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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
23 January 2025 (23.01.2025)

WIPO | PCT

(10) International Publication Number
WO 2025/017646 A1

(51) International Patent Classification:
H04W 72/50 (2023.01) *G06F 9/44* (2018.01)
G06F 9/50 (2006.01)

(21) International Application Number:
PCT/IN2024/051204

(22) International Filing Date:
12 July 2024 (12.07.2024)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
202321047646 14 July 2023 (14.07.2023) IN

(71) Applicant: **JIO PLATFORMS LIMITED** [IN/IN]; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN).

(72) Inventors: **MURARKA, Ankit**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **BHATNAGAR, Aayush**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **KISHORE, Jugal**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **KUMAR, Gaurav**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **SAHU, Kishan**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **KUMAR, Rahul**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **MEENA, Sunil**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **GURBANI, Gourav**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **CHAUDHARY, Sanjana**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **GANVEER, Chandra**; Office - 101,

(54) **Title:** METHOD AND SYSTEM FOR OPTIMAL ALLOCATION OF RESOURCES FOR EXECUTING KPI REQUESTS

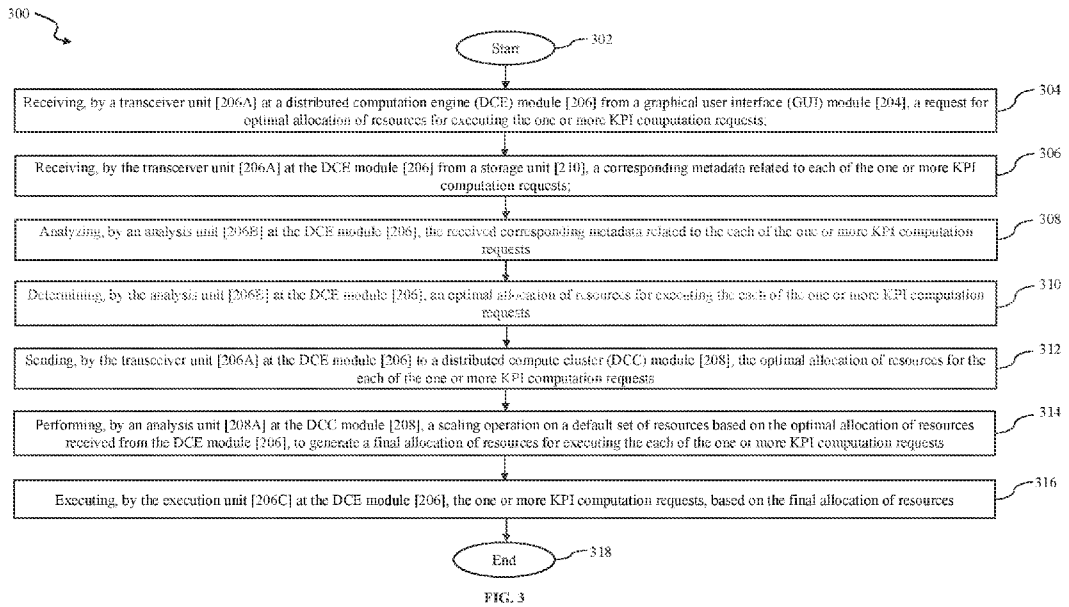


FIG. 3

(57) **Abstract:** The present disclosure relates to a method and a system for optimal allocation of resources for executing one or more KPI computation requests. The disclosure encompasses receiving a request for optimal resource allocation from a GUI module [204] and corresponding metadata related to each KPI computation request from a storage unit. An analysis unit [206b] at a DCE module [206] analyses the received metadata and determines an optimal allocation of resources for executing each KPI computation request. The determined optimal allocation is sent to a DCC module [208], which performs a scaling operation on a default set of resources based on the optimal allocation received from the DCE module [206]. This generates a final allocation of resources for executing each KPI computation request. The execution unit at the DCE module [206] then executes the one or more KPI computation requests based on the final allocation of resources.

[Continued on next page]

WO 2025/017646 A1

Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **DE, Supriya Kaushik**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **KUMAR, Debashish**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **TILALA, Mehul**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **VISHWAKARMA, Dharmendra Kumar**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **KUMAR, Yogesh**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **PATNAM, Niharika**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **GARG, Harshita**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **KUSHWAHA, Avinash**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **SONI, Sajal**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **TELGOTE, Kunal**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN). **RAJANI, Manasvi**; Office - 101, Saffron, Nr. Centre Point, Panchwati 5 Rasta, Ambawadi, Ahmedabad - 380006, Gujarat, Ahmedabad 380006 (IN).

(74) **Agent: SAHNEY, Garima**; SAIKRISHNA & ASSOCIATES ADVOCATES, B-140, Sector 51, Noida-201301, Uttar Pradesh, Noida 201301 (IN).

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

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *of inventorship (Rule 4.17(iv))*

Published:

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

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))*

METHOD AND SYSTEM FOR OPTIMAL ALLOCATION OF RESOURCES FOR EXECUTING KPI REQUESTS

TECHNICAL FIELD

[0001] Embodiments of the present disclosure generally relate to method and system for optimal allocation of resources for executing Key Performance Indicator (KPI) requests.

BACKGROUND

[0002] The following description of the related art is intended to provide background information pertaining to the field of the disclosure. This section may include certain aspects of the art that may be related to various features of the present disclosure. However, it should be appreciated that this section is used only to enhance the understanding of the reader with respect to the present disclosure, and not as admissions of the prior art.

[0003] Existing solutions in the existing art typically rely on manual or static approaches for resource allocation in distributed cluster computing. These solutions often struggle to dynamically adjust resources based on the varying demands of different user requests, leading to either underutilization or overallocation of resources. Underutilization can result in idle resources, which is a waste of computing power and financial resources. On the other hand, overallocation can lead to unnecessary expenditure and reduced efficiency, as resources are allocated to tasks that do not require them. Moreover, traditional methods may not effectively analyse the specific requirements of each request, such as the data size and frequency involved in the user request, leading to suboptimal performance. They also lack the ability to predict future resource requirements based on historical usage patterns, which can be crucial for efficiently handling varying workloads.

[0004] Thus, there exists an imperative need in the art for a method and system for optimal allocation of resources for executing KPI requests.

OBJECTS OF THE DISCLOSURE

[0005] Some of the objects of the present disclosure, which at least one embodiment disclosed herein satisfies are listed herein below.

[0006] It is an object of the present disclosure to provide a system and a method for optimal allocation of resources for executing KPI request.

[0007] It is another object of the present disclosure to provide a solution that enables identification of optimal amount of resources (CPU, RAM, and disc space) for executing KPI request.

[0008] It is yet another object of the present disclosure to provide a solution that improves performance by scaling resources allocated for executing KPI request, thereby improving response time.

[0009] It is yet another object of the present disclosure to provide a solution that ensures computing resources are utilised optimally, wherein default allocation of resources can be scaled to match workload requirements, thereby resulting in cost saving by minimising wasted resources and maximising utilisation of available computing power.

[0010] It is yet another object of the present disclosure to provide a solution that facilitates automatic scaling of resources for each KPI request in distributed cluster computing module, thereby resulting in faster and efficient execution of KPI request.

SUMMARY OF THE DISCLOSURE

[0011] This section is provided to introduce certain aspects of the present disclosure in a simplified form that are further described below in the detailed description. This summary is not intended to identify the key features or the scope of the claimed subject matter.

[0012] According to an aspect of the present disclosure, a method for optimal allocation of resources for executing one or more key performance indicators (KPI) computation requests is disclosed. The method includes receiving, by a transceiver unit at a distributed computation engine (DCE) module from a graphical user interface (GUI) module, a request for optimal allocation of resources for executing the one or more KPI computation requests. Next, the method includes receiving, by the transceiver unit at the DCE module from a storage unit, a corresponding metadata related to each of the one or more KPI computation requests. Next, the method includes analysing, by an analysis unit at the DCE module, the received corresponding metadata related to the each of

the one or more KPI computation requests. Next, the method includes determining, by the analysis unit at the DCE module, an optimal allocation of resources for executing the each of the one or more KPI computation requests. Next, the method includes sending, by the transceiver unit at the DCE module to a distributed compute cluster (DCC) module, the optimal allocation of resources for the each of the one or more KPI computation requests. Next, the method includes performing, by an analysis unit at the DCC module, a scaling operation on a default set of resources based on the optimal allocation of resources received from the DCE module, to generate a final allocation of resources for executing the each of the one or more KPI computation requests. Thereafter, the method includes executing, by the execution unit at the DCE module, the one or more KPI computation requests, based on the final allocation of resources.

[0013] In an exemplary aspect of the present disclosure, the analysing, by the analysis unit at the DCE module, the received metadata related to the each of the one or more KPI computation requests, is based on applying one or more machine learning (ML) based techniques.

[0014] In an exemplary aspect of the present disclosure, the scaling operation comprises at least one of an up-scaling operation and a downscaling operation, wherein the up-scaling operation refers to an addition of resources to the default set of resources, and the downscaling operation refers to a removal of resources from the default set of resources.

[0015] In an exemplary aspect of the present disclosure, sending, by the transceiver unit at the DCE module to the GUI module, a response related to executing the one or more KPI computation requests, based on the final allocation of resources, for one or more users; and displaying, by the GUI module, the response related to executing the one or more KPI computation requests.

[0016] In an exemplary aspect of the present disclosure, prior to the receiving, by the transceiver unit at the DCE module from the GUI module, the request for optimal allocation of resources for executing the one or more KPI computation requests. The method comprises: receiving, by the transceiver unit at the GUI module, a creation of the request for optimal allocation of resources for executing the one or more KPI computation requests, by manual inputs of one or more users.

[0017] In an exemplary aspect of the present disclosure, the analysis of the received corresponding metadata related to the each of the one or more KPI computation requests, by the analysis unit, is

performed based on one or more machine learning (ML) techniques, for a pre-defined period of time.

[0018] According to another aspect of the present disclosure, a system for optimal allocation of resources for executing one or more key performance indicators (KPI) computation requests is disclosed. The system comprising a distributed computation engine (DCE) module. The DCE module comprising a transceiver unit configured to receive, from a graphical user interface (GUI) module, a request for optimal allocation of resources for executing the one or more KPI computation requests; receive, from a storage unit, a corresponding metadata related to each of the one or more KPI computation requests. Further, the DCE module comprising an analysis unit connected at least to the transceiver unit, the analysis unit being configured to analyse the received corresponding metadata related to the each of the one or more KPI computation requests; determine an optimal allocation of resources for executing the each of the one or more KPI computation requests. Further, the transceiver unit further configured to send, to a distributed compute cluster (DCC) module, the optimal allocation of resources for the each of the one or more KPI computation requests. The DCC module comprising an analysis unit configured to perform a scaling operation on a default set of resources based on the optimal allocation of resources received from DCE module, to generate a final allocation of resources for executing the each of the one or more KPI computation requests; and the DCE module further comprising an execution unit configured to execute the one or more KPI computation requests, based on the final allocation of resources.

[0019] According to yet another aspect of the present disclosure, a non-transitory computer-readable storage medium storing instructions for managing connection between two or more peer network entities is disclosed. The instructions include executable code which, when executed by a processor, may cause the processor to receive, from a graphical user interface (GUI) module, a request for optimal allocation of resources for executing the one or more KPI computation requests; receive, from a storage unit, a corresponding metadata related to each of the one or more KPI computation requests; analyse the received corresponding metadata related to the each of the one or more KPI computation requests; determine an optimal allocation of resources for executing the each of the one or more KPI computation requests; perform a scaling operation on a default set of resources based on the optimal allocation of resources received from DCE module, to generate a final allocation of resources for executing the each of the one or more KPI computation

requests; and execute the one or more KPI computation requests, based on the final allocation of resources.

BRIEF DESCRIPTION OF THE DRAWINGS

[0020] The accompanying drawings, which are incorporated herein, and constitute a part of this disclosure, illustrate exemplary embodiments of the disclosed methods and systems in which like reference numerals refer to the same parts throughout the different drawings. Components in the drawings are not necessarily to scale, emphasis instead being placed upon clearly illustrating the principles of the present disclosure. Also, the embodiments shown in the figures are not to be construed as limiting the disclosure, but the possible variants of the method and system according to the disclosure are illustrated herein to highlight the advantages of the disclosure. It will be appreciated by those skilled in the art that disclosure of such drawings includes disclosure of electrical components or circuitry commonly used to implement such components.

[0021] FIG.1 illustrates an exemplary block diagram of a network performance management system, in accordance with the exemplary embodiments of the present invention.

[0022] FIG.2 illustrates an exemplary block diagram of a system for optimal allocation of resources for executing KPI request, in accordance with exemplary embodiments of the present disclosure.

[0023] FIG.3 illustrates an exemplary method flow diagram indicating the process for optimal allocation of resources for executing KPI request, in accordance with exemplary embodiments of the present disclosure.

[0024] FIG.4 illustrates an exemplary block diagram of a computing device upon which an embodiment of the present disclosure may be implemented.

[0025] The foregoing shall be more apparent from the following more detailed description of the disclosure.

DETAILED DESCRIPTION

[0026] In the following description, for the purposes of explanation, various specific details are set forth in order to provide a thorough understanding of embodiments of the present disclosure. It will be apparent, however, that embodiments of the present disclosure may be practiced without these specific details. Several features described hereafter may each be used independently of one another or with any combination of other features. An individual feature may not address any of the problems discussed above or might address only some of the problems discussed above.

[0027] The ensuing description provides exemplary embodiments only, and is not intended to limit the scope, applicability, or configuration of the disclosure. Rather, the ensuing description of the exemplary embodiments will provide those skilled in the art with an enabling description for implementing an exemplary embodiment. It should be understood that various changes may be made in the function and arrangement of elements without departing from the spirit and scope of the disclosure as set forth.

[0028] Specific details are given in the following description to provide a thorough understanding of the embodiments. However, it will be understood by one of ordinary skill in the art that the embodiments may be practiced without these specific details. For example, circuits, systems, processes, and other components may be shown as components in block diagram form in order not to obscure the embodiments in unnecessary detail.

[0029] Also, it is noted that individual embodiments may be described as a process which is depicted as a flowchart, a flow diagram, a data flow diagram, a structure diagram, or a block diagram. Although a flowchart may describe the operations as a sequential process, many of the operations may be performed in parallel or concurrently. In addition, the order of the operations may be re-arranged. A process is terminated when its operations are completed but could have additional steps not included in a figure.

[0030] The word “exemplary” and/or “demonstrative” is used herein to mean serving as an example, instance, or illustration. For the avoidance of doubt, the subject matter disclosed herein is not limited by such examples. In addition, any aspect or design described herein as “exemplary” and/or “demonstrative” is not necessarily to be construed as preferred or advantageous over other aspects or designs, nor is it meant to preclude equivalent exemplary structures and techniques known to those of ordinary skill in the art. Furthermore, to the extent that the terms “includes,”

"has," "contains," and other similar words are used in either the detailed description or the claims, such terms are intended to be inclusive—in a manner similar to the term "comprising" as an open transition word—without precluding any additional or other elements.

[0031] As used herein, a "processing unit" or "processor" or "operating processor" or "module" includes one or more processors, wherein processor refers to any logic circuitry for processing instructions. A processor may be a general-purpose processor, a special purpose processor, a conventional processor, a digital signal processor, a plurality of microprocessors, one or more microprocessors in association with a DSP core, a controller, a microcontroller, Application Specific Integrated Circuits, Field Programmable Gate Array circuits, any other type of integrated circuits, etc. The processor may perform signal coding data processing, input/output processing, and/or any other functionality that enables the working of the system according to the present disclosure. More specifically, the processor or processing unit is a hardware processor.

[0032] As used herein, "storage unit" or "memory unit" refers to a machine or computer-readable medium including any mechanism for storing information in a form readable by a computer or similar machine. For example, a computer-readable medium includes read-only memory ("ROM"), random access memory ("RAM"), magnetic disk storage media, optical storage media, flash memory devices or other types of machine-accessible storage media. The storage unit stores at least the data that may be required by one or more units of the system to perform their respective functions.

[0033] As used herein, the distributed compute cluster (DCC) module refers to a component responsible for dynamically scaling computational resources based on real-time demand and optimal allocation strategies determined by machine learning analysis. The DCC module analyses incoming resource allocation instructions, performs up-scaling or downscaling operations on the default set of resources, and generates a final allocation of resources tailored to execute specific key performance indicator (KPI) computation requests. The DCC module ensures efficient and effective resource utilization, minimizing wastage and enhancing the system's ability to handle varying workloads in distributed cluster computing environments.

[0034] As used herein, the distributed computation engine (DCE) module refers to a component that manages the reception, analysis, and execution of resource allocation requests for KPI computations. The DCE module includes a transceiver unit for receiving requests and metadata, an analysis unit for determining optimal resource allocation using machine learning techniques,

and an execution unit for carrying out the KPI computations based on the allocated resources. This module ensures efficient processing and optimal resource utilization in a distributed computing environment.

[0035] As discussed in the background section, the existing solutions in the existing art typically rely on manual or static approaches for resource allocation in distributed cluster computing. These solutions often struggle to dynamically adjust resources based on the varying demands of different user requests, leading to either underutilization or overallocation of resources. Underutilization can result in idle resources, which is a waste of computing power and financial resources. On the other hand, overallocation can lead to unnecessary expenditure and reduced efficiency, as resources are allocated to tasks that do not require them. Moreover, traditional methods may not effectively analyse the specific requirements of each request, such as the data size and frequency involved in the user request, leading to suboptimal performance. They also lack the ability to predict future resource requirements based on historical usage patterns, which can be crucial for efficiently handling varying workloads.

[0036] To overcome these and other inherent problems in the art, the present disclosure proposes a solution of automating the process of resource allocation in distributed cluster computing through the use of artificial intelligence and machine learning techniques. The system involves a distributed computation engine (DCE) that receives requests for key performance indicators (KPIs) and the corresponding metadata from a storage unit. By analysing this metadata with machine learning algorithms, the DCE determines the optimal allocation of resources such as CPU, RAM, and disk space for executing the KPI requests. This allocation is then sent to a distributed compute cluster (DCC), which automatically scales the resources to match the workload requirements based on the optimal allocation. This approach addresses the problems of manual and static resource allocation by dynamically adjusting resources according to the specific demands of each request. It ensures that resources are utilized efficiently, thereby avoiding underutilization and overallocation. Additionally, by leveraging machine learning algorithms to analyse historical usage patterns, the system can predict future resource requirements and adapt accordingly, leading to improved performance and efficient handling of varying workloads in distributed cluster computing.

[0037] The present invention also relates to a system for optimal allocation of resources for executing one or more key performance indicators (KPI) computation requests. The system comprises a distributed computation engine (DCE) module which is configured to receive a

request for optimal allocation of resources for executing one or more KPI computation requests from a graphical user interface (GUI) module. The DCE module is also configured to receive a corresponding metadata related to each of the one or more KPI computation requests from a storage unit. The DCE module is also configured to analyse the received corresponding metadata related to the each of the one or more KPI computation requests. The DCE module is also configured to determine an optimal allocation of resources for executing the each of the one or more KPI computation requests. The DCE module is also configured to send the optimal allocation of resources for the each of the one or more KPI computation requests to a distributed compute cluster (DCC) module. Further, the DCC module is configured to perform a scaling operation on a default set of resources based on the optimal allocation of resources received from the DCE module for generating a final allocation of resources for executing the each of the one or more KPI computation requests. Furthermore, the DCE module is further configured to execute the one or more KPI computation requests based on the final allocation of resources.

[0038] Hereinafter, exemplary embodiments of the present disclosure will be described with reference to the accompanying drawings.

[0039] FIG. 1 illustrates an exemplary block diagram of a network performance management system [100], in accordance with the exemplary embodiments of the present invention. Referring to FIG. 1, the network performance management system [100] comprises various sub-systems such as: integrated performance management system [100a], normalization layer [100b], computation layer [100d], anomaly detection layer [100o], streaming engine [100l], load balancer [100k], operations and management system [100p], API gateway system [100r], analysis engine [100h], parallel computing framework [100i], forecasting engine [100t], distributed file system [100j], mapping layer [100s], distributed data lake [100u], scheduling layer [100g], reporting engine [100m], message broker [100e], graph layer [100f], caching layer [100c], service quality manager [100q] and correlation engine[100n]. Exemplary connections between these subsystems is also as shown in FIG.1. However, it will be appreciated by those skilled in the art that the present disclosure is not limited to the connections shown in the diagram, and any other connections between various subsystems that are needed to realise the effects are within the scope of this disclosure.

[0040] Following are the various components of the network performance management system [100], the various components may include:

[0041] Integrated performance management (IPM) system [100a] comprises of one or more 5G performance engine [100v] and one or more 5G Key Performance Indicator (KPI) Engine [100u].

[0042] 5G Performance Management Engine [100v]: The 5G Performance Management engine [100v] is a crucial component of the integrated system, responsible for collecting, processing, and managing performance counter data from various data sources within the network. The gathered data includes metrics such as connection speed, latency, data transfer rates, and many others. This raw data is then processed and aggregated as required, forming a comprehensive overview of network performance. The processed information is then stored in a Distributed Data Lake [100u], a centralized, scalable, and flexible storage solution, allowing for easy access and further analysis. The 5G Performance Management engine [100v] also enables the reporting and visualization of this performance counter data, thus providing network administrators with a real-time, insightful view of the network's operation. Through these visualizations, operators can monitor the network's performance, identify potential issues, and make informed decisions to enhance network efficiency and reliability.

[0043] 5G Key Performance Indicator (KPI) Engine [100u]: The 5G Key Performance Indicator (KPI) Engine is a dedicated component tasked with managing the KPIs of all the network elements. It uses the performance counters, which are collected and processed by the 5G Performance Management engine from various data sources. These counters, encapsulating crucial performance data, are harnessed by the KPI engine [100u] to calculate essential KPIs. These KPIs might include data throughput, latency, packet loss rate, and more. Once the KPIs are computed, they are segregated based on the aggregation requirements, offering a multi-layered and detailed understanding of network performance. The processed KPI data is then stored in the Distributed Data Lake [100u], ensuring a highly accessible, centralized, and scalable data repository for further analysis and utilization. Similar to the Performance Management engine, the KPI engine [100u] is also responsible for reporting and visualization of KPI data. This functionality allows network administrators to gain a comprehensive, visual understanding of the network's performance, thus supporting informed decision-making and efficient network management.

[0044] Ingestion layer [not shown]: The Ingestion layer forms a key part of the Integrated Performance Management system. Its primary function is to establish an environment capable of handling diverse types of incoming data. This data may include Alarms, Counters, Configuration

parameters, Call Detail Records (CDRs), Infrastructure metrics, Logs, and Inventory data, all of which are crucial for maintaining and optimizing the network's performance. Upon receiving this data, the Ingestion layer processes it by validating its integrity and correctness to ensure it is fit for further use. Following validation, the data is routed to various components of the system, including the Normalization layer, Streaming Engine, Streaming Analytics, and Message Brokers. The destination is chosen based on where the data is required for further analytics and processing. By serving as the first point of contact for incoming data, the Ingestion layer plays a vital role in managing the data flow within the system, thus supporting comprehensive and accurate network performance analysis.

[0045] Normalization layer [100b]: The Normalization Layer [100b] serves to standardize, enrich, and store data into the appropriate databases. It takes in data that has been ingested and adjusts it to a common standard, making it easier to compare and analyse. This process of "normalization" reduces redundancy and improves data integrity. Upon completion of normalization, the data is stored in various databases like the Distributed Data Lake [100u], Caching Layer [100c], and Graph Layer [100f], depending on its intended use. The choice of storage determines how the data can be accessed and used in the future. Additionally, the Normalization Layer [100b] produces data for the Message Broker, a system that enables communication between different parts of the performance management system through the exchange of data messages. Moreover, the Normalization Layer [100b] supplies the standardized data to several other subsystems. These include the Analysis Engine [100h] for detailed data examination, the Correlation Engine [100n] for detecting relationships among various data elements, the Service Quality Manager [100q] for maintaining and improving the quality of services, and the Streaming Engine [100l] for processing real-time data streams. These subsystems depend on the normalized data to perform their operations effectively and accurately, demonstrating the Normalization Layer's [100b] critical role in the entire system.

[0046] Caching layer [100c]: The Caching Layer [100c] in the Integrated Performance Management system plays a significant role in data management and optimization. During the initial phase, the Normalization Layer [100b] processes incoming raw data to create a standardized format, enhancing consistency and comparability. The Normalizer Layer then inserts this normalized data into various databases. One such database is the Caching Layer [100c]. The Caching Layer [100c] is a high-speed data storage layer which temporarily holds data that is likely to be reused, to improve speed and performance of data retrieval. By storing frequently accessed

data in the Caching Layer [100c], the system significantly reduces the time taken to access this data, improving overall system efficiency and performance. Further, the Caching Layer [100c] serves as an intermediate layer between the data sources and the sub-systems, such as the Analysis Engine [100h], Correlation Engine [100n], Service Quality Manager, and Streaming Engine. The Normalization Layer [100b] is responsible for providing these sub-systems with the necessary data from the Caching Layer [100c].

[0047] Computation layer [100d]: The Computation Layer [100d] in the Integrated Performance Management system serves as the main hub for complex data processing tasks. In the initial stages, raw data is gathered, normalized, and enriched by the Normalization Layer [100b]. The Normalizer Layer then inserts this standardized data into multiple databases including the Distributed Data Lake [100u], Caching Layer [100c], and Graph Layer [100f], and also feeds it to the Message Broker [100e]. Within the Computation Layer [100d], several powerful sub-systems such as the Analysis Engine [100h], Correlation Engine [100n], Service Quality Manager, and Streaming Engine, utilize the normalized data. These systems are designed to execute various data processing tasks. The Analysis Engine performs in-depth data analytics to generate insights from the data. The Correlation Engine [100n] identifies and understands the relations and patterns within the data. The Service Quality Manager assesses and ensures the quality of the services. And the Streaming Engine processes and analyses the real-time data feeds. In essence, the Computation Layer [100d] is where all major computation and data processing tasks occur. It uses the normalized data provided by the Normalization Layer [100b], processing it to generate useful insights, ensure service quality, understand data patterns, and facilitate real-time data analytics.

[0048] Message broker [100e]: The Message Broker [100e], an integral part of the Integrated Performance Management system, operates as a publish-subscribe messaging system. It orchestrates and maintains the real-time flow of data from various sources and applications. At its core, the Message Broker [100e] facilitates communication between data producers and consumers through message-based topics. This creates an advanced platform for contemporary distributed applications. With the ability to accommodate a large number of permanent or ad-hoc consumers, the Message Broker [100e] demonstrates immense flexibility in managing data streams. Moreover, it leverages the filesystem for storage and caching, boosting its speed and efficiency. The design of the Message Broker [100e] is centred around reliability. It is engineered to be fault-tolerant and mitigate data loss, ensuring the integrity and consistency of the data. With its robust design and

capabilities, the Message Broker [100e] forms a critical component in managing and delivering real-time data in the system.

[0049] Graph layer [100f]: The Graph Layer [100f], serving as the Relationship Modeler, plays a pivotal role in the Integrated Performance Management system. It can model a variety of data types, including alarm, counter, configuration, CDR data, Infra-metric data, 5G Probe Data, and Inventory data. Equipped with the capability to establish relationships among diverse types of data, the Relationship Modeler offers extensive modelling capabilities. For instance, it can model Alarm and Counter data, Vprobe and Alarm data, elucidating their interrelationships. Moreover, the Modeler should be adept at processing steps provided in the model and delivering the results to the system requested, whether it be a Parallel Computing system, Workflow Engine, Query Engine, Correlation System [100n], 5G Performance Management Engine, or 5G KPI Engine [100u]. With its powerful modelling and processing capabilities, the Graph Layer [100f] forms an essential part of the system, enabling the processing and analysis of complex relationships between various types of network data.

[0050] Scheduling layer [100g]: The Scheduling Layer [100g] serves as a key element of the Integrated Performance Management System, endowed with the ability to execute tasks at predetermined intervals set according to user preferences. A task might be an activity performing a service call, an API call to another microservice, the execution of an Elastic Search query, and storing its output in the Distributed Data Lake [100u] or Distributed File System or sending it to another micro-service. The versatility of the Scheduling Layer [100g] extends to facilitating graph traversals via the Mapping Layer to execute tasks. This crucial capability enables seamless and automated operations within the system, ensuring that various tasks and services are performed on schedule, without manual intervention, enhancing the system's efficiency and performance. In sum, the Scheduling Layer [100g] orchestrates the systematic and periodic execution of tasks, making it an integral part of the efficient functioning of the entire system.

[0051] Analysis Engine [100h]: The Analysis Engine [100h] forms a crucial part of the Integrated Performance Management System, designed to provide an environment where users can configure and execute workflows for a wide array of use-cases. This facility aids in the debugging process and facilitates a better understanding of call flows. With the Analysis Engine [100h], users can perform queries on data sourced from various subsystems or external gateways. This capability allows for an in-depth overview of data and aids in pinpointing issues. The system's flexibility

allows users to configure specific policies aimed at identifying anomalies within the data. When these policies detect abnormal behaviour or policy breaches, the system sends notifications, ensuring swift and responsive action. In essence, the Analysis Engine [100h] provides a robust analytical environment for systematic data interrogation, facilitating efficient problem identification and resolution, thereby contributing significantly to the system's overall performance management.

[0052] Parallel Computing Framework [100i]: The Parallel Computing Framework [100i] is a key aspect of the Integrated Performance Management System, providing a user-friendly yet advanced platform for executing computing tasks in parallel. This framework showcases both scalability and fault tolerance, crucial for managing vast amounts of data. Users can input data via Distributed File System (DFS) [100j] locations or Distributed Data Lake (DDL) indices. The framework supports the creation of task chains by interfacing with the Service Configuration Management (SCM) Sub-System. Each task in a workflow is executed sequentially, but multiple chains can be executed simultaneously, optimizing processing time. To accommodate varying task requirements, the service supports the allocation of specific host lists for different computing tasks. The Parallel Computing Framework [100i] is an essential tool for enhancing processing speeds and efficiently managing computing resources, significantly improving the system's performance management capabilities.

[0053] Distributed File System [100j]: The Distributed File System (DFS) [100j] is a critical component of the Integrated Performance Management System, enabling multiple clients to access and interact with data seamlessly. This file system is designed to manage data files that are partitioned into numerous segments known as chunks. In the context of a network with vast data, the DFS [100j] effectively allows for the distribution of data across multiple nodes. This architecture enhances both the scalability and redundancy of the system, ensuring optimal performance even with large data sets. DFS [100j] also supports diverse operations, facilitating the flexible interaction with and manipulation of data. This accessibility is paramount for a system that requires constant data input and output, as is the case in a robust performance management system.

[0054] Load Balancer [100k]: The Load Balancer (LB) [100k] is a vital component of the Integrated Performance Management System, designed to efficiently distribute incoming network traffic across a multitude of backend servers or microservices. Its purpose is to ensure the even

distribution of data requests, leading to optimized server resource utilization, reduced latency, and improved overall system performance. The LB [100k] implements various routing strategies to manage traffic. These include round-robin scheduling, header-based request dispatch, and context-based request dispatch. Round-robin scheduling is a simple method of rotating requests evenly across available servers. In contrast, header and context-based dispatching allow for more intelligent, request-specific routing. Header-based dispatching routes requests based on data contained within the headers of the Hypertext Transfer Protocol (HTTP) requests. Context-based dispatching routes traffic based on the contextual information about the incoming requests. For example, in an event-driven architecture, the LB [100k] manages event and event acknowledgments, forwarding requests or responses to the specific microservice that has requested the event. This system ensures efficient, reliable, and prompt handling of requests, contributing to the robustness and resilience of the overall performance management system.

[0055] Streaming Engine [100l]: The Streaming Engine [100l], also referred to as Stream Analytics, is a critical subsystem in the Integrated Performance Management System. This engine is specifically designed for high-speed data pipelining to the User Interface (UI). Its core objective is to ensure real-time data processing and delivery, enhancing the system's ability to respond promptly to dynamic changes. Data is received from various connected subsystems and processed in real-time by the Streaming Engine [100l]. After processing, the data is streamed to the UI, fostering rapid decision-making and responses. The Streaming Engine [100l] cooperates with the Distributed Data Lake [100u], Message Broker [100e], and Caching Layer [100c] to provide seamless, real-time data flow. Stream Analytics is designed to perform required computations on incoming data instantly, ensuring that the most relevant and up-to-date information is always available at the UI. Furthermore, this system can also retrieve data from the Distributed Data Lake [100u], Message Broker [100e], and Caching Layer [100c] as per the requirement and deliver it to the UI in real-time. The streaming engine's [100l] ultimate goal is to provide fast, reliable, and efficient data streaming, contributing to the overall performance of the management system.

[0056] Reporting Engine [100m]: The Reporting Engine [100m] is a key subsystem of the Integrated Performance Management System. The fundamental purpose of designing the Reporting Engine [100m] is to dynamically create report layouts of API data, catered to individual client requirements, and deliver these reports via the Notification Engine (not shown). The REM serves as the primary interface for creating custom reports based on the data visualized through the client's dashboard. These custom dashboards, created by the client through the User Interface

(UI), provide the basis for the Reporting Engine [100m] to process and compile data from various interfaces. The main output of the Reporting Engine [100m] is a detailed report generated in Excel format. The Reporting Engine's [100m] unique capability to parse data from different subsystem interfaces, process it according to the client's specifications and requirements, and generate a comprehensive report makes it an essential component of this performance management system. Furthermore, the Reporting Engine [100m] integrates seamlessly with the Notification Engine (not shown) to ensure timely and efficient delivery of reports to clients via email, ensuring the information is readily accessible and usable, thereby improving overall client satisfaction and system usability.

[0057] As shown in FIG.2, an exemplary block diagram of a system [200] for optimal allocation of resources for executing KPI request is shown, in accordance with the exemplary embodiments of the present invention. The system [200] comprises at least one user device [202], at least one GUI module [204]. In an embodiment, the GUI module comprising at least one transceiver unit [204a]. The system [200] further comprises at least one distributed computation engine (DCE) module [206], at least one distributed computer cluster (DCC) module [208] and at least one storage unit [210]. In an embodiment, the DCE module [206] comprises at least one transceiver unit [206a], analysis unit [206b] and execution unit [206c]. In an embodiment, the DCC module [208] comprises at least one transceiver unit [208a], analysis unit [208b], worker node [208c] and cluster manager [208d]. Also, all of the components/ units of the system [200] are assumed to be connected to each other unless otherwise indicated below. Also, in FIG. 2 only a few units are shown, however, the system [200] may comprise multiple such units or the system [200] may comprise any such numbers of said units, as required to implement the features of the present disclosure. Further, in an implementation, the system [200] may reside in a server or a user device [202]. In yet another implementation, the system [200] may reside partly in the server and partly in the user device [202].

[0058] The system [200] for optimal allocation of resources for executing one or more key performance indicators (KPI) computation requests comprises the distributed computation engine (DCE) module [206]. The DCE module [206] comprises the transceiver [206a] configured to receive, from the graphical user interface (GUI) module [204], a request for optimal allocation of resources for executing the one or more KPI computation requests. The transceiver [206a] serves as the primary interface for initiating the resource allocation process. Upon reception of a request from the GUI module [204], which acts as the front-end platform through which users input their

KPI-related queries and parameters, the transceiver unit [206a] is configured for capturing these inputs and channelling them into the DCE module [206]. The request refers to a user-initiated demand for the computation of one or more Key Performance Indicators (KPIs). The request includes, but not limited only to parameters such as the type of KPI to be computed, the data range, and frequency of the data involved. The request may also contain metadata detailing the characteristics of the data, such as size and historical usage patterns to facilitate the system to analyse and determine the optimal allocation of resources needed to efficiently process and compute the requested KPIs. For example, the request might involve a user asking for the computation of KPIs related to network performance over the past month. The request would specify the KPIs needed, such as average latency and packet loss rates, and include details about the data, like daily usage logs and file sizes. The metadata assists the system analyse the requirements and allocate the optimal resources to process the request efficiently.

[0059] The KPIs further comprise key performance indicators, which are quantifiable measures of performance over time for specific objectives. Examples of KPIs include UE mobility, which measures the ability of a user equipment (UE) to maintain connectivity while moving across different network cells or areas; packet loss, which indicates the percentage of data packets lost during transmission and affects the quality and reliability of the network; latency, which refers to the time delay experienced in the network and is particularly important for real-time applications and services; and call-drop rate, which measures the frequency at which active calls are unexpectedly terminated and is critical for maintaining service quality in voice communication. These KPIs are essential for assessing the efficiency and effectiveness of various network functions, allowing for the tracking of performance trends, and enabling informed decision-making to improve overall performance. By regularly monitoring these KPIs, organizations can identify areas needing improvement, optimize resource allocation, and deliver high-quality services.

[0060] In an aspect, prior to the Distributed Computation Engine (DCE) module [206] receiving, from the Graphical User Interface (GUI) module [204], the request for optimal allocation of resources for executing one or more Key Performance Indicator (KPI) computation requests, a transceiver unit [204a] at the GUI module [204] is configured to receive the creation of the request for optimal allocation of resources for executing one or more KPI computation requests. The configuration allows users to manually create and submit requests for optimal resource allocation

through the GUI module [204], which are then transmitted to the DCE module [206] for processing and execution.

[0061] The transceiver unit [206a] is further configured to receive, from a storage unit [210], corresponding metadata related to each of the one or more KPI computation requests. This metadata contains detailed information about the performance data that facilitates in the computation of KPIs. Metadata may include, but is not limited to, the size of data sets, the frequency of data generation, and other data characteristics that directly impact the resource needs for processing the KPI requests. For example, the system [200] is tasked with computing a KPI related to the average network latency of a telecommunications network function, such as the Access and Mobility Management Function (AMF) in a 5G network. The transceiver unit [206a] receives a request from the GUI module [204] to calculate this KPI. To fulfil this request, the transceiver unit [206a] also retrieves corresponding metadata from the storage unit [210]. The metadata might include information such as the size of the data set containing latency measurements (e.g., 500 MB), the frequency at which these measurements are recorded (e.g., every 10 minutes), and the time period over which the data is to be analysed (e.g., the past 30 days).

[0062] The analysis unit [206b] is connected at least to the transceiver unit [206a]. The analysis unit [206b] is configured to analyse the received corresponding metadata related to each of the one or more KPI computation requests. The analysis involves examining the details provided in the metadata, such as the data sizes, the frequency of the data, and other relevant information for each network function. The analysis is performed to understand the load usage patterns and the computational requirements for executing the one or more KPI computation requests.

[0063] The analysis unit [206b] is further configured to determine an optimal allocation of resources for executing each of the one or more KPI computation requests. This involves assessing the computational requirements based on the analysis of the received metadata. The analysis unit [206b] configured to analyse, at the DCE module [206], the received metadata related to the each of the one or more KPI computation requests, is based on applying one or more machine learning (ML) based techniques. The analysis performed based the on one or more machine learning (ML) techniques, is for a pre-defined period of time. The one or more ML techniques based model is trained based on historical data on network performance, resource usage, and key performance indicators (KPIs) from various network functions such as access and mobility management function (AMF), policy control function (PCF), and diameter routing agent (DRA). By

determining the optimal allocation of resources, such as CPU, RAM, and disk space, the system can ensure that each of the one or more KPI computation requests is executed effectively, without underutilization or overutilization of resources. Examples of machine learning (ML) techniques include, but not limited only to regression analysis, clustering, and time-series forecasting. Regression analysis can predict the resource requirements based on historical data sizes and request frequencies. Clustering can group similar requests to optimize resource allocation patterns. Time-series forecasting can predict future resource demands based on trends observed in the data over predefined periods. By applying these ML techniques, the system can dynamically and accurately determine the optimal resources needed for executing KPI computation requests, thereby enhancing efficiency and performance in distributed cluster computing environments.

[0064] In an embodiment, the analysis of the received corresponding metadata related to each of the one or more Key Performance Indicator (KPI) computation requests is performed by the analysis unit [208b]. The analysis is based on one or more machine learning (ML) techniques and is carried out over a predefined period of time. The use of ML techniques allows for the identification of patterns and trends within the metadata, which can inform the optimal allocation of resources for the execution of the KPI computation requests. The predefined period of time refers to a specific duration over which the analysis is conducted to ensure relevance and accuracy in the resource allocation process.

[0065] The transceiver unit [206a] is further configured to send to a distributed compute cluster (DCC) module [208], the optimal allocation of resources for each of the one or more KPI computation requests. The transmission occurs after the analysis unit [206b] has determined the most optimal allocation of resources, such as CPU, RAM, and disk space, needed to execute the one or more KPI computation requests effectively. The optimal allocation of resources is based on the analysis of the received metadata, which includes the load usage patterns over a predefined number of days.

[0066] The DCC module [208] is communicatively coupled to the DCE module [206]. The DCC module [208] comprises an analysis unit [208b] configured to perform a scaling operation on a default set of resources based on the optimal allocation of resources received from the DCE module [206], to generate a final allocation of resources for executing the each of the one or more KPI computation requests. The scaling operation comprises at least one of an up-scaling operation and a downscaling operation, wherein the up-scaling operation refers to an addition of resources to the

default set of resources, and the downscaling operation refers to a removal of resources from the default set of resources. The result of the scaling operation is the generation of a final allocation of resources that is configured for specific requirements of each of the one or more KPI computation requests. By dynamically adjusting the resource allocation, the DCC module [208] ensures that the distributed compute cluster can efficiently handle the workload, optimizing both performance and resource utilization.

[0067] The DCE module [206] further comprises an execution unit [206C] that is communicatively coupled to the analysis unit [206b]. The execution unit [206C] is configured to execute the one or more KPI computation requests based on the final allocation of resources determined by the analysis unit [206b]. The final allocation of resources is optimized for each of the one or more KPI computation requests, ensuring that the necessary computing resources, such as CPU, RAM, and disk space, are available to efficiently process the requests. each of the one or more KPI computation requests, the execution unit [206C] ensures that the distributed computation engine can handle varying workloads with optimal efficiency.

[0068] The DCC module [208] further comprises the worker node [208c] configured to run application code within the cluster. In a server clustering setup, the worker node [208c] facilitates in executing the actual tasks and computations required by user requests. The worker node's ability to run application code allows it to contribute to the overall processing power of the cluster, thereby enhancing the system's capacity to handle concurrent requests and large datasets efficiently.

[0069] The DCC module [208] further comprises the cluster manager [208d] configured to control session traffic and distribute activity equally among the available servers within the cluster. The cluster manager [208d] acts as the managing entity within the server clustering setup, ensuring that all incoming requests are efficiently allocated to the appropriate worker nodes [208c]. By managing and balancing the workload across the cluster, the cluster manager [208d] facilitates in maintaining high availability and optimal performance. The cluster manager [208d] facilitates in ensuring that no single server is overburdened while others remain idle, thereby maximizing the utilization of resources and improving the overall efficiency and reliability of the distributed compute cluster.

[0070] The transceiver unit [206a] of the DCE module [206] is further configured to send to the GUI module [204], a response related to executing the one or more KPI computation requests, based on the final allocation of resources. The response is communicated back to the GUI module

[204] to provide feedback to the user regarding the status and results of the KPI computation requests. The final allocation of resources, determined by the DCE module [206], ensures that the KPI computation requests are executed efficiently, utilizing the optimal resources necessary for their completion. The response refers to the output or feedback provided by the system after executing one or more Key Performance Indicator (KPI) computation requests. The response is generated based on the final allocation of resources determined by the distributed compute cluster (DCC) module. The response includes detailed information on the execution status, performance metrics, and results of the KPI computations, which are then sent back to the graphical user interface (GUI) module for display to the user. For example, after a user submits a request for KPI computation, the system processes the request and allocates the necessary resources. Once the computation is complete, the system generates a response that includes details such as completion status (e.g., successful or failed), the time taken for the computation, and the computed KPI values. This response is then sent back to the GUI module, where it is displayed to the user, providing them with immediate feedback on their request.

[0071] The GUI module [204] is configured to display the response related to executing the one or more KPI computation requests. The response provides the user with information regarding the outcome of the one or more KPI computation requests, including the status of execution and any relevant results or metrics. The display of the response on the GUI module [204] allows for a user-friendly interface, enabling users to easily understand and interpret the results of their requests. It would be appreciated by the person skilled in the art that display the response enhances the user experience by providing a clear and accessible way to view the results of KPI computation requests within the distributed cluster computing environment.

[0072] It accomplishes this by utilizing sophisticated machine learning algorithms that process the metadata to predict the resource requirements with high accuracy, ensuring that each computation request is matched with the necessary CPU, RAM, and disc space. This process ensures that the system is prepared to efficiently handle each KPI request by dynamically adjusting resource allocations, thus enhancing the overall responsiveness and efficacy of the distributed cluster computing environment.

[0073] Referring to FIG. 3 an exemplary method flow diagram [300], for optimal allocation of resources for executing KPI request, in accordance with exemplary embodiments of the present invention is shown. In an implementation the method [300] is performed by the system [200]. As

shown in FIG.3, the method [300] starts at step [302] when a user device [202] creates or executes request.

[0074] At step [304], the method [300] as disclosed by the present disclosure comprises receiving, by a transceiver unit [206a] at a distributed computation engine (DCE) module [206] from a graphical user interface (GUI) module [204], a request for optimal allocation of resources for executing the one or more KPI computation requests. The transceiver [206a] serves as the primary interface for initiating the resource allocation process. Upon reception of a request from the GUI module [204], which acts as the front-end platform through which users input their KPI-related queries and parameters, the transceiver unit [206a] is configured for capturing these inputs and channelling them into the DCE module [206]. The request contains inquiries that are made by a user of GUI module [204] to determine as to how the resources should be optimally allocated to handle computation requests related to one or more KPIs. The one or more KPI computation requests are requests for optimal resource allocation through the GUI module [204], which are then transmitted to the DCE module [206], In an exemplary aspect, one or more user or network administrator may request optimal allocation of resources for executing the one or more KPI computation request related to one or more network functions such as, but not limited to, AMF, SMF and PCF etc., and the DCE module [206] receives the request. In an aspect, prior to the Distributed Computation Engine (DCE) module [206] receiving, from the Graphical User Interface (GUI) module [204], the request for optimal allocation of resources for executing one or more Key Performance Indicator (KPI) computation requests, a transceiver unit [204a] at the GUI module [204] is configured to receive the creation of the request for optimal allocation of resources for executing one or more KPI computation requests. The configuration allows users to manually create and submit requests for optimal resource allocation through the GUI module [204], which are then transmitted to the DCE module [206] for processing and execution.

[0075] Next, at step [306], the method [300] as disclosed by the present disclosure comprises receiving, by the transceiver unit [206a] at the DCE module [206] from a storage unit [210], a corresponding metadata related to each of the one or more KPI computation requests. Metadata may include, but is not limited to, the size of data sets, the frequency of data generation, and other data characteristics that directly impact the resource needs for processing the KPI requests. In an exemplary aspect, the metadata refers to data that provides information about other data (KPI related network function data). The metadata may contain details such as quarterly, hourly, daily, and weekly data sizes, as well as the frequency of the data for the network functions of the network.

[0076] Next, at step [308], the method [300] as disclosed by the present disclosure comprises analysing, by an analysis unit [206b] at the DCE module [206], the received corresponding metadata related to the each of the one or more KPI computation requests. The analysis involves examining the details provided in the metadata, such as the data sizes, the frequency of the data, and other relevant information for each network function. The analysis is performed to understand the load usage patterns and the computational requirements for executing the one or more KPI computation requests.

[0077] Next, at step [310], the method [300] as disclosed by the present disclosure comprises determining, by the analysis unit [206b] at the DCE module [206], an optimal allocation of resources for executing the each of the one or more KPI computation requests. This involves assessing the computational requirements based on the analysis of the received metadata. The analysis unit [206b] configured to analyse, at the DCE module [206], the received metadata related to the each of the one or more KPI computation requests, is based on applying one or more machine learning (ML) based techniques. The analysis performed based the on one or more machine learning (ML) techniques, is for a pre-defined period of time. The predefined period of time refers to a specific duration over which the analysis is conducted to ensure relevance and accuracy in the resource allocation process.

[0078] The one or more ML techniques-based model is trained based on historical data on network performance, resource usage, and key performance indicators (KPIs) from various network functions such as access and mobility (AMF), policy and control function (PCF), and diameter routing agent (DRA). For example, if there is a call drop issues in the network because of the overutilization of the network resource, the one or more machine learning model, which is already trained on historical data related to that issue, performs an analysis to determine how much resources are to be optimally allocated to that network for executing each of the one or more KPI computation requests to rectify the issue of call drop. By determining the optimal allocation of resources, such as CPU, RAM, and disk space, the system can ensure that each of the one or more KPI computation requests is executed effectively, without underutilization or overutilization of resources. In an implementation of the invention, the analysis unit [206b] analyses the received metadata related to the each of the one or more KPI computation requests based on applying one or more machine learning (ML) based techniques or model. The ML model is pre-trained based on test data related to network function performance, trends, anomaly detection and metadata

related to the KPIs. In an implementation, the analysis of the received corresponding metadata related to the each of the one or more KPI computation requests, by the analysis unit [206b], is performed based on one or more machine learning (ML) techniques, for a pre-defined period of time. The ML based model analyses the metadata of the network function over the predefined number of days, load usage patterns over the predefined number of days such as, but not limited to, 30 days or 50 days or the like.

[0079] Next, at step [312], the method [300] as disclosed by the present disclosure comprises sending, by the transceiver unit [206a] at the DCE module [206] to a distributed compute cluster (DCC) module [208], the optimal allocation of resources for the each of the one or more KPI computation requests. The transmission occurs after the analysis unit [206b] has determined the most optimal allocation of resources, such as CPU, RAM, and disk space, needed to execute the one or more KPI computation requests effectively. The optimal allocation of resources is based on the analysis of the received metadata, which includes the load usage patterns over a predefined number of days.

[0080] Next, at step [314], the method [300] as disclosed by the present disclosure comprises performing, by an analysis unit [208b] at the DCC module [208], a scaling operation on a default set of resources based on the optimal allocation of resources received from the DCE module [206], to generate a final allocation of resources for executing the each of the one or more KPI computation requests. The scaling operation comprises at least one of an up-scaling operation and a downscaling operation, wherein the up-scaling operation refers to an addition of resources to the default set of resources, and the downscaling operation refers to a removal of resources from the default set of resources. The result of the scaling operation is the generation of a final allocation of resources that is configured for specific requirements of each of the one or more KPI computation requests. By dynamically adjusting the resource allocation, the DCC module [208] ensures that the distributed compute cluster can efficiently handle the workload, optimizing both performance and resource utilization.

[0081] Next, at step [316], the method [300] as disclosed by the present disclosure comprises executing, by the execution unit [206c] at the DCE module [206], the one or more KPI computation requests, based on the final allocation of resources.

[0082] The method [300] implemented by the system [200] comprises the DCE module [206], which is configured to execute via the execution unit [206c] the one or more KPI computation requests based on the final allocation of resources. The final allocation of resources is optimized for each of the one or more KPI computation requests, ensuring that the necessary computing resources, such as CPU, RAM, and disk space, are available to efficiently process the requests. By dynamically adjusting the resource allocation in response to the demands of each of the one or more KPI computation requests, the execution unit [206C] ensures that the distributed computation engine can handle varying workloads with optimal efficiency. In an implementation, the resource allocation may have no further modifications or updates, since the resources are present in sufficient number or capacity to manage KPI computation requests.

[0083] In an implementation of the present disclosure, the method [300] further comprises sending, by the DCE module [206] via the transceiver unit [206a] to the GUI module [204], a response related to executing the one or more KPI computation requests, based on the final allocation of resources, for display via the GUI module [204] for one or more users or network administrator. In an implementation, the method [300] comprises displaying, by the GUI module [204], the response related to executing the one or more KPI computation requests.

[0084] Thereafter, the method [300] terminates at step [318].

[0085] Although the present method has been explained with reference to one KPI request, it will be appreciated by those skilled in the art that the present disclosure is not limited thereto. The present disclosure encompasses parallelly handling multiple KPI requests received at the DCE module [206].

[0086] Referring to FIG. 4, which illustrates an exemplary block diagram of a computing device [400] (also referred to herein as computer system [400]) upon which an embodiment of the present disclosure may be implemented. In an implementation, the computing device [400] implements the method for providing an automated scaling of resources in distributed cluster computing using ML based techniques within a network performance management system [100] using the system [200]. In another implementation, the computing device [400] itself implements the method for providing an automated scaling of resources in distributed cluster computing using ML based techniques within a network performance management system [100] using one or more units

configured within the computing device [400], wherein said one or more units are capable of implementing the features as disclosed in the present disclosure.

[0087] The computing device [400] may include a bus [402] or other communication mechanism for communicating information, and a processor [404] coupled with bus [402] for processing information. The processor [404] may be, for example, a general purpose microprocessor. The computing device [400] may also include a main memory [406], such as a random access memory (RAM), or other dynamic storage device, coupled to the bus [402] for storing information and instructions to be executed by the processor [404]. The main memory [406] also may be used for storing temporary variables or other intermediate information during execution of the instructions to be executed by the processor [404]. Such instructions, when stored in non-transitory storage media accessible to the processor [404], render the computing device [400] into a special-purpose machine that is customized to perform the operations specified in the instructions. The computing device [400] further includes a read only memory (ROM) [408] or other static storage device coupled to the bus [402] for storing static information and instructions for the processor [404].

[0088] A storage device [410], such as a magnetic disk, optical disk, or solid-state drive is provided and coupled to the bus [402] for storing information and instructions. The computing device [400] may be coupled via the bus [402] to a display [412], such as a cathode ray tube (CRT), for displaying information to a computer user. An input device [414], including alphanumeric and other keys, may be coupled to the bus [402] for communicating information and command selections to the processor [404]. Another type of user input device may be a cursor controller [416], such as a mouse, a trackball, or cursor direction keys, for communicating direction information and command selections to the processor [404], and for controlling cursor movement on the display [412]. This input device typically has two degrees of freedom in two axes, a first axis (e.g., x) and a second axis (e.g., y), that allow the device to specify positions in a plane.

[0089] The computing device [400] may implement the techniques described herein using customized hard-wired logic, one or more ASICs or FPGAs, firmware, and/or program logic which in combination with the computing device [400] causes or programs the computing device [400] to be a special-purpose machine. According to one embodiment, the techniques herein are performed by the computing device [400] in response to the processor [404] executing one or more sequences of one or more instructions contained in the main memory [406]. Such instructions may be read into the main memory [406] from another storage medium, such as the storage device

[410]. Execution of the sequences of instructions contained in the main memory [406] causes the processor [404] to perform the process steps described herein. In alternative embodiments, hard-wired circuitry may be used in place of or in combination with software instructions.

[0090] The computing device [400] also may include a communication interface [418] coupled to the bus [402]. The communication interface [418] provides a two-way data communication coupling to a network link [420] that is connected to a local network [422]. For example, the communication interface [418] may be an integrated services digital network (ISDN) card, cable modem, satellite modem, or a modem to provide a data communication connection to a corresponding type of telephone line. As another example, the communication interface [418] may be a local area network (LAN) card to provide a data communication connection to a compatible LAN. Wireless links may also be implemented. In any such implementation, the communication interface [418] sends and receives electrical, electromagnetic, or optical signals that carry digital data streams representing various types of information.

[0091] The computing device [400] can send messages and receive data, including program code, through the network(s), the network link [420] and the communication interface [418]. In the Internet example, a server [430] might transmit a requested code for an application program through the Internet [428], the ISP [426], the local network [422], host [424] and the communication interface [418]. The received code may be executed by the processor [404] as it is received, and/or stored in the storage device [410], or other non-volatile storage for later execution.

[0092] The computing device [400] encompasses a wide range of electronic devices capable of processing data and performing computations. Examples of computing device [400] include, but are not limited only to, personal computers, laptops, tablets, smartphones, servers, and embedded systems. The devices may operate independently or as part of a network and can perform a variety of tasks such as data storage, retrieval, and analysis. Additionally, computing device [400] may include peripheral devices, such as monitors, keyboards, and printers, as well as integrated components within larger electronic systems, showcasing their versatility in various technological applications.

[0093] As is evident from the above, the present disclosure provides a technically advanced solution for method and system for optimal allocation of resources for executing KPI request. To identify and allocate the most optimal resources for each KPI request execution, Machine Learning

models are leveraged to analyse the metadata of the network function over the predefined number of days, i.e., we determine the load usage patterns over the predefined number of days. The exemplary predefined number of days can be 30 days or 50 days or the like. By examining the metadata using ML models, the present invention determines the optimal resources needed to efficiently serve each request. This enables the system and the method, disclosed by the present disclosure, to allocate the appropriate resources to the application, ensuring efficient handling of the request. The invention disclosed by the present disclosure offers several advantages, which enumerated as follows:

[0094] Resource scaling/identification: For each KPI request execution, the system and the method disclosed by the present disclosure identifies optimal amount of resources (CPU, RAM, and disc space) and scales the allocated resources accordingly to server the request.

[0095] Improved Performance: By optimally scaling resources, the system can handle more concurrent requests or larger datasets, resulting in an improved response time.

[0096] Efficient Resource Utilisation: By optimally scaling resources, the system and the method disclosed by the present disclosure ensures that computing resources do not stay idle during period of high demand as well as more resources can be added from an available pool/cluster to match the workload requirements. This efficient resource utilisation leads to cost savings by minimising wasted resources and maximising the utilisation of available computing power.

[0097] The system and the method disclosed by the present disclosure facilitates automatic scaling of resources for each user request in distributed cluster computing. This means tasks can be completed faster and more efficiently.

[0098] It is emphasized that the advantages of the system and the method disclosed by the present disclosure are not limited to the above list and other advantages are possible, which would be obvious to a person skilled in the art upon reading of the present disclosure.

[0099] According to yet another aspect of the present disclosure, a non-transitory computer-readable storage medium storing instructions for managing connection between two or more peer network entities is disclosed. The instructions include executable code which, when executed by a processor, may cause the processor to receive, from a graphical user interface (GUI) module

[204], a request for optimal allocation of resources for executing the one or more KPI computation requests; receive, from a storage unit [210], a corresponding metadata related to each of the one or more KPI computation requests; analyse the received corresponding metadata related to the each of the one or more KPI computation requests; determine an optimal allocation of resources for executing the each of the one or more KPI computation requests; perform a scaling operation on a default set of resources based on the optimal allocation of resources received from DCE module [206], to generate a final allocation of resources for executing the each of the one or more KPI computation requests; and execute the one or more KPI computation requests, based on the final allocation of resources.

[0100] Further, in accordance with the present disclosure, it is to be acknowledged that the functionality described for the various the components/units can be implemented interchangeably. While specific embodiments may disclose a particular functionality of these units for clarity, it is recognized that various configurations and combinations thereof are within the scope of the disclosure. The functionality of specific units as disclosed in the disclosure should not be construed as limiting the scope of the present disclosure. Consequently, alternative arrangements and substitutions of units, provided they achieve the intended functionality described herein, are considered to be encompassed within the scope of the present disclosure.

[0101] While considerable emphasis has been placed herein on the disclosed embodiments, it will be appreciated that many embodiments can be made and that many changes can be made to the embodiments without departing from the principles of the present disclosure. These and other changes in the embodiments of the present disclosure will be apparent to those skilled in the art, whereby it is to be understood that the foregoing descriptive matter to be implemented is illustrative and non-limiting.

We Claim:

1. A method for optimal allocation of resources for executing one or more key performance indicators (KPI) computation requests, the method comprising:
   - receiving, by a transceiver unit [206a] at a distributed computation engine (DCE) module [206] from a graphical user interface (GUI) module [204], a request for optimal allocation of resources for executing the one or more KPI computation requests;
   - receiving, by the transceiver unit [206a] at the DCE module [206] from a storage unit [210], a corresponding metadata related to each of the one or more KPI computation requests;
   - analysing, by an analysis unit [206b] at the DCE module [206], the received corresponding metadata related to the each of the one or more KPI computation requests;
   - determining, by the analysis unit [206b] at the DCE module [206], an optimal allocation of resources for executing the each of the one or more KPI computation requests;
   - sending, by the transceiver unit [206a] at the DCE module [206] to a distributed compute cluster (DCC) module [208], the optimal allocation of resources for the each of the one or more KPI computation requests;
   - performing, by an analysis unit [208b] at the DCC module [208], a scaling operation on a default set of resources based on the optimal allocation of resources received from the DCE module [206], to generate a final allocation of resources for executing the each of the one or more KPI computation requests; and
   - executing, by an execution unit [206c] at the DCE module [206], the one or more KPI computation requests, based on the final allocation of resources.

2. The method as claimed in claim 1, wherein the analysing, by the analysis unit [206b] at the DCE module [206], the received metadata related to the each of the one or more KPI computation requests, is based on applying one or more machine learning (ML) based techniques.

3. The method as claimed in claim 2, wherein the analysis performed based on the on one or more machine learning (ML) techniques, is for a pre-defined period of time.

4. The method as claimed in claim 1, wherein the scaling operation comprises at least one of an up-scaling operation and a downscaling operation, wherein the up-scaling operation refers to an addition of resources to the default set of resources, and the downscaling operation refers to a removal of resources from the default set of resources.

5. The method as claimed in claim 1, the method further comprising:

- sending, by the transceiver unit [206a] at the DCE module [206] to the GUI module [204], a response related to executing the one or more KPI computation requests, based on the final allocation of resources, for one or more users; and
- displaying, by the GUI module [104], the response related to executing the one or more KPI computation requests.

6. The method as claimed in claim 1, wherein prior to the receiving, by the transceiver unit [206a] at the DCE module [206] from the GUI module [204], the request for optimal allocation of resources for executing the one or more KPI computation requests, the method comprises:

- receiving, by a transceiver unit [204a] at the GUI module [204], a creation of the request for optimal allocation of resources for executing the one or more KPI computation requests, by manual inputs of one or more users.

7. A system [200] for optimal allocation of resources for executing one or more key performance indicators (KPI) computation requests, the system [200] comprising:

- a distributed computation engine (DCE) module [206] comprising:
    - a transceiver unit [206a] configured to:
        - receive, from a graphical user interface (GUI) module [204], a request for optimal allocation of resources for executing the one or more KPI computation requests;
        - receive, from a storage unit [210], a corresponding metadata related to each of the one or more KPI computation requests;
    - an analysis unit [206b] connected at least to the transceiver unit [206a], the analysis unit [206b] being configured to:
        - analyse the received corresponding metadata related to the each of the one or more KPI computation requests;

- determine an optimal allocation of resources for executing the each of the one or more KPI computation requests;

the transceiver unit [206a] further configured to send, to a distributed compute cluster (DCC) module [208], the optimal allocation of resources for the each of the one or more KPI computation requests;

- the DCC module [208] comprising:
  - an analysis unit [208b] configured to perform a scaling operation on a default set of resources based on the optimal allocation of resources received from DCE module [206], to generate a final allocation of resources for executing the each of the one or more KPI computation requests; and
- the DCE module [206] further comprising an execution unit [206c] configured to execute the one or more KPI computation requests, based on the final allocation of resources.

8. The system [200] as claimed in claim 7, wherein the analysis unit [206b] configured to analyse, at the DCE module [206], the received metadata related to the each of the one or more KPI computation requests, is based on applying one or more machine learning (ML) based techniques.

9. The system [200] as claimed in claim 8, wherein the analysis performed based on the on one or more machine learning (ML) techniques, is for a pre-defined period of time.

10. The system [200] as claimed in claim 7, wherein the scaling operation comprises at least one of an up-scaling operation and a downscaling operation, wherein the up-scaling operation refers to an addition of resources to the default set of resources, and the downscaling operation refers to a removal of resources from the default set of resources.

11. The system [200] as claimed in claim 7, the system [200] further comprising:

- the transceiver unit [206a] of the DCE module [206] further configured to send to the GUI module [204], a response related to executing the one or more KPI computation requests, based on the final allocation of resources; and
- the GUI module [204] configured to display the response related to executing the one or more KPI computation requests.

12. The system [200] as claimed in claim 7, wherein prior to the DCE module [206] receiving, from the GUI module [204], the request for optimal allocation of resources for executing the one or more KPI computation requests, a transceiver unit [204a] at the GUI module [204] is configured to receive a creation of the request for optimal allocation of resources for executing one or more KPI computation requests, by manual inputs of one or more users.

13. A non-transitory computer-readable storage medium storing instruction for optimal allocation of resources for executing one or more key performance indicators (KPI) computation requests, the storage medium comprising executable code which, when executed by one or more units of a system, causes:

- a distributed computation engine (DCE) module [206] comprising:
    - a transceiver unit [206a] to:
        - receive, from a graphical user interface (GUI) module [204], a request for optimal allocation of resources for executing the one or more KPI computation requests;
        - receive, from a storage unit [210], a corresponding metadata related to each of the one or more KPI computation requests;
    - an analysis unit [206b] connected at least to the transceiver unit [206a], the analysis unit [206b] being to:
        - analyse the received corresponding metadata related to the each of the one or more KPI computation requests;
        - determine an optimal allocation of resources for executing the each of the one or more KPI computation requests;

    the transceiver unit [206a] further to send, to a distributed compute cluster (DCC) module [208], the optimal allocation of resources for the each of the one or more KPI computation requests;
- the DCC module [208] comprising:
    - an analysis unit [208b] to perform a scaling operation on a default set of resources based on the optimal allocation of resources received from DCE module [206], to generate a final allocation of resources for executing the each of the one or more KPI computation requests; and
- the DCE module [206] further comprising an execution unit [206c] to execute the one or more KPI computation requests, based on the final allocation of resources.

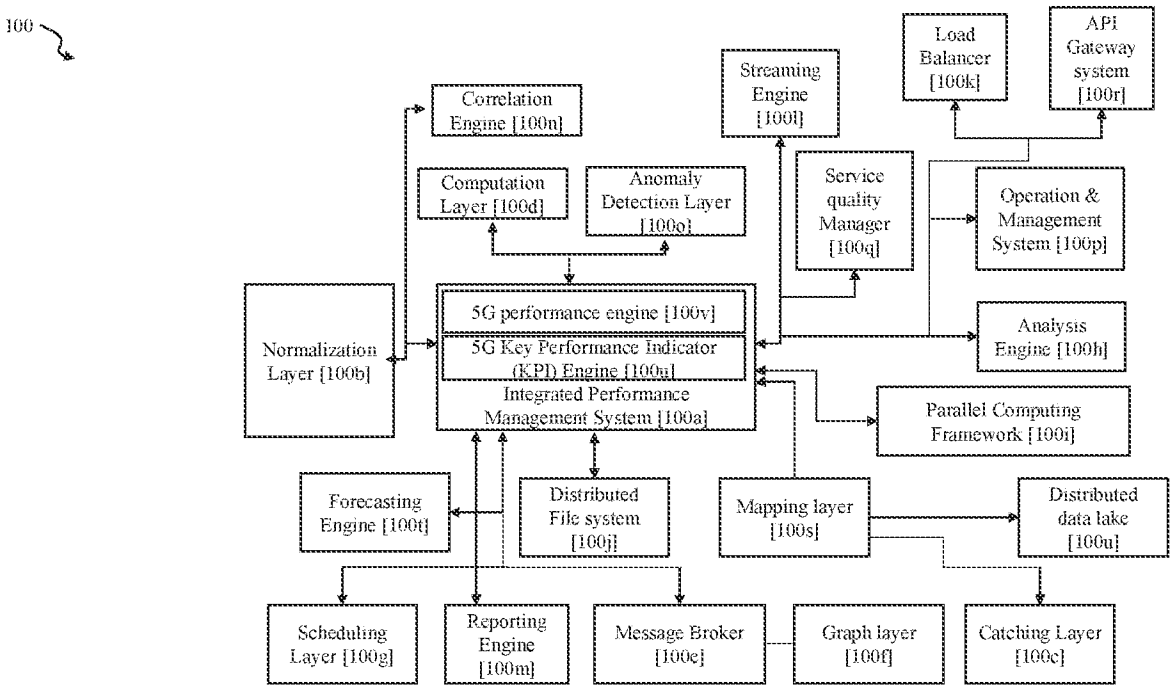

100
Load Balancer [100k]
API Gateway system [100r]
Streaming Engine [100l]
Correlation Engine [100n]
Computation Layer [100d]
Anomaly Detection Layer [100o]
Service quality Manager [100q]
Operation & Management System [100p]
Normalization Layer [100b]
5G performance engine [100v]
5G Key Performance Indicator (KPI) Engine [100u]
Integrated Performance Management System [100a]
Analysis Engine [100h]
Parallel Computing Framework [100i]
Forecasting Engine [100t]
Distributed File system [100j]
Mapping layer [100s]
Distributed data lake [100u]
Scheduling Layer [100g]
Reporting Engine [100m]
Message Broker [100e]
Graph layer [100f]
Catching Layer [100c]

FIG. 1

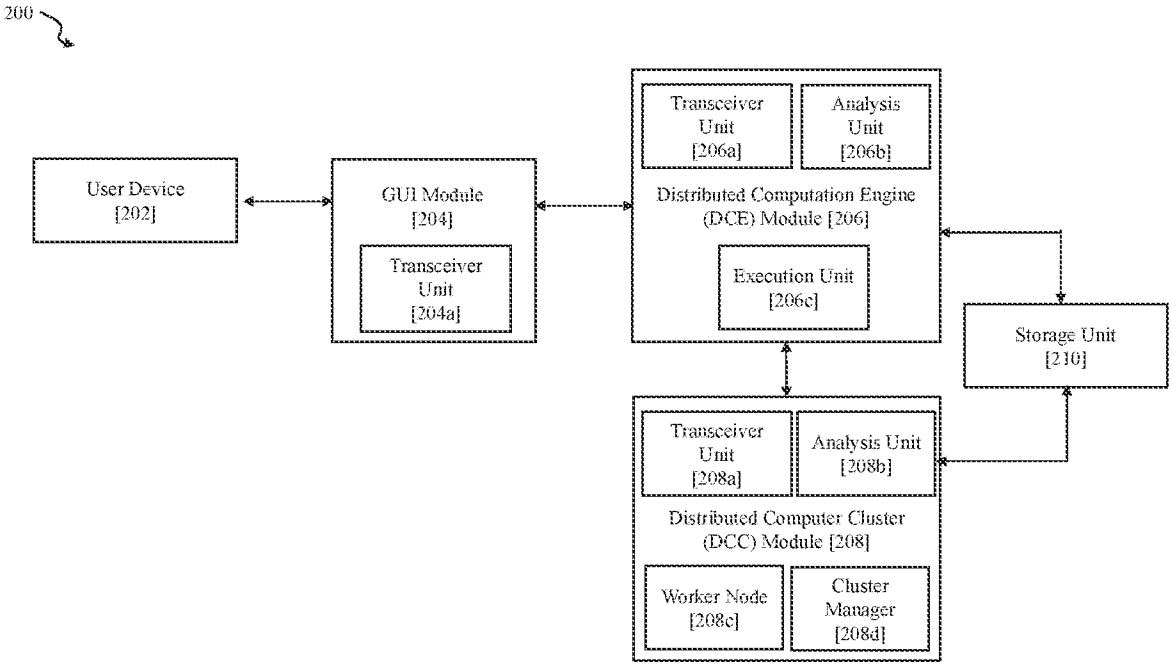

200
User Device
[202]
GUI Module
[204]
Transceiver
Unit
[204a]
Transceiver
Unit
[206a]
Analysis
Unit
[206b]
Distributed Computation Engine
(DCE) Module [206]
Execution Unit
[206c]
Storage Unit
[210]
Transceiver
Unit
[208a]
Analysis Unit
[208b]
Distributed Computer Cluster
(DCC) Module [208]
Worker Node
[208c]
Cluster
Manager
[208d]

FIG. 2

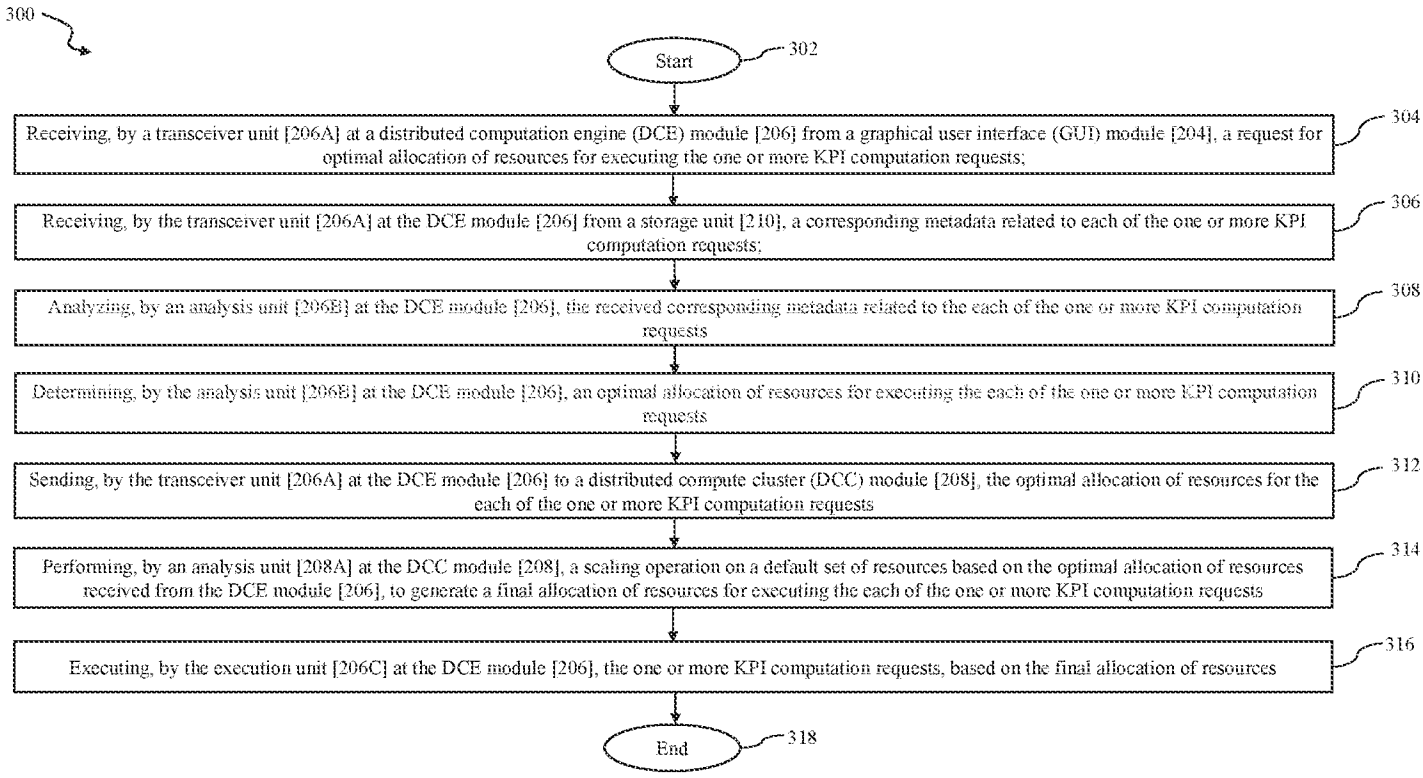

300
Start
302
Receiving, by a transceiver unit [206A] at a distributed computation engine (DCE) module [206] from a graphical user interface (GUI) module [204], a request for optimal allocation of resources for executing the one or more KPI computation requests;
304
Receiving, by the transceiver unit [206A] at the DCE module [206] from a storage unit [210], a corresponding metadata related to each of the one or more KPI computation requests;
306
Analyzing, by an analysis unit [206B] at the DCE module [206], the received corresponding metadata related to the each of the one or more KPI computation requests
308
Determining, by the analysis unit [206B] at the DCE module [206], an optimal allocation of resources for executing the each of the one or more KPI computation requests
310
Sending, by the transceiver unit [206A] at the DCE module [206] to a distributed compute cluster (DCC) module [208], the optimal allocation of resources for the each of the one or more KPI computation requests
312
Performing, by an analysis unit [208A] at the DCC module [208], a scaling operation on a default set of resources based on the optimal allocation of resources received from the DCE module [206], to generate a final allocation of resources for executing the each of the one or more KPI computation requests
314
Executing, by the execution unit [206C] at the DCE module [206], the one or more KPI computation requests, based on the final allocation of resources
316
End
318

FIG. 3

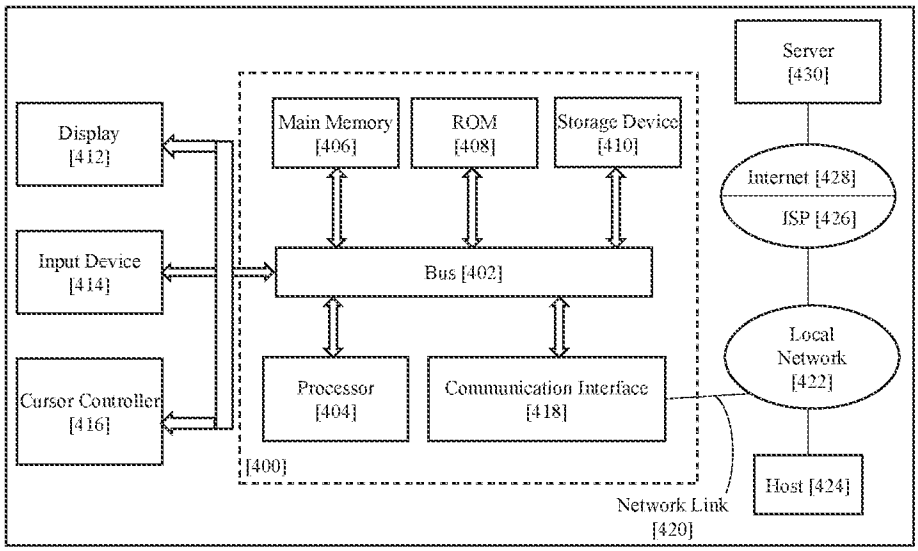

Server
[430]
Display
[412]
Main Memory
[406]
ROM
[408]
Storage Device
[410]
Internet [428]
ISP [426]
Input Device
[414]
Bus [402]
Local
Network
[422]
Cursor Controller
[416]
Processor
[404]
Communication Interface
[418]
[400]
Host [424]
Network Link
[420]

FIG. 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IN2024/051204

A. CLASSIFICATION OF SUBJECT MATTER
H04W72/50,G06F9/50,G06F9/44 Version=2024.01

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04W, G06F

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

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

PatSeer, IPO Internal Database

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	WO2021164857A1 (TELEFONAKTIEBOLAGET LM ERICSSON (PUBL)) 26 AUGUST 2021(26/08/2021) Paragraphs [0006], [0014], [0021], [0046]	1-13
Y	US11061794B1 (EMC IP HOLDING CO LLC) 13 JULY 2021 (13/07/2021) Column 4 lines 27-34, Column 5 lines 1-3, Column 9 lines 22-28, Column 8 lines 48-54, Column 10 lines 3-7	1-13

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"D" document cited by the applicant in the international application
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
13-11-2024	13-11-2024
Name and mailing address of the ISA/ Indian Patent Office Plot No.32, Sector 14,Dwarka,New Delhi-110075 Facsimile No.	Authorized officer Md Shahnawaz Alam Telephone No. +91-1125300200