(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5934392号
(P5934392)
(45)発行日　平成28年6月15日(2016.6.15)　(24)登録日　平成28年5月13日(2016.5.13)

(51)Int.Cl.		F I		
G06F 11/34	(2006.01)	G06F	11/34	S
G06F 11/28	(2006.01)	G06F	11/28	340A
G06F 21/56	(2013.01)	G06F	11/28	340C
		G06F	21/56	320

請求項の数 40　(全 41 頁)

(21)出願番号　特願2014-558752 (P2014-558752)
(86)(22)出願日　平成25年1月30日 (2013.1.30)
(65)公表番号　特表2015-511737 (P2015-511737A)
(43)公表日　平成27年4月20日 (2015.4.20)
(86)国際出願番号　PCT/US2013/023874
(87)国際公開番号　WO2013/130212
(87)国際公開日　平成25年9月6日 (2013.9.6)
審査請求日　平成27年8月24日 (2015.8.24)
(31)優先権主張番号　13/406,272
(32)優先日　平成24年2月27日 (2012.2.27)
(33)優先権主張国　米国 (US)

早期審査対象出願

(73)特許権者　595020643
クゥアルコム・インコーポレイテッド
ＱＵＡＬＣＯＭＭ　ＩＮＣＯＲＰＯＲＡＴＥＤ
アメリカ合衆国、カリフォルニア州　９２１２１－１７１４、サン・ディエゴ、モアハウス・ドライブ　５７７５
(74)代理人　100108855
弁理士　蔵田　昌俊
(74)代理人　100109830
弁理士　福原　淑弘
(74)代理人　100103034
弁理士　野河　信久
(74)代理人　100075672
弁理士　峰　隆司

最終頁に続く

(54)【発明の名称】グラフィック処理ユニットのためのアプリケーションの検証

(57)【特許請求の範囲】
【請求項１】
方法であって、
サーバ・デバイスの外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを、前記サーバ・デバイスによって受信することと、
前記アプリケーションが前記ＧＰＵにおいて非効率的に動作するであろうことを前記サーバ・デバイスによって判定することと、
前記アプリケーションが前記ＧＰＵにおいて非効率的に動作するであろうことを判定することに基づいて、前記サーバ・デバイスによって、前記ＧＰＵにおいて、前記受信したアプリケーションよりもむしろ効率的に動作するであろう、修正されたバージョンのアプリケーションを生成することと、
前記サーバ・デバイスにおける前記修正されたバージョンのアプリケーションの実行中における前記修正されたバージョンのアプリケーションの分析を前記サーバ・デバイスによって実行することと、
前記分析を実行することは、
仮想ＧＰＵモデルを実行することと、
前記仮想ＧＰＵモデルにおいて前記修正されたバージョンのアプリケーションを実行することと、
前記仮想ＧＰＵモデルにおける前記修正されたバージョンのアプリケーションの実行

中、前記仮想ＧＰＵモデルの機能をモニタすることと、を備え、
前記分析に基づいて、前記修正されたバージョンのアプリケーションが、１または複数のパフォーマンス基準を満足しているか否かを判定することと、
前記アプリケーションが、前記１または複数のパフォーマンス基準を満足しているのであれば、前記アプリケーションの前記修正されたコードと前記アプリケーションの検証とを前記デバイスへ送信することと、
を備える方法。
【請求項２】
前記パフォーマンス基準は、前記アプリケーションが、悪意のあるコードではないとの判定と、前記アプリケーションが、エラーのある傾向にないとの判定とのうちの少なくとも１つを備える、請求項１に記載の方法。
【請求項３】
前記パフォーマンス基準は、前記アプリケーションのコードが、既知のウィルスのコードを含んでいないとの判定と、前記アプリケーションのコードのコンパイル中にエラーが発見されないと判定されたとの判定と、前記アプリケーションの実行中にアウト・オブ・バウンズのメモリ・アクセスがないと判定されたとの判定と、前記アプリケーションの実行中に前記デバイスのシステム・バスがオーバロードしていないと判定されたとの判定と、前記アプリケーションのタスクがしきい実行時間内に実行を完了したとの判定と、前記アプリケーションのタスクが少なくともしきい実行レートで実行しているとの判定と、のうちの１または複数を含む、請求項１に記載の方法。
【請求項４】
少なくとも、前記アプリケーションのコードを既知のウィルスのコードと比較することと、前記アプリケーションのコードのコンパイル中に何らかのエラーが発見されたか否かを判定することとによって、コンパイル前およびコンパイル中に、前記アプリケーションの分析を実行することをさらに備える、請求項１に記載の方法。
【請求項５】
仮想デバイス・モデルを実行することと、
前記仮想ＧＰＵモデルにおける前記アプリケーションの実行中、前記仮想デバイス・モデルの機能をモニタすることと、
をさらに備える請求項１に記載の方法。
【請求項６】
前記仮想ＧＰＵモデルにおいて前記アプリケーションを実行することは、
前記仮想ＧＰＵモデルにおいて実行しているアプリケーションに、ＧＰＵ入力を入力すること
を備える、請求項１に記載の方法。
【請求項７】
前記実行されたアプリケーションによって実行される機能をモニタすること、をさらに備える請求項１に記載の方法。
【請求項８】
前記機能をモニタすることは、
前記実行されたアプリケーションによるメモリ・アクセスをモニタすることと、
実行のレートをモニタすることと、
実行時間をモニタすることと、
のうちの１または複数を備える、請求項７に記載の方法。
【請求項９】
前記アプリケーションを受信することはさらに、前記サーバ・デバイスの外部にある前記デバイスに存在するＧＰＵの識別情報を受信することを備え、方法はさらに、
前記ＧＰＵの前記受信された識別情報に基づいて、複数の仮想ＧＰＵモデルのうちの特定の仮想ＧＰＵモデルを識別することを備え、前記仮想ＧＰＵモデルを実行することは、前記識別された特定の仮想ＧＰＵモデルを実行することを備え、前記仮想ＧＰＵモデルに

おいて、前記修正されたバージョンのアプリケーションを実行することは、前記識別された特定の仮想ＧＰＵモデルにおいて、前記修正されたバージョンのアプリケーションを実行することを備える請求項１に記載の方法。
【請求項１０】
前記修正されたバージョンのアプリケーションを生成することは、前記ＧＰＵの受信された識別情報に基づいて、前記アプリケーションのコードを修正することをさらに備える、請求項９に記載の方法。
【請求項１１】
前記アプリケーションをハードウェア・エミュレーション・ボードにおいて実行することと、
前記実行中に、前記ハードウェア・エミュレーション・ボードの機能をモニタすることと
をさらに備え、
前記アプリケーションが、１または複数のパフォーマンス基準を満足するか否かを判定することは、前記アプリケーションが、前記モニタすることのうちの少なくとも１つに基づいて、１または複数のパフォーマンス基準を満足するか否かを判定することを備える、請求項１に記載の方法。
【請求項１２】
前記アプリケーションを受信することは、前記アプリケーションのソース・コードおよび中間コードのうちの少なくとも１つを受信することを備え、
前記方法はさらに、
前記アプリケーションのオブジェクト・コードを生成するために、前記アプリケーションのソース・コードおよび中間コードのうちの少なくとも１つをコンパイルすることと、
前記アプリケーションのオブジェクト・コードを前記デバイスへ送信することと、
を備える請求項１に記載の方法。
【請求項１３】
装置であって、
メモリと、
エミュレータ・ユニットとを備え、
前記エミュレータ・ユニットは、
前記装置の外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信し、
前記アプリケーションが前記ＧＰＵにおいて非効率的に動作するであろうことを判定し、
前記アプリケーションが前記ＧＰＵにおいて非効率的に動作するであろうことを判定することに基づいて、前記ＧＰＵにおいて、前記受信したアプリケーションよりも効率的に動作するであろう、修正されたバージョンのアプリケーションを生成し、
前記装置における前記修正されたバージョンのアプリケーションの実行中における前記修正されたバージョンのアプリケーションの分析を実行し、
実行中における前記修正されたバージョンのアプリケーションの分析を実行することにおいて、
前記メモリに格納された仮想ＧＰＵモデルを実行することと、
前記仮想ＧＰＵモデルにおいて前記修正されたバージョンのアプリケーションを実行することと、
前記仮想ＧＰＵモデルにおける前記修正されたバージョンのアプリケーションを実行する中、前記仮想ＧＰＵモデルの機能をモニタすることと、を備え、
前記分析に基づいて、前記修正されたバージョンのアプリケーションが、１または複数のパフォーマンス基準を満足しているか否かを判定し、
前記アプリケーションが、前記１または複数のパフォーマンス基準を満足しているのであれば、前記アプリケーションの前記修正されたコードと前記アプリケーションの検証

とを前記デバイスへ送信する、
ように構成された、装置。
【請求項１４】
前記パフォーマンス基準は、前記アプリケーションが、悪意のあるコードではないとの判定と、前記アプリケーションが、エラーのある傾向にないとの判定とのうちの少なくとも１つを備える、請求項１３に記載の装置。
【請求項１５】
前記パフォーマンス基準は、前記アプリケーションのコードが、既知のウィルスのコードを含んでいないとの判定と、前記アプリケーションのコードのコンパイル中にエラーが発見されないと判定されたとの判定と、前記アプリケーションの実行中にアウト・オブ・バウンズのメモリ・アクセスがないと判定されたとの判定と、前記アプリケーションの実行中に前記デバイスのシステム・バスがオーバロードしていないと判定されたとの判定と、前記アプリケーションのタスクがしきい実行時間内に実行を完了したとの判定と、前記アプリケーションのタスクが少なくともしきい実行レートで実行しているとの判定と、のうちの１または複数を含む、請求項１３に記載の装置。
【請求項１６】
前記エミュレータ・ユニットは、少なくとも、前記アプリケーションのコードを既知のウィルスのコードと比較することと、前記アプリケーションのコードのコンパイル中に何らかのエラーが発見されたか否かを判定することとによって、コンパイル前およびコンパイル中に、前記アプリケーションの分析を実行するように構成された、請求項１３に記載の装置。
【請求項１７】
前記エミュレータ・ユニットはさらに、
前記メモリに格納された仮想デバイス・モデルを実行し、
前記仮想ＧＰＵモデルにおけるアプリケーションの実行中に、前記仮想デバイス・モデルの機能をモニタする
ように構成された、請求項１３に記載の装置。
【請求項１８】
前記エミュレータ・ユニットは、前記仮想ＧＰＵモデルにおける前記修正されたバージョンのアプリケーションの実行中、前記仮想ＧＰＵモデルにおいて実行している前記修正されたバージョンのアプリケーションへ、前記メモリに格納されたＧＰＵ入力を入力する、請求項１３に記載の装置。
【請求項１９】
前記エミュレータ・ユニットはさらに、前記実行された修正されたバージョンのアプリケーションによって実行される機能をモニタするように構成された、請求項１３に記載の装置。
【請求項２０】
前記エミュレータ・ユニットは、前記実行された修正されたバージョンのアプリケーションによるメモリ・アクセス、実行のレート、および実行時間のうちの１または複数をモニタするように構成された、請求項１９に記載の装置。
【請求項２１】
前記エミュレータ・ユニットはさらに、
サーバ・デバイスの外部にある前記デバイスに存在するＧＰＵの識別情報を受信し、
前記ＧＰＵの前記受信された識別情報に基づいて、複数の仮想ＧＰＵモデルのうちの特定の仮想ＧＰＵモデルを識別するように構成され、
前記エミュレータ・ユニットは、前記識別された特定の仮想ＧＰＵモデルを少なくとも実行することによって、前記仮想ＧＰＵモデルを実行するように構成され、
前記エミュレータ・ユニットは、前記識別された特定の仮想ＧＰＵモデルにおいて、前記修正されたバージョンのアプリケーションを少なくとも実行することによって、前記仮想ＧＰＵモデルにおいて、前記修正されたバージョンのアプリケーションを実行する

ように構成された、請求項１３に記載の装置。
【請求項２２】
前記エミュレータ・ユニットはさらに、
前記ＧＰＵの前記受信された識別に基づいて、前記アプリケーションのコードを少なくとも修正することによって、前記修正されたバージョンのアプリケーションを生成する
ように構成された、請求項２１に記載の装置。
【請求項２３】
前記エミュレータ・ユニットは、ハードウェア・エミュレーション・ボードを備え、前記ハードウェア・エミュレーション・ボードは、
前記修正されたバージョンのアプリケーションの実行中に、前記アプリケーションの分析を実行するために、前記修正されたバージョンのアプリケーションを実行する、請求項１３に記載の装置。
【請求項２４】
前記エミュレータ・ユニットは、前記アプリケーションのソース・コードおよび中間コードのうちの少なくとも１つを受信し、
前記エミュレータ・ユニットはさらに、
前記修正されたバージョンのアプリケーションのオブジェクト・コードを生成するために、前記アプリケーションのソース・コードおよび中間コードのうちの少なくとも１つをコンパイルし、
前記修正されたバージョンのアプリケーションのオブジェクト・コードを前記デバイスへ送信する
ように構成された、請求項１３に記載の装置。
【請求項２５】
サーバ・デバイスであって、
メモリと、
前記サーバ・デバイスの外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信する手段と、
前記アプリケーションが前記ＧＰＵにおいて非効率的に動作するであろうことを判定する手段と、
前記アプリケーションが前記ＧＰＵにおいて非効率的に動作するであろうことを判定することに基づいて、前記ＧＰＵにおいて、前記受信したアプリケーションよりも効率的に動作するであろう、修正されたバージョンのアプリケーションを生成する手段と、
前記サーバ・デバイスにおける前記修正されたバージョンのアプリケーションの実行中における前記修正されたバージョンのアプリケーションの分析を実行する手段と、
前記分析を実行する前記手段は、
前記メモリにおいて仮想ＧＰＵモデルを実行する手段と、
前記仮想ＧＰＵモデルにおいて前記修正されたバージョンのアプリケーションを実行する手段と、
前記仮想ＧＰＵモデルにおける前記修正されたバージョンのアプリケーションの実行中、前記仮想ＧＰＵモデルの機能をモニタする手段と、を備え、
前記分析に基づいて、前記修正されたバージョンのアプリケーションが、１または複数のパフォーマンス基準を満足しているか否かを判定する手段と、
前記アプリケーションが、前記１または複数のパフォーマンス基準を満足しているのであれば、前記アプリケーションの前記修正されたコードと前記アプリケーションの検証とを前記デバイスへ送信する手段と、
を備えるサーバ・デバイス。
【請求項２６】
１または複数のプロセッサに対して、
サーバ・デバイスの外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを、前記サーバ・デバイスによって受信す

ることと、
前記アプリケーションが前記ＧＰＵにおいて非効率的に動作するであろうことを前記サーバ・デバイスによって判定することと、
前記アプリケーションが前記ＧＰＵにおいて非効率的に動作するであろうことを判定することに基づいて、前記サーバ・デバイスによって、前記ＧＰＵにおいて、前記受信したアプリケーションよりもむしろ効率的に動作するであろう、修正されたバージョンのアプリケーションを生成することと、
前記サーバ・デバイスにおける前記修正されたバージョンのアプリケーションの実行中における前記修正されたバージョンのアプリケーションの分析を、前記サーバ・デバイスによって実行することと、前記分析を実行させる命令群は、前記１または複数のプロセッサに対して、
仮想ＧＰＵモデルを実行させ、
前記仮想ＧＰＵモデルにおいて前記修正されたバージョンのアプリケーションを実行させ、
前記仮想ＧＰＵモデルにおける前記修正されたバージョンのアプリケーションの実行中、前記仮想ＧＰＵモデルの機能をモニタさせ、
前記分析のうちの少なくとも１つに基づいて、前記修正されたバージョンのアプリケーションが、１または複数のパフォーマンス基準を満足しているか否かを判定することと、
前記アプリケーションが、前記１または複数のパフォーマンス基準を満足しているのであれば、前記アプリケーションの前記修正されたコードと前記アプリケーションの検証とを前記デバイスへ送信することと、
をさせる命令群を備えた非一時的なコンピュータ読取可能な記憶媒体。
【請求項２７】
方法であって、
デバイスのグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信することと、
前記デバイスの識別されたＧＰＵと関連付けられた仮想ＧＰＵモデルにおける前記アプリケーションの検証のために、前記デバイスの外部にあるサーバ・デバイスへ、前記アプリケーションおよび前記ＧＰＵの識別情報を送信することと、
前記ＧＰＵにおいてより効率的に実行するであろう、修正されたバージョンの前記アプリケーションを、前記サーバ・デバイスから受信することと、
前記修正されたバージョンのアプリケーションが、前記ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、前記サーバ・デバイスから受信することと、
を備える方法。
【請求項２８】
前記受信した検証に基づいて、前記ＧＰＵにおいて前記修正されたバージョンのアプリケーションを実行すること、をさらに備える請求項２７に記載の方法。
【請求項２９】
前記修正されたバージョンのアプリケーションを受信することは、前記修正されたバージョンのアプリケーションのソース・コード、前記修正されたバージョンのアプリケーションの中間コード、および、前記修正されたバージョンのアプリケーションのコンパイルされたコードのうちの少なくとも１つを受信することを備え、
前記アプリケーションを送信することは、前記アプリケーションのソース・コード、前記アプリケーションの中間コード、および、前記アプリケーションのコンパイルされたコードのうちの少なくとも１つを送信することを備える、
請求項２７に記載の方法。
【請求項３０】
前記修正されたバージョンのアプリケーションを前記ＧＰＵにおいて実行すること

をさらに備える請求項２７に記載の方法。
【請求項３１】
前記アプリケーションを送信することは、前記アプリケーションのソース・コードおよび前記アプリケーションの中間コードのうちの少なくとも１つを送信することを備え、
前記修正されたバージョンのアプリケーションを受信することは、前記サーバ・デバイスからの前記修正されたバージョンのアプリケーションの、コンパイルされたオブジェクト・コードを受信することを備え、
前記方法はさらに、
前記修正されたバージョンのアプリケーションの、コンパイルされたオブジェクト・コードを、前記ＧＰＵにおいて実行すること
を備える、請求項２７に記載の方法。
【請求項３２】
前記サーバ・デバイスへ前記アプリケーションを送信することは、前記アプリケーションを一度だけ前記サーバ・デバイスへ送信することを備え、
前記サーバ・デバイスから前記検証を受信することは、前記検証を、前記サーバ・デバイスから、一度だけ受信することを備える、請求項２７に記載の方法。
【請求項３３】
装置であって、
グラフィック処理ユニット（ＧＰＵ）と、
前記ＧＰＵによって実行されるべきアプリケーションを格納するように動作可能なデバイス・メモリと、
前記装置の識別されたＧＰＵと関連付けられた仮想ＧＰＵモデルにおける前記アプリケーションの検証のために、前記装置の外部にあるサーバ・デバイスへ前記アプリケーションおよび前記ＧＰＵの識別情報を送信し、
前記ＧＰＵにおいてより効率的に動作するであろう、修正されたバージョンのアプリケーションを、前記サーバ・デバイスから受信し、
前記修正されたバージョンのアプリケーションが、前記ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、前記サーバ・デバイスから受信する
ように構成されたプロセッサと、
を備える装置。
【請求項３４】
前記プロセッサはさらに、前記受信された検証に基づいて、前記修正されたバージョンのアプリケーションを実行するように前記ＧＰＵに対して指示するように構成され、
前記ＧＰＵは、前記プロセッサからの指示に応じて、前記アプリケーションを実行するように動作可能な、請求項３３に記載の装置。
【請求項３５】
前記プロセッサは、前記修正されたバージョンのアプリケーションのソース・コード、前記修正されたバージョンのアプリケーションの中間コード、および、前記修正されたバージョンのアプリケーションのコンパイルされたコードのうちの少なくとも１つを受信し、
前記プロセッサは、前記アプリケーションのソース・コード、前記アプリケーションの中間コード、および、前記アプリケーションのコンパイルされたコードのうちの少なくとも１つを送信する、請求項３３に記載の装置。
【請求項３６】
前記ＧＰＵは、前記修正されたバージョンのアプリケーションを実行するように構成された、請求項３３に記載の装置。
【請求項３７】
前記プロセッサは、前記アプリケーションのソース・コードおよび前記アプリケーションの中間コードのうちの少なくとも１つを送信し、

前記プロセッサは、前記サーバ・デバイスから、前記修正されたバージョンのアプリケーションの、コンパイルされたオブジェクト・コードを少なくとも受信することによって、前記修正されたバージョンのアプリケーションを受信するように構成され、
前記ＧＰＵは、前記修正されたバージョンのアプリケーションのコンパイルされたオブジェクト・コードを実行するように構成された、
請求項３３に記載の装置。
【請求項３８】
前記プロセッサは、前記アプリケーションを一度だけ前記サーバ・デバイスへ送信し、前記プロセッサは、前記検証を、前記サーバ・デバイスから、一度だけ受信する、請求項３３に記載の装置。
【請求項３９】
デバイスであって、
グラフィック処理ユニット（ＧＰＵ）と、
前記ＧＰＵによって実行されるべきアプリケーションを受信する手段と、
前記デバイスの識別されたＧＰＵと関連付けられた仮想ＧＰＵモデルにおける前記アプリケーションの検証のために、前記デバイスの外部にあるサーバ・デバイスへ、前記アプリケーションおよび前記ＧＰＵの識別情報を送信する手段と、
前記ＧＰＵにおいてより効率的に動作するであろう、修正されたバージョンの前記アプリケーションを、前記サーバ・デバイスから受信する手段と、
前記修正されたバージョンのアプリケーションが、前記ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、前記サーバ・デバイスから受信する手段と、
を備えるデバイス。
【請求項４０】
１または複数のプロセッサに対して、
デバイスのグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信することと、
前記デバイスの識別されたＧＰＵと関連付けられた仮想ＧＰＵモデルにおける前記アプリケーションの検証のために、前記デバイスの外部にあるサーバ・デバイスへ、前記アプリケーションおよび前記ＧＰＵの識別情報を送信することと、
前記ＧＰＵにおいてより効率的に動作するであろう、修正されたバージョンの前記アプリケーションを、前記サーバ・デバイスから受信することと、
前記修正されたバージョンのアプリケーションが、前記ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、前記サーバ・デバイスから受信することと、
を実行させる命令群を備えた非一時的なコンピュータ読取可能な記憶媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本開示は、グラフィック処理ユニット（ＧＰＵ）上で実行するアプリケーションに向けられており、特に、このようなアプリケーションの検証に関する。
【背景技術】
【０００２】
グラフィック処理ユニット（ＧＰＵ）は、従来、非常に制限された機能柔軟性しか提供しない固定機能パイプラインにおけるグラフィック関連処理のみを実行することに制限されている。より新たなＧＰＵは、プログラムを実行するプログラマブル・コアを含んでおり、これによって、従来のＧＰＵに比べてより高い機能柔軟性を提供する。このプログラマブル・コアは、グラフィック関連アプリケーションと非グラフィック関連アプリケーションとの両方を実行しうる。
【発明の概要】

【０００３】
一般に、本開示は、グラフィック処理ユニット（ＧＰＵ）において、実行されるべき潜在的に問題のあるアプリケーションを、実行前に識別するための技法に関する。問題のあるアプリケーションの例は、限定される訳ではないが、悪意のあるアプリケーションのみならず、非効率的な、あるいは、エラーのある傾向にあるアプリケーションを含む。例えば、ＧＰＵを収容するデバイスの外部のサーバ・デバイスが、このアプリケーションを検証しうる。アプリケーションの検証は、アプリケーションが、１または複数の基準を満足することを意味しうる。一例として、検証は、アプリケーションが、悪意のあるアプリケーションでも、エラーのある傾向にあるアプリケーションでも、非効率的なアプリケーションでもないことを、ある保証レベルで判定することを意味しうる。サーバ・デバイスは、そのプログラムを実行することがＧＰＵにとって安全であるか、あるいは推奨できないかを示すインジケーションを、デバイスへ送信しうる。その後、デバイスは、受信したインジケーションに基づいて、このプログラムをＧＰＵにおいて実行することを選択しうる。

【０００４】
一例では、本開示は、サーバ・デバイスの外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを、サーバ・デバイスを用いて受信すること、を含む方法を記載する。この方法はまた、サーバ・デバイスにおけるアプリーションのコンパイル前およびコンパイル中におけるアプリケーションの分析と、サーバ・デバイスにおけるアプリケーションの実行中におけるアプリケーションの分析と、のうちの少なくとも１つを、サーバ・デバイスを用いて実行すること、をも含みうる。この方法はさらに、これら分析のうちの少なくとも１つに基づいて、アプリケーションが、１または複数のパフォーマンス基準を満足するか否かを判定することと、アプリケーションが１または複数のパフォーマンス基準を満足しているのであれば、アプリケーションの検証をデバイスへ送信することと、を含む。

【０００５】
別の例では、本開示は、サーバ・デバイスの外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信するように動作可能なエミュレータ・ユニットを含む装置を記述する。エミュレータ・ユニットはまた、装置におけるアプリーションのコンパイル前およびコンパイル中におけるアプリケーションの分析と、装置におけるアプリケーションの実行中におけるアプリケーションの分析と、のうちの少なくとも１つを、実行するように動作可能である。エミュレータ・ユニットはまた、これら分析のうちの少なくとも１つに基づいて、アプリケーションが、１または複数のパフォーマンス基準を満足するか否かを判定し、アプリケーションが１または複数のパフォーマンス基準を満足しているのであれば、アプリケーションの検証をデバイスへ送信するように動作可能である。

【０００６】
別の例では、本開示は、サーバ・デバイスの外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信する手段を含むサーバを記述する。このサーバ・デバイスはまた、サーバ・デバイスにおけるアプリーションのコンパイル前およびコンパイル中におけるアプリケーションの分析と、サーバ・デバイスにおけるアプリケーションの実行中におけるアプリケーションの分析と、のうちの少なくとも１つを実行する手段をも含む。サーバ・デバイスはさらに、これら分析のうちの少なくとも１つに基づいて、アプリケーションが、１または複数のパフォーマンス基準を満足するか否かを判定する手段と、アプリケーションが１または複数のパフォーマンス基準を満足しているのであれば、アプリケーションの検証をデバイスへ送信する手段とを含む。

【０００７】
別の例において、本開示は、１または複数のプロセッサに対して、サーバ・デバイスの外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行される

べきアプリケーションを、サーバ・デバイスを用いて受信させる命令群を備える非一時的なコンピュータ読取可能な記憶媒体を記述する。これら命令群はさらに、１または複数のプロセッサに対して、サーバ・デバイスにおけるアプリーションのコンパイル前およびコンパイル中におけるアプリケーションの分析と、サーバ・デバイスにおけるアプリケーションの実行中におけるアプリケーションの分析と、のうちの少なくとも１つを、サーバ・デバイスを用いて実行させる。これら命令群はまた、１または複数のプロセッサに対して、これら分析のうちの少なくとも１つに基づいて、アプリケーションが、１または複数のパフォーマンス基準を満足するか否かを判定させ、アプリケーションが１または複数のパフォーマンス基準を満足しているのであれば、アプリケーションの検証をデバイスへ送信させる。

【０００８】

別の例において、本開示は、デバイスのグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信することと、アプリケーションの検証のために、アプリケーションを、デバイスの外部にあるサーバ・デバイスへ送信することと、を含む方法を記述する。この方法はさらに、アプリケーションが、ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、サーバ・デバイスから受信すること、を含む。

【０００９】

別の例では、本開示は、グラフィック処理ユニット（ＧＰＵ）と、ＧＰＵによって実行されるべきアプリケーションを格納するように動作可能なデバイス・メモリとを含む装置を記述する。この装置また、アプリケーションを、装置の外部にあるサーバ・デバイスへ送信し、アプリケーションが、ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、サーバ・デバイスから受信する、ように動作可能なプロセッサを含む。

【００１０】

別の例では、本開示は、グラフィック処理ユニット（ＧＰＵ）を含むデバイスを記述する。このデバイスはまた、ＧＰＵによって実行されるべきアプリケーションを受信する手段と、アプリケーションの検証のために、アプリケーションを、デバイスの外部にあるサーバ・デバイスへ送信する手段と、を含む。デバイスはさらに、ＧＰＵにおける実行のための１または複数の基準をアプリケーションが満足していることを示す検証を、サーバ・デバイスから受信する手段、を含む。

【００１１】

別の例では、本開示は、１または複数のプロセッサに対して、デバイスのグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信させ、アプリケーションの検証のために、アプリケーションを、デバイスの外部にあるサーバ・デバイスへ送信させる命令群を備える非一時的なコンピュータ読取可能な記憶媒体を記述する。これら命令群はさらに、プロセッサに対して、アプリケーションが、ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、サーバ・デバイスから受信させる。

【００１２】

本開示の１または複数の態様の詳細は、添付図面および以下の詳細説明において述べられる。本開示のその他の特徴、目的、および利点は、詳細説明および添付図面から、および特許請求の範囲から明らかになる。

【図面の簡単な説明】

【００１３】

【図１】図１は、本開示の１または複数の態様を実現するように動作可能でありうるシステムの例を例示するブロック図である。

【図２】図２は、本開示の１または複数の態様を実現するように動作可能でありうるデバイスの動作例を例示するフローチャートである。

【図３】図３は、本開示の１または複数の態様を実現するように動作可能でありうるサー

バの動作例を例示するフローチャートである。
【図４】図４は、本開示の１または複数の態様を実現するように動作可能でありうるサーバの別の動作例を例示するフローチャートである。
【図５】図５は、図１に例示されたデバイスの例をさらに詳細に例示するブロック図である。
【発明を実施するための形態】
【００１４】
一般に、本開示は、グラフィック処理ユニット（ＧＰＵ）上で実行されるべきアプリケーションの適切な機能を保証するための技法に関する。以前のいくつかのＧＰＵは、プログラミング機能を提供しない固定機能ハードウェア・パイプラインしか含んでいなかった。しかしながら、新たなＧＰＵは、機能柔軟性を高めるために、プログラマブル・シェダ・コアを考慮している。例えば、これらのＧＰＵは、固定機能ハードウェア・パイプラインの構成要素にすでに委任されている機能を実行する頂点シェダおよびフラグメント・シェダのようなアプリケーションを実行する。
【００１５】
プログラマブル・シェダ・コアは、機能柔軟性を考慮に入れているが、これはまた、ＧＰＵの誤った使用または次善的な使用を招く。例えば、悪意のある開発者は、サービス強制停止攻撃またはウィルスを生成するアプリケーションを開発しうる。それでも、いくつかの事例では、悪意のある意図をもたない開発者が、非効率的な、または、エラーのある傾向にあるアプリケーションを不注意に開発しうる。問題のあるアプリケーション（例えば、悪意のある、非能率的な、エラーのある傾向にあるアプリケーション）は、ＧＰＵや、ＧＰＵが提供されているデバイスの動作に実質的に悪影響を与えうる。
【００１６】
本開示の技法は、ＧＰＵによる実行前に、悪意のある、非効率的な、および／または、エラーのある傾向にある、ＧＰＵ実行アプリケーションを識別することを支援しうる。例えば、本開示の技法は、クラウド・ベースのソリューションに向けられ、ここでは、ＧＰＵを収納するデバイスの外部にあり、ＧＰＵを収納するデバイスに、１または複数のネットワーク接続を介して接続されたサーバ・デバイスが、アプリケーションの実行のためのエミュレータとして機能する。サーバは、あたかもアプリケーションがＧＰＵにおいて実行しているように、このアプリケーションの結果をエミュレートしうる。この結果に基づいて、サーバは、アプリケーションを検証し（例えば、プログラムが悪意のあるものか、非効率的なものか、エラーのある傾向にあるか否かを判定し）、これを、ＧＰＵを収納するデバイスへ示しうる。ＧＰＵは、その後、受信したインジケーションに基づいて、アプリケーションを実行しうる。
【００１７】
アプリケーションンを検証するためにサーバが検証処理を実行するさまざまな手法が存在しうる。検証処理はソフトウェア処理でありうる。ソフトウェア処理は、汎用プロセッサおよび／または特別目的ハードウェアと連携して実行されうる。例えば、サーバは、仮想モデル・ソフトウェアを実行しうる。仮想モデルは、サーバに対して、ＧＰＵ、または、アプリケーションが動作するＧＰＵを含む実際のデバイスを、エミュレートさせる。代替例では、仮想モデルの代わりに、または、仮想モデルに加えて、サーバは、アプリケーションを検証するためのハードウェア・エミュレーション・ボードを含みうる。サーバはまた、ＧＰＵによって実行されるアプリケーションのセキュリティ違反をテストするように特に設計されているアプリケーションを含みうる。
【００１８】
ＧＰＵによって実行されるべきアプリケーションを検証するために、サーバは、静的分析、動的分析、またはこれらの組み合わせを実行しうる。静的分析は、アプリケーションを実行すること無く、実行されうるアプリケーションを分析することを称する。例えば、静的分析は、コンパイル中に実行されうる。コンパイル中、サーバは、例えば、２つの限定しない例として、プログラムにおける無限ループや、アプリケーション内のアレイ位置

へのアウト・オブ・バウンズ・アクセスのような、アプリケーションにおけるエラーを識別しうる。

【００１９】

動的分析は、実行中のアプリケーションの分析を称しうる。これは、結果的に、問題のあるアプリケーション（例えば、悪意のあるアプリケーション、非効率的なアプリケーション、および、エラーのある傾向にあるアプリケーション）を識別しうる。例えば、サーバは、コンパイルされたコードを実行しうる。また、サーバは、実行されたコードに、仮入力値を提供しうる。仮入力値は、例えば、異なる入力画像や、異なるサイズを持つ入力画像等でありうる。

【００２０】

検証処理を実行するサーバは、この結果と、実行されたコードによって実行される機能とをモニタしうる。例えば、サーバは、ＧＰＵの仮想モデルによるメモリ・アクセスをモニタし、このメモリ・アクセスが、アウト・オブ・バウンズ・メモリ・アクセスであるか否かを判定しうる。サーバはまた、ＧＰＵの仮想モデルが情報を書き込むメモリ・アドレスをモニタしうる。ＧＰＵの仮想モデルのメモリ・アクセスと、ＧＰＵの仮想モデルが情報を書き込んでいるメモリ・アドレスとに基づいて、サーバは、このアプリケーションがエラーのある傾向にあるものであるか否かを判定することができうる。このようなメモリ・トラッキングは、特に、アプリケーションが、ポインタを用いて変数を読んだり、変数に書き込んだりする場合に有用でありうる。

【００２１】

サーバはまた、サービス強制停止攻撃を生成またはイネーブルするアプリケーションを検出しうる。例えば、サーバは、ＧＰＵの仮想モデルがアプリケーションを実行することができるレートをモニタしうる。サーバは、遅い応答、意図しない停止、またはハンギングを検出すると、このアプリケーションが、サービス強制停止攻撃のために設計されたアプリケーション、または非常に貧弱に設計されたアプリケーションであると判定しうる。いずれのケースであれ、このようなアプリケーションの実行は、ユーザ経験にネガティブなインパクトを与えうる。

【００２２】

いくつかの例では、サーバは、アプリケーションを検証することに加えて、このアプリケーションを調整および最適化することが可能でありうる。例えば、サーバは、ソース・コード、またはソース・コードの一部を挿入またはリプレースするか、または、コンパイルされたコードがどれくらい良好に動作するかを判定するため統計を収集しうる。いくつかの例では、サーバは、アプリケーションを検証し、アプリケーション・コアを最適化または調整しうる。このような検証後、デバイスは、さらなる検証または最適化を必要とすることなく、ユーザが実行したいだけ、頻繁にアプリケーションを実行しうる。さらに、いくつかの例では、あるアプリケーションの検証後、サーバは、このアプリケーションがすでに検証されていることを示すインジケーションを格納しうる。サーバは、同じソース・コードまたはプレ・コンパイルされたオブジェクト・コードを再度受信すると、まず、このコードが同一であることを確認し、同一であれば、直ちにこのアプリケーションを検証しうる。

【００２３】

図１は、本開示の１または複数の態様を実施するように動作可能でありうるシステムの例を例示するブロック図である。例えば、図１は、デバイス１２、ネットワーク２２、検証サーバ・デバイス２４、およびアプリケーション・サーバ・デバイス３８を含むシステム１０を例示する。図１では、１つのデバイス１２、検証サーバ・デバイス２４、およびアプリケーション・サーバ・デバイス３８しか例示されていないが、その他の例では、システム１０は、複数のデバイス１２、検証サーバ２４、およびアプリケーション・サーバ３８を含みうる。システム１０は、より詳細に説明されるように、デバイス１２の外部にある検証サーバ・デバイス２４において、アプリケーション２０の検証がなされることを示すクラウド・ベースのシステムと称されうる。例えば、本開示の技法は、（例えば、デ

バイス１２の外部にある検証サーバ・デバイス２４内のような）クラウド内においてアプリケーション２０を検証することに向けられうる。

【００２４】

デバイス１２の例は、限定される訳ではないが、メディア・プレーヤのようなビデオ・デバイス、セット・トップ・ボックス、モバイル電話のような無線ハンドセット、携帯情報端末（ＰＤＡ）、デスクトップ・コンピュータ、ラップトップ・コンピュータ、ゲーム・コンソール、ビデオ会議ユニット、タブレット・コンピューティング・デバイス等を含む。検証サーバ・デバイス２４およびアプリケーション・サーバ・デバイス３８の例は、限定される訳ではないが、ラップトップ、デスクトップ、ウェブ・サーバ等を含む。一般に、検証サーバ・デバイス２４およびアプリケーション・サーバ・デバイス３８は、本開示における検証サーバ・デバイス２４およびアプリケーション・サーバ・デバイス３８に起因する機能を実行することが可能な任意のタイプのデバイスでありうる。

【００２５】

ネットワーク２２は、デバイス１２が、検証サーバ・デバイス２４およびアプリケーション・サーバ・デバイス３８とセキュアに通信することを可能にしうる。セキュリティ目的のために、デバイス１２と検証サーバ・デバイス２４との間の何れの通信も、暗号化されうるか、そうではない場合には、セキュア化を図られうる。また、さらなる保護のために、デバイス１２と検証サーバ・デバイス２４との間の何れかの通信が、ユーザ許可を必要としうる。

【００２６】

いくつかの例において、ネットワーク２２は、デバイス１２、検証サーバ・デバイス２４、およびアプリケーション・サーバ・デバイス３８のうちの何れか１つによって送信された情報が、他のデバイスによってではなく、意図されたデバイス（単数または複数）のみによってのみ受信されたことを保証しうる。ネットワーク２２は、ローカル・エリア・ネットワーク（ＬＡＮ）、広域ネットワーク（ＷＡＮ）、インターネット等でありうる。デバイス１２、検証サーバ・デバイス２４、およびアプリケーション・サーバ・デバイス３８は、無線で、または有線リンクによってネットワーク２２に接続されうる。いくつかの例では、デバイス１２が、検証サーバ・デバイス２４および／またはアプリケーション・サーバ・デバイス３８にダイレクトに接続されることが可能でありうる。例えば、デバイス１２は、無線接続または有線接続によって、検証サーバ・デバイス２４および／またはアプリケーション・サーバ・デバイス３８とダイレクトに通信しうる。これらの例において、ネットワーク２２は、システム１０において必要とされない場合がありうる。

【００２７】

図１に例示されるように、デバイス１２は、ＧＰＵ１４、プロセッサ１６、およびデバイス・メモリ１８を含みうる。デバイス１２は、図１で例示されたものに加えた構成要素を含みうる。例えば、図５は、図１に例示されたものよりもより多くの構成要素を含むデバイス１２の例を例示する。

【００２８】

ＧＰＵ１４およびプロセッサ１６の例は、限定される訳ではないが、デジタル信号プロセッサ（ＤＳＰ）、汎用マイクロプロセッサ、特定用途向け集積回路（ＡＳＩＣ）、フィールド・プログラマブル・ロジック・アレイ（ＦＰＧＡ）、または他の等価な統合または離散された論理回路を含む。さらに、ＧＰＵ１４およびプロセッサ１６は、個別の構成要素として例示されているが、本開示の態様はそれに限定されない。代替例では、ＧＰＵ１４およびプロセッサ１６は、共通の集積回路の一部でありうる。例示および説明を容易にする目的のために、ＧＰＵ１４およびプロセッサ１６は、個別の構成要素として例示されている。

【００２９】

デバイス・メモリ１８の例は、限定される訳ではないが、ランダム・アクセス・メモリ（ＲＡＭ）、読取専用メモリ（ＲＯＭ）、または電子的消去可能プログラマブル読取専用メモリ（ＥＥＰＲＯＭ）を含む。デバイス・メモリ１８の例はまた、例えばＣＤ－ＲＯＭ

またはその他の光ディスク記憶装置、磁気ディスク記憶装置、またはその他の磁気記憶デバイス、フラッシュ・メモリのような記憶デバイスを含みうる。一般に、デバイス・メモリ１８は、命令群またはデータ構造の形態で所望のプログラム・コードを格納するために使用され、かつ、ＧＰＵ１４およびプロセッサ１６によってアクセスされうる媒体を含みうる。いくつかの例では、デバイス・メモリ１８は、例えばコンピュータ読取可能な記憶デバイスのような１または複数のコンピュータ読取可能な記憶媒体を備えうる。例えば、本開示では、いくつかの実施の例において、デバイス・メモリ１８は、ＧＰＵ１４およびプロセッサ１６に対して、ＧＰＵ１４およびプロセッサ１６に割り当てられた機能を実行させる命令群を含みうる。

【００３０】

デバイス・メモリ１８は、いくつかの例では、非一時的な記憶媒体であると考えられうる。「非一時的な」という用語は、記憶媒体が、搬送波または伝搬信号で具体化されないことを示しうる。しかしながら、「非一時的な」という用語は、デバイス・メモリ１８が、移動可能ではないことを意味するように解釈されるべきではない。一例として、デバイス・メモリ１８は、デバイス１２から取り除かれ、別のデバイスへ移されうる。別の例として、デバイス・メモリ１８に実質的に類似した記憶デバイスが、デバイス１２へ挿入されうる。ある例において、非一時的な記憶媒体は、（例えば、ＲＡＭにおいて）経時的に変動しうるデータを格納しうる。

【００３１】

ＧＰＵ１４は、１または複数のソフトウェア・アプリケーションを実行するように動作可能でありうる。例えば、ＧＰＵ１４は、１または複数のソフトウェア・アプリケーションが実行しうるプロセッサ・コアを含みうる。ＧＰＵ１４で実行するアプリケーションは、例えば、グラフィック・データを生成するための頂点シェダおよびフラグメント・シェダのようなグラフィック・アプリケーションでありうる。しかしながら、ＧＰＵ１４において動作するアプリケーションが、グラフィック処理に無関係になるとが可能でありうる。例えば、開発者は、ＧＰＵ１４の高い並列処理を活用し、ＧＰＵ１４の高い並列処理を活用するグラフィック処理に無関係なソフトウェア・アプリケーションを開発することが有益であると考えうる。これらのケースでは、ＧＰＵ１４は、汎用ＧＰＵ（ＧＰ－ＧＰＵ）と称されうる。

【００３２】

一例として、図１は、アプリケーション２０を実行するＧＰＵ１４を例示する。アプリケーション２０は、ＧＰＵ１４で実行する非グラフィック・アプリケーションまたはグラフィック・アプリケーションでありうる。アプリケーション２０は、アプリケーション２０がＧＰＵ１４で実行していることを示すために、ＧＰＵ１４内の破線のボックス内に例示されている。ＧＰＵ１４は、実際には、アプリケーション２０を含んでいない。例えば、図１に例示されるように、アプリケーション２０は、デバイス・メモリ１８に格納されうる。

【００３３】

アプリケーション２０は、種々様々な異なるプログラミング・アプリケーション処理インタフェース（ＡＰＩ）を用いて開発されうる。例えば、開発者は、ＯｐｅｎＧＬ、ＯｐｅｎＣＬ、ＷｅｂＧＬ、およびＷｅｂＣＬのような任意のプログラミングＡＰＩを用いてアプリケーション２０を開発してきたかもしれない。一般に、ＯｐｅｎＧＬまたはＷｅｂＧＬのＡＰＩを用いて開発されたアプリケーションは、グラフィック処理のために設計されている。ＯｐｅｎＣＬまたはＷｅｂＣＬのＡＰＩを用いて開発されたアプリケーションは、グラフィック処理に無関係な処理のために設計されている。ＯｐｅｎＧＬ、ＯｐｅｎＣＬ、ＷｅｂＧＬ、およびＷｅｂＣＬのＡＰＩは、例示目的のために提供され、限定すると考えられるべきではない。本開示の技法は、上記提供された例に加えて、ＡＰＩに拡張可能でありうる。一般に、本開示の技法は、アプリケーション２０を開発するために開発者によって利用される任意の技法に拡張可能でありうる。

【００３４】

例示されるように、デバイス・メモリ１８は、アプリケーション２０を格納しうる。例えば、デバイス１２のユーザは、デバイス１２に対して、ネットワーク２２を介して、アプリケーション・サーバ・デバイス２０からアプリケーション２０をダウンロードさせうる。一方、デバイス１２は、デバイス・メモリ１８にアプリケーション２０を格納しうる。デバイス１２が、デバイス・メモリ１８にアプリケーション２０を格納する他の手法がありうる。例えば、デバイス１２のユーザは、アプリケーション２０を格納するデバイス１２へ、フラッシュ・ドライブを挿入し、デバイス１２は、このフラッシュ・ドライブからアプリケーション２０を取得し、アプリケーション２０をデバイス・メモリ１８に格納しうる。この例において、アプリケーション・サーバ・デバイス３８は必要とされない場合がありうる。デバイス１２が、デバイス・メモリ１８にアプリケーション２０を格納する手法を記載している上記例は、例示目的のために提供され、限定すると考えられるべきではない。本開示の技法は、アプリケーション２０がデバイス・メモリ１８にロードされうるあらゆる技法に適用可能でありうる。

【００３５】

デバイス・メモリ１８は、アプリケーション２０のソース・コード、アプリケーション２０の中間表現、またはアプリケーション２０のオブジェクト・コードを格納しうる。アプリケーション２０のソース・コードは、アプリケーション２０が開発されたプログラミング言語におけるテキストでありうる。アプリケーション２０のオブジェクト・コードは、アプリケーション２０のコンパイルの結果生じる２進数のビットでありうる。例えば、アプリケーション・サーバ・デバイス３８は、アプリケーション２０のソース・コードをコンパイルし、デバイス１２は、アプリケーション２０の、このプレ・コンパイルされたオブジェクト・コードをダウンロードしうる。アプリケーション２０の中間表現は、ソース・コードおよびオブジェクト・コードに対する中間物でありうる。例えば、アプリケーション２０の中間表現では、アプリケーション２０のソース・コードの変数が、これら変数がデバイス・メモリ１８内に格納されるためのレジスタ識別子またはメモリ識別子と交換されうる。

【００３６】

例えばアプリケーション２０のようなアプリケーションを実行するために、ＧＰＵ１４のプログラマブル・コア（単数または複数）の能力は、ＧＰＵ１４の機能を増加させる。しかしながら、アプリケーションを実行するＧＰＵ１４の能力は、ＧＰＵ１４の誤使用または次善使用をもたらし、デバイス１２を、悪意のあるアプリケーションまたはエラーのある傾向にあるアプリケーションに対して、より影響を受け易くしうる。例えば、例えばプロセッサ１６のような中央処理ユニット（ＣＰＵ）において専ら動作するアプリケーションは、アプリケーションにアクセス可能であるデバイス・メモリ１８内の格納位置、および、デバイス・メモリ１８のメモリ量を割り当てる仮想マシン設定においてアプリケーションを実行する。アプリケーションは、プロセッサ１６の仮想マシンに制限されるので、アプリケーションは、アウト・オブ・バウンズのメモリ・アドレスにアクセスすることはできず、プロセッサ１６の仮想マシンによって、特にこの仮想マシンに提供されるメモリ・アドレスへアクセスすることに限定される。このように、プロセッサ１６において動作しているアプリケーションが、一転してネガティブな方式で、プロセッサ１６およびデバイス１２へ、ドラスティックに悪影響を与えるということは難しいであろう。

【００３７】

いくつかの事例では、ＧＰＵ１４の仮想マシンを実現することは現実的ではない場合がありうる。例えば、ＧＰＵ１４の高い並列処理能力は、仮想マシンの実行によく適していないことがありうる。例えば、仮想マシンがＧＰＵ１４において動作するはずであった場合、仮想マシンは、ＧＰＵ１４のリソースを支配し、恐らくは、他のアプリケーションが、ＧＰＵ１４において動作することを制限するだろう。したがって、いくつかの事例では、仮想マシンは、ＧＰＵ１４において動作する悪意のある、またはエラーのある傾向にあるアプリケーションのネガティブなインパクトを制限することができないことがありうる。

【００３８】
例えばアプリケーション２０のように、ＧＰＵ１４において動作するアプリケーションは、「ネイティブに」（すなわち、仮想マシンに制限されずに）動作するアプリケーションとして考えられうる。アプリケーション２０のネイティブな実行によって、アプリケーション２０は、デバイス・メモリ１８の大部分にアクセスすることが可能となりうる。このようなアクセスによって、例えば悪意のあるアプリーションまたは貧弱に設計された（例えば、エラーのある傾向にある）アプリケーションのように、問題のあるアプリケーションは、ＧＰＵ１４およびデバイス１２のパフォーマンス能力にネガティブなインパクトを与えるようになりうる。
【００３９】
一例として、アプリケーション２０の開発者は、実行された場合に、サービス強制停止攻撃をデバイス１２に発動するか、または、デバイス１２のパフォーマンスにインパクトを与えるウィルスを伝搬させるように、アプリケーション２０を開発しうる。例えば、ＧＰＵ１４がアプリケーション２０を実行した場合、アプリケーション２０は、ユーザ・インタフェースのグラフィック・コンテンツをレンダリングするようなその他任意のタスクをＧＰＵ１４が実行できないようにＧＰＵ１４を制御しうる。これによって、デバイス１２は、「ハング」させうる。これは、デバイス１２の機能にドラスティックにインパクトを与えうる。いくつかのケースでは、アプリケーション２０の開発者は、アクセスが制限されねばならないデバイス・メモリ１８の一部にアクセスするためのアプリケーションを開発しうる。アプリケーション２０は、デバイス・メモリ１８のこれらの部分に、ウィルスのための命令群を格納しうる。その後、プロセッサ１６またはＧＰＵ１４は、デバイス・メモリ１８のこれら部分にアクセスし、プロセッサ１６またはＧＰＵ１４は、格納されたウィルスを偶然的に実行させうる。悪意のあるアプリケーションの追加の例が存在し、本開示の態様は、サービス強制停止攻撃またはウィルスに制限されていると考慮されるべきでない。
【００４０】
別の例として、アプリケーション２０の開発者は、アプリケーション２０が非効率的またはエラーのある傾向にあるように、アプリケーション２０を不注意に開発しうる。例えば、エラーのある傾向にあるアプリケーションは、無限ループ、アレイへのアウト・オブ・バウンズ・アクセス、または、デバイス・メモリ１８のメモリ位置に対するアウト・オブ・バウンズ・アクセスを含みうる。非効率的なアプリケーションは、ＧＰＵ１４の機能を適切に利用しないことがありうる。例えば、非効率的なアプリケーションは、ＧＰＵ１４のプログラマブル機能を適切に利用しないことがありうる。
【００４１】
いくつかのケースでは、アプリケーション・サーバ・デバイス３８は、悪意のあるアプリケーションまたはエラーのある傾向にあるアプリケーションから、潜在的に、僅かな保護しか提供しない。例えば、アプリケーション・サーバ・デバイス３８の所有者は、アプリケーション・サーバ・デバイス３８に格納されたアプリケーションの何れも、悪意のあるアプリケーションでも、エラーのある傾向にあるアプリケーションでもないことを保証しうる。しかしながら、これは、すべての事例におけるケースではない場合がありうる（例えば、アプリケーション・サーバ・デバイス３８の所有者が、安全かつ適切な動作を保証しない場合がありうる）。または、アプリケーション・サーバ・デバイス３８の所有者からの意図された「保証」が、信頼できないものである場合がありうる。
【００４２】
本開示の技法は、ＧＰＵ１４（例えば、アプリケーション２０）で実行されるべきアプリケーションが、悪意のあるアプリケーションであるのみならず、非効率的なアプリケーション、あるいはエラーのある傾向にあるアプリケーションのような問題のあるアプリケーションであるか否かを、実行前に識別することを支援しうる。例えば、本開示の技法は、ＧＰＵ１４がアプリケーション２０を実行する前に、アプリケーション２０を検証しうる。アプリケーション２０の検証は、アプリケーション２０が、１または複数のパフォー

マンス基準を満足していることを意味しうる。例えば、検証は、アプリケーション２０が、悪意のあるアプリケーションでも、非効率的なアプリケーションでも、エラーのある傾向にあるアプリケーションでもないことを、ある保証レベルで判定することを意味しうる。本開示に記載された技法の例は、ＧＰＵ１４がアプリケーション２０を実行することが安全であるか推奨できないかを示すインジケーションをデバイス１２へ送信しうる。その後、プロセッサ１６は、受信したインジケーションに基づいて、アプリケーション２０を実行するようにＧＰＵ１４に指示することを選択しうる。

【００４３】

例えば、この指示が好ましい、すなわち、このプログラムが、悪意のあるものではなく、非効率的でもなく、および／または、エラーのある傾向にあるものでもないことを示すのであれば、プロセッサ１６は、ＧＰＵ１４に対して、アプリケーション２０を実行することを指示しうる。いくつかの例において、プロセッサ１６は、このインジケーションが好ましくない場合であっても、ＧＰＵ１４に対してアプリケーション２０を実行するように指示しうる。例えば、アプリケーション２０が、悪意のあるものではなく、または、エラーのある傾向にあるものでもないのであれば、プロセッサ１６は、実行がＧＰＵ１４またはデバイス１２を潜在的に害することがないのであれば、ＧＰＵ１４に対して、アプリケーション２０を実行するように指示しうるが、可能な限り効率的に実行できない場合がありうる。

【００４４】

いくつかの例では、本開示の技法は、ＧＰＵ１４において実行されるべき非効率的なアプリケーションを、調整しうるか、または最適化しうる。例えば、アプリケーション２０の開発者は、悪意のある意図を持っていないかもしれないし、アプリケーション２０がエラーのある傾向にはないように、アプリケーション２０を開発したかもしれない。それでもやはり、アプリケーション２０が、ＧＰＵ１４のリソースを効率的に利用できないことがありうる。

【００４５】

一例として、アプリケーション２０の機能のうちの１つは、タスクをワークグループに分割しうる。そして、ＧＰＵ１４の並列処理を活用するために、このワークグループにおいて、並行処理を実行しうる。例えば、アプリケーション２０は、画像をブロックに分割し、これらブロック上で並行処理を実行しうる。ブロックのおのおののサイズは、ＧＰＵ１４において利用可能なローカル・メモリの量に基づきうる。

【００４６】

アプリケーション２０の開発者は、様々な異なるＧＰＵで動作するようにアプリケーション２０を設計したいと思っているかもしれないので、異なるＧＰＵは、異なる量のローカル・メモリを含みうるので、開発者は、例えばＧＰＵ１４のような特定のＧＰＵにおいてどれだけの量のローカル・メモリが利用可能であるのかを事前に知らないのかもしれない。これに対処するために、開発者は、可変サイズのブロックを利用するようにアプリケーション２０を開発しうる。いくつかの事例では、可変サイズのブロックを利用することは、固定サイズのブロックを利用することよりも、非効率的でありうる。本開示の技法は、アプリケーション２０が固定サイズのブロックを利用できるように、ＧＰＵ１４において利用可能なメモリの量に基づいて、アプリケーション２０を調整または最適化しうる。

【００４７】

別の例として、アプリケーション２０は、行列演算を実行しうる。アプリケーション２０の開発者は、行ベースの行列演算または列ベースの行列演算を実行するようにアプリケーション２０を開発したかもしれない。いくつかの事例では、ＧＰＵ１４は、列ベースの行列演算に比べて、行ベースの行列演算を実行することにより適しているかもしれないし、その逆であるかもしれない。この例において、本開示の技法は、アプリケーション２０が列ベースの行列演算を用いるのであれば、ＧＰＵ１４をより効率的に利用するために、行ベースの行列演算を実行するようにアプリケーション２０を修正しうる。

【００４８】

また別の例として、開発者は、より古いバージョンのＧＰＵのためにアプリケーション２０を開発したかもしれず、アプリケーション２０が、ＧＰＵ１４のために最適化されていないかもしれない。本開示の技法は、アプリケーション２０が、例えばＧＰＵ１４のようなより新しいＧＰＵのためにより最適化されるように、アプリケーション２０を修正しうる。その後、ＧＰＵ１４は、より新しいＧＰＵで実行するように最適化されたアプリケーション２０を実行しうる。

【００４９】

本開示の技法によれば、検証サーバ・デバイス２４は、アプリケーション２０を検証し、いくつかの例では、アプリケーション２０を最適化または調整しうる。アプリケーション２０を検証するために、検証サーバ・デバイス２４は、アプリケーション２０が１または複数のパフォーマンス基準を満足するか否かを判定する検証処理を実施しうる。例えば、アプリケーション２０が悪意のあるアプリケーションであるか、エラーのある傾向にあるアプリケーションであるか、あるいは、非効率的なアプリケーションであるかを、あるリーズナブルな保証レベルで判定しうる。アプリケーション２０が、エラーのある傾向にあるアプリケーションまたは非効率的なアプリケーションである場合の例において、検証サーバ・デバイス２４は、アプリケーション２０におけるエラーを修正すること、または、アプリケーション２０がより効率的になるように最適化すること、を試みうる。

【００５０】

アプリケーション２０が問題のあるアプリケーションではないことを絶対的に保証することは一般には困難でありうる。なぜなら、アプリケーション２０がＧＰＵ１４およびデバイス１２に影響を与えうる様々な方式のすべてをテストすることは困難でありうるからである。アプリケーション２０が、問題のあるアプリケーションではないという絶対的な保証は困難かもしれないが、検証サーバ・デバイス２４は、アプリケーション２０が、問題のあるアプリケーションではないことをリーズナブルな確実さで保証するために、異なるタイプの分析を適用しうる。

【００５１】

図１に例示されるように、検証サーバ・デバイス２４は、デバイス１２の外部にある。したがって、「クラウド」におけるアプリケーション２０の検証であると称されうるアプリケーション２０の検証およびアプリケーション２０の最適化は、デバイス１２からオフロードされうる。なぜなら、検証サーバ・デバイス２４は、デバイス１２の外部にあるサーバであるからである。検証サーバ・デバイス２４へのアプリケーション２０の検証をオフロードすることによって、アプリケーション２０が悪意のあるアプリケーションであるか、またはエラーのある傾向にあるアプリケーションであるケースのように、ＧＰＵ１４およびデバイス１２へネガティブなインパクトを与えるアプリケーション２０の可能性が低減されうる。さらに、検証サーバ・デバイス２４へのアプリケーション２０の最適化をオフロードすることによって、省電力および処理効率が実現されうる。なぜなら、プロセッサ１６は、アプリケーション２０を検証または最適化する電力およびクロック・サイクルを消費する必要はないからである。

【００５２】

アプリケーション２０が、アプリケーション２０を検証するための検証サーバ・デバイス２４のために満足する必要がありうるパフォーマンス基準のさまざまな例が存在しうる。一般に、パフォーマンス基準は、静的分析、動的分析、またはこれらの組み合わせの一部でありうる。静的分析は、アプリケーション２０が、静的分析に関連付けられた１または複数のパフォーマンス基準を満足していることを保証するために、アプリケーション２０を実行することなく実行されうるアプリケーション２０の分析を称する。動的分析は、アプリケーション２０が、動的分析に関連付けられた１または複数のパフォーマンス基準を満足することを保証するための、実行中におけるアプリケーション２０の分析を称する。

【００５３】

検証サーバ・デバイス２４は、静的分析、動的分析、または、静的分析と動的分析との

両方を実行するように動作可能でありうる。例示の目的のために、検証サーバ・デバイス２４は、静的分析と動的分析との両方を実行するように動作可能なものとして記載されているので、アプリケーション２０が、静的分析と動的分析との両方に関連付けられたパフォーマンス基準を満足することを保証するように動作可能である。代替例では、検証サーバ・デバイス２４は、静的分析または動的分析のうちの１つを実行するように動作可能であり、これら代替例では、検証サーバ・デバイス２４は、検証サーバ・デバイス２４が実行するように動作可能な分析のタイプに関連付けられたパフォーマンス基準（例えば、静的分析または動的分析に関連付けられたパフォーマンス基準）を、アプリケーション２０が満足することを保証するように動作可能でありうる。

【００５４】

図１に例示されるように、検証サーバ・デバイス２４は、エミュレータ・ユニット２６およびサーバ・メモリ２８を含んでいる。サーバ・メモリ２８は、１または複数のＧＰＵモデル３０、１または複数のＧＰＵ入力３２、および１または複数のデバイス・モデル３４を定義するデータおよび／または命令群を含みうる。エミュレータ・ユニット２６は、ＧＰＵモデル３０およびデバイス・モデル３４のうちの１または複数を実行するように動作可能な処理ユニットでありうる。別の例として、エミュレータ・ユニット２６は、ＧＰＵでありうるハードウェア・エミュレーション・ボードでありうる。いくつかの例において、エミュレータ・ユニット２６は、同じ回路、または、分離された別個の回路の一部でありうる２つの部分を含みうる。ここで、第１の部分は、デバイス・モデル３４およびＧＰＵモデル３０のうちの１または複数を実行するように動作可能な処理ユニットであり、第２の部分は、ハードウェア・エミュレーション・ボード（例えば、ＧＰＵ）である。エミュレータ・ユニット２６の例は、限定される訳ではないが、ＤＳＰ、汎用マイクロプロセッサ、ＡＳＩＣ、ＦＰＧＡ、あるいは、その他の等価な統合された論理回路または個別の論理回路を含む。

【００５５】

サーバ・メモリ２８は、デバイス・メモリ１８に類似しうる。例えば、サーバ・メモリ１８は、任意の媒体でありうる。これは、命令群、データ、および／または、データ構造の形態で所望のプログラム・コードを格納するために使用され、エミュレータ・ユニット２６によってアクセスされ、エミュレータ・ユニット２６に対して、エミュレータ・ユニット２６に割り当てられた機能の１または複数を実行させる。デバイス・メモリ１８に類似して、サーバ・メモリ２８は、いくつかの例において、デバイス・メモリ１８に関して前述されたように、非一時的な記憶媒体として考慮されうる。

【００５６】

例示されるように、サーバ・メモリ２８は、１または複数のＧＰＵモデル３０、ＧＰＵ入力３２、およびデバイス・モデル３４を定義する命令群および／またはデータを格納しうる。すべての例において、サーバ・メモリ２８が、１または複数のＧＰＵモデル３０、ＧＰＵ入力３２、およびデバイス・モデル３４を格納する必要はないかもしれない。例えば、サーバ・メモリ２８は、ＧＰＵモデル３０およびＧＰＵ入力３２を格納しうるが、デバイス・モデル３４を格納しないかもしれない。検証サーバ・デバイス２４が、静的分析のみを実行するように動作可能であれば、ＧＰＵモデル３０、ＧＰＵ入力３２、およびデバイス・モデル３４は必要とされないかもしれない。いくつかの例において、エミュレータ・ユニット２６が動的分析を実行するのは、ＧＰＵモデル３０、ＧＰＵの入力３２、およびデバイス・モデル３４を用いてである。

【００５７】

１または複数のＧＰＵモデル３０のおのおのは、特定のＧＰＵタイプに対応し、１または複数のデバイス・モデル３４のおのおのは、特定のデバイス・タイプに対応しうる。例えば、ＧＰＵモデル３０のおのおのは、並列処理機能、ローカル・メモリ利用可能性、および、そのＧＰＵタイプのＧＰＵの機能を定義するその他任意の適切な特性の観点から、対応するＧＰＵの構成をモデル化しうる。デバイス・モデル３４のおのおのは、メモリ構成、プロセッサ速度、システム・バス速度、デバイス・メモリ、および、そのデバイス・

タイプのデバイスの機能を定義するその他任意の適切な特性の観点から、対応するデバイス・タイプの構成をモデル化しうる。例えば、異なるベンダが、異なる機能特性を持つ異なるタイプのデバイスを提供し、デバイス・モデル３４が、これら異なるデバイス・タイプのおのおののためのモデルでありうる。

【００５８】

１または複数のＧＰＵモデル３０およびデバイス・モデル３４はおのおの、エミュレータ・ユニット２６が実行できうる仮想モデル・ソフトウェアとして考慮されうる。例えば、エミュレータ・ユニット２６が、ＧＰＵモデル３０のうちの１つを実行する場合、エミュレータ・ユニット２６は、実行されたＧＰＵモデル３０が対応するＧＰＵをエミュレートする。エミュレータ・ユニット２６は、ＧＰＵモデル３０のうちの１つ、および、デバイス・モデル３４のうちの１つを実行した場合、ＧＰＵに含まれるこのようなデバイスが、実行されたＧＰＵモデル３０が対応するＧＰＵに含まれているかのように、実行されたデバイス・モデル３４が対応するデバイスをエミュレートする。いくつかの例において、ＧＰＵベンダおよびデバイス・ベンダは、ＧＰＵモデル３０およびデバイス・モデル３４をそれぞれ提供しうる。サーバ・メモリ２８がＧＰＵモデル３０およびデバイス・モデル３４を格納するその他の手法が存在しうる。そして、本開示の態様は、ベンダがＧＰＵモデル３０およびデバイス・モデル３４を提供する特定の例に限定されない。

【００５９】

例えば、エミュレータ・ユニット２６が、ＧＰＵモデル３０のうちの１つを実行した場合、エミュレータ・ユニット２６は、（２つの例として、）エミュレータ・ユニット２６のローカル・メモリ利用可能性および並列処理能力が、ＧＰＵモデル３０のうちの実行された１つに関連付けられたＧＰＵタイプに機能的に等価であるかのように機能しうる。同様に、エミュレータ・ユニット２６が、デバイス・モデル３４のうちの１つを実行した場合、エミュレータ・ユニット２６は、（４つの例として、）エミュレータ・ユニット２６のデバイス・メモリ、システム・バス速度、プロセッサ速度、およびメモリ構成が、デバイス・モデル３４のうちの実行された１つに関連付けられたデバイス・タイプに機能的に等価であるかのように機能しうる。言い換えれば、ＧＰＵモデル３０のうちの１つを実行することによって、エミュレータ・ユニット２６に対して、ＧＰＵモデル３０のうちの実行された１つに関連付けられたＧＰＵとして機能させる。ＧＰＵモデル３０のうちの１つと、デバイス・モデル３４のうちの１つとを実行することによって、エミュレータ・ユニット２６に対して、ＧＰＵモデル３０のうちの実行された１つに関連付けられたＧＰＵを含む、デバイス・モデル３４のうちの実行された１つに関連付けられたデバイスとして機能させる。

【００６０】

複数のＧＰＵモデル３０のうちの１つが、一般的なＧＰＵモデル３０でありうる。また、複数のデバイス・モデル３４のうちの１つが、一般的なデバイス・モデル３４でありうる。いくつかの例において、サーバ・メモリ２８は、複数のＧＰＵモデルおよびデバイス・モデルの代わりに、一般的なＧＰＵモデルおよび一般的なデバイス・モデルを格納しうる。一般的なＧＰＵモデルおよびデバイス・モデルは、特定のＧＰＵまたはデバイス・タイプに相当しないかもしれないが、静的分析および動的分析のために適しうる。いくつかの例において、サーバ・メモリ２８は、ＧＰＵ１４に相当するＧＰＵモデルを格納していないのであれば、一般的なＧＰＵモデルは、検証目的のために適切でありうる。一般的なＧＰＵモデルおよび一般的なデバイス・モデルは、ほとんどのＧＰＵまたはデバイスに共通する動作のベース・プロファイルに一致しうる。

【００６１】

一般的なＧＰＵモデルおよび一般的なデバイス・モデルによってモデル化されうるさまざまなタイプのＧＰＵおよびデバイスが存在しうる。一例として、一般的なＧＰＵモデルは、他のＧＰＵに比べて平均的な並列処理能力およびローカル・メモリ利用可能性を持つＧＰＵをモデル化しうる。一般的なデバイス・モデルは、他のデバイスに比べて平均的なメモリ構成、プロセッサ速度、システム・バス速度、およびデバイス・メモリを持つデバ

イスをモデル化しうる。

【００６２】

ＧＰＵ１４における実行のために、アプリケーション２０を検証および／または最適化するための例示的な例として、デバイス１２は、アプリケーション・サーバ・デバイス３８から、アプリケーション２０をダウンロードしうる。前述したように、アプリケーション２０は、ソース・コード、中間表現、または、プレ・コンパイルされたオブジェクト・コードでありうる。その後、プロセッサ１６は、デバイス１２にアプリケーション２０をインストールしうる。アプリケーション２０が、例えば、プレ・コンパイルされたオブジェクト・コードではなく、ソース・コードまたは中間表現にあるのであれば、インストールの一部は、アプリケーション２０のコードをコンパイルするコンパイラを実行しているプロセッサ１６でありうる。

【００６３】

いくつかの例において、コンパイルする前に、アプリケーション２０のダウンロードされたコードが、ソース・コードまたは中間表現である場合、プロセッサ１６は、デバイス１２に対して、検証のための検証サーバ・デバイス２４へ、アプリケーション２０のダウンロードされたコードを送信させうる。いくつかの例では、アプリケーション２０のダウンロードされたコードが、プレ・コンパイルされたオブジェクト・コードである場合、プロセッサ１６は、ＧＰＵに対してアプリケーション２０を実行することを許可する前に、デバイス１２に対して、プレ・コンパイルされたオブジェクト・コードを、検証サーバ・デバイス２４へ送信させうる。

【００６４】

セキュリティ目的のために、プロセッサ１６は、デバイス１２が検証サーバ・デバイス２４へ送信するアプリケーション２０のダウンロードされたコードを暗号化しうるか、もしくはセキュアにしうる。いくつかの例において、プロセッサ１６は、アプリケーション２０のダウンロードされたコードを、検証サーバ・デバイス２４へ送信する前に、ユーザからの許可を必要としうる。さらに、動的分析のいくつかの例では、プロセッサ１６が、デバイス１２に対して、ＧＰＵ１４のＧＰＵタイプを、または、ＧＰＵ１４のＧＰＵタイプとデバイス１２のデバイス・タイプとの両方を、検証サーバ・デバイス２４へ送信させうる。これらの事例のうちのいくつかでは、プロセッサ１６は、ＧＰＵ１４のＧＰＵタイプ、または、ＧＰＵ１４のＧＰＵタイプとデバイス１２のデバイス・タイプを、検証サーバ・デバイス２４へ送信する前に、ユーザからの許可を必要としうる。

【００６５】

エミュレータ・ユニット２６は、アプリケーション２０が、静的分析に関連付けられたパフォーマンス基準を満足するか否かを判定するために、アプリケーション２０について静的分析を実行するように動作可能でありうる。例えば、エミュレータ・ユニット２６は、アプリケーション２０を実行することなく、アプリケーション２０を分析しうる。一例として、エミュレータ・ユニット２６は、ウィルスのためのコードであると知られているコードを識別するために、アプリケーション２０のダウンロードされたコードを解析しうる。例えば、サーバ・メモリ２８は、既知のウィルスのコードを格納しうる。そして、エミュレータ・ユニット２６は、アプリケーション２０のダウンロードされたコードを、既知のウィルスのコードと比較しうる。アプリケーション２０のダウロードされたコードが、既知のウィルスのコードを含んでいないと判定することは、アプリケーション２０を検証するために満足される必要のあるパフォーマンス基準の一例でありうる。

【００６６】

静的分析の一部として、コンパイル中におけるアプリケーション２０のエラーを識別するために、アプリケーション２０のダウンロードされたコードが、アプリケーション２０のソース・コードまたは中間表現である例では、エミュレータ・ユニット２６が、アプリケーション２０のダウンロードされたコードをコンパイルしうる。例えば、エミュレータ・ユニット２６は、エミュレータ・ユニット２６内の破線によって示されるように、コンパイラ３６を実行しうる。コンパイラ３６を用いたアプリケーション２０のコンパイルは

、アプリケーション２０内のあらゆる無限ループ、または、アプリケーション２０内のメモリ・アレイに対するアウト・オブ・バウンズ・アクセスを識別しうる。この例において、コンパイル中に発見されうるエラーがアプリケーション２０に無いと判定することは、アプリケーション２０を検証するために満足される必要のあるパフォーマンス基準の別の例でありうる。

【００６７】

静的分析は、発見されうる悪意のあるコード、非効率性、およびエラーのタイプに限定されうる。例えば、アプリケーション２０のダウンロードされたコードが、プレ・コンパイルされたオブジェクト・コードであれば、エミュレータ・ユニット２６が、コンパイル中に、アプリケーション２０のエラーを識別することは可能ではないかもしれない。なぜなら、アプリケーション２０のコードは既に、プレ・コンパイルされたオブジェクト・コードであるからである。別の例として、アプリケーション２０が、格納のためポインタに依存しているのであれば、アプリケーション２０において何れかのアウト・オブ・バウンズ・メモリ・アクセス・エラーがあるか否かを、アプリケーション２０をコンパイルすることに単純に基づいて判定することは可能ではないかもしれない。

【００６８】

アプリケーション２０が問題のあるものであるか（例えば、非効率的であるか、エラーのある傾向にあるか、または悪意があるか）をさらに判定するために、エミュレータ・ユニット２６は、動的分析を実行しうる。前述したように、動的分析は、実行中におけるアプリケーション２０の分析を称する。いくつかの例では、動的分析を実行するために、エミュレータ・ユニット２６は、エミュレータ・ユニット２６自身を、あたかもＧＰＵ１４であるかのように見せうる。例えば、いくつかの事例では、アプリケーション２０のダウンロードされたコードを送信することに加えて、プロセッサ１６が、デバイス１２に対して、ＧＰＵ１４のＧＰＵタイプを、検証サーバ・デバイス２４のエミュレータ・ユニット２６へ送信させうるか、または、ＧＰＵ１４のＧＰＵタイプとデバイス１２のデバイス・タイプとの両方を、ネットワーク２２を介して検証サーバ・デバイス２４のエミュレータ・ユニット２６へ送信させうる。一方、エミュレータ・ユニット２６は、ＧＰＵモデル３０のうちの何れが、ＧＰＵ１４のＧＰＵタイプに対応しているのかを識別しうる。そして、検証サーバ・デバイス２４におけるＧＰＵ１４をエミュレートするために、ＧＰＵモデル３０のうちの１つを実行させうる。エミュレータ・ユニット２６がさらにデバイス・タイプを受信する例では、エミュレータ・ユニット２６は、デバイス・モデル３４のうちの何れが、デバイス１２のデバイス・タイプに対応するのかを識別しうる。そして、検証サーバ・デバイス２４においてデバイス１２をエミュレートするために、デバイス・モデル３４のうちのそれを実行しうる。

【００６９】

デバイス１２が、ＧＰＵ１４のＧＰＵタイプ、および／または、デバイス１２のデバイス・タイプを送信しない例では、エミュレータ・ユニット２６は、一般的なＧＰＵモデル、および／または、一般的なデバイス・モデルを実行しうる。あるいは、デバイス１２が、ＧＰＵ１４のＧＰＵタイプ、および／または、デバイス１２のデバイス・タイプを送信するが、ＧＰＵモデル３０およびデバイス・モデル３４のうちの何れも、ＧＰＵタイプおよびデバイス・タイプに対応していないのであれば、エミュレータ・ユニット２６は、一般的なＧＰＵモデル、および／または、一般的なデバイス・モデルを実行しうる。エミュレータ・ユニット２６が、ハードウェア・エミュレーション・ボードであるか、または、ハードウェア・エミュレーション・ボードを含んでいる例では、このようなハードウェア・エミュレーション・ボードは、少なくとも部分的に、一般的なデバイスにおける一般的なＧＰＵとして機能するように設計されうる。

【００７０】

エミュレータ・ユニット２６が、エミュレータ・ユニット２６自身をＧＰＵ１４になるように、または、デバイス１２の一部としてＧＰＵ１４となるように、エミュレートすると、エミュレータ・ユニット２６は、アプリケーション２０を実行しうる。例えば、エミ

ュレータ・ユニット２６は、アプリケーション２０のソース・コードまたは中間コードを受信したのであれば、コンパイラ３６によってソース・コードをコンパイルし、結果として得られるオブジェクト・コードを実行しうる。エミュレータ・ユニット２６は、アプリケーション２０の、プレ・コンパイルされたオブジェクト・コードを受信すると、アプリケーション２０の、プレ・コンパイルされたオブジェクト・コードを実行しうる。

【００７１】

本開示の技法は、いくつかの例において、少なくとも部分的に、（例えば、ＧＰＵモデル３０のうちの１つのような）ＧＰＵ１４のタイプに基づいて、エミュレータ・ユニット２６が仮想モデルを実行することによって実行されるものと考えられうる。その後、エミュレータ・ユニット２６がアプリケーション２０を実行した場合、アプリケーション２０は、（例えば、エミュレータ・ユニット２６で実行しているＧＰＵモデル３０のうちの１つのような）仮想モデルにおいて実行していると考えられうる。例えば、ＧＰＵ１４に対応するＧＰＵモデル３０のＧＰＵモデルと、アプリケーション２０との両方が、エミュレータ・ユニット２６において動作する。本開示の技法では、ＧＰＵ１４に対応するＧＰＵモデルの実行によって、エミュレータ・ユニット２６は、あたかもＧＰＵ１４であるかのように機能するので、エミュレータ・ユニット２６がアプリケーション２０を実行する場合、アプリケーション２０は、ＧＰＵ１４に対応するＧＰＵモデルにおいて動作しうる。

【００７２】

動的分析の一部として、エミュレータ・ユニット２６は、エミュレータ・ユニット２６において動作しているアプリケーション２０の仮入力値を受信しうる。例示されるように、サーバ・メモリ２８は、１または複数のＧＰＵ入力３２を格納しうる。これら１または複数のＧＰＵ入力３２は、異なるグラフィック画像またはオブジェクトのための値でありうる。いくつかの例において、これらの異なる画像のおのおのは、異なるサイズからなりうる。アプリケーション２０が、グラフィック処理に関連していない例では、ＧＰＵ入力３２は、非グラフィック入力でありうる。エミュレータ・ユニット２６が、可能な入力値のすべての置換および組み合わせをテストすることを保証することは困難でありうる。したがって、サーバ・メモリ２８は、アプリケーション２０が悪意のあるものでも、高くエラーのある傾向にあるアプリケーション（例えば、問題のあるアプリケーション）でもないというリーズナブルなレベルの保証を与えるために、例えばサンプルまたはテスト入力のような、十分な数および／または範囲のＧＰＵ入力３２を格納しうる。ＧＰＵ入力３２は、ＧＰＵ１４によって処理およびレンダリングされるべき、異なるタイプの画像またはオブジェクトを含みうる。

【００７３】

アプリケーション２０の実行中、エミュレータ・ユニット２６は、ＧＰＵ入力３２の値を入力しうる。そして、ＧＰＵモデル３０の、実行されたＧＰＵモデルの機能を分析しうる。エミュレータ・ユニット２６が、ハードウェア・エミュレーション・ボードである例において、エミュレータ・ユニット２６は、ハードウェア・エミュレーション・ボードの機能を分析しうる。例えば、エミュレータ・ユニット２６は、ＧＰＵモデル３０の、実行されたＧＰＵモデルによるメモリ・アクセスをモニタしうる。この例において、エミュレータ・ユニット２６は、ＧＰＵモデル３０の、実行されたＧＰＵモデルによるメモリ・アクセスの何れかが、サーバ・メモリ２８のアウト・オブ・バウンズ・メモリ・アクセスであるか否かを判定しうる。別の例として、エミュレータ・ユニット２６は、ＧＰＵモデル３０の実行されたＧＰＵモデルが、サーバ・メモリ２８に情報を書き込んでいる場合、メモリ・アドレスをモニタしうる。ＧＰＵモデルが情報を書き込んでいる場合におけるメモリ・アドレスとＧＰＵモデルのメモリ・アクセスに基づいて、エミュレータ・ユニット２６は、アプリケーション２０が、エラーのある傾向にあるか否かを判定することができうる。このようなメモリ・トラッキングは、アプリケーション２０がポインタを用いて変数を読み取ったり、または、変数に書き込んでいる場合には特に有用でありうる。

【００７４】

例えば、実行されたＧＰＵモデルが、アウト・オブ・バウンズのメモリ位置に情報を書

き込んでいるか、または、アウト・オブ・バウンズのメモリ位置から情報を読み取っているのであれば、エミュレータ・ユニット２６は、アプリケーション２０が、エラーのある傾向にあり、恐らくは悪意のあるものであると判定しうる。例えば、実行されたＧＰＵモデルが、存在しないメモリ位置に情報を書き込んだり、または、存在しないメモリ位置から情報を読み取っているのであれば、エミュレータ・ユニット２６は、アプリケーション２０が、エラーのある傾向にあると判定しうる。実行されたＧＰＵモデルが、ＧＰＵモデルのために確保されていないメモリ位置に情報を書き込んでいるのであれば、エミュレータ・ユニット２６は、アプリケーション２０がエラーのある傾向にあり、恐らくは悪意のあるものであると判定しうる。例えば、エミュレータ・ユニット２６は、アプリケーション２０がアクセスすることができないメモリ位置にウィルスをロードすることをアプリケーション２０が試みていると判定しうる。

【００７５】

アプリケーション２０が、実行中に（例えばアクセスから）情報を読み取ったり、（例えばアクセスへ）情報を書き込む場合における制限は、動的分析に関連付けられたパフォーマンス基準の例でありうる。例えば、パフォーマンス基準は、アプリケーション２０がアクセスすることを許可されているメモリ位置の制限でありうる。アプリケーション２０の実行により、制限されたメモリ位置以外のメモリ位置に、ＧＰＵモデル３０のＧＰＵモデルがアクセスするのであれば、アプリケーション２０は、パフォーマンス基準を破っている場合がありうる。例えば、パフォーマンス基準にしたがって、許可されている制限されたメモリ位置以外のアクセスのしきい数が存在しうる。このしきい数は、アプリケーション２０が、制限されたメモリ・アクセス以外のメモリ位置へのアクセスを試みていないとの高いレベルの保証を与えるために、ゼロでありうる。

【００７６】

また、エミュレータ・ユニット２６がデバイス・モデル３４のうちの１つを実行する例では、エミュレータ・ユニット２６は同様に、デバイス・モデル３４の実行されたデバイス・モデルの機能を分析しうる。例えば、エミュレータ・ユニット２６は、エミュレータ・ユニット２６がＧＰＵモデル３０のうちの１つを実行している間、デバイス・モデル３４のうちの実行された１つによって実行された機能をモニタしうる。例えば、デバイス・モデル３４のうちの１つの実行により、バス・システムを含むエミュレータ・ユニット２６デバイス１２となりうる。エミュレータ・ユニット２６は、アプリケーション２０の実行によって、システム・バスに対して、オーバロードを引き起こし、その結果、デバイス１２がスロー・ダウンしているか否かを判定しうる。

【００７７】

システム・バスがオーバロードしているか否かを判定するためにシステム・バスをモニタすることは、動的分析に関連付けられたパフォーマンス基準の例でありうる。例えば、アプリケーション２０の実行が、システム・バスに対してオーバロードを引き起こされるのであれば、アプリケーション２０は、パフォーマンス基準を破っている場合がありうる。この例において、パフォーマンス基準は、システム・バスに対するあるレベルのオーバロードを許容しうる。なぜなら、システム・バスの如何なるオーバロードをも許さないことは可能ではないことがありうるからである。例えば、パフォーマンス基準は、システム・バス・オーバロードのパーセンテージ量しきい値を確立しうる。システム・バス・オーバロードが、許容可能なパーセンテージ未満であれば、パフォーマンス基準が満足される。そうではない場合には、パフォーマンス基準は満足されない。

【００７８】

エミュレータ・ユニット２６は、同様に、例えば、サービス強制停止攻撃のような悪意のあるアプリケーションを検出しうる。例えば、エミュレータ・ユニット２６は、ＧＰＵモデル３０のＧＰＵモデルがアプリケーション２０を実行することができるレートをモニタしうる。エミュレータ・ユニット２６は、遅い応答、意図しない終了、またはハンギングを検出すると、アプリケーション２０が、サービス強制停止攻撃のために設計されたアプリケーションであるか、または非常に貧弱に設計されたアプリケーションであると判定

しうる。この例において、パフォーマンス基準は、アプリケーション２０の特定のタスクのためのしきい実行時間または実行レートでありうる。アプリケーション２０が、特定のタスクを完了するためにしきい実行時間よりも長い時間を要するか、あるいは、しきい実行レート未満のレートでタスクを実行しているのであれば、アプリケーション２０は、パフォーマンス基準を破っている場合がありうる。

【００７９】

悪意のあるアプリケーションまたはエラーのある傾向にあるアプリケーションをエミュレータ・ユニット２６が検出する別の例として、エミュレータ・ユニット２６は、アプリケーション２０によって発行された指示をモニタしうる。例えば、いくつかの例では、アプリケーション２０によって発行された指示は、９６ビットのワードでありうる。しかしながら、９６ビットからなるすべての組み合わせが、必ずしも、有効な指示を示している訳ではない。いくつかの例において、ＧＰＵ１４は、無効の指示を無視するように設計されうる。しかしながら、これは、ＧＰＵ１４のすべての例のケースであるとは限らないかもしれない。ＧＰＵ１４が、不要に無効な指示を実行することを回避するために、エミュレータ・ユニット２６は、実行中にアプリケーション２０によって発行された指示が、有効な指示であるか、無効な指示であるかを判定しうる。アプリケーション２０が無効な指示を発行しているとエミュレータ・ユニット２６が判定すると、エミュレータ・ユニット２６は、アプリケーション２０が、悪意のあるアプリケーション、エラーのある傾向にあるアプリケーション、または、非効率的なアプリケーションであると判定しうる。

【００８０】

別の例として、実行中に、アプリケーション２０は、レジスタへのデータの書き込みを行いうるか、レジスタからデータの読み取りを行いうる。悪意のあるアプリケーション、エラーのある傾向のあるアプリケーション、または非効率的なアプリケーションは、書き込まれていないレジスタからデータを読み取りうる。アプリケーション２０が、以前に書き込まれていないデータをレジスタから読み取ることを試みると、アプリケーション２０によって読み取られたデータは、意味のないデータ（すなわち、初期化されていないデータ）となりうる。このような初期化されていないデータの読み取りの結果、予測不能の挙動をもたらしうる。いくつかの例では、エミュレータ・ユニット２６が、アプリケーション２０が実行中にどのレジスタに書き込むのかをモニタし、アプリケーション２０が、以前に書き込まれていないレジスタから読み取りをしているか否かを判定しうる。書き込まれていないレジスタからアプリケーション２０が読み取りをしているとエミュレータ・ユニット２６が判定すると、エミュレータ・ユニット２６は、アプリケーション２０が、悪意のあるアプリケーション、エラーのある傾向にあるアプリケーション、または非効率的なアプリケーションであると判定しうる。

【００８１】

静的分析および動的分析に関連付けられたパフォーマンス基準が満足されているとエミュレータ・ユニット２６が判定すると、検証サーバ・デバイス２４は、アプリケーション２０が、静的分析、動的分析、または静的分析と動的分析との両方に関連付けられた１または複数のパフォーマンス基準を、ある保証レベルで満足する（例えば、アプリケーション２０を検証する）ことを示すインジケーションをデバイス１２へ送信しうる。このケースでは、検証サーバ・デバイス２４は、アプリケーション２０が、ＧＰＵ１４による使用のために検証されたことを示すインジケーションを提供しうる。そうではない場合、いくつかの例において、検証サーバ・デバイス２４は、アプリケーション２０がＧＰＵ１４による使用のために検証されておらず、これによって、ＧＰＵ１４がアプリケーション２０を実行することは推奨されないことを示すインジケーションをデバイス１２へ送信しうる。これに応じて、プロセッサ１６は、受信したインジケーションに基づいて、ＧＰＵ１４に対して、アプリケーション２０を実行するように指示しうる。

【００８２】

検証サーバ・デバイス２４が、アプリケーション２０のソース・コードまたは中間コードを受信した例では、エミュレータ・ユニット２６はまた、コンパイラ３６によってコン

パイルされたように、アプリケーション２０のコンパイルされたオブジェクト・コードを送信しうる。このように、アプリケーション２０のコンパイルも、デバイス１２からオフロードされ、例えば検証サーバ・デバイス２４のような外部デバイスへオフロードされうる。

【００８３】

また、検証サーバ・デバイス２４は、アプリケーション２０を最適化または調整することを課せられうる。例えば、エミュレータ・ユニット２６は、アプリケーション２０のソース・コードまたは中間コードを受信しうる。静的分析および／または動的分析の一部として、エミュレータ・ユニット２６は、アプリケーション２０が幾分エラーのある傾向にあるか、または、ＧＰＵ１４の機能を非効率的に利用するであろうと判定しうる。これらの例では、アプリケーション２０がＧＰＵ１４において非効率的に、または、エラーを伴って動作する場合、エミュレータ・ユニット２６は、ＧＰＵ１４がアプリケーション２０を実行することは推奨されないことを示すインジケーションをデバイス１２へ送信するのではなく、アプリケーション２０のエラーの修正を試みるか、または、ＧＰＵ１４のためにアプリケーション２０を調整することを試みうる。

【００８４】

エミュレータ・ユニット２６が、エラーを修正することができるか、または、アプリケーション２０をより効率的にすることができるのであれば、エミュレータ・ユニット２６は、ＧＰＵ１４が実行すべきオブジェクト・コードを生成するために、修正されたアプリケーション２０のコードをコンパイルしうる。エミュレータ・ユニット２６は、その後、結果的に得られたオブジェクト・コードを、ＧＰＵ１４が、このオブジェクト・コードを実行すべきであることを示すインジケーションとともにデバイス１２へ送信しうる。このケースでは、ＧＰＵ１４は、アプリケーション２０のオリジナルのコードから生成されたオブジェクト・コードではなく、修正されたコードから生成されたオブジェクト・コードを実行しうる。あるいは、エミュレータ・ユニット２６は、コンパイルすることなく、アプリケーション２０の修正されたコードを送信しうる。

【００８５】

これらの例のうちの何れかでは、アプリケーション２０の検証は、修正されたアプリケーション２０のコードの送信（例えば、修正されたコード、または結果として得られたオブジェクト・コードの送信）の一部であると考えらえうる。例えば、デバイス１２が、検証サーバ・デバイス２４から、修正されたアプリケーション２０のコードを受信した場合、デバイス１２は、修正されたアプリケーション２０のコードは、実行に適していることを自動的に判定しうる。なぜなら、デバイス１２は、検証サーバ・デバイス２４から、修正されたアプリケーション２０のコードを受信しているからである。この意味において、デバイス１２が検証サーバ・デバイス２４から受信した検証は、明示的な検証または暗黙的な検証でありうる。明示的な検証または暗黙的な検証の何れのケースであれ、エミュレータ・ユニット２６は、アプリケーション２０、または、修正されたアプリケーション２０のバージョンが、１または複数のパフォーマンス基準を満足することを、ある保証レベルで判定しうる。

【００８６】

エミュレータ・ユニット２６は、アプリケーション２０のエラーを修正することができないのであれば、ＧＰＵ１４においてアプリケーション２０を実行することは推奨されないことを示すインジケーションを送信しうる。エミュレータ・ユニット２６は、アプリケーション２０をより効率的にすることができないのであれば、ＧＰＵ１４がアプリケーション２０を実行することが適していることを示すインジケーションをデバイス１２へ未だに送信しうる。なぜなら、アプリケーション２０は、完全に効率的ではないものの、エラーのある傾向にあるものでも、悪意のあるものでもないからである。

【００８７】

アプリケーション２０を調整または最適化するために、エミュレータ・ユニット２６は、アプリケーション２０のコード（例えば、ソース・コードまたは中間コード）の挿入、

リプレース、または修正を、ある別の手法で行いうる。いくつかの例において、エミュレータ・ユニット２６は、コンパイルされたアプリケーション２０のコードがどれくらい良好に動作するのかを判定するために、統計を収集しうる。例えば、アプリケーション２０は、変数値をアレイに格納するために、アレイ・インデクスを利用しうる。エミュレータ・ユニット２６は、アプリケーション２０によって利用されているアレイ・インデクスが、範囲内にあることをチェックするコードを、アプリケーション２０のソース・コードに追加しうる。エミュレータ・ユニット２６は、アレイ・インデクスが範囲内に無い場合に、アプリケーション２０を停止させるコードを、アプリケーション２０のソース・コードに追加しうる。エミュレータ・ユニット２６は、その後、ＧＰＵ１４によるアプリケーション２０の実行のためのオブジェクト・コードを生成するために、修正されたソース・コードをコンパイルしうる。

【００８８】

最適化または調整は、例えばアプリケーション２０のようなアプリケーションが、一般に、ＧＰＵ１４の高いレベルの並列処理を活用するように開発されているという仮定に基づきうる。開発者は、ＧＰＵ１４の並列処理を活用することを意図していないのであれば、ＧＰＵ１４で動作するのではなく、むしろプロセッサ１６で動作するように、アプリケーション２０を開発するであろう。

【００８９】

例えば、アプリケーション２０の開発者は、画像のブロックにおける画像処理を並列的に実行するようにアプリケーション２０を開発したかもしれない。前述したように、画像のブロックのサイズは、ＧＰＵ１４において利用可能なローカル・メモリの量に基づきうる。開発者は、ＧＰＵ１４においてどれくらい資金が利用可能であるのかを知らないかもしれないので、より効率的な固定サイズのブロックではなく、可変サイズのブロックを用いるようにアプリケーション２０を開発しうる。例えば、ブロックのサイズは、実行中に変わらないので、固定サイズのブロックがより効率的でありうる。

【００９０】

いくつかの例では、ＧＰＵ１４に対応するＧＰＵモデル３０のＧＰＵモデルは、ＧＰＵ１４のローカル・メモリのサイズを示す情報を含みうるので、エミュレータ・ユニット２６は、ブロックの最適なサイズを決定しうる。この例において、エミュレータ・ユニット２６は、ＧＰＵ１４において利用可能なローカル・メモリの量、ＧＰＵ１４のローカル・メモリに書き込む、または、ＧＰＵ１４のローカル・メモリから読み取る必要のあるデータの量、および、アプリケーション２０の開発者に利用可能ではないかもしれないその他の情報に基づいて、ブロックの最適なサイズを選択しうる。この開示の態様では、エミュレータ・ユニット２６は、ＧＰＵ１４に対応するＧＰＵモデル３０のＧＰＵモデルでアプリケーション２０を実行しうるので、どれくらいの量のローカル・メモリが利用可能であり、どれくらいの量のデータがローカル・メモリに書き込まれる必要があり、どれくらいの量のデータがローカル・メモリから読み取られる必要があるのかを知るであろう。

【００９１】

これらの例では、エミュレータ・ユニット２６は、ブロック・サイズを、最適に決定されたサイズに固定するために、アプリケーション２０のソース・コードまたは中間コードを更新するか、あるいは、そうではない場合には、修正しうる。言い換えれば、エミュレータ・ユニット２６は、ＧＰＵ１４の並列処理を最良に利用するために、ブロックの最適なサイズを決定しうる。エミュレータ・ユニット２６は、その後、この修正されたアプリケーション２０のコードをコンパイルし、結果的に得られたオブジェクト・コードを、ＧＰＵ１４における実行のためにデバイス１２へ送信しうる。このように、修正されたアプリケーション２０をＧＰＵ１４が実行する場合、修正されたアプリケーション２０は、オリジナルのアプリケーション２０に比べて、ＧＰＵ１４においてより効率的に動作しうる。

【００９２】

最適化のための別の例では、前述したように、アプリケーション２０は、行列演算を実

行しうる。この例では、エミュレータ・ユニット２６は、列ベースの行列演算または行ベースの行列演算の何れが、ＧＰＵ１４によってより容易に取り扱われるのかを判定しうる。例えば、エミュレータ・ユニット２６は、ＧＰＵ１４に対応するＧＰＵモデル３０のＧＰＵモデルに対して、行ベースの行列演算を用いて、および列ベースの行列演算を用いて、アプリケーション２０を実行させうる。エミュレータ・ユニット２６は、列ベースの行列演算と、行ベースの行列演算との効率（例えば、メモリへのアクセス数、処理時間の長さ、およびその他のこのような効率尺度）を比較しうる。エミュレータ・ユニット２６は、測定された効率に基づいて、アプリケーション２０のコードを修正しうる。例えば、列ベースの演算が、行ベースの演算よりもより効率的に実行されるのであれば、エミュレータ・ユニット２６は、行列演算が、列ベースの演算として実行されるように、アプリケーション２０のコードを修正しうる。同様に、行ベースの演算が、列ベースの演算よりもより効率的に実行されるのであれば、エミュレータ・ユニット２６は、行列演算が、行ベースの演算として実行されるように、アプリケーション２０のコードを修正しうる。

【００９３】

最適化の別の例では、前述したように、アプリケーション２０の開発者は、より古いバージョンのＧＰＵで動作するようにアプリケーション２０を開発したかもしれない。このケースでは、アプリケーション２０は、例えばＧＰＵ１４のようなＧＰＵで適切に動作しうる。しかしながら、アプリケーション２０は、ＧＰＵ１４の機能を十分に活用していないことがありうる。例えば、アプリケーション２０は、ＧＰＵ１４が並列的に処理すべきグラフィック・データまたは非グラフィック・データの量を不必要に制限しうる。なぜなら、より古いバージョンのＧＰＵは、処理能力が制限されていることがありうるからである。この例において、エミュレータ・ユニット２６は、アプリケーション２０が実行された場合に、ＧＰＵ１４に対して、より多くのデータを並列的に処理させることができるように、アプリケーション２０のコードを修正しうる。アプリケーション２０がより新しいＧＰＵにおける動作により適するように、エミュレータ・ユニット２６がアプリケーション２０を修正しうる、その他の手法の例が存在し、本開示の態様は、前述した例に限定されると考えられるべきではない。

【００９４】

アプリケーション２０を最適化した後に、エミュレータ・ユニット２６は、修正済みまたは更新済みのアプリケーション２０のコードをデバイス１２へ送信しうる。この例において、プロセッサ１６は、アプリケーション２０のコードを、エミュレータ・ユニット２６から受信すると、コンパイルしうる。そして、結果的に得られたオブジェクト・コードを実行するようにＧＰＵ１４に対して指示しうる。その他のいくつかの例では、エミュレータ・ユニット２６は、修正されたアプリケーション２０を、コンパイラ３６によってコンパイルし、結果的に得られたオブジェクト・コードを、デバイス１２へ送信しうる。この例において、プロセッサ１６は、ＧＰＵ１４に対して、受信されたアプリケーション２０のオブジェクト・コードを実行されうる。

【００９５】

いくつかの例において、エミュレータ・ユニット２６は、アプリケーション２０を検証し、アプリケーション２０を一度最適化または調整しうる。このような検証後、ＧＰＵ１４は、さらなる検証または最適化を必要とすることなく、必要に応じて、アプリケーション２０を実行しうる。さらに、いくつかの例では、エミュレータ・ユニット２６は、アプリケーション２０を検証した後、このアプリケーション２０がすでに検証されていることを示すインジケーションを、サーバ・メモリ２８に格納しうる。これらの例では、エミュレータ・ユニット２６は、検証のためのコードを受信した場合、先ず、サーバ・メモリ２８に格納されたインジケーションに基づいて、このコードを既に検証しているか否かを判定しうる。エミュレータ・ユニット２６がこのコードを以前に検証しているのであれば、エミュレータ・ユニット２６は、受信されたコードを直ちに有効にしうる。例えば、エミュレータ・ユニット２６は、アプリケーション２０がデバイス１２から受信されると、アプリケーション２０を検証しうる。その後、エミュレータ・ユニット２６は、デバイス１

２以外のデバイスから、アプリケーション２０のコードを受信しうる。このケースでは、エミュレータ・ユニット２６は、先ず、受信されたコードが、エミュレータ・ユニット２６が以前に検証したコードと同じであることを判定しうる。そして、同じであると判定されると、受信されたコードを直ちに有効にしうる。このように、エミュレータ・ユニット２６は、以前に検証されたコードのために、静止分析および／または動的分析を再び実行する必要がないことがありうる。

【００９６】

図２は、デバイス１２の動作の例を例示するフローチャートである。例示のみの目的のために、図１が参照される。デバイス１２は、ＧＰＵ１４によって実行されるべきであるアプリケーション２０を受信しうる（４０）。例えば、デバイス１２は、アプリケーション・サーバ・デバイス３８からアプリケーション２０をダウンロードしうる。別の例として、アプリケーション２０は、デバイス・メモリ１８にプレ・インストールされうる。前述したように、デバイス１２は、アプリケーション２０のソース・コード、中間コード（例えば、アプリケーション２０の中間表現）、またはオブジェクト・コードを受信しうる。

【００９７】

デバイス１２は、検証サーバ・デバイス２４へ、アプリケーション２０のコードを送信しうる（４２）。例えば、デバイス１２は、アプリケーション２０の検証のために、アプリケーション２０のソース・コード、中間コード、またはオブジェクト・コードを検証サーバ・デバイス２４へ送信しうる。いくつかの例において、デバイス１２は、アプリケーション２０のコードを、検証のために一度、検証サーバ・デバイス２４へ送信しうる。その後、デバイス１２のＧＰＵ１４は、その後の検証を必要とすることなく、必要に応じて、アプリケーション２０を実行しうる。

【００９８】

検証のために、アプリケーション２０のコードを、検証サーバ・デバイス２４へ送信することに応じて、デバイス１２は、検証サーバ・デバイス２４から検証を受信しうる（４４）。あるいは、デバイス１２は、検証失敗、または、検証または検証失敗の何れかを受信しうる。サーバ・デバイス２４からの検証は、アプリケーション２０が１または複数のパフォーマンス基準を満足することを示しうる。アプリケーション２０が、１または複数のパフォーマンス基準を満足していないのであれば、検証サーバ・デバイス２４は、アプリケーション２０がパフォーマンス基準を満足していないことを示しうる。例えば、検証は、アプリケーション２０が、静的分析、動的分析、または、静的分析と動的分析との両方に関連付けられたパフォーマンス基準を満足することを示しうる。いくつかの例において、検証サーバ・デバイス２４は、アプリケーション２０をより効率的にするように、または、よりエラーの少ない傾向になるように、アプリケーション２０を最適化または調整しうる。このケースでは、検証は、修正されたアプリケーション２０のバージョンが、１または複数のパフォーマンス基準を満足していることを示しうる。

【００９９】

いくつかの例において、デバイス１２のプロセッサ１６は、検証に基づいて、デバイス１２のＧＰＵ１４に対して、アプリケーション２０を実行するように指示しうる（４８）。例えば、アプリケーション２０がパフォーマンス基準を満足していることを検証サーバ・デバイス２４が示すのであれば、プロセッサ１６は、ＧＰＵ１４に対して、アプリケーション２０を実行するように指示しうる。そうではない場合、プロセッサ１６は、ＧＰＵ１４がアプリケーション２０を実行することを許可しないことがありうる。

【０１００】

いくつかの代替例では、デバイス１２は、実行前に、修正されたアプリケーション２０のバージョンを受信しうる（４６）。図２では、ブロック４４からブロック４６、および、ブロック４６からブロック４８への破線は、ブロック４６の機能が、すべての例において必要とされている訳ではないことを示すために使用される。例えば、検証サーバ・デバイス２４は、アプリケーション２０を最適化または調整できうる。そして、修正されたア

プリケーション２０のバージョンを送信しうる。別の例として、デバイス１２は、アプリケーション２０のソース・コードまたは中間コードを送信し、コンパイルされたバージョンのアプリケーション２０を、検証サーバ・デバイス２４から受信しうる。さらに別の例として、デバイス１２は、コンパイルされたバージョンのコードを、検証サーバ・デバイス２４によって修正された（例えば、最適化または調整のために修正された）ものとして受信しうる。これらの例において、プロセッサ１６は、ＧＰＵ１４に対して、修正されたバージョンのアプリケーション２０を実行するように指示しうる（４８）。

【０１０１】

図３は、検証サーバ・デバイス２４の動作の例を例示するフローチャートである。例示のみの目的のために、図１が参照される。検証サーバ・デバイス２４は、ＧＰＵ１４によって実行されるべきアプリケーション２０を、デバイス１２から受信しうる（５０）。例えば、検証サーバ・デバイス２４は、アプリケーション２０のソース・コード、中間コード、またはオブジェクト・コードを、ネットワーク２２を介してデバイス１２から受信しうる。

【０１０２】

検証サーバ・デバイス２４は、アプリケーション２０について、静的分析および動的分析のうちの少なくとも１つを実行しうる（５２）。例えば、静的分析の一部として、検証サーバ・デバイス２４のエミュレータ・ユニット２６は、アプリケーション２０のコードをコンパイルし、アプリケーション２０のコンパイル中におけるエラーをモニタしうる。動的分析の一部として、検証サーバ・デバイス２４のエミュレータ・ユニット２６は、ＧＰＵ１４の仮想モデル、または、ＧＰＵ１４の仮想モデルおよびデバイス１２の仮想モデルを実行しうる。前述したように、ＧＰＵモデル３０およびデバイス・モデル３４は、ＧＰＵ１４およびデバイス１２の仮想モデルをそれぞれ含みうる。いくつかの例において、ＧＰＵモデル３０およびデバイス・モデル３４は、一般的なＧＰＵモデルおよび一般的なデバイス・モデルを含みうる。

【０１０３】

例えば、エミュレータ・ユニット２６は、デバイス１２から、ＧＰＵ１４および／またはデバイス１２の識別情報を受信しうる。エミュレータ・ユニット２６は、ＧＰＵモデル３０のうちのどれがＧＰＵ１４に対応し、デバイス・モデル３４のうちのどれがデバイス１２に対応しているのかを判定し、対応するＧＰＵモデルおよびデバイス・モデルを実行しうる。ＧＰＵ１４およびデバイス１２のための対応するＧＰＵモデルおよび／またはデバイス・モデルが存在しない場合、または、エミュレータ・ユニット２６がＧＰＵ１４および／またはデバイス１２の識別情報を受信しなかった場合、エミュレータ・ユニット２６は、一般的なＧＰＵモデルおよびデバイス・モデルを実行しうる。

【０１０４】

動的分析の一部として、エミュレータ・ユニット２６は、アプリケーション２０を実行し、アプリケーション２０を分析するために、ＧＰＵ入力３２とともにアプリケーション２０を入力しうる。これらの例において、アプリケーション２０は、エミュレータ・ユニット２６で動作している、対応するＧＰＵ１４の仮想モデルで動作しているものと考えられうる。このように、エミュレータ・ユニット２６は、あたかもアプリケーション２０がＧＰＵ１４において実行しているかのように、アプリケーション２０を実行させる。エミュレータ・ユニット２６は、例えば、メモリ・アクセス、実行レート、終了インスタンス、および、ＧＰＵ１４の機能に関連するその他の機能のような、対応するＧＰＵ１４の仮想モデルによって実行された機能をモニタしうる。

【０１０５】

エミュレータ・ユニット２６は、アプリケーション２０が１または複数のパフォーマンス基準を満足するか否かを判定しうる（５４）。１または複数のパフォーマンス基準は、静的分析に関連付けられたパフォーマンス基準、および、動的分析に関連付けられたパフォーマンス基準でありうる。例えば、１または複数のパフォーマンス基準は、静的分析中にアプリケーション２０をコンパイルすることによって評価されるように、アプリケーシ

ョン２０のコンパイル中にエラーが無いという基準でありうる。別の例として、１または複数のパフォーマンス基準は、動的分析中にアプリケーション２０を実行し、アプリケーション２０にＧＰＵ入力３２を提供することによって評価されるように、ＣＰＵ１４がその他のタスクを並列的に実行することができないように、アプリケーション２０がアウト・オブ・バウンズのメモリ位置にアクセスすることも無く、かつ、ＧＰＵ１４のリソースを使い果たすこともないという基準でありうる。アプリケーション２０が満足しているとエミュレータ・ユニット２６が判定するその他のパフォーマンス基準の例が存在しうる。

【０１０６】

検証サーバ・デバイス２４は、この判定に基づいて、アプリケーション２０の検証を、デバイス１２へ送信しうる（５６）。例えば、アプリケーション２０が１または複数のパフォーマンス基準を満足するのであれば、検証サーバ・デバイス２４は、アプリケーション２０の検証を、デバイス１２へ送信しうる。そうではなく、アプリケーション２０が、１または複数のパフォーマンス基準を満足しないのであれば、検証サーバ・デバイス２４は、検証失敗を送信しうる。例えば、アプリケーション２０が１または複数のパフォーマンス基準を満足しているとエミュレータ・ユニット２６が判定すると、検証サーバ・デバイス２４は、このことを示すインジケーションをデバイス１２へ送信しうる。あるいは、アプリケーション２０が１または複数のパフォーマンス基準を満足していないとエミュレータ・ユニット２６が判定すると、検証サーバ・デバイス２４は、このことを示すインジケーションをデバイス１２へ送信しうる。

【０１０７】

図４は、検証サーバ・デバイス２４の別の動作の例を例示するフローチャートである。例示のみの目的のために、図１および図３が参照される。図３と同様に、検証サーバ・デバイス２４は、ＧＰＵ１４によって実行されるべきアプリケーション２０を、デバイス１２から受信しうる（５８）。この例において、エミュレータ・ユニット２６は、アプリケーション２０を最適化または調整するために、アプリケーション２０（例えば、アプリケーション２０のソース・コードまたは中間コード）を修正しうる。例えば、アプリケーション２０がＧＰＵ１４においてより効率的に実行できるように、エミュレータ・ユニット２６は、アプリケーション２０のコードを修正しうる。その後、検証サーバ・デバイス２４は、修正されたアプリケーション２０をデバイス１２へ送信しうる（６２）。いくつかの例において、検証サーバ・デバイス２４は、修正されたアプリケーション２０のソース・コードまたは中間コードを送信しうる。別の例として、検証サーバ・デバイス２４は、修正されたアプリケーションのコードをコンパイルし、結果として得られたオブジェクト・コードをデバイス１２へ送信しうる。

【０１０８】

図５は、図１のデバイスの例をさらに詳細に例示するブロック図である。例えば、図５は、図１のデバイス１２をさらに詳細に例示する。例えば、前述したように、デバイス１２の例は、限定される訳ではないが、モバイル無線電話、ＰＤＡ、ビデオ・ディスプレイを含むビデオ・ゲーム・コンソール、モバイル・ビデオ会議ユニット、ラップトップ・コンピュータ、デスクトップ・コンピュータ、テレビジョン・セット・トップ・ボックス等を含む。

【０１０９】

図５に例示されるように、デバイス１２は、ＧＰＵ１４、プロセッサ１６、デバイス・メモリ１８、トランシーバ・モジュール６４、ユーザ・インタフェース６６、ディスプレイ６８、およびディスプレイ・プロセッサ７０を含みうる。ＧＰＵ１４、プロセッサ１６、およびデバイス・メモリ１８は、図１に例示されたものに実質的に類似しうるか、または、同一でありうる。簡潔さの目的のために、図１において図示されず、図５において図示されている構成要素のみが、さらに詳細に記載される。

【０１１０】

デバイス１２は、明瞭さの目的のために、図５において図示されない追加のモジュールまたはユニットを含みうる。例えば、デバイス１２は、スピーカおよびマイクロホンを含

みうる。これらはいずれも図５に図示されておらず、デバイス１２がモバイル無線電話またはスピーカである例において、電話通信を有効にする。あるいは、デバイス１２は、デバイス１２がメディア・プレーヤである場合、スピーカを含みうる。さらに、デバイス１２に示されるさまざまなモジュールおよびユニットは、デバイス１２のすべての例において必ずしも必要とされる訳ではない。例えば、ユーザ・インタフェース６６およびディスプレイ６８は、デバイス１２がデスクトップ・コンピュータであるか、または、外部ユーザ・インタフェースまたはディスプレイとインタフェースするために設けられたその他のデバイスである例では、デバイス１２の外部にありうる。

【０１１１】

ユーザ・インタフェース６６の例は、限定される訳ではないが、トラックボール、マウス、キーボード、およびその他のタイプの入力デバイスを含む。ユーザ・インタフェース６６はまた、タッチ・スクリーンでありうる。そして、ディスプレイ６８の一部として組み込まれうる。トランシーバ・モジュール６４は、デバイス１２と、別のデバイスまたはネットワークとの間の無線通信または有線通信を可能にするための回路を含みうる。トランシーバ・モジュール６４は、１または複数の変調器、復調器、増幅器、アンテナ、および、有線通信または無線通信のための他のこのような回路を含みうる。ディスプレイ６８は、液晶ディスプレイ（ＬＣＤ）、有機発光ダイオード・ディスプレイ（ＯＬＥＤ）、陰極線管（ＣＲＴ）ディスプレイ、プラズマ・ディスプレイ、偏光ディスプレイ、またはその別のタイプのディスプレイ・デバイスを含みうる。

【０１１２】

いくつかの例では、ＧＰＵ１４は、ディスプレイ６８における表示のためのグラフィック・データを生成した後、結果として得られたグラフィック・データを、一時的記憶のためにデバイス・メモリ１８へ出力しうる。ディスプレイ・プロセッサ７０は、デバイス・メモリ１８からグラフィック・データを取得し、このグラフィック・データにポスト処理を実行し、結果として得られたグラフィック・データをディスプレイ６８へ出力しうる。例えば、ディスプレイ・プロセッサ７０は、さらなるエンハンスメントを実行しうるか、または、ＧＰＵ１４によって生成されたグラフィック・データをスケールしうる。

【０１１３】

１または複数の例では、記載された機能は、ハードウェア、ソフトウェア、ファームウェア、またはこれらの任意の組み合わせで実現されうる。これら機能は、ソフトウェアで実施されるのであれば、１または複数の命令群またはコードとしてコンピュータ読取可能な媒体に格納されうる。コンピュータ読取可能な媒体は、コンピュータ・データ記憶媒体を含みうる。データ記憶媒体は、本開示において記述された技術を実施するための命令群、コード、および／または、データ構造を検索するために１または複数のコンピュータまたは１または複数のプロセッサによってアクセスされうる任意の利用可能な媒体でありうる。限定するのではなく、例として、このようなコンピュータ読取可能な媒体は、ランダム・アクセス・メモリ（ＲＡＭ）、読取専用メモリ（ＲＯＭ）、ＥＥＰＲＯＭ、ＣＤ－ＲＯＭまたはその他の光ディスク記憶装置、磁気ディスク記憶装置またはその他の磁気記憶デバイス、または、命令群またはデータ構造の形態で所望のプログラム・コードを格納するために使用され、かつ、コンピュータによってアクセスされることが可能なその他任意の媒体を備えうる。本明細書で使用されるようにディスク（ｄｉｓｋおよびｄｉｓｃ）は、コンパクト・ディスク（ｄｉｓｃ）（ＣＤ）、レーザ・ディスク（ｄｉｓｃ）、光ディスク（ｄｉｓｃ）、デジタル多用途ディスク（ｄｉｓｃ）（ＤＶＤ）、フロッピー（登録商標）ディスク（ｄｉｓｋ）、Ｂｌｕ－ｒａｙ（登録商標）ディスク（ｄｉｓｃ）を含む。ここで、ｄｉｓｋは通常、データを磁気的に再生し、ｄｉｓｃは、レーザを用いてデータを光学的に再生する。前述した組み合わせもまた、コンピュータ読取可能な媒体の範囲内に含まれるべきである。

【０１１４】

このコードは、例えば、１または複数のデジタル信号プロセッサ（ＤＳＰ）、汎用マイクロプロセッサ、特定用途向けＩＣ（ＡＳＩＣ）、フィールド・プログラマブル・ロジッ

ク・アレイ（ＦＰＧＡ）、またはその他の等価な統合またはディスクリートな論理回路のような１または複数のプロセッサによって実行されうる。したがって、本明細書で使用されているように、用語「プロセッサ」は、前述した構成、または、本明細書に記載された技術の実施のために適切なその他任意の構成のうちの何れかを称しうる。さらに、これら技法は、１または複数の回路または論理要素で完全に実現されうる。

【０１１５】

本開示の技法は、無線ハンドセット、集積回路（ＩＣ）、またはＩＣのセット（すなわち、チップ・セット）を含む広範なデバイスまたは装置において実現される。本開示では、さまざまな構成要素、モジュール、またはユニットが、開示された技法を実行するように構成されたデバイスの機能的な態様を強調するために記載されているが、必ずしも、別のハードウェア・ユニットによる実現を必要としている訳ではない。むしろ、前述したように、さまざまなユニットは、適切なソフトウェアおよび／またはファームウェアと連携して、ハードウェア・ユニット内に結合されうるか、または、前述した１または複数のプロセッサを含む相互運用的なハードウェア・ユニットの集合によって提供されうる。

【０１１６】

さまざまな例が記載された。これらの例およびその他の例は、以下の特許請求の範囲のスコープ内である。

以下に本願発明の当初の特許請求の範囲に記載された発明を付記する。

［Ｃ１］

方法であって、

サーバ・デバイスの外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを、前記サーバ・デバイスを用いて受信すること、

前記サーバ・デバイスにおける前記アプリーションのコンパイル前およびコンパイル中における前記アプリケーションの分析と、前記サーバ・デバイスにおける前記アプリケーションの実行中における前記アプリケーションの分析と、のうちの少なくとも１つを前記サーバ・デバイスにおいて実行することと、

前記分析のうちの少なくとも１つに基づいて、前記アプリケーションが、１または複数のパフォーマンス基準を満足しているか否かを判定することと、

前記アプリケーションが、前記１または複数のパフォーマンス基準を満足しているのであれば、前記アプリケーションの検証を前記デバイスへ送信することと、

を備える方法。

［Ｃ２］

前記パフォーマンス基準は、前記アプリケーションが、悪意のあるコードではないとの判定と、前記アプリケーションが、エラーのある傾向にないとの判定とのうちの少なくとも１つを備える、Ｃ１に記載の方法。

［Ｃ３］

前記パフォーマンス基準は、前記アプリケーションのコードが、既知のウィルスのコードを含んでいないとの判定と、前記アプリケーションのコードのコンパイル中にエラーが発見されないと判定されたとの判定と、前記アプリケーションの実行中にアウト・オブ・バウンズのメモリ・アクセスがないと判定されたとの判定と、前記アプリケーションの実行中に前記デバイスのシステム・バスがオーバロードしていないと判定されたとの判定と、前記アプリケーションのタスクがしきい実行時間内に実行を完了したとの判定と、前記アプリケーションのタスクが少なくともしきい実行レートで実行しているとの判定と、のうちの１または複数を含む、Ｃ１に記載の方法。

［Ｃ４］

前記アプリーションのコンパイル前およびコンパイル中における前記アプリケーションの分析は、

前記アプリケーションのコードを既知のウィルスのコードと比較することと、

前記アプリケーションのコードのコンパイル中に何らかのエラーが発見されたか否かを

判定することと
を備える、Ｃ１に記載の方法。
［Ｃ５］
前記アプリケーションの実行中に前記アプリケーションの分析を実行することは、
仮想ＧＰＵモデルを実行することと、
前記仮想ＧＰＵモデルにおいて前記アプリケーションを実行することと、
前記仮想ＧＰＵモデルにおける前記アプリケーションの実行中、前記仮想ＧＰＵモデルの機能を分析することと
を備える、Ｃ１に記載の方法。
［Ｃ６］
仮想デバイス・モデルを実行することと、
前記デバイス・ＧＰＵモデルにおける前記アプリケーションの実行中、前記仮想デバイス・モデルの機能を分析することと、
をさらに備えるＣ５に記載の方法。
［Ｃ７］
前記仮想ＧＰＵモデルにおいて前記アプリケーションを実行することは、
前記仮想ＧＰＵモデルにおいて実行しているアプリケーションに、ＧＰＵ入力を入力すること
を備える、Ｃ５に記載の方法。
［Ｃ８］
前記実行されたアプリケーションによって実行される機能をモニタすること、をさらに備えるＣ５に記載の方法。
［Ｃ９］
前記機能をモニタすることは、
前記実行されたアプリケーションによるメモリ・アクセスをモニタすることと、
実行のレートをモニタすることと、
実行時間をモニタすることと、
のうちの１または複数を備える、Ｃ８に記載の方法。
［Ｃ１０］
前記アプリケーションのコードを修正することと、
前記修正されたアプリケーションのコードを、前記デバイスへ送信することと、
をさらに備えるＣ１に記載の方法。
［Ｃ１１］
前記アプリケーションが前記ＧＰＵにおいて非効率的に動作するであろうことを判定することをさらに備え、
前記アプリケーションのコードを修正することは、前記判定に基づいて、前記アプリケーションのコードを修正することを備える、Ｃ１０に記載の方法。
［Ｃ１２］
前記アプリケーションの実行中に前記アプリケーションの分析を実行することは、
前記アプリケーションをハードウェア・エミュレーション・ボードにおいて実行することと、
前記実行中に、前記ハードウェア・エミュレーション・ボードの機能を分析することと
を備える、Ｃ１に記載の方法。
［Ｃ１３］
前記アプリケーションを受信することは、前記アプリケーションのソース・コードおよび中間コードのうちの少なくとも１つを受信することを備え、
前記方法はさらに、
前記アプリケーションのオブジェクト・コードを生成するために、前記アプリケーションのソース・コードおよび中間コードのうちの少なくとも１つをコンパイルすることと、
前記アプリケーションのオブジェクト・コードを前記デバイスへ送信することと、

を備えるＣ１に記載の方法。
［Ｃ１４］
装置であって、
前記装置の外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信し、
前記装置における前記アプリーションのコンパイル前およびコンパイル中における前記アプリケーションの分析と、前記装置における前記アプリケーションの実行中における前記アプリケーションの分析と、のうちの少なくとも１つを実行し、
前記分析のうちの少なくとも１つに基づいて、前記アプリケーションが、１または複数のパフォーマンス基準を満足しているか否かを判定し、
前記アプリケーションが、前記１または複数のパフォーマンス基準を満足しているのであれば、前記アプリケーションの検証を前記デバイスへ送信する、
ように動作可能なエミュレータ・ユニット、を備える装置。
［Ｃ１５］
前記パフォーマンス基準は、前記アプリケーションが、悪意のあるコードではないとの判定と、前記アプリケーションが、エラーのある傾向にないとの判定とのうちの少なくとも１つを備える、Ｃ１４に記載の装置。
［Ｃ１６］
前記パフォーマンス基準は、前記アプリケーションのコードが、既知のウィルスのコードを含んでいないとの判定と、前記アプリケーションのコードのコンパイル中にエラーが発見されないと判定されたとの判定と、前記アプリケーションの実行中にアウト・オブ・バウンズのメモリ・アクセスがないと判定されたとの判定と、前記アプリケーションの実行中に前記デバイスのシステム・バスがオーバロードしていないと判定されたとの判定と、前記アプリケーションのタスクがしきい実行時間内に実行を完了したとの判定と、前記アプリケーションのタスクが少なくともしきい実行レートで実行しているとの判定と、のうちの１または複数を含む、Ｃ１４に記載の装置。
［Ｃ１７］
前記エミュレータ・ユニットは、前記アプリケーションのコードを既知のウィルスのコードと比較し、
前記コンパイル前および前記コンパイル中に、前記アプリケーションの分析を実行するために、前記アプリケーションのコードのコンパイル中に何らかのエラーが発見されたか否かを判定する、Ｃ１４に記載の装置。
［Ｃ１８］
メモリをさらに備え、
前記アプリケーションの実行中に、前記アプリケーションの分析を実行するために、前記エミュレータ・ユニットは、前記メモリに格納された仮想ＧＰＵモデルを実行し、前記仮想ＧＰＵモデルにおいて前記アプリケーションを実行し、前記仮想ＧＰＵモデルにおける前記アプリケーションの実行中に、前記仮想ＧＰＵモデルの機能を分析するように動作可能である、Ｃ１４に記載の装置。
［Ｃ１９］
前記エミュレータ・ユニットはさらに、
前記メモリに格納された仮想デバイス・モデルを実行し、
前記仮想ＧＰＵモデルにおけるアプリケーションの実行中に、前記仮想デバイス・モデルの機能を分析する
ように動作可能である、Ｃ１８に記載の装置。
［Ｃ２０］
メモリをさらに備え、
前記エミュレータ・ユニットは、前記仮想ＧＰＵモデルにおけるアプリケーションの実行中、前記仮想ＧＰＵモデルにおいて実行しているアプリケーションへ、前記メモリに格納されたＧＰＵ入力を入力する、Ｃ１８に記載の装置。

［Ｃ２１］
前記エミュレータ・ユニットはさらに、前記実行されたアプリケーションによって実行される機能をモニタするように動作可能である、Ｃ１８に記載の装置。
［Ｃ２２］
前記エミュレータ・ユニットは、前記実行されたアプリケーションによるメモリ・アクセス、実行のレート、および実行時間のうちの１または複数をモニタするように動作可能である、Ｃ２１に記載の装置。
［Ｃ２３］
前記エミュレータ・ユニットはさらに、
前記アプリケーションのコードを修正し、
前記修正されたアプリケーションのコードを前記デバイスへ送信する
ように動作可能である、Ｃ１４に記載の装置。
［Ｃ２４］
前記エミュレータ・ユニットはさらに、
前記アプリケーションが前記ＧＰＵにおいて非効率的に動作するであろうことを判定し、
前記判定に基づいて、前記アプリケーションのコードを修正する
ように動作可能である、Ｃ２３に記載の装置。
［Ｃ２５］
前記エミュレータ・ユニットは、ハードウェア・エミュレーション・ボードを備え、
前記ハードウェア・エミュレーション・ボードは、前記アプリケーションの実行中に、前記アプリケーションの分析を実行するために、前記アプリケーションを実行する、Ｃ１４に記載の装置。
［Ｃ２６］
前記エミュレータ・ユニットは、前記アプリケーションのソース・コードおよび中間コードのうちの少なくとも１つを受信し、
前記エミュレータ・ユニットはさらに、
前記アプリケーションのオブジェクト・コードを生成するために、前記アプリケーションのソース・コードおよび中間コードのうちの少なくとも１つをコンパイルし、
前記アプリケーションのオブジェクト・コードを前記デバイスへ送信する
ように動作可能である、Ｃ１４に記載の装置。
［Ｃ２７］
サーバ・デバイスであって、
前記サーバ・デバイスの外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信する手段と、
前記サーバ・デバイスにおける前記アプリーションのコンパイル前およびコンパイル中における前記アプリケーションの分析と、前記サーバ・デバイスにおける前記アプリケーションの実行中における前記アプリケーションの分析と、のうちの少なくとも１つを実行する手段と、
前記分析のうちの少なくとも１つに基づいて、前記アプリケーションが、１または複数のパフォーマンス基準を満足しているか否かを判定する手段と、
前記アプリケーションが、前記１または複数のパフォーマンス基準を満足しているのであれば、前記アプリケーションの検証を前記デバイスへ送信する手段と、
を備えるサーバ・デバイス。
［Ｃ２８］
１または複数のプロセッサに対して、
サーバ・デバイスの外部にあるデバイスに存在するグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを、前記サーバ・デバイスを用いて受信することと、
前記サーバ・デバイスにおける前記アプリーションのコンパイル前およびコンパイル

中における前記アプリケーションの分析と、前記サーバ・デバイスにおける前記アプリケーションの実行中における前記アプリケーションの分析と、のうちの少なくとも１つを、前記サーバ・デバイスを用いて実行することと、
　前記分析のうちの少なくとも１つに基づいて、前記アプリケーションが、１または複数のパフォーマンス基準を満足しているか否かを判定することと、
　前記アプリケーションが、前記１または複数のパフォーマンス基準を満足しているのであれば、前記アプリケーションの検証を前記デバイスへ送信することと、
をさせる命令群を備えた非一時的なコンピュータ読取可能な記憶媒体。
［Ｃ２９］
　方法であって、
　デバイスのグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信することと、
　前記アプリケーションの検証のために、前記デバイスの外部にあるサーバ・デバイスへ、前記アプリケーショを送信することと、
　前記アプリケーションが、前記ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、前記サーバ・デバイスから受信することと、
を備える方法。
［Ｃ３０］
　前記受信した検証に基づいて、前記ＧＰＵにおいてアプリケーションを実行すること、をさらに備えるＣ２９に記載の方法。
［Ｃ３１］
　前記アプリケーションを受信することは、前記アプリケーションのソース・コード、前記アプリケーションの中間コード、および、前記アプリケーションのコンパイルされたコードのうちの少なくとも１つを受信することを備え、
　前記アプリケーションを送信することは、前記アプリケーションのソース・コード、前記アプリケーションの中間コード、および、前記アプリケーションのコンパイルされたコードのうちの少なくとも１つを送信することを備える、
Ｃ２９に記載の方法。
［Ｃ３２］
　前記修正されたバージョンのアプリケーションを、前記サーバ・デバイスから受信することと、
　前記修正されたバージョンのアプリケーションを前記ＧＰＵにおいて実行することと、
をさらに備えるＣ２９に記載の方法。
［Ｃ３３］
　前記アプリケーションを送信することは、前記アプリケーションのソース・コードおよび前記アプリケーションの中間コードのうちの少なくとも１つを送信することを備え、
　前記方法はさらに、
　前記アプリケーションの、コンパイルされたオブジェクト・コードを、前記サーバ・デバイスから受信することと、
　前記アプリケーションの、コンパイルされたオブジェクト・コードを、前記ＧＰＵにおいて実行することと
を備える、Ｃ２９に記載の方法。
［Ｃ３４］
　前記サーバ・デバイスへ前記アプリケーションを送信することは、前記アプリーションを一度だけ前記サーバ・デバイスへ送信することを備え、
　前記サーバ・デバイスから前記検証を受信することは、前記検証を、前記サーバ・デバイスから、一度だけ受信することを備える、Ｃ２９に記載の方法。
［Ｃ３５］
　装置であって、
　グラフィック処理ユニット（ＧＰＵ）と、

前記ＧＰＵによって実行されるべきアプリケーションを格納するように動作可能なデバイス・メモリと、
前記装置の外部にあるサーバ・デバイスへ前記アプリケーションを送信し、
前記アプリケーションが、前記ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、前記サーバ・デバイスから受信する
ように動作可能なプロセッサと、
を備える装置。
［Ｃ３６］
前記プロセッサはさらに、前記受信された検証に基づいて、前記アプリケーションを実行するように前記ＧＰＵに対して指示するように動作可能であり、
前記ＧＰＵは、前記プロセッサからの指示に応じて、前記アプリケーションを実行するように動作可能である、Ｃ３５に記載の装置。
［Ｃ３７］
前記プロセッサは、前記アプリケーションのソース・コード、前記アプリケーションの中間コード、および、前記アプリケーションのコンパイルされたコードのうちの少なくとも１つを受信し、
前記プロセッサは、前記アプリケーションのソース・コード、前記アプリケーションの中間コード、および、前記アプリケーションのコンパイルされたコードのうちの少なくとも１つを送信する、Ｃ３５に記載の装置。
［Ｃ３８］
前記プロセッサはさらに、前記修正されたバージョンのアプリケーションを、前記サーバ・デバイスから受信するように動作可能であり、
前記ＧＰＵはさらに、前記修正されたバージョンのアプリケーションを実行するように動作可能である、Ｃ３５に記載の装置。
［Ｃ３９］
前記プロセッサは、前記アプリケーションのソース・コードおよび前記アプリケーションの中間コードのうちの少なくとも１つを送信し、
前記プロセッサはさらに、前記アプリケーションのコンパイルされたオブジェクト・コードを受信するように動作可能であり、
前記ＧＰＵはさらに、前記アプリケーションのコンパイルされたオブジェクト・コードを実行するように動作可能である、
Ｃ３５に記載の装置。
［Ｃ４０］
前記プロセッサは、前記アプリーションを一度だけ前記サーバ・デバイスへ送信し、前記プロセッサは、前記検証を、前記サーバ・デバイスから、一度だけ受信する、Ｃ３５に記載の装置。
［Ｃ４１］
デバイスであって、
グラフィック処理ユニット（ＧＰＵ）と、
前記ＧＰＵによって実行されるべきアプリケーションを受信する手段と、
前記アプリケーションの検証のために、前記デバイスの外部にあるサーバ・デバイスへ、前記アプリケーションを送信する手段と、
前記アプリケーションが、前記ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、前記サーバ・デバイスから受信する手段と、
を備えるデバイス。
［Ｃ４２］
１または複数のプロセッサに対して、
デバイスのグラフィック処理ユニット（ＧＰＵ）によって実行されるべきアプリケーションを受信することと、
前記アプリケーションの検証のために、前記デバイスの外部にあるサーバ・デバイス

へ、前記アプリケーションの送信することと、
　前記アプリケーションが、前記ＧＰＵにおける実行のための１または複数の基準を満足していることを示す検証を、前記サーバ・デバイスから受信することと、
を実行させる命令群を備えた非一時的なコンピュータ読取可能な記憶媒体。

【図１】

図１

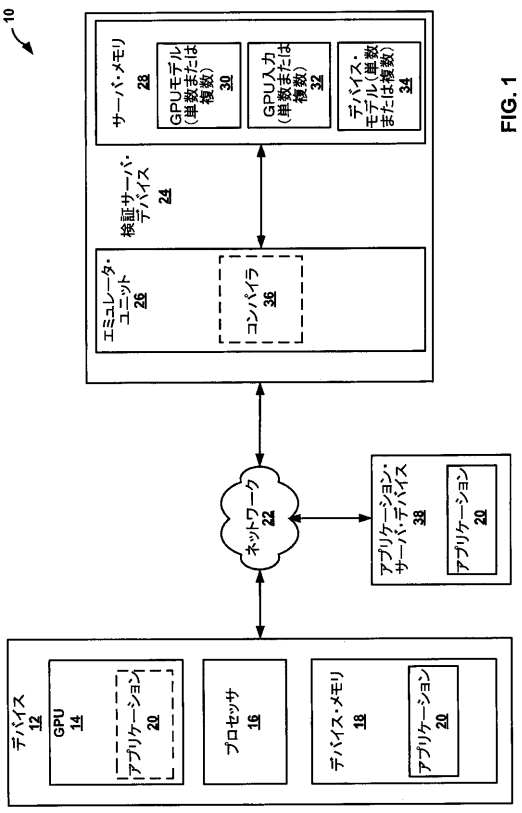

FIG. 1

【図２】

図２

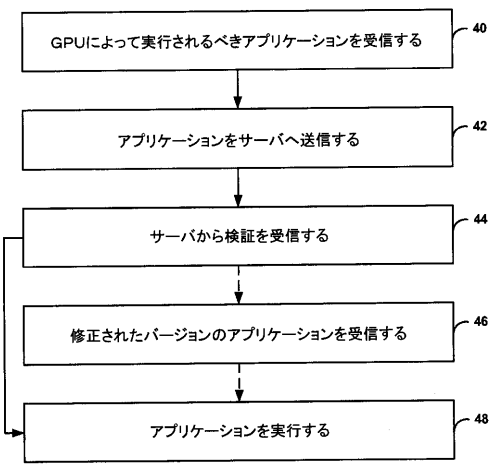

FIG. 2

【図３】

図３

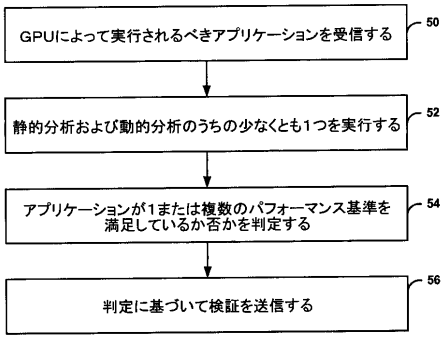

FIG. 3

【図４】

図４

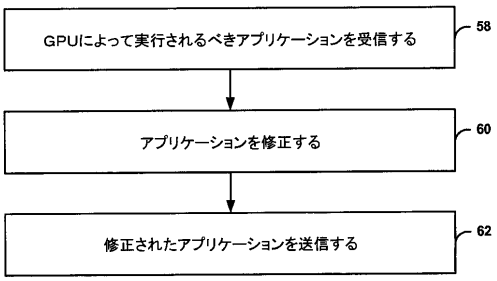

FIG. 4

【図５】

図５

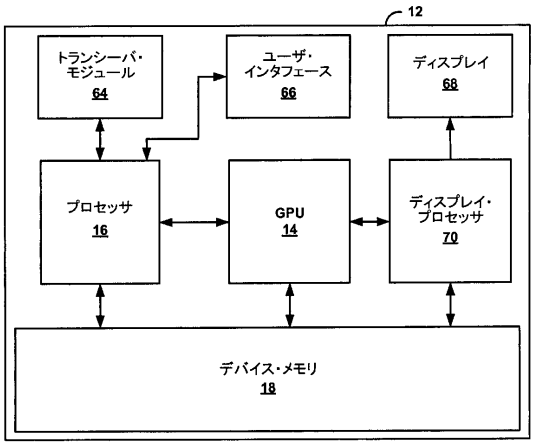

FIG. 5

フロントページの続き

(74)代理人 100153051
弁理士 河野 直樹
(74)代理人 100140176
弁理士 砂川 克
(74)代理人 100158805
弁理士 井関 守三
(74)代理人 100179062
弁理士 井上 正
(74)代理人 100124394
弁理士 佐藤 立志
(74)代理人 100112807
弁理士 岡田 貴志
(74)代理人 100111073
弁理士 堀内 美保子
(72)発明者 ボウルド、アレクセイ・ブイ.
アメリカ合衆国、カリフォルニア州 ９２１２１、サン・ディエゴ、モアハウス・ドライブ ５７７５
(72)発明者 ユン、ジャイ・チュンスプ
アメリカ合衆国、カリフォルニア州 ９２１２１、サン・ディエゴ、モアハウス・ドライブ ５７７５

審査官 多賀 実

(56)参考文献 特開２００３－１１４８１３（ＪＰ，Ａ）
国際公開第２００８／０３８３８９（WO，Ａ１）
特開２００９－０７０３７１（ＪＰ，Ａ）
米国特許第０７０９５４１６（ＵＳ，Ｂ１）
米国特許出願公開第２００３／０００５４２５（ＵＳ，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ９／４５
Ｇ０６Ｆ９／４５５
Ｇ０６Ｆ１１／２８－１１／３６
Ｇ０６Ｆ２１／５６