(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4177960号
(P4177960)

(45)発行日　平成20年11月5日(2008.11.5)　(24)登録日　平成20年8月29日(2008.8.29)

(51)Int.Cl.		F I		
G06F 12/00	(2006.01)	G06F	12/00	591
G06F 9/45	(2006.01)	G06F	9/44	320C
G06F 9/06	(2006.01)	G06F	9/06	410B

請求項の数 16　(全 13 頁)

(21)出願番号	特願2000-525818 (P2000-525818)
(86)(22)出願日	平成10年12月17日 (1998.12.17)
(65)公表番号	特表2001-527237 (P2001-527237A)
(43)公表日	平成13年12月25日 (2001.12.25)
(86)国際出願番号	PCT/US1998/026769
(87)国際公開番号	WO1999/032978
(87)国際公開日	平成11年7月1日 (1999.7.1)
審査請求日	平成17年10月19日 (2005.10.19)
(31)優先権主張番号	08/994,098
(32)優先日	平成9年12月19日 (1997.12.19)
(33)優先権主張国	米国(US)

(73)特許権者　500046438
マイクロソフト　コーポレーション
アメリカ合衆国　ワシントン州　９８０５２－６３９９　レッドモンド　ワン　マイクロソフト　ウェイ
(74)代理人　100089705
弁理士　社本　一夫
(74)代理人　100140109
弁理士　小野　新次郎
(74)代理人　100075270
弁理士　小林　泰
(74)代理人　100080137
弁理士　千葉　昭男
(74)代理人　100096013
弁理士　富田　博行

最終頁に続く

(54)【発明の名称】増分不要情報収集

(57)【特許請求の範囲】

【請求項１】

プロセッサ・ユニットとシステム・メモリと複数のプログラム・モジュールとを含むコンピュータ・システムであって、前記システム・メモリの少なくとも一部はヒープを含み、前記ヒープからオブジェクトが動的に配分され、前記複数のプログラム・モジュールは、１又は複数のオブジェクトのために前記ヒープからメモリを動的に配分する１又は複数のプログラム又は機能と、前記１又は複数のオブジェクトのために前記１又は複数のプログラム又は機能によって動的に配分されたメモリを再生することを周期的に試みる際に前記ヒープの全体を走査するメイン・ガーベッジ・コレクタとを含む、コンピュータシステムにおいて、前記メイン・ガーベッジ・コレクタを待機するのではなく、前記１又は複数のオブジェクトを配分した前記プログラム又は機能が終了した又は前記プログラム又は機能から出た直後に、前記１又は複数のオブジェクトと関連付けられている動的に配分されたメモリの少なくとも一部を再生する方法であって、

配分されたある１つのオブジェクトに対して、前記オブジェクトを最初に動的に配分したプログラム又は機能に対して前記オブジェクトがローカルであるかどうかを示すマーカを含む追加的情報を前記オブジェクトの中に記憶するステップと、

前記オブジェクトを最初に動的に配分した前記プログラム又は機能が終了した又は前記プログラム又は機能から出た直後であって、前記メイン・ガーベッジ・コレクタがメモリの再生を試みる前に、前記プログラム又は機能によって配分された少なくとも１つのオブジェクトについて前記追加的情報を走査し、前記少なくとも１つのオブジェクトが前記プ

ログラム又は機能に対してローカルであることを示すマーカを前記少なくとも１つのオブジェクトが有するかどうかを判断するステップと、
前記少なくとも１つのオブジェクトが前記プログラム又は機能に対してローカルである場合には、前記少なくとも１つのオブジェクトと関連付けられている動的に配分されたメモリを再生し、前記少なくとも１つのオブジェクトが前記プログラム又は機能に対してローカルでない場合には、前記少なくとも１つのオブジェクトと関連付けられている動的に配分されたメモリを再生するステップを延期するステップと、
を含むことを特徴とする方法。
【請求項２】
請求項１記載の方法において、
オブジェクトを配分する前に、オブジェクトを動的に配分する開始ヒープ位置をセーブするステップと、
動的に配分される最後のオブジェクトの終了ヒープ位置をセーブするステップと、
前記開始ヒープ位置と前記終了ヒープ位置との間で、前記ヒープから１又は複数のオブジェクトをフェッチし、前記フェッチされた１又は複数のオブジェクトのそれぞれについて前記追加的情報を走査して、前記フェッチされた１又は複数のオブジェクトのそれぞれがローカルであるかどうかを判断するステップと、
を更に含むことを特徴とする方法。
【請求項３】
請求項２記載の方法において、前記追加的情報はスレッド識別子とスタック識別子とを含み、この方法は、更に、
前記ヒープからフェッチされた前記１又は複数のオブジェクトのそれぞれについて前記追加的情報を走査し、前記ヒープからフェッチされた前記１又は複数のオブジェクトの中のどのオブジェクトが前記プログラム又は機能によって配分されたかを判断するステップを含むことを特徴とする方法。
【請求項４】
請求項１記載の方法において、
前記少なくとも１つのオブジェクトを動的に配分した前記プログラム又は機能が実行されている間、前記少なくとも１つのオブジェクトを連続的にモニタし、前記少なくとも１つのオブジェクトが前記プログラム又は機能の外部で参照されるかどうかを判断するステップと、
前記少なくとも１つのオブジェクトが前記プログラム又は機能の外部で参照される場合には、前記少なくとも１つのオブジェクトがローカルではないことを示すように前記マーカを設定するステップと、
を更に含むことを特徴とする方法。
【請求項５】
請求項１記載の方法において、前記少なくとも１つのオブジェクトは前記プログラム又は機能に対してローカルであり、前記メイン・ガーベッジ・コレクタの動作の前に再生されることを特徴とする方法。
【請求項６】
請求項１記載の方法において、前記少なくとも１つのオブジェクトは参照する側のオブジェクトにおいて参照され、この方法は、更に、
（ｉ）前記参照する側のオブジェクトと前記少なくとも１つのオブジェクトとが異なるプログラム又は機能によって配分されるか、又は、（ｉｉ）前記参照する側のオブジェクトが前記プログラム又は機能の外部で参照される場合には、前記少なくとも１つのオブジェクトはローカルではないことを示すように前記少なくとも１つのオブジェクトのマーカを設定するステップを含むことを特徴とする方法。
【請求項７】
請求項１記載の方法において、前記プログラム又は機能はジャバ仮想計算機の内部で実行されることを特徴とする方法。

【請求項８】
請求項１ないし請求項７のいずれかの請求項に記載の方法に含まれるステップをコンピュータに実行させる命令から構成されるコンピュータ・プログラムが記憶されたコンピュータ可読な記憶媒体。
【請求項９】
プロセッサ・ユニットとシステム・メモリと複数のプログラム・モジュールとを含むコンピュータ・システムであって、前記システム・メモリの少なくとも一部はヒープを含み、前記ヒープからオブジェクトが動的に配分され、前記複数のプログラム・モジュールは、１又は複数のオブジェクトのために前記ヒープからメモリを動的に配分する１又は複数のプログラム又は機能と、前記１又は複数のオブジェクトのために前記１又は複数のプログラム又は機能によって動的に配分されたメモリを周期的に再生することを試みる際に前記ヒープの全体を走査するメイン・ガーベッジ・コレクタとを含む、コンピュータシステムにおいて、前記メイン・ガーベッジ・コレクタを待機するのではなく、前記１又は複数のオブジェクトを配分した前記プログラム又は機能が終了した又は前記プログラム又は機能から出た直後に、前記１又は複数のオブジェクトと関連付けられている動的に配分されたメモリの少なくとも一部を再生する方法であって、
配分されたある１つのオブジェクトに対して、前記オブジェクトを最初に動的に配分したプログラム又は機能に対して前記オブジェクトがローカルであるかどうかを示す追加的情報を前記オブジェクトと関連付けるステップと、
前記オブジェクトを最初に動的に配分した前記プログラム又は機能が終了した又は前記プログラム又は機能から出た直後であって、前記メイン・ガーベッジ・コレクタがメモリの再生を試みるより前に、前記プログラム又は機能によって配分された少なくとも１つのオブジェクトについて前記追加的情報を走査し、前記少なくとも１つのオブジェクトが前記プログラム又は機能に対してローカルであることを示す追加的情報を前記少なくとも１つのオブジェクトが有するかどうかを判断するステップと、
前記少なくとも１つのオブジェクトが前記プログラム又は機能に対してローカルである場合には、前記少なくとも１つのオブジェクトと関連付けられている動的に配分されたメモリを再生し、前記少なくとも１つのオブジェクトが前記プログラム又は機能に対してローカルでない場合には、前記少なくとも１つのオブジェクトと関連付けられている動的に配分されたメモリを再生するステップを後の時点まで延期するステップと、
を含むことを特徴とする方法。
【請求項１０】
請求項９記載の方法において、
前記プログラム又は機能によって配分されたすべてのオブジェクトについて、走査する位置の範囲を決定するステップと、
前記少なくとも１つのオブジェクトと関連付けられた情報とを求めて前記位置の範囲だけを走査するステップと、
を更に含むことを特徴とする方法。
【請求項１１】
請求項１０記載の方法において、前記追加的情報は前記オブジェクトを最初に動的に配分した前記プログラム又は機能と実行の対応するスレッドとを識別し、この方法は、更に、
前記１又は複数のオブジェクトのそれぞれと関連付けられた前記追加的情報に基づいて、前記位置の範囲の中の１つ又は複数のオブジェクトの中のどのオブジェクトが前記実行の対応するスレッドにおいて前記プログラム又は機能によって配分されたのかを判断するステップを含むことを特徴とする方法。
【請求項１２】
請求項９記載の方法において、前記プログラム又は機能が実行される間に前記少なくとも１つのオブジェクトが前記プログラム又は機能の外部で参照される場合には、前記少なくとも１つのオブジェクトはローカルでないことを示すステップを更に含むことを特徴と

する方法。
【請求項１３】
請求項９記載の方法において、前記少なくとも１つのオブジェクトは前記プログラム又は機能に対してローカルであり、前記メイン・ガーベッジ・コレクタの動作よりも前に再生されることを特徴とする方法。
【請求項１４】
請求項９記載の方法において、前記少なくとも１つのオブジェクトは参照する側のオブジェクトにおいて参照され、この方法は、更に、
（ｉ）前記参照する側のオブジェクトと前記少なくとも１つのオブジェクトとが異なるプログラム又は機能によって配分されるか、又は、（ｉｉ）前記参照する側のオブジェクトが前記プログラム又は機能の外部で参照される場合には、前記少なくとも１つのオブジェクトはローカルではないことを示すステップを含むことを特徴とする方法。
【請求項１５】
請求項９記載の方法において、前記プログラム又は機能はジャバ仮想計算機の内部で実行されることを特徴とする方法。
【請求項１６】
請求項９ないし請求項１５のいずれかの請求項に記載の方法に含まれるステップをコンピュータに実行させる命令から構成されるコンピュータ・プログラムが記憶されているコンピュータ可読な記憶媒体。
【発明の詳細な説明】
【０００１】
【技術分野】
この発明は一般にコンピュータ・システムに関し、特に該システムのメモリ部分の管理に関する。
【０００２】
【背景技術】
コンピュータ・システムの多くはオブジェクトを使用して情報を管理する。オブジェクトとは、特定の属性を共用し、等速呼び出しメモリ（ＲＡＭ）の記憶域を占めるデータである。オブジェクトはメモリ内で重複することはできない。ライブ・オブジェクトとはコンピュータ・システムによって目下実行中の計算プロセスに必要なオブジェクトのことである。システム内の全てのオブジェクトが常時ライブであるならば、メモリの管理は関係ない。システムの起動時に各オブジェクトに割振られるスペースは再生する必要がないからである。しかし、多くのシステムでは、ライブ・オブジェクトの寿命は前もって予測し得ない可変的なものである。このようなシステムでは、メモリ資源を保存するべき場合は、満了した、すなわち死んだオブジェクトを認識し、メモリからそれらを排除することが必要である。
【０００３】
不要情報とは、このようなデータが割振られたプログラム、方法、機能、またはサブルーチンの実行にもはや使用されない、コンピュータ・システムのメモリに格納されたデータのことである。便宜上、データを割振るプログラム、方法、機能、またはサブルーチンを間にプログラムまたは機能と呼ぶ。不要情報収集とはもはや使用されない動的に割振られたメモリ内にデータを位置指定し、その五のメモリ割振り要求を満たすためにメモリを再生するプロセスである。不要情報の収集によって、プログラマはプログラムの終了時にもはや必要ない場合にデータをメモリから削除する心配をする必要がないので、プログラマの生産性が著しく高まるという利得が得られる。従って、不要情報の収集によって、プログラマおよびデザイナーはその努力を基本アルゴリズム、ユーザー・インターフェース、および汎用の機能性の設計のような高レベルの仕事に集中できるようになる。更に、低レベルの多くのプログラミングの問題点を除去することによって、不要情報の収集によりプログラミング・エラーの確率が低下する。不要情報収集のこのような利点は相互に複合して、より低い開発コストでソフトウェアの機能性と信頼性が向上する。

【０００４】
不要情報の収集は多くの状況で実行することができる。例えば、利用できるメモリ内に残された記憶容量が或る所定レベル以下に減少した場合は、メモリをできる限り回復するために不要情報の収集が実行される。更に、プログラム、もしくは機能は不要情報コレクタを呼び出すことによって強制的に不要情報を収集することができる。最後に、不要情報コレクタを再生されるべきオブジェクトを探索する背景的なタスクとしてランしてもよい。しかし、従来形の不要情報コレクタは、呼び出すことができたとしても、もはや使用されないメモリ領域の探索で全てのメモリを探査するためにシステムの周期的な停止を実行することによって動作する。従来形の不要情報コレクタには多くの重要な欠点がある。このような欠点の１つは、オブジェクトの割振り、および割振り解除に関して、記憶機構の処理能力は一般に例えばスタックの割振りよりも大幅に低いことである。更に、メモリを割振るために必要な時間は極めてルーズに限定されている。すなわち、割振り時間の限定は、マウスのトラッキング、対話形のマルチメディア装置制御、およびバーチャル・リアリティ・システムのような高度に対話形の、もしくはリアルタイムのシステムを確実にプログラミングできるほど充分には厳密ではない。最後に、ある種の不要情報コネクタでは、メモリの読み出しおよび書き込みに関連する性能上の不利点は、システム全体の性能が許容限度を超えて遅くなる程大きい。

【０００５】
上記のような問題点は生来の限定性と特殊性を備えたシステムでは更に悪化する。例えば、マイクロソフト・ウインドゥズＣＥは、携帯用パソコンから専門的な業務用コントローラおよび顧客用電子機器にいたる広範な埋込み式製品で使用できるコンパクトで、効率がよく、基準化が可能なオペレーティング・システムである。マイクロソフト・ウインドゥズＣＥを使用した多くの機器は、機器のコストを低く、サイズをコンパクトに抑え、また電力消費の効率を高めるために、１メガバイトのような比較的少量の等速呼び出しメモリ（ＲＡＭ）を搭載することを意図している。その上、マイクロソフト・ウインドゥズＣＥを使用するように設計された機器は標準的には、マイクロソフト・ウインドゥズＮＴのようなよりパワフルなオペレーティング・システムをランするために設計されたコンピュータに標準的に見られるよりも出力が小さいプロセッサを有している。このような生来の限界と特殊性を備えたシステムの場合、利用できる記憶容量を最大限に活用することが不可欠である。このようなシステムで利用できる記憶容量を効果的に、かつ効率よく最大限にすることが必要である。

【０００６】
【発明の開示】
本発明は、プログラム又は機能が実行されている間に可能な限り多くの一時（テンポラリ）オブジェクトを除去することによって、主不要情報コレクタ（メイン・ガーベッジ・コレクタ、主たるゴミ（ガーベッジ）収集手段、main garbage collector）がトリガされないようにする方法に関する。

【０００７】
プログラム中には、オブジェクトを配分する（割り振る、割り当てる、allocate）コマンドと、オブジェクトを配分しないコマンドとが存在する。オブジェクトは、ヒープから配分されるのが典型的である。本発明の１つの特徴によると、あるプログラム・コマンドがオブジェクトを配分すると、当該プログラムが終了した後の時点で、識別を容易にするオブジェクトに情報が記憶される。この情報は、例えば、スレッド識別子とスタック番号とマーク・ビットとを含む。

【０００８】
本発明によれば、メイン・ガーベッジ・コレクタを待機することなく、前記のようなスペースを再生することが可能にある。プログラムが実行されている間に、配分されたオブジェクトが別のオブジェクトの中に記憶されないのであれば、当該オブジェクトを破棄して当該オブジェクトが占めているスペースを再生することができる。なるほど、メイン・ガーベッジ・コレクタが起動されれば、当該オブジェクトが占めているスペースは再生さ

れる。しかし、本発明によると、メイン・ガーベッジ・コレクタがそのタスクを実行するのよりも早く、メモリ空間を空けることができ、これが本発明の利点の１つである。そのようなことが可能なのは、前述したように、本発明によれば、あるスペースを配分したプログラムが終了すると直ちに当該スペースを再生するからである。更に、本発明によるインクリメンタルなガーベッジ・コレクタ（incremental garbage collector、増分不要情報コレクタ）の場合には、メイン・ガーベッジ・コレクタが行うようにヒープの全体を走査するのではなく、ヒープの中の配分された部分だけを走査することができるのである。

【０００９】

【発明を実施する最良の形態】

以下の詳細な説明では、説明の一部をなし、この発明を実施できる特定の実施例が示されている添付図面を参照する。従って、以下の詳細な説明は限定的な意味で受け取られるべきではなく、この発明は添付の特許請求の範囲によって定義されるものである。

【００１０】

詳細な説明には３つの項目がある。第１の項目はこの発明の実施例を実施できるハードウェアと動作環境を記載する。第２の項目はこの発明の１実施例のシステム・レベルを説明する。最後に、第３の項目はこの発明を実施する方法を説明する。

ハードウェアおよび動作環境

図１はそれと組合わせてこの発明の実施例を実施できるハードウェアおよび動作環境の図面である。図１は、それと組合わせてこの発明を実施できる適切なコンピュータ・ハードウェアと適切なコンピュータ環境の簡単に基本的に説明するものである。必ずしもそうである必要はないが、この発明をパーソナル・コンピュータのようなコンピュータによって実行されるプログラム・モジュールのような、コンピュータが実行できる命令という基本的文脈で説明する。一般に、プログラム・モジュールには特定のタスクを実行し、または特定の種類の抽象データを実施するルーチン、プログラム、オブジェクト、構成要素、データ構造などが含まれている。

【００１１】

更に、この発明は携帯用機器、マルチプロセッサ・システム、マイクロプロセッサを使用した、またはプログラム可能な顧客用電子機器、ネットワーク・パソコン、ミニコンピュータ、メインフレーム・コンピュータなどを服務その他の構造のコンピュータ・システムでも実施できることが当業者には理解されよう。この発明は更に、通信ネットワークを関してリンクされる遠隔処理装置によってタスクが実行される分散形のコンピュータ環境でも実施できる。分散形のコンピュータ環境では、プログラム・モジュールを局所的、および遠隔地の双方の記憶装置内にロケートすることができる。

【００１２】

この発明を実行するための図１に示したハードウェアおよび動作環境の例には、プロセッサ・ユニット２１、システム・メモリ２２、およびシステム・メモリを含む様々なシステムの構成要素をプロセッサ・ユニット２１に作用的に結合するシステム・バス２３を含むコンピュータ２０の形式の汎用計算機を含んでいる。プロセッサ・ユニット２１は単一でも、１つ以上あってもよく、従ってコンピュータ２０のプロセッサは単一の中央処理装置（ＣＰＵ）または、複数の処理装置を構成し、いずれも共通して並列処理環境と呼ばれる。コンピュータ２０は従来形のコンピュータ、分散形のコンピュータ、またはその他の種類のコンピュータでもよく、このように本発明は限定的なものではない。

【００１３】

システム・バス２３はメモリ・バス、またはメモリ・コントローラ、周辺バス、および多様なバス構造のいずれかを採用した局所的バスを含む幾つかの種類のバス構造のいずれでもよい。システム・メモリも単にメモリと呼び、読み出し専用メモリ（ＲＯＭ）２４と、等速呼び出しメモリ（ＲＡＭ）２５とを含んでいる。例えば起動中に情報をコンピュータ２０内の素子間で伝送することを補助する基本ルーチンを含む基本入力／出力システム（ＢＩＯＳ）２６がＲＯＭ２４内に格納されている。コンピュータ２０は更に、ハードディスク（図示せず）から読み出し、またはこれに書き込むためのハードディスク・ドライブ

２７と、取り出し可能な磁気ディスク２９から読み出し、またはこれに書き込むための磁気ディスク・ドライブ２８と、ＣＤ　ＲＯＭまたはその他の光学媒体のような取り出し可能な光学ディスク３１から読み出し、またはこれに書き込むための光学ディスクドライブ３０とを含んでいる。

【００１４】

ハードディスク・ドライブ２７と、磁気ディスク・ドライブ２８と、光学ディスクドライブ３０とはそれぞれハードディスク・ドライブ・インターフェース３２と、磁気ディスク・ドライブ・インターフェース３３と、光学ディスク・ドライブ・インターフェース３４のそれぞれによってシステム・バス２３に接続されている。駆動機構と関連するコンピュータによる読み出しが可能な媒体によって、コンピュータによる読み出しが可能な命令、データ構造、プログラム・モジュール、およびコンピュータ２０用のその他のデータが持久的に記憶される。磁気カセット、フラッシュメモリ・カード、ディジタル・ビデオディスク、ベルヌーイ・カートリッジ、等速呼び出しメモリ（ＲＡＭ）、読み出し専用メモリ（ＲＯＭ）などのような、コンピュータによってアクセス可能なデータを記憶できる任意の種類のコンピュータ読み出し可能媒体を動作環境で使用してもよいことが当業者には理解されよう。

【００１５】

ハードディスク、磁気ディスク２９、光学ディスク３１、ＲＯＭ２４、またはＲＡＭ２５にはオペレーティング・システム３５、単数または複数の応用プログラム３６、その他のプログラム・モジュール３７、およびプログラム・データ３８を含む多数のプログラム・モジュールを格納できる。ユーザーはキーボード４０およびポインタ４２のような入力装置を介してコマンドおよび情報をパーソナル・コンピュータ２０に入力できる。その他の入力装置（図示せず）にはマイクロフォン、ジョイスティック、ゲームパッド、サテライトディッシュ、スキャナなどが含まれる。これらの、およびその他の入力装置はシステムに結合されたシリアルポート・インターフェース４６を介してプロセッサ・ユニット２１に接続される場合が多いが、パラレルポート、ゲームポート、または汎用シリアル・バス（ＵＳＢ）のようなその他のインターフェースに接続してもよい。ビデオ・アダプタ４８のようなインターフェースを介してモニタ４７またはその他の種類のディスプレー装置もシステム・バス２３に接続される。モニタに加えて、コンピュータは標準的にはスピーカおよびプリンタのようなその他の周辺出力機器（図示せず）を含んでいる。

【００１６】

コンピュータ２０は遠隔コンピュータ４９のような単数または複数の遠隔コンピュータへの論理接続を利用してネットワーク化された環境で動作することができる。これらの論理接続はコンピュータ２０、またはその一部に結合された通信機器によって達成されるが、この発明は特定の種類の通信機器に限定されるものではない。遠隔コンピュータ４９は別のコンピュータ、サーバー、ルーター、ネットワーク・パソコン、クライアント、同類の機器、またはその他の共通のネットワーク・ノードでよく、図１には記憶装置５０だけが図示されているが、コンピュータ２０に関連して前述した素子の多く、または全てを含んでいる。図１に示した論理接続には企業内情報通信ネットワーク（ＬＡＮ）５１および広域ネットワーク（ＷＡＮ）５２が含まれる。このようなネットワーク環境はあらゆる種類のネットワークであるオフィス・ネットワーク、企業間ネットワーク、イントラネット、およびインターネットで一般的である。

【００１７】

ＬＡＮネットワーク環境で使用される場合、コンピュータ２０は通信機器の一種であるネットワーク・インターフェースまたはアダプタ５３を介して局域内ネットワーク５１に接続される。ＷＡＮネットワーク環境で使用される場合は、コンピュータ２０は標準的には一種の通信機器であるモデム５４、またはインターネットのような広域ネットワーク５２を介して通信を達成するためのその他の種類の通信機器を含んでいる。内蔵、または外部装着式でよいモデム５４はシリアルポート・インターフェース４６を介してシステム・バス２３に接続される。ネットワーク環境では、パーソナル・コンピュータ２０、またはそ

の一部に関連して示したプログラム・モジュールは、遠隔記憶装置内に格納してもよい。図示したネットワーク接続は一例であり、コンピュータ間で通信リンクを確立するための他の通信機器を使用してもよいことが理解されよう。

【００１８】

連係してこの発明の実施例を実施できるハートウェアおよび動作環境を上記で説明した。この発明と連係して実施できるコンピュータは従来形のコンピュータ、分散形コンピュータ、または他の任意の種類のコンピュータでよく、この発明はなんら限定されるものではない。このようなコンピュータは標準的にはプロゼンサとして単数または複数の処理装置、およびメモリのようなコンピュータによる読み出し可能媒体か含まれる。コンピュータは更に、ネットワーク・アダプタまたはモデムのような通信機器を含んでいるので、他のコンピュータに通信可能に結合することができる。

システム・レベルの概要

図２（ａ）はＪａｖａ仮想計算機（ＪＶＭ）２０８の環境内でプログラムまたは機能２０４の動作中に一時的オブジェクトを除去する技術のシステム・レベルの概要を示している。ＪＶＭはプログラムまたは機能２０４を読み取り、実行するためにオペレーティング・システム内でランする固有プログラムである。この実施態様でのプログラムまたは機能２０４はＪａｖａコードである。

【００１９】

プログラムまたは機能２０４の実行中、或るコマンドによって、例えば可変記号によってオブジェクトを参照することによって、ヒープ２１２からオブジェクトが割振られるようにできる。ヒープ・マネージャ２１６はそこからオブジェクトを割振りできるヒープ２１２内のアドレスを保存する。より多くのオブジェクトがヒープ２１２から割振られると共に、ヒープ・マネージャ２１６はオペレーティング・システムに追加のメモリ・スペースを要求することができる。オブジェクトがヒープ２１２から割振られると、前記オブジェクトはこれに関連する参照カウントをも有している。基準カウントが範囲外になると、可変記号が参照されたオブジェクトの基準カウントが減分される。基準カウントが０であるオブジェクトは不要情報収集の候補である。

【００２０】

不要情報コレクタ２２０は起動されると、ヒープ２１２を走査して基準カウントが０であるオブジェクトを探索し、オブジェクトが占めるメモリを後の使用のために利用できるようにする。不要情報コレクタ２２０の動作は公知であるが、本質的にＪａｖａは以下の環境で不要情報の収集を実行する。すなわち（１）不要情報の収集が必要である場合－ヒープ２１２内に残された記憶容量がある所定レベル以下に達すると、再生可能なメモリを再生するために不要情報の収集が行われる；（２）不要情報の収集が要求された場合－Ｊａｖａ不要情報コレクタであるシステムｇｃを呼び出すことによってＪａｖａ内で強制的に不要情報を収集することができる；または（３）Ｊａｖａが再生されるべきオブジェクトを探索する背景タスクを実行する場合；である。

【００２１】

図２（ｂ）はこの発明の実施例として増分不要情報コレクタ２７０を示している。増分不要情報コレクタ２７０はＪＶＭの統合部品である。先行技術の場合と同様に、ヒープ・マネージャ２６６はそこからオブジェクトを割振ることができるヒープ２６２のアドレスを保存する。機能２５４が起動、または実行されると、増分不要情報コレクタ２７０はヒープ・マネージャ２６６からのアドレスを保存し、そこからオブジェクトを割振ることができる次のメモリを示す。機能２５４を出ると、増分不要情報コレクタ２７０はヒープ２６２から割振られた最後のオブジェクトのアドレスを保存する。ＪＶＭ２５８内での機能２５４の実行中、オブジェクトがヒープ２６２から割振られると、アドレス情報がオブジェクト内に格納され、それによって増分不要情報コレクタ２７０による事後の前記オブジェクトの識別が促進される。

【００２２】

その上、前述したように、増分不要情報コレクタ２７０は機能２５４によって割振られた

ヒープの領域に関する情報、すなわち基本的にヒープ２６２から割振られたオブジェクトの開始アドレスと終了アドレスを有している。プログラムまたは機能２５４が終了し、またはそこから出ると直ちに、増分不要情報コレクタ２７０はヒープから割振られたオブジェクトの開始アドレスから始まって終了アートレスまでヒープを走査し、オブジェクト内に格納されている追加情報を読み出す。増分不要情報コレクタ２７０がオブジェクトを不要情報であると識別すると、コレクタは直ちに前記オブジェクトが占めるスペースを再生して、利用できるようにする。従って、増分不要情報コレクタ２７０は不要情報コレクタ２７４が動作するまで待機せずにオブジェクトが占めるスペースを再生する。加えて、増分不要情報コレクタ２７０はヒープ全体を走査する必要はなく、ヒープ全体ではなくヒープの割振られた領域の開始アドレスから終了アドレスまでだけを走査すればよい。

【００２３】

不要情報コレクタ２２０と２７０を基準カウント技術に関して説明してきたが、その他の不要情報収集の技術が公知である。これらのその他の技術には例えば、据置き基準カウント、マーク掃引収集、マーク・コンパクト収集および複写不要情報収集が含まれる。

発明の実施方法

前項ではこの発明の実施例の動作をシステム・レベルで説明してきた。本項はこのような実施例をコンピュータで実施する方法を説明する。

【００２４】

図３の流れ図のステップ３００では、プログラムまたは機能コマンドまたはコードが実行される。ステップ３０２は、実行されるコードにとってヒープからオブジェクトを割振る必要があるか否かが判定される。ステップ３０２でコードがヒープからオブジェクトを割振らないものと判定された場合は、制御はステップ３０６へと移行する。そうではなく、ステップ３０２でコードがヒープからオブジェクトを割振るものと判定された場合は、制御はステップ３０４に移行する。

【００２５】

ステップ３０４で、オブジェクトに関する情報が割振られたオブジェクトに格納される。オブジェクトに格納された情報については以下により詳細に説明する。次に制御はステップ３０６に進み、プログラムまたは機能が終了したか否かが判定される。プログラムまたは機能が未だ終了していない場合は、次にコマンドがステップ３０８で取出され、制御はステップ３００に移行する。しかし、ステップ３０６でプログラムまたは機能が終了したものと判定されると、ステップ３１０で増分不要情報の収集が実行される。

【００２６】

図４は前述のようにオブジェクトに追加された情報を示している。この情報は（１）オブジェクトを割振る機能またはプログラムのためのスレッドの識別子３１５、（２）機能またはスタック番号３２０、および（３）設定されると、オブジェクトがいずれかの態様で機能の外側に格納されていることを示すマーク・ビット３２５から構成されている。スレッド識別子３１５は公知のように、オペレーティング・システムまたはＪＶＭのいずれかから検索される。機能、またはスタック番号３２０はどの機能がオブジェクトを割振ったかを示す方法である。機能（“機能１”）が別の機能（“機能２”）を呼び出した場合、双方とも同じスレッド識別子を有することになる。機能２がルーチンから出て、増分不要情報コレクタが起動した場合、機能によって割振られたオブジェクトは依然として機能１に利用できることは重要である。

【００２７】

オブジェクトがいずれかの態様で機能の外側に格納されている場合は、前述のようにマーク・ビット３２５が設定される。オブジェクトを割振る機能の外側に格納されていないオブジェクトは機能に対してローカルなオブジェクトと呼ばれる。例えば、マーク・ビット３２５はオブジェクトが大域可変記号内に格納され、戻され、または例外として投出されると、マーク・ビット３２５が設定される。

【００２８】

通常は、マーク・ビット３２５は、オブジェクトが別のオブジェクト（“オブジェクト２

”）に格納されるた場合にもオブジェクト（“オブジェクト１”）用に設定される。しかし、（オブジェクト１がそこに格納される）オブジェクト２がオブジェクト１を割振ったものと同じ機能で割振られ、かつオブジェクト２が同じ機能の外側で参照されない場合は、マーク・ビット３２５はオブジェクト１またはオブジェクト２のいずれにも設定されない。しかし、オブジェクト２のマーク・ビット３２５が後に設定される場合は、増分不要情報コレクタはオブジェクト２内に格納された全てのオブジェクトを吟味する。オブジェクト２に格納されたオブジェクトにマーク・ビットが設定されていない場合、増分不要情報コレクタはそのような時にマーク・ビットを設定する。加えて、以下のコマンドのいずれかが実行される時には常に、使用中のオブジェクト用のマーク・ビット３２５も設定される。すなわち、ＡＡＳＴＯＲＥ，ＡＲＥＴＵＲＮ，ＡＴＨＲＯＷ，ＰＵＴＦＩＥＬＤ，ＰＵＴＦＩＥＬＤ　ＦＡＳＴ、ＰＵＴＳＴＡＴＩＣ、またはＰＵＴＳＴＡＴＩＣ　ＦＡＳＴである。このリストはその他のコマンドを排するものではなく、基本的に、マーク・ビット３２５は機能の外側にオブジェクトを格納できる、または格納する（リストしたような）コマンドが利用するオブジェクト用に設定される。

【００２９】

図５は増分不要情報コレクタがヒープを走査し、オブジェクト内に格納された追加情報を読み出す際に増分不要情報コレクタがヒープを走査し、オブジェクト内に格納された追加情報を読み出す際に辿るステップを示している。増分不要情報コレクタはステップ４００でヒープからオブジェクトを取り出し、次にステップ４０２で、オブジェクト内に格納されたスレッド識別子が増分不要情報コレクタを呼び出した機能またはプログラムのスレッド識別子に対応するか否かを判定する。オブジェクトのスレッド識別子が対応しない場合は、制御はステップ４１０に移行する。スレッド識別子が対応する場合は、制御はステップ４０４に移行し、そこでスタック番号が機能の呼び出しによって割当てられた番号に対応するか否かが判定される。スタック番号が対応しない場合は、制御はステップ４１０に移行する。スタック番号が対応する場合は、制御はステップ４０６に移行し、そこでオブジェクト内に格納されたマーク・ビットが設定されているか否かが判定される。マーク・ビットが設定されている場合は、制御はステップ４０８に移行し、そこでオブジェクトが同じスレッド識別子とスタック番号を有する別のオブジェクト（すなわちオブジェクト２）に格納されているか否かが判定される。否である場合は、制御はステップ４１６に移行し、そこでオブジェクト・ヒープの終端に達しているか否かが判定される。オブジェクトが同じスレッド識別子とスタック番号を有する別のオブジェクト内に格納されている場合は、制御はステップ４１０に移行し、そこでオブジェクト２用にマーク・ビットが設定されているか否かが判定される。設定されている場合は、制御はステップ４１６に移行する。オブジェクト２用にマーク・ビットが設定されていない場合は、制御はステップ４１２に移行し、そこでオブジェクトが他のどこかで参照されているか否かが判定される。参照されている場合は、制御はステップ４１６に移行する。オブジェクトが他のどこかで参照されていない場合は、オブジェクトが占めるスペースはステップ４１４で再生される。

ステップ４０６でマーク・ビットが設定されていない場合は、オブジェクトが占めるスペースはステップ４１４で再生され、制御はステップ４１６に移行する。ステップ４１６はオブジェクト・ヒープの終端に達しているか否かが判定される。オブジェクト・ヒープの終端に達していない場合は、増分不要情報コレクタはステップ４１８に示すように次のオブジェクトを照準し、制御はステップ４００に移行する。これに対してオブジェクト・ヒープの終端に段している場合は、増分不要情報コレクタは終了する。

【図面の簡単な説明】

【図１】はそれと連係して本発明を実施できるハードウェアおよび動作環境のブロック図である。

【図２（ａ）】は一時的オブジェクトを除去する技術のシステム・レベルの概略を示したブロック図である。

【図２（ｂ）】は本発明の実施例として増分不要情報収集モジュールを示したブロック図である。

【図３】はヒープからオブジェクトを割振るプロセスの流れ図である。
【図４】はオブジェクト・ヒープ内の割当てられたオブジェクトに格納される追加情報を示している。
【図５】は増分不要情報コレクタ・モジュールがヒープを走査し、オブジェクト内に格納された追加情報を読み出す際に増分不要情報コレクタ・モジュールが辿るステップを示している。

【図１】

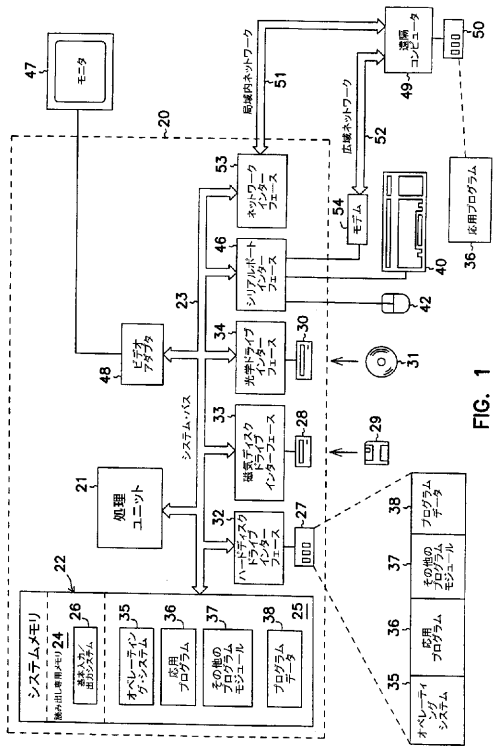

FIG. 1

【図２（ａ）】

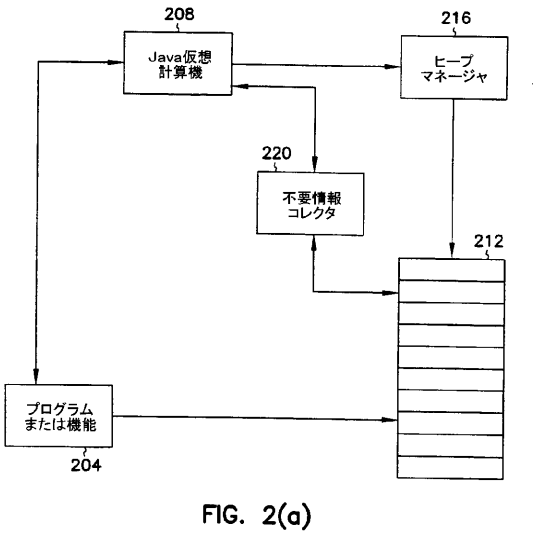

FIG. 2(a)

【図２（ｂ）】

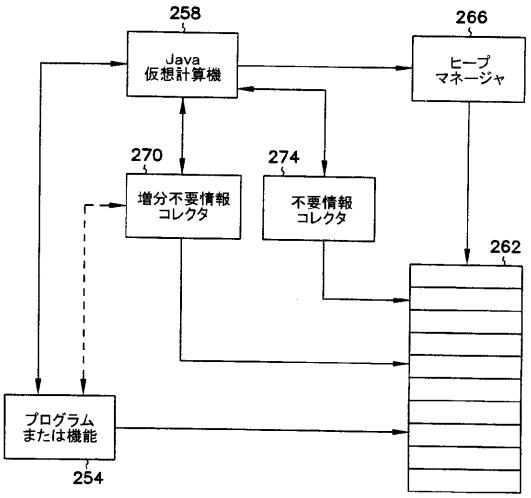

FIG. 2(b)

【図３】

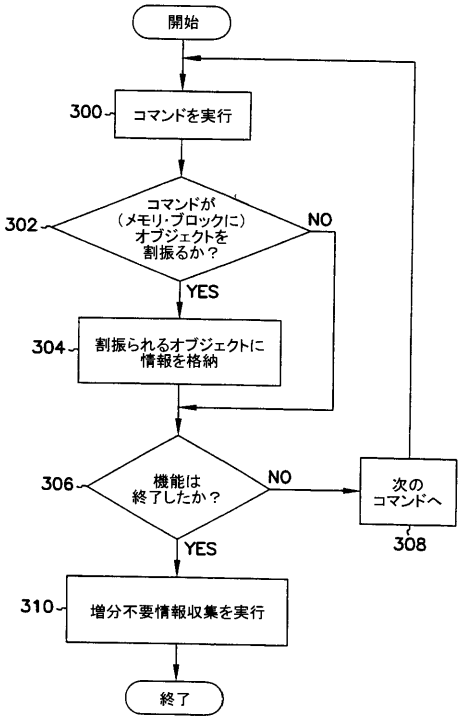

FIG. 3

【図４】

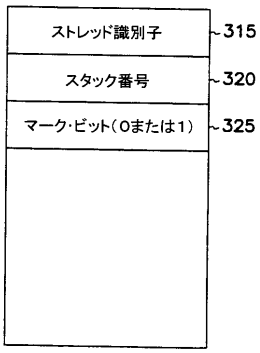

FIG. 4

【図５】

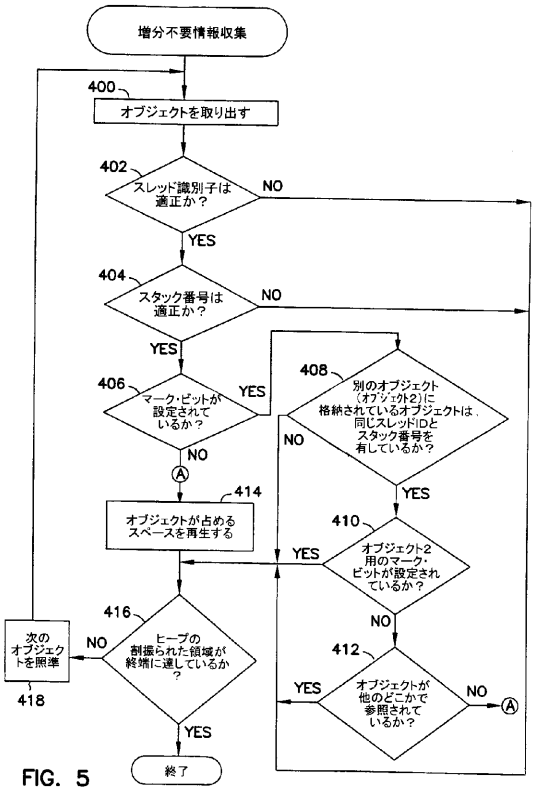

FIG. 5

フロントページの続き

(74)代理人 100096068
弁理士 大塚 住江
(72)発明者 ソーントリー ディヴィッド エム
アメリカ合衆国 ワシントン州 ９８０５３ レッドモンド トゥーハンドレッドアンドサーティナインス プレイス ノースイースト ５４３２
(72)発明者 マークリー マイケル イー
アメリカ合衆国 ワシントン州 ９８０５３ レッドモンド トゥーハンドレッドアンドセヴンティサード アベニュー ノースイースト ３０１４
(72)発明者 ギルバート マーク
カナダ オンタリオ エヌ２エル ３ダブリュー１ ウォータールー バタヴィア プレイス ３２７

審査官 田中 秀人

(56)参考文献 特開平０８－１２３７００（ＪＰ，Ａ）
特開平０７－１０５１０８（ＪＰ，Ａ）
米国特許第０５５３５３９０（ＵＳ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 12/00 - 12/06
G06F 9/06