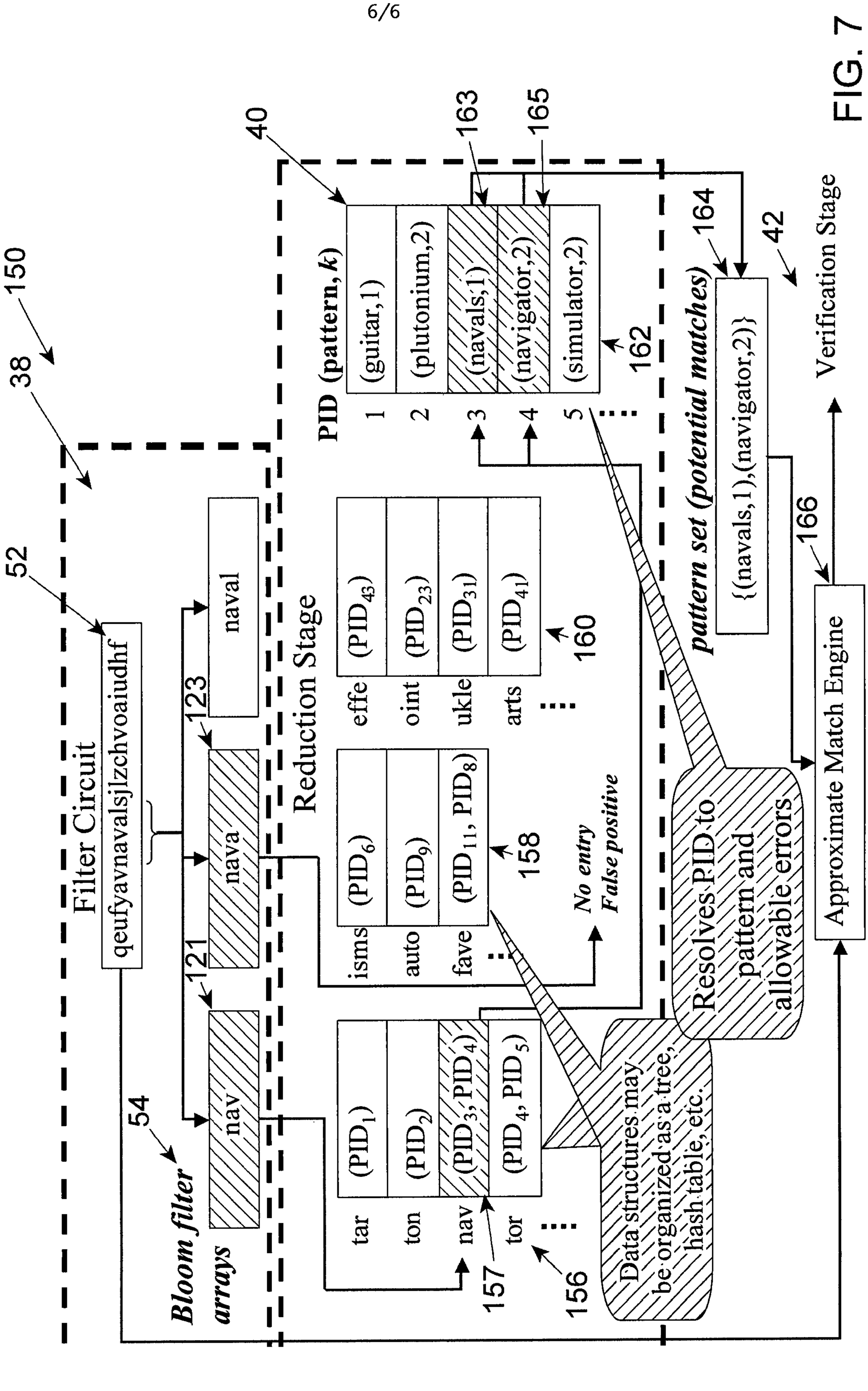
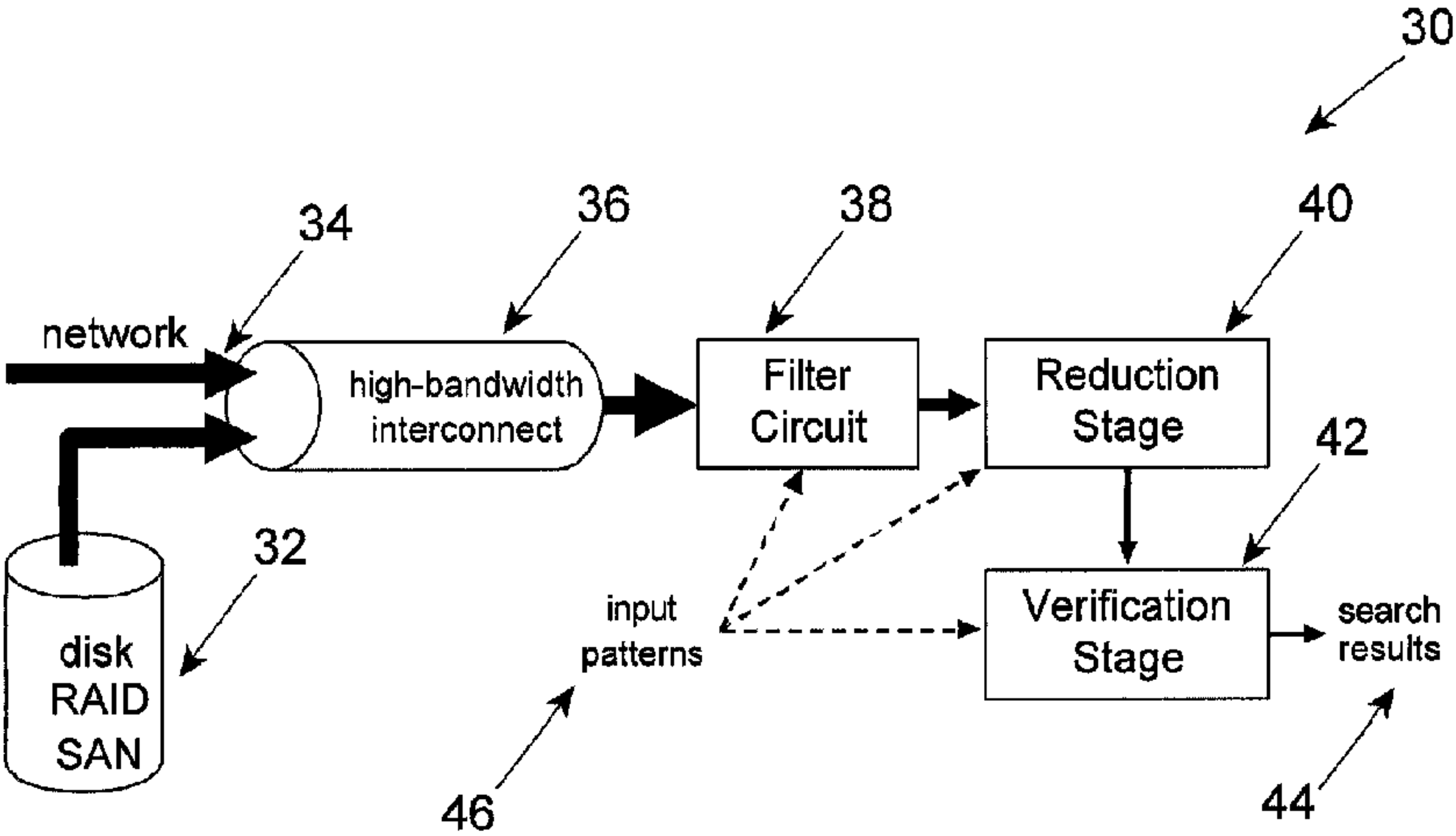


FIG. 7
40
163
165
150
38
52
PID (pattern, k)
1 (guitar,1)
2 (plutonium,2)
3 (navals,1)
4 (navigator,2)
5 (simulator,2)
162
164
42
Verification Stage
pattern set (potential matches)
{(navals,1),(navigator,2)}
166
Approximate Match Engine
Filter Circuit
qeufyavnavalsjlzchvoaiudhf
121
nava
123
naval
Reduction Stage
isms (PID6)
auto (PID9)
fave (PID11, PID8)
158
No entry
False positive
effe (PID43)
oint (PID23)
ukle (PID31)
arts (PID41)
160
Resolves PID to pattern and allowable errors
54
Bloom filter arrays
nav
tar (PID1)
ton (PID2)
nav (PID3, PID4)
tor (PID4, PID5)
157
156
Data structures may be organized as a tree, hash table, etc.


30
34
36
38
40
network
high-bandwidth interconnect
Filter Circuit
Reduction Stage
42
32
disk RAID SAN
input patterns
Verification Stage
search results
46
44