(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4965210号
(P4965210)
(45)発行日　平成24年7月4日(2012.7.4)　(24)登録日　平成24年4月6日(2012.4.6)

(51)Int.Cl.　F I
GO6F 12/00 (2006.01)　GO6F 12/00 531J

請求項の数 1 (全 16 頁)

(21)出願番号　特願2006-261837(P2006-261837)
(22)出願日　平成18年9月27日(2006.9.27)
(65)公開番号　特開2007-95064(P2007-95064A)
(43)公開日　平成19年4月12日(2007.4.12)
審査請求日　平成21年5月20日(2009.5.20)
(31)優先権主張番号　11/236450
(32)優先日　平成17年9月27日(2005.9.27)
(33)優先権主張国　米国(US)

(73)特許権者　390009531
インターナショナル・ビジネス・マシーンズ・コーポレーション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国１０５０４　ニューヨーク州　アーモンク　ニュー　オーチャード　ロード
(74)代理人　100108501
弁理士　上野　剛史
(74)代理人　100112690
弁理士　太佐　種一
(74)代理人　100091568
弁理士　市位　嘉宏

最終頁に続く

(54)【発明の名称】ファイル・システムの稠密診断データを取得および送信するためにコンピュータに実施させる方法

(57)【特許請求の範囲】
【請求項１】
ファイル・システムの稠密診断データを取得および送信するためにコンピュータに実施させる方法であって、
（ａ）診断ファイル・システムを生成するステップと、
（ｂ）前記診断ファイル・システムをマウントするステップと、
（ｃ）前記ファイル・システムのファイル・システム・メタデータを前記診断ファイル・システムに書き込むステップとを有し、
前記ステップ（ｃ）が、
（ｃ１）前記ファイル・システムにおいて複数のデータ単位を識別するステップと、
（ｃ２）前記複数のデータ単位のうち一のデータ単位をトラバースして、当該一のデータ単位が前記ファイル・システム・メタデータを含むかどうかを判定するステップと、
（ｃ３）前記一のデータ単位が前記ファイル・システム・メタデータを含むとの判定に応答して、前記ファイル・システム・メタデータを前記一のデータ単位から前記診断ファイル・システムに書き込むステップと、
（ｃ４）前記一のデータ単位がユーザ・データを含むかどうかを判定するステップと、
（ｃ５）前記一のデータ単位がユーザ・データを含むとの判定に応答して、スパース・ファイルを前記診断ファイル・システムに書き込むステップと、
（ｃ６）前記複数のデータ単位のうち他のデータ単位がトラバースされていないとの判定に応答して、当該他のデータ単位の各々ごとに、前記ステップ（ｃ２）ないしステップ（

ｃ５）を繰り返すステップとを有し、
（ｄ）前記ステップ（ｃ）が完了した段階で、前記診断ファイル・システムを圧縮して稠密診断データを形成するステップと、
（ｅ）前記稠密診断データをオフサイトのシステムに送信するステップとをさらに有する、方法。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明は一般に、ファイル・システムの整合性を維持および改善することに関する。より詳細には、本発明は、ファイル・システム・メタデータを稠密フォーマットで取得して、そのようなデータが専門家の検査および修復のためにオフサイトに送られ得るようにするための方法に関する。
【背景技術】
【０００２】
コンピュータ分野では、ファイル・システムは、ファイルを見つけアクセスし易くするように、コンピュータ・ファイルおよびそれらが含むデータを保存し組織化するための構造である。ファイル・システムは、ハード・ディスクまたはＣＤ－ＲＯＭなどのストレージ装置を使用することができ、ファイルの物理的ロケーションを維持することを必要とする。代替として、ファイル・システムは、仮想ファイル・システムとすることができる。仮想ファイル・システムは、ネットワークを介した仮想データまたはデータのためのアクセス方式としてのみ存在する。
【０００３】
ファイル・システムは、２つのタイプのデータから作成される。通常大部分を占める第１のタイプのデータは、ユーザ・データである。ユーザ・データの内容タイプは、例えば、テキスト、グラフィックス、音楽、およびコンピュータ命令とすることができる。第２のタイプのデータは、ファイル・システム・メタデータである。ファイル・システム・メタデータは、ユーザ・データ以外のすべてのデータである。メタデータは、ユーザ・データを含むファイルの統計および構造を処理システムに通知する。
【０００４】
ファイル・システム・メタデータは、スーパーブロックを含み、スーパブロックは、ファイル・システム概要を提供し、その他の情報へのポインタを含む。ｉノードは、各ファイルに関連付けられたファイル・システム・メタデータである。ｉノードは、バイト数によるファイル長、関連する装置識別子、ユーザ識別子、グループ識別子、一意とし得るｉノード番号、ファイル・モード、タイムスタンプ、および参照カウントを示す。
【０００５】
ブロックは、ファイルに割り当てられ得る最小単位のディスク・ストレージである。例えば、プロセッサは、特定のファイル・システムにおいて、１ブロックを１０２４バイトに設定することができる。これは、ファイルがほぼ常に１つ以上のブロックを満たし、最終ブロックはデータによって部分的にのみ占有されることを意味する。
【０００６】
ファイルの部分は、複数のブロックに存在することができ、時にはディスク・ドライブ内に散在させられる。ｉノードは、ｉノード内で複数のブロックをリストすることができる。より大きなファイルの場合、ｉノードは、間接ブロックを含むことができ、間接ブロックは、さらなるブロックのリストをポイントすることができる。しばしば、これは、後続のより深いブロックの層をポイントする多数のレベルの間接ブロックからなるツリー状構造を生じさせる。
【０００７】
ファイル・システムは、非常に大きなデータ構造になる傾向にある。プロセッサがファイル・システムに変更を加える場合、プロセッサはしばしば、多くの別個の書き込み操作を必要とする。ときには、エラーまたはその他の障害、例えば、電源障害の発生などは、

一連の書き込みを途中で妨げることがある。

【０００８】

こうした状況でプロセッサがエラーに遭遇すると、競合条件が発生することがある。競合条件は、１つの電子装置における２つのイベントが、どちらが装置の状態または出力に影響を与えるかを識別するために互いに本質的に競合するときに発生するものであり、その場合、最初に到着したイベントまたは信号が、装置の状態を制御する。ジャーナル・ファイル・システムなどのファイル・システムの文脈では、ファイル・システムを更新する場合、１）ファイルのディレクトリ・エントリを削除し、２）そのファイルのｉノードをフリー・スペース・マップ内でフリー・スペースとしてマークするという２つのステップが発生する。

【０００９】

電源が障害を起こし、ステップ１がクラッシュの直前に発生した場合、孤立したｉノードが存在するようになり、実際に割り当てられているよりも多くのブロックが、ストレージに割り当てられているように見える。ステップ２がクラッシュの直前に発生した場合、まだ削除されていないｉノードがフリーとしてマークされ、おそらく何か他のものによって上書きされることになる。

【００１０】

特定のタイプのファイル・システムであるジャーナル・ファイル・システムは、さらなる障害モードを有する。前述の２つのステップに加えて、ジャーナル・ファイル・システムは、トランザクションのために行われた変更をコミットする第３のステップを有する。ものごとが正常に機能した場合、プロセッサは、トランザクションのすべてをジャーナル・ログにコミットするか、さもなければ、トランザクションの何もジャーナル・ログにコミットしない。ジャーナル・ファイル・システムでは、プロセッサは、メタデータを整合性のある状態に確立するため、ジャーナル・ログを再生することができる。しかし、ジャーナル・ファイル・システムは、ある書き込みが失敗したのに、プロセッサがトランザクションのその他の部分をジャーナル・ログに書き込んだときに発生するようなＩ／Ｏエラー処理の失敗が起こると、不整合になることがある。

【００１１】

ファイル・システムの保守および回復を遠隔地にアウトソーシングすることに伴う危険は、送られるファイル・システムの詳細が傍受され得ることである。これは、データがインターネットを介して送信される場合に特に当てはまる。暗号化データであっても、傍受されれば、危険にさらされ得る。

【発明の開示】

【発明が解決しようとする課題】

【００１２】

したがって、可能であればインターネットを介した機密情報の送信を回避することが長年にわたって望まれている。

【課題を解決するための手段】

【００１３】

本発明の側面は、ファイル・システムの稠密診断データを取得および送信するためにコンピュータに実施させる方法を提供する。プロセッサは、ファイル・システム内の各データ単位を識別する。プロセッサは、データ単位がファイル・システム・メタデータを含むかどうかを判定する。データ単位がファイル・システム・メタデータを含むと判定された場合、プロセッサは、メタデータのデータ単位を診断ファイルに書き込む。次にプロセッサは、データ単位がユーザ・データを含むかどうかを判定する。データ単位がユーザ・データを含む場合、プロセッサは、スパース・オブジェクト（sparse object）を診断ファイルに書き込む。

【００１４】

本発明の特色と考えられる新規な特徴が、添付の特許請求の範囲において説明される。しかし、本発明自体は、本発明の好ましい使用方法、さらなる目的、および利点と同様、

本発明の例示的な実施形態についての以下の詳細な説明を添付の図面と併せて読みながら参照することによって最も良く理解されよう。

【発明を実施するための最良の形態】

【００１５】

ここで図面を、特に図１を参照すると、本発明の実施形態を実施し得るデータ処理システムのブロック図が示されている。示された例では、データ処理システム１００は、ノース・ブリッジおよびメモリ・コントローラ・ハブ（ＭＣＨ）１０８と、サウス・ブリッジおよび入力／出力（Ｉ／Ｏ）コントローラ・ハブ（ＩＣＨ）１１０とを含むハブ構成を利用する。処理ユニット１０２、メイン・メモリ１０４、およびグラフィックス・プロセッサ１１８は、ノース・ブリッジおよびメモリ・コントローラ・ハブ１０８に接続する。グラフィックス・プロセッサ１１８は、アクセラレーテッド・グラフィックス・ポート（ＡＧＰ）またはグラフィックス・プロセッサ１１８を介して、ノース・ブリッジおよびメモリ・コントローラ・ハブ１０８に接続してもよい。

【００１６】

示された例では、ローカル・エリア・ネットワーク（ＬＡＮ）アダプタ１１２、オーディオ・アダプタ１１６、キーボードおよびマウス・アダプタ１２０、モデム１２２、読み取り専用メモリ（ＲＯＭ）１２４、ユニバーサル・シリアル・バス（ＵＳＢ）ポートおよびその他の通信ポート１３２、ならびにＰＣＩ／ＰＣＩｅデバイス１３４が、バス１３８を介して、サウス・ブリッジおよびＩ／Ｏコントローラ・ハブ１１０に接続する。ＰＣＩ／ＰＣＩｅデバイスは、例えば、イーサネット（登録商標）・アダプタ、アドイン・カード、およびノートブック・コンピュータ用のＰＣカードを含むことができる。ＰＣＩはカード・バス・コントローラを使用するが、ＰＣＩｅは使用しない。ＲＯＭ１２４は、例えば、フラッシュ基本入力／出力システム（ＢＩＯＳ：basic input/output system）とすることができる。

【００１７】

ハード・ディスク・ドライブ１２６、およびＣＤ－ＲＯＭドライブ１３０は、バス１４０を介して、サウス・ブリッジおよびＩ／Ｏコントローラ・ハブ１１０に接続する。ハード・ディスク・ドライブ１２６、およびＣＤ－ＲＯＭドライブ１３０は、例えば、統合ドライブ電子（ＩＤＥ）またはシリアル・アドバンスト・テクノロジ・アタッチメント（ＳＡＴＡ）インタフェースを使用することができる。スーパーＩ／Ｏ（ＳＩＯ）デバイス１３６は、サウス・ブリッジおよびＩ／Ｏコントローラ・ハブ１１０に接続されることができる。

【００１８】

オペレーティング・システムは、処理ユニット１０２上で動作し、図１のデータ処理システム１００内の様々なコンポーネントを調整および制御する。オペレーティング・システムは、ＩＢＭ（ＩＢＭはＩＢＭ Ｃｏｒｐｏｒａｔｉｏｎの商標）コーポレーションから入手可能なアドバンスト・インタラクティブ・エグゼクティブ（ＡＩＸ（Ｒ））などの市販のオペレーティング・システムとすることができる。ＡＩＸは、ＩＢＭ Ｃｏｒｐｏｒａｔｉｏｎの登録商標である。

【００１９】

本発明の様々な実施形態は、重荷となるユーザ・データを送信する面倒を伴わずに、ファイル・システム不整合の詳細だけを送信することを可能にする。ユーザ・データは、ファイル・システム・データの大部分をなす傾向にあり、したがって、場所確保形式（place-keeping form）で以外は送信されない。ファイル・システム整合性チェック・プログラムは、ファイル・システム内の複数のデータ単位を識別する。識別するための１つの方法は、ファイル・システム整合性チェック・プログラムが、１つのデータ単位から別のデータ単位に移動しながら、ファイル・システムのツリー状構造を識別することである。いくつかの実施形態は、ブロックと同じ大きさのデータ単位を使用することができる。別の実施形態は、例えば、セクタなど、より大きなデータ単位を使用することができる。データ単位がファイル・システム・メタデータまたはメタデータを含む場合、ソフトウェアは、

メタデータの単位を、診断ファイル・システムの一部とすることができる診断ファイルに書き込みまたはコピーする。データ単位がユーザ・データを含む場合、ソフトウェアは、スパース・オブジェクトを診断ファイルに書き込みまたはコピーする。その後、診断ファイルをバックアップまたはその他の方法で統合するステップは、スパース・オブジェクトを、元のユーザ・ファイルのコンパクトではあるが完全に空で何も書き込まれていないバージョンとして扱う。スパース・オブジェクトは、ファイルのメタデータ内のヌル・ポインタとすることができる。このポインタは指示対象をポイントしないので、保存された基となるデータがなく、単に空またはすべてが「０」のデータ単位を表すにすぎない。したがって、そのようなスパース・オブジェクトは、ある意味で、ファイル・システムを構成するディスクまたはその他の媒体上のデータ単位内の「０」領域の代わりに用いられる場所確保データとして動作する。したがって、処理システムは、ソース・ファイル・システムと比べて非常に縮小された形式で各スパース・オブジェクトを送信することができる。

【００２０】

図２は、本発明の一実施形態に従って、顧客システム２０１がどのように専門家システム２０５と対話することができるかを示している。顧客システム２０１は、例えば、図１の処理システム１００を使用して動作することができる。管理者は、ファイル・システムが異常な挙動をしていると判定することができる。それに応じて、管理者は、稠密診断データを含む稠密診断ファイルを生成するために、顧客システム２０１を操作することができる。

【００２１】

専門家システム２０５は、顧客システム２０１から診断ファイル・システムを受信することができる。専門家システム２０５は、診断ファイル・システムにサービスを提供して、サービス済ファイル・システムを形成する。専門家システム２０５は、例えば、図１の処理システム１００を使用して動作することができる。サービス提供は、例えば、診断ファイル・システム上で保守を実行して、サービス済ファイル・システムを形成することを含むことができる。加えて、サービス提供は、診断ファイル・システム上で回復動作を実行することを含むことができる。請求システム２０７は、診断ファイル・システムにサービス提供した際に提供されたサービスに対する請求を顧客に行うことができるように、専門家システム２０５上の動作を監視することができる。専門家システム２０５は、顧客システム２０１にサービス済ファイル・システムを返送する。

【００２２】

稠密ファイル・フォーマットは、スパース・ファイルが圧縮されたときに生じる結果である。管理者は、とりわけ、以下で説明されるようにユーザ・データを縮小するファイル整合性チェックを実行するため、コマンド・ライン・エディタでコマンドを入力することができる。プロセッサは、ユーザ・データに格納されている内容のタイプさえ隠蔽する程度までユーザ・データを縮小することができるので、結果的な１つ以上のファイルは稠密であり、ファイル・システム・エラーのソースにより直接的に関係するメタデータを有する。したがって、稠密診断データという用語は、本発明の様々な例示的な実施形態から生じる１つ以上の圧縮されたファイルを示すのに利用される。言い換えると、稠密診断データは、ユーザ・データを除去するかまたはユーザ・データの代わりに場所確保データを使用し、次にその結果の１つ以上のファイルを圧縮した後に生成される。

【００２３】

顧客システム２０１は、例えば、インターネット２０３などのネットワークを介して、稠密診断データを送信する。稠密診断データは、専門家システム２０５に到着する。専門家システム２０５は、データ回復専門家の制御下にあることができる。データ回復専門家は、管理者との確立された信用関係を有していないこともある。加えて、インターネット２０３は、安全でないデータ送信手段として知られている。

【００２４】

図３は、本発明の一実施形態による、ソース・ファイル・システムから診断ファイルへの変換を仲介するオペレーティング・システムの動作を示している。オペレーティング・

システム３０１は、例えば、図１の処理ユニット１０２などのプロセッサ上で動作することができる。ソース・ファイル・システム３０３は、例えば、図１のハード・ディスク・ドライブ１２６などに保存されることができる。管理者は、コマンド・ライン・エディタ３１１を使用して、コマンドを個々にオペレーティング・システム３０１に入力することができる。１つ以上のコマンドが、診断ファイル３０５を生成することができる。加えて、ユーティリティ・プログラムが、コマンド・ライン・エディタ３１１の代わりに使用されることができ、その場合、ユーティリティ・プログラムは、オペレーティング・システムにコマンドを書き込む。

【００２５】

図４は、従来のファイル・フォーマットの一例を示している。シーク・コマンドを使用するとき、ファイル・ポインタが、例えば、「０」ブロック４１３などの物理ディスク・アドレス上を通過するとしても、ファイル４１１に割り当てられた各ビットは、「０」に設定される。このフォーマットでは、プロセッサは、ファイルのすべての割り当てビットおよびブロックを物理的に書き込み、物理レベルにおいて圧縮は利用されない。

【００２６】

図５は、診断ファイルに関連し得る、スパース・オブジェクトと呼ばれることもある、スパース・ファイル・フォーマットの一例を示している。このフォーマットは、各ビットの論理書き込みを実行するプロセッサを伴うが、実際には、プロセッサは、ファイル・ポインタが指示するデータだけを物理的に保存する。ファイル・ポインタは、次の物理書き込みが生じることになるロケーションである。プロセッサがファイル書き込みの間にギャップが存在することを許可する場合、例えば、ブロック・ポインタ・テーブル４１４などのメタデータ内に注記が作成される。プロセッサは、「０」からなるブロックが論理的に記録されるヌル・ポインタ４１５、４１６、４１７、および４１８を設定することによって、ブロック・ポインタ・テーブル４１４に注記を作成する。言い換えると、各ヌル・ポインタは、さもなければハード・ドライブに物理的に書き込まれていた「０」からなるブロックのための一種の場所確保データとして動作する。「０」からなる大きな領域は後の読み取りにおいて回復され得るので、このフォーマットはスパース（sparse）である。さらに、ファイル・メタデータは、「０」からなる領域をユーザ・データに保存するのではなく、「０」ブロックへの参照が保持される場所である。「０」からなる領域または「０」ブロックは、スパース・オブジェクトによって表され、いくつかのファイル・システムは、例えば、プログラムがオペレーティング・システムに読み取りを要求したときなど、オペレーティング・システムの通常の動作中に、「０」からなる大きな領域をスパース・オブジェクトから読み取ることができる。

【００２７】

稠密ファイル・フォーマットは、スパース・ファイルが圧縮されたときに生じる結果である。プロセッサは、各「０」ブロックを著しく圧縮することができる。いくぶん無秩序な「１」と「０」のシーケンスから構成されるデータも同様に圧縮されるが、おそらくそれほどは圧縮されない。したがって、「０」ブロックは、ファイルのすべてが圧縮形式で物理的に書き込まれるように、いくぶん無秩序なデータと一緒に圧縮される。診断データとして始まった１つ以上のファイルは、圧縮が完了すると、稠密診断データになることができる。

【００２８】

図６は、本発明の一実施形態による、ファイル・システムに関する診断データを取得する最初のステップを示している。管理者は、コマンド・ラインを入力して、各ステップを実行するようオペレーティング・システムに命令することによって、これらのステップを実行するようプロセッサに命令することができる。自動的に次々と実行されるコマンドのスクリプトにすべてのステップを統合することも、等しく適している。オペレーティング・システムは、例えば、図３のオペレーティング・システム３０１とすることができる。プロセッサは、診断ファイル・システムを生成するためのコマンドを受け取る（ステップ５０１）。診断ファイル・システムは、図３の診断ファイル３０５に関連することができ

る。ＵＮＩＸ（Ｔｈｅ Ｏｐｅｎ Ｇｒｏｕｐの商標）ライクなシステムでは、新規に生成されたファイル・システムをオペレーティング・システムでアクセス可能にするには、コマンドが必要なことがある。この例示的な実施形態では、そのコマンドは「mount」である。プロセッサは、診断ファイル・システムをマウントするためのmountコマンドを受け取る（ステップ５０３）。これで、診断ファイル・システムは、データを収集するための準備が整う。プロセッサは、診断ファイルにメタデータを抽出するためのコマンドを受け取る（ステップ５０５）。そのコマンドは、例えば、ファイル・システム整合性チェックfsckとすることができる。診断ファイルは、スパース・オブジェクトを含むことができる。診断ファイルは、診断ファイル・システムの構成要素であることによって、診断ファイル・システムに関連することができる。

【００２９】

fsckは、多くのデータを生成し、それらは、プロセッサが診断ファイル・システムを生成したときに割り当てられたストレージを埋め尽くす可能性がある。したがって、プロセッサは、抽出されたメタデータが診断ファイル・システムに収まるかどうかを調べるためにテストを行う（ステップ５０７）。収まらないという判定を下した場合、プロセッサは、より大きな診断ファイル・システムを生成するためのコマンドを受け取ることができる（ステップ５０９）。プロセッサは、ステップ５０９の後、診断ファイル・システムのマウントおよびその他のステップを実行することができる。

【００３０】

プロセッサが抽出されたメタデータは診断ファイル・システムに収まるという判定を下した場合、処理は結合子「Ａ」から図７へと続いて行く。

【００３１】

ファイル・システムを生成し、ファイル・システムをマウントする代替方法が存在する。例えば、診断データを記録する目的で、専用ファイル・システムが存在することができ、稠密診断データを専用ファイル・システムに書き込む動作の前に削除されることができる。

【００３２】

データ処理システムが回復を行う１つの方法は、プロセッサが次にファイル・システムをマウントするとき、ファイル・システムのデータ構造全体にわたる完全なウォーク（walk）またはトラバース（traverse）を実行することである。トラバースは、その他のステップと同様、どのような不整合でも検出し、訂正することができる。この修復を実行するのに使用される１つのツールが、ファイル整合性チェックまたはfsckコマンドである。その他のファイル整合性チェック・プログラムは、「チェック・ディスク」プログラムとしても知られるマイクロソフト社の「chkdsk」を含む。コンピュータ管理者は、ファイル・システムを有するオペレーティング・システムを利用するコンピュータでfsckを使用する。そのようなオペレーティング・システムは、例えば、操作がＵＮＩＸに類似したＡＩＸおよびＬｉｎｕｘ（ＬｉｎｕｘはＬｉｎｕｓ Ｔｏｒｖａｌｄｓの商標）を含む。Ｌｉｎｕｘは、リーナス・トーバルズの登録商標である。ＵＮＩＸは、オープン・グループの登録商標である。その他のオペレーティング・システムのクラスは、Ｍｉｃｒｏｓｏｆｔ（ＭｉｃｒｏｓｏｆｔはＭｉｃｒｏｓｏｆｔ Ｃｏｒｐｏｒａｔｉｏｎの商標）コーポレーションによる様々な世代のＷｉｎｄｏｗｓ（ＷｉｎｄｏｗｓはＭｉｃｒｏｓｏｆｔ Ｃｏｒｐｏｒａｔｉｏｎの商標）オペレーティング・システム、およびＡｐｐｌｅ（ＴＭ）コーポレーションによるＭａｃＯＳを含む。ファイル・システムの例は、ＩＢＭの拡張ジャーナル・ファイル・システムまたはＪＦＳ２などのジャーナル・ファイル・システムを含む。

【００３３】

Ｕｎｉｘライクのオペレーティング・システムのファイル・システムは、いくつかの点で冗長データ用にストレージが割り当てられているかのように振舞う、冗長データを保存するための便利な方法を提供する。その方法は、サイズが少なくとも１ブロックのスパース・オブジェクトを生成することを含む。処理システムが最初にファイルを開くとき、プ

ロセッサは、ファイルの先頭をポイントするファイル・ポインタを生成することができる。ファイル・ポインタは、プロセッサが次にデータを書き込むべき場所を示す場所確保データとして動作し、その場合、プロセッサは、ファイルを線形であるとして扱う。言い換えると、プロセッサは、次に書き込みコマンドを実行するとき、ファイル・ポインタがある場所に書き込みを行う。書き込みコマンドを伴わずにプロセッサがファイル・ポインタを進める場合、ファイル・システムによってファイル用に予約された領域ができるが、ファイルへのブロックの対応する割り当ては行われない。代わりに、ファイル・システム・メタデータは、拡張空ヘッドの存在を数バイトでファイルに保存するが、プロセッサは、どのデータ単位も使用されているとしてマークしない。データ単位は、例えば、ブロックとすることができる。多くのブロックがこの方法で使用されているとして注記されれば、数バイトが多くのブロックを表し得るので、実際に使用されるディスク・スペースが大幅に節約される。ファイル・ポインタを前方および反対方向に動かす一般的な方法は、Ｕｎｉｘライクなシステムで普通に利用可能なlseekシステム・コールを使用する。その他のシステムでは、その他のファイル・ポインタ移動コマンドが存在する。

【００３４】

ファイル整合性チェックが探すものの１つは、データ構造における不整合である。ファイル整合性チェック・ソフトウェアは、ファイル・システム用のデータ構造を探査することによって不整合を探す。ファイル・システム用のデータ構造は、ルートとルートから延びる１つ以上のブランチとを有するツリーから構成される。例えば、ディレクトリ構造は、ルートから始まり、ルートの下に１つ以上のディレクトリを有することができる。同様に、各ファイルは、ブロックに保存されたその構成部分を有し、ブロックも、多数のブランチおよび層を有することができ、その場合、ブロックは、各分岐点におけるノードである。さらなるブロックへのポインタまたは参照を含むブロックは、そのようなポインタをメタデータとして保存する。

【００３５】

ファイル整合性チェック・ソフトウェアは、データ単位からデータ単位へトラバースすることによってツリーを探査し、各データ単位は、ブロックとすることができる。トラバースとは、ファイル整合性チェック・ソフトウェアが、ブロックをその下のさらなるブランチの存在について検査し、いくつか存在する場合、ソフトウェアが、今度は各ブランチを検査し、すべてのブランチが検査され尽くすまで、それを続けることを意味する。検査とは、多くのことを意味する。一般に、検査とは、プロセッサがさらなるブランチに沿ってさらなる参照またはポインタを探すことを意味する。

【００３６】

整合性のあるファイル・システムでは、各データ単位は、そのデータ単位への参照を正確に１つだけ有し、これは、プロセッサがデータ単位を１回だけトラバースすることを意味する。やはり、データ単位は、例えば、ブロックとすることができる。しかし、不整合なファイル・システムでは、ブロックが複数回参照されることがある。ソフトウェアは、トラバースすることによってファイルの単位を識別することができる。例えば、ファイル整合性チェック・ソフトウェアは、第１のファイルをトラバースし、あるブロックが第１のファイルに割り当てられていることを注記することができる。ソフトウェアは、第２のファイルをトラバースし、その同じブロックが第２のファイルにも割り当てられていることを注記することができる。これは本当は、ファイル・システムが、事故的に同じブロックを２回割り当てたもので、第１のファイルのデータは第２のファイルのデータによって上書きされる可能性がある。すべてのファイルをトラバースすることによって、ファイル整合性チェックは、多くの問題を発見することができる。

【００３７】

ファイル整合性チェックは、質問／回答形式によるユーザとの対話に応じて、問題を修復することができる。しかし、この機能は、非常に複雑かつ専門的なので、ファイル・システムの管理者は、できるだけ多くのデータを回復するため、専門家からの外部支援を求めることがある。多くの専門家は高い適格性を備えているが、残念ながら、ファイル・シ

ステムの管理者と専門家の間に確立されたレベルの信用がないこともある。言い換えると、ファイル・システム上のユーザ・データが危険にさらされることから保護する機構が存在することが必要である。

【００３８】

ソフトウェア開発者は、fsckおよびその他のファイル整合性チェッカにコードを追加することができる。追加のコードは、各ファイルに割り当てられたブロックをトラバースするときに、追加のステップを実行する。不整合を認識するのに加えて、fsckは、各ファイルのある様相のコピーを書き込むことができる。トラバースおよび書き込みの処理は、図６のステップ５０５で実行される。コピーは、プロセッサが診断ファイル・システムを生成したときに割り当てられたストレージを埋め尽くす可能性がある。

【００３９】

図７は、本発明の一実施形態による、１つ以上のファイル・システムに関する診断データをパッケージ化して送信するステップを示している。図３のオペレーティング・システム３０１などのオペレーティング・システムは、図７のステップを実行することができる。代替として、図７のステップは、ユーティリティ・プログラムによって実施されることができ、その場合、ユーティリティ・プログラムは、コマンドをオペレーティング・システムに書き込む。一般に、図７のステップは、より小さく、ネットワークまたはその他の手段を介し転送するのがより容易なデータ構造を生成する汎用的機能を達成する。最初、オペレーティング・システムは、診断ファイル・システムを単一のファイルにバックアップまたはその他の方法で統合する（ステップ５１１）。その後、オペレーティング・システムは、ファイルを圧縮することができる（ステップ５１３）。オペレーティング・システムが圧縮を行った場合、それは、プロセッサが、繰り返しおよび冗長性を利用してデータを要約し、短縮された要約を保存することを意味する。例えば、４０９６個の「０」からなるブロックを記述するための、４０９６個の「０」をすべて示すのとは別の方法は、単に電子的に４０９６×０を保存することであり、そのほうがはるかに短い。加えて、ヌル・ポインタがすでにメタデータ内に存在していることがあるので、単にヌル・ポインタを保存することが、圧縮として動作する。ヌル・ポインタの大きなシーケンスさえも、圧縮されることができる。例えば、１０００個のヌル・ポインタからなるシーケンスを記述するための、物理的に各ヌル・ポインタを保存するのとは別の方法は、電子的に１０００×ｎｕｌｌを保存することである。その後、オペレーティング・システムは、圧縮されたファイルまたは稠密診断データを送信する（ステップ５１５）。したがって、オペレーティング・システムは、ユーザ・データを稠密診断データに変換する。推定上、多くの小さなファイルを有する１０２４ギガバイトの拡張ジャーナル・ファイル・システム（ＪＦＳ２）ファイル・システムは、約１６０メガバイトの稠密診断データに変換されることができ、これは６０００分の１の圧縮である。

【００４０】

図８は、拡張ファイル整合性チェック・ソフトウェアを使用して診断データを取得する詳細なステップを示している。拡張ファイル整合性チェックは、fsckの変形とすることができる。fsckは、ファイル・システム内の各データ単位をトラバースする（ステップ６２１）。fsckを実行するプロセッサは、データ単位がファイル・システム・メタデータを含むかどうかを判定することができる（ステップ６２３）。「含む」とは、データ単位が、ファイル・システムを構成するすべてのメタデータの少なくとも一部を含むことを意味する。fsckは、データ単位がファイル・システム・メタデータであるとの判定に応答して、メタデータの単位を書き込む（ステップ６２５）。さもなければ、ステップ６２３および６２５は、ステップ６２７に続く。fsckは、データ単位がユーザ・データを含むかどうかを判定する（ステップ６２７）。fsckは、データ単位がユーザ・データであるとの判定に応答して、スパース・オブジェクトを診断ファイルに書き込む（ステップ６２９）。スパース・オブジェクトの書き込みは、プロセッサがメタデータ内のヌル・ポインタを書き込むことを含むことができる。ヌル・ポインタは、例えば、図５のブロック・ポインタ・テーブル４１４のヌル４１５とすることができる。fsckは、トラバースすべきさらなるデー

タ単位が残っているかどうかを判定する（ステップ６３１）。ステップ６２５および６２９の各書き込みは、図３の診断ファイル３０５へのものとすることができ、その場合、診断ファイルは、診断ファイル・システムに関連することができる。スパース・オブジェクトが非スパース・オブジェクトと異なるのは、ブロックが空であるかまたは有効でないデータで満たされていることを注記することを除き、ディスクへの書き込みがデータを変化させない点である。ブロック全体を占有する代わりに、fsckは、メタデータに関連するすべてのスパース・オブジェクトのリストを単に作成するにすぎない。代替として、fsckは、lseekシステム・コールを単に使用して、書き込みのためにオープンされたファイルのファイル位置を進める。lseekシステム・コールは、Ｕｎｉｘライクのオペレーティング・システムで利用可能なファイル処理システム・コールである。基本的に、lseekコマンドは、現在のファイル末尾を越えてファイル位置を動かすように、オペレーティング・システムに命令する。lseekシステム・コールがブロック全体にわたってファイル位置を進めた場合、オペレーティング・システムは、ディスク・ドライブ上のストレージの物理ブロックを割り当てることなく、ブロックがスパース・ブロックまたはスパース・オブジェクトであることを記録する。スパース・オブジェクトは、ストレージの仮想ブロックであるが、物理的ストレージの完全なブロックのスペースを占有しない。むしろ、メタデータは、とりわけ、ファイルに関連するそのようなブロックの番号を識別する。それにも関わらず、プロセッサは、ファイルによって占有される全スペースがディレクトリ・リスティング中に決定されるとき、そのようなスパース・オブジェクトを計算に追加する。

【００４１】

図９は、本発明の一実施形態による、ＡＩＸオペレーティング・システムのコマンド・ラインで入力されるコマンドの一例を示している。山括弧<>で括られたテキストは、ユーザ指定の文字列を示す。例えば、<新規ファイル・システム・サイズ> ７１１は、新規ファイル・システムに割り当てられる５１２バイトのブロックの個数を指定する数とすることができる。

【００４２】

第１のステートメントは、crfsコマンド７０１を使用して、ファイル・システムを生成する。フラグは以下の通りである。

【００４３】

-v jfs2 ７０３は、仮想ファイル・システムのタイプを指定する。

【００４４】

-g rootvg ７０５は、ファイル・システムを作成すべき既存のボリューム・グループを指定する。ボリューム・グループは、１つ以上の物理的ボリュームの集まりである。

【００４５】

-m /newfs ７０７は、ファイル・システムが利用可能にされるディレクトリであるマウント・ポイントを指定する。

【００４６】

-a size=<新規ファイル・システム・サイズ> ７０９は、５１２バイトのブロック数でＪＦＳ２のサイズを指定し、その際、指定されたサイズが物理的パーティション・サイズで均等に割り切れない場合、プロセッサは、均等に割り切れる最も近い数に切り上げる。

【００４７】

診断ファイル・システムは、図３の診断ファイル３０５に関連することができる。

【００４８】

mount /newfs ７２１は、ファイル・システムをマウントするようプロセッサに命令する。

【００４９】

次がファイル整合性チェックであり、以下のフラグを含むfsck ７２２コマンドを使用する。

【００５０】

-n ７２３は、非対話モードで、すなわち、修復についてユーザに問い合せず、指定さ

れたファイル・システムを変更せずにファイル整合性チェックを実行するようプロセッサに命令する。

【００５１】

-o metacapture=/newfs/out1 ７２５は、メタデータおよびスパース・オブジェクトのコピーをスパース・オブジェクトを有するファイルに書き込むようプロセッサに命令するパラメータを識別し、その場合、ファイルは、/newfs/out1またはユーザ指定のディレクトリに置かれる。

【００５２】

<ファイル・システム・マウント・ポイント> ７３５は、プロセッサが診断を実行するファイル・システムの識別情報である。

【００５３】

プロセッサは、残りの２つのコマンド・ラインに基づいて統合を行う。backup ７３９は、fsck ７２２で生成されたすべてのファイルを単一のファイルに配置するようプロセッサに命令する。すなわち、診断ファイル・システムを単一のファイルに配置する。backup ７３９は、以下のフラグを含む。

【００５４】

-f <目的ファイル名> ７４１は、出力を保存する１つ以上の装置を識別する。

【００５５】

-0 /newfs ７４３は、バックアップのためのソース・ファイル・システムを指定する。

【００５６】

compress ７５１は、スパース・オブジェクトを圧縮トークンで置き換えるステップである。compress ７５１がとる唯一のパラメータは、稠密診断データが保存されるファイルである<目的ファイル名>である。

【００５７】

統合は、ファイル整合性チェックによって提供された個々のファイルを使い、各ファイルを圧縮することによってもなされ得ることが理解される。その場合、各ファイルが稠密診断データになり、ファイルの集まりに追加される各後続ファイルが、稠密診断データに加わる。

【００５８】

したがって、本発明の側面は、ファイル・システム不整合を修復するために任命された専門家からさえもユーザ・データが保護された方法で、ファイル・システム不整合に関する詳細を取得するためのコンピュータ実施方法を提供する。加えて、本発明の側面は、ファイル・システム開発者が、ファイル・システム不整合に寄与するかもしれない開発中のソフトウェアにおいて問題を診断することを可能にする。さらに、スパース・オブジェクトは、スパース・オブジェクトを論理ボリュームに抽出できる、またはスパース・オブジェクトを直接見てファイル・システム問題をデバッグできるサービス・チームに送られることができる。

【００５９】

本発明の実施形態は、完全なハードウェア実施形態、完全なソフトウェア実施形態、またはハードウェア要素とソフトウェア要素の両方を含む実施形態の形を取ることができる。好ましい一実施形態では、本発明は、ファームウェア、常駐ソフトウェア、マイクロコードなどの、ただしこれらに限定されないソフトウェアで実施される。

【００６０】

さらに、本発明は、コンピュータもしくは任意の命令実行システムによってまたはそれらに関連して使用するためのプログラム・コードを提供するコンピュータ使用可能またはコンピュータ可読媒体等から取得可能なコンピュータ・プログラムの形を取ることができる。この説明では、コンピュータ使用可能またはコンピュータ可読媒体等は、命令実行システム、機器、または装置によってまたはそれらに関連して使用するためのプログラムを格納、保存、伝達、伝播、または転送することができる任意の機器とすることができる。

【００６１】

媒体は、電子、磁気、光学、電磁気、赤外線、または半導体システム（または機器もしくは装置）、あるいは伝播媒体とすることができる。コンピュータ可読媒体の例は、半導体またはソリッド・ステート・メモリ、磁気テープ、着脱可能コンピュータ・ディスケット、ランダム・アクセス・メモリ（ＲＡＭ）、読み取り専用メモリ（ＲＯＭ）、固定磁気ディスク、および光ディスクを含む。光ディスクの現行例は、コンパクト・ディスク－読み取り専用メモリ（ＣＤ－ＲＯＭ）、コンパクト・ディスク－リード／ライト（ＣＤ－Ｒ／Ｗ）、およびＤＶＤを含む。

【００６２】

プログラム・コードを保存または実行し、あるいはその両方を行うのに適したデータ処理システムは、システム・バスを介してメモリ要素に直接的または間接的に結合される少なくとも１つのプロセッサを含む。メモリ要素は、プログラム・コードを実際に実行するときに利用されるローカル・メモリ、大容量ストレージ、実行中に大容量ストレージからコードを取り出さなければならない回数を減らすためにプログラム・コードの少なくとも一部を一時的に記憶するキャッシュ・メモリを含むことができる。

【００６３】

（キーボード、ディスプレイ、ポインティング・デバイスなどを含むが、これらに限定されない）入力／出力またはＩ／Ｏ装置は、直接的にまたは仲介Ｉ／Ｏコントローラを介してシステムに結合されることができる。

【００６４】

データ処理システムが、その他のデータ処理システムまたは遠隔プリンタもしくはストレージに仲介私設または公衆ネットワークを介して結合されることができるように、ネットワーク・アダプタもシステムに結合されることができる。モデム、ケーブル・モデム、およびイーサネット（登録商標）・カードは、現在利用可能なタイプのネットワーク・アダプタのうちのほんの２、３の実例である。

【００６５】

本発明の説明は、例示および説明の目的で提示されたもので、網羅的であること、または本発明を開示された形態に限定することは意図されていない。当業者には、多くの変更および変形が明らかであろう。本発明の例示的な実施形態は、本発明の原理および実際的な応用を最も良く説明し、企図される特定の使用に適した様々な変更を有する様々な実施形態について当業者が本発明を理解できるように選択され、説明された。

【図面の簡単な説明】

【００６６】

【図１】本発明の実施形態が実施され得るデータ処理システムのブロック図である。

【図２】本発明の一実施形態に従って、顧客システムがどのように専門家システムと対話することができるかを示した図である。

【図３】本発明の一実施形態による、ソース・ファイル・システムから診断ファイルへの変換を仲介するオペレーティング・システムの動作を示した図である。

【図４】本発明の一実施形態による、従来のファイル・フォーマットの一例を示した図である。

【図５】本発明の一実施形態による、スパース・ファイル・フォーマットの一例を示した図である。

【図６】本発明の一実施形態による、ファイル・システムに関する診断データを取得する最初のステップを示した図である。

【図７】本発明の一実施形態による、ファイル・システムに関する診断データをパッケージ化して送信するステップを示した図である。

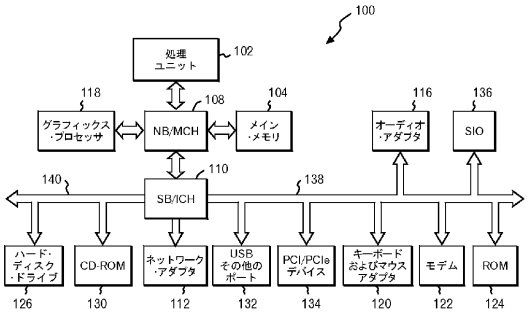
【図８】本発明の一実施形態による、拡張ファイル整合性チェック・ソフトウェアを使用して診断データを取得する詳細なステップを示した図である。

【図９】本発明の一実施形態による、ＡＩＸオペレーティング・システムのコマンド・ラインで入力されるコマンドの一例を示した図である。

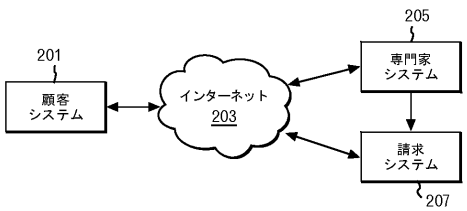
【符号の説明】

【００６７】
２０１ 顧客システム
２０３ インターネット
２０５ 専門家システム
２０７ 請求システム
３０１ オペレーティング・システム
３０３ ソース・ファイル・システム
３０５ 診断ファイル
３１１ コマンド・ライン・エディタ
４１１ ファイル
４１３ 「０」ブロック
４１４ ブロック・ポインタ・テーブル
４１５ ヌル・ポインタ
４１６ ヌル・ポインタ
４１７ ヌル・ポインタ
４１８ ヌル・ポインタ

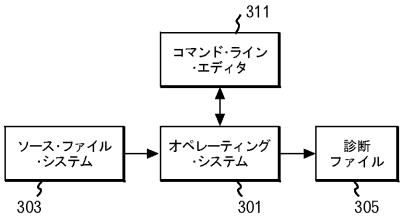
【図１】



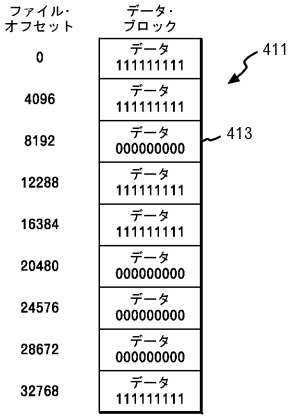
【図２】



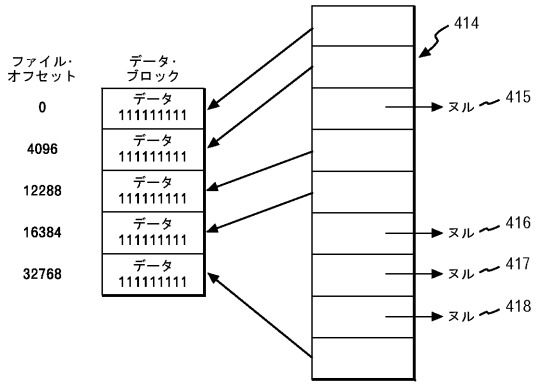
【図３】



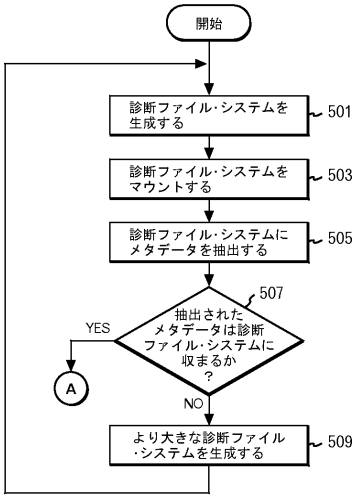
【図４】



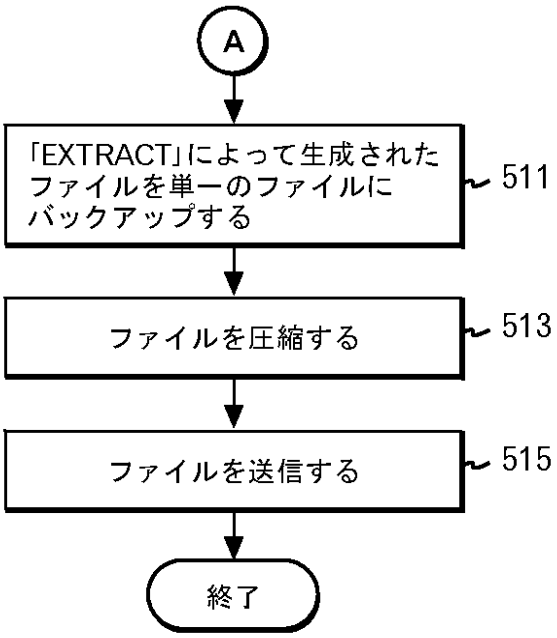
【図５】


ファイル・
オフセット
データ・
ブロック
0
4096
12288
16384
32768
データ
1111111111
414
ヌル
415
416
417
418

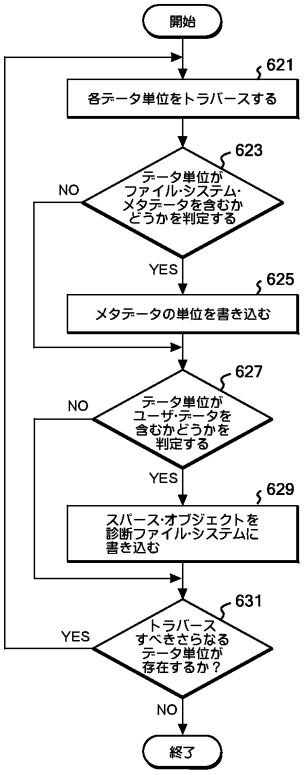
【図６】


開始
診断ファイル・システムを
生成する
501
診断ファイル・システムを
マウントする
503
診断ファイル・システムに
メタデータを抽出する
505
507
抽出された
メタデータは診断
ファイル・システムに
収まるか
？
YES
A
NO
より大きな診断ファイル
・システムを生成する
509

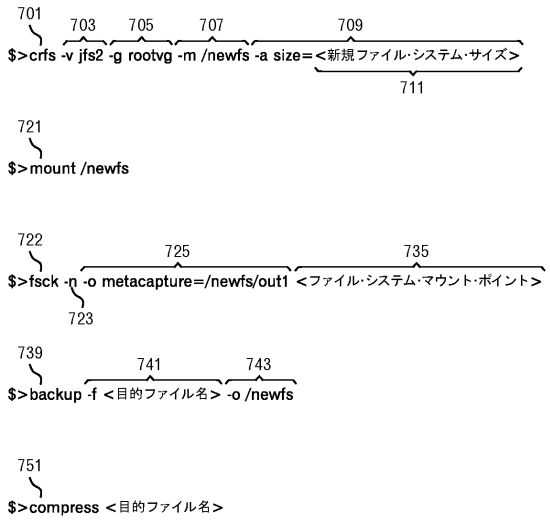
【図７】


A
「EXTRACT」によって生成された
ファイルを単一のファイルに
バックアップする
511
ファイルを圧縮する
513
ファイルを送信する
515
終了

【図８】


開始
621
各データ単位をトラバースする
623
データ単位が
ファイル・システム・
メタデータを含むか
どうかを判定する
NO
YES
625
メタデータの単位を書き込む
627
データ単位が
ユーザ・データを
含むかどうかを
判定する
NO
YES
629
スパース・オブジェクトを
診断ファイル・システムに
書き込む
631
トラバース
すべきさらなる
データ単位が
存在するか？
YES
NO
終了

【図９】


701
703
705
707
709
$>crfs -v jfs2 -g rootvg -m /newfs -a size=<新規ファイル･システム･サイズ>
711
721
$>mount /newfs
722
725
735
$>fsck -n -o metacapture=/newfs/out1 <ファイル･システム･マウント･ポイント>
723
739
741
743
$>backup -f <目的ファイル名> -o /newfs
751
$>compress <目的ファイル名>

フロントページの続き

(74)代理人 100086243
弁理士 坂口 博
(72)発明者 ジャネット・エリザベス・アドキンス
アメリカ合衆国７８７４６ テキサス州オースチン セッジフィールド・ドライブ ５８０２
(72)発明者 マーク・アレン・グラブズ
アメリカ合衆国７８６８１ テキサス州ラウンド・ロック ボブキャット・ドライブ ８６１０

審査官 桜井 茂行

(56)参考文献 特開２００４－０３８９２９（ＪＰ，Ａ）
米国特許出願公開第２００３／０１８２３０１（ＵＳ，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ １２／００
ＪＳＴＰｌｕｓ（ＪＤｒｅａｍＩＩ）