

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号
特許第6370202号
(P6370202)

(45) 発行日 平成30年8月8日 (2018.8.8)

(24) 登録日 平成30年7月20日 (2018.7.20)

(51) Int.Cl.	F I
G06F 3/12 (2006.01)	G06F 3/12 340
G06T 11/20 (2006.01)	G06T 11/20 300
B41J 5/30 (2006.01)	G06F 3/12 311
G06F 9/48 (2006.01)	G06F 3/12 320
H04N 1/387 (2006.01)	G06F 3/12 345
請求項の数 9 (全 22 頁) 最終頁に続く	

(21) 出願番号	特願2014-233765 (P2014-233765)	(73) 特許権者	000001007
(22) 出願日	平成26年11月18日 (2014.11.18)		キヤノン株式会社
(65) 公開番号	特開2016-99663 (P2016-99663A)		東京都大田区下丸子3丁目30番2号
(43) 公開日	平成28年5月30日 (2016.5.30)	(74) 代理人	100090273
審査請求日	平成29年11月15日 (2017.11.15)		弁理士 國分 孝悦
		(72) 発明者	西川 尚之
			東京都大田区下丸子3丁目30番2号 キヤノン株式会社内
		審査官	田川 泰宏
		最終頁に続く	

(54) 【発明の名称】 画像処理装置及び画像処理方法

(57) 【特許請求の範囲】

【請求項 1】

複数ページの処理を並行して行うページ処理手段と、
前記ページ処理手段における負荷レベルをページごとに検出する負荷判定手段と、
前記負荷判定手段による検出結果に応じて前記ページ処理手段の制御を行う制御手段と

を有し、

前記ページ処理手段は、処理対象のページに含まれる1又は2以上の画像データにエッジ抽出処理のスレッドを割り当て、当該スレッドにより抽出された各画像データのエッジを合成し、

前記制御手段は、前記負荷判定手段による判定の結果に応じて、前記ページ処理手段により割り当てられるスレッドの優先順位を指定するスレッド優先順位指定手段を有することを特徴とする画像処理装置。

【請求項 2】

前記負荷判定手段は、2以上の検査処理の基準を有し、処理開始からの経過時間に応じて前記2以上の検査処理の基準のうちから1の基準を選択することを特徴とする請求項1に記載の画像処理装置。

【請求項 3】

前記スレッド優先順位指定手段は、前記負荷判定手段により負荷レベルが閾値よりも高いと検出されたページのスレッドの優先順位を上げることを特徴とする請求項1又は2に

記載の画像処理装置。

【請求項 4】

前記スレッド優先順位指定手段は、前記負荷レベルが閾値よりも高いと検出されたページとは異なるページを処理するスレッドのうちの 1 又は 2 以上の優先順位を下げることを特徴とする請求項 3 に記載の画像処理装置。

【請求項 5】

複数のコアを有しており、

前記スレッド優先順位指定手段は、前記 1 又は 2 以上の優先順位を下げる際に、前記複数のコアの間の接続関係を参照して優先順位を下げるのが可能なスレッドを抽出することを特徴とする請求項 4 に記載の画像処理装置。

10

【請求項 6】

複数のコアを有しており、

前記スレッド優先順位指定手段は、前記 1 又は 2 以上の優先順位を下げる際に、前記複数のコアの間の処理速度の相対関係を参照して優先順位を下げるのが可能なスレッドを抽出することを特徴とする請求項 4 に記載の画像処理装置。

【請求項 7】

前記ページ処理手段により合成されたデータをスプールするスプーラと、

前記スプーラから前記データを読み出して画像を形成する画像形成手段と、

を有することを特徴とする請求項 1 乃至 6 のいずれか 1 項に記載の画像処理装置。

【請求項 8】

20

複数ページの処理を並行して行うページ処理ステップと、

前記ページ処理ステップにおける負荷レベルをページごとに検出する負荷判定ステップと、

前記負荷判定ステップにおける検出結果に応じて前記ページ処理ステップの制御を行う制御ステップと、

を有し、

前記ページ処理ステップでは、処理対象のページに含まれる 1 又は 2 以上の画像データにエッジ抽出処理のスレッドを割り当て、当該スレッドにより抽出された各画像データのエッジを合成し、

前記制御ステップは、前記負荷判定ステップにおける判定の結果に応じて、前記ページ処理ステップにおいて割り当てられるスレッドの優先順位を指定するスレッド優先順位指定ステップを有することを特徴とする画像処理方法。

30

【請求項 9】

コンピュータに、

複数ページの処理を並行して行うページ処理ステップと、

前記ページ処理ステップにおける負荷レベルをページごとに検出する負荷判定ステップと、

前記負荷判定ステップにおける検出結果に応じて前記ページ処理ステップの制御を行う制御ステップと、

を実行させ、

40

前記ページ処理ステップでは、処理対象のページに含まれる 1 又は 2 以上の画像データにエッジ抽出処理のスレッドを割り当て、当該スレッドにより抽出された各画像データのエッジを合成し、

前記制御ステップは、前記負荷判定ステップにおける判定の結果に応じて、前記ページ処理ステップにおいて割り当てられるスレッドの優先順位を指定するスレッド優先順位指定ステップを有することを特徴とするプログラム。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、文字や写真、図形等、多彩な印刷に好適な画像処理装置及び画像処理方法等

50

に関し、特にグラフィックス、図形描画の高速化処理に関する画像処理装置及び画像処理方法等に関する。

【背景技術】

【0002】

従来、図形を描画するために図形が持つ座標情報等から図形の輪郭情報（エッジ情報とも呼ぶ）を抽出し、該輪郭情報に基づいて画像を形成するレンダリング手法は一般に広く利用されている（例えば、特許文献1）。輪郭情報は直線で表現されるが、該直線を表現するための手法としてDDA（digital differential analyzer）アルゴリズムが知られている（非特許文献1）。

【0003】

一般にレンダリング処理において常に高速に描画処理が実行できることは重要な要件であるが、実際にレンダリング処理に要する時間は、図形の種類や数に依存して変化するためこれを実現することは容易ではない。

【0004】

図1は前述したレンダリング処理を含むプリントジョブにおけるデータ処理時間の変化の一例を示したグラフである。図1において、縦軸は処理時間（ミリ秒）を示しており、横軸はプリントジョブのページを示している。図1に示す例では、1ページ目では処理を完了するために約400ミリ秒かかっており、2ページ目では約550ミリ秒の時間を消費する。図1において、三角点で変化している線グラフはジョブに含まれる単純図形の数（相対値）を示しており、丸点で変化している線グラフは単純図形以外の図形（便宜上、ここでは複雑図形という）の数（相対値）を示している。単純図形とは、矩形及び三角形等の単純な図形であり、複雑図形には五角形以上の多角形等が含まれる。複数ページにわたるプリントジョブを分析すると、単純図形が含まれている場合、複雑図形が含まれている場合のいずれにおいても、データ処理の時間に相違があるものの、概ねグラフィックオブジェクト数に対して同様の相関関係を有していることが確認できる。該プリントジョブのようなデータ群に対してレンダリング処理の高速化を目的とした技術は多数提案されている（例えば特許文献2、特許文献3、特許文献4）。

【0005】

図2は、従来の画像処理装置の構成を示す図である。従来の画像処理装置では、印刷が開始されると入力データ（PDL（page description language）データ）910が中間データ生成モジュール911に渡り、中間データ生成モジュール911でPDLインタプリタ及び描画処理（輪郭抽出）等が行われる。そして、各ページにおけるグラフィックスデータ等が後段の画像形成処理に適した形式のデータ（以下、中間データという）に変換される。中間データはディスプレイリスト等とよばれることもある。中間データは中間データスプーラ912に一旦ストアされ、その後に後段の画像形成モジュール913に転送される。画像形成モジュールは中間データの情報に基づいてメモリにピクセルデータを展開する。メモリに展開されたピクセルデータは一旦ビットマップデータ用バッファメモリ914にストアされ、その後にエンジンインタフェース（I/F：interface）915を経由してプリンタエンジン916に転送される。中間データ生成モジュール911、画像形成モジュール913及びエンジンI/F915は制御部900により制御される。

【先行技術文献】

【特許文献】

【0006】

【特許文献1】特開平04-170686号公報

【特許文献2】特開2003-51019号公報

【特許文献3】特開2011-34139号公報

【特許文献4】特開2006-155308号公報

【非特許文献】

【0007】

【非特許文献1】David Rogers, "Procedural Elements for Computer Graphics" McGraw

10

20

30

40

50

-Hill Education

【発明の概要】

【発明が解決しようとする課題】

【0008】

画像処理装置を含む印刷システムにおいては、プリンタエンジンが始動され一旦印刷が開始されると簡単にプリンタエンジンが止められないということもあり、どのようなデータが入ってきても一定の速度でエンジンに印刷データが供給しなくてはならない。このため、時間変動におけるばらつきが大きい場合はスプールすべきデータ量を増やす必要があり、スプーラに用いるメモリサイズを大きく確保する必要がある。しかしながら、メモリサイズの増大は画像処理装置のコストの増加に直結する。

10

【0009】

このような問題に着目した技術として、処理にかかる時間を予測することでプリント処理（レンダリング処理）のスケジュールを変更する技術が例えば特許文献4に記載されている。しかしながら、この技術によれば所期の目的は達成できるものの、実際にこの予測値に基づく処理の切り替えを試みてみると、処理速度の予測そのものが容易ではない。

【0010】

本発明は、メモリサイズの増大を抑制しながら高速印刷を行うことができる画像処理装置及び画像処理方法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】

【0011】

20

処理にかかる時間を予測することでプリント処理のスケジュールを変更する場合、処理を実行する前に得られる情報としては、そのページに含まれるグラフィック数に関する情報が挙げられる。一般的に、グラフィックが多く含まれるページほどその処理にかかる時間が長いという傾向がある。図3は、テストデータにおけるPDLデータの各ページに含まれるグラフィック数と処理時間との関係の例を示すグラフである。図3から、大局的にはグラフィックス数が多くなるほど処理時間が長くなるものの、個々のデータを比較すると処理時間の予測にそのまま用いることが容易ではないことが読み取れる。例えば、ページiは1500個のグラフィックオブジェクト（図形）を含むページであって、その処理に約100ms要している。その一方で、ページjは2000個のグラフィックオブジェクトを含むページであって、その処理に約50ms要している。つまり、この例では、ページjでは、グラフィックオブジェクトの数が4/3倍であるものの、その処理は約1/2の時間で済むのである。

30

【0012】

上記の検証結果に基づき、本願発明者は予測される時間と実際の処理時間との乖離が画像処理装置の特性に及ぼす影響を調査する目的でシミュレーションによる検証を行った。図4は、図2に示す画像処理装置においてエンジンがストールしないために必要となる初期スプール数をシミュレーションで検査した結果を示す図である。

【0013】

図4に示すグラフの横軸は処理が重いページ（以下、ヘビーページということがある）がジョブを占める割合（以下、ヘビーページ含有率という）を示し、縦軸は初期スプールに必要なページ数を示している。ここで、ヘビーページとは処理負荷が重いページを意味し、ヘビーページに該当するか否かは、画像処理装置の用途等に応じて適宜決定することができる。例えば、具体的な定義は各状況により異なる。例えば、100PPM（page per minute）の印刷速度が要求されるプリンタにおいては、その処理に1秒間以上の時間を要するものをヘビーページと定義することができる。また、抽象化するために印刷データにおける各ページの処理時間群を正規分布で捉え適当に設定した閾値（ 1.5σ ）以上の時間を要するページをヘビーページと定義してもよい。図4に示す例では、その処理に1秒間以上の時間を要するページをヘビーページと定義してある。

40

【0014】

このシミュレーションでは、ページの処理時間をランダムに生成し、予測時間は正確に

50

予測できる場合（ミス率：０％）と、予測できない場合（ミス率：１０％、２０％）とでその違いを比較している。画像処理装置の基本動作としてはページ時間の各予測に基づいて、処理ページがヘビーページである場合に、該ページにかかる処理時間に対しては加速処理を適用してその挙動を観察してみた。この加速処理では、シミュレーション上では約半分の時間で処理完了する条件とした。

【００１５】

図４において、線Ａ（二重線、四角点）はプリント処理前の処理予測時間が常に正しかった場合の初期スプール数、つまり予測ミス率が０％時の特性を示している。線Ｂ（破線、黒塗三角点）は予測ミス率が１０％の場合の特性を示し、線Ｃ（実線、白抜き三角点）は予測ミス率が２０％の場合の特性を示している。ここで、ミス率とは、プリント処理時間の予測時間の正確さを表し、総ページ数中の予測が（一定値±１０％よりも）外れた割合を示している。予測通りに処理できたページ数をＰ１、収まらなかったページ数を予測ミスページＰ２とした場合、ミス率は「 $P2 / (P1 + P2) \times 100$ 」で求められる。

10

【００１６】

このシミュレーションの結果によれば、ヘビーページ含有率が１４％程度の場合、予測ミス率が０％のときは、初期スプール数として１９０ページ程度がストアされれば、その後はプリンタエンジンがストールしないで動作ができる。一方、予測ミス率が１０％のときは、５３０ページ程度の初期スプール数が必要とされ、予測ミス率が２０％のときは、７５０ページ程度の初期スプール数が必要とされる。つまり、精度の低い予測の結果に応じた制御を行う場合には、スプーラに大きなサイズのメモリが必要とされ、コストの増加を避けられない。換言すれば、従来の画像処理装置においては、経験的に又は実験的にスプーラ容量を決定していたため、該容量を減らすための分析や工夫がなされないまま無駄なコストを発生させていたということになる。

20

【００１７】

本願発明者は、このような知見に基づいて更に鋭意検討を重ねた結果、以下に示す発明の諸態様に想到した。

【００１８】

本発明に係る画像処理装置は、複数ページの処理を並行して行うページ処理手段と、前記ページ処理手段における負荷レベルをページごとに検出する負荷判定手段と、前記負荷判定手段による検出結果に応じて前記ページ処理手段の制御を行う制御手段と、を有し、前記ページ処理手段は、処理対象のページに含まれる１又は２以上の画像データにエッジ抽出処理のスレッドを割り当て、当該スレッドにより抽出された各画像データのエッジを合成し、前記制御手段は、前記負荷判定手段による判定の結果に応じて、前記ページ処理手段により割り当てられるスレッドの優先順位を指定するスレッド優先順位指定手段を有することを特徴とする。

30

【００１９】

本発明に係る画像処理方法は、複数ページの処理を並行して行うページ処理ステップと、前記ページ処理ステップにおける負荷レベルをページごとに検出する負荷判定ステップと、前記負荷判定ステップにおける検出結果に応じて前記ページ処理ステップの制御を行う制御ステップと、を有し、前記ページ処理ステップでは、処理対象のページに含まれる１又は２以上の画像データにエッジ抽出処理のスレッドを割り当て、当該スレッドにより抽出された各画像データのエッジを合成し、前記制御ステップは、前記負荷判定ステップにおける判定の結果に応じて、前記ページ処理ステップにおいて割り当てられるスレッドの優先順位を指定するスレッド優先順位指定ステップを有することを特徴とする。

40

【発明の効果】

【００２０】

本発明によれば、負荷レベルに応じてスレッドの優先順位が指定されるため、負荷レベルの変動を抑えることができ、これにより、高速処理を行う場合であってもスプーラに準備すべきメモリサイズを抑えることができる。

50

【図面の簡単な説明】

【0021】

【図1】データ処理時間の変化の一例を示したグラフである。

【図2】従来の画像処理装置の構成を示す図である。

【図3】グラフィック数と処理時間との関係の例を示すグラフである。

【図4】エンジンがストールしないために必要となる初期スプール数をシミュレーションで検査した結果を示す図である。

【図5】第1の実施形態に係る画像処理装置の機能構成を示す図である。

【図6】(a)は中間データ生成モジュールの機能構成を示す図であり、(b)は中間データ生成モジュールの処理を示す図である。

10

【図7】図6に示す検査処理における状態遷移を示す状態遷移図である。

【図8】第1の実施形態における加速処理制御部102の動作を示す図である。

【図9】第1の実施形態に係る画像処理装置のハードウェア構成を示す図である。

【図10】第1の実施形態におけるページ処理スケジュールを示す図である。

【図11】(a)は第1の実施形態におけるコア割り当て処理を示すフローチャートであり、(b)はCPU構成情報の一例を示す図である。

【図12】図11に示す処理を適用した具体例を示す図である。

【図13】(a)は第2の実施形態に係る画像処理装置に用いられるCPUチップの構成を示す図であり、(b)はCPUチップの接続形態の一例を示す図であり、(c)はCPUチップの接続形態の他の一例を示す図であり、(d)はコアの組み合わせテストの処理を示すフローチャートである。

20

【図14】(a)は第2の実施形態におけるコア割り当て処理を示すフローチャートであり、(b)はCPU特性データの一例を示す図である。

【図15】第2の実施形態の効果を示す図である。

【図16】第3の実施形態における負荷レベル検出部の動作を示すフローチャートである。

【図17】図16に示す検査処理における状態遷移を示す状態遷移図である。

【図18】(a)は一般的な画像処理装置のページ処理スケジュールを示す図であり、(b)は第3の実施形態に係る画像処理装置のページ処理スケジュールを示す図である。

【図19】第3の実施形態に係る画像処理装置の動作の一例を示す図である。

30

【図20】第4の実施形態に係る画像処理装置の機能構成を示す図である。

【図21】第4の実施形態における加速処理制御部の動作を示す図である。

【発明を実施するための形態】

【0022】

以下、添付の図面を参照しながら本発明の実施形態について詳細に説明する。

【0023】

(第1の実施形態)

まず、第1の実施形態について説明する。図5は、第1の実施形態に係る画像処理装置の機能構成を示す図である。

【0024】

40

第1の実施形態に係る画像処理装置には、制御部100、中間データ生成モジュール111、中間データスプーラ112、画像形成モジュール113、ビットマップデータ用バッファメモリ114及びエンジンI/F115が含まれている。制御部100には、負荷レベル検出部101及び加速処理制御部102が含まれている。中間データ生成モジュール111はページ処理手段の一例であり、負荷レベル検出部101は負荷判定手段の一例であり、加速処理制御部102は制御手段の一例である。

【0025】

第1の実施形態では、PDLデータが入力データ110として中間データ生成モジュール111に入力され、図形等のデータは輪郭等のデータに置き換えられた後、中間データスプーラ112に一時的に保存される。制御部100は、例えば、中間データ生成モジュ

50

ール111、画像形成モジュール113及びエンジンI/F115との間で制御信号の送受信を行い、中間データ生成モジュール111、画像形成モジュール113及びエンジンI/F115を制御する。中間データ生成モジュール111の制御においては、負荷レベル検出部101がその時点で処理対象となっているページの負荷レベルを検出する。負荷レベルの検出の詳細については後述する。本実施形態では、3種類の負荷レベルが定義されており、最も軽い負荷レベルをレベル1、最も重い負荷レベルをレベル3、これらの間の負荷レベルをレベル2としている。例えば、負荷レベルがレベル1のページは600ms未満で処理可能なページであり、負荷レベルがレベル3のページは処理に1s以上かかるページであり、負荷レベルがレベル2のページは処理に600ms以上1s未満かかるページである。換言すると、例えば、レベル1のページの処理では、処理開始から300ms経過した時点で50%以上の処理が完了する。

10

【0026】

本実施形態では、負荷レベル検出部101により、実際に処理をさせている途中の進捗をリアルタイムに計測しながら処理を進める。負荷レベル検出部101にて検出された負荷レベル、即ち検出結果に基づいて、加速処理制御部102がCPUのコアの割り当て数を切り替えることで中間データ生成モジュール111個々の処理能力を適応的に調整する。なお、中間データが中間データスプーラ112にスプールされると、この中間データは画像形成モジュール113に順次入力され、ビットマップデータ用バッファメモリ（ビデオメモリ）114上にピクセルデータを展開される。これらはエンジンI/F115を経由してプリンタエンジン116に画像データとして出力される。

20

【0027】

次に、中間データ生成モジュール111の詳細について説明する。図6(a)は、中間データ生成モジュール111の機能構成を示す図であり、図6(b)は、中間データ生成モジュール111の処理を示す図である。

【0028】

図6(a)に示すように、中間データ生成モジュール111にPDLインタプリタ120が含まれている。PDLインタプリタ120は、入力された入力データ110を解読し、解読されたPDLデータが描画インターフェース(Drawing I/F)を経由して、プリントシステム側に受け渡される。プリントシステム側では、データ受け取り部121が図形、文字、画像等のPDLオブジェクトを受け取り、適時、内部スプーラ(図示せず)にデータを蓄えた後に後段の処理ユニットに受け渡す。例えば、PDLオブジェクトがイメージである場合は、該イメージのデータは画像処理部(画像処理スレッド)(図示せず)に受け渡される。PDLオブジェクトが図形である場合は、該図形のデータはエッジ抽出処理部(エッジ抽出処理スレッド)122-1~122-nに受け渡される。このとき、1つのページに2以上の図形が存在する場合は、各図形に対してエッジ抽出処理スレッドが割り当てられる。そして、それぞれのエッジ抽出処理で抽出されたエッジデータが適宜後段のデータ合成処理部(データ合成処理スレッド)123に受け渡され、ページ上に合成されると共に、つまり重ね合わせの処理が行われると共に、タイル状のデータにフォーマットされる。フォーマットされたデータは中間データとして中間データスプーラ112に書き出される。その後、中間データスプーラ112に書き出された中間データは、画像形成モジュール113及びビットマップデータ用バッファメモリ114を経由してエンジンI/F115に転送される。

30

40

【0029】

図形のデータは、図6(b)に示すように、実体としては図形の種類を表す図形種情報及びそれを構成する座標データの群で構成されている。これらデータからエッジ抽出処理部(エッジ抽出処理スレッド)122-1~122-nによりエッジデータが抽出される。図形1及び図形2が存在する場合、これらは互いに独立した別々のスレッドで処理がなされるために同時に2つの図形のエッジデータが抽出される。この処理は図6(b)ではスレッド-1、スレッド-2として表現されている。また、これらの処理の過程において色情報抽出処理により図形の輪郭の中の塗り情報が同時に抽出される。スレッド-3では

50

スレッドー 1 及びスレッドー 2 により抽出されたエッジデータ及び色情報を用いてデータ合成処理を行う。例えば、図形 2 が赤の単色（不透明）で塗られており、図形 1 が青の単色（不透明）で塗られていて、図形 1 が図形 2 の上に描画されている場合、重なる部分では上層の色である青が塗られることになる。この状態では 2 つの色情報（青及び赤）が保存される。一方、スレッドー 3 が処理の対象とする領域（図 6 の例では矩形タイル）においては図形 1 及び図形 2 の該領域におけるエッジデータの合成、例えば重なり処理の結果、不要になったエッジの消去、が行われる。なお、このような処理では、重なり状況によっては、該領域に関して保持すべき色情報の数が増えることもある。例えば、図形 1 が赤の単色（不透明）で塗られており、図形 2 が青の単色（透過）で塗られている場合で、かつ図形 2 が図形 1 の上に重なっている場合は、透過合成の計算により該重なっている領域の色が計算される。例えば、青と赤を合成すると紫となるため、スレッドー 3 の処理においては赤、青、紫の色情報が保持される。

10

【0030】

このような処理により、従来のペインターズ・アルゴリズムでは困難であった図形（重なった図形を含む）描画処理の並列処理、及び処理時間の調整が可能となる。一般に平面上の図形の描画は上書きの論理があるためにペインターズ処理の場合は全ての処理を順番（シーケンシャルに）実行していく必要がある。これに対し、第 1 の実施形態によれば図形の描画順序を保持しながら適宜、部分領域毎に合成処理を進めるために漸次（プログレッシブ）に、かつ時間独立で処理を進めることが可能である。例えば、図形のエッジ（輪郭）データを生成する処理は、各々独立に処理することが可能である。また、各領域における重ね処理についても、個々の領域において時間独立に処理が可能となる。重ね処理は、例えば、下層の赤色の図形が上層の青色の図形により上書きされる場合、その領域では青色しか描画されない等の処理である。

20

【0031】

次に、負荷レベル検出部 101 の詳細について説明する。図 7 は、図 6 に示す検査処理における状態遷移を示す状態遷移図である。負荷レベル検出部 101 は、ページ処理の負荷をリアルタイムに検出する。負荷レベル検出部 101 による処理は本体の処理に対してはオーバーヘッドとなるため、常に検出処理の時間については考慮する必要がある。本実施形態では最初の検出は一定時間（例えば、600ms の半分の時間（300ms））を経過した時点で、データ処理の進捗（与えられたデータの何割を処理したか）を検査するように構成することでオーバーヘッドを軽減する。

30

【0032】

図 7 において、状態はアイドル（idle）状態からスタートし、最初の検査では周期 f_1 にて検査 i_1 を行う。適宜時間が経過した後に、条件に応じて異なる 2 つの基準が異なる検査処理のいずれかに移行する。どちらの検査処理に移行するかは、例えば処理開始からの経過時間に応じて選択する。一方では、検査周期を f_2 へ変更した上で検査 i_1 とは異なる検査 i_2 を行う。もう一方では、検査周期を f_3 へ変更した上で検査 i_1 とは異なる検査 i_3 を行う。これらは例えばページに含まれるオブジェクトの種類に応じて適宜検査ロジックが切り替えられるように構成されている。例えば、図形の複雑さの違いや、図形や画像の内容に依存して処理内容を適切に切り替える。一般に、処理が進む中で何度も検査をするとオーバーヘッドだけが増えるという課題があるが、第 1 の実施形態によれば処理が半分進むまでは一切、進捗の確認を行わず、半分進んだ所で、軽いか、重いかの判別を開始する。そして、重いデータと判断された場合はより入念な検査を行い、その程度がどのレベルに該当するのかを精密に検査する。

40

【0033】

次に、加速処理制御部 102 の詳細について説明する。図 8 は、第 1 の実施形態における加速処理制御部 102 の動作を示す図である。加速処理制御部 102 は処理単位（スレッド、プロセス等）に対して処理速度の加速減速処理を指示する。図 8 中で処理単位 A, B, C, D はそれぞれ異なる処理単位を示す。図 8 では、例としてページ i 及びページ j の処理が時間軸上で重なった状態で行われる場合を示している。ここでは、図形のエッジ

50

抽出処理は制御部100側からみればスレッドという単位の固まりで認識されている。図8ではページiの処理をページjの処理よりも処理速度の観点から優先して処理する例を示している。これを実現するために、加速処理制御部102が、ページiを処理しているスレッドの優先順位をページjのスレッドの優先順位よりも上げる指令（「+指令」）を発し、ページjに対しては処理スレッドの優先順位を下げる指令（「-指令」）を発する。このようなスレッドの優先順位は加速処理制御部102スレッド優先順位指定手段により行われる。具体的には「+指令」の場合はスレッドを制御する属性パラメータ（Thread関数の属性数値）をインクリメントし、「-指令」の場合はスレッドを制御する属性パラメータの数値をデクリメントする。この結果、CPUコアのスケジュールが変更される。つまり、割り当てられるCPUタイム時間がスレッド単位で変更される。このようにスレッド優先順位指定手段がスレッドの優先順位を指定する。

10

【0034】

図5に示す機能構成は、例えば図9に示すハードウェア構成により実現できる。例えば上記機能構成を実現するプログラムを図9に示すハードウェア構成を備えたコンピュータで実行することができる。図9は、第1の実施形態に係る画像処理装置のハードウェア構成を示す図である。第1の実施形態に係る画像処理装置では、1ボードに2つのCPUソケットが用意されており、これらにCPUチップ200及びCPUチップ300が搭載されている。CPUチップ200には、第1コア201、第2コア202、第3コア203及び第4コア204、これらに対応する4個のL2キャッシュメモリ211～214、ラストレベルキャッシュメモリ222並びにメモリ221が含まれる。ラストレベルキャッシュメモリ222に接続されたプロセッサ間インターコネクト223も含まれる。CPUチップ300には、第5コア305、第6コア306、第7コア307及び第8コア308、これらに対応する4個のL2キャッシュメモリ315～318、ラストレベルキャッシュメモリ322並びにメモリ321が含まれる。ラストレベルキャッシュメモリ322に接続されたプロセッサ間インターコネクト323も含まれる。プロセッサ間インターコネクト223及びプロセッサ間インターコネクト323は高速の接続線であり、互いに結合されている。

20

【0035】

メモリ221及び321の全アドレス空間は二分されており、上位側のアドレス空間はCPUチップ200のメモリ221に、下位側のアドレス空間はCPUチップ300のメモリ321に割り当てられている。CPUチップ200、300は各ラストレベルキャッシュメモリ222、322を経由して全てのアドレス空間に適宜アクセスができる。図6に示すように、CPUチップ200、300には、それぞれ4個のコアが搭載されているので、本実施形態では、1ボードで8個のコアを利用することができる。

30

【0036】

次に、第1の実施形態におけるページ処理スケジュールについて説明する。図10は、第1の実施形態におけるページ処理スケジュールを示す図である。第1の実施形態では、上記のようにマルチコアCPUを用いており、プリント（レンダリング）処理の通常動作としてはページの処理にマルチスレッド処理を適用し、前ページの処理が完了していない場合でも次ページの処理を開始する。例えば、図10に示すように、一つのページ群（4つのページ1～4）の処理を同時に開始し、この処理の開始から一定期間経過した後は、このページ群の処理が完了しているか否かに拘わらず、次のページ群（4つのページ5～8）の処理を開始する。そして、ページ1～4の処理は互いに並行に進むため、これらのページの処理が重なっている時間帯ではCPUチップ200が同時並行で処理を行う。この結果、4つの処理が並行で動作した場合でもそれぞれの処理時間はシングルコアで処理したときの処理時間と比べて変わらず、処理を同時に行った分だけ早く処理が完了する。なお、シングルコアのCPUが用いられる場合は、当該CPUは時分割処理でみせかけ上の並列処理を行うだけで処理能力が増えているわけではないため、4つ同時処理を行っても処理を加速することにはならない。

40

【0037】

50

次に、第1の実施形態におけるコア割り当て処理について説明する。図11(a)は、第1の実施形態におけるコア割り当て処理を示すフローチャートであり、図11(b)は、CPU構成情報の一例を示す図である。

【0038】

第1の実施形態では、図11(a)に示すように、稼働開始時にCPU構成情報を取得する(ステップS11)。CPU構成情報は、例えば図11(b)に示すように、処理コア(起点側)に対して他のコアが同一ソケット上のコアか、別ソケット上のコアかを判定する表(テーブルデータ)であり、複数のコアの間の接続関係を示している。第1コア201を起点とすると、CPU構成情報(テーブルデータ)から、第2コア202、第3コア203及び第4コア204が第1コア201と同一ソケット上に存在していることが判明する。また、第5コア305、第6コア306、第7コア307及び第8コア308が別ソケット上に存在するということも判明する。一般に、同一ソケット上のコアの間では、ラストレベルキャッシュメモリが共有されているため、比較的オーバーヘッドなしでデータのやりとりが可能である。一方、別ソケット上のコアとの間では、プロセッサ間インターコネクトを経由したデータのやりとりが必要となるため、ある程度のオーバーヘッドが発生する。

【0039】

CPU構成情報の取得後、印刷対象のページの処理を実行する(ステップS12)。次いで、負荷レベル検出部101が、その時点で処理対象とするページがヘビーページであるか否か、即ち負荷レベルが閾値よりも高いか否かを検出する(ステップS13)。ヘビーページと検出されなかった場合は通常のページとして処理を進める(ステップS17)。一方、ヘビーページとして検出された場合は負荷レベルを把握する(ステップS14)。負荷レベルは、上述のように負荷レベル検出部101により検出される。その後、負荷レベルに応じて他のコアに応援要請を行う(ステップS15)。例えば、負荷レベルが2であれば処理を行っている起点のコアと同一ソケット上のコアに応援要請を行い、負荷レベルが3であれば起点のコアとは別ソケット上のコアにも応援要請を行う。ここでいう応援要請とは、CPUタイムの融通の要請をいう。そして、他に処理すべきページがあるか否か、即ち、処理を続行するか否かの判定を行い(ステップS16)、他に処理すべきページがあればステップS12に戻り、そうでなければ処理を終了する。

【0040】

次に、図11に示す処理を適用した具体例について説明する。第1の実施形態では、図形の処理にあたって、図6に示す機構でページ内部の処理を並列処理で高速化しているものの、それだけでは、負荷レベルによっては予め想定している規程時間内に処理が完了しないこともあり得る。そこで、第1の実施形態では、負荷レベルに応じたコアの割り当ても行っている。即ち、図8に示すように、加速処理制御部102がヘビーページを処理しているスレッドの優先順位を上げる指令を発する。ただし、例えばヘビーページで使用しているスレッドの優先順位を上げただけでは、他のスレッドにおけるCPUタイムが奪われてしまう。そこで、第1の実施形態では、CPUタイムを奪っても問題のないページ、及びそれを処理しているコアを制御部100が把握する。図12に示す具体例では、ページ4が処理負荷の重いページ(ヘビーページ)であり、これを第4コア204が処理するものとする。また、制御部100が、CPUタイムを奪っても問題のないページとしてページ2、5を把握し、これらを処理しているコアとして第2コア202、第5コア305を把握しているとする。このような場合、図11に示す処理によれば、第4コア204を起点とすると、同一ソケット上にあるコアは第2コア202であり、別ソケット上にあるコアは第5コア305となる。このため、負荷レベルが2であれば第2コア202への応援要請を行い、負荷レベルが3であれば第2コア202及び第5コア305の双方への応援要請を行う。つまり、負荷レベルに応じて第2コア202、第5コア305へのCPUタイムの融通の要請が行われる。

【0041】

このような処理によれば、負荷が比較的小さいヘビーページの処理では、同一ソケット

上のコアに応援要請を行うので効率がよく、負荷が大きいヘビーページの処理では、別ソケット上のコアにも応援要請を行うので、起点コアの負荷をより低減することができる。

【0042】

(第2の実施形態)

次に、第2の実施形態について説明する。第2の実施形態では、CPUチップの構成が第1の実施形態とは相違し、更にこれに伴ってコアの割り当ての処理も第1の実施形態と相違している。他の構成は第1の実施形態と同様である。図13(a)は、第2の実施形態に係る画像処理装置に用いられるCPUチップの構成を示す図である。

【0043】

このCPUチップ400には、第1コア401及び第2コア402、これらに対応する2個のL2キャッシュメモリ411～412、ラストレベルキャッシュメモリ422並びにメモリ421が含まれる。ラストレベルキャッシュメモリ422に接続されたプロセッサ間インターコネクト423も含まれる。つまり、CPUチップ400は2個のコアを備えたデュアルコアプロセッサである。このようなCPUチップ400を用いて8コアの画像処理装置を構成するには、4個のCPUチップ400が必要とされる。そして、その接続形態として種々のものがあり、例えば、図13(b)又は図13(c)に示す接続形態が挙げられる。

10

【0044】

図13(b)に示す例では、2個のCPUソケットが設けられたボードが2個使用される。そして、これら2個のボードが高速スイッチングハブ431により互いに接続されている。ボード内では2個のCPUチップがプロセッサ間インターコネクト423により互いに接続されている。

20

【0045】

図13(c)に示す例では、4個のCPUソケットが設けられたボードが1個使用される。そして、ボード内では各CPUチップが他の3個のCPUチップにプロセッサ間インターコネクト423により接続されている。

【0046】

いずれの例においても、各CPUチップ400のメモリ421の全アドレス空間は適宜分割して割り当てられており、各CPUチップ400は各ラストレベルキャッシュメモリ422を経由して全てのアドレス空間に適宜アクセスができる。

30

【0047】

第2の実施形態では、第1の実施形態のようにCPU構成情報をスタティックかつ正確に把握することは容易ではない。そこで、第2の実施形態では、画像処理装置の内部でコアの組み合わせテストを行い、CPU特性データを更新する。図13(d)は、コアの組み合わせテストの処理を示すフローチャートである。

【0048】

まず、CPU構成情報を読み出す(ステップS21)。次いで、コアの組み合わせテストを行う(ステップS22)。例えば、任意の1個のコアに対して負荷の重い処理を実行させた状態において、他の7個のコアに応援要請を行わせ、その実行速度の比率を相対的に計測する。その後、テスト結果をCPU特性データとして書き出して記録する(ステップS23)。

40

【0049】

このように適宜特性をプロファイリングすることにより、画像処理装置の構成が変化した場合でも優れたパフォーマンスが得られるようにコアの割り当てを調整することができる。

【0050】

図14(a)は、第2の実施形態におけるコア割り当て処理を示すフローチャートであり、図14(b)は、CPU特性データの一例を示す図である。

【0051】

第2の実施形態では、図14(a)に示すように、稼働開始時にCPU特性データを読

50

み出す（ステップS 3 1）。CPU特性データは、例えば図1 4（b）に示すように、処理コア（起点側）に対して他のコアへ応援要請した場合の処理速度の相対的な関係を示す表（テーブルデータ）であり、複数のコアの間の処理速度の相対関係を示している。一般にプロセッサ間インターコネクト等を経由する場合は、そのオーバーヘッド等により同一の処理であっても速度が低下する。CPU特性データは、それらの程度を相対的な指標を示している。速度レベルは1から3に分類されており、CPU特性データは、図1 3（d）のテストの結果から作成されている。

【0 0 5 2】

CPU特性データの取得後、印刷対象のページの処理を実行する（ステップS 3 2）。次いで、負荷レベル検出部1 0 1が、その時点で処理対象とするページがヘビーページであるか否か、即ち負荷レベルが閾値よりも高いか否かを検出する（ステップS 3 3）。ヘビーページと検出されなかった場合は通常のページとして処理を進める（ステップS 3 7）。一方、ヘビーページとして検出された場合は負荷レベルを把握する（ステップS 3 4）。負荷レベルは、上述のように負荷レベル検出部1 0 1により検出される。その後、負荷レベルに応じて他のコアに応援要請を行う（ステップS 3 5）。そして、他に処理すべきページがあるか否か、即ち、処理を続行するか否かの判定を行い（ステップS 3 6）、他に処理すべきページがあればステップS 3 2に戻り、そうでなければ処理を終了する。

【0 0 5 3】

図1 5は、第2の実施形態の効果を示す図である。図1 5の縦軸には、2 5個のページの個々の処理時間を示している。横軸のページの番号は、大きくなるほど処理時間が長くなるように付してある。図1 5に示すように、第2の実施形態によれば、参考例と比較して、負荷が大きいものほど処理時間が短縮され、その一方で、負荷が小さいものほど処理時間が長くなっている。この結果、処理時間のばらつきが抑制されている。このことは、安定した速度で中間データを生成することができることを意味する。なお、参考例は、図2に示す従来の画像処理装置に相当する。

【0 0 5 4】

（第3の実施形態）

次に、第3の実施形態について説明する。第3の実施形態では、負荷レベル検出部1 0 1の構成及び動作が第1の実施形態と相違する。他の構成は第1の実施形態と同様である。図1 6は、負荷レベル検出部1 0 1の動作を示すフローチャートである。

【0 0 5 5】

負荷レベルの検出処理が開始されると（ステップS 4 1）、Get Page（）関数により対象となるページのプロパティ情報にアクセスする（ステップS 4 2）。これにより、処理対象のページに対して検査が開始できる状態となる。次いで、検査処理1として周期 f_1 で定義されたサイクルで検査処理を行う（ステップS 4 3）。単純には周期 f_1 のサイクル（タイミング）で次のステップS 4 5に進んで良いかを調査する。該判定はフラグにセットされ、分岐判断を行う（ステップS 4 4）。ステップS 4 3及びS 4 4は条件が合致するまでループし、この処理ループにおけるオーバーヘッドが最小になるように構成されている。

【0 0 5 6】

ステップS 4 5に進んで良い、つまり調査可能と判断されると、進捗確認処理1としてページ処理の進捗率を確認する（ステップS 4 5）。ここでは、上記の3レベルの負荷レベルが定義されているため、3 0 0 m sを経過した時点でページデータ処理の5 0 %以上が完了しているか否かの確認を行う。そして、分岐判断を行う（ステップS 4 6）。

【0 0 5 7】

5 0 %以上の進捗が確認できた場合は、変数levelに1を代入し（ステップS 4 7）、ステップS 4 6の終了処理に進む。進捗が5 0 %未満であった場合は、検査タイミング周期を変更する（ステップS 4 8）。この場合、つまり進捗が5 0 %未満の場合、当該ページの処理が重く、より正確な情報に基づいて応援（スレッド優先順位のインクリメント）を要請することが確定している。このため、多少オーバーヘッドが増えても構わない

10

20

30

40

50

という理由から検査タイミングの周期等、その手続きを変更する。つまり、ステップ S 4 3 よりも頻繁、かつ正確に検査をするように設定する。次いで、検査処理 2 として周期 f_2 で定義されたサイクルで検査処理を行う（ステップ S 4 9）。単純には周期 f_2 のサイクル（タイミング）で次のステップ S 5 1 に進んで良いかを調査する。該判定はフラグにセットされ、分岐判断を行う（ステップ S 5 0）。ステップ S 4 9 及び S 5 0 は条件が合致するまでループし、この処理ループにおけるオーバーヘッドはステップ S 4 3 及び S 4 4 におけるオーバーヘッドより大きくても構わない。

【0058】

ステップ S 5 1 に進んで良い、つまり調査可能と判断されると、進捗確認処理 2 としてページ処理の進捗率を確認する（ステップ S 5 1）。処理の進捗は時間に比例することを前提として考えると、例えば 300ms で 50%、330ms 経過した時点では 55% を期待する。続けて 360ms 経過した時点では 60% が本来期待される進捗率である。ステップ S 5 1 では、このような期待される進捗度合いからの遅れを算出する。単純には進捗遅延率（達成値／目標値）を算出する。そして、負荷レベルがレベル 2、レベル 3 のいずれであるかを確定するに十分な検査ができたか否かを判定する（ステップ S 5 2）。未だ不十分であればステップ S 4 9 に戻る。

【0059】

十分な検査ができたと判定された場合、負荷レベルがレベル 2 であるか否かの判定を行う（ステップ S 5 3）。そして、レベル 2 でなければ、つまりレベル 3 であれば、変数 `level` に 3 を代入し（ステップ S 5 4）、ステップ S 5 6 の終了処理に進む。一方、レベル 2 であれば、変数 `level` に 2 を代入し（ステップ S 5 5）、ステップ S 5 6 の終了処理に進む。

【0060】

図 1 7 は、図 1 6 に示す検査処理における状態遷移を示す状態遷移図である。状態はアイドル (`idle`) 状態からスタートし、最初の検査では周期 f_1 にて検査 i_1 を行う。適宜時間が経過した後に、検査周期を f_2 へ変更した上で検査 i_1 とは異なる検査 i_2 を行う。一般に、処理が進む中で何度も検査をするとオーバーヘッドも増えてしまい全体のパフォーマンスに影響が出てしまうが、第 3 の実施形態によれば処理が半分進むまでは一切の進捗状況の確認をしないため、オーバーヘッドによる時間のロスが効果的に抑制できる。一方で、処理が半分進んだところで、軽いか重いかというマクロ的なレベルでの判定を行い、更に重いデータと判断された場合は図 1 6 のステップ S 4 9 以降の手順で精査を行った上で、負荷レベルがレベル 2、レベル 3 のいずれであるかの判定を行う。

【0061】

次に、第 3 の実施形態におけるページ処理スケジュールについて、一般的な画像処理装置と比較して説明する。図 1 8 (a) は、一般的な画像処理装置のページ処理スケジュールを示す図であり、図 1 8 (b) は、第 3 の実施形態に係る画像処理装置のページ処理スケジュールを示す図である。

【0062】

一般的な画像処理装置では、各ページのレンダリング処理は前ページのレンダリング処理が終了してから開始される。一方、第 3 の実施形態では、第 1 の実施形態と同様にマルチコア CPU を用いており、プリント（レンダリング）処理の通常動作としてはページの処理にマルチスレッド処理を適用している。そして、ページ処理のスケジュールとしては前ページの処理が完了していないときでも次ページの処理を開始する。図 1 8 (b) に示す例ではページ 2 の処理が完了する前にページ 3 の処理を開始し、ページ 3 の処理が完了する前にページ 4 の処理を開始する。このとき、2 つのページの処理が重なっている時間帯では CPU が同時並行で処理を行う。この結果、2 つの処理が並行で動作した場合でもそれぞれの処理時間はシングルコアで処理したときの処理時間と比べて変わらず、処理を同時に行った分だけ早く処理が完了する。なお、シングルコアの CPU が用いられる場合は、当該 CPU は時分割処理でみせかけ上の並列処理を行うだけで処理能力が増えているわけではないため、2 つ同時処理を行っても処理を加速することにはならない。図 1 8 に

10

20

30

40

50

おける処理タイミングはプリンタエンジンの設計速度（PPM）等から一定に規定されるものである。例えば、100PPMのプリンタエンジンの場合には600msが1ページ毎の基準時間となる。または、中間データスプーラ112の状況等から速度指定が可変であったとしても、ある時間帯においては常に一定にページ処理を開始する。

【0063】

次に、処理対象の複数ページ中に負荷の高いヘビーページが含まれている場合の動作について説明する。図19は、第3の実施形態に係る画像処理装置の動作の一例を示す図である。

【0064】

ここでは、ページ1からページ6までの6ページについて連続して印刷処理が行われることとし、これら6ページの印刷処理が一定の間隔で開始される。ページ1については時刻 t_0 から時刻 t_6 までの時間をかけて処理（レンダリング）がなされ、その出力データが中間データスプーラ112に出力される。スプールされたデータは、時刻 t_8 まで中間データスプーラ112に保持され、その後、エンジンI/F115に出力される。ページ2についてはページ1の処理開始時刻 t_0 から遅れた時刻 t_1 から処理が開始され、時刻 t_7 で完了する。同様に、時刻 t_2 ではページ3の処理が開始され、時刻 t_3 ではページ4の処理が開始され、時刻 t_5 ではページ6の処理が開始される。従って、この例では時刻 t_5 から時刻 t_6 までの間に6ページ分の処理が同時並行で動作することになる。ここでは、説明の都合上、CPUのコア数が十分に確保できていることを想定している。例えば1ページの処理に2コア、6ページ並列で処理することを前提に12コア以上のCPUが使用されていることを想定している。

【0065】

図19に示す例では、ページ3が処理負荷の重いページ（ヘビーページ）である。そして、ページ3の処理は時刻 t_2 から開始されているが、時刻 t_5 を経過したあたりで途中までの進捗状況が把握できるようになる。この進捗状況は、負荷レベル検出部101により把握することが可能である。第3の実施形態でも、図6に示す機構でページ内部の処理を並列処理で高速化しているものの、それだけでは、負荷レベルによっては予め想定している規程時間内に処理が完了しないこともあり得る。そこで、第3の実施形態では、負荷レベルに応じたコアの割り当ても行うようにしている。即ち、図8に示すように、加速処理制御部102がヘビーページを処理しているスレッドの優先順位を上げる指令を発する。ただし、例えばヘビーページで使用しているスレッドの優先順位を上げただけでは、他のスレッドにおけるCPUタイムが奪われてしまう。例えば、図19に示す例で、ページ3で使用しているスレッドの優先順位を上げると、ページ1及びページ2の時刻 t_5 以降の処理に影響が及ぶこととなる。そこで、第3の実施形態では、ヘビーページの次のページの処理時間を融通するように構成している。図19に示す例では、ページ4の処理を行うスレッドに対して優先順位を下げるよう指令を出す。

【0066】

このような加速処理制御が行われると、加速処理制御が行われない場合は時刻 t_a までかかってしまうページ3の処理が、時刻 t_b で完了することとなる。その結果、エンジンI/F115へデータを出力する時刻 t_{10} までにスプールが間に合う。

【0067】

このように、第3の実施形態によれば、各ページの加減速処理を、スレッドの優先順位を各ページの状況に応じて適宜調整することで実現しており、プリンタエンジンのスループットの低下を効果的に防止することが可能となる。

【0068】

（第4の実施形態）

次に、第4の実施形態について説明する。第4の実施形態では、機能構成が第3の実施形態と相違する。他の構成は第3の実施形態と同様である。図20は、第4の実施形態に係る画像処理装置の機能構成を示す図である。

【0069】

第4の実施形態に係る画像処理装置には、中間データ生成モジュール111から提供されるヒント情報を保持するヒント（Hint）情報保持部501が含まれている。加速処理制御部102は負荷レベル検出部101により検出された負荷レベルに応じて、適宜スレッドの優先順位を変更するが、その際に、ヒント情報保持部501に保持されているヒント情報に基づいて優先順位の指定を行う。

【0070】

図21は、第4の実施形態における加速処理制御部102の動作を示す図である。加速処理制御部102は処理単位（スレッド、プロセス等）に対して処理速度の加速減速処理を指示する。図21中で処理単位A、B、C、D、E、Fはそれぞれ異なる処理単位を示す。図21では、例としてページ3、ページ4及びページ5の処理が時間軸上重なった状態で処理されている場合を示している。ここでは、図形のエッジ抽出処理は制御部100側からみればスレッドという単位の固まりで認識されている。図21ではページ3の処理をページ4、5の処理よりも処理速度の観点から優先して処理する例を示している。これを実現するために、加速処理制御部102は、ページ3を処理しているスレッドの優先順位を、ページ3、4のスレッドの優先順位よりも上げる指令を発する。即ち、加速処理制御部102はスレッドを制御する属性パラメータを変更することでCPUコアのスケジュールを変更する。つまり、割り当てられるCPUタイム時間がスレッド単位で変更される。

10

【0071】

第4の実施形態では、優先順位の数値を1から10の自然数のうちから指定可能にしてあり、デフォルトでは標準値で5が指定されている。ここでは、優先順位の数値が大きいほど優先度が高く、10において最優先とされる。

20

【0072】

図21に示すように、図形のエッジ抽出を行う処理では複数のスレッドで処理が行われる。データ等の状況によりスレッドの優先順位を一律に上げるのではなく、処理状況に応じて少しだけ優先度を変更した状態で処理を行った方が都合が良い場合がある。例えば、複雑な輪郭が多くある箇所はシンプルな輪郭の部分よりも時間がかかるため、その度合いに応じて優先度を変更した方が好都合である。そこで、本実施形態では、このような情報は各処理系でまとめられてヒント情報としてヒント情報保持部501に参照が可能な状態で保持される。そして、このヒント情報を加速処理制御部102へ通知することで各スレッドの優先順位の値を制御することが可能となる。

30

【0073】

図21において、ページ3ではAのスレッドよりもBのスレッドの方がより高い数値が指定される。つまり、標準値が5のところ、Aのスレッドには「+1」の指令がされるため、その優先順位は6となり、Bのスレッドには「+2」の指令がされるため、その優先順位は7となる。その一方で、ページ4、5では、「-1」又は「-2」の指令がされ、優先順位が下がることとなる。なお、ページ3の処理が終了した場合には、ページ4、5の優先順位はデフォルトの標準値5に戻す。

【0074】

このように、第4の実施形態は、スレッドの優先順位の値の幅を増した構成となっており、これにより、調整の粒度を下げ、つまりは細粒化し、よりきめ細かく調整できる。また、カレントページ、即ちその時点で処理対象としているページがヘビーである場合でも他のページに一律に応援要請をするのではなく、各ページにおけるその時点での進捗状況に応じて応援要請の程度を調整する。つまり、融通を要請するCPUタイムの程度を各ページの進捗状況に応じて調整する。

40

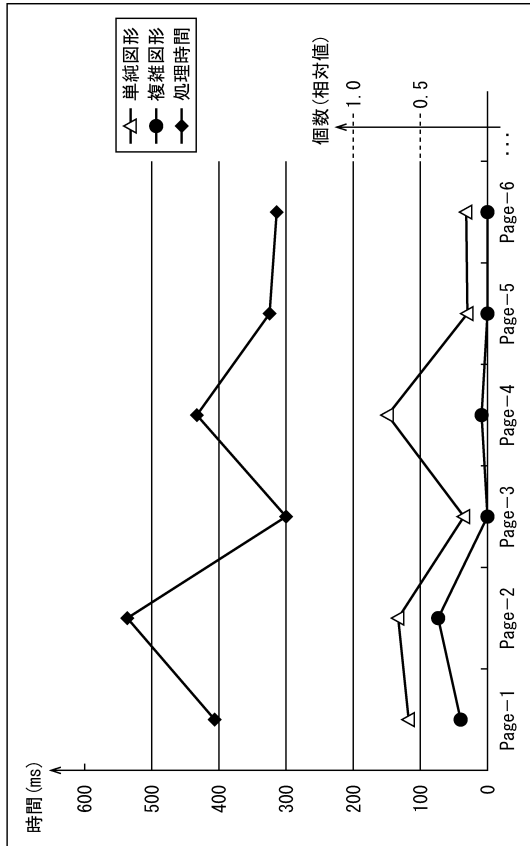
【符号の説明】

【0075】

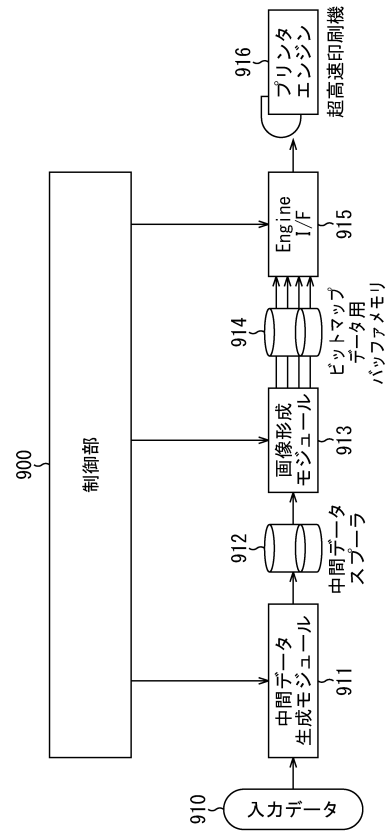
101：負荷レベル検出部 102：加速処理制御部 111：中間データ生成モジュール 112：中間データスプーラ 113：画像形成モジュール 501：ヒント情報保持部

50

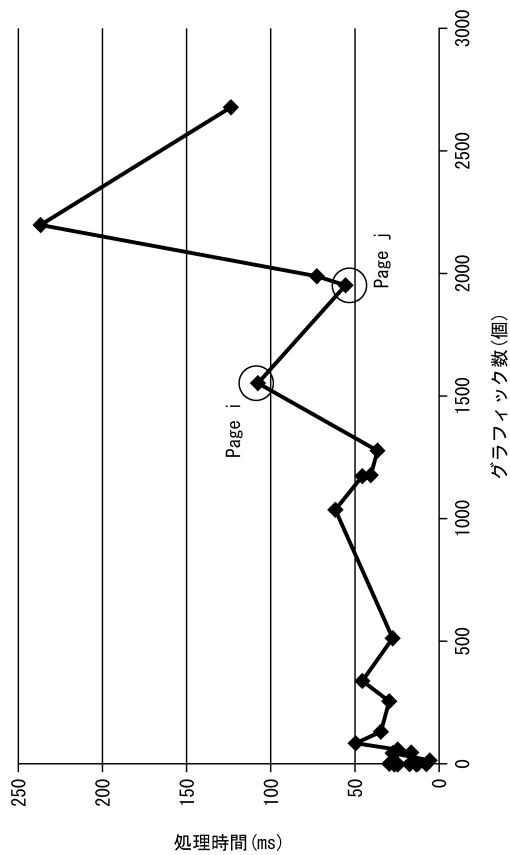
【図 1】



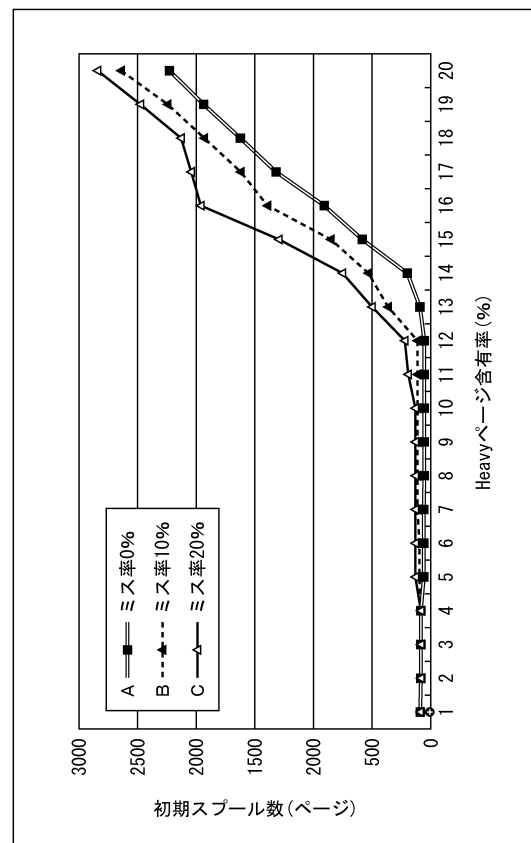
【図 2】



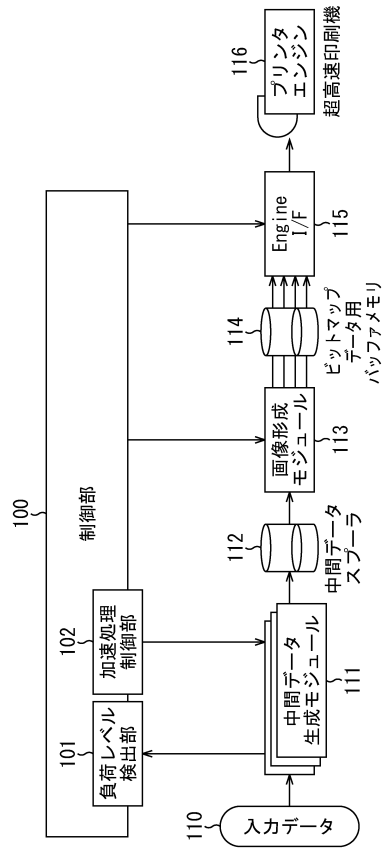
【図 3】



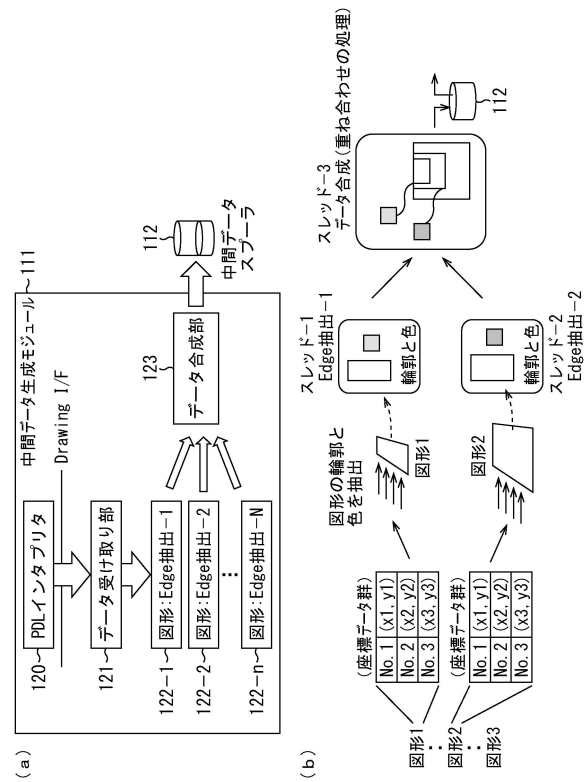
【図 4】



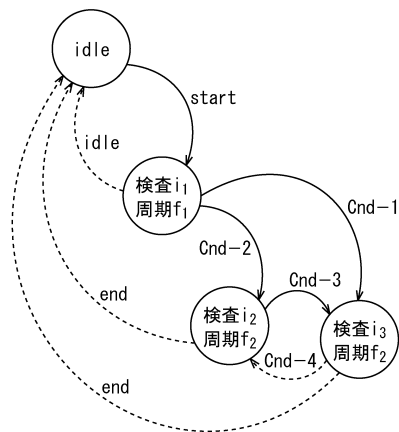
【図 5】



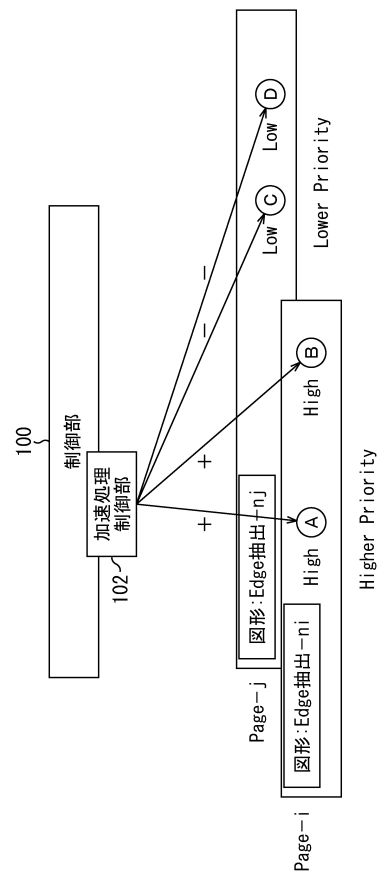
【図 6】



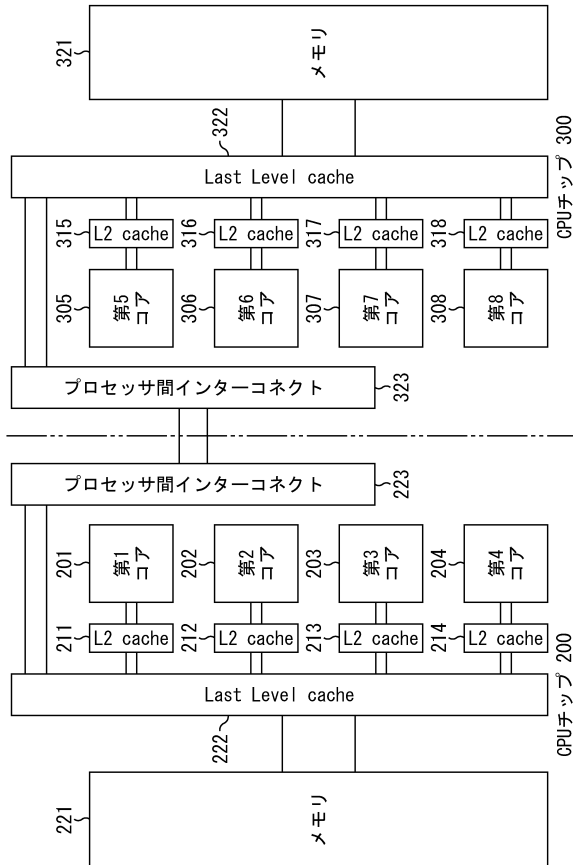
【图 7】



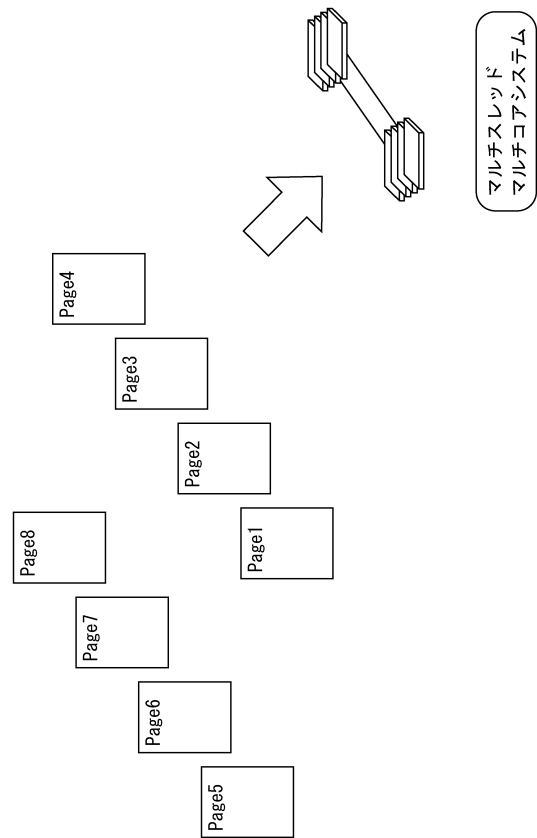
【图 8】



【図 9】



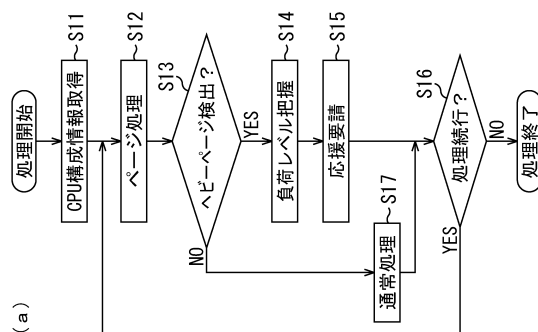
【図 10】



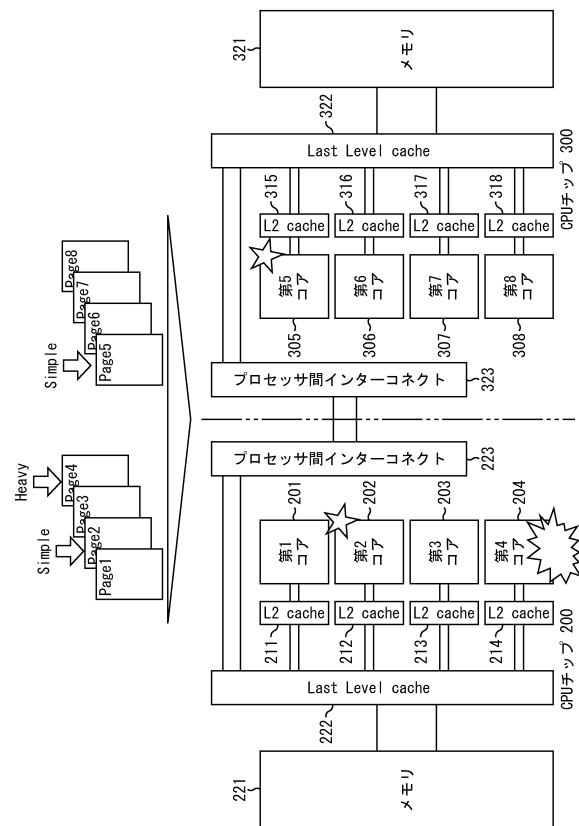
【図 11】

(b) CPU構成情報

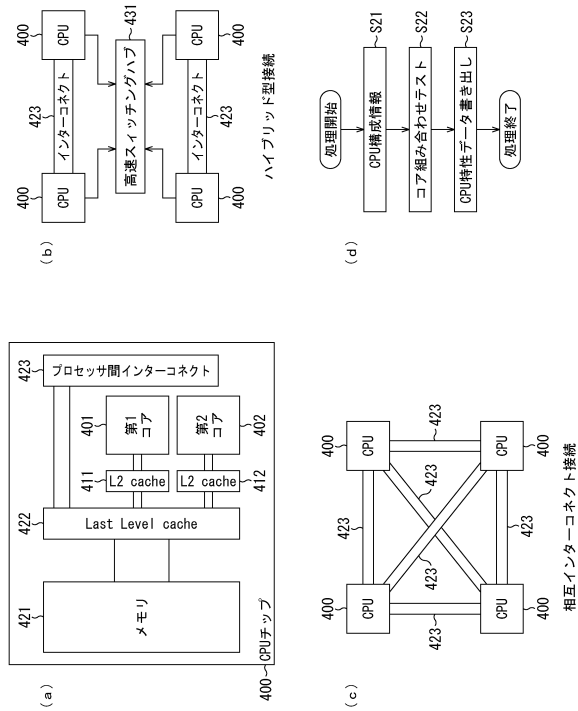
コアNo. (起点側)	同一ソケット上のコアNo.	別ソケット上のコアNo.
1	2, 3, 4	5, 6, 7, 8
2	1, 3, 4	5, 6, 7, 8
3	1, 2, 4	5, 6, 7, 8
4	1, 2, 3	5, 6, 7, 8
5	6, 7, 8	1, 2, 3, 4
6	5, 7, 8	1, 2, 3, 4
7	5, 6, 8	1, 2, 3, 4
8	5, 6, 7	1, 2, 3, 4



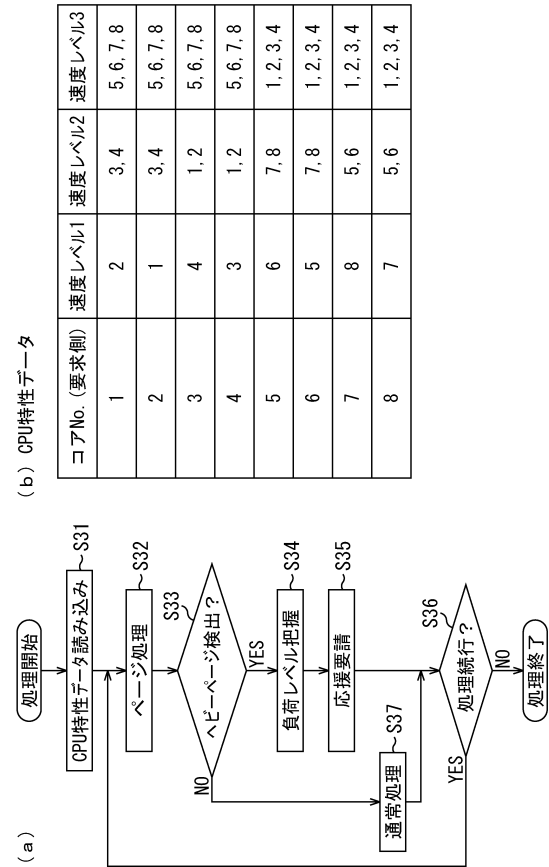
【図 12】



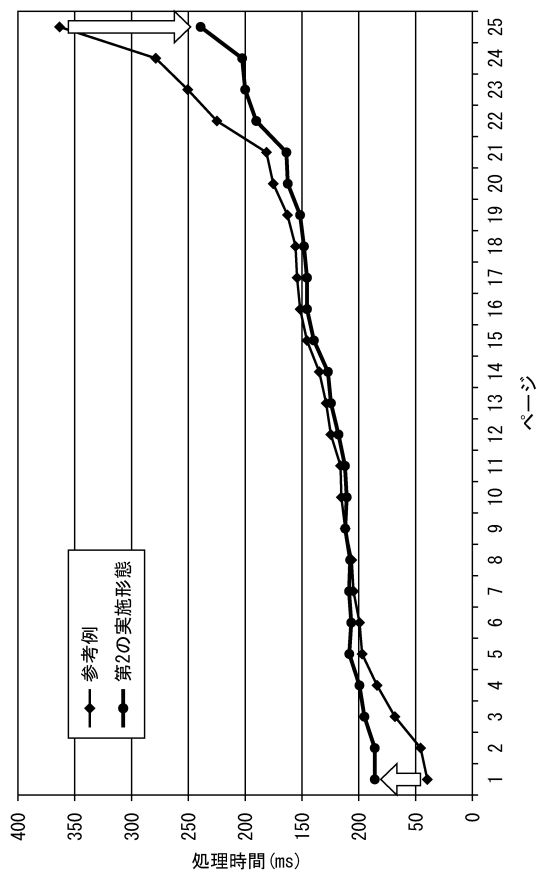
【図 13】



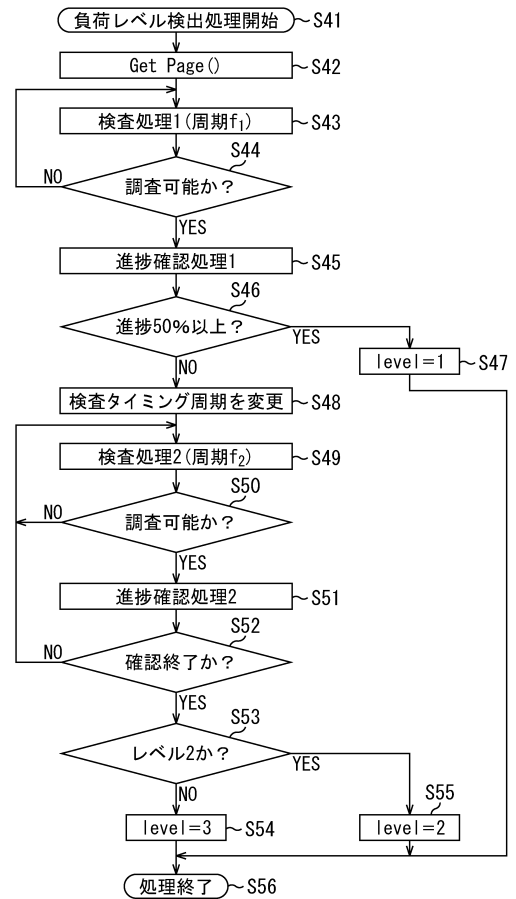
【図 14】



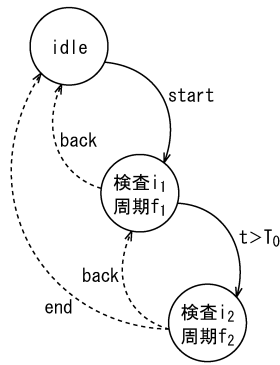
【図 15】



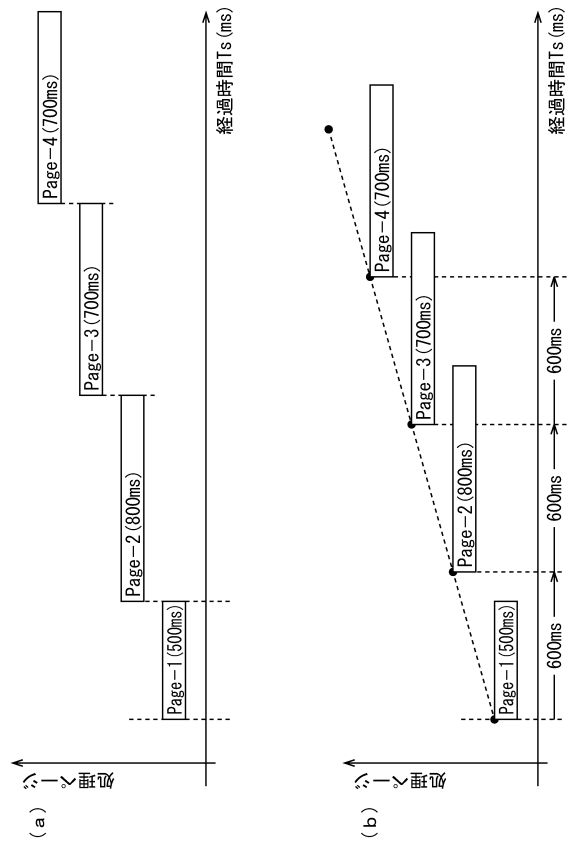
【図 16】



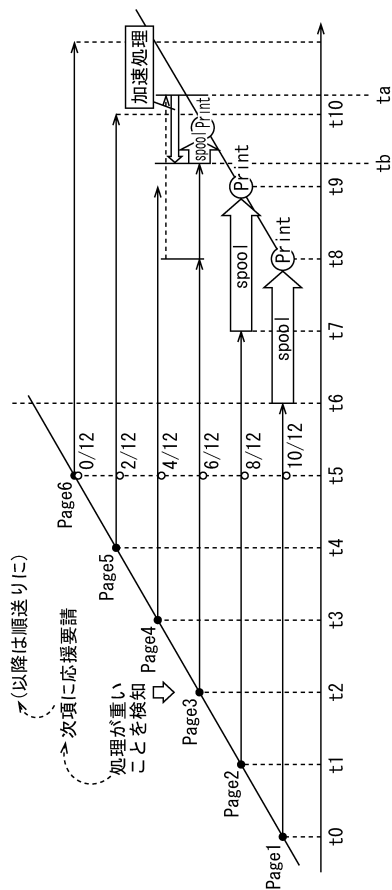
【図 17】



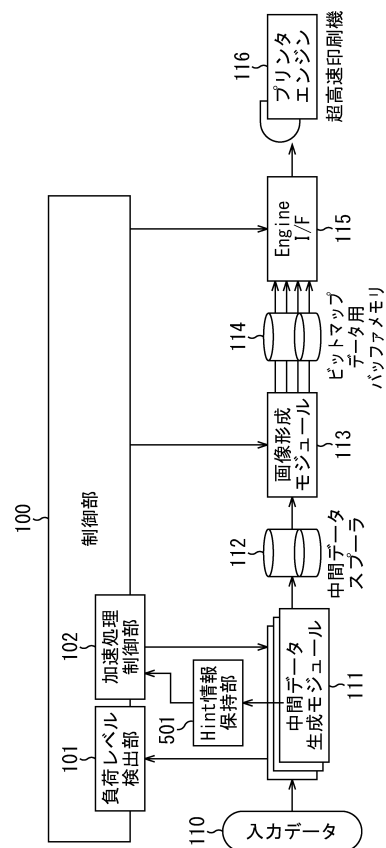
【図 18】



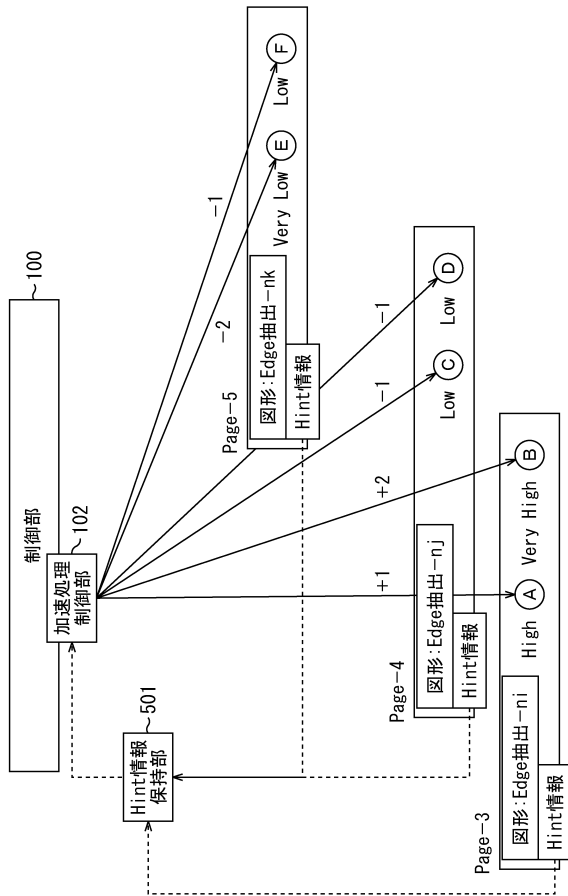
【図 19】



【図 20】



【图 2 1】



フロントページの続き

(51)Int.Cl. F I
G 0 6 F 3/12 3 4 7
B 4 1 J 5/30 Z
G 0 6 F 9/48 3 0 0 B
H 0 4 N 1/387

(56)参考文献 特開2014-091240 (JP, A)
特開2008-276407 (JP, A)
米国特許出願公開第2008/0270753 (US, A1)
特開2012-185697 (JP, A)
米国特許出願公開第2012/0229834 (US, A1)
特開2004-326307 (JP, A)
特開2000-222590 (JP, A)
特開2009-025939 (JP, A)
特開平08-221373 (JP, A)
特開2002-108841 (JP, A)

(58)調査した分野(Int.Cl., DB名)
G 0 6 F 3/12
B 4 1 J 5/30
G 0 6 F 9/48
G 0 6 T 11/20
H 0 4 N 1/387