(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2006-190022
(P2006-190022A)
(43)公開日　平成18年7月20日(2006.7.20)

(51)Int.Cl.			F I			テーマコード（参考）
G06F	9/54	(2006.01)	G06F	9/46	480B	
G06F	9/46	(2006.01)	G06F	9/46	430	

審査請求 未請求　請求項の数 3　OL　（全 11 頁）

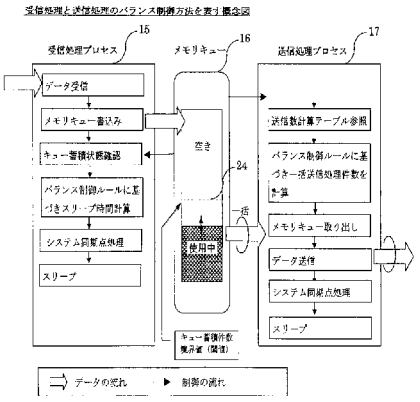
(21)出願番号　特願2005-653(P2005-653)
(22)出願日　平成17年1月5日(2005.1.5)

(71)出願人　000233055
日立ソフトウエアエンジニアリング株式会社
神奈川県横浜市鶴見区末広町一丁目１番４３
(74)代理人　100093492
弁理士　鈴木　市郎
(74)代理人　100078134
弁理士　武　顕次郎
(74)代理人　100087354
弁理士　市村　裕宏
(72)発明者　上戸　康司
東京都品川区東品川４丁目１２番７号　日立ソフトウエアエンジニアリング株式会社内

最終頁に続く

(54)【発明の名称】送受信処理のフロー制御方法

(57)【要約】
【課題】受信、送信の各プロセスの適切なバランス制御を行い、高負荷時には一括処理を行うことにより、処理時間を短縮してスループットの向上を図る。
【解決手段】受信処理プロセス１５と送信処理プロセス１７との間にトランザクションを受け渡すためのメモリキュー１６を配置し、双方のプロセスがキューのリソース使用状態を確認し、キューの総使用可能ブロック数に対する「キューの蓄積率」や前回、今回の各処理時におけるキューの状態を比較して算出した「キューの増減件数」や「キューの増加率」と言った指標値をもとにプロセス間のバランスを制御する。また、一括して処理するトランザクション件数を変動させ、Ｉ／Ｏ同期処理回数とＩ／Ｏ同期処理に掛かるＣＰＵ処理時間を低減するように構成される。これにより、高負荷時において、ＣＰＵリソースを各プロセスに適確に分配することができ、スループットの向上を計ることができる。
【選択図】　図２



【特許請求の範囲】
【請求項１】
受信処理プロセスと送信処理プロセスとの間にメモリキューを備え、前記受信処理プロセスと送信処理プロセスとの間で受信データに対する処理を施したデータを受け渡すコンピュータシステムにおける送受信処理のフロー制御方法において、前記メモリキューの使用状況に応じて、１回の送受信処理で扱う送信件数を変動させ、送受信のシステム同期点処理に費やすトータル時間を調整することを特徴とする送受信処理のフロー制御方法。
【請求項２】
前記１回の送受信処理で扱う送信件数を、メモリキューの蓄積率とキュー蓄積率の増加率とをパラメータとして決定することを特徴とする請求項１記載の送受信処理のフロー制御方法。
【請求項３】
受信処理プロセスと送信処理プロセスとの間にメモリキューを備え、前記受信処理プロセスと送信処理プロセスとの間で受信データに対する処理を施したデータを受け渡すコンピュータシステムにおける送受信処理のフロー制御方法において、前記受信プロセスは、メモリキューに蓄積されているデータの増減件数の指数乗に比例して自身のスリープ時間を調整すると共に、自受信プロセスをスリープさせるタイミングをメモリキューの蓄積件数により決定し、前記送信プロセスは、メモリキューの蓄積率とキュー蓄積率の増加率とに基づいて、一括して送信する送信件数を、予め定めた送信数計算テーブルにより決定し、メモリキューの蓄積件数に応じて自送信プロセスのスリープ時間を調整することを特徴とする送受信処理のフロー制御方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、送受信処理のフロー制御方法に係り、特に、オンラインシステムのトランザクション処理における送受信処理のフロー制御方法に関する。
【背景技術】
【０００２】
オンラインシステムにおけるトランザクション量の増加に対応するための方法に関する従来技術として、例えば、特許文献１、特許文献２等に記載された技術が知られている。
【０００３】
特許文献１に記載の従来技術は、ノード間で処理を分散させる方法に関するもので、各ノードのＣＰＵ負荷値、メモリ負荷値、ディスク負荷値と言ったリソース資源の負荷値を監視して、処理負荷の少ないノードに負荷を分散させるというものである。
【０００４】
また、特許文献２に記載の従来技術は、高負荷時に受信処理を優先して受信メッセージの取りこぼしを回避し、受信メッセージが一定時間途絶えた時点で、メッセージ受信連絡を継続して行い、その時点で内部処理を行うというものである。
【０００５】
また、ノード内における各プロセス間でのトランザクション処理のバランス制御を行う方法に関する技術として、受信処理専用プロセスと送信処理専用プロセスとを保有し、各プロセスがトラフィック量に応じて動作した自身のＣＰＵ使用時間を監視して、適宜スリープ制御することにより、各プロセス間でスリープ中のＣＰＵを受け渡す方法が知られている。そして、この技術は、特許文献２に記載されているような、各プロセス間にメッセージキューを配置してメッセージを受け渡す手段を実装している。
【特許文献１】特開２０００－１３７６９２号公報
【特許文献２】特開平１１－２８２７８３号公報
【発明の開示】
【発明が解決しようとする課題】
【０００６】

前述した特許文献１に記載の従来技術は、ノードを増加してトランザクション量の増加に対処するというものであり、保有するハードウェア資産のみでは処理能力を上げることができないという問題点を有しており、また、特許文献２に記載の従来技術は、高負荷が継続すると受信メッセージが大量に蓄積されてしまい、メモリ資源を圧迫する上、未処理のメッセージがノード内部に滞在する時間が増して、処理のリアルタイム性が失われてしまうという問題点を有している。

【０００７】

また、一般のノード内における各プロセス間でのトランザクション処理バランス制御方法の技術は、一定のトランザクション量に対してはＣＰＵを効率的にプロセス間で配分しあうことができるが、マシンのＣＰＵ能力を超える高負荷状態が継続した場合、トランザクションの処理が追いつかず、システム内部に未処理メッセージが蓄積されていってしまうという問題点を有している。

【０００８】

本発明の目的は、前述した従来技術の問題点を解決し、受信、送信の各プロセスの適切なバランス制御を行い、高負荷時には一括処理を行うことにより、処理時間を短縮してスループットを向上させることができるようにしたオンラインシステムのトランザクション処理における送受信処理のフロー制御方法を提供することにある。

【課題を解決するための手段】

【０００９】

本発明によれば前記目的は、受信処理プロセスと送信処理プロセスとの双方が、受信したトランザクションを蓄えるメモリキューのリソース状態を確認して、キューの総使用可能ブロック数に対する「キューの蓄積率」や前回処理時と今回処理時とにおけるキューの状態を比較して算出した「キューの増減件数」や「キューの増加率」と言った指標値をもとにプロセス間のバランスを制御することにより達成される。

【００１０】

また、前記目的は、キューに蓄積されたデータによって使用されているブロック数が一定値を超えた場合に、一括して処理するデータの処理件数を前述のバランス制御に基づいて変動させ、Ｉ／Ｏ同期処理回数とＩ／Ｏ同期処理とに掛かるＣＰＵ処理時間を低減することにより達成される。

【００１１】

すなわち、本発明によれば前記目的は、受信処理プロセスと送信処理プロセスとの間にメモリキューを備え、前記受信処理プロセスと送信処理プロセスとの間で受信データに対する処理を施したデータを受け渡すコンピュータシステムにおける送受信処理のフロー制御方法において、前記メモリキューの使用状況に応じて、１回の送受信処理で扱う送信件数を変動させ、送受信のシステム同期点処理に費やすトータル時間を調整することにより達成される。

【００１２】

さらに、前記目的は、受信処理プロセスと送信処理プロセスとの間にメモリキューを備え、前記受信処理プロセスと送信処理プロセスとの間で受信データに対する処理を施したデータを受け渡すコンピュータシステムにおける送受信処理のフロー制御方法において、前記受信プロセスが、メモリキューに蓄積されているデータの増減件数の指数乗に比例して自身のスリープ時間を調整すると共に、自受信プロセスをスリープさせるタイミングをメモリキューの蓄積件数により決定し、前記送信プロセスが、メモリキューの蓄積率とキュー蓄積率の増加率とに基づいて、一括して送信する送信件数を、予め定めた送信数計算テーブルにより決定し、メモリキューの蓄積件数に応じて自送信プロセスのスリープ時間を調整することにより達成される。

【発明の効果】

【００１３】

本発明によれば、受信処理プロセスと送信処理プロセスとの双方が、トランザクションを処理する都度、キューの状態を監視して適切なバランス制御を行うため、キューを圧迫

するのを未然に防止でき、システムを健全な状態に保つことができる。

【００１４】

また、本発明によれば、通常負荷時には、着信したトランザクションを順次処理することにより、トランザクションがノード内部に滞在する時間を短くすることを優先して作用させることができ、高負荷時には、負荷量に応じた一括トランザクション方法によりＩ／Ｏ処理回数を少なくして、スループットが向上するように作用させることができるので、トラフィック状況に応じて最適なトランザクション処理を実現することができる。

【発明を実施するための最良の形態】

【００１５】

以下、本発明による送受信処理のフロー制御方法の実施形態を図面により詳細に説明する。

【００１６】

図１は本発明の一実施形態による送受信処理のフロー制御を実施するコンピュータシステムの構成を示すブロック図である。図１において、１１、１１’はＬＡＮ、１２はＯＳ、１３はオンライントランザクション制御処理プログラム（ＯＬＴＰ）、１４はＴＣＰ／ＩＰ通信ソフトウェア、１５は受信処理プロセス、１６はメモリキュー、１７は送信処理プロセス、１８はコンピュータシステムである。

【００１７】

図１に示すコンピュータシステム１８は、オンライントランザクション処理を実施するものであり、受信したトランザクションデータをノード（コンピュータシステム１８）の内部で加工処理を行って送信する処理を実行するものである。内部処理の例としては、専用プロトコル変換、回線サービス変換、データ暗号化、データ分析、データ振分サービス等を挙げることができる。コンピュータシステム１８を用いる具体的なシステム構成の例としては、例えば、オンライントランザクション受発注システムとして、複数の機関投資家が持つ端末等から受けた株価注文データを、内部で専用プロトコルに変換して証券会社のサーバに送る等の形態である。

【００１８】

前述したような内部処理と送受信の処理とを行うコンピュータシステム１８は、ＯＳ１２の下で動作するオンライントランザクション制御処理プログラム１３（以下、ＯＬＴＰという）とＴＣＰ／ＩＰ通信対応ソフトウェア１４とを用いたソフトウェアにより構成される。ＯＬＴＰ１３は、データベースの一貫性を保つため、トランザクション更新前と更新後の状態をトランザクション処理単位に保存する。これを同期点処理と言うが、この同期点処理に費やす時間は、ファイルＩ／Ｏ処理を伴うことにより非常に大きいことが知られており、また、１件単位に同期点処理を行うより、複数件を纏めて同期点処理を行う方が、その処理における全体の総処理時間は短くなり、処理効率が向上するという特性がある。

【００１９】

そして、図１に示すコンピュータシステム１８は、前述したソフトウェア構成の下に受信処理を行う受信処理プロセス１５と送信処理を行う送信処理プロセス１７とからなり、両プロセス間にデータを受け渡すためのメモリキュー１６を配置して構成されたバッファリング方式のものである。そして、受信処理プロセス１５は、ＬＡＮ１１から到来するトランザクションを受信し、受信したデータをメモリキュー１６に一旦格納する。送信処理プロセス１７は、メモリキュー１６に格納されたデータをＬＡＮ１１’に送信する。なお、ＬＡＮ１１、１１’は、異なるものであっても、同一のものであってもよい。前述したような送受信フローの途中で、ＯＬＴＰ１３は、受信処理プロセス１５または送信処理プロセス１７と連携して、受信したトランザクションに対して前述で挙げたような各種の内部処理を加える。

【００２０】

図２は受信処理プロセスと送信処理プロセスとのバランス制御について説明する図である。この図には、受信処理プロセス１５が行う処理フローの概念と送信処理プロセス１７

が行う処理フローの概念とを示しており、これらの詳細については、図３、図４に示すフローにより後述する。ここでは、各プロセスでの処理について簡単に説明する。

【００２１】

受信処理プロセス１５は、データを受信してそのデータをメモリキュー１６に書き込んだ後、メモリキュー１６の蓄積状態（蓄積ブロック数）を確認して、図３に示す受信処理プロセスのバランス制御フローに基づいてスリープ時間を計算する。その後、受信処理プロセス１５は、トランザクション処理状態を保存するためのシステム同期点処理を行った後、計算したスリープ時間分スリープする。

【００２２】

また、送信処理プロセス１７は、図４に示す送信処理プロセスのバランス制御フローに基づき、メモリキュー１６の蓄積状態（蓄積ブロック数）を確認して一括送信処理件数を計算する。その後、送信処理プロセス１７は、一括送信処理件数分のデータをメモリキュー１６から取り出して、データ送信を行った後、トランザクション処理状態を保存するためのシステム同期点処理を行い、判断基準に応じてスリープする。

【００２３】

前述した受信処理プロセスと送信処理プロセスとのバランス制御において、内部処理は、受信処理プロセスでのデータ受信の処理とメモリキュー書き込みの処理との間、または、送信処理プロセスでのメモリキュー取り出しの処理とデータ送信の処理との間で行われる。

【００２４】

図３は受信処理プロセス１５での処理動作の詳細を説明するフローチャートであり、次に、これについて説明する。なお、ここで説明する処理は、図２に示す受信処理プロセス１５での「キュー蓄積状態確認」、「バランス制御ルールに基づきスリープ時間計算」、「スリープ」の処理である。

【００２５】

（１）まず、受信したデータをメモリキュー１６に書き込むために新たに使用したメモリキュー１６のブロック数を確認する（ステップ３０１）。

【００２６】

（２）受信処理プロセスと送信処理プロセスとのバランス制御に必要な各種の情報を格納している共用メモリから蓄積ブロック数３１８と前回スリープ時間３１７とを読み込む。前回スリープ時間３１７は、受信処理プロセス１５が前回処理時にスリープした時間を表す変数である。蓄積ブロック数３１８は、メモリキューの蓄積状況をブロック数で表したもので、受信処理プロセスと送信処理プロセスとの双方により更新される変数である（ステップ３０２）。

【００２７】

（３）ステップ３０２の処理で読み出した蓄積ブロック数に今回受信したデータで使用されるブロック数を加算して、現在蓄積ブロック数を計算する（ステップ３０３）。

【００２８】

（４）受信処理プロセスが前回の処理時点で確認した蓄積ブロック数３２１を読み込む（ステップ３０４）。

【００２９】

（５）ステップ３０３の処理で計算したメモリキューの現在蓄積ブロック数から、ステップ３０４の処理で読み込んだ前回の処理時点の蓄積ブロック数３２１を引き算して、ブロック数の増減（差分）を計算する（ステップ３０５）。

【００３０】

（６）次に、メモリキューの現在蓄積ブロック数が予め設定した閾値（境界値）３１９を越えたか否かを確認する。この閾値（境界値）３１９は、初期に設定した後は更新されない値であり、メモリキューの蓄積ブロック数が一定値以上を超えた場合にフロー制御を発動させるために定める値であり、図２に示すキュー蓄積件数境界値（閾値）２４に該当する。この値は、メモリキューの総使用可能ブロック数に対して割合が大き過ぎると、フロ

ー制御が殆ど発動されず、ＣＰＵリソースの有効な配分が行われ難くなるが、逆に小さ過ぎると、フロー制御の発動が頻繁に行われ、却ってシステムリソースの有効活用を抑制することがあるので、これを考慮して設定する（ステップ３０６）。

【００３１】

（７）ステップ３０６の確認で、メモリキューの現在蓄積ブロック数が閾値を越えていた場合、次に、ステップ３０５の処理で計算したブロック増減（差分）を評価して、蓄積ブロック数が増加または変動しない傾向にあるか減少する傾向にあるかを判断する（ステップ３０７）。

【００３２】

（８）ステップ３０７の判断で、蓄積ブロック数が増加または変動しない傾向にあった場合、受信処理プロセスに対して、今回スリープさせる時間を、例えば、

$$\text{今回スリープ時間［秒］} = \text{前回スリープ時間} + \text{増減ブロック数}^2 \times \text{スリープ単価}$$

として算出する（ステップ３０８）。

【００３３】

（９）ステップ３０７の判断で、蓄積ブロック数が減少する傾向にあった場合、受信処理プロセスに対して、今回スリープさせる時間を、例えば、

$$\text{今回スリープ時間［秒］} = \text{前回スリープ時間} - \text{増減ブロック数}^2 \times \text{スリープ単価}$$

として算出する（ステップ３０９）。

【００３４】

前述したステップ３０８、３０９での処理における今回スリープ時間の算出に使用するスリープ単価３２０は、マシン性能に応じてユーザチューニングを可能とする予め定められた値であり、予め定めて初期に設定した後は更新されない値である。また、前述のスリープ時間の計算式は、増減ブロック数の指数乗（前述の例では２乗としているが、任意数乗に設定されてよい）として、急速な増減傾向に対処したものである。また、算出される今回スリープ時間に上限値または下限値を設けてもよい。

【００３５】

なお、受信処理プロセス側をスリープさせる目的は、送信処理プロセス側へのＣＰＵリソース配分率を増加させ、送信処理プロセスによるメモリキューからのデータ取り出し処理を優先させるためである。

【００３６】

（10）ステップ３０８、３０９での今回スリープ時間算出の処理の後、算出した今回スリープ時間で、前回スリープ時間３１７のパラメータを更新する（ステップ３１０）。

【００３７】

（11）次に、蓄積ブロック数３１８、前回処理時の蓄積ブロック数３２１の両パラメータを、ステップ３０３の処理で求めた現在蓄積ブロック数に更新する（ステップ３１１）。

【００３８】

（12）その後、ステップ３１０、ステップ３１１の処理で更新した各パラメータの値を共用メモリに書き込んで保存し、ステップ３０８またはステップ３０９の処理で算出した今回スリープ時間分、受信処理プロセスをスリープさせて、ここでの処理を終了する（ステップ３１２、３１３）。

【００３９】

（13）ステップ３０６の判定で、メモリキューの現在蓄積ブロック数が閾値を越えていなかった場合、前回スリープ時間３１７のパラメータをリセット（０を設定）する。なお、前回スリープ時間３１７の初期値は０であるとする（ステップ３１４）。

【００４０】

（14）次に、蓄積ブロック数３１８、前回処理時の蓄積ブロック数３２１の両パラメータを、ステップ３０３の処理で求めた現在蓄積ブロック数に更新する（ステップ３１５）。

【００４１】

（15）その後、ステップ３１４、ステップ３１５の処理で更新した各パラメータの値を共用メモリに書き込んで保存し、ここでの処理を終了する（ステップ３１６）。

【００４２】
図４は送信処理プロセス１７での処理動作の詳細を説明するフローチャートであり、次に、これについて説明する。なお、ここで説明する処理は、図２に示す「送信処理プロセス１７での「送信数計算テーブル参照」、「バランス制御ルールに基づき一括送信処理件数を計算」、「スリープ」の処理である。
【００４３】
（１）まず、共用メモリより蓄積ブロック数４１１とキューの総使用可能ブロック数４１３とを読み込む。蓄積ブロック数４１１は、メモリキューの蓄積状況をブロック数で表したもので、受信処理プロセスと送信処理プロセスとの双方から更新される変数である。また、キューの総使用可能ブロック数４１３は、メモリキューに割当てるリソース容量から決定されるブロック数であり、予め定めて設定した後は更新されない値である（ステップ４０１）。
【００４４】
（２）次に、今回処理時のメモリキューの蓄積率（キューの総使用可能ブロック数に対する使用ブロック数）を、
（蓄積ブロック数／総使用可能ブロック数）×１００
として計算する（ステップ４０２）。
【００４５】
（３）次に、共用メモリから前回処理時のキューの蓄積率４１２の値を読み込む。なお、前回処理時のキューの蓄積率４１２の値は、１～１００（≠０）の値であるとする（ステップ４０３）。
【００４６】
（４）次に、メモリキューの蓄積率の増加率（前回処理時のメモリキュー蓄積率に対する今回処理時のメモリキュー蓄積率の増分）を計算する。なお、ここでの計算の詳細とその考え方については、図５を参照して後述する（ステップ４０４）。
【００４７】
（５）その後、ステップ４０２の処理で計算した今回処理時のメモリキューの蓄積率と、ステップ４０４の処理で計算したメモリキュー増加率とに基づいて、図６に示す共用メモリ内の送信数計算テーブルを参照し、１つのトランザクションで処理する件数（一括送信処理件数）を決定する。なお、送信数計算テーブルの詳細は、図６を参照して後述する（ステップ４０５）。
【００４８】
（６）次に、前回処理時のキューの蓄積率４１２のパラメータに対して、ステップ４０２の処理で計算した今回処理時のメモリキューの蓄積率により共用メモリを更新する（ステップ４０６）。
【００４９】
（７）次に、蓄積ブロック数４１１のパラメータに対して、ステップ４０５の処理で決定した一括送信処理件数を引き算した値を、新たな蓄積ブロック数４１１として共用メモリを更新する（ステップ４０７）。
【００５０】
前述した処理を実行することにより、一括送信処理件数を決定することができ、引き続き、決定された一括送信処理件数に基づいたスリープ時間を決定する処理が行われるので、つぎに、これについて説明する。
【００５１】
（８）ステップ４０７の処理で更新した蓄積ブロック数４１１のパラメータと、通常時の一括送信処理件数４１４とを共用メモリから読み込む。通常時の一括送信処理件数４１４は、予め定めて設定した後は更新されない値であり、送信処理プロセスが１回の処理でメモリキューから何件のデータ（ブロック）を取り出して処理をするかを表す基準値である。（ステップ４０８）。
【００５２】

（９）次に、蓄積ブロック数４１１と通常時の一括送信処理件数４１４とを比較し、蓄積ブロック数４１１が通常時の一括送信処理件数４１４より大きいか否かを判定する。なお、ここでの比較判定は、通常時の一括送信処理件数（ブロック数）＝通常時の一括送信処理件数１データ当り１ブロックを使用するものとして行う（ステップ４０９）。

【００５３】

（10）ステップ４０９の判定で、蓄積ブロック数４１１が、通常時の一括送信処理件数４１４に満たない場合、共用メモリよりスリープ時間４１５を読み込み、送信処理プロセスを、その時間だけスリープさせることとして、ここでの処理を終了する。このスリープ時間４１５は、予め定めて設定した後は更新されない値である（ステップ４１０）。

【００５４】

（11）ステップ４０９の判定で、蓄積ブロック数４１１が、通常時の一括送信処理件数４１４より大きかった場合、送信処理プロセスをスリープさせないこととして、ここでの処理を終了する。

【００５５】

送信処理プロセス側をスリープさせる目的は、受信処理プロセス側へのＣＰＵリソース配分率を増やして、受信処理プロセスによるメモリキューへの書込み処理を優先させるためである。これにより、受信処理プロセス側がメモリキューに一定件数を蓄積した時点で、送信処理プロセス側は一括してトランザクション送信処理を行うことができる。なお、データ送信処理で、メモリキューに蓄積されているデータ件数が、通常時の一括送信処理件数４１４に満たない場合、蓄積されているデータ件数の全ての処理を行う。

【００５６】

図５はメモリキューの蓄積率と増加率との考え方を説明すると共に、その計算式を示す図である。ここで説明する計算式は、前述で説明した図４に示すフローのステップ４０４の処理で使用される。

【００５７】

メモリキューの前回までの蓄積分としてのブロック数を表す前回蓄積ブロック数を（ｃ）、今回蓄積分としてのブロック数を前回蓄積ブロック数に加えた今回蓄積ブロック数を（ｂ）、キューの総使用可能ブロック数を（ａ）とすると、メモリキューの使用状況は、図５（ａ）に示すように表すことができる。そして、一般に、キュー蓄積率［％］は、図５（ｂ）に示すように、

キュー蓄積率［％］＝蓄積使用ブロック数／キューの総使用可能ブロック数×100

として表すことができる。

【００５８】

このことから、前回処理時のキュー蓄積率は、

前回処理時のキュー蓄積率

＝蓄積ブロック数（ｃ）／キューの総使用可能ブロック数（ａ）×100 ……（１）

として求められ、また、今回処理時のキュー蓄積率は、

今回処理時のキュー蓄積率

＝蓄積ブロック数（ｂ）／キューの総使用可能ブロック数（ａ）×100 ……（２）

として求めることができる。

【００５９】

式（１）、式（２）からメモリキューの蓄積率の増加率は、

キュー増加率［％］

＝(今回処理時のキュー蓄積率－前回処理時のキュー蓄積率)／前回処理時のキュー蓄積率

として求めることができる。

【００６０】

図６は送信数計算テーブルの構成例を示す図である。このテーブルは、図４に示すフローのステップ４０５の処理で、１トランザクションで処理する件数（一括送信処理件数）を決定するために参照される。

【００６１】

図６に示す送信数計算テーブルの値は、通常時の一括送信処理件数を仮に１０件として定め、この値を基準値とし、基準値に対して増加率と蓄積率の重みを乗じたものである。増加率と蓄積率の重みを、図６のカッコ内に示した値として、この値を用いて、

送信数計算テーブルの値
＝基準値×増加率×蓄積率

として算出したものである。

【００６２】

但し、このテーブルは、システムの特性に応じてチューニング可能としておく。また、このテーブルは、データ１件あたりで使用するブロック数を１ブロックとして表現し、図の説明を簡易化して示している。なお、蓄積率、増加率は、図４に示すフローの処理により算出されたものである。

【００６３】

前述した本発明の実施形態における各処理は、処理プログラムとして構成することができ、この処理プログラムは、ＨＤ、ＤＡＴ、ＦＤ、ＭＯ、ＤＶＤ－ＲＯＭ、ＣＤ－ＲＯＭ等の記録媒体に格納して提供することができる。

【図面の簡単な説明】

【００６４】

【図１】本発明の一実施形態による送受信処理のフロー制御を実施するコンピュータシステムの構成を示すブロック図である。

【図２】受信処理プロセスと送信処理プロセスとのバランス制御について説明する図である。

【図３】受信処理プロセス１５での処理動作の詳細を説明するフローチャートである。

【図４】送信処理プロセス１７での処理動作の詳細を説明するフローチャートである。

【図５】メモリキューの蓄積率と増加率との考え方を説明すると共に、その計算式を示す図である。

【図６】送信数計算テーブルの構成例を示す図である。

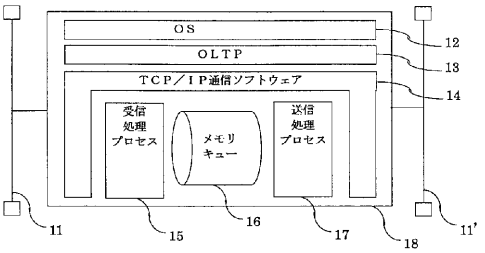
【符号の説明】

【００６５】

１１、１１’　ＬＡＮ
１２　ＯＳ
１３　オンライントランザクション制御処理プログラム（ＯＬＴＰ）
１４　ＴＣＰ／ＩＰ通信ソフトウェア
１５　受信処理プロセス
１６　メモリキュー
１７　送信処理プロセス
１８　コンピュータシステム

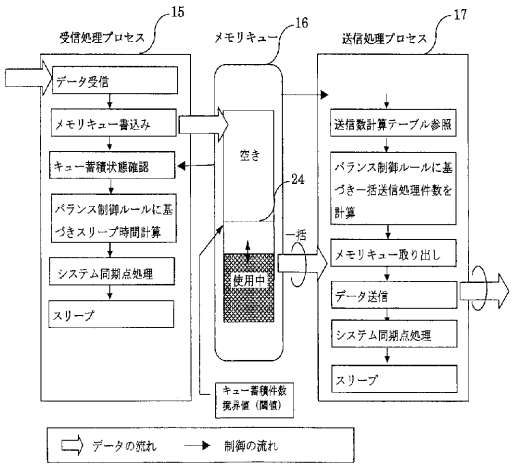
【図１】

システム構成図



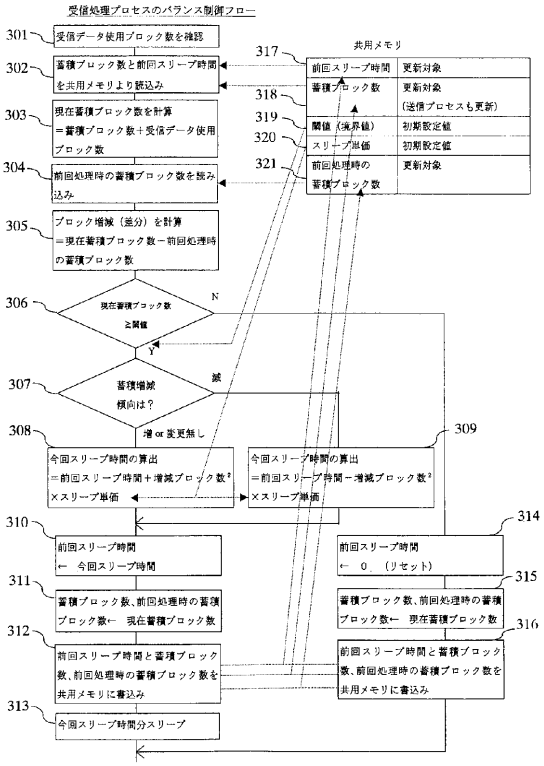
【図２】

受信処理と送信処理のバランス制御方法を表す概念図



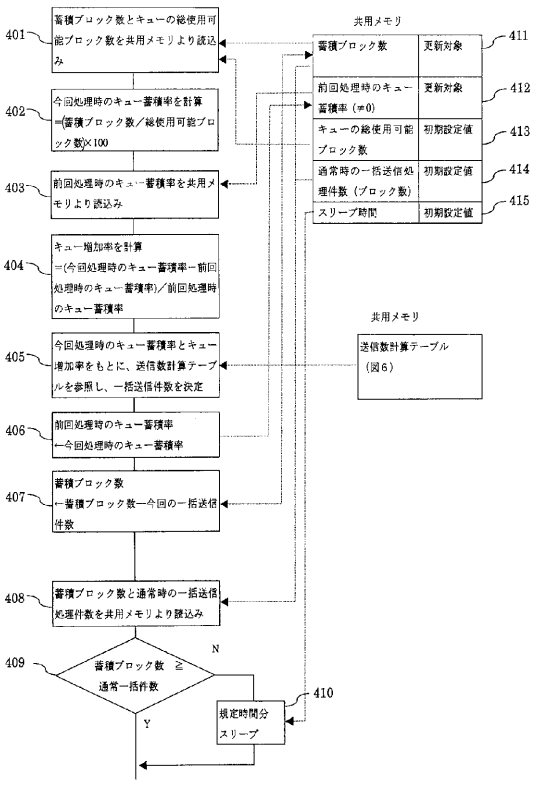
【図３】

受信処理プロセスのバランス制御フロー



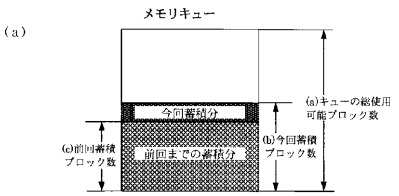
【図４】

送信処理プロセスのバランス制御フロー



【図５】

メモリキュー蓄積率と増加率の考え方



（b）

キュー蓄積率［%］＝蓄積ブロック数 ／ キューの総使用可能ブロック数 × 100

前回処理時のキュー蓄積率＝前回蓄積ブロック数(c) ／ キューの総使用可能ブロック数(a) × 100

今回処理時のキュー蓄積率＝今回蓄積ブロック数(b) ／ キューの総使用可能ブロック数(a) × 100

キュー増加率［%］＝（今回処理時のキュー蓄積率 － 前回処理時のキュー蓄積率）／ 前回処理時のキュー蓄積率

【図６】

送信数計算テーブル

件＝ブロック

増加率＼蓄積率		~10% (1)	~20% (2)	~50% (5)	~80% (8)	~100% (10)	備考
~20%減	(0.8)	8 件	16 件	40 件	64 件	80 件	
~10%減	(0.9)	9 件	18 件	45 件	72 件	90 件	
10%減~10%増	(1.0)	10 件	20 件	50 件	80 件	100 件	
10%増~	(1.1)	11 件	22 件	55 件	88 件	110 件	
20%増~	(1.2)	12 件	24 件	60 件	96 件	120 件	

通常時の一括送信処理件数（基準値）

（ ）内に表した数値：一括送信処理件数を算出するのに使用する増加率と蓄積率の重み

フロントページの続き

(72)発明者　肱岡　克典
東京都品川区東品川４丁目１２番７号　日立ソフトウエアエンジニアリング株式会社内
(72)発明者　鈴木　大介
東京都品川区東品川４丁目１２番７号　日立ソフトウエアエンジニアリング株式会社内