(19) DANMARK

(10) **DK/EP 4030702 T3**

Patent- og Varemærkestyrelsen

(12) Oversættelse af europæisk patentskrift

(51) Int.Cl.: ***H 04 L 45/00 (2022.01)*** ***H 04 L 12/42 (2006.01)*** ***H 04 L 45/28 (2022.01)*** ***H 04 L 45/745 (2022.01)*** ***H 04 L 45/7453 (2022.01)*** ***H 04 L 45/748 (2022.01)***

(45) Oversættelsen bekendtgjort den: **2025-03-17**

(80) Dato for Den Europæiske Patentmyndigheds bekendtgørelse om meddelelse af patentet: **2025-02-26**

(86) Europæisk ansøgning nr.: **21152734.6**

(86) Europæisk indleveringsdag: **2021-01-21**

(87) Den europæiske ansøgnings publiceringsdag: **2022-07-20**

(30) Prioritet: **2021-01-19 US 202117151882**

(84) Designerede stater: **AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR**

(73) Patenthaver: **Drivenets Ltd., 4 HaSheizaf Street, 4366411 Raanana, Israel**

(72) Opfinder: **KRAYDEN, Amir, , 4647000 Hertzelia, Israel**
LEV, Yuval, , 7087015 Gan Yavne, Israel
SANDLER, Evgeny, , 4680439 Herzliya, Israel
ZILBERMAN, Alexander, , 3824055 Hadera, Israel

(74) Fuldmægtig i Danmark: **Zacco Denmark A/S, Arne Jacobsens Allé 15, 2300 København S, Danmark**

(54) Benævnelse: **FREMGANGSMÅDE TIL AT IMPLEMENTERE EN KONSISTENT HASHING I ET KOMMUNIKATIONSNETVÆRK**

(56) Fremdragne publikationer:
EP-A1- 1 876 788
EP-B1- 1 876 788
WO-A1-2008/003596
US-A1- 2013 268 644
US-A1- 2018 270 153
US-B1- 9 979 648
MAENPAA ERICSSON A SWAMINATHAN S DAS QUALCOMM J ET AL: "A Topology Plug-in for REsource LOcation And Discovery; draft-maenpaa-p2psip-topologyplugin-00.txt", A TOPOLOGY PLUG-IN FOR RESOURCE LOCATION AND DISCOVERY; DRAFT-MAENPAA-P2PSIP-TOPOLOGYPLUGIN-00.TXT, INTERNET ENGINEERING TASK FORCE, IETF; STANDARDWORKINGDRAFT, INTERNET SOCIETY (ISOC) 4, RUE DES FALAISES CH- 1205 GENEVA, SWITZERLAND, 6 July 2009 (2009-07-06), XP015063125
ION STOICA ET AL: "Chord", COMPUTER COMMUNICATION REVIEW, ACM, NEW YORK, NY, US, vol. 31, no. 4, 27 August 2001 (2001-08-27), pages 149 - 160, XP058178055, ISSN: 0146-4833, DOI: 10.1145/964723.383071

DESCRIPTION

Description

TECHNICAL FIELD

[0001] The present disclosure generally relates to the field of communication systems. More particularly, the present disclosure relates to systems implementing infrastructure for multiservice routing, virtual network function virtualization and software-defined networking.

GLOSSARY

[0002]

ECMP - Equal Cost Multi-Path.

EPC - Evolved Packet Core.

FRR - Fast Re-Route.

LPM - Longest Prefix Match.

NAT - Network Address Translation.

NCP - Network Cloud Packet Forwarder.

NPU - Network Processing Unit.

TCAM - Ternary Content-Addressable Memory.

WCMP - Weighted Cost Multi-Path.

VRF - Virtual Routing and Forwarding.

BACKGROUND

[0003] High-scale stateful load balancing is an important part of many network functions and an active field of academic and practical research. The use of load balancing is part of many different layers in networking functionalities, including but not limited to: Link-aggregation traffic

distribution, ECMP / WCMP (Equal Cost Multi-Path, Weighted Cost Multi-Path), L4-L7 Session Load Balancing, Distributed Data-base implementation, Mobility EPC (Evolved Packet Core) functions, NAT (Network Address Translations) functions, and the like.

[0004] Classical non stateful (Link-aggregation, ECMP / ECMP) or stateful (L4-L7 Session Load Balancing, etc.) are all revolving around the way to choose the destination or eventual server / micro-service which should handle a traffic item. The way in which the existing destinations are obtained, managed, accounted for or being classified are beyond the scope of this disclosure, so are the exact ways by which load parameters / abilities of these destinations are conveyed to the load balancing device / devices (which can be based on out-of-band communication or in-band communication using special streams or piggi-bagging existing ones, where problems are more complicated in case of a distributed load-balancing architecture).

[0005] J. Maenpaa et al. describe in their paper "A Topology Plug-in for Resource Location And Discovery, draft-maenpaa-p2psip-topologyplugin-00", a peer-to-peer signaling protocol (RELOAD) that can be used to maintain an overlay network, and to store data in and retrieve data from the overlay. The document defines a new topology plug-in for RELOAD that is more appropriate for real world large scale overlays.

[0006] US 2018/270153 relates to entropy in routing tables that may be increased to perform packet forwarding. Hash tables that store forwarding routes may be divided into multiple hash table segments and forwarding routes may be stored across the hash table segments in different route segments.

SUMMARY

[0007] The invention is defined by referring to the appended claims.

[0008] It is an object of the present disclosure to provide a novel and improved method that allows affecting changes in a communication network comprising a plurality of network processing units and/or clusters thereof, while adding/removing/overcoming a failure of a network processing unit.

[0009] It is another object to provide a novel method that enables obtaining an improved load balancing performance.

[0010] It is another object to provide a novel method that enables obtaining an improved scaling up of existing networks that comprise a vast number of network processing units.

[0011] It is yet another object to provide a novel and improved method for utilizing different load-balancing groups in conveying communications in a communication network.

[0012] Other objects will become apparent from the following description.

[0013] According to a first example of the there is provided a method for use in a communication network, wherein the method comprises the steps of:

1. (i) providing a plurality of network processing units (NPU's) comprised in the communication network;
2. (ii) establishing a replication of at least one of that plurality of network processing units;
3. (iii) arranging the plurality of network processing units and the at least one replication, in a virtual ring configuration;
4. (iv) associating a unique primary virtual identification and a corresponding unique backup virtual identification with each active and available network processing unit and the at least one replication;
5. (v) establishing a list of hash values, associated with the primary virtual identifications of said active and available network processing units and hash values associated with the backup virtual identification of corresponding active and available network processing units;
6. (vi) implementing a ring consistent hashing algorithm for carrying out a search resolution for a consistent hashing (e.g., by using an arbitrary key K); and
7. (vii) in a case of a change (i.e., adding/removing/failure) in at least one of the active and available entities having a respective unique primary virtual identification, using the corresponding backup virtual identification to maintain the ring continuity.

[0014] By yet another example, the search resolution for a consistent hashing is conducted based on searching for a packet destination ID using a key built from said packet's meta-data. Optionally, the packet's meta-data is derived from the packet's networking headers.

[0015] According to still another example, the search resolution for a consistent hashing is done by retrieving a respective destination ID while using a Ternary Content-Addressable Memory (TCAM) algorithm.

[0016] In accordance with another embodiment, the method provided further comprises a step of composing a route space of the networking subnets and/or prefixes, and wherein the search function is based on a Longest Prefix Match (LPM) algorithm.

[0017] According to yet another embodiment, the same search function (LPM) is preserved, and the space is built by using possible virtual identifications as prefixes.

[0018] By still another embodiment of the present disclosure, the method provided further comprises a step of utilizing different load-balancing groups as VRF (Virtual Routing and Forwarding) tables, which in turn are used as a parameter while carrying out the LPM algorithm search, thereby generating a dedicated consistent hashing space for use as a route space for conveying traffic in the communication network.

BRIEF DESCRIPTION OF THE DRAWINGS

[0019] The accompanying drawing, which is incorporated herein and constitute a part of this specification, illustrates an embodiment of the disclosure and, together with the description, serve to explain the principles of the embodiments disclosed herein.

FIG. 1 exemplifies a ring virtual configuration of software consistent hashing space, construed in accordance with an embodiment of the present invention;

FIG. 2 exemplifies a search resolution for a software consistent hashing using an arbitrary key, for the network exemplified in FIG.1;

FIG. 3 exemplifies a search resolution for a software consistent hashing using an arbitrary key after a node failure;

FIG. 4 exemplifies a ring virtual configuration of software consistent hashing space for a network comprising NPUs;

FIG. 5 exemplifies a search resolution for a software consistent hashing using an arbitrary key, for the NPUs network exemplified in FIG.4;

FIG. 6 exemplifies a search resolution for a software consistent hashing using an arbitrary key after an NPU node failure;

FIG. 7 illustrates a ring virtual configuration of software consistent hashing space of load-balancing groups, construed in accordance with an embodiment of the present invention; and

FIG. 8 exemplifies a search resolution for a software consistent hashing after packet classification utilizing different load-balancing groups as VRF (Virtual Routing and Forwarding) tables while carrying out a LPM algorithm search.

DESCRIPTION OF EXEMPLARY EMBODIMENTS

[0020] Some of the specific details and values in the following detailed description refer to certain examples of the disclosure. However, this description is provided only by way of example and is not intended to limit the scope of the invention in any way. As will be appreciated by those skilled in the art, the claimed distributed routing system may be implemented by using other devices that are known in the art *per se.* The scope of the invention can be summarized by referring to the appended claims.

[0021] With the rise of distributed architectures, consistent hashing became a major player.

The term "hashing" refers to a process to map data of arbitrary size to fixed-size values. Hashing has many applications in computer science, such as for example, checksum. In order to verify the integrity of a dataset, a server hashes a dataset and indicates the hash value to a client. Then, the client hashes its version of the dataset and compares the hash values. If the two are equal, the integrity is is likely to be verified (e.g., a problem may arise when two distinct pieces of data have the same hash value (a collision)).

[0022] One of the challenges associated with this methodology is spreading the values across a domain. Load distribution is therefore the process of spreading load across network processing units.

[0023] The term "network processing unit" as used herein throughout the specification and claims is used to denote any number of the group that comprises a node, a server, a computation unit, and the like. Load balancing is one example of load distribution. It is used in distributing a set of tasks over a set of resources. For example, load balancing may be used to distribute API requests among web server instances.

[0024] However, when dealing with data, we the term sharding may be used. A database shard is a horizontal partition of data in a database. A typical example is a database partitioned in three shards where each shard has a subset out of the total data.

[0025] Load balancing and sharding share some common challenges. Spreading data evenly is used for example to guarantee that a node is not overloaded compared to the others. In some cases, load balancing and sharding also need to associate tasks or data to the same node.

[0026] The principle of mod-n hashing is the following. The load needs to be distributed among the nodes based on key identifiers. Each key is hashed using a hashing function to transform an input into an integer. Then, a modulo based on the number of nodes is performed. If sharding is implemented and data is distributed based on the mod-n strategy, in a case where the number of nodes should be scaled, one would need to perform a rebalancing operation.

[0027] Now, if vast numbers (millions or even billions) of data have been stored and it is now required to rebalance a substantial part thereof, the rebalancing should preferably be a process wherein: the distribution remain as uniform as possible based on the new number of nodes, and the number of keys that need to be migrated should be limited. This is the purpose of consistent hashing algorithms, algorithms that maintain the key consistent until a certain point in order to keep the distribution uniform.

[0028] A ring consistent hashing algorithm is an algorithm that is based on a ring (an end-to-end connected array), and the first operation is to create the ring. A ring has a fixed-length and it may be partitioned into a predetermined number of partitions, and the network processing units (nodes in this example) are located in these partitions.

[0029] In general, a load-balancing device requires a function F(K), which is configured to match a key (e.g., network headers, data-base keys, and the like) with a node ID.

[0030] Let us define function F(K) as:

$$F(K): \{K\} \rightarrow \{N\}$$

in which {K} is a set of available keys, and {N} is the set of active and available nodes.

[0031] Most systems use a naive but very scalable implementation, whereby a hash function (H) is used:

$$F(K) = H(K) \% N$$

[0032] In the above equation H is a Hash function which translates any arbitrary key into a number, while N is the number of current active and available nodes. Now, let us assume that H in reality is configured to operate on a known key set, and has a bounded run-time complexity, such that F(K) is considered O(1).

[0033] Next, the problem is what happens when N (the number of available and active network processing units, e.g., nodes) changes due to an addition, removal or failure of a network element. In such a case, the problem that arises is that the entire mapping logic which is defined by F, changes. Consequently, a complete traffic shuffling to all destinations occurs, regardless of the fact whether the actual item was originally destined to a changed node (resulting, in many changes within the system, large blast radius and a need to move content between places).

[0034] Mathematically the result stems from the use of the modulo operator (%). When the number of nodes N changes to N',

$$F(K) != F'(K) \ [(F(K) = H(K)\%N),\ F'(K) = H(K) \% N']$$

for every H(K)>=N'.

[0035] To mitigate the problem without increasing memory footprint, yet while taking into account different node processing capabilities (in terms of memory, CPU, networking bandwidth and delay, etc.), one may use the function M(K), by implementing the following steps:

1. 1. Creating a virtual node ID for every active and available node, wherein the virtual node ID is defined by:
   V_{ID} = H($NODE\__{ID}$, R), in which $NODE\__{ID}$ is the physical node ID, and R is the replica number which creates a weight of the number of times that this node should be used in calculations, thereby enabling the use of unequal hashing;
2. 2. Creating an ordered list L of the V_{ID}'s (preferably their hash values)
3. 3. When a packet / key needs to be resolved, retrieving function F(K) by taking the following steps:

1. 1. Calculating the hash value for the key $N_{ID} = H(K)$
2. 2. Determining the minimal element in L which is larger or equal to N_{ID}:

   `Actual Node ID = Search(L, N_ID, [V_ID >= N_ID])`

[0036] The above method can be implemented effectively using a software wherein $O(\log_2 N)$ run-time complexity and O(N) space complexity, using a binary search or tree structures.

[0037] This implementation may be regarded as a geometrical representation of a circle, in which points comprised in the circle are the various V_{ID}'s. It is no longer sensitive to a change in N, since F(K) does not use the modulo operation. A node addition, removal or failure would mark the relevant V_{ID}'s as inactive and the search will return the next available item (in a clockwise direction) resulting in a different resolution for items destined only to it - which simplifies the change and consequently, results on average in only [K/N] items being changed.

[0038] FIG. 1 demonstrates an example of the above methodology which depicts a software consistent hashing space creation, construed in accordance with an embodiment of the present invention. The representation depicts four physical nodes designated A, B, C and D and replicas of nodes A and D. This configuration is then used for creating a consistent hashing space. For every active and available node of the six entities depicted in this representation, 1 to 6, a virtual node ID was created (V_{ID}), and an ordered list L of the hash values of the V_{ID}'s, established.

[0039] FIG. 2 illustrates the network presented in FIG.1, wherein an example of a search resolution for a software consistent hashing is depicted, using an arbitrary key K.

[0040] FIG. 3 depicts an example of the search resolution for the software consistent hashing on an arbitrary key K shown in FIG. 2, after a failure of the node C,1. Obviously, as would be appreciated by those skilled in the art, such a process as the one in Figs 2 and 3, *mutates mutandis* may be carried out for each of the nodes which experiences a change (addition, removal, multiple failures).

[0041] In view of the above, the present invention seeks to provide a solution for additional cases which aim to effectively implement a scalable version of the algorithm using an NPU (Network Processing Unit) so that a single NCP or even clusters of NCPs working together, can operate accordingly. Solving these problems has a number of obvious advantages. To name but few, a better overall scaling (from servers which can handle tens of Gb/s to small 1hundreds of Gb/s, up to NCPs which can handle 4Tb's to many hundreds of Tb/s at a line-rate); a better overall space, and power efficiency; a better Search performance (O(1) hardware search); a better overall load-balancing performance (processing packets using hardware); and a better overall failure / change mitigation rate.

[0042] In order to reach a solution to the above challenge, the following two issues must be addressed. First, how to represent the consistent hashing space in an NPU, and second, how to implement the Search function in an NPU.

[0043] The solution found was implementing a packet forwarding based on searching for the destination ID (the next node) using a key built from the packet's meta-data (mainly the networking headers) . Searching is done for that key inside a fast search memory (physical or algorithmic TCAM [a Content Addressable Memory]), resulting in a respective destination ID.

[0044] The possible route space is composed of the networking subnets / prefixes (being de-facto an ordered list) while the search function is based on an LPM (Longest Prefix Match). Thus, one of the advantages of the present invention is that it provides a logical linkage between a search for the next closest consistent-hashing ring member and the operation of the Longest Prefix Matching (LPM).

[0045] However, in a consistent hashing case, as the case with the problem which this embodiment of the present invention seeks to solve, the solution proposed is to ensure that the same search function (LPM) is preserved, but the space is built differently, namely, by using the possible V_{ID}'s as prefixes.

[0046] Adopting this approach, allows implementing a general packet-processing mechanism to solve the problem, by achieving all the aforementioned issues without a need to employ any special hardware mechanism. The search of N_{ID} is done immediately by hardware at a line-rate for all possible node ID's.

[0047] In order to mitigate a failure, every V_{ID} entry results in a primary (i.e., the node itself) and backup V_{ID} entry. This is handled by using a hardware, similar to implementation of an FRR (Fast Re-Route) mechanisms. Removal or addition of nodes are handled as addition or removal (atomic operations) of nodes in a route. In other words, one may consider the mechanism described hereinabove as a non-trivial use of FRR (fast re-route) mechanism to implement the treatment of node in which a change has occurred. To do that, a primary destination pointer and backup destination pointer are used, and once the primary destination pointer goes down, the backup destination pointer is used to point (i.e., identify) to the next valid member of the ring.

[0048] In a way similar to FIG. 1, FIG. 4 demonstrates an example of a network representation of network comprising network processing units (NPUs), for a software consistent hashing space creation, construed in accordance with an embodiment of the present invention. The representation comprises four physical NPUs designated A, B, C and D and respective replicas of NPUs A and D, for creating a consistent hashing space. For every TCAM entry of an NPU/replica depicted in this representation, 1 to 6, a primary virtual node ID and a backup virtual node ID were created (V_{ID}) . An ordered list L of the hash values of the primary V_{ID}'s and the backup V_{ID}'s is shown in this figure.

[0049] FIG. 5 illustrates the network presented in FIG.1, wherein an example of a search resolution for an NPU assisted consistent hashing is depicted, using an arbitrary key K.

[0050] FIG. 6 depicts an example of the search resolution for the NPU assisted consistent hashing on an arbitrary key K present in FIG. 5, after a failure of the NPU C,1.

[0051] According to another embodiment of the present disclosure illustrated in FIG. 7, the method provided further comprises a step of implementing a plurality of different load-balancing groups as VRF (Virtual Routing and Forwarding) tables, which in turn are used as a parameter while carrying out the LPM algorithm search, thereby creating a dedicated consistent hashing space for use as a route space for conveying traffic in the communication network. The term "load-balancing group" as used herein throughout the specification and claims, is used to denote a set of specific destinations for a certain subset of the keys. Preferably, this is step is carried out by implementing the above-described technique (i.e., generating a consistent hashing ring structure) for each of these load-balancing groups.

[0052] FIG. 8 exemplifies a search resolution for a software consistent hashing. After affecting a packet classification, the different load-balancing groups are used as VRF (Virtual Routing and Forwarding) tables while carrying out an LPM algorithm search. The results of the LPM algorithm search are used in creating a dedicated consistent hashing space that in turn may be used as a route space for conveying traffic in the communication network.

[0053] One of the advantages of this aspect of the invention is that load-balancing groups are implemented in a software after determining what is the communication packet (i.e., packet classification), and then selecting the appropriate ring and searching thereat. If a packet destination is shared among more than one ring, in case of failure of the node which is the shared destination, all rings will have to be updated regarding the change that took place in order to overcome that node failure (which next hope was selected).

[0054] Using the novel NPU's network described hereinabove, the packet classification is carried out by using hardware (HW), and immediately thereafter, while still using the same memory (as the packet classification enables obtaining the respective VRF parameter), the appropriate entry is derived from the suitable VRF / load-balancing group. If a destination NPU that is shared by more than one ring fails, the HW mechanism will update every entry that includes that NPU with the NPU that is the backup NPU for the failing NPU.

[0055] The present invention has been described using detailed descriptions of embodiments thereof that are provided by way of example and are not intended to limit the scope of the invention in any way.

[0056] The scope of the invention is limited only by the following claims.

REFERENCES CITED IN THE DESCRIPTION

Cited references

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- US2018270153A [0006]

Patentkrav

1. Fremgangsmåde til brug i et kommunikationsnetværk, hvor fremgangsmåden omfatter trinene:

(i) tilvejebringelse af en flerhed af netværksbehandlingsenheder, NPU'er, (A-D) omfattet af kommunikationsnetværket;

og hvor fremgangsmåden er **kendetegnet ved, at** den omfatter yderligere følgende trin:

(ii) etablering af en replikation af mindst én af flerheden af netværksbehandlingsenheder (A,2 D,2);

(iii) anbringelse af flerheden af netværksbehandlingsenheder og den mindst ene replikation i en virtuel ringkonfiguration;

(iv) associering af en unik primær virtuel identifikation og en tilsvarende unik virtuel backup-identifikation med enhver aktiv og tilgængelig netværksbehandlingsenhed (H(A,1), H(B,1), H(C,1), H(D,1)) og den mindst ene replikation (H(A,2), H(D,2));

(v) etablering af en liste over hashværdier, der er forbundet med de primære virtuelle identifikationer af de aktive og tilgængelige netværksbehandlingsenheder, og hashværdier, der er forbundet med den virtuelle backup-identifikation af tilsvarende aktive og tilgængelige netværksbehandlingsenheder (H(A,2), (H(D,1), H(A,1), H(C,1), H(D,2), H(B,1));

(vi) implementering af en ringkonsistent hashing-algoritme til at udføre en søgeopløsning for en konsistent hashing (K, H(K)); og

(vii) i tilfælde af en ændring i mindst én af de aktive og tilgængelige netværksbehandlingsenheder med en vis unik primær virtuel identifikation (H(C,1)) (0,1), anvendelse af den tilsvarende virtuelle backup-identifikation (H(D,2)) for at opretholde ringekontinuiteten.

2. Fremgangsmåde ifølge krav 1, hvor søgeopløsningen for en konsistent hashing udføres baseret på søgning efter et pakkedestinations-ID under anvendelse af en nøgle bygget fra pakkens metadata.

3. Fremgangsmåde ifølge krav 2, hvor pakkens metadata er afledt fra pakkens netværksoverskrifter.

4. Fremgangsmåde ifølge krav 2, hvor søgeopløsningen for en konsistent hashing udføres ved at hente et respektive destinations-ID, mens der anvendes en ternær indholdsadresserbar hukommelses-, TCAM-, algoritme.

5. Fremgangsmåde ifølge krav 2, hvor den yderligere omfatter et trin med at sammensætte et ruterum af netværksundernettene og/eller præfikserne, og hvor en søgefunktion er baseret på en LPM-algoritme (Longest Prefix Match).

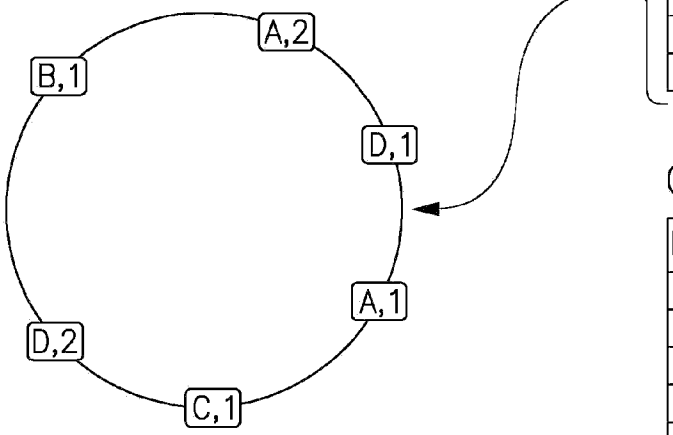
6. Fremgangsmåde ifølge krav 5, hvor den samme LPM-søgefunktion bevares, og ruterummet er bygget ved at anvende mulige virtuelle identifikationer som præfikser.

7. Fremgangsmåde ifølge krav 1, hvor søgeopløsningen for en konsistent hashing udføres baseret på søgning efter et pakkedestinations-ID ved hjælp af en nøgle bygget fra pakkens metadata,
hvor fremgangsmåden yderligere omfatter et trin med at sammensætte et ruterum af netværksundernet og/eller præfikser, og hvor en søgefunktion er baseret på en LPM-algoritme (Longest Prefix Match), og
hvor fremgangsmåden yderligere omfatter et trin med at bruge forskellige belastningsbalancerende grupper som Virtual Routing and Forwarding-, VRF-, tabeller, som igen bruges som parametre under udførelse af LPM-algoritmesøgningerne, og derved muliggør generering af et dedikeret konsistent ha-

shing-rum til brug som et ruterum til at formidle trafik i kommunikationsnetværket.

DRAWINGS

Drawing



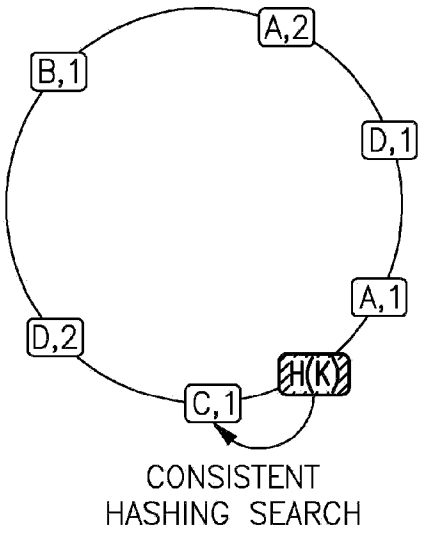
PHYSICAL NODES:

PHYSICAL NODES ID	REPLICAS
A	2
B	1
C	1
D	2

ORDERED V_{ID}'S LIST L:

PHYSICAL NODES ID	V_{ID}
1	H(A,2)
2	H(D,1)
3	H(A,1)
4	H(C,1)
5	H(D,2)
6	H(B,1)

FIG.1



RESOLVED NODES:

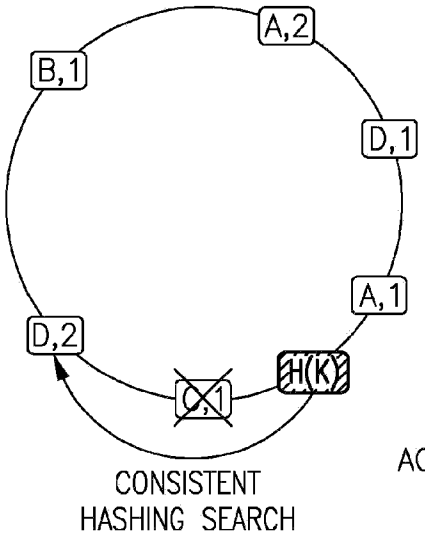
KEY	N_{ID}
K	H(K)

ORDERED V_{ID}'S LIST L:

KEY	N_{ID}
1	H(A,2)
2	H(D,1)
3	H(A,1)
4	H(C,1)
5	H(D,2)
6	H(B,1)

ACTUAL NODE ID

FIG.2



RESOLVED NODES:

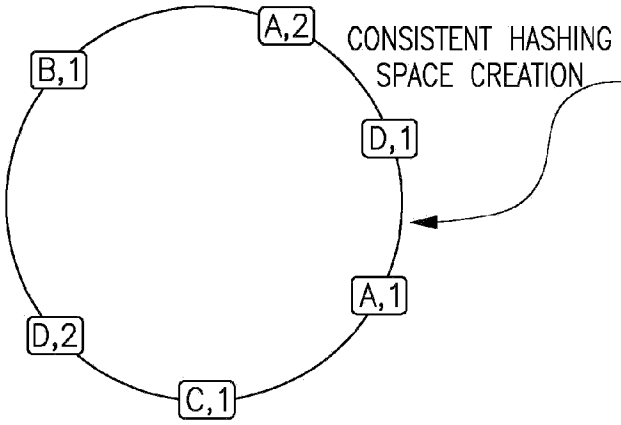
KEY	N_{ID}
K	H(K)

ACTUAL NODE ID

ORDERED V_{ID}'S LIST L:

PHYSICAL NODES	V_{ID}
1	H(A,2)
2	H(D,1)
3	H(A,1)
4	~~H(C,1)~~
5	H(D,2)
6	H(B,1)

FIG.3



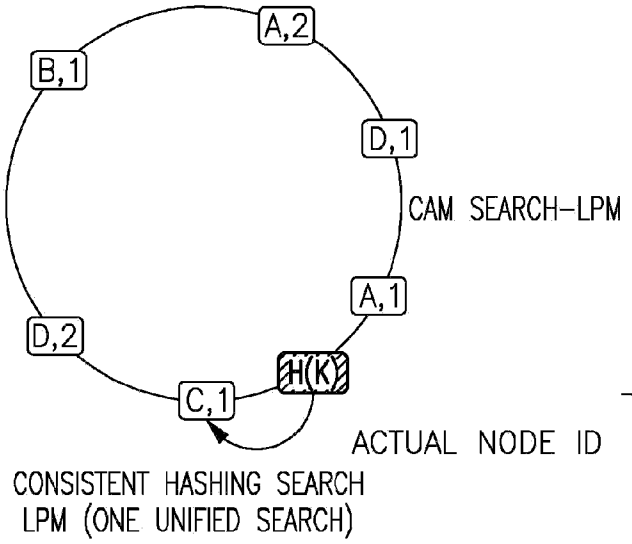
PHYSICAL NODES:

PHYSICAL NODES ID	REPLICAS
A	2
B	1
C	1
D	2

NPUs TCAM CONTENT LIST L:

TCAM ENTRY	PRIMARY V_{ID}	BACKUP V_{ID}
1	H(A,2)	H(D,1)
2	H(D,1)	H(A,1)
3	H(A,1)	H(C,1)
4	H(C,1)	H(D,2)
5	H(D,2)	H(B,1)
6	H(B,1)	H(A,2)

FIG.4



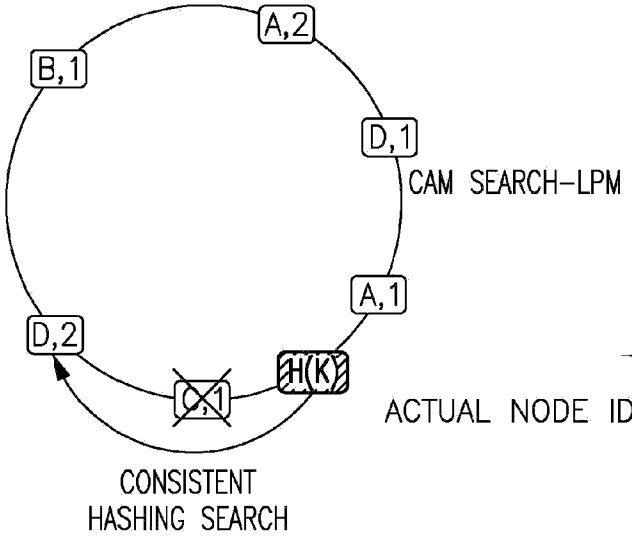
RESOLVED NODES:

KEY	N_{ID}
K	H(K)

NPUs TCAM CONTENT LIST L:

TCAM ENTRY	PRIMARY V_{ID}	BACKUP V_{ID}
1	H(A,2)	H(D,1)
2	H(D,1)	H(A,1)
3	H(A,1)	H(C,1)
4	H(C,1)	H(D,2)
5	H(D,2)	H(B,1)
6	H(B,1)	H(A,2)

FIG.5



RESOLVED NODES:

KEY	N_{ID}
K	H(K)

NPUs TCAM CONTENT LIST L:

TCAM ENTRY	PRIMARY V_{ID}	BACKUP V_{ID}
1	H(A,2)	H(D,1)
2	H(D,1)	H(A,1)
3	H(A,1)	H(C,1)
4	~~H(C,1)~~	H(D,2)
5	H(D,2)	H(B,1)
6	H(B,1)	H(A,2)

FIG.6

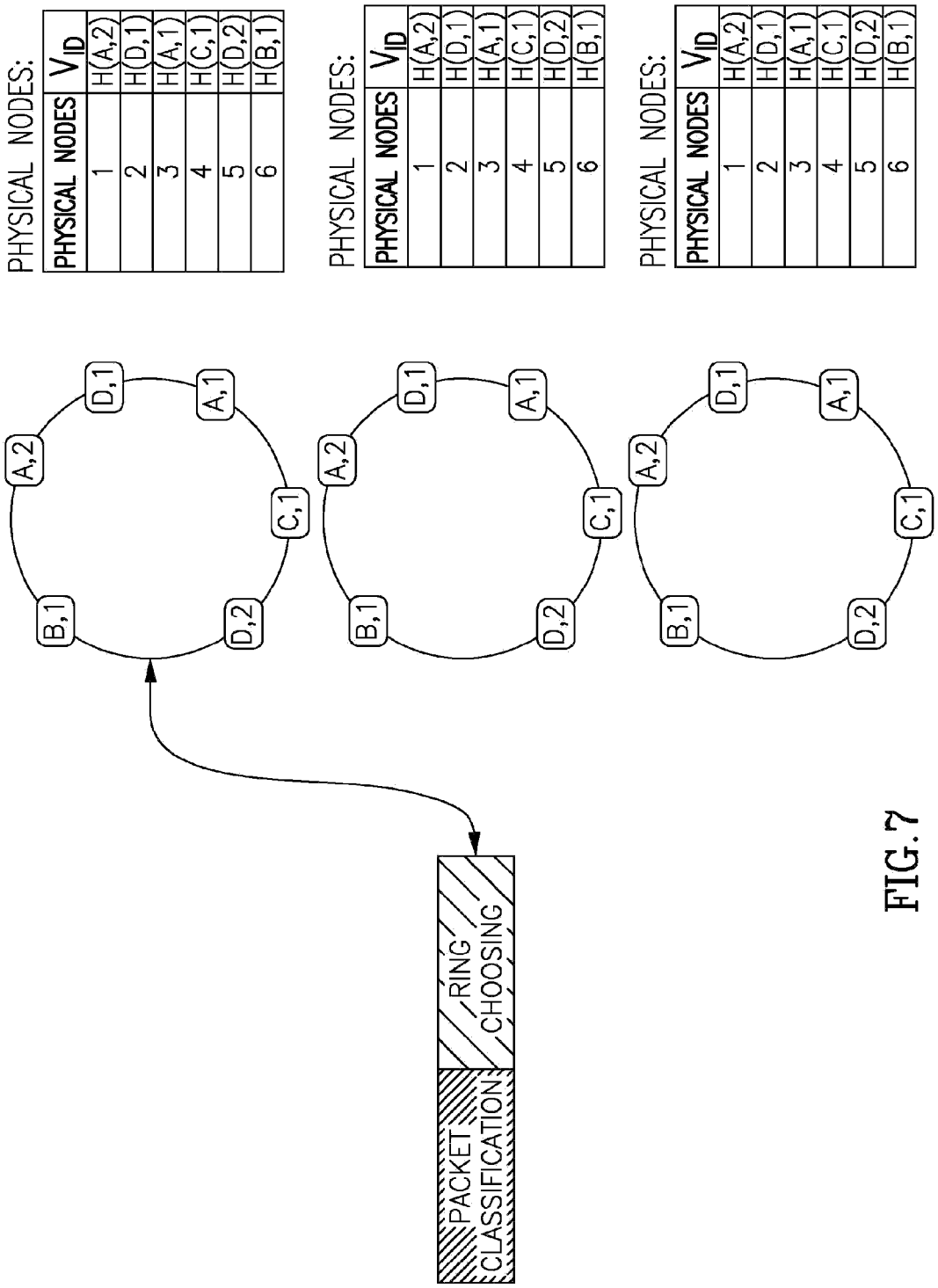

PHYSICAL NODES:
PHYSICAL NODES | V_{ID}
1 | H(A,2)
2 | H(D,1)
3 | H(A,1)
4 | H(C,1)
5 | H(D,2)
6 | H(B,1)
PHYSICAL NODES:
PHYSICAL NODES | V_{ID}
1 | H(A,2)
2 | H(D,1)
3 | H(A,1)
4 | H(C,1)
5 | H(D,2)
6 | H(B,1)
PHYSICAL NODES:
PHYSICAL NODES | V_{ID}
1 | H(A,2)
2 | H(D,1)
3 | H(A,1)
4 | H(C,1)
5 | H(D,2)
6 | H(B,1)
A,2
D,1
A,1
C,1
B,1
D,2
PACKET CLASSIFICATION
RING CHOOSING

FIG.7

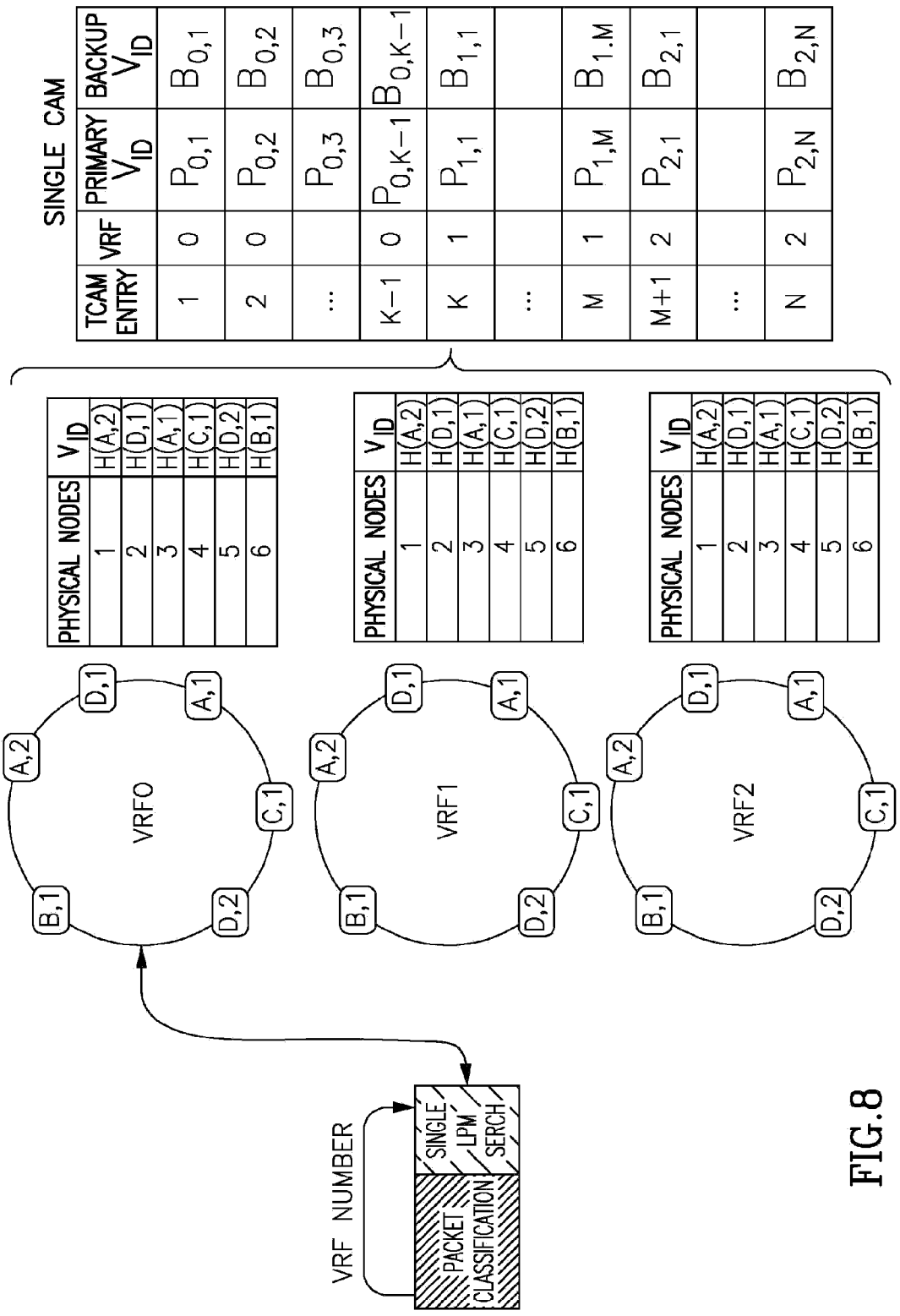

SINGLE CAM
TCAM ENTRY
VRF
PRIMARY V_{ID}
BACKUP V_{ID}
1 0 $P_{0,1}$ $B_{0,1}$
2 0 $P_{0,2}$ $B_{0,2}$
... $P_{0,3}$ $B_{0,3}$
K−1 0 $P_{0,K-1}$ $B_{0,K-1}$
K 1 $P_{1,1}$ $B_{1,1}$
... ...
M 1 $P_{1,M}$ $B_{1.M}$
M+1 2 $P_{2,1}$ $B_{2,1}$
... ...
N 2 $P_{2,N}$ $B_{2,N}$
PHYSICAL NODES V_{ID}
1 H(A,2)
2 H(D,1)
3 H(A,1)
4 H(C,1)
5 H(D,2)
6 H(B,1)
VRF0
VRF1
VRF2
A,2
D,1
A,1
B,1
C,1
D,2
VRF NUMBER
PACKET CLASSIFICATION
SINGLE LPM SERCH

FIG.8