(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5939123号
(P5939123)
(45)発行日　平成28年6月22日(2016.6.22)　(24)登録日　平成28年5月27日(2016.5.27)

(51)Int.Cl.　F I
G06F 17/30 (2006.01)　G06F 17/30 110C
G06F 17/30 415

請求項の数 6　(全 27 頁)

(21)出願番号　特願2012-224604(P2012-224604)
(22)出願日　平成24年10月9日(2012.10.9)
(65)公開番号　特開2014-78085(P2014-78085A)
(43)公開日　平成26年5月1日(2014.5.1)
審査請求日　平成27年6月4日(2015.6.4)

(73)特許権者　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100089118
弁理士　酒井　宏明
(72)発明者　上田　晴康
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内
(72)発明者　前田　高光
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

審査官　川▲崎▼　博章

最終頁に続く

(54)【発明の名称】実行制御プログラム、実行制御方法および情報処理装置

(57)【特許請求の範囲】
【請求項１】
コンピュータに、
形式の異なる複数の入力ファイルを読み込み、
各入力ファイルの間で種別が共通するカラムのデータを突合キーとして追加した中間ファイルを、前記入力ファイルごとに生成し、
前記突合キーに基づいて、各中間ファイル内のデータをソートし、
データがソートされた各中間ファイルから、前記入力ファイルの各形式のデータをそれぞれ抽出して、前記各入力ファイルに対してデータがソートされた複数の出力ファイルを生成し、
生成した複数の出力ファイルを、データを突合する突合プログラムに入力する
処理を実行させることを特徴とする実行制御プログラム。
【請求項２】
前記出力ファイルを生成する処理は、前記各中間ファイルのレコードごと、前記入力ファイルの各形式のデータを抽出し、
前記入力する処理は、前記各中間ファイルのレコードから前記各形式のデータが抽出されるたびに、抽出されたデータをプロセス間通信で前記突合プログラムに出力することを特徴とする請求項１に記載の実行制御プログラム。
【請求項３】
前記コンピュータは、複数のコンピュータに分散して保持されるデータを分散処理する

分散処理システムを形成し、
前記突合プログラムの処理結果を、他のコンピュータが共有でアクセスできる前記分散処理システム上の共有ファイルシステムに格納する処理をさらにコンピュータに実行させることを特徴とする請求項１または２に記載の実行制御プログラム。
【請求項４】
前記格納する処理は、前記突合プログラムが処理結果を出力するたびに、プロセス間通信で、前記共有ファイルシステムに処理結果を格納することを特徴とする請求項３に記載の実行制御プログラム。
【請求項５】
コンピュータが、
形式の異なる複数の入力ファイルを読み込み、
各入力ファイルの間で種別が共通するカラムのデータを突合キーとして追加した中間ファイルを、前記入力ファイルごとに生成し、
前記突合キーに基づいて、各中間ファイル内のデータをソートし、
データがソートされた各中間ファイルから、前記入力ファイルの各形式のデータをそれぞれ抽出して、前記各入力ファイルに対してデータがソートされた複数の出力ファイルを生成し、
生成した複数の出力ファイルを、データを突合する突合プログラムに入力する
を実行することを特徴とする実行制御方法。
【請求項６】
形式の異なる複数の入力ファイルを読み込む読込部と、
前記読込部によって読み込まれた各入力ファイルの間で種別が共通するカラムのデータを突合キーとして追加した中間ファイルを、前記入力ファイルごとに生成する第１生成部と、
前記突合キーに基づいて、各中間ファイル内のデータをソートするソート部と、
前記ソート部によってデータがソートされた各中間ファイルから、前記入力ファイルの各形式のデータをそれぞれ抽出して、前記各入力ファイルに対してデータがソートされた複数の出力ファイルを生成する第２生成部と、
前記第２生成部によって生成された複数の出力ファイルを、データを突合する突合プログラムに入力する入力部と
を有することを特徴とする情報処理装置。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、実行制御プログラム、実行制御方法および情報処理装置に関する。
【背景技術】
【０００２】
クラウドコンピューティングの普及に伴い、クラウド上に保存される大量のデータを複数のサーバで分散して処理を実行する分散処理システムが利用されている。分散処理システムとしては、HDFS（Hadoop Distributed File System）とMapReduceとを基盤技術とするHadoop（登録商標）が知られている。
【０００３】
HDFSは、複数のサーバにデータを分散格納するファイルシステムである。MapReduceは、HDFS上のデータをタスクと呼ばれる単位で分散処理する仕組みであり、Map処理、Shuffleソート処理、Reduces処理を実行する。なお、Map処理とReduces処理は、一般的にJava（登録商標）で開発され、Shuffleソート処理は、Hadoopに標準装備されている。このようなHadoopでは、１種類の入力をMapReduceで処理して１種類の出力を得るのが一般的である。
【０００４】
近年では、Hadoopを活用して、Hadoopには標準で装備されない、バッチ処理などの業務

プログラム（以下、適宜「外部プログラム」と記載する）を効率的に実行することが行われている。外部プログラムは、形式の異なる複数の入力を処理対象とし、一般的にJava（登録商標）以外のプログラムで開発される。

【０００５】

例えば、外部プログラムをHadoopで実行する技術として、Hadoopの標準ツールであるHadoop Streamingが知られている。Hadoop Streamingは、Map処理またはReduce処理で外部プログラムを呼び出す技術である。具体的には、Map処理またはReduce処理において、タスク１つにつき外部プログラムを１回呼び出し、外部プログラムの標準出力に処理結果を出力する。

【０００６】

また、複数種類の入力を処理するHadoop関連の技術として、reduce side joinが知られている。例えば、突合せ処理の外部プログラムを実行する場合に、入力種類ごとに入力ファイル名、入力フォーマットを処理するクラス、Map処理を実行するクラスを定義してMap処理を実行し、突合せキーと突合せ対象のデータとを対応付けたデータを出力する。続いて、Shuffleソート処理において、突合せキーでソートされ、突合せキーごとにグループ化されたデータを出力する。その後、入力種類ごとにReduceクラスを定義してReduce処理を実行し、マッチング処理を実行する。

【先行技術文献】

【非特許文献】

【０００７】

【非特許文献１】Deans and S. Ghemawat, MapReduce著 「Simplified Data Processing on Large Clusters, Proceedings of the 6th Symposium on Operating Systems Design and Implementation」、pp.137-150、December 6 2004

【非特許文献２】Apache Hadoop 1.0.3 documentation, 「Hadoop Streaming」,URL「http://hadoop.apache.org/docs/r1.0.3/streaming.html」

【非特許文献３】Tom White著、「Hadoop第二版、オライリージャパン、8.3.2 reduce側結合」、P269-272、2011年7月発行

【発明の概要】

【発明が解決しようとする課題】

【０００８】

しかしながら、複数入力を処理対象とする外部プログラムをHadoopなどの分散処理システムで実行させるには制約が多いことから、現実的には実行させるのが難しいという問題がある。

【０００９】

具体的には、Hadoop Streamingは、標準入出力経由で処理対象のデータを出力するので、引数や環境変数で処理対象のデータを受信する外部プログラムを呼び出して実行することができない。

【００１０】

また、reduce side joinでは、既存の外部プログラムと同様の処理を実行するプログラムを再開発してHadoopに移植することになり、再開発のリスクや移植によるリスクがあり、頻繁に外部プログラムを移植することができず、開発性が低い。

【００１１】

例えば、reduce side joinを用いる場合には、外部プログラムが処理対象とする入力ごとに、Map処理で処理対象となるMap処理クラスを実装し、さらに、入力ごとにReduce処理で処理対象となるReduce処理クラスを実装する。また、これらのクラスを実装する際に、外部プログラムが処理対象とするRDB（Relational Database)のファイルまたはRDBのファイルをアンロードしたCSV（Comma Separated Values）ファイルとは異なるKVS（Key Value Store）方式でデータを定義する。また、Java以外で開発されたプログラムであっても、reduce side joinのReduce処理で呼び出せるように、Javaに再開発して移植する。

【００１２】
つまり、reduce side joinを用いる場合には、複数種類の入力を１種類の入力のように見せかけて、Map処理およびReduce処理を実行し、Reduce処理において移植した外部プログラムを呼び出して実行することになる。
【００１３】
ところが、MapクラスやReduceクラスは、外部プログラムが処理する何百カラムの複雑なデータに対して、装置の支援無しに人手によって実装することになり、時間もかかり、人為的なミスのリスクも増える。また、外部プログラムの移植は、リスクが高く、好ましい手法とは言えない。なぜなら、外部プログラムは、複雑な業務ロジックが実装され、既に多くのテストを繰り返して実行実績も豊富なこともあり、簡単に移植できるものではないからである。
【００１４】
このように、Hadoop Streamingやreduce side joinを用いて外部プログラムを実行する場合には、作業時間の増加、人為的ミスの増加、外部プログラムの移植に伴うリスクなどがあり、外部プログラムをHadoopで実行するのが難しく、Hadoopの開発性の低下にもなる。
【００１５】
１つの側面では、複数入力を処理対象とする外部プログラムを分散処理システムで実行させる際の制約を緩和することができる実行制御プログラム、実行制御方法および情報処理装置を提供することを目的とする。
【課題を解決するための手段】
【００１６】
第１の案では、コンピュータは、形式の異なる複数の入力ファイルを読み込み、各入力ファイルの間で種別が共通するカラムのデータを突合キーとして追加した中間ファイルを、入力ファイルごとに生成する。コンピュータは、突合キーに基づいて、各中間ファイル内のデータをソートする。コンピュータは、データがソートされた各中間ファイルから、入力ファイルの各形式のデータをそれぞれ抽出して、各入力ファイルに対してデータがソートされた複数の出力ファイルを生成する。コンピュータは、生成した複数の出力ファイルを、データを突合する突合プログラムに入力する。
【発明の効果】
【００１７】
本発明の１実施態様によれば、複数入力を処理対象とする外部プログラムを分散処理システムで実行させる際の制約を緩和することができる。
【図面の簡単な説明】
【００１８】
【図１】図１は、実施例１に係る分散処理システムの全体構成例を示す図である。
【図２】図２は、実施例１に係るマスタ計算機の機能構成を示す機能ブロック図である。
【図３】図３は、設定ファイルＤＢに記憶される情報の例を示す図である。
【図４】図４は、タスクリストＤＢに記憶される情報の例を示す図である。
【図５】図５は、スレーブ計算機に送信されるReduceタスク情報の例を示す図である。
【図６】図６は、実施例１に係るスレーブ計算機の機能構成を示す機能ブロック図である。
【図７】図７は、実施例１に係るシステムが実行する外部プログラム実行処理の流れを示すシーケンス図である。
【図８】図８は、実施例１に係るMap処理の流れを示すフローチャートである。
【図９】図９は、実施例１に係るReduce処理の流れを示すフローチャートである。
【図１０】図１０は、実施例２に係るReduce処理の初期化の流れを示すフローチャートである。
【図１１】図１１は、実施例２に係るReduce処理の本処理および完了処理の流れを示すフローチャートである。

【図１２】図１２は、実施例２に係る出力読み込みスレッドの起動処理の流れを示すフローチャートである。
【図１３】図１３は、Map処理の具体例を説明する図である。
【図１４】図１４は、シャッフルソート処理の具体例を説明する図である。
【図１５】図１５は、Reduce処理の具体例を説明する図である。
【図１６】図１６は、突合プログラム実行後の出力先の具体例を説明する図である。
【図１７】図１７は、ハードウェア構成例を示す図である。
【発明を実施するための形態】
【００１９】
以下に、本願の開示する実行制御プログラム、実行制御方法および情報処理装置の実施例を図面に基づいて詳細に説明する。なお、この実施例によりこの発明が限定されるものではない。
【実施例１】
【００２０】
［全体構成］
図１は、実施例１に係る分散処理システムの全体構成例を示す図である。図１に示すように、この分散処理システムは、分析者端末２、入力ＤＢ（DataBase）サーバ３、マスタ計算機１０、スレーブ計算機２０、スレーブ計算機３０がネットワーク１を介して相互に通信可能に接続される。
【００２１】
この分散処理システムでは、Hadoop（登録商標）などの分散処理フレームワークを使用した分散処理アプリケーションが各計算機で実行されており、データ基盤としてHDFSなどを使用する。分析者端末２は、分散処理システムを利用して、分散処理フレームワークとは異なるフレームワークで実装された外部プログラムを実行し、データの解析を行うユーザの端末である。
【００２２】
入力ＤＢサーバ３は、外部プログラムが処理対象とする複数種類の入力ファイルを記憶するデータベースサーバである。例えば、入力ＤＢサーバ３は、商品マスタＤＢと売上明細ＤＢとを記憶する。また、入力ＤＢサーバ３が記憶するＤＢは、ＲＤＢのファイルまたはＲＤＢのファイルをアンロードしたＣＳＶファイル等で構成されているものとする。
【００２３】
マスタ計算機１０は、分散処理システムを統括的に管理するサーバである。例えば、マスタ計算機１０は、どのデータがいずれのスレーブ計算機に格納されているのかを管理し、各スレーブ計算機に割当てるタスクやジョブなどを管理する。
【００２４】
スレーブ計算機２０とスレーブ計算機３０は、分散処理アプリケーションを実装し、Map処理やReduce処理を実行して、ＨＤＦＳで管理されるデータを分散処理するサーバである。この各スレーブ計算機は、形式の異なる複数の入力ファイルを読み込み、各入力ファイルの間で種別が共通するカラムのデータを突合キーとして追加した中間ファイルを、前記入力ファイルごとに生成する。そして、各スレーブ計算機は、突合キーに基づいて、各中間ファイル内のデータをソートする。続いて、各スレーブ計算機は、データがソートされた各中間ファイルから、入力ファイルの各形式のデータをそれぞれ抽出して、各入力ファイルに対してデータがソートされた複数の出力ファイルを生成する。その後、各スレーブ計算機は、生成した複数の出力ファイルを、データを突合する突合プログラムに入力する。
【００２５】
このように、各スレーブ計算機は、Hadoopとはフレームワークが異なる突合プログラムなどの外部プログラムをHadoopで呼び出す際に、突合プログラムの設定に従って各入力ファイルから突合キーを抽出および突合キーでソートして新ファイルを生成する。そして、各スレーブ計算機は、新ファイルを入力ファイルに指定して突合プログラムを呼び出す。

【００２６】
つまり、各スレーブ計算機は、Hadoopにおいて複数種類の入力ファイルを読込む際に、外部プログラムに直接読込ませるという制約を設ける。このことにより、各スレーブ計算機は、外部プログラムの設定を与えることで、複数種類の入力ファイルを実行する外部プログラムを呼び出すことができる。したがって、複数入力を処理対象とする外部プログラムを分散処理システムで実行させる際の制約を緩和することができる。
【００２７】
［マスタ計算機の構成］
図２は、実施例１に係るマスタ計算機の機能構成を示す機能ブロック図である。図２に示すように、マスタ計算機１０は、通信制御部１１、記憶部１２、制御部１３を有する。なお、通信制御部１１は、例えばネットワークインタフェースカードなどである。記憶部１２は、メモリやハードディスクなどの記憶装置である。制御部１３は、プロセッサなどの電子回路である。
【００２８】
通信制御部１１は、各スレーブ計算機、分析者端末２、入力ＤＢサーバ３との間で情報を送受信する処理部である。例えば、通信制御部１１は、各スレーブ計算機からMap要求やReduce要求を受信し、各スレーブ計算機にMapタスク情報やReduceタスク情報を送信する。また、通信制御部１１は、入力ＤＢサーバ３から入力ファイル等を受信する。
【００２９】
記憶部１２は、設定ファイルＤＢ１２ａとタスクリストＤＢ１２ｂとを記憶する記憶部である。また、記憶部１２は、各種処理の途中経過を記憶する一時領域や、分散処理アプリケーション等を記憶する。
【００３０】
設定ファイルＤＢ１２ａは、Javaフレームワークなどの分散処理フレームワーク以外のフレームワークを用いて実装される外部プログラムの処理内容にしたがって設定された設定ファイルを記憶する。ここで記憶される条件は、管理者等によって設定される。また、外部プログラムは、例えばNetCOBOL（登録商標）などで実装された突合プログラムが該当する。
【００３１】
図３は、設定ファイルＤＢに記憶される情報の例を示す図である。なお、図３では、入力ファイルが２つであり、出力ファイルが２つである場合を例示している。入力ファイルとは、突合プログラムが処理対象とする種類の異なる複数のファイルであり、出力ファイルとは、各スレーブ計算機が外部プログラムを実行した結果を格納するＨＤＦＳ上の共有ディレクトリ上の共有ファイルである。
【００３２】
図３に示すように、設定ファイルＤＢ１２ａは、ジョブ設定を行うために設定された各種条件を記憶する。具体的には、設定ファイルには、入力ファイルごとに、入力データファイル名、入力フォーマット処理クラス名、入力フォーマット処理オプション、入力ファイル環境変数名が設定される。
【００３３】
例えば、入力データファイル名０１として「/usr/transaction.txt」が設定されており、これは、入力ファイルの１つが「/usr/transaction.txt」であることを示す。また、入力フォーマット処理クラス名０１として「固定長フォーマット」が設定されており、これは、入力ファイルを再構築する際に固定長フォーマットを用いることを示す。また、入力フォーマット処理オプション０１として「レコード長：80BYTE」が設定されており、これは、入力ファイルを再構成する際に80バイトの大きさで再構築することを示す。また、入力ファイル環境変数名０１として「TRAN」が設定されており、これは、入力ファイル０１がトランザクションファイルであることを示す。
【００３４】
さらに、設定ファイルには、出力データディレクト名、出力ファイルス名、出力ファイ

ル環境変数名が設定される。例えば、出力データディレクトリ名として「/output」が設定されており、これは、各スレーブ計算機が外部プログラムの処理結果を出力する先のHDFSディレクトとして「/output」が設定されていることを示す。また、出力ファイル名０１として「processed.txt」が設定されており、これは、スレーブ計算機２０が正常に終了した処理結果を「/output」のファイル「processed.txt」に出力することを示す。なお、図３には、エラーが発生した場合には、「/output」のファイル「errord.txt」に出力することも設定されている。また、出力ファイル環境変数名として「PROCOUT」が設定されており、これは、出力ファイルの環境変数として「PROCOUT」を指定することを示す。

【００３５】

さらに、設定ファイルには、Reduceアプリケーションやソートキーが設定される。例えば、Reduceアプリケーションには「a.out %in01 %in02 %out01 %out02」が設定されており、これは、外部プログラムに入力ファイルを渡す際に、１番目の入力ファイルは第１引数で渡し、２番目の入力ファイルは第２引数として渡すことを示す。また、ソートキー０１には「4BYTE～8BYTE」が設定されており、これは、入力ファイル０１の「4BYTE～8BYTE」をソートキーとして使用することを示す。

【００３６】

タスクリストＤＢ１２ｂは、Mapタスク管理部１５等によって生成されたMap処理またはReduce処理のタスクリストを記憶する。図４は、タスクリストＤＢに記憶される情報の例を示す図である。図４に示すように、タスクリストＤＢ１２ｂは、「タスクＩＤ、入力情報、クラス名」を記憶する。ここで、「タスクＩＤ」は、タスクを識別する識別子であり、「入力情報」は、入力データとして処理する情報を示し、「クラス名」は、Map処理の際に処理対象とするクラス名を示す。

【００３７】

例えば、タスクＩＤ「task__m__1」には、入力情報としてファイル名「/usr/transaction.txt」、「開始位置＝0」、「サイズ＝1000」が設定されており、クラス名として「フォーマット処理＝本フォーマット処理０１」、「マップ処理＝本Map処理０１」が設定されている。これは、入力フォーマット０１を用いたMap処理０１では、「/usr/transaction.txt」の0バイト目から1000バイト目までを入力データとして使用することを示している。なお、入力フォーマット０１を使用するとは、例えば、図３に示した入力ファイルに関する設定条件のうち「０１」に該当する条件を使用することを示す。

【００３８】

図２に戻り、制御部１３は、設定読込部１４、Mapタスク管理部１５、Reduceタスク管理部１６を有し、これらによって、各スレーブ計算機が実行するMap処理およびReduce処理を管理する処理部である。

【００３９】

設定読込部１４は、設定ファイルＤＢ１２ａから設定ファイルを読込んで、Mapタスク管理部１５やReduceタスク管理部１６に、設定ファイルに設定される各種条件を通知する処理部である。また、設定読込部１４は、各スレーブ計算機に対しても、設定ファイルに設定される各種条件を通知する。

【００４０】

Mapタスク管理部１５は、各スレーブ計算機で実行されるMapタスクを管理する処理部である。例えば、Mapタスク管理部１５は、Mapタスク準備として、入力ファイルを先頭から所定サイズごとに区切り、入力ファイルの番号でフォーマット処理およびマップ処理のクラス名を切替えて、タスクリストＤＢ１２ｂに登録する。

【００４１】

また、Mapタスク管理部１５は、各スレーブ計算機からのタスク要求を受信すると、Mapタスク情報を応答して、各スレーブ計算機にMapタスクを割当てる。例えば、Mapタスク管理部１５は、タスクＩＤが「task__m__1」のレコードをタスクリストＤＢ１２ｂから読み出して、スレーブ計算機２０に送信する。

【００４２】

Reduceタスク管理部１６は、各スレーブ計算機で実行されるReduceタスクを管理する処理部である。具体的には、Reduceタスク管理部１６は、各スレーブ計算機からのタスク要求を受信すると、Reduceタスク情報を応答して、各スレーブ計算機にReduceタスクを割当てる。図５は、スレーブ計算機に送信されるReduceタスク情報の例を示す図である。図５に示すように、スレーブ計算機に送信されるReduceタスク情報は、「タスクＩＤ、Map結果情報、クラス名」を対応付けて記憶する。

【００４３】

「タスクＩＤ」は、Reduceタスクを識別する識別子である。「Map結果情報」は、Map処理の結果を特定する情報である。「クラス名」は、Reduce処理の際に処理対象とするクラス名を示す。図５の場合、タスクＩＤ「task__r__1」のReduceタスクは、スレーブ計算機２０で処理されたMapタスクＩＤ「task__m__1」のMap処理の結果とスレーブ計算機３０で処理されたMapタスクＩＤ「task__m__2」のMap処理の結果とを用いて、Reduce処理を実行することを示す。なお、タスクＩＤ「task__r__1」のReduceタスクは、クラス名として「本Reduceクラス」が設定されている。

【００４４】

［スレーブ計算機の構成］

次に、スレーブ計算機について説明するが、スレーブ計算機２０とスレーブ計算機３０とは同様の構成を有するので、ここでは、スレーブ計算機２０について説明する。図６は、実施例１に係るスレーブ計算機の機能構成を示す機能ブロック図である。図６に示すように、スレーブ計算機２０は、通信制御部２１、記憶部２２、制御部２３を有する。なお、通信制御部２１は、例えばネットワークインタフェースカードなどである。記憶部２２は、メモリやハードディスクなどの記憶装置である。制御部２３は、プロセッサなどの電子回路である。

【００４５】

通信制御部２１は、マスタ計算機１０、入力ＤＢサーバ３、分析者端末２との間で情報を送受信する処理部である。例えば、通信制御部２１は、マスタ計算機１０のMapタスク要求やReduceタスク要求を送信し、マスタ計算機１０からMapタスク情報やReduceタスク情報を受信する。また、通信制御部２１は、入力ＤＢサーバ３から入力ファイル等を受信する。

【００４６】

記憶部２２は、中間ファイルＤＢ２２ａと一時ファイルＤＢ２２ｂとを記憶する記憶部である。また、記憶部２２は、分散処理アプリケーションや外部プログラム、入力ＤＢサーバ３から取得された入力ファイルや入力データ等を記憶する。

【００４７】

中間ファイルＤＢ２２ａは、制御部２３等によって読み込まれた入力ファイルや入力データ、また、Map処理の結果やReduce処理の結果を記憶する。すなわち、中間ファイルＤＢ２２ａは、スレーブ計算機２０が外部プログラムを呼び出す前に実行した処理の結果等を記憶する。一時ファイルＤＢ２２ｂは、外部プログラムによって処理された処理結果を記憶する。

【００４８】

制御部２３は、Map処理部２４とReduce処理部２５を有し、これらによって、Map処理およびReduce処理を実行し、さらに、外部プログラムを呼び出して実行する処理部である。Map処理部２４は、Map初期化部２４ａ、フォーマット初期化部２４ｂ、抽出部２４ｃを有し、これらによってMap処理を実行する処理部である。

【００４９】

Map初期化部２４ａは、マスタ計算機１０から受信したMapタスク情報にしたがって、Mapタスクの初期化を実行する処理部である。具体的には、Map初期化部２４ａは、どのようなフォーマットで入力ファイルを読み込み、また、ソートキーをどのように読込むかを、設定ファイルにしたがって設定する。

【００５０】

例えば、Map初期化部２４ａは、受信したMapタスク情報から「本Map処理０１」を抽出する。続いて、Map初期化部２４ａは、抽出した「本Map処理０１」から「０１」を抽出し、「０１」に対応する「ソートキー０１＝４BYTE～８BYTE」を設定ファイルから取得する。そして、Map処理部２４は、取得した情報をフォーマット初期化部２４ｂ、抽出部２４ｃ、Reduce処理部２５等に出力する。

【００５１】

フォーマット初期化部２４ｂは、Map処理実行結果を出力する中間ファイルのフォーマットの初期化を実行する処理部である。具体的には、フォーマット初期化部２４ｂは、複数の入力を読込むためのアダプタを入力ファイルごとに割当てる。

【００５２】

例えば、フォーマット初期化部２４ｂは、Map初期化部２４ａから受信した「本Map処理０１」の「０１」に対応する入力フォーマット処理クラス名や入力フォーマット処理オプション等を設定ファイルから取得して初期化を実行する。ここでは、フォーマット初期化部２４ｂは、「入力フォーマット処理クラス名０１＝固定長フォーマット」と、「入力フォーマット処理オプション０１＝レコード長（80BYTE）」等を取得する。

【００５３】

抽出部２４ｃは、Map処理を実行し、入力ファイル等から該当するデータを抽出して中間ファイルを生成する処理部である。具体的には、抽出部２４ｃは、フォーマット初期化部２４ｂによって設定されたフォーマットで入力ファイルの各レコードを読み込む。そして、抽出部２４ｃは、入力からソートキーを抽出してMap出力のＫｅｙとする。また、抽出部２４ｃは、何番目の入力であったかを示すファイルインデックスをＫｅｙまたはＶａｌｕｅの一部とし、読み込まれたレコード全体をＫｅｙまたはＶａｌｕｅの一部として、Map処理結果として出力する。

【００５４】

例えば、抽出部２４ｃは、Map初期化部２４ａから受信した「ソートキー０１＝４BYTE～8BYTE」にしたがって、入力ファイルのレコード（入力行）からソートキーを抽出する。続いて、抽出部２４ｃは、抽出したソートキーと、ソートキーの抽出元の入力行とを対応付けた中間ファイルを生成して、中間ファイルＤＢ２２ａに格納する。つまり、抽出部２４ｃは、ソートキーをＫｅｙ、入力行をＶａｌｕｅとするＫＶＳ形式の中間ファイルを生成する。なお、マスタ計算機１０は、ここで格納された中間ファイルを読み出して、Map処理結果を管理する。

【００５５】

Reduce処理部２５は、Shuffle処理部２５ａ、Reduce初期化部２５ｂ、再構築部２５ｃを有し、これらによって外部プログラムを呼び出してReduce処理を実行する処理部である。すなわち、Reduce処理部２５は、外部プログラムに適した複数の出力ファイルを生成して、外部プログラムを呼び出す。

【００５６】

Shuffle処理部２５ａは、Hadoopのreduce side joinなどで実行されるシャッフルソート処理を実行する処理部である。例えば、Shuffle処理部２５ａは、マスタ計算機１０に対してReduceタスク要求を送信して、Reduceタスク情報を受信する。そして、Shuffle処理部２５ａは、受信したReduceタスク情報のMap結果情報にしたがって、自装置がシャッフルソート処理の対象とするMap処理結果を各スレーブ計算機から収集して中間ファイルＤＢ２２ａに格納する。一例を挙げると、Shuffle処理部２５ａは、MapタスクＩＤ「task_m_2」のMap処理の結果をスレーブ計算機３０から取得し、MapタスクＩＤ「task_m_1」のMap処理の結果を自装置内の中間ファイルＤＢ２２ａから取得する。なお、割当てられる手法は、Hadoopの分散手法を用いる。

【００５７】

その後、Shuffle処理部２５ａは、抽出部２４ｃによって抽出されたソートキーを用いて、収集したMap処理の結果をソートする。そして、Shuffle処理部２５ａは、ソートした結果を同一キーでグループ化して、中間ファイルＤＢ２２ａに格納する。なお、各スレー

ブ計算機で分散処理されたシャッフルソート処理結果は、マスタ計算機１０によって各スレーブ計算機から収集されて管理される。

【００５８】

Reduce初期化部２５ｂは、Shuffle処理部２５ａが取得したReduceタスク情報にしたがって、Reduce処理を実行する前段階として初期化を実行する処理部である。具体的には、Reduce初期化部２５ｂは、どのように外部プログラムを呼び出し、入力ファイルをどのように渡すかを設定する。

【００５９】

例えば、Reduce初期化部２５ｂは、「a.out %in01 %in02 %out01 %out02」と指定して、１番目の入力ファイルを外部プログラムの第１引数に渡す。あるいは、Reduce初期化部２５ｂは、「ENVNAME.01=TRAN」と指定して、入力ファイル０１のファイル名をTRAN環境変数に設定して、外部プログラムを呼び出すと設定する。

【００６０】

また、Reduce初期化部２５ｂは、外部プログラムへの入力に該当するファイル名を決定し、外部プログラムを決定したファイル名を用いて呼び出すための準備を実行する。例えば、Reduce初期化部２５ｂは、呼出コマンドの引数の「%in01」を「./in01」に置き換えたり、ユーザ設定の環境変数TRANにファイル名「./in01」を設定したりする。

【００６１】

再構築部２５ｃは、Reduce初期化部２５ｂよって初期化された情報を用いて、マスタ計算機１０から指定されたReduceタスクを実行して出力ファイルを生成する。そして、再構築部２５ｃは、出力ファイルを入力ファイルに指定して、外部プログラムを実行する処理部である。すなわち、Reduce初期化部２５ｂは、マスタ計算機１０から指定されたReduceタスクを実行し、Shuffle処理部２５ａの処理結果から、該当するレコードを読み出して該当するファイルに書き込む。

【００６２】

このとき、再構築部２５ｃは、Shuffle処理部２５ａから出力されたＫＶＳ形式の処理結果を、元の入力ファイルと同じファイル形式に再変換する。そして、再構築部２５ｃは、Reduce初期化部２５ｂによって設定された当該ファイルの位置等を引数や環境変数を指定して、外部プログラムを呼び出して実行する。

【００６３】

また、再構築部２５ｃは、外部プログラムの実行によって得られた結果を、一時ファイル２２ｂに格納する。このとき、再構築部２５ｃは、設定ファイル等にしたがって、正常結果とエラー結果とを区別して、出力することもできる。

【００６４】

［シーケンス］

次に、図７を用いて、図１に示した分散処理システムで外部プログラムを実行する際の全体的な処理の流れを説明する。図７は、実施例１に係るシステムが実行する外部プログラム実行処理の流れを示すシーケンス図である。なお、ここで説明を簡略化するために、１台のスレーブ計算機２０を例にして説明する。

【００６５】

図７に示すように、マスタ計算機１０の設定読込部１４は、設定ファイルＤＢ１２ａから設定情報を読込む（Ｓ１０１）。続いて、Mapタスク管理部１５は、入力ＤＢサーバ３から入力ファイルが予め読み込まれたＨＤＦＳなどの共有ファイルシステムから入力ファイルを１つ読込み（Ｓ１０２）、データを所定サイズごとに分割し（Ｓ１０３）、タスクリストに登録する（Ｓ１０４）。例えば、Mapタスク管理部１５は、入力ファイルの番号で、フォーマット処理およびMap処理のクラス名を切替えて登録する。

【００６６】

そして、マスタ計算機１０のMapタスク管理部１５は、入力ファイルが他にも存在する場合には（Ｓ１０５：Ｙｅｓ）、Ｓ１０２に戻って以降の処理を繰り返す。一方、Mapタスク管理部１５は、入力ファイルが他に存在しない場合には（Ｓ１０５：Ｎｏ）、事前準

備が終了したことを示す通知を各スレーブ計算機に送信する（Ｓ１０６とＳ１０７）。なお、スレーブ計算機が定期的にタスク要求Ｓ１０８を実行するように構成したり、スレーブ計算機がタスク要求Ｓ１０８を実行した場合に、Ｓ１１０のMapタスク応答が返らないように構成することもでき、このような場合は、Ｓ１０６およびＳ１０７を省略することもできる。

【００６７】

この終了通知を受信したスレーブ計算機２０のMap処理部２４は、Mapタスク要求をマスタ計算機１０に送信する（Ｓ１０８とＳ１０９）。この要求を受信したマスタ計算機１０のMapタスク管理部１５は、図４のタスクリストから該当するタスクを抽出して、Mapタスクとしてスレーブ計算機２０に応答する（Ｓ１１０とＳ１１１）。

【００６８】

スレーブ計算機２０のMap処理部２４は、受信したMapタスクにしたがってMap処理を実行する（Ｓ１１２）。そして、Reduce処理部２５は、Map処理が終了すると、Reduceタスク要求をマスタ計算機１０に送信する（Ｓ１１３とＳ１１４）。この要求を受信したマスタ計算機１０のReduceタスク管理部１６は、図５に示すReduceタスクをスレーブ計算機２０に応答する（Ｓ１１５とＳ１１６）。その後、スレーブ計算機２０のReduce処理部２５は、受信したReduceタスクにしたがってReduce処理を実行する（Ｓ１１７）。

【００６９】

（Map処理）

次に、図７に示したMap処理について説明する。図８は、実施例１に係るMap処理の流れを示すフローチャートである。図８に示すように、スレーブ計算機２０のMap初期化部２４ａは、クラス名、または、クラス特有の定義より、入力番号（ＮＮ）を抽出する（Ｓ２０１）。例えば、Map初期化部２４ａは、マスタ計算機１０から受信したMapタスク情報から「本Maps処理０１」の「０１」を上記ＮＮとして抽出する。

【００７０】

続いて、Map初期化部２４ａは、入力番号（ＮＮ）とジョブ設定（Mapタスク情報）とからソートキーの読込み位置を抽出する（Ｓ２０２）。例えば、Map初期化部２４ａは、ＮＮ＝０１であることから、ソートキー（ＮＮ）＝ソートキー（０１）と特定し、「ソートキー＝０１」に対応付けられる「4BYTE～8BYTE」を抽出する。

【００７１】

その後、Map処理部２４のフォーマット初期化部２４ｂは、クラス名、または、クラス特有の定義より、入力番号（ＮＮ）を抽出する（Ｓ２０３）。例えば、フォーマット初期化部２４ｂは、Ｓ２０１と同様の手法で、「０１」を上記ＮＮとして抽出する。

【００７２】

続いて、フォーマット初期化部２４ｂは、入力番号（ＮＮ）とジョブ設定（Mapタスク情報）とから入力フォーマットのオプションを抽出し初期化を実行する（Ｓ２０４）。例えば、フォーマット初期化部２４ｂはＮＮ＝０１であることから、「入力フォーマット処理オプション０１」に対応付けられる「レコード長：80BYTE」を抽出し、入力ファイルのレコード長を初期化する。

【００７３】

その後、抽出部２４ｃは、入力ファイルから行であるレコードを読込み、読込んだ入力行からソートキーの値を抽出する（Ｓ２０５）。そして、抽出部２４ｃは、ソートキー（Ｋｅｙ）を「ソートキーの値」、値（Ｖａｌｕｅ）を「ＮＮ、入力行」とするＫＶＳ形式の中間ファイルを生成して、中間ファイルＤＢ２２ａに出力する（Ｓ２０６）。

【００７４】

（Reduce処理）

次に、図７に示したReduce処理について説明する。図９は、実施例１に係るReduce処理の流れを示すフローチャートである。図９に示すように、Reduce処理部２５のShuffle処理部２５ａは、各スレーブ計算機からMapタスク結果を収集し（Ｓ３０１）、収集した結果をソートし（Ｓ３０２）、ソートした結果を同一キーでグループ化する（Ｓ３０３）。

【００７５】
続いて、Reduce初期化部２５ｂは、全ての入力番号（ＮＮ）について、Ｓ３０４からＳ３１１を実行する。具体的には、Reduce初期化部２５ｂは、入力ＮＮ用一時ファイルを一時領域である一時ファイルＤＢ２２ｂにオープン（生成）する（Ｓ３０５）。続いて、Reduce初期化部２５ｂは、フォーマット処理ＮＮに対応する書き込みフォーマット処理を入力ＮＮ用一時ファイルと関連付ける（Ｓ３０６）。
【００７６】
その後、Reduce初期化部２５ｂは、書き込みフォーマット処理を入力フォーマット処理ＮＮのオプションで初期化する（Ｓ３０７）。例えば、Reduce初期化部２５ｂは、入力フォーマット処理オプション０１に対応する「レコード長：80BYTE」に、出力先のファイルを初期化する。
【００７７】
続いて、Reduce初期化部２５ｂは、入力ファイル番号ＮＮ、一時ファイル名、出力フォーマットクラスを対応付けた入力対応表を記憶部２２等に登録する（Ｓ３０８）。例えば、Reduce初期化部２５ｂは、「入力ファイル番号、ファイル名、出力フォーマットクラス」として「０１、./tmp/in01.txt、固定長フォーマット出力クラス」や「０２、./tmp/in02.txt、改行ありフォーマット出力クラス」を生成する。
【００７８】
その後、Reduce初期化部２５ｂは、ジョブ設定（Reduceタスク情報）で指定されたReduceアプリケーションの引数の「%inNN」を入力ＮＮ用一時ファイルのファイル名で置き換える（Ｓ３０９）。例えば、Reduce初期化部２５ｂは、Reduceアプリケーション「a.out %in01 %in02」を「a.out ./tmp/in01.txt ./tmp/in02.txt」に書き換える。
【００７９】
そして、Reduce初期化部２５ｂは、ジョブ設定で指定された入力ファイル環境変数名ＮＮを参照して、環境変数に入力ＮＮ用一時ファイルのファイル名を設定する（Ｓ３１０）。例えば、Reduce初期化部２５ｂは、環境変数の「TRAN」に「./tmp/in01.txt」、「MASTER」に「./tmp/in02.txt」を設定する。
【００８０】
その後、再構築部２５ｃは、全てのキーの値のリストでＳ３１２～Ｓ３１６を実行する。具体的には、再構築部２５ｃは、値＝｛ＮＮ、入力行｝としてＮＮ（入力ファイル番号）と入力行を抽出する（Ｓ３１３）。
【００８１】
そして、再構築部２５ｃは、入力対応表とＮＮとから出力フォーマットクラスオブジェクトを抽出し（Ｓ３１４）、出力フォーマットクラスオブジェクト経由で入力行を一時ファイルに出力する（Ｓ３１５）。このとき、実際にファイルの中身が出力されることとなる。
【００８２】
その後、再構築部２５ｃは、Reduceアプリケーションの文字列と環境変数で、外部プログラムを呼び出して実行し、終了するまで待機する（Ｓ３１７）。そして、再構築部２５ｃは、カレントディレクトリの全てあるいは一部のファイルをジョブ設定の出力ディレクトリのサブディレクトリにコピーする（Ｓ３１８）。
【００８３】
このようにすることで、外部プログラムをHadoopに移植することもなく、複雑で膨大なタスク定義を実装することもなく、Hadoopで外部プログラムを呼び出して実行することができる。この結果、作業時間の削減、人為的ミスの削減、外部プログラムの移植に伴うリスクが低減でき、Hadoopの開発性を向上させることもできる。
【実施例２】
【００８４】
実施例１では、外部プログラムの入力ファイルをディスク上のファイルとする例で説明したが、これに限定されるものではなく、例えば、いわゆる名前付きパイプ（named pip

e）を使用することもできる。また、外部プログラムの出力ファイルを取得し、MapReduceの入力として使用できる共有ファイルシステム上にコピーすることもできる。

【００８５】

そこで、実施例２では、名前付きパイプおよび共有ファイルシステムを使用する例を説明する。なお、マスタ計算機１０が実行する処理、各スレーブ計算機が実行するMap処理とShuffle処理については、実施例１と同様なので、説明を省略する。ここでは、実施例１とは異なるReduce処理について説明する。なお、図１０のＳ４０１の前に図９のＳ３０１からＳ３０３と同様の処理が実行される。

【００８６】

（Reduce処理の初期化処理）

図１０は、実施例２に係るReduce処理の初期化の流れを示すフローチャートである。図１０に示すように、スレーブ計算機２０のReduce初期化部２５ｂは、全ての入力番号（ＮＮ）について、Ｓ４０１からＳ４０９を実行する。

【００８７】

具体的には、Reduce初期化部２５ｂは、入力用名前付きパイプＮＮを一時領域にオープンする（Ｓ４０２）。ここで一時領域とは、例えば、スレーブ計算機２０の記憶部２２の一時ファイルＤＢ２２ｂなどである。

【００８８】

続いて、Reduce初期化部２５ｂは、フォーマット処理ＮＮに対応する出力フォーマット処理を入力ＮＮ用一時ファイルと関連付ける（Ｓ４０３）。その後、Reduce初期化部２５ｂは、出力フォーマット処理をフォーマット処理ＮＮのオプションで初期化する（Ｓ４０４）。

【００８９】

続いて、Reduce初期化部２５ｂは、入力用名前付きパイプの書き込み用スレッドを立ち上げる（Ｓ４０５）。その後、Reduce初期化部２５ｂは、入力ファイル番号ＮＮ、入力用名前付きパイプＮＮのファイル名、出力フォーマットクラス、書き込みスレッドＩＤを対応付けた入力対応表を記憶部２２等に登録する（Ｓ４０６）。

【００９０】

例えば、Reduce初期化部２５ｂは、「入力ファイル番号、ファイル名、出力フォーマットクラス、書き込みスレッドＩＤ」として「０１、./tmp/in01.txt、固定長フォーマット出力クラス、スレッド１０１」を生成する。また、Reduce初期化部２５ｂは、「０２、./tmp/in02.txt、改行ありフォーマット出力クラス、スレッド１０２」を生成する。

【００９１】

その後、Reduce初期化部２５ｂは、ジョブ設定で指定されたReduceアプリケーションの引数の「%inNN」を入力用名前付きパイプＮＮのファイル名で置き換える（Ｓ４０７）。例えば、Reduce初期化部２５ｂは、Reduceアプリケーション「a.out %in01 %in02 %out01 %out02」を「a.out ./tmp/in01.txt ./tmp/in02.txt %out01 %out02」に書き換える。

【００９２】

そして、Reduce初期化部２５ｂは、ジョブ設定で指定された入力ファイル環境変数名ＮＮを参照して、環境変数に入力用名前付きパイプＮＮのファイル名を設定する（Ｓ４０８）。例えば、Reduce初期化部２５ｂは、環境変数の「TRAN」に「./tmp/in01.txt」、「MASTER」に「./tmp/in02.txt」を設定する。

【００９３】

その後、Reduce初期化部２５ｂは、全ての出力番号（ＮＮ）について、Ｓ４１０からＳ４１９を実行する。具体的には、Reduce初期化部２５ｂは、出力用名前付きパイプＮＮを一時領域にオープンする（Ｓ４１１）。ここで一時領域とは、例えば、スレーブ計算機２０の記憶部２２の一時ファイルＤＢ２２ｂなどである。

【００９４】

続いて、Reduce初期化部２５ｂは、ジョブ設定の出力フォーマット処理ＮＮに対応する

出力フォーマット処理クラスを出力ファイルＮＮと関連付ける（Ｓ４１２）。その後、Reduce初期化部２５ｂは、出力フォーマット処理を出力フォーマット処理ＮＮのオプションで初期化する（Ｓ４１３）。

【００９５】

続いて、Reduce初期化部２５ｂは、読み込んだ出力を書き出す共有ファイルのファイル名を、ジョブ設定の出力データディレクトリ名と出力ファイル名ＮＮとから生成する（Ｓ４１４）。その後、Reduce初期化部２５ｂは、出力読み込みスレッドの起動処理を実行する（Ｓ４１５）。

【００９６】

そして、Reduce初期化部２５ｂは、Ｓ４１５の起動処理が終了すると、Ｓ４１６を実行する。すなわち、Reduce初期化部２５ｂは、出力ファイル番号ＮＮ、出力用名前付きパイプＮＮのファイル名、出力フォーマットクラス、出力読み込みスレッドＩＤ、共有ファイルのファイル名を対応付けた出力対応表を記憶部２２等に登録する。例えば、Reduce初期化部２５ｂは、「０１、./tmp/out01.txt、改行ありフォーマット出力クラス、スレッド１０３、/output/processed.txt」や「０２、./tmp/out02.txt、固定長フォーマット出力クラス、スレッド１０４、/output/error.txt」を生成する。

【００９７】

その後、Reduce初期化部２５ｂは、ジョブ設定で指定されたReduceアプリケーションの引数の「%outNN」を出力用名前付きパイプＮＮのファイル名で置き換える（Ｓ４１７）。例えば、Reduce初期化部２５ｂは、Reduceアプリケーション「a.out ./tmp/in01.txt ./tmp/in02.txt %out01 %out02」を「a.out ./tmp/in01.txt ./tmp/in02.txt ./tmp/out01.txt ./tmp/out02.txt」に書き換える。

【００９８】

そして、Reduce初期化部２５ｂは、ジョブ設定で指定された出力ファイル環境変数名NNを参照して、環境変数に出力用名前付きパイプＮＮのファイル名を設定する（Ｓ４１８）。その後、再構築部２５ｃは、Reduceアプリケーションの文字列と環境変数で、外部プログラムを呼び出す（Ｓ４２０）。

【００９９】

（Reduce処理の本処理および完了処理）

続いて、Reduce処理の本処理および完了処理を説明する。図１１は、実施例２に係るReduce処理の本処理および完了処理の流れを示すフローチャートである。図１１に示すように、スレーブ計算機２０の再構築部２５ｃは、全てのキーの値のリストでＳ５０１～Ｓ５０５を実行する。具体的には、再構築部２５ｃは、値＝{ＮＮ、入力行}としてＮＮ（入力ファイル番号）と入力行を抽出する（Ｓ５０２）。

【０１００】

そして、再構築部２５ｃは、入力対応表とＮＮとから出力フォーマットクラスオブジェクトを抽出し（Ｓ５０３）、出力フォーマットクラスオブジェクト経由で入力行を名前付きパイプに書き込む（Ｓ５０４）。

【０１０１】

その後、再構築部２５ｃは、入力用名前付きパイプをクローズし（Ｓ５０６）、Reduceアプリケーションが終了するまで待機する（Ｓ５０７）。そして、再構築部２５ｃは、入力用名前付きパイプの書き込みスレッド全てを終了する（Ｓ５０８）。

【０１０２】

続いて、再構築部２５ｃは、出力用名前付きパイプの読み込みスレッド全てが終了するまで待機する（Ｓ５０９）。その後、再構築部２５ｃは、カレントディレクトリの全てあるいは一部のファイルをジョブ設定の出力ディレクトリのサブディレクトリにコピーする（Ｓ５１０）。

【０１０３】

（スレッド起動処理）

続いて、図１０のＳ４１５に示したスレッド起動処理を説明する。図１２は、実施例２

に係る出力読み込みスレッドの起動処理の流れを示すフローチャートである。図１２に示すように、Reduce初期化部２５ｂは、共有ファイルのファイル名を書き込みモードでオープンし（Ｓ６０１）、共有ファイルを出力フォーマット処理クラスと関連付けて、初期化する（Ｓ６０２）。

【０１０４】

その後、Reduce初期化部２５ｂは、名前付きパイプを読み込みモードでオープンし（Ｓ６０３）、名前付きパイプを出力フォーマットに対応する入力フォーマット読み込みクラスと関連付けて、初期化する（Ｓ６０４）。

【０１０５】

そして、再構築部２５ｃは、名前付きパイプからデータが読み込める間、Ｓ６０５～Ｓ６０８を繰り返す。具体的には、再構築部２５ｃは、名前付きパイプから入力フォーマット読み込みクラスオブジェクト経由で１行読み込む（Ｓ６０６）。その後、再構築部２５ｃは、出力フォーマットクラスオブジェクト経由で入力行を共有ファイルに出力する（Ｓ６０７）。

【実施例３】

【０１０６】

次に、入力ファイル０１として商品マスタファイル、入力ファイル０２として売上明細ファイルを用いて、Hadoopで使用されるJavaフレームワークとは異なるNetCOBOLで実装された突合プログラムをHadoop内で読み出して実行する例を説明する。なお、実施例３では、１台のスレーブ計算機に１つのMap処理が割当てられている例で説明するが、これに限定されるものではなく、入力データの量やスレーブ計算機の台数等に応じて、Map処理の割当を適宜変更してもよい。

【０１０７】

図１３は、Map処理の具体例を説明する図であり、図１４は、シャッフルソート処理の具体例を説明する図であり、図１５は、Reduce処理の具体例を説明する図であり、図１６は、突合プログラム実行後の出力先の具体例を説明する図である。

【０１０８】

図１３に示すように、各スレーブ計算機がアクセス可能なHDFSディレクトリには、突合プログラムが処理対象とする入力ファイル０１と入力ファイル０２とが格納されている。

【０１０９】

入力ファイル０１は、商品マスタであり、「商品ＩＤ、商品名、単価、限定割引」として「0001、お茶、140、－」、「0011、梅おにぎり、110、10%」、「0012、鮭おにぎり、120、－」を記憶する。また、入力ファイル０２は、売上明細ファイルであり、「伝票ＩＤ、商品ＩＤ、個数」として「0001、0012、2」、「0001、0001、1」、「0002、0011、1」を記憶する。

【０１１０】

また、各スレーブ計算機は、「入力０１：行順ファイルカラム１抽出、入力０２：行順ファイル２抽出」が記述された設定ファイルを読み込む。このような状態において、スレーブ計算機２０は、入力ファイル０１を処理するMapタスクがマスタ計算機１０より指定されている。このスレーブ計算機２０は、設定ファイル「入力０１：行順ファイルカラム１抽出」にしたがって、入力ファイル０１のカラム１である「商品ＩＤ」をキーに設定する。

【０１１１】

そして、スレーブ計算機２０は、各レコードから商品ＩＤをキーとして抽出し、抽出したキーと抽出元の入力ファイルを示す情報と抽出元のレコードとを対応付けたＫＶＳ形式の中間ファイルを生成する。例えば、スレーブ計算機２０は、商品ＩＤが0001のレコードについて「Key、Value」として「Ｋ（0001）、Ｖ（1、0001、お茶、140、－）」を生成して、中間ファイルに格納する。

【０１１２】

同様に、スレーブ計算機３０は、入力ファイル０２を処理するMapタスクがマスタ計算

機１０より指定されている。このスレーブ計算機３０は、設定ファイル「入力０２：行順ファイルカラム２抽出」にしたがって、入力ファイル０２のカラム２である「商品ＩＤ」をキーに設定する。

【０１１３】

そして、スレーブ計算機３０は、各レコードから商品ＩＤをキーとして抽出し、抽出したキーと抽出元の入力ファイルを示す情報と抽出元のレコードとを対応付けたＫＶＳ形式の中間ファイルを生成する。例えば、スレーブ計算機３０は、伝票ＩＤが0001のレコードについては、商品ＩＤ「0012」をキーとして抽出し、「Key、Value」として「Ｋ（0012）、Ｖ（2、0001、0012、2）」を生成して、中間ファイルに格納する。

【０１１４】

その後、図１４に示すように、スレーブ計算機２０は、キーが「0001－0010」の範囲のデータを担当することがマスタ計算機１０によって指定されており、該当するデータを取得してシャッフルソートを実行する。具体的には、このスレーブ計算機２０は、自装置のMap処理結果とスレーブ計算機３０のMap処理結果から、「Ｋｅｙ」が「0001－0010」の範囲内であるレコードを読み出して、当該Ｋｅｙでソートする。

【０１１５】

例えば、スレーブ計算機２０は、自装置のMap処理結果から「Ｋ（0001）、Ｖ（1、0001、お茶、140、－）」を取得し、スレーブ計算機３０のMap処理結果から「Ｋ（0001）、Ｖ（2、0001、0001、1）」を取得する。そして、スレーブ計算機２０は、これらをソートして、Ｋ（0001）にＶ｛入力（1）、商品ＩＤ（0001）、商品名（お茶）、単価（140）、限定割引（－）｝を対応付けたレコードを中間ファイルに出力する。同様に、スレーブ計算機２０は、Ｋ（0001）にＶ｛入力（2）、伝票ＩＤ（0001）、商品ＩＤ（0001）、個数（1）｝を対応付けたレコードを中間ファイルに出力する。

【０１１６】

同様に、スレーブ計算機３０は、キーが「0011－0020」の範囲のデータを担当することがマスタ計算機１０によって指定されており、該当するデータを取得してシャッフルソートを実行する。具体的には、このスレーブ計算機３０は、自装置のMap処理結果とスレーブ計算機３０のMap処理結果から、「Ｋｅｙ」が「0011－0020」の範囲内であるレコードを読み出して、当該Ｋｅｙでソートする。

【０１１７】

例えば、スレーブ計算機３０は、スレーブ計算機２０のMap処理結果から「Ｋ（0011）、Ｖ（1、0011、梅おにぎり、110、10%）」と「Ｋ（0012）、Ｖ（1、0012、鮭おにぎり、120、－）」とを取得する。また、スレーブ計算機３０は、自装置のMap処理結果から「Ｋ（0012）、Ｖ（2、0001、0012、2）」と「Ｋ（0011）、Ｖ（2、0002、0011、1）」とを取得する。

【０１１８】

そして、スレーブ計算機３０は、これらをソートして、Ｋ（0011）にＶ｛入力（1）、商品ＩＤ（0011）、商品名（梅おにぎり）、単価（110）、限定割引（10%）｝を対応付けたレコードを中間ファイルに出力する。同様に、スレーブ計算機３０は、Ｋ（0011）に、Ｖ｛入力（2）、伝票ＩＤ（0002）、商品ＩＤ（0011）、個数（1）｝を対応付けたレコードを中間ファイルに出力する。

【０１１９】

また、スレーブ計算機３０は、これらをソートして、Ｋ（0012）にＶ｛入力（1）、商品ＩＤ（0012）、商品名（鮭おにぎり）、単価（120）、限定割引（－）｝を対応付けたレコードを中間ファイルに出力する。同様に、スレーブ計算機３０は、Ｋ（0012）にＶ｛入力（2）、伝票ＩＤ（0001）、商品ＩＤ（0012）、個数（2）｝を対応付けたレコードを中間ファイルに出力する。

【０１２０】

その後、図１５に示すように、スレーブ計算機２０は、シャッフルソートした結果に対してReduce処理を実行する。具体的には、スレーブ計算機２０は、「Ｋ（0001）、Ｖ｛入

力（1）、商品ＩＤ（0001）、商品名（お茶）、単価（140）、限定割引（－）｝」からＶ｛入力（1）、商品ＩＤ（0001）、商品名（お茶）、単価（140）、限定割引（－）｝を抽出する。そして、スレーブ計算機２０は、抽出したＶ｛入力（1）、商品ＩＤ（0001）、商品名（お茶）、単価（140）、限定割引（－）｝から入力（1）を削除し、入力ファイルと同じ形式のレコードに変換する。その後、スレーブ計算機２０は、入力ファイル０１用の名前付きパイプに、変換後の「商品ＩＤ（0001）、商品名（お茶）、単価（140）、限定割引（－）」を入力する。

【０１２１】

同様に、スレーブ計算機２０は、「Ｋ（0001）、Ｖ｛入力（2）、伝票ＩＤ（0001）、商品ＩＤ（0001）、個数（1）｝」からＶ｛入力（2）、伝票ＩＤ（0001）、商品ＩＤ（0001）、個数（1）｝」を抽出する。そして、スレーブ計算機２０は、抽出したＶ｛入力（2）、伝票ＩＤ（0001）、商品ＩＤ（0001）、個数（1）｝から入力（2）を削除し、入力ファイルと同じ形式のレコードに変換する。その後、スレーブ計算機２０は、入力ファイル０２用の名前付きパイプに、変換後の「伝票ＩＤ（0001）、商品ＩＤ（0001）、個数（1）」を入力する。

【０１２２】

また、スレーブ計算機３０は、「Ｋ（0011）、Ｖ｛入力（1）、商品ＩＤ（0011）、商品名（梅おにぎり）、単価（110）、限定割引（10%）｝」からＶ｛入力（1）、商品ＩＤ（0011）、商品名（梅おにぎり）、単価（110）、限定割引（10%）｝を抽出する。そして、スレーブ計算機３０は、抽出したＶ｛入力（1）、商品ＩＤ（0011）、商品名（梅おにぎり）、単価（110）、限定割引（10%）｝から入力（1）を削除し、入力ファイルと同じ形式のレコードに変換する。その後、スレーブ計算機３０は、入力ファイル０１用の名前付きパイプに、変換後の「商品ＩＤ（0011）、商品名（梅おにぎり）、単価（110）、限定割引（10%）」を入力する。

【０１２３】

また、スレーブ計算機３０は、「Ｋ（0011）、Ｖ｛入力（2）、伝票ＩＤ（0002）、商品ＩＤ（0011）、個数（1）｝」からＶ｛入力（2）、伝票ＩＤ（0002）、商品ＩＤ（0011）、個数（1）｝を抽出する。そして、スレーブ計算機３０は、抽出したＶ｛入力（2）、伝票ＩＤ（0002）、商品ＩＤ（0011）、個数（1）｝から入力（2）を削除し、入力ファイルと同じ形式のレコードに変換する。その後、スレーブ計算機３０は、入力ファイル０２用の名前付きパイプに、変換後の「伝票ＩＤ（0002）、商品ＩＤ（0011）、個数（1）」を入力する。

【０１２４】

また、スレーブ計算機３０は、「Ｋ（0012）、Ｖ｛入力（1）、商品ＩＤ（0012）、商品名（鮭おにぎり）、単価（120）、限定割引（－）｝」からＶ｛入力（1）、商品ＩＤ（0012）、商品名（鮭おにぎり）、単価（120）、限定割引（－）｝を抽出する。そして、スレーブ計算機３０は、抽出したＶ｛入力（1）、商品ＩＤ（0012）、商品名（鮭おにぎり）、単価（120）、限定割引（－）｝から入力（1）を削除し、入力ファイルと同じ形式のレコードに変換する。その後、スレーブ計算機３０は、入力ファイル０１用の名前付きパイプに、変換後の「商品ＩＤ（0012）、商品名（鮭おにぎり）、単価（120）、限定割引（－）」を入力する。

【０１２５】

同様に、スレーブ計算機３０は、「Ｋ（0012）、Ｖ｛入力（2）、伝票ＩＤ（0001）、商品ＩＤ（0012）、個数（2）｝」からＶ｛入力（2）、伝票ＩＤ（0001）、商品ＩＤ（0012）、個数（2）｝を抽出する。そして、スレーブ計算機３０は、抽出したＶ｛入力（2）、伝票ＩＤ（0001）、商品ＩＤ（0012）、個数（2）｝から入力（2）を削除し、入力ファイルと同じ形式のレコードに変換する。その後、スレーブ計算機３０は、入力ファイル０２用の名前付きパイプに、変換後の「伝票ＩＤ（0001）、商品ＩＤ（0012）、個数（2）」を入力する。

【０１２６】

その後、図１６に示すように、スレーブ計算機２０は、突合プログラムの処理結果を共有のHDFSディレクトリに出力する。具体的には、スレーブ計算機２０は、設定ファイルで指定される「std_output」に正常処理結果を出力し、設定ファイルで指定される「error_output」に異常処理結果を出力する。なお、スレーブ計算機３０も同様に処理結果を共有のHDFSディレクトリに出力する。

【０１２７】

上述したシステムは、複数の入力から突合せを行う外部プログラムに対して、MapReduceの枠組みにおいて、外部プログラムを修正なしに呼び出すことができるので、Hadoopや外部プログラムの開発生産性を向上させることができる。

【０１２８】

また、上述したシステムは、外部プログラムの入力ファイルをディスク上のファイルとする代わりに、名前付きパイプを用いることができるので、入力レコードの書き出しと既存プログラムの実行を並行動作させることができ、処理の高速化が図れる。また、ディスク書き込みよりも高速なプロセス間通信、すなわちメモリ間コピーを用いることができるので、スループットが向上する。また、全ての入力ファイルへの出力を同期して出力することでメモリ使用量を削減することもできる。

【０１２９】

また、上述したシステムは、既存プログラムの出力ファイルを回収し、MapReduceの入力として使える共有ファイルシステム上にコピーすることができる。また、上述したシステムは、外部プログラムの出力ファイルとして名前付きパイプを用い、外部プログラムがレコードを書き出すと同時に共有ファイルシステムに書き出すことができる。

【実施例４】

【０１３０】

さて、これまで本発明の実施例について説明したが、本発明は上述した実施例以外にも、種々の異なる形態にて実施されてよいものである。そこで、以下に異なる実施例を説明する。

【０１３１】

（システム）

また、本実施例において説明した各処理のうち、自動的におこなわれるものとして説明した処理の全部または一部を手動的におこなうこともできる。あるいは、手動的におこなわれるものとして説明した処理の全部または一部を公知の方法で自動的におこなうこともできる。この他、上記文書中や図面中で示した処理手順、制御手順、具体的名称、各種のデータやパラメータを含む情報については、特記する場合を除いて任意に変更することができる。

【０１３２】

また、図示した各装置の各構成要素は機能概念的なものであり、必ずしも物理的に図示の如く構成されていることを要しない。すなわち、各装置の分散・統合の具体的形態は図示のものに限られない。つまり、その全部または一部を、各種の負荷や使用状況などに応じて、任意の単位で機能的または物理的に分散・統合して構成することができる。さらに、各装置にて行なわれる各処理機能は、その全部または任意の一部が、ＣＰＵおよび当該ＣＰＵにて解析実行されるプログラムにて実現され、あるいは、ワイヤードロジックによるハードウェアとして実現され得る。

【０１３３】

（ハードウェア構成）

ところで、上記の実施例で説明した各種の処理は、あらかじめ用意されたプログラムをパーソナルコンピュータやワークステーションなどのコンピュータシステムで実行することによって実現することができる。そこで、以下では、上記の実施例と同様の機能を有するプログラムを実行するコンピュータの一例を説明する。

【０１３４】

図１７は、ハードウェア構成例を示す図である。図１７に示すように、コンピュータ１

００は、ＣＰＵ１０１、メモリ１０２、ディスクドライブ１０３、ＨＤＤ（Hard Disk Drive）１０４、通信制御部１０５、キーボード１０６、ディスプレイ１０７を有する。また、図１７に示した各部は、バス１００ａで相互に接続される。

【０１３５】

通信制御部１０５は、ＮＩＣ（Network Interface Card）などのインタフェースである。ＨＤＤ１０４は、図２や図６等に示した機能を実行するプログラムとともに、実施例１や実施例２で説明した各テーブル等を記憶する。記録媒体の例としてＨＤＤ１０４を例に挙げたが、ＲＯＭ（Read Only Memory）、ＲＡＭ、ＣＤ－ＲＯＭ等の他のコンピュータが読み取り可能な記録媒体に各種プログラムを格納しておき、コンピュータに読み取らせることとしてもよい。なお、記録媒体を遠隔地に配置し、コンピュータが、その記憶媒体にアクセスすることでプログラムを取得して利用してもよい。また、その際、取得したプログラムをそのＧＷ装置自身の記録媒体に格納して用いてもよい。

【０１３６】

ＣＰＵ１０１は、図２に示した各処理部と同様の処理を実行するプログラムを読み出してＲＡＭに展開することで、図２等で説明した各機能を実行するプロセスを動作させる。すなわち、このプロセスは、設定読込部１４、Mapタスク管理部１５、Reduceタスク管理部１６を実行する。このようにコンピュータ１００は、プログラムを読み出して実行することでマスタ計算機１０として動作する。

【０１３７】

また、ＣＰＵ１０１は、図６に示した各処理部と同様の処理を実行するプログラムを読み出してＲＡＭに展開することで、図６等で説明した各機能を実行するプロセスを動作させる。すなわち、このプロセスは、Map初期化部２４ａ、フォーマット初期化部２４ｂ、抽出部２４ｃ、Shuffle処理部２５ａ、Reduce初期化部２５ｂ、再構築部２５ｃを実行する。このようにコンピュータ１００は、プログラムを読み出して実行することでスレーブ計算機２０として動作する。

【０１３８】

また、コンピュータ１００は、媒体読取装置によって記録媒体から上記プログラムを読み出し、読み出された上記プログラムを実行することで上記した実施例と同様の機能を実現することもできる。なお、この他の実施例でいうプログラムは、コンピュータ１００によって実行されることに限定されるものではない。例えば、他のコンピュータまたはサーバがプログラムを実行する場合や、これらが協働してプログラムを実行するような場合にも、本発明を同様に適用することができる。

【０１３９】

以上の各実施例を含む実施形態に関し、さらに以下の付記を開示する。

【０１４０】

（付記１）コンピュータに、

形式の異なる複数の入力ファイルを読み込み、

各入力ファイルの間で種別が共通するカラムのデータを突合キーとして追加した中間ファイルを、前記入力ファイルごとに生成し、

前記突合キーに基づいて、各中間ファイル内のデータをソートし、

データがソートされた各中間ファイルから、前記入力ファイルの各形式のデータをそれぞれ抽出して、前記各入力ファイルに対してデータがソートされた複数の出力ファイルを生成し、

生成した複数の出力ファイルを、突合データを処理する突合プログラムに入力する

処理を実行させることを特徴とする実行制御プログラム。

【０１４１】

（付記２）前記出力ファイルを生成する処理は、前記各中間ファイルのレコードごと、前記入力ファイルの各形式のデータを抽出し、

前記入力する処理は、前記各中間ファイルのレコードから前記各形式のデータが抽出されるたびに、抽出されたデータをプロセス間通信で前記突合プログラムに出力することを

特徴とする付記１に記載の実行制御プログラム。
【０１４２】
（付記３）前記コンピュータは、複数のコンピュータに分散して保持されるデータを分散処理する分散処理システムを形成し、
前記突合プログラムの処理結果を、他のコンピュータが共有でアクセスできる前記分散処理システム上の共有ファイルシステムに格納する処理をさらにコンピュータに実行させることを特徴とする付記１または２に記載の実行制御プログラム。
【０１４３】
（付記４）前記格納する処理は、前記突合プログラムが処理結果を出力するたびに、プロセス間通信で、前記共有ファイルシステムに処理結果を格納することを特徴とする付記３に記載の実行制御プログラム。
【０１４４】
（付記５）コンピュータが、
形式の異なる複数の入力ファイルを読み込み、
各入力ファイルの間で種別が共通するカラムのデータを突合キーとして追加した中間ファイルを、前記入力ファイルごとに生成し、
前記突合キーに基づいて、各中間ファイル内のデータをソートし、
データがソートされた各中間ファイルから、前記入力ファイルの各形式のデータをそれぞれ抽出して、前記各入力ファイルに対してデータがソートされた複数の出力ファイルを生成し、
生成した複数の出力ファイルを、データを突合する突合プログラムに入力する
を実行することを特徴とする実行制御方法。
【０１４５】
（付記６）形式の異なる複数の入力ファイルを読み込む読込部と、
前記読込部によって読み込まれた各入力ファイルの間で種別が共通するカラムのデータを突合キーとして追加した中間ファイルを、前記入力ファイルごとに生成する第１生成部と、
前記突合キーに基づいて、各中間ファイル内のデータをソートするソート部と、
前記ソート部によってデータがソートされた各中間ファイルから、前記入力ファイルの各形式のデータをそれぞれ抽出して、前記各入力ファイルに対してデータがソートされた複数の出力ファイルを生成する第２生成部と、
前記第２生成部によって生成された複数の出力ファイルを、データを突合する突合プログラムに入力する入力部と
を有することを特徴とする情報処理装置。
【０１４６】
（付記７）メモリと、
前記メモリに接続されるプロセッサと、を有し、
前記プロセッサは、
形式の異なる複数の入力ファイルを読み込み、
各入力ファイルの間で種別が共通するカラムのデータを突合キーとして追加した中間ファイルを、前記入力ファイルごとに生成し、
前記突合キーに基づいて、各中間ファイル内のデータをソートし、
データがソートされた各中間ファイルから、前記入力ファイルの各形式のデータをそれぞれ抽出して、前記各入力ファイルに対してデータがソートされた複数の出力ファイルを生成し、
生成した複数の出力ファイルを、データを突合する突合プログラムに入力する
処理を実行させることを特徴とする情報処理装置。
【０１４７】
（付記８）形式の異なる複数の入力ファイルを読み込み、
各入力ファイルの間で種別が共通するカラムのデータを突合キーとして追加した中間フ

ァイルを、前記入力ファイルごとに生成し、
前記突合キーに基づいて、各中間ファイル内のデータをソートし、
データがソートされた各中間ファイルから、前記入力ファイルの各形式のデータをそれぞれ抽出して、前記各入力ファイルに対してデータがソートされた複数の出力ファイルを生成し、
生成した複数の出力ファイルを、データを突合する突合プログラムに入力する処理をコンピュータに実行させる実行制御プログラムを記憶する、コンピュータ読み取り可能な記憶媒体。

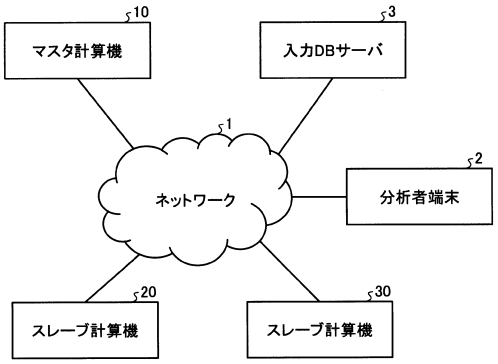
【符号の説明】

【０１４８】

１　ネットワーク
２　分析者端末
３　入力ＤＢサーバ
１０　マスタ計算機
１１　通信制御部
１２　記憶部
１２ａ　設定ファイルＤＢ
１２ｂ　タスクリストＤＢ
１３　制御部
１４　設定読込部
１５　Mapタスク管理部
１６　Reduceタスク管理部
２０、３０　スレーブ計算機
２１　通信制御部
２２　記憶部
２２ａ　中間ファイルＤＢ
２２ｂ　一時ファイルＤＢ
２３　制御部
２４　Map処理部
２４ａ　Map初期化部
２４ｂ　フォーマット初期化部
２４ｃ　抽出部
２５　Reduce処理部
２５ａ　Shuffle処理部
２５ｂ　Reduce初期化部
２５ｃ　再構築部

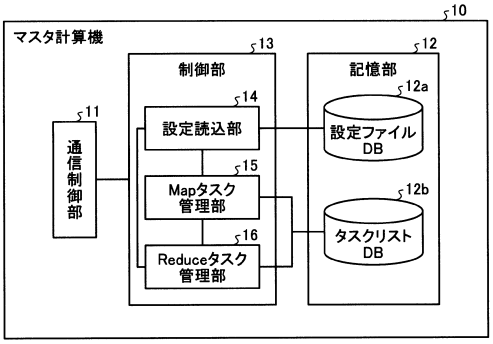
【図１】

実施例1に係る分散処理システムの全体構成例を示す図



【図２】

実施例1に係るマスタ計算機の機能構成を示す機能ブロック図



【図３】

設定ファイルDBに記憶される情報の例を示す図

ジョブ設定
入力データファイル名01= /usr/transaction.txt
入力データファイル名02= /usr/master.txt
入力フォーマット処理クラス名01= 固定長フォーマット
入力フォーマット処理オプション01= レコード長:80BYTE
入力フォーマット処理クラス名02= 改行ありフォーマット
入力フォーマット処理オプション02= 改行コード:¥r¥n
Reduceアプリケーション= a.out %in01 %in02 %out01 %out02
入力ファイル環境変数名01= TRAN
入力ファイル環境変数名02= MASTAR
出力データディレクトリ名= /output/
出力ファイル名01= processed.txt
出力ファイル名02= error.txt
出力ファイル環境変数名01= PROCOUT
出力ファイル環境変数名02= ERROUT
ソートキー01= 4BYTE～8BYTE
ソートキー02= 12BYTE～16BYTE

【図４】

タスクリストDBに記憶される情報の例を示す図

タスクID	入力情報	クラス名
task_m_1	ファイル名=/usr/transaction.txt	フォーマット処理=本フォーマット処理01
	開始位置=0	マップ処理=本Map処理01
	サイズ=1000	
task_m_2	ファイル名=/usr/transaction.txt	フォーマット処理=本フォーマット処理01
	開始位置=1001	マップ処理=本Map処理01
	サイズ=1000	
task_m_3	ファイル名=/usr/transaction.txt	フォーマット処理=本フォーマット処理01
	開始位置=2001	マップ処理=本Map処理01
	サイズ=1000	
task_m_4	ファイル名=/usr/master.txt	フォーマット処理=本フォーマット処理02
	開始位置=0	マップ処理=本Map処理02
	サイズ=1000	
task_m_5	ファイル名=/usr/master.txt	フォーマット処理=本フォーマット処理02
	開始位置=1001	マップ処理=本Map処理02
	サイズ=1000	

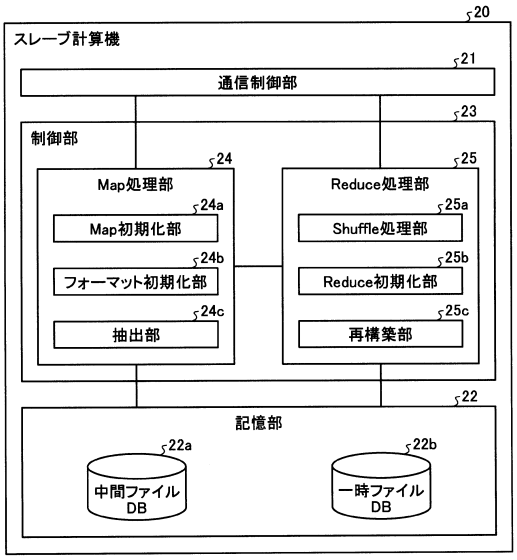
【図５】

スレーブ計算機に送信されるReduceタスク情報の例を示す図

タスクID	Map結果情報		クラス名
task_r_1	Mapタスク ID	スレーブ計算機名	本Reduceクラス
	task_m_1	スレーブ計算機20	
	task_m_2	スレーブ計算機30	

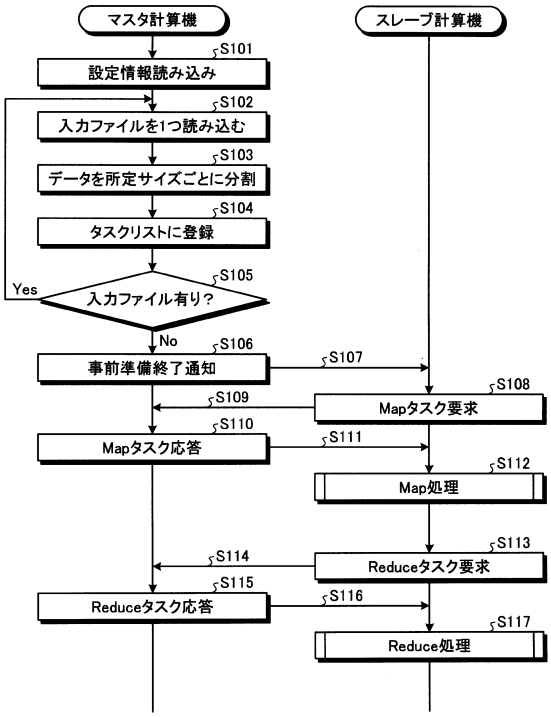
【図６】

実施例１に係るスレーブ計算機の機能構成を示す機能ブロック図



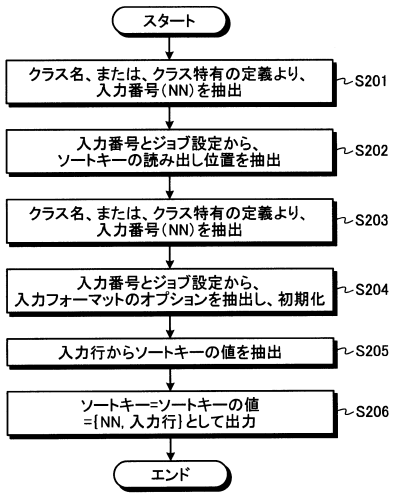
【図７】

実施例１に係るシステムが実行する
外部プログラム実行処理の流れを示すシーケンス図



【図８】

実施例１に係るMap処理の流れを示すフローチャート



【図９】

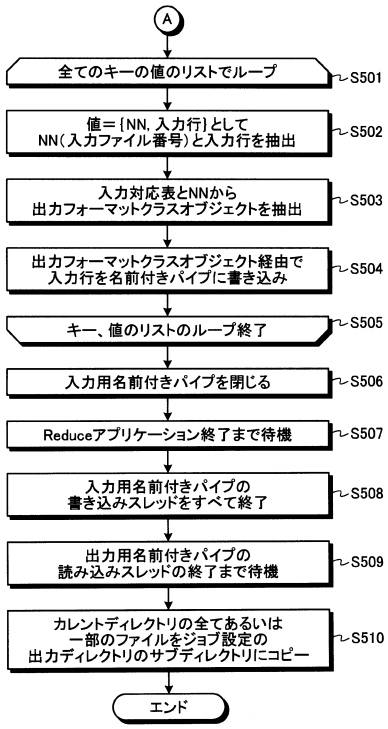
実施例1に係るReduce処理の流れを示すフローチャート

【図１０】

実施例2に係るReduce処理の初期化の流れを示すフローチャート

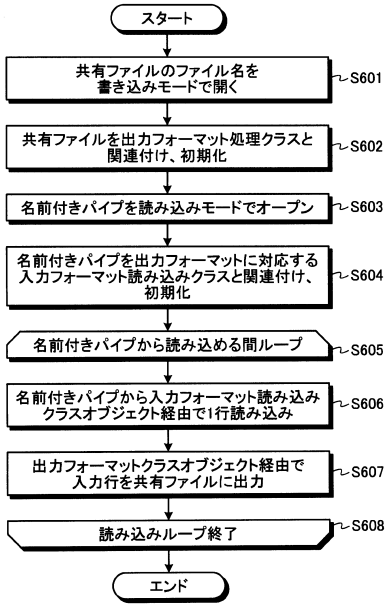
【図１１】

実施例2に係るReduce処理の
本処理および完了処理の流れを示すフローチャート

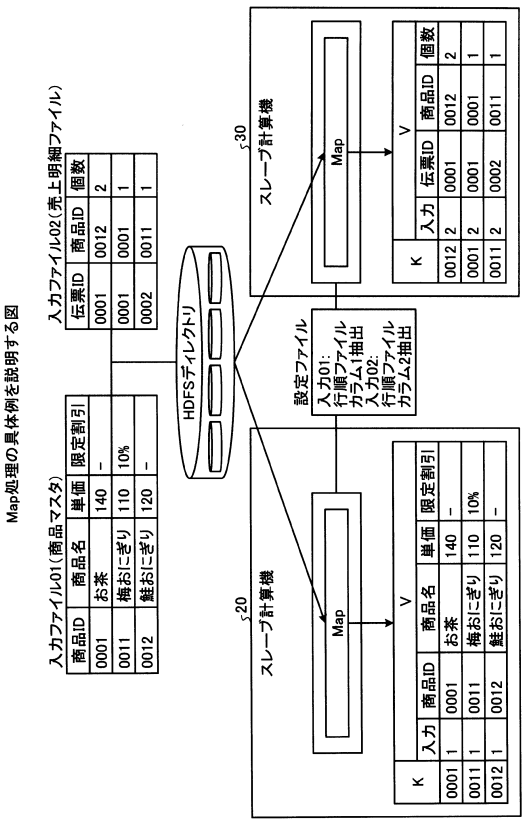


【図１２】

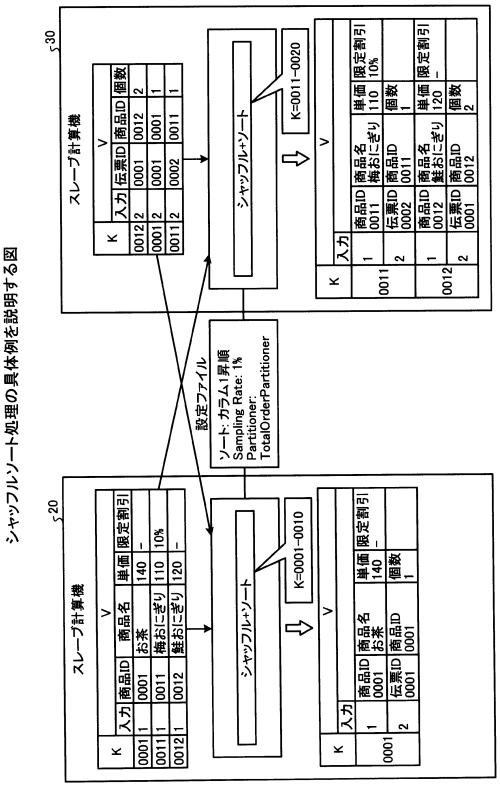
実施例2に係る出力読み込みスレッドの起動処理の流れを示すフローチャート



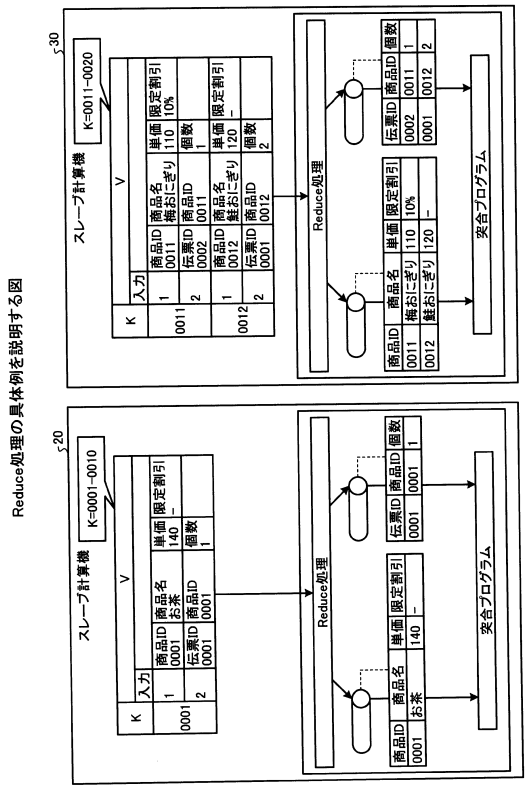
【図１３】



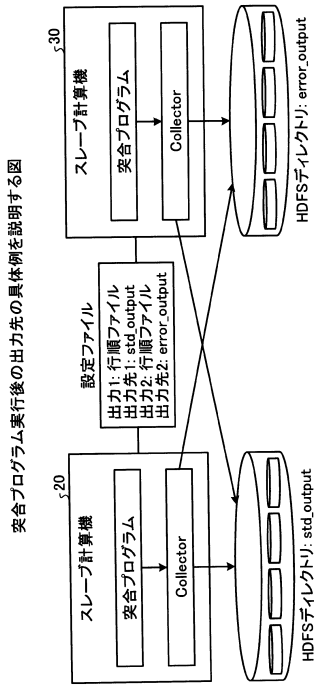
【図１４】



【図１５】

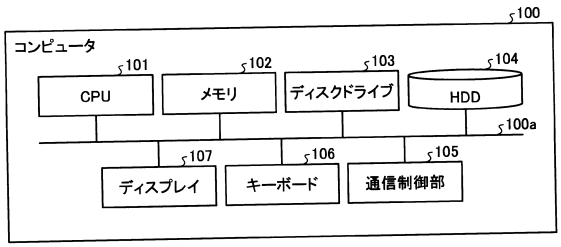


【図１６】



【図１７】

ハードウェア構成例を示す図



フロントページの続き

(58)調査した分野(Int.Cl., ＤＢ名)
Ｇ０６Ｆ　１７／３０