MINISTERO DELLO SVILUPPO ECONOMICO
DIREZIONE GENERALE PER LA LOTTA ALLA CONTRAFFAZIONE
UFFICIO ITALIANO BREVETTI E MARCHI

UIBM

DOMANDA DI INVENZIONE NUMERO	**102021000029084**
Data Deposito	**17/11/2021**
Data Pubblicazione	**17/05/2023**

Classifiche IPC

Sezione	Classe	Sottoclasse	Gruppo	Sottogruppo
G	06	F	9	50

Titolo

Metodo e sistema per il calcolo distribuito ad alto rendimento di lavori computazionali

Titolo: "Metodo e sistema per il calcolo distribuito ad elevato throughput di job computazionali"

DESCRIZIONE

Campo di applicazione

La presente invenzione si inserisce nel settore del calcolo distribuito per l'elaborazione di lavori, o job, computazionali.

Sfondo dell'invenzione

Nell'era dell'internet delle cose (Internet of Things, IoT) e dei Big Data, i file system distribuiti e l'elaborazione in cloud, o cloud computing, sono elementi fondamentali per la gestione e l'elaborazione dei dati. I server per cloud mondiali di proprietà delle più grandi aziende tecnologiche sono la risorsa più preziosa su cui possiamo contare oggi per eseguire calcoli distribuiti in outsourcing; l'elaborazione in edge, o edge computing, è divenuta indispensabile poiché cresce sempre più il volume di dati gestiti dalle aziende per qualsivoglia scopo.

Numerosi anni di progressi tecnologici hanno spianato la strada al cloud computing verso l'Industria 4,0, rendendo possibile la realizzazione di un'ampia gamma di soluzioni cloud e modificando in modo irreversibile il nostro modo di guardare le cose. Di conseguenza, negli ultimi quindici anni sono nate numerose nuove aziende che operano nel settore del cloud, come Snowflake, Cloudflare, Databricks e i noti leader del settore tech Google, Microsoft, Amazon, IBM devono la loro posizione di giganti

dell'informatica anche all'introduzione del cloud.

Il cloud computing consente l'elaborazione efficiente e flessibile di enormi carichi di lavoro tramite uno o più server in outsourcing. In particolare, il campo del calcolo distribuito studia il coordinamento di unità di calcolo collegate in rete poste in diverse posizioni, che possono eseguire congiuntamente attività di calcolo, o task, disparate. Infine, il paradigma del calcolo in griglia, o grid computing, estende il concetto di calcolo distribuito ammettendo eterogeneità nella composizione dei computer in rete e considerando che un grosso job computazionale originario può essere suddiviso in singoli task, distribuiti nella rete.

Esistono oggi alcuni progetti che si propongono di distribuire job computazionali su una grid di dispositivi comuni, in particolare di tipo desktop, in un quadro di grid computing. Questi hanno lo scopo di sfruttare la notevole insita potenza computazionale diffusa che si trova all'interno dei dispositivi utente comuni. Tale potenza rimane solitamente inutilizzata durante l'inattività dei proprietari umani, ad esempio di notte.

L'attuazione di calcoli distribuiti su dispositivi esistenti e comunemente attivi non è solo un'implementazione del paradigma del grid computing, ma anche un'alternativa ecologica alla costruzione di infrastrutture di cloud computing come i data center, in quanto garantisce calcoli altamente parallelizzati e a bassa intensità e realizza dispersione del calore, senza richiedere quindi ulteriore raffreddamento diverso da quello fornito dall'ambiente.

Molte soluzioni operative note appartengono alla categoria del calcolo su base volontaria, o Volunteer Computing, nel quale gli utenti mettono a disposizione i propri

dispositivi per l'hosting di calcoli intensivi esterni su base volontaria. Esempi di progetti di calcolo distribuito sono BOINC, Distributed.net, HTCondor e Folding@Home.

BOINC è una piattaforma per il calcolo distribuito ad elevata intensità, o High-Throughput computing, in cui i nodi di lavoro sono computer desktop e portatili, tablet e smartphone offerti volontariamente dai loro proprietari. Numerosi progetti sono collegati a BOINC e usano la relativa infrastruttura distribuita. Ad esempio: SETI@Home per l'elaborazione di segnali digitali dei dati di radiotelescopi; Einstein@Home per la ricerca di deboli segnali astrofisici provenienti da stelle di neutroni rotanti; IBM World Community Grid per la ricerca scientifica su temi legati alla sanità, alla povertà e alla sostenibilità; Climateprediction.net per simulazioni di modelli climatici.

HTCondor è un altro software open source per il calcolo distribuito capace di aumentare il throughput di calcolo, sviluppato presso l'Università del Wisconsin-Madison. HTCondor offre un meccanismo di accodamento di lavori, o job queuing, criteri di scheduling, uno schema di priorità, monitoraggio di risorse e gestione di risorse e può integrare sia risorse dedicate (cluster montati su rack) sia macchine desktop non dedicate (grazie al Volunteer Computing) in un unico ambiente informatico.

Sommario dell'invenzione

Uno scopo della presente invenzione è quello di realizzare un ambiente innovativo di calcolo distribuito ad elevato throughput, atto a ricevere e risolvere diversi job computazionali.

Un altro scopo dell'invenzione è quello di offrire un'alternativa sostenibile

all'incremento del consumo di risorse da parte del cloud computing.

Un altro scopo ancora dell'invenzione è quello di diffondere il calcolo distribuito su un elevato numero di dispositivi utente, come personal computer, smartphone, tablet, console di gioco e smart TV, mantenendo un elevato livello di efficienza.

Questo ed altri scopi vengono conseguiti grazie ad un metodo e ad un sistema di calcolo distribuito per l'elaborazione di job computazionali in accordo con una qualunque delle annesse rivendicazioni.

L'invenzione fornisce una piattaforma clienti in cui un'entità cliente può caricare specifiche di job e quindi il relativo job computazionale, eseguibili da una grid di dispositivi utente. Un sistema software di servizi interni interroga i dispositivi utente per valutare uno stato della grid. Quindi, in base alle specifiche di job e allo stato della grid, seleziona uno schema di partizionamento del job.

Il flusso di input dei dati di job viene caricato e suddiviso in pezzi, o chunk, di input di dati, in base a parametri di suddivisione dello schema di partizionamento. I chunk di input vengono inclusi in file di attività, o task, eseguibili che vengono distribuiti ai dispositivi utente in base a parametri di distribuzione dello schema di suddivisione. Dal momento che durante la configurazione dello schema di partizionamento si tiene conto dello stato della grid, ciascun dispositivo riceverà un task opportuno per il quale è possibile prevedere un tempo di esecuzione e di invio prestabilito.

Preferibilmente, le fasi di suddivisione e distribuzione hanno luogo già durante il caricamento del flusso di input. Pertanto, non è necessario che vengano salvati

stabilmente i dati di job completi nei database che supportano il sistema servizi interni, ma è possibile semplicemente salvare temporaneamente i chunk di input, e parte di essi possono essere eliminati anche prima che venga completato il caricamento, o upload.

I dispositivi utente ricevono ed eseguono i chunk di input e ottengono chunk di output che vengono reinviati al sistema servizi interni. Realizzazioni vantaggiose prevedono soluzioni atte a compensare risposte mancanti o inutilizzabili da uno o più dispositivi utente, sia duplicando preventivamente i task perché vengano eseguiti da dispositivi distinti, sia assegnando successivamente i task non completati ad altri dispositivi di calcolo.

Infine, i chunk di output vengono assemblati per ottenere un risultato di job, accessibile al cliente dalla piattaforma.

Pertanto, l'invenzione rende il calcolo distribuito efficiente ed affidabile, con rischi minimi di ritardo di consegna del risultato del job. Le specifiche di job per diversi job consentono di assegnare priorità al calcolo a determinati job, con un'opportuna distribuzione dei task a dispositivi utente con prestazioni più o meno elevate.

Gli utenti che offrono la propria capacità di calcolo possono essere ricompensati con crediti o sconti, per motivare la partecipazione. Grazie al fatto che il calcolo non è concentrato nei server centrali, si ha un notevole incremento della dissipazione del calore, poiché ogni dispositivo utente è naturalmente raffreddato dall'ambiente. Si evita così di dover prevedere ingombranti sistemi di raffreddamento, il che consente di conseguire notevoli miglioramenti dal punto di vista della sostenibilità economica e ambientale.

Breve descrizione dei disegni

La presente invenzione verrà ora descritta in maggior dettaglio con riferimento alle tavole di disegno annesse, in cui sono illustrate alcune realizzazioni dell'invenzione.

La FIG. 1 è un diagramma schematico di un sistema di calcolo distribuito per l'elaborazione di job computazionali in accordo con una realizzazione dell'invenzione,

le FIGG. 2 e 3 sono rappresentazioni schematiche di diverse fasi dei processi per suddividere dati di job in chunk di input, distribuire task a dispositivi utente, ricevere chunk di output, e assemblarli come risultato di job, di un metodo di calcolo distribuito per l'elaborazione di job computazionali in accordo con una realizzazione dell'invenzione,

le FIGG. 4 e 5 sono rappresentazioni schematiche di fasi di preselezione e raggruppamento per l'assegnazione di task a dispositivi utente, secondo il metodo delle figure 2 e 3,

la FIG. 6 è una rappresentazione schematica dell'attribuzione ai cluster di dimensioni di chunk di cluster e numeri di chunk di cluster, come parametri per la suddivisione di dati di job in chunk di input, secondo il metodo delle figure 2 e 3, e

la FIG. 7 è una rappresentazione schematica della generazione di code di task di cluster e di code di task di dispositivo individuali, per assegnare task ai dispositivi utente, secondo il metodo delle figure 2 e 3.

Descrizione dettagliata

Sono descritti un metodo di calcolo distribuito per l'elaborazione di job computazionali, e un sistema elettronico 100 configurato per eseguire il metodo.

Il sistema 100 comprende un software lato cliente 110, fornito ad entità cliente 200. Le entità cliente 200 sono soggetti interessati a far elaborare al sistema 100 uno o più job computazionali 400, per ottenere rispettivi risultati di job 500.

Il software lato cliente 110 include una piattaforma informatica cliente 111 per l'upload di job computazionali 400 da parte delle entità cliente 200, e per altre funzioni specificate di seguito. Più in dettaglio, il software lato cliente 110 comprende una pluralità di servizi software cliente 112, 113, 114 configurati per eseguire singole funzioni relative alla piattaforma.

Ciascun job computazionale 400 è composto da dati di job con una certa dimensione di job. I dati di job possono essere immessi nella piattaforma cliente 111 in forma di flusso di dati di input. A tal fine, il software lato cliente 110 include un servizio di upload di job 112.

Comunemente, le entità cliente 200 forniscono job computazionali 400 con dimensioni di job relativamente elevate, e di conseguenza la ricezione dei singoli flussi di input richiede un intervallo di tempo di upload non trascurabile.

Per ragioni che risulteranno chiare di seguito, l'upload del flusso di input è preceduto dall'immissione nella piattaforma clienti 111 di specifiche di job, sempre dalle entità cliente 200, le specifiche di job essendo caricate dal servizio di upload di job 112. La specifica di job di un dato job computazionale 400 include parametri che caratterizzano il rispettivo job computazionale 400. Esempi preferiti di tali parametri

sono la dimensione del job, il formato di input e/o output del job, il linguaggio di programmazione, l'esecuzione mediante algoritmi di libreria o algoritmi personalizzati, l'oggetto del job (come operazioni multimediali, elaborazione di dati tabulari, operazioni di testo o web testing), il livello di priorità di esecuzione e le limitazioni geografiche nella posizione di esecuzione.

Il sistema 100 comprende anche un sistema di memorizzazione dati 120, comprendente uno o più database 121, 122, 123, 124, 125, in posizioni uguali o diverse, alcuni di essi essendo database in cloud in alcune realizzazioni. Il sistema di memorizzazione dati 120 è gestito dal proprietario del sistema elettronico, e non è inteso per essere accessibile alle entità cliente 200 o ad entità utente 300, come descritte di seguito. Il servizio di upload di job 112 è configurato per effettuare l'upload del flusso di input e delle specifiche di job nel sistema di memorizzazione dati 120.

Un sistema software di servizi interni 130 viene eseguito in uno o più database 121, 122, 123, 124, 125 del sistema di memorizzazione dati 120, per ricevere e gestire i dati di job ricevuti attraverso la piattaforma clienti 111, e per svolgere altre funzioni illustrate di seguito. Il sistema servizi interni 130 può comprendere una pluralità di servizi interni 131, 132, 133, 134, 135, 136, 137 con algoritmi che eseguono funzioni diverse, tra le funzioni che verranno descritte di seguito.

Il sistema 100 comprende un software lato utente 140, fornito alle entità utente 300. Le entità utente 300 sono soggetti che possiedono rispettivi dispositivi utente 310, adatti all'installazione e all'esecuzione del software lato utente 140, e che sono interessati ad offrire potenza computazionale. Tra gli esempi di dispositivi utente 310 sono inclusi

personal computer, smartphone, tablet, console di gioco e smart TV. Preferibilmente, la potenza computazionale viene offerta in cambio di denaro, servizi o altri crediti secondo un tariffario prestabilito.

Il software lato utente 140 può comprendere un'applicazione di dispositivo 141 e più servizi utente 142 134, 142, 143, con algoritmi che eseguono funzioni diverse, tra le funzioni che verranno descritte di seguito. L'applicazione di dispositivo 141 è configurata per fornire un'interfaccia grafica utente, e per attivare i servizi utente 142, 143, 144, ove necessario.

I dispositivi utente 310 sono remoti dal sistema di memorizzazione 120. Quando i dispositivi utente 310 eseguono il software lato utente 140, essi sono almeno periodicamente in comunicazione di segnale con il sistema di memorizzazione 120. Così, i dispositivi utente 310 formano collettivamente una griglia, o grid, in comunicazione di segnale con il sistema di memorizzazione 120. La grid viene impostata con l'aggiunta progressiva di dispositivi utente 310.

Il sistema servizi interni 130 è configurato per interrogare con query e ricevere periodicamente risposte dai dispositivi utente 310. In particolare, queste funzioni vengono svolte da un database di gestione grid 121 del sistema di memorizzazione 120, preferibilmente un database Firebase, dove vengono quindi memorizzate le risposte provenienti dai dispositivi utenti 310.

Più in dettaglio, i dispositivi utente 310 vengono interrogati circa un rispettivo stato di dispositivo. Lo stato di dispositivo può essere espresso in termini di parametri come parametri di capacità di calcolo, caratterizzanti le massime prestazioni potenziali

del dispositivo utente 310, e parametri di disponibilità, caratterizzanti i limiti correnti e/o storici che impediscono l'uso delle massime prestazioni potenziali.

Esempi di parametri di capacità di calcolo sono i valori di RAM, CPU e memoria installate. I parametri di disponibilità includono lo stato di potenza del dispositivo utente 310 (acceso/spento), e valori di RAM, CPU, memoria e larghezza di banda attualmente disponibili, perché non altrimenti in uso dal dispositivo utente 310. Altri parametri di disponibilità includono valori di soglia di utilizzo per RAM, CPU, memoria, larghezza di banda e soglie di tempo. I valori di soglia d'uso possono essere impostati dalle entità utente 300 in modo che la capacità di calcolo utilizzata dal software lato utente non copra l'intera capacità di calcolo disponibile. Le soglie di tempo sono una o più finestre temporali di disponibilità impostabili dall'entità utente 300, eventualmente con valori di soglia d'uso diversi per finestre temporali di disponibilità diverse.

Altri esempi di parametri di disponibilità sono la batteria rimanente, la temperatura interna, l'attività corrente del dispositivo utente 310, la posizione del dispositivo utente 310 e dati accelerometrici.

Altri parametri di disponibilità si riferiscono all'efficienza della comunicazione, o sull'altro lato alle limitazioni d'uso della capacità di calcolo dovute ad inefficienze di comunicazione, come tipo di rete, velocità di rete, latenza di rete e indirizzo IP.

Un altro parametro ancora è correlato a dati storici sulle effettive prestazioni di calcolo dei dispositivi utente 310. Le prestazioni vengono valutate in base all'esecuzione di task, ossia pacchetti di file eseguibili e chunk di dati. Come descritto di seguito, ai dispositivi utente 310 vengono assegnati task, generati da chunk di dati di input, al fine

di ottenere parti dei risultati di job 500. I task distribuiti a dispositivi utente 310 diversi saranno generalmente diversi. Così, in una realizzazione, i dati storici includono le prestazioni del dispositivo utente 310 su uno o più di tali task recenti, ad esempio rappresentati da un tempo di esecuzione e da un completamento dell'esecuzione.

In altre realizzazioni, le prestazioni vengono valutate sulla base dell'esecuzione di uno o più recenti task di prova prestabiliti. Il task di prova può essere inviato dal sistema di memorizzazione 120 al dispositivo utente 310 con la query di stato, senza alcuna funzione di ottenimento del risultato di job 500, ma solo al fine di testare la capacità di calcolo e la disponibilità del dispositivo utente 310, nonché la sua precisione di calcolo. Il task di prova può essere lo stesso per tutti i dispositivi utente 310, in modo che le relative prove prestazionali siano comparabili.

Il software lato utente 140 è configurato per ricevere le query di stato dal sistema di memorizzazione 120, per valutare almeno alcuni dei parametri di capacità di calcolo e disponibilità del dispositivo utente 310, e per generare e inviare una risposta di stato che include i parametri di capacità di calcolo e di disponibilità valutati.

Nelle realizzazioni preferite, le query di stato includono prime query di stato più frequenti, e seconde query di stato meno frequenti. Le prime query di stato sono denominate anche query di tipo heartbeat, che richiedono al dispositivo utente 310 soltanto una risposta disponibile/non disponibile. Le seconde query di stato possono includere richieste relative ad una parte o alla totalità dei parametri sopra descritti, ossia una quantità di parametri maggiore di quelli richiesti per la query heartbeat.

La valutazione della capacità di calcolo e della disponibilità può includere

l'esecuzione del task di prova, per ottenere un chunk di dati di output di prova. Ciò consente al sistema interno di verificare un tempo di prova di esecuzione e di consegna del task di prova, e un'integrità o correttezza del chunk di output di prova, ossia la corrispondenza tra il chunk di output di prova e un prestabilito chunk di risultato previsto per il task di prova.

Il sistema di memorizzazione 120 dovrebbe teoricamente ricevere le risposte di stato da tutti i dispositivi utente 310 interrogati. Tuttavia, nelle applicazioni reali, le risposte di stato vengono ricevute soltanto da alcuni dei dispositivi utente interrogati 310. Infatti, possono mancare risposte di stato da dispositivi utente spenti 310, da dispositivi utente offline 310, e da dispositivi utente 310 che per qualsivoglia ragione non riescono a ricevere la query di stato, a valutare lo stato del dispositivo, o ad inviare la risposta di stato.

Il sistema servizi interni 130 comprende un servizio di gestione grid 131, che è configurato per definire uno stato di grid in funzione almeno delle risposte di stato ricevute, memorizzate nel database di gestione grid 121, preferibilmente in funzione delle risposte di stato ricevute, dei tempi di risposta delle risposte di stato ricevute, di eventuali risposte di stato mancanti, e dell'integrità di qualunque chunk di output di prova. Giova quindi rilevare che alcuni dei parametri di capacità di calcolo e disponibilità possono non essere determinati dal software lato utente 140, ma dal sistema servizi interni 130, in base alle risposte di stato ricevute/mancanti, dei tempi di risposta e dell'integrità dei chunk di output di test.

Preferibilmente, definire lo stato di grid comprende elaborare almeno alcuni dei

parametri di capacità di calcolo e disponibilità di ciascuno stato di dispositivo per ottenere per ciascun dispositivo utente 310 uno o più punteggi di dispositivo 150. Pertanto, definire lo stato di grid comprende ordinare i dispositivi utente in uno o più indici di grid 160 in base a rispettivi punteggi di dispositivo 150. Punteggi di dispositivo distinti 150 possono essere basati su gruppi distinti di parametri di capacità di calcolo e di disponibilità.

Un primo punteggio di dispositivo preferito 150 è un punteggio di capacità di calcolo. Il punteggio di capacità può essere una funzione crescente dei valori di disponibilità di RAM, CPU e memoria e una funzione decrescente con l'instabilità della connessione, la batteria scarica, la temperatura interna elevata e l'attività intensa del dispositivo.

Un secondo punteggio di dispositivo preferito 150 è un punteggio di tasso di guasto. Il punteggio del tasso di guasto può essere funzione dei dati storici di guasto dei dispositivi utente 310. Tra i dati storici di guasto possono essere inclusi eventi passati di mancata risposta ad un task, ritardi nella risposta ad un task e nella risposta con risultati danneggiati o errati per un task.

Un terzo punteggio di dispositivo preferito 150 è un punteggio di bilanciamento, che fornisce punteggi statisticamente più alti ai dispositivi utente 310 storicamente meno utilizzati. Il punteggio di bilanciamento può essere ottenuto ad esempio moltiplicando un numero generato casualmente con l'intervallo di tempo intercorso dall'ultima assegnazione di un task al dispositivo utente.

Un quarto punteggio preferito 150 è un punteggio complessivo, che è una

combinazione di due o più, preferibilmente della totalità degli altri punteggi di dispositivo 150, compresi i tre sopra descritti e qualunque altro ulteriore punteggio di dispositivo 150 possa essere previsto da un esperto del settore. Preferibilmente, il punteggio complessivo è una funzione crescente dei punteggi di capacità e bilanciamento, e una funzione decrescente con il punteggio di tasso di guasto.

Come i dispositivi utente 310 vengono periodicamente interrogati sul loro stato, i punteggi di dispositivo 150 e gli indici di grid 160 vengono periodicamente aggiornati e possono variare da una query all'altra. Aggiornamenti frequenti sono utili per evitare problemi come uno stato offline di un dispositivo utente 310 mentre è registrato nello stato di grid come online, e per impedire così l'assegnazione di task a dispositivi utente non disponibili 310.

Secondo un aspetto dell'invenzione, il sistema servizi interni è configurato per selezionare per ciascun job computazionale 400 un rispettivo schema di partizionamento job, preferibilmente diverso per diversi job computazionali 400. Lo schema di partizionamento job include parametri almeno per la suddivisione dei dati di job e la loro distribuzione ai dispositivi utente 310.

Lo schema di partizionamento job, unitamente ai relativi parametri, vengono calcolati in funzione dello stato di grid e della specifica di job. Pertanto, diversi stati di grid e diverse specifiche di job determineranno generalmente diversi schemi di partizionamento.

Giova rilevare che la specifica di job è sufficiente a tal fine, anche in assenza di dati di job completi disponibili. Così, poiché l'input della specifica di job è antecedente a

quella dei dati di job, lo schema di partizionamento può essere selezionato dal sistema servizi interni 130 prima che abbia inizio l'intervallo di tempo di upload o durante l'intervallo di tempo di upload, senza attendere il completamento del flusso di input completo dei dati di job.

Successivamente, il sistema servizi interni 130 è configurato per suddividere i dati di job, inclusi in ciascun flusso di input, in chunk di input 410 di dati, secondo i parametri di suddivisione dello schema di partizionamento selezionato per il job computazionale 400.

Preferibilmente, la fase di suddivisione viene eseguita in tempo reale, o on-the-fly. In altri termini, essa ha inizio durante l'intervallo di tempo di upload. Inoltre, ciascun chunk di input 410 è preferibilmente memorizzato nel sistema di memorizzazione 120 solo temporaneamente. I chunk di input 410 vengono memorizzati su un database di accodamento, o queuing, 122 del sistema di memorizzazione 120 (vale a dire, un primo database di queuing), che è preferibilmente un database Redis. Il database di queuing 123 è configurato per la memorizzazione a breve termine per supportare operazioni di lettura e scrittura frequenti.

I chunk di input 410 possono essere utilizzati e cancellati prima della fine dell'intervallo di tempo di upload, cioè prima dell'avvenuto upload dei dati di job completi attraverso la piattaforma clienti 111. Più in dettaglio, ciascun chunk di input 410 resta memorizzato da un rispettivo istante di salvataggio ad un rispettivo istante di cancellazione. L'istante di salvataggio ha luogo durante l'intervallo di tempo di upload, poiché l'istante di salvataggio dell'ultimo chunk di input 410 segna la fine dell'upload del

flusso di input. Per almeno un chunk di input 410, preferibilmente per la maggior parte dei chunk di input 410, a seconda della dimensione di job, anche l'istante di cancellazione ha luogo durante l'intervallo di tempo di upload. Così, l'uso di alcuni chunk di input 410, come descritto di seguito, può essere completato anche prima che siano stati ricevuti i dati di job completi.

Vantaggiosamente, la memoria del database di queuing 122 può non essere mai occupata dai dati di job completi di un job computazionale 400, poiché sono parzialmente cancellati prima della ricezione completa. Ciò consente di risparmiare spazio di memorizzazione.

Il sistema servizi interni 130 è configurato per generare uno o più task 420 per ciascun chunk di input 410. Come descritto più avanti in maggior dettaglio, gli uno o più task 420 per ciascun chunk di input 410 sono un unico task 420, o una pluralità di task identici 420. Ciascun task 420 comprende un rispettivo chunk di input 410, ed un file eseguibile che include istruzioni per eseguire calcoli sul chunk di input 410.

Il sistema servizi interni 130 è quindi configurato per assegnare ed inviare ciascun task 420 ad uno o più rispettivi dispositivi utente 310, per l'esecuzione del task 420 da parte dei dispositivi utente 310. Giova rilevare che alcune sotto-fasi di assegnazione possono implicare l'interazione del sistema servizi interni 130 con i dispositivi utente 310.

L'assegnazione dei task 420 comporta di selezionare i dispositivi utente 310 in base ai parametri di distribuzione dello schema di partizionamento del rispettivo job computazionale 400.

Verranno ora descritte caratteristiche preferite dello schema di partizionamento del job, per determinare i parametri di suddivisione e distribuzione, che sono correlati l'uno all'altro. Le seguenti fasi, salvo diversa indicazione, sono preferibilmente eseguite da un servizio di partizionamento 132 del sistema servizi interni 130.

In alcune realizzazioni, selezionare lo schema di suddivisione del job comprende scartare, per il job computazionale 400, un numero di dispositivi utente 310, e almeno pre-selezionare per il job computazionale 400 i dispositivi utente rimanenti 310. Dalla descrizione che segue risulterà evidente che nelle realizzazioni preferite non a tutti i dispositivi pre-selezionati 310 sarà necessariamente assegnato un task 420.

Più in dettaglio, i parametri di distribuzione dello schema di partizionamento includono uno o più intervalli target per rispettivi parametri di capacità di calcolo e disponibilità, o loro combinazioni. Ad esempio, un intervallo target può essere applicato ad uno o più punteggi di dispositivo 150 e/o indici di grid 160, che sono combinazioni dei parametri di capacità di calcolo e disponibilità, in modo che l'assegnazione sia basata sul punteggio di dispositivo 150 o su una posizione di dispositivo nell'indice di grid 160.

Così, allo scopo di assegnare i task 400 ai dispositivi utente 310, lo schema di partizionamento job prevede che il sistema servizi interni 130 scarti per il job computazionale 400 dispositivi utente incompatibili 310 e almeno pre-selezioni per il job computazionale 400 i rimanenti dispositivi utente compatibili 310, a seconda del fatto che il loro stato di dispositivo presenti parametri di capacità di calcolo e di disponibilità (singolarmente, o combinati in punteggi di dispositivo 150 e/o indici di grid 160) che sono incompatibili o compatibili con uno o più intervalli target. Qualunque

dispositivo utente 310 dal quale sia stato ricevuto un chunk di output di prova corrotto può essere automaticamente scartato come incompatibile.

Gli intervalli target vengono determinati in funzione delle specifiche di job, più preferibilmente in funzione del livello di priorità di esecuzione, cioè un livello di priorità desiderato selezionato dal cliente per il job 400. Durante l'input della specifica di job potranno essere selezionati livelli di priorità diversi, che potranno essere disponibili a prezzi diversi. Job computazionali 400 con livelli di priorità più elevati determineranno la selezione di un intervallo target relativamente elevato per il punteggio complessivo o per il punteggio di capacità computazionale, mentre livelli di priorità più bassi determineranno la selezione di un intervallo target relativamente basso.

Ulteriori intervalli target per scartare o pre-selezionare dispositivi utente 310 sono determinabili in funzione di limitazioni geografiche nella posizione di esecuzione, che possono essere selezionate in fase di input della specifica di job, ad esempio per limitazioni di sicurezza nazionali che possono risultare applicabili per particolari job computazionali.

Nella figura 4 è illustrato un esempio di pre-selezione, in cui i dispositivi utente sono ordinati per punteggio di dispositivo complessivo. Un gruppo di dispositivi utente 310, indicato complessivamente con 311, è pre-selezionato in base ad un intervallo target sul punteggio complessivo del dispositivo. Quindi, alcuni dispositivi del gruppo 311, indicati con 312, vengono scartati in base ad un altro criterio, come una limitazione geografica.

Nelle realizzazioni preferite, selezionare lo schema di suddivisione del job

comprende raggruppare i dispositivi utente 310 in cluster 320. Il raggruppare in cluster segue preferibilmente la preselezione di dispositivi utente 310 in base agli intervalli target, come sopra descritto. In altri termini, sono raggruppati in cluster 320 solo i dispositivi utente pre-selezionati 310. Potranno essere tuttavia previste altre realizzazioni in cui tutti i dispositivi utente 310 sono idonei per il raggruppamento in cluster, e non viene eseguita alcuna fase di pre-selezione.

Per il raggruppamento in cluster, viene selezionato un numero di cluster 320. Questo numero può essere un numero fisso o può essere determinato in base alle specifiche di job e/o allo stato di grid.

Una volta selezionato il numero di cluster 320, i dispositivi utente 310 vengono assegnati ai cluster 320, preferibilmente in base ai loro punteggi di dispositivo 150 e/o in base agli indici di grid 160. Così, l'assegnazione dei dispositivi utente 310 a diversi cluster 320 è in ultima analisi basata sui rispettivi parametri di capacità di calcolo e disponibilità.

In maggior dettaglio, a ciascun cluster 320 viene attribuito un intervallo di punteggi di cluster, e i dispositivi utente 310 vengono assegnati al cluster 320 se i relativi parametri di capacità di calcolo e disponibilità sono compatibili con l'intervallo di punteggi di cluster. Analogamente agli intervalli target utilizzati per la pre-selezione, gli intervalli di punteggi di cluster possono essere intervalli di valori applicati ai punteggi di dispositivo 150 o alla posizione del dispositivo negli indici di grid 160, per selezionare quali dispositivi utente 310 saranno inclusi o esclusi dai diversi cluster 320.

Nella realizzazione preferita, questa assegnazione è basata sul punteggio di

capacità. In ogni caso, poiché la pre-selezione si basa sul punteggio complessivo, anche l'assegnazione ai cluster 320 si basa almeno indirettamente sul punteggio complessivo, e perciò su qualunque punteggio di dispositivo 150 utilizzato per determinare il punteggio complessivo.

Dopo la fase di raggruppamento in cluster, selezionare lo schema di partizionamento job comprende selezionare dimensioni di chunk di cluster 411 e attribuirle ai diversi cluster 320. Le dimensioni di chunk di cluster 411 sono parametri di suddivisione che vengono utilizzati per suddividere i dati di job nei chunk di input 410. Più in dettaglio, ciascuna dimensione di chunk di cluster 411 rappresenta un valore della dimensione dei dati che sono destinati ad essere inclusi nei chunk di input 410 per un cluster 320.

Quando si selezionano le dimensioni di chunk di cluster 411, dimensioni di chunk di cluster 411 maggiori vengono preferibilmente attribuite a cluster 320 aventi intervalli di punteggi di cluster compatibili con migliori parametri di capacità di calcolo e disponibilità, e dimensioni di chunk di cluster 411 minori vengono attribuite a cluster 320 aventi intervalli di punteggi di cluster compatibili con parametri di capacità di calcolo e disponibilità peggiori.

Quando si selezionano le dimensioni di chunk di cluster 411, viene preferibilmente selezionata in primo luogo una dimensione minima da attribuire al cluster 320 avente l'intervallo di punteggi del cluster più basso. Successivamente, le dimensioni di cluster 411 dei cluster rimanenti 320 vengono selezionate maggiori, preferibilmente pari a multipli della dimensione minima, secondo rispettivi fattori di

espansione.

La dimensione minima può essere scelta da un tecnico del ramo in modo da bilanciare l'esigenza di prevedere dimensioni ridotte, per accelerare l'esecuzione di task da parte dei dispositivi utente 310, con la necessità di prevedere dimensioni che non siano eccessivamente ridotte, così da evitare di moltiplicare il numero di chunk di input 410 e impedire l'insorgere di problemi di comunicazione del sistema servizi interni 130 con la grid.

La dimensione minima può essere fissa, ma può essere anche personalizzata, immessa come input dalle entità cliente 200 nell'ambito della specifica di job.

Nelle realizzazioni preferite, selezionare lo schema di suddivisione job comprende inoltre assegnare ai cluster 320 rispettivi numeri di chunk di cluster 412, ad esempio in forma di fattori di crowding, o affollamento, rappresentanti il rapporto tra il numero di chunk di cluster 412 e il numero di dispositivi utente 310 di ciascun cluster 320. Come descritto in maggior dettaglio in seguito, il numero di chunk di cluster 412 è un parametro di distribuzione che viene utilizzato in una o più fasi durante il processo di assegnazione dei task 420 ai dispositivi utente 310. Più in dettaglio, il numero di chunk di cluster 412 rappresenta un numero di chunk di input 410 che sono destinati ad essere assegnati a ciascun cluster 320.

I numeri di chunk di cluster 412 vengono selezionati in base alle specifiche di job, e in particolare alla dimensione di job, e preferibilmente anche al livello di priorità. Numeri superiori di chunk di cluster 412 vengono assegnati a tutti i cluster 320 per job computazionali 400 con dimensione di job maggiore, e numeri inferiori di chunk di

cluster 412 vengono assegnati a tutti i cluster 320 per job computazionali 400 con dimensione di job minore.

Inoltre, numeri superiori di chunk di cluster 412 vengono assegnati a cluster 320 con migliori parametri di capacità di calcolo e disponibilità, per job computazionali 400 con livelli di priorità più elevati, e numeri inferiori di chunk di cluster 412 vengono assegnati a cluster 320 con migliori parametri di capacità di calcolo e disponibilità, per job computazionali 400 con livelli di priorità inferiori. Al contrario, numeri di chunk di cluster inferiori 412 vengono assegnati a cluster 320 aventi parametri di capacità di calcolo e disponibilità peggiori, per job computazionali 400 con livelli di priorità più elevati, e numeri superiori di chunk di cluster 412 vengono assegnati a cluster 320 aventi parametri di capacità di calcolo e disponibilità peggiori, per job computazionali 400 con livelli di priorità inferiori.

Pertanto, i dati di job vengono suddivisi in modo da generare, per ciascun cluster 320, chunk di input 410 in un numero non superiore, preferibilmente pari, al rispettivo numero di chunk di cluster 412, e con una dimensione non maggiore di, preferibilmente uguale, alla rispettiva dimensione di chunk di cluster 411, fino al raggiungimento della dimensione di job. In alcuni casi, il numero e la dimensione effettivi possono essere minori del numero di chunk di cluster selezionato 412 e della dimensione del chunk di cluster 411, essendo la dimensione reale di job diversa dalla somma delle dimensioni di chunk di tutti i chunk previsti.

Preferibilmente, la suddivisione dei dati di job comprende una fase di pre-suddivisione, in cui il flusso di input è suddiviso in chunk intermedi, con una

dimensione fissa maggiore delle dimensioni di chunk di cluster 411 selezionate per tutti i chunk di input 410. Quindi, in una fase di suddivisione finale, i chunk intermedi vengono suddivisi nei chunk di input 410 per ottenere le dimensioni di chunk di cluster 411 sopra descritte.

Contestualmente alla generazione progressiva dei chunk di input 410 mediante suddivisione dei dati di job come descritto, dai chunk di input 410 vengono generati i task 420. In alcune realizzazioni, da ciascun chunk di input 410 viene generato un solo task 420. Viceversa, nella realizzazione preferita, selezionare lo schema di partizionamento job comprende selezionare uno o più parametri di replica, in particolare un fattore di replica rappresentante un rapporto tra il numero di task 420 e il numero di chunk di input 410, benché possano essere selezionati in alternativa altri parametri di replica simili o equivalenti. È possibile selezionare un solo parametro di replica per tutti i cluster 320, oppure possono essere selezionati parametri di replica diversi per rispettivi cluster 320.

Il fattore di replica può essere selezionato pari a uno, ad indicare che viene creato un solo task 420 per ciascun chunk di input 410, o maggiore di uno, ad indicare che vengono creati più task 420 per almeno alcuni chunk di input 410, vale a dire un task originario e una o più repliche di task. Ad esempio, per un fattore di replica due, viene creata una replica di attività per ciascun chunk di input 410. Per un fattore di replica compreso tra uno e due, una certa frazione dei chunk di input 410 origina rispettive repliche di task, e una frazione complementare dei chunk di input 410 non origina alcuna replica di task.

Con fattori di replica più elevati verrà eseguito un maggior numero di task identici 410 da dispositivi utente distinti 310, come descritto di seguito. Si riducono così i rischi di mancata, errata o ritardata consegna dei risultati di calcolo. Tuttavia, fattori di replica più elevati comportano anche un incremento della potenza computazionale totale utilizzata per eseguire il job computazionale 400, per cui un tecnico del ramo potrà scegliere un fattore di replica bilanciato in base alla specifica di job e allo stato di grid.

Per aumentare il fattore di replica a due, senza esaurire il numero massimo di task simultanei 420 inviati ad un cluster 320, il fattore di crowding di ciascun cluster 320 è preferibilmente selezionato non superiore a 0,5.

Giova rilevare che la distinzione tra un task originale e una corrispondente replica di task può essere puramente lessicale, non essendo prevista alcuna distinzione gerarchica insita in tali task, che hanno in generale lo stesso contenuto ma identificativi distinti.

Pertanto, il metodo preferito per assegnare i task 420 ai dispositivi utente 320 comprende pre-assegnare gruppi di task di cluster ai cluster 320, in forma di code di cluster 421 memorizzate nel database di queuing 122. Giova rilevare che l'assegnazione dei task 420 alle code di cluster 421 viene eseguita progressivamente, mentre è in corso l'upload del flusso di input, contestualmente alla formazione dei chunk di input 410.

Ciascun gruppo di task di cluster comprende tutti i task 420 (originali e repliche) che originano dai chunk di input 410 aventi la dimensione di chunk di cluster 411 attribuita al cluster 320.

Preferibilmente, la pre-assegnazione dei gruppi di task di cluster ai cluster 310 è

l'ultima funzione eseguita dal servizio di partizionamento 132.

Una successiva fase di assegnazione dei task 420 ai dispositivi utente 310 consiste nell'alimentare i task 420 di ciascuna coda di cluster 421 in code di dispositivo individuali 422 di tutti i dispositivi del cluster. Ciò viene preferibilmente eseguito da un servizio di alimentazione 133 del sistema servizi interni 130. Le code di dispositivo individuali 422 possono essere memorizzate in un database di queuing, che può essere identico al database di queuing 122 che memorizza le code di cluster 421, cioè il primo database di queuing 122, oppure un secondo distinto database di queuing 123, come illustrato nella figura 1.

Nella realizzazione preferita, alimentare i task 420 nelle code di dispositivo 422 comprende il rimescolamento, o shuffling, dei task 420 di ciascuna coda di cluster 421 con ordini diversi per i diversi dispositivi utente 310 del cluster 320. I diversi ordini delle code di dispositivo individuali 422 possono essere selezionati come ordini casuali o con specifici schemi di shuffling che renderanno gli ordini diversi l'uno dall'altro. Così, i task 420 vengono alimentati nelle code di dispositivo individuali 422 nei rispettivi diversi ordini.

Le code di dispositivo 422 vengono inviate dal sistema servizi interni 130 ai rispettivi dispositivi utente 310, che includono tutti i task 420 della coda 422, oppure semplicemente un elenco di puntatori o indicatori riferiti ai task 420. Il software lato utente 140 comprende un servizio di download di task 142 per scaricare i task 420 e/o la rispettiva coda di dispositivo 422.

Le code di dispositivo individuali 422 includono etichette di task, che indicano

stati di assegnazione di ciascun task. Gli stati di assegnazione includono uno stato non ancora assegnato ("libero" nella figura 7) e uno stato già assegnato ("occupato" nella figura 7). In alcune realizzazioni, può essere previsto anche uno stato di assegnazione definente uno stato già assegnato e completato.

Inizialmente, le etichette indicano lo stato non ancora assegnato per tutti i task. Quindi, assegnare i task 420 comprende selezionare, da parte di un dispositivo utente 310 disponibile del cluster 320 che esegue il software lato utente 140, un task specifico 420, dalla relativa coda di dispositivo individuale 422, che non sia stato finora assegnato ad un altro dispositivo utente 310, e quindi viene etichettato come non ancora assegnato. Preferibilmente, il task selezionato 420 è il primo task 420 nella relativa coda di dispositivo 422 che non sia stato ancora selezionato da un altro dispositivo utente 310. Questo sarà diverso per molti dispositivi utente 310 grazie alla fase di shuffling.

Quando un dispositivo utente 310 seleziona un task 420, il software lato utente 140 è configurato per inviare un segnale di assegnazione al sistema servizi interni 130. Quindi, il sistema servizi interni 130 è configurato per confermare l'assegnazione del task specifico selezionato 420 al dispositivo utente 310. In particolare, il sistema servizi interni 130 aggiorna l'etichetta su uno stato di assegnazione indicante uno stato già assegnato per quel task specifico 420, in tutte le code di dispositivo individuali 422 del cluster 320. Di conseguenza, il task 420 non è più ritenuto disponibile, e non sarà selezionato da altri dispositivi utente 310 del cluster 320.

Il software lato utente 140 è configurato per ricevere qualunque task 420 assegnato al dispositivo utente 310, e per eseguire tali task 420, eseguendo il file

eseguibile del task 420 sul chunk di input 410 del task 420. L'esecuzione del task 420 genera nel dispositivo utente 310 un chunk di output 430 dei dati. Il software lato utente 140 è configurato per reinviare il chunk di output 430 al sistema di memorizzazione 120. Il software lato utente 140 comprende un servizio di upload di chunk di output 143 per caricare i chunk di output 430 nel sistema di memorizzazione 120.

Di conseguenza, il sistema di memorizzazione 120 riceve diversi chunk di output 430 da diversi dispositivi utente 310 per ciascun job computazionale 400. La funzione di memorizzazione dei chunk di output 430 viene preferibilmente eseguita su un database di bucket, o secchio, 124 del sistema di memorizzazione 120, che è preferibilmente un cloud database. In una realizzazione, il database di bucket 124 è configurato per creare temporaneamente un bucket virtuale di memorizzazione di ciascun chunk di output 430. Così, il database di bucket 124 può avere il funzionamento come una cartella web con limitazioni di accesso.

Giova rilevare che la generazione di alcuni task 420, la consegna di questi task 420 ai dispositivi utente 310, e la ricezione di alcuni chunk di output 430 possono aver luogo durante l'intervallo di tempo di upload, e possono proseguire progressivamente in modo simile a quanto descritto per suddividere il flusso di input. Ciò è schematicamente illustrato nella figura 3, dove il segno di spunta rappresenta la ricezione avvenuta di chunk di output 430 da alcuni dispositivi utente 310, e il segno della clessidra rappresenta la ricezione in attesa di chunk di output 430 da altri dispositivi utente 310.

In assenza di repliche (o fattore di replica pari a 1), i chunk di output 430 saranno generalmente differenti l'uno dall'altro, poiché i task 420 originati da un job

computazionale 400 sono generalmente differenti. Viceversa, con fattori di replica più elevati, è previsto che i chunk di output 430 originati da ciascuna replica di task siano uguali ai chunk di output 430 originati dai corrispondenti task originali, a meno che l'esecuzione di un task 420 non comporti un processo stocastico.

Ciò può tuttavia non essere applicabile al caso di chunk di output corrotti 430, in cui si sia verificata corruzione di dati in fase di trasmissione al dispositivo utente 310 del task 420, in fase di esecuzione del task 420, o in fase di trasmissione al sistema di memorizzazione 120 del chunk di output 430.

In caso di task deterministici 420 per i quali sia stata generata almeno una replica di task, e quindi per i quali siano stati ricevuti più chunk di output 430, il sistema servizi interni 130 è preferibilmente configurato per verificare l'integrità del chunk di output 430 confrontando i chunk di output 430 originati dallo stesso chunk di input 410.

Questa funzione viene preferibilmente eseguita da un servizio di convalida 134 del sistema servizi interni 130. Nel caso in cui si riscontri una mancata corrispondenza, il sistema servizi interni 130 è preferibilmente configurato per generare un'ulteriore replica di task per il chunk di input 410 dal quale sono stati ricevuti risultati corrotti. Questa ulteriore replica può essere inviata a, ed eseguita da, un ulteriore dispositivo utente 310, al fine di produrre un ulteriore chunk di output 430 per lo stesso chunk di input 410, per effettuare un ulteriore confronto sui chunk di output 430, ed identificare i chunk di output intatti e corrotti 430.

In modo simile, l'ulteriore replica può essere eseguita direttamente dal servizio di convalida 134.

Anche in assenza di repliche di task, il sistema servizi interni 130, in particolare il servizio di convalida 134, è preferibilmente configurato per verificare l'integrità dei chunk di output 430 confrontando parametri di chunk, come dimensioni e formati di chunk di output, con corrispondenti parametri attesi. Nel caso in cui da questa verifica emerga una corruzione dei chunk, si possono adottare azioni correttive simili a quelle sopra descritte per il caso di mancata corrispondenza di chunk di output.

Inoltre, il sistema servizi interni 130, in particolare il servizio di convalida 134, è configurato per accertare l'arrivo puntuale dei chunk di output 430, cioè l'arrivo prima di un tempo massimo prestabilito.

Preferibilmente, per le attività replicate 420, accertare l'arrivo puntuale dei chunk di output 430 comprende accertare che sia arrivato almeno un chunk di output 430 per ciascun chunk di input 410. In altri termini, anche nel caso in cui sia stato ricevuto puntualmente solo un chunk di output 430 benché siano stati originati più task 420 dal chunk di input 410, la ricezione di eventuali altri chunk di output 430 per un qualunque task identico 420 non viene attesa oltre il tempo massimo prestabilito, e viene confermato il complessivo arrivo puntuale per quel chunk di input 410.

Ciò riduce fortemente i ritardi di consegna del risultato di job completo 500 tanto più quanto elevato è il fattore di replica, poiché ogni mancata o ritardata consegna di un chunk di output 430 può essere compensata dall'arrivo puntuale di un altro chunk di output 430 originato dallo stesso chunk di input 410. È quindi basso il rischio di contemporanee mancate o ritardate consegne di chunk per più dispositivi utente 310. Tuttavia, la mancata replica dei chunk di output 430 può ridurre la possibilità di

verifiche di integrità.

L'arrivo puntuale può non verificarsi in caso di arrivo ritardato, ma anche in caso di errore in fase di trasmissione al dispositivo utente 310 del task 420, in fase di esecuzione del task 420, o in fase di trasmissione al sistema di memorizzazione 120 del chunk di output 430.

Preferibilmente, per almeno alcuni tipi di guasto in fase di esecuzione di un task 420, il software lato utente 140 è configurato per inviare un segnale di errore al sistema servizi interni 130. Viceversa, per altri tipi di guasto, ad esempio nel caso in cui il dispositivo utente 310 sia spento, il sistema servizi interni 130 non riceverà semplicemente alcuna risposta dal dispositivo utente 310 che ha ricevuto il task 420.

Più in dettaglio, la verifica dell'arrivo puntuale di un chunk di output 430 da uno specifico dispositivo utente 310 può comportare più fasi e più casi. In un primo caso, un mancato arrivo puntuale viene determinato quando si riceve un segnale di errore dal dispositivo utente 310. In un altro caso, un mancato arrivo puntuale viene determinato quando non viene ricevuta alcuna risposta di stato dal dispositivo utente 310 in risposta ad una query heartbeat di stato che segue la consegna del chunk di input 410 al dispositivo utente 310. In questo caso, preferibilmente il mancato arrivo puntuale viene confermato solo se la replica dal dispositivo utente 310 non viene ricevuta anche dopo un certo tempo da una query di conferma del sistema interno al dispositivo utente 310. In un altro caso ancora, il mancato arrivo puntuale viene determinato quando non viene ricevuto alcun chunk di output 430 fino alla scadenza del tempo massimo prestabilito.

Il sistema servizi interni 130, in particolare il servizio di convalida 134, è

configurato per compiere azioni mitigative nel caso in cui non sia stato ricevuto alcun chunk di output 430 per uno specifico chunk di input 410 entro il tempo massimo prestabilito (includendo tutti i casi sopra descritti). Le azioni mitigative sono simili a quelle descritte per il caso dei risultati corrotti. In particolare, queste comprendono riassegnare il task 420 relativo allo specifico chunk di input 410, o una sua copia, ad un altro dispositivo utente 310 o per l'esecuzione da parte dello stesso servizio di convalida 134. Preferibilmente, il chunk di input ritardato o errato 410 viene riassegnato sul servizio di convalida 134, in modo che non si incorra in alcun ulteriore rischio e non vengano compromesse le prestazioni complessive del job computazionale 400, ma la decisione è sempre soggetta dalla capacità corrente del sistema di memorizzazione 130.

Alla ricezione di uno o più chunk di output 430 per ogni singolo chunk di input 410 originato da un job computazionale 400, il sistema servizi interni 130 è configurato per assemblare questi chunk di output 430 ottenendo in tal modo il risultato di job 500. Parte di un risultato di job assemblato 500 è illustrato nella figura 3. Questa funzione viene preferibilmente eseguita da un servizio di assemblaggio 135 del sistema servizi interni 130.

Infine, il software lato cliente 110 è configurato in modo da trasmettere in output alle entità cliente 200 i risultati di job assemblati 500, preferibilmente consentendo il download dei risultati di job 500 attraverso la piattaforma clienti 111. Più in dettaglio, il software lato cliente 110 comprende un servizio di download risultati 113 per scaricare il risultato di job 500 e/o uno o più chunk di output 430.

Benché si siano finora descritti i componenti e i servizi principali che elaborano

il job computazionale 400, saranno preferibilmente previsti ulteriori componenti e servizi, come descritto di seguito.

Preferibilmente, il sistema servizi interni 130 comprende un servizio di ricompensa, o rewarding, 136 configurato per applicare il tariffario prestabilito per generare un addebito alle entità cliente 200 per l'elaborazione del job computazionale 400, e per generare un accredito alle entità di utente 300 per l'esecuzione dei task assegnati 420.

Più in dettaglio, il servizio di rewarding 136 è configurato per registrare i valori di addebito e accredito in un database di consistenza 125 del sistema di memorizzazione 120, che è preferibilmente un database in linguaggio di interrogazione strutturato (Structured Query Language, SQL) configurato per supportare transazioni atomiche informatiche.

Inoltre, il software lato cliente 110 comprende un servizio di addebito 114 configurato per scaricare i valori di addebito dal sistema di memorizzazione 120, e fornirli in uscita attraverso la piattaforma clienti 111.

Inoltre, il software lato utente 140 comprende un servizio di accredito 144 configurato per scaricare e fornire in uscita sul dispositivo utente 320 i valori di accredito tratti dal sistema di memorizzazione 120.

Preferibilmente, il sistema servizi interni 130 comprende un servizio di transazione 137, configurato per eseguire transazioni atomiche sul database di consistenza 125. Le transazioni atomiche eseguite includono modifiche di variabili che devono essere effettuate contemporaneamente per tutti i servizi del sistema servizi

interni 120. In particolare, esempi preferiti di tali modifiche di variabili comprendono modifiche nello stato di grid, modifiche nello stato di assegnazione dei task 420, modifiche nello stato dei job computazionali 400, e modifiche nello stato delle piattaforme clienti 111.

5

RIVENDICAZIONI

1. Metodo per il calcolo distribuito di job computazionali (400) composti da dati di job, comprendente:

- predisporre un sistema di memorizzazione dati (120) comprendente uno o più database (121, 122, 123, 124, 125),

- impostare una grid di dispositivi utente (310) provenienti da entità utente (300), i dispositivi utente (310) essendo remoti dal sistema di memorizzazione (120) e almeno periodicamente in comunicazione di segnale con il sistema di memorizzazione (120), ciascun dispositivo utente (310) avendo una capacità di calcolo prestabilita,

- interrogare periodicamente con query ciascun dispositivo utente (310) su un rispettivo stato di dispositivo, nella forma di parametri di capacità di calcolo e disponibilità,

- ricevere nel sistema di memorizzazione (120) risposte di stato da almeno alcuni dispositivi utente (310), che includono i loro rispettivi stati di dispositivo, e definire uno stato di grid in funzione almeno delle risposte di stato ricevute,

- fornire ad entità clienti (200) una piattaforma informatica clienti (111),

- ricevere specifiche di job, caricate nel sistema di memorizzazione (120) attraverso la piattaforma clienti (111) dalle entità cliente (200), ciascuna specifica di job includendo parametri caratterizzanti un rispettivo job computazionale (400),

- selezionare per ciascun job computazionale (400) un rispettivo schema di partizionamento del job in funzione dello stato di grid e della specifica di job,

- ricevere flussi di input di dati, ciascuno dei quali include i dati di job di un rispettivo job computazionale (400), ed essendo caricato attraverso la piattaforma clienti (111) da

un'entità cliente (200),

- suddividere i dati di job inclusi in ciascun flusso di input in chunk di input (410) di dati, secondo parametri di suddivisione dello schema di partizionamento selezionato per il job computazionale (400),
- generare uno o più task (420) per ciascun chunk di input (410), ciascun task (420) includendo il rispettivo chunk di input (410) e istruzioni eseguibili per eseguire calcoli sul rispettivo chunk di input (410),
- assegnare ed inviare ciascun task (420) ad uno o più rispettivi dispositivi utente (310), selezionati in base a parametri di distribuzione dello schema di partizionamento selezionato per il rispettivo job computazionale (400),
- ricevere nel sistema di memorizzazione (120) chunk di output (430) di dati dai dispositivi utente (310), ciascun chunk di output (430) essendo ottenuto da un dispositivo utente (310) mediante esecuzione di un task (420),
- accertare l'arrivo puntuale dei chunk di output (430),
- fornire come output alle entità cliente (200) risultati di job (500) attraverso la piattaforma clienti (111), ciascun risultato di job (500) includendo dati ottenuti assemblando i chunk di output (430) relativi al corrispondente job computazionale (400).

2. Metodo secondo la rivendicazione 1, in cui:

- per ciascun job computazionale (400), ricevere il rispettivo flusso di input richiede un intervallo di tempo di upload,

- le fasi di suddividere i dati del job, generare i task (420) e inviare i task (420) ai dispositivi utente (310) iniziano durante l'intervallo di tempo di upload,

- ciascun chunk di input (410) resta memorizzato temporaneamente nel sistema di memorizzazione (120), da un rispettivo istante di salvataggio che ha luogo durante l'intervallo di tempo di upload, ad un rispettivo istante di cancellazione,

- per almeno un chunk di input (410), l'istante di cancellazione ha luogo durante l'intervallo di tempo di upload del rispettivo job computazionale (400).

3. Metodo secondo la rivendicazione 1 o 2, in cui::

- i parametri di distribuzione dello schema di partizionamento includono uno o più intervalli target per rispettivi parametri di capacità di calcolo e disponibilità o loro combinazioni, gli intervalli target essendo determinati in funzione delle specifiche di job,

- lo schema di partizionamento prevede, al fine di assegnare task (420) di un job computazionale (400) a dispositivi utente (310), di scartare dispositivi utente incompatibili (310), aventi stato di dispositivo con parametri di capacità di calcolo e disponibilità incompatibili con gli uno o più intervalli target, e di almeno pre-selezionare i rimanenti dispositivi utente (310) compatibili, aventi stato di dispositivo con parametri di capacità di calcolo e disponibilità compatibili con gli uno o più intervalli target.

4. Metodo secondo una qualunque delle rivendicazioni da 1 a 3, in cui::

- selezionare lo schema di partizionamento job comprende raggruppare i dispositivi

utente (310) in cluster (320), ciascun cluster (320) comprendendo dispositivi utente (310) selezionati in base a valori dei rispettivi parametri di capacità di calcolo e disponibilità,

- assegnare i task (420) ai dispositivi utente (310) comprende:

    - pre-assegnare gruppi di task di cluster a cluster distinti (320), in forma di code di cluster (421),

    - alimentare i task (420) di ciascuna coda di cluster (421) a code di dispositivo individuali (422) di tutti i dispositivi utente (310) del cluster (320),

    - selezionare, da parte di un dispositivo utente (310) disponibile del cluster (320), un task (420) specifico dalla rispettiva coda di dispositivo individuale (422), che non sia stato ancora assegnato ad un altro dispositivo utente (310), e confermare l'assegnazione del task specifico selezionato (420) a detto dispositivo utente (310) disponibile.

5. Metodo secondo la rivendicazione 4, in cui:

- alimentare i task (420) delle code di cluster (421) alle code di dispositivo (422) comprende uno shuffling dei task (420) di ciascuna coda di cluster (421) con ordini diversi, preferibilmente casuali, per i diversi dispositivi utente (310) del cluster (320), e alimentare i task (420) alle code di dispositivo individuali (422) nei rispettivi ordini diversi,

- selezionare il task specifico (420) da parte del dispositivo utente disponibile (310) comprende selezionare il primo task (420) nella relativa coda di dispositivo (422) che

non sia stato ancora selezionato da un altro dispositivo utente (310).

6. Metodo secondo la rivendicazione 4 o 5, in cui:

- selezionare lo schema di partizionamento job comprende assegnare ai cluster (320), come parametro di suddivisione, rispettive diverse dimensioni di chunk di cluster (411), e come parametro di distribuzione, rispettivi numeri di chunk di cluster (412), in base alle specifiche di job, preferibilmente in cui dimensioni di chunk di cluster (411) maggiori vengono assegnate a cluster (320) aventi valori più elevati dei parametri di capacità di calcolo e disponibilità, e dimensioni di chunk di cluster (411) minori vengono assegnate a cluster (320) aventi valori inferiori dei parametri di capacità di calcolo e disponibilità,

- suddividere i dati di job genera per ciascun cluster (320), e alimenta alla relativa coda di cluster (421), chunk di input (410) in numero non superiore, preferibilmente pari al rispettivo numero di chunk di cluster (412), e con una dimensione non maggiore di, preferibilmente uguale alla rispettiva dimensione di chunk di cluster (411).

7. Metodo secondo una qualunque delle rivendicazioni da 1 a 6, in cui::

- lo schema di partizionamento job selezionato per ciascun job computazionale (400) comprende uno o più parametri di replica,

- generare uno o più task (420) per ciascun chunk di input (410) di un job computazionale (400) comprende generare più task replicati (420) per almeno alcuni chunk di input (410), in modo che i task (420) siano più numerosi dei chunk di input

(410) secondo gli uno o più parametri di replica, e

- accertare l'arrivo puntuale dei chunk di output (430) comprende accertare che sia arrivato almeno un chunk di output (430) per ciascun chunk di input (410).

8. Metodo secondo una qualunque delle rivendicazioni da 1 a 7, in cui, dopo aver accertato l'arrivo puntuale dei chunk di output (430), nel caso in cui non sia stato ricevuto alcun chunk di output (430) per uno specifico chunk di input (410) entro un tempo massimo prestabilito, il metodo comprende riassegnare il task (420) relativo allo specifico chunk di input (410) ad un altro dispositivo utente (310) o ad un servizio software di convalida (134) in esecuzione sul sistema di memorizzazione (120).

9. Metodo secondo una qualunque delle rivendicazioni da 1 a 8, in cui::

- definire lo stato di grid comprende elaborare almeno i parametri di capacità di calcolo e disponibilità di ciascuno stato di dispositivo per ottenere per ciascun dispositivo utente (310) uno o più punteggi di dispositivo (150),

- definire lo stato di grid comprende ordinare i dispositivi utente (310) in uno o più indici di grid (160) in base a rispettivi punteggi di dispositivo (150), e

- detti parametri di distribuzione dello schema di partizionamento job sono impostati per assegnare i task (420) ai dispositivi utente (310) in base alle posizioni dei dispositivi in detti uno o più indici di grid (160).

10. Metodo secondo una qualunque delle rivendicazioni da 1 a 9, in cui i parametri di

capacità di calcolo e di disponibilità di ciascuno stato di dispositivo comprendono uno o più tra:

- accenso/spento, intervallo di tempo di disponibilità, attività corrente del dispositivo utente (310), batteria rimanente, temperatura interna, posizione del dispositivo utente e/o dati accelerometrici,

- valori di installazione, disponibilità e/o soglia d'uso per RAM, CPU, memoria e/o larghezza di banda,

- tipo di rete, velocità, latenza e/o indirizzo IP,

- prestazioni su un task di prova recente o su un task (420) recente generato da un chunk di input (410).

11. Sistema elettronico (100) per il calcolo distribuito di job computazionali (400) con il metodo secondo una qualsiasi delle rivendicazioni da 1 a 10, comprendente:

- detto sistema di memorizzazione dati (120),

- un software lato cliente (110) comprendente detta piattaforma clienti (111), con algoritmi per l'input di dette specifiche di job, l'input di detti flussi di input di dati e l'output di detti risultati di job (500),

- un sistema software di servizi interni (130), per l'esecuzione nel sistema di memorizzazione dati (120), con algoritmi per interrogare con query i dispositivi utente (310) sullo stato di dispositivo, ricevere dette risposte di stato e definire detto stato di grid, selezionare detti schemi di partizionamento job, suddividere detti dati di job, generare detti task (420), assegnare ed inviare i task (420) ai dispositivi utente (310),

ricevere detti chunk di output (430) dai dispositivi utente (310), accertare l'arrivo puntuale dei chunk di output (430), ed assemblare i chunk di output (430) in detti risultati di job (500),

- un software lato utente (140), per l'esecuzione nei dispositivi utente (310), con algoritmi per valutare i parametri di capacità di calcolo e disponibilità di un dispositivo utente (310), generare ed inviare dette risposte di stato, ricevere i task (420) assegnati al dispositivo utente (310), eseguire i task (420) per generare detti chunk di output (430), ed inviare i chunk di output (430) al sistema di memorizzazione (120).

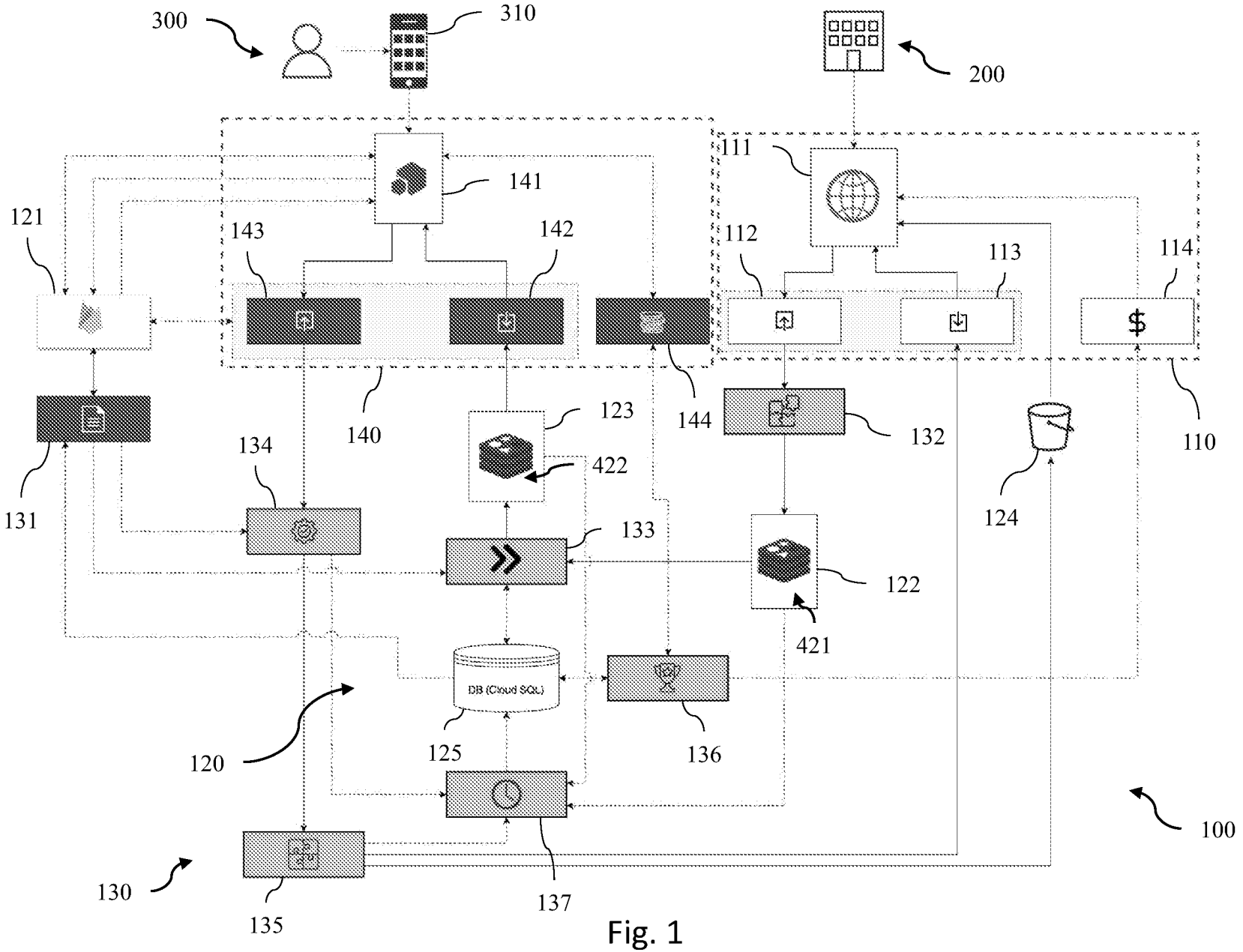
300
310
200
111
141
121
143
142
112
113
114
140
144
123
132
110
134
422
131
124
133
122
421
DB (Cloud SQL)
120
125
136
100
130
135
137

Fig. 1

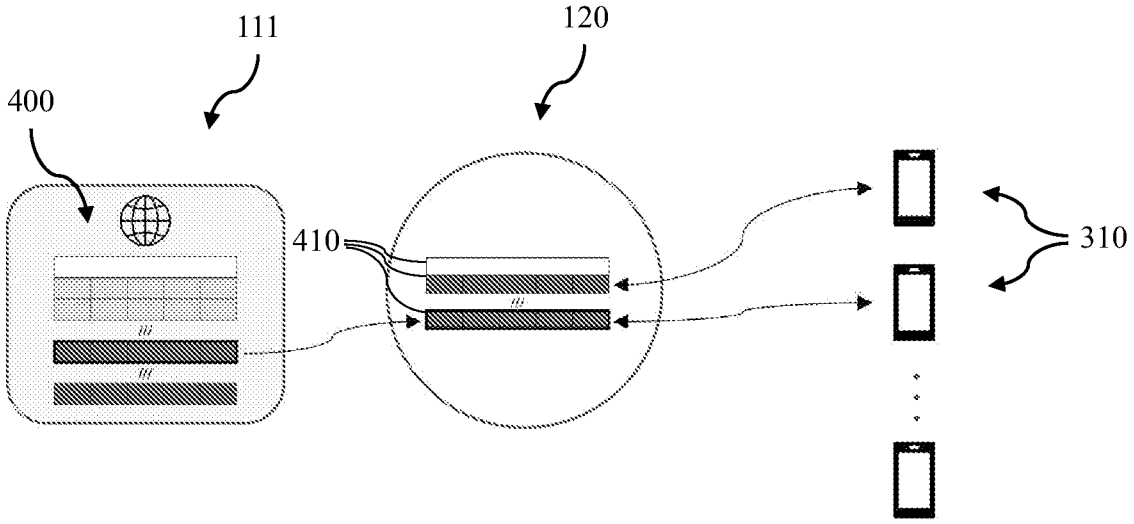

Fig. 2

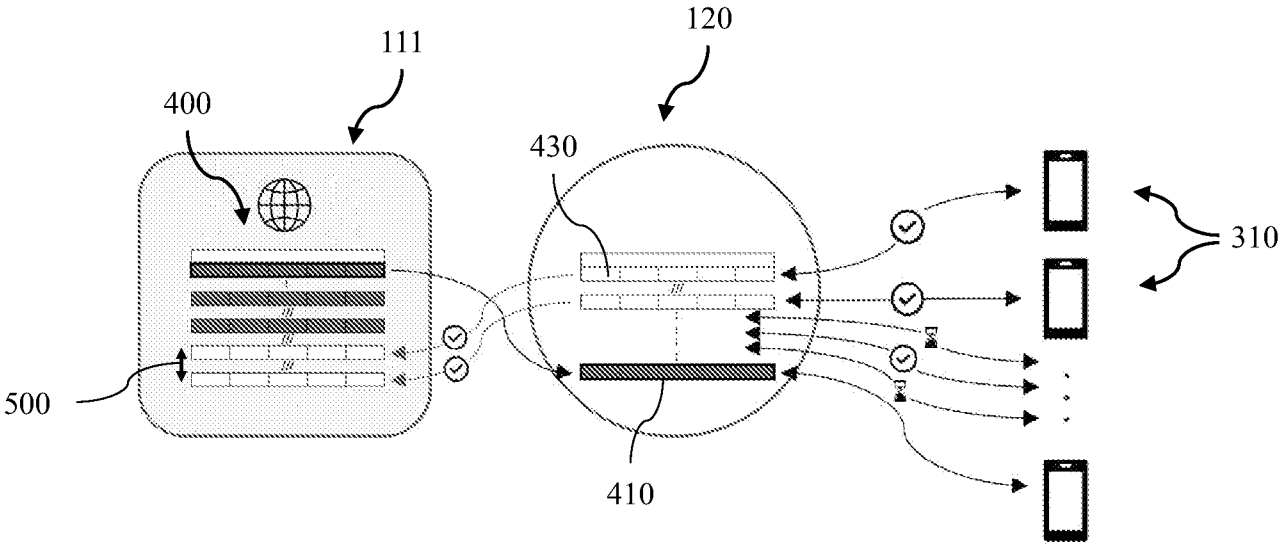

Fig. 3

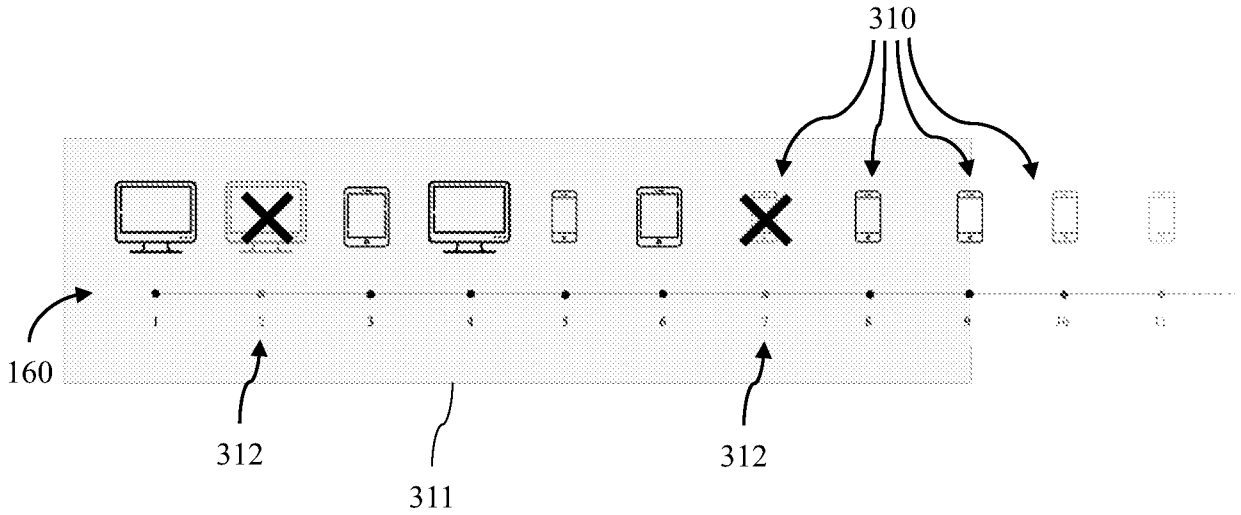
310
160
312
311
312

Fig. 4

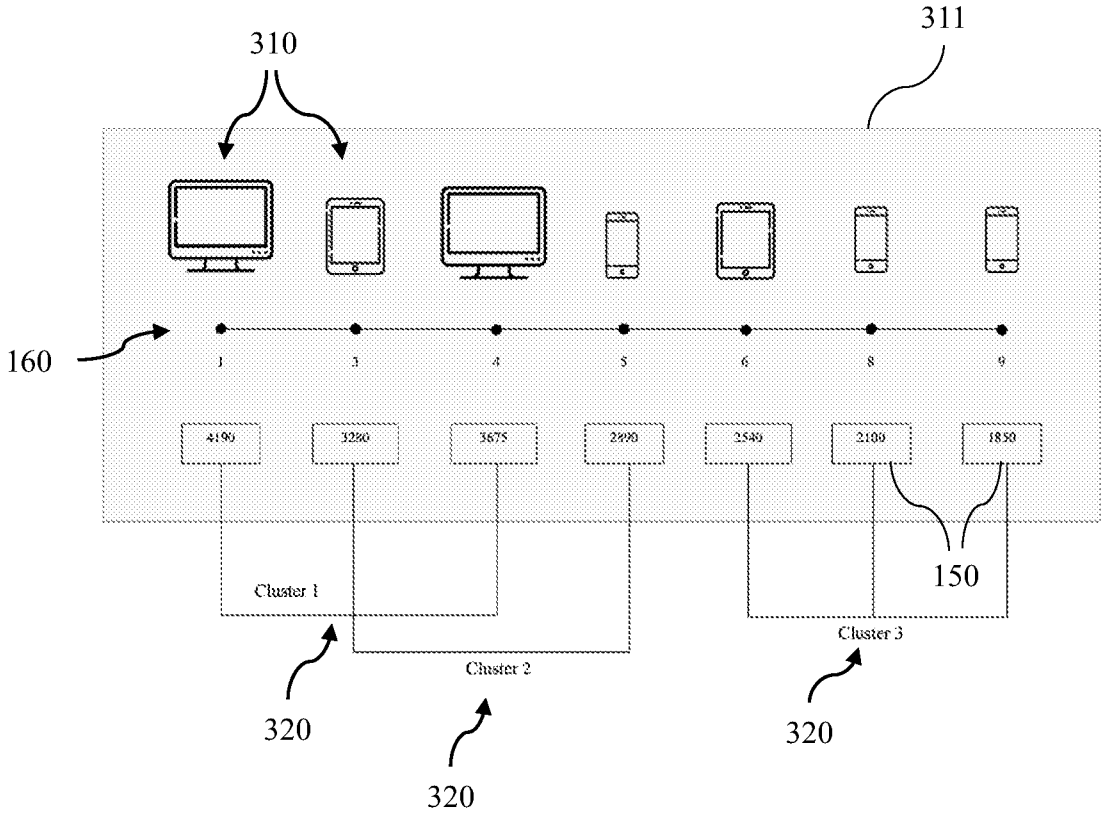
310
311
160
1
3
4
5
6
8
9
4190
3280
3675
2890
2540
2100
1850
150
Cluster 1
Cluster 2
Cluster 3
320
320
320

Fig. 5

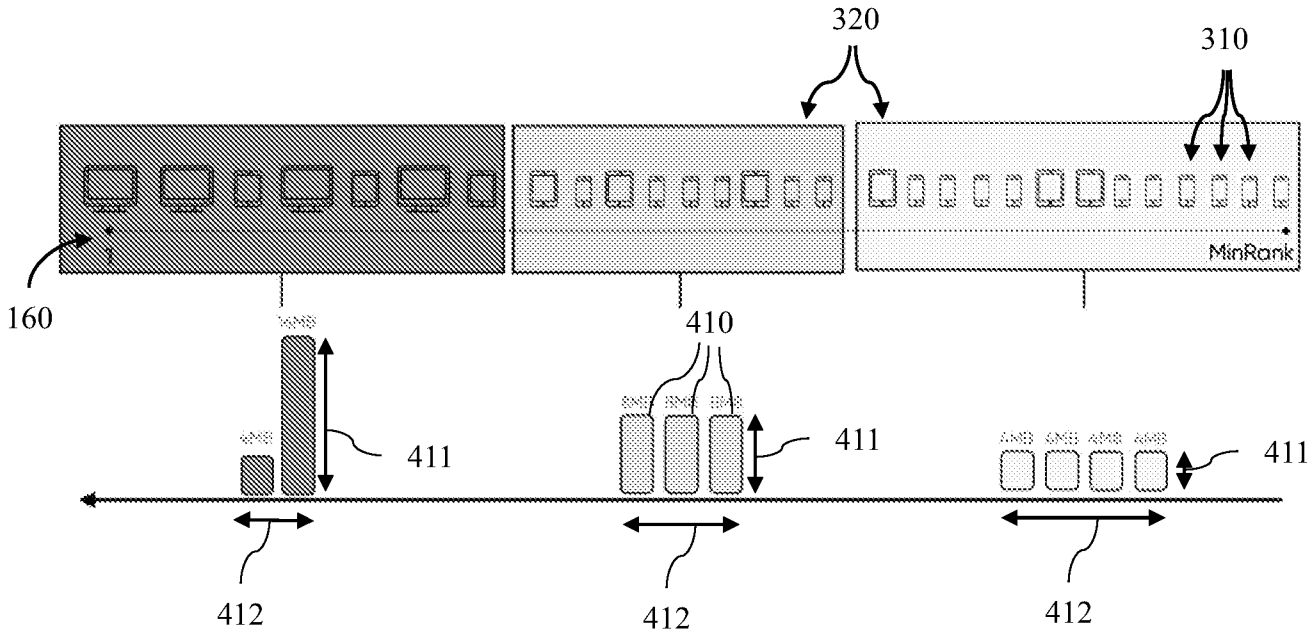
320
310
MinRank
160
410
411
411
411
412
412
412

Fig. 6

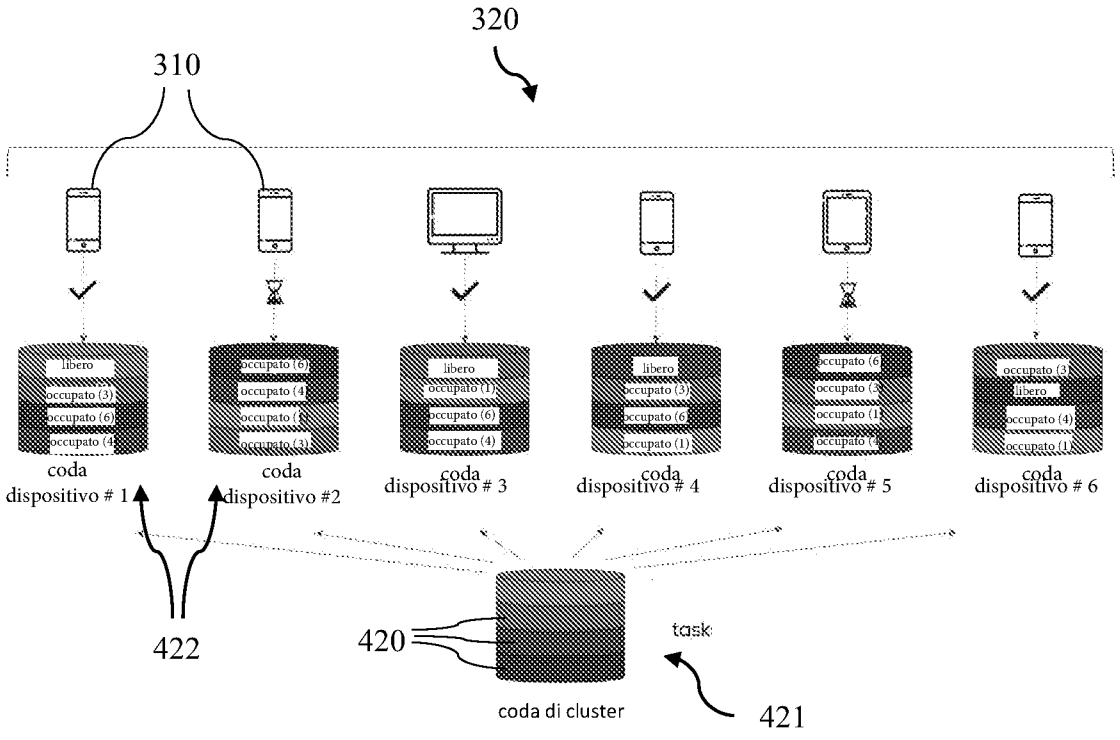
320
310
libero
occupato (3)
occupato (6)
occupato (4)
occupato (6)
occupato (4)
libero
occupato (1)
occupato (6)
occupato (4)
libero
occupato (3)
occupato (6)
occupato (1)
occupato (6
occupato (3
occupato (1
occupato (3)
libero
occupato (4)
occupato (1)
coda dispositivo # 1
coda dispositivo #2
coda dispositivo # 3
coda dispositivo # 4
coda dispositivo # 5
coda dispositivo # 6
422
420
task
coda di cluster
421

Fig. 7