(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2011-86295
(P2011-86295A)
(43)公開日　平成23年4月28日(2011.4.28)

(51)Int.Cl.		F I		テーマコード（参考）
G06F 11/34	(2006.01)	G06F 11/34	S	5B042
G06F 11/28	(2006.01)	G06F 11/28	340C	

審査請求 未請求　請求項の数 20　ＯＬ　外国語出願　（全 63 頁）

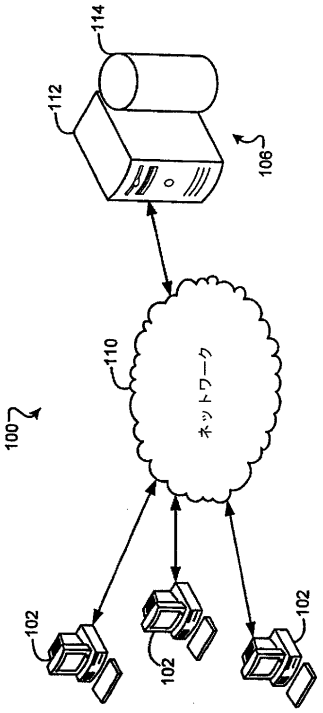
(21)出願番号　特願2010-232543 (P2010-232543)
(22)出願日　平成22年10月15日 (2010.10.15)
(31)優先権主張番号　12/580,901
(32)優先日　平成21年10月16日 (2009.10.16)
(33)優先権主張国　米国 (US)

(71)出願人　300015447
エスアーペー　アーゲー
ＳＡＰ　ＡＧ
ドイツ連邦共和国，６９１９０　バルドルフ，ディートマル－ホップ－アレー　１６
Ｄｉｅｔｍａｒ－Ｈｏｐｐ－Ａｌｌｅｅ　１６，　６９１９０　Ｗａｌｌｄｏｒｆ，　Ｇｅｒｍａｎｙ
(74)代理人　100108453
弁理士　村山　靖彦
(74)代理人　100064908
弁理士　志賀　正武
(74)代理人　100089037
弁理士　渡邊　隆

最終頁に続く

(54)【発明の名称】応答時間に基づいてサービスリソース消費を推定すること

(57)【要約】
【課題】コンピュータシステムの待ち行列モデルを生成するためのコンピュータ実施方法を提供すること。
【解決手段】本開示の実装形態は、複数のサービス要求を備えるワークロードを定義するステップであって、それぞれのサービス要求が複数のクラスのうちの1つのクラスに対応するステップと、サービス要求を受信および処理するコンピュータシステムにそのワークロードを適用するステップと、ワークロードの要求ごとにコンピュータシステムの応答時間を測定するステップと、応答時間に基づくクラスごとの平均サービス需要、およびコンピュータシステムを表す基本的な待ち行列モデルを推定するステップと、平均サービス需要およびワークロードの特徴に基づいて待ち行列モデルを生成するステップとを含む、コンピュータ実施方法を提供する。
【選択図】図１



【特許請求の範囲】
【請求項１】
複数のサービス要求を備えるワークロードを定義するステップであって、それぞれのサービス要求が複数のクラスのうちの1つのクラスに対応するステップと、
サービス要求を受信および処理するコンピュータシステムに前記ワークロードを適用するステップと、
前記ワークロードの要求ごとに前記コンピュータシステムの応答時間を測定するステップと、
前記応答時間に基づくクラスごとの平均サービス需要、および前記コンピュータシステムを表す基本的な待ち行列モデルを推定するステップと、
前記平均サービス需要および前記ワークロードの特徴に基づいて前記待ち行列モデルを生成するステップとを備える、待ち行列モデルを生成するコンピュータ実施方法。
【請求項２】
前記ワークロードのそれぞれの要求に対応する複数の到着待ち行列長を決定するステップをさらに備え、平均サービス需要を推定するステップが前記複数の到着待ち行列長にさらに基づく、請求項1に記載の方法。
【請求項３】
前記複数の到着待ち行列長のそれぞれの到着待ち行列長が、要求の発着時間を報告するログファイルから決定される、請求項2に記載の方法。
【請求項４】
前記ワークロードのそれぞれの要求に対応する複数の残余時間を決定するステップをさらに備え、平均サービス需要を推定するステップが前記複数の残余時間にさらに基づく、請求項1に記載の方法。
【請求項５】
平均サービス需要を推定するステップが、前記測定された応答時間に基づいて直線回帰および最尤法解析のうちの1つを使用して前記基本的な待ち行列モデルにおける平均サービス需要を推定するステップを備える、請求項1に記載の方法。
【請求項６】
前記待ち行列モデルを生成するステップが、前記平均サービス需要を使用して前記基本的な待ち行列モデルをパラメータ化するステップを備える、請求項1に記載の方法。
【請求項７】
複数の入力を使用して前記待ち行列モデルを処理することによってコンピュータシステムの性能を評価するステップをさらに備える、請求項1に記載の方法。
【請求項８】
1つまたは複数のプロセッサに結合されており、前記1つまたは複数のプロセッサによって実行されると、前記1つまたは複数のプロセッサに、
複数のサービス要求を備えるワークロードを定義するステップであって、それぞれのサービス要求が複数のクラスのうちの1つのクラスに対応するステップと、
サービス要求を受信および処理するコンピュータシステムに前記ワークロードを適用するステップと、
前記ワークロードの要求ごとに前記コンピュータシステムの応答時間を測定するステップと、
前記応答時間に基づくクラスごとの平均サービス需要、および前記コンピュータシステムを表す基本的な待ち行列モデルを推定するステップと、
前記平均サービス需要および前記ワークロードの特徴に基づいて前記待ち行列モデルを生成するステップとを備える動作を実行させる命令を内部に格納した、コンピュータ可読記憶媒体。
【請求項９】
前記動作が、前記ワークロードのそれぞれの要求に対応する複数の到着待ち行列長を決定するステップをさらに備え、平均サービス需要を推定するステップが前記複数の到着待

ち行列長にさらに基づく、請求項8に記載の記憶媒体。
【請求項１０】
前記複数の到着待ち行列長のそれぞれの到着待ち行列長が、要求の発着時間を報告するログファイルから決定される、請求項9に記載の記憶媒体。
【請求項１１】
前記動作が、前記ワークロードのそれぞれの要求に対応する複数の残余時間を決定するステップをさらに備え、平均サービス需要を推定するステップが前記複数の残余時間にさらに基づく、請求項8に記載の記憶媒体。
【請求項１２】
平均サービス需要を推定するステップが、前記測定された応答時間に基づいて直線回帰および最尤法解析のうちの1つを使用して前記基本的な待ち行列モデルにおける平均サービス需要を推定するステップを備える、請求項8に記載の記憶媒体。
【請求項１３】
前記待ち行列モデルを生成するステップが、前記平均サービス需要を使用して前記基本的な待ち行列モデルをパラメータ化するステップを備える、請求項8に記載の記憶媒体。
【請求項１４】
複数の入力を使用して前記待ち行列モデルを処理することによってコンピュータシステムの性能を評価するステップをさらに備える、請求項8に記載の記憶媒体。
【請求項１５】
サービス要求を受信および処理するコンピュータシステムと、
1つまたは複数のプロセッサと、
前記1つまたは複数のプロセッサに結合されており、前記1つまたは複数のプロセッサによって実行されると、前記1つまたは複数のプロセッサに、
複数のサービス要求を備えるワークロードを定義するステップであって、それぞれのサービス要求が複数のクラスのうちの1つのクラスに対応するステップと、
前記コンピュータシステムに前記ワークロードを適用するステップと、
前記ワークロードの要求ごとに前記コンピュータシステムの応答時間を測定するステップと、
前記応答時間に基づくクラスごとの平均サービス需要、および前記コンピュータシステムを表す基本的な待ち行列モデルを推定するステップと、
前記平均サービス需要および前記ワークロードの特徴に基づいて前記待ち行列モデルを生成するステップとを備える動作を実行させる命令を内部に格納したコンピュータ可読記憶媒体とを備える、システム。
【請求項１６】
前記動作が、前記ワークロードのそれぞれの要求に対応する複数の到着待ち行列長を決定するステップをさらに備え、平均サービス需要を推定するステップが前記複数の到着待ち行列長にさらに基づく、請求項15に記載のシステム。
【請求項１７】
前記複数の到着待ち行列長のそれぞれの到着待ち行列長が、要求の発着時間を報告するログファイルから決定される、請求項16に記載のシステム。
【請求項１８】
前記動作が、前記ワークロードのそれぞれの要求に対応する複数の残余時間を決定するステップをさらに備え、平均サービス需要を推定するステップが前記複数の残余時間にさらに基づく、請求項15に記載のシステム。
【請求項１９】
平均サービス需要を推定するステップが、前記測定された応答時間に基づいて直線回帰および最尤法解析のうちの1つを使用して基本的な待ち行列モデルにおける平均サービス需要を推定するステップを備える、請求項15に記載の記憶媒体。
【請求項２０】
前記コンピュータシステムが、アプリケーションを実行するアプリケーションサーバ、

および前記要求を生成する1つまたは複数のクライアントシステムを備える、請求項15に記載の記憶媒体。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、応答時間に基づいてサービスリソース消費を推定することに関する。

【背景技術】

【０００２】

ソフトウェアおよびハードウェアシステムのスケーラビリティを予測するために、解析法またはシミュレーションによって性能モデルが実装されうる。より具体的には、ソフトウェアおよびハードウェアシステムへの要求が、その要求に応答するために使用されるシステムリソースに要求を出す。このようなシステムの性能予測を得るためのモデリングおよび評価技法はあるが、サービス需要の良い推定を提供できるリソース消費推定方法はほとんどない。サービス需要は性能モデルの指定において重要なパラメータである。したがって、代表的であり堅牢であるモデルを定義するためのサービス需要の正確な推定が所望される。

【０００３】

従来、中央処理装置(CPU)使用率は、ソフトウェアおよびハードウェアシステムのサービス需要を推定するために実装されてきた。しかし、CPU使用率の使用にはいくつか欠乏しているものがある。たとえば、CPU使用率測定は、ハードウェア上で実行されるオペレーティングシステムへのアクセスを必要とし、また専門のCPUサンプリング計測も必要とする。CPUサンプリング計測は通常のシステム活動を干渉して、不正確なCPU使用率測定を提供する可能性がある。いくつかのシナリオでは、たとえば仮想環境などにおいて、ハイパーバイザオーバーヘッドのフィルタリングが必要なために正確なCPU使用率サンプリングがより困難である。他のシナリオでは、モデル化されるために、システムはCPU使用率データをサーバに提供しないウェブサービスプロバイダなどの第三者に所有される。

【発明の概要】

【課題を解決するための手段】

【０００４】

本開示の実装により、コンピュータシステムの待ち行列モデルを生成するためのコンピュータ実施方法が提供される。ある実装形態では、本方法は複数のサービス要求を備えるワークロードを定義するステップであって、それぞれのサービス要求が複数のクラスのうちの1つのクラスに対応するステップと、サービス要求を受信および処理するコンピュータシステムにそのワークロードを適用するステップと、ワークロードの要求ごとにコンピュータシステムの応答時間を測定するステップと、応答時間に基づくクラスごとの平均サービス需要、およびコンピュータシステムを表す基本的な待ち行列モデルを推定するステップと、平均サービス需要およびワークロードの特徴に基づいて待ち行列モデルを生成するステップとを含む。ワークロードの特徴は、1秒あたり平均5つの要求を有する指数到着時間間隔などの、待ち行列内に到着する要求の到着時間間隔分布を含むことができるが、これに限定されない。

【０００５】

いくつかの実装形態では、本方法は、ワークロードのそれぞれの要求に対応する複数の到着待ち行列長を決定するステップであって、平均サービス需要を推定するステップが複数の到着待ち行列長にさらに基づくステップをさらに含む。複数の到着待ち行列長のそれぞれの到着待ち行列長は、要求の発着時間を報告するログファイルから決定されうる。

【０００６】

いくつかの実装形態では、本方法は、ワークロードのそれぞれの要求に対応する複数の残余時間を決定するステップであって、平均サービス需要を推定するステップが複数の残余時間にさらに基づくステップをさらに含む。それぞれの残余時間は、処理されるべき要求の到着時に作業中の要求の処理を完了するために残っている時間に対応できる。

【０００７】
いくつかの実装形態では、平均サービス需要を推定するステップは、測定された応答時間に基づいて、直線回帰法解析および最尤法解析のうちの1つを使用して基本的な待ち行列モデル内の平均サービス需要を推定するステップを含む。
【０００８】
いくつかの実装形態では、待ち行列モデルを生成するステップは、平均サービス需要を使用して基本的な待ち行列モデルをパラメータ化するステップを含む。
【０００９】
いくつかの実装形態では、基本的な待ち行列モデルは待ち行列モデルの特徴における仮定を含み、その特徴はスケジューリングおよびサービス需要分布のうちの少なくとも1つを備える。
【００１０】
いくつかの実装形態では、コンピュータシステムはアプリケーションを実行するアプリケーションサーバ、および要求を生成する1つまたは複数のクライアントシステムを含む。
【００１１】
いくつかの実装形態では、本方法は、待ち行列モデルを処理することによってコンピュータシステムの性能を評価するステップをさらに含む。
【００１２】
本開示は、1つまたは複数のプロセッサに結合されており、1つまたは複数のプロセッサによって実行されると、その1つまたは複数のプロセッサに本明細書で提供された方法の実施に従って動作を実行させる命令を内部に格納したコンピュータ可読記憶媒体も提供する。
【００１３】
本開示は、本明細書で提供された方法を実施するためのシステムをさらに提供する。システムは、サービス要求を受信および処理するコンピュータシステム、1つまたは複数のプロセッサ、ならびに1つまたは複数のプロセッサに結合されており、1つまたは複数のプロセッサによって実行されると、その1つまたは複数のプロセッサに本明細書で提供された方法の実施に従って動作を実行させる命令を内部に格納したコンピュータ可読記憶媒体を含む。
【００１４】
本開示による方法は、本明細書に記載される態様および特徴のどのような組合せも含むことができることを理解されたい。すなわち、本開示による方法は、本明細書に特に記載された態様および特徴の組合せに限定されず、提供された態様および特徴のどのような組合せも含む。
【００１５】
本開示の1つまたは複数の実装形態の詳細は、添付の図面および以下の記述において説明される。本開示の他の特徴および諸利点は、記述および図面、ならびに特許請求の範囲から明らかになるであろう。
【図面の簡単な説明】
【００１６】
【図１】本開示の実装による例示的なシステムアーキテクチャの略図である。
【図２】例示的な企業資源計画(ERP)システムの機能ブロック図である。
【図３】例示的なERPアプリケーションに対応する例示的な待ち行列モデルの機能ブロック図である。
【図４】本開示の実装形態による応答時間ベースの手法の概要を示す図である。
【図５】測定された応答時間と測定された待ち行列長との間の直線関係を示すグラフである。
【図６Ａ】本開示の実装形態による応答時間手法に基づく応答時間予測と従来の利用ベースの手法との比較を示すグラフである。

【図６Ｂ】本開示の実装形態による応答時間手法に基づく応答時間予測と従来の利用ベースの手法との比較を示すグラフである。
【図７Ａ】異なるサーバ利用レベルでの例示的な平均相対誤差を示すグラフである。
【図７Ｂ】異なるサーバ利用レベルでの例示的な平均相対誤差を示すグラフである。
【図７Ｃ】異なるサーバ利用レベルでの例示的な平均相対誤差を示すグラフである。
【図８Ａ】図7Aの平均相対誤差に対応する例示的な累積分布関数(CDF)を示す図である。
【図８Ｂ】図7Bの平均相対誤差に対応する例示的な累積分布関数(CDF)を示す図である。
【図８Ｃ】図7Cの平均相対誤差に対応する例示的な累積分布関数(CDF)を示す図である。
【図９】本開示の実装形態によって実行されうる例示的なステップを示す流れ図である。
【図１０】本開示の実装形態を実行するために使用されうる例示的なコンピュータシステムの略図である。
【発明を実施するための形態】
【００１７】
様々な図面における同様の参照記号は同様の要素を示している。
【００１８】
図1を特に参照すると、例示的なシステム100が示されている。図１の例示的システム100は、ネットワーク110を介して1つまたは複数のバックエンドサーバシステム106と通信する複数のクライアントコンピュータ102を含む企業資源計画(ERP)システムとして提供されうる。ネットワークは、ローカルエリアネットワーク(LAN)、ワイドエリアネットワーク(WAN)、インターネット、セルラー式ネットワーク、またはいくつもの数のモバイルクライアントとサーバとを接続するそれらの組合せなどの、大規模コンピュータネットワークとして提供されうる。いくつかの実装形態では、クライアント102はサーバシステム106に直接(たとえば、ネットワークを通じて接続せずに)接続されてよい。
【００１９】
クライアントコンピュータ102は、デスクトップコンピュータ、ラップトップコンピュータ、ハンドヘルドコンピュータ、携帯情報端末(PDA)、セルラー式電話、ネットワークアプライアンス、カメラ、スマートフォン、EGPRS(enhanced general packet radio service)移動電話、メディアプレイヤー、ナビゲーション装置、電子メール装置、ゲーム機、あるいはこれらのデータ処理装置または他のデータ処理装置のうちのいずれかの2つ以上の組合せを含む、様々な形式の処理装置を表すが、これらに限定されない。サーバシステム106は、アプリケーションサーバ112およびデータベース114を含み、ウェブサーバ、アプリケーションサーバ、プロキシサーバ、ネットワークサーバ、および/またはサーバファームを含む様々な形式のサーバを表すことを目的としているが、これらに限定されない。一般に、サーバシステム106はアプリケーションサービスを求めるユーザ要求を受け取って、ネットワーク110を介してこのようなサービスをいくつもの数のクライアント装置102にも提供する。いくつかの実装形態では、サーバシステム106は、サービスプロバイダがウェブサービスに関するデータを管理およびアクセスできる中心点を提供できる。
【００２０】
動作中、複数のクライアント102がネットワーク110を通じてサーバシステム106と通信できる。たとえばブラウザベースのアプリケーションなどのアプリケーションを実行するために、それぞれのクライアント102はサーバシステム106との対応するセッションを確立できる。それぞれのセッションは、サーバシステム106とそれぞれの個々のクライアント102との間の双方向情報交換を含むことができる。この双方向情報交換は、サーバシステム106に伝えられる、クライアント102で生成された要求を含むことができる。サーバシステム106は要求を受信し、複数の要求を待ち行列に入れ、要求に基づいて処理を実行し、要求しているクライアント102に応答を提供する。
【００２１】
本開示の実装形態は、図1を参照して上記で論じたシステム100などのコンピュータソフトウェアおよびハードウェアシステム上のサービス需要の正確な推定を対象にする。本開示の実装形態を示すためにERPシステムが使用されるが、本開示の適用性はERPシステムに

限定されない。より具体的には、本開示は、どのようなタイプの多段配列プログラムアプリケーション、ハードウェアのモデリング(たとえば、ディスクドライブまたはメモリ)、および/またはネットワークリンクも含む、どのようなコンピュータソフトウェアおよびハードウェアシステムにも適用できるが、これらに限定されない。

【００２２】

本開示の実装形態は、マルチクラスワークロードについてのサービス需要を推定するために、測定された要求の応答時間を使用する推定方法を提供する。応答時間は、要求を完了するための端から端までの時間として提供され、サービス時間(たとえば、その要求に関心が向けられている、またはサービスされている時間)およびリソース競合のための遅延(たとえば、バッファリングのための待ち時間)の両方を含む。ワークロード(すなわち要求のボディ)はいくつかのクラス(すなわちトランザクションのタイプ)を含むことができる。本開示の実装形態は、応答時間測定に基づいてクラスごとのサービス需要とも呼ばれるサービス時間の推定に取り組む。

【００２３】

開示された応答時間ベースの手法が改良された性能予測をもたらすことを示している本開示の実装形態を示すために、非限定的な例として実社会の産業ERPアプリケーションが使用される。応答時間はアプリケーションを実行することによってしばしば直接ログを取られる、および/または外部オブザーバによって得ることができるので、本開示の応答時間ベースの手法はより広範囲に適用可能である。さらに、利用データをサーバに公開しない、サービスプロバイダなどの第三者によって所有されるシステムにとって、応答時間は容易に入手可能であってよい。

【００２４】

本開示のサービス需要推定方法は、これに限定されないが、先着順(FCFS)スケジューリングを有するシステムで実装することができ、直線回帰推定手法および最尤推定手法に基づく。応答時間ベースの直線回帰は、要求の平均応答時間と要求がシステムに到着するときに見られる待ち時間長とを関連付ける正確な方程式を使用し、これらは両方ともアプリケーションログから容易に得ることができる。最尤手法は測定された応答時間の分布全体を考慮し、推定精度の向上を実現できる。本開示は、位相型分布を使用して、尤度値を求めるための、およびリソース消費を推定するための計算法を提供する。

【００２５】

この情報は、基本的なシステムのモデルを作る待ち行列モデルをパラメータ化するために使用される。本明細書で使用されるように、パラメータ化はモデル、この場合は待ち行列モデルの完全なまたは関連する仕様に必要なパラメータを決定および定義する処理を含む。待ち行列モデルは、実際の待ち行列システムに近似するモデルであり、定常状態性能測定を使用して待ち行列行動を解析できる。待ち行列モデルは、A/B/S/K/N/Discとして提供されるケンドールの記号を使用して表すことができ、上記でAは到着時間間隔分布であり、Bはサービス時間分布であり、Sはたとえばサーバの数であり、Kはシステム容量であり、Nは呼び人口であり、Discは仮定されるサービス規律である。いくつかの例では、K、N、およびDiscは省略され、表記法はA/B/Sになる。分布のための標準的な表記法Aおよび/またはBは、マルコフ(指数)分布を意味するM、k位相を有するアーラン分布を意味するEk、退化(または決定論的)分布を意味するD(定数)、一般分布を意味するG(任意)、および位相型分布を意味するPHを含む。

【００２６】

以下でより詳細に論じる計算結果は、利用ベースの手法と比較して応答時間ベースの手法の改良された精度を示している。さらに、本開示の応答時間ベースの手法の本質的な利点は、サーバでの応答時間がシステム内の要求によって招かれるすべてのレイテンシ低下に依存することである。したがって、応答時間は帯域幅、メモリ、および入力/出力(I/O)競合遅延を説明するので、応答時間は本質的により包括的な性能のディスクリプタである。本開示の応答時間ベースの手法は、そうでなければ従来の手法によって無視されてしまう遅延を容易に説明する。

【００２７】
次に図2を参照すると、簡略化された例示的なアプリケーションサーバシステム200が示されている。図2の簡略化されたアーキテクチャは、本開示の実装形態を示すために使用される非限定的な例として提示される。システム200はアプリケーションサーバシステム202およびワークロードジェネレータ204を含む。ワークロードジェネレータ204は、アプリケーションサーバシステム202に要求を転送する複数のクライアント装置（たとえばクライアント102）として機能し、アプリケーションサーバシステム202のリソースにサービス需要を出す。システム200は、SAP AG of Walldorf, Germanyによって提供されるSAP Business Suiteに含まれるものなどの1つまたは複数のアプリケーションプログラムのうちのERPアプリケーションを含むことができる。図2の例示的システム200は2層構造で提供され、共通の仮想マシン上にインストールされたアプリケーションサーバおよびデータベースサーバを含む。図2の例において、および本説明のために、他のどのような仮想マシンも物理的なマシン上で実行しない。しかし本開示はこのような構成に限定されないことを理解されたい。

【００２８】
システム200は、これに限定されないが、標準的なビジネストランザクション(たとえば販売注文および請求書の作成および/または一覧作成)を含みうる動作（たとえば販売および流通)のワークロードを使用してストレステストにかけられる。ワークロードジェネレータ204は、N個のユーザ206の固定グループによって要求が発行される、クローズドタイプのワークジェネレータでよい。ユーザによって提出された要求が完了すると、新しい要求をシステムに提出する前に、指数分布の思考時間(たとえば、平均時間Zは10秒に等しい)が終了する。すべてのユーザ206は要求の同じシーケンスをERPアプリケーションに循環的に提出する。要求の提出後、アプリケーションサーバは複数の並行ユーザ206からの要求を処理する。この処理はデータベース208から抽出した情報を使用して実現できる。アプリケーションサーバは、待ち行列内の要求をバッファしてその要求を作業過程212に転送する、ディスパッチャ210を使用する。作業過程212は、要求を並行して供給する個別のオペレーティングシステムスレッドとして提供されうる。作業過程212は、対話型プログラムを実行する対話処理として実行されてもよく、データベース変更を実行するアップデート処理として実行されてもよい。

【００２９】
要求のリソース消費を推定するために測定された応答時間（R_{MEAS}）が使用されれば、ERPアプリケーション（たとえば図2の）の性能を説明する基本的な待ち行列モデルのパラメータ化が著しく改良される。このタスクのために考慮された性能モデルは、これに限定されないが、基本的なM/M/1//Nモデル300を含み、その例示的な構造は図3に示されている。図3の例示的モデル300は、遅延サーバ302、ならびにウェイティングバッファ306、作業過程およびデータベースシステム308を含むERPシステム304を含む。M/M/1//Nモデル300は、N個のユーザの有限母集団によって生成された要求を有するM/M/1/待ち行列である。これは、ユーザの思考時間のモデルを作る遅延ステーション、続いてクローズドループ上の待ち行列ステーションで構成されるクローズド待ち行列ネットワークとしても見ることができる。ウェイティングバッファ306はERPシステム304における流入制御を表し、作業過程において実行されるとサーバは要求のリソース消費のモデルを作る。

【００３０】
データベースサーバの利用は5%未満であるのが一般的である。したがってデータベース層での待ち行列の影響はごくわずかであり、データベースから暫定的データへの合計時間

【００３１】
【数１】

$$(T_{MEAS}^{DB})$$

【００３２】

は作業過程におけるサービス需要の構成要素として使用されうる。待ち行列モデルにおける思考時間およびサービス需要の両方が指数分布されると仮定されうる。この仮定は、ERPシステムにおける実際のサービス需要分布の良い近似として検証されている。測定されたサービス需要(すなわち標準偏差と平均との比率)の変動係数(CV)はCV=[1:02;1:35]における異なるバリデーションに値域を定めるのに対し、理論的な指数はCV=1である。モデル仮定と一致して、バリデーションについてのすべての思考時間も指数分布を使用して生成されている。

【００３３】

待ち行列モデルをパラメータ化する場合、基本的な待ち行列モデルが初めに決定される。基本的な待ち行列モデルは、コンピュータシステム（たとえばCPU）のリソースからの要求の発着を表す確率論的モデルである。基本的な待ち行列モデルは、リソースモデル化によって供給される要求の性能に影響を及ぼす現象を捉えるためにアナリストの抽象的概念として提供される。コンピュータシステムを説明するためにこのような待ち行列モデルについて定義される特徴は、リソーススケジューリング規律およびそれらのサービス時間分布、ならびに他のリソースとの相互接続および/またはそれぞれのリソースでの要求到着の統計的特徴を含む。この情報に基づいて、たとえば標準的なマルコフ連鎖理論を使用して基本的な待ち行列モデルが提供されうる。その次に、基本的な待ち行列モデルおよび測定された応答時間を処理することによって決定されうる推定された平均サービス需要に基づいて、最後のまたは対象の待ち行列モデルがパラメータ化されうる。本開示において提供される平均サービス需要の推定によって、基本的な待ち行列モデルにおける1つまたは複数のリソースモデル化についてのサービス時間分布のパラメータ化ができるようになる。

【００３４】

モデルによって予測された応答時間が、すべての可能な数のユーザNのための実際のシステムの測定された応答時間(R_{MEAS})に正確に一致するように、サーバでの要求の平均サービス需要(E[D])が決定される。軽負荷と重負荷とでERPシステムの動作に非常に異なって影響を及ぼすキャッシングの大きな役割のため、サービス需要はモデルにおけるユーザNの数の関数として指定される(すなわちE[D]≡E[D](N))。この推定手法は複雑なソフトウェアシステムのモデルを作る際に通常使用される。

【００３５】

次に図4を参照すると、本開示は応答時間測定(R_{MEAS})に基づいて待ち行列モデルの直接パラメータ化を提供する。より具体的には、測定された応答時間の分布またはモーメントに最もよく一致するようにサービス需要が推定される。これは図4に示されるようなモデリング方法論における重要なパラダイム変化を必要とする。図4のモデリング方法論はデータ収集400、モデリング仮定(たとえばスケジューリングおよびサービス分布)402、需要推定404、ならびにモデル生成およびソリューション406を含む。モデルのソリューションは、平均応答時間、平均スループット、およびランダムな時点で待ち行列内の一定数の要求を観測する確率についての値によって提供されうる。

【００３６】

応答時間はスケジューリングまたはサービス需要分布に関連するいくつかのシステムプロパティに依存するので、スケジュール上の仮定または一般的な形式の分布は、サービス需要推定活動の開始より前に行われる。仮定は、スケジューリングのタイプ（たとえばFCFSスケジューリング）、モデル化されるべきワークロードクラスの数、およびそれぞれの要求のサービス需要の分布(たとえば、指数分布)を含むことができるが、これらに限定されない。したがって、返されたサービス需要は、それらが使用されることになるモデルの特徴に依存する。対象のモデルに関連する可能な限り高いサービス需要推定を決定するために、最後のモデルの特徴の予備仮定が提供される。モデリング活動の目的は、実験的観測とモデル予測との間のよい合致を得ることなので、本開示の応答時間ベースの手法はモデル仮定の下で最高のパラメータを返すことによってこの目的を実現することにより強い焦点を置く。

【００３７】
上記で論じたように、例示的ERPシステムは指数分布されたサービス需要を有するFCFS待ち行列としてモデル化されうる。たとえば、指数分布はイベントが一定の平均レートで連続的および単独で発生する処理におけるイベント間の時間を説明できる。これらの仮定の下で、n個のウェイティング要求およびサービス中の1つの要求を有するシステムを発見する要求は、以下のように確率変数の合計の分布によって与えられる応答時間分布を受信する:

$R=T+D^1+D^2+...+D^n+D^{n+1}$ (1)

上式で、Tはサービス中の要求の完了前の残余時間であり、1≦i≦nであるD^iはi番目に待ち行列に入れられた要求のサービス需要であり、D^{n+1}は新たに到着した要求のサービス需要である。指数分布の定義によって、TはDと等しく、したがって確率変数Rの分布はn+1個の指数確率変数のコンボリューションである点に留意されたい。

【００３８】
平均応答時間が以下の関係を満たすのがわかっていることを、単純な微分によって示すことができる。

$E[R]=E[D](1+E[A])$ (2)

上式でE[D]は平均サービス需要であり、E[A]は到着時に要求によって観測される平均待ち行列長である。式2は、新しい要求の到着時にERPシステムにおける要求の平均数の知識がある場合は要求の平均サービス需要を推定するために使用されうる。値Aは、実行された実験のログファイルから得ることができ、要求の発着時間を報告する。

【００３９】
【数２】

$$E[D]\left(D_{EST}^{RSP}\right)$$

【００４０】
の推定は、X連続要求（たとえば、X=20）のグループについてR_{MEAS}および測定されたA値を平均化することによって得られたE[R]およびE[A]のサンプルのシーケンスを使用して式2の直線回帰によって得ることができる。ERPシステムにおけるE[R]とE[A]との間の測定された直線関係の例示的なグラフィカルな説明は図5で提供される。

【００４１】
式2より、直線回帰によってE[D]の値を求めることができる:

【００４２】
【数３】

$$E[R_{MEAS}^i]=E[D](1+E[A_{MEAS}^i]) \qquad (3)$$

【００４３】
上式で、指数iは測定活動において収集されたi番目の値を意味し、

【００４４】
【数４】

$$E[D_{MEAS}^i]$$

【００４５】
および

【００４６】

【数５】

$$E[A^{i}_{MEAS}]$$

【００４７】
はそれぞれX連続iサンプル（たとえばX=20）で平均される応答時間および到着待ち時間長である。
【００４８】
上述の式を基礎として使用して、本開示は直線回帰技法および最尤技法に基づいてマルチクラス要求のためのサービス需要推定アルゴリズムを提供し、それぞれについては例示的な待ち行列シナリオを使用して以下でさらに詳細に説明する。例示的な説明のためだけに、ユーザが合計I個のサンプルの入力トレースをサービス需要推定アルゴリズムに提供したと仮定する。これは、これらに限定されないが、クラスcの要求についての測定された応答時間(R_c)のシーケンス、およびそれぞれのクラスc要求によって到着時に見られる対応するクラスk要求の待ち行列長
【００４９】
【数６】

$$(A^{c}_{k})$$

【００５０】
を含むことができる。ワークロードクラスの数はKで表される。
【００５１】
ある実装形態では、指数サービスの仮定が廃棄されるか複数のクラスが考慮される場合、式3で与えられた直線関係は変更しない。したがって、複数のクラスを有する非生産形式待ち行列ネットワークにおけるFCFS待ち行列を推定するために導出された数式の変形が以下のように提供される:
【００５２】
【数７】

$$E[R_c] = E[T_c] + \sum_{k=1}^{k} E[D_k]\left(1_{k,c} + E[A^{c}_{k}]\right) \qquad (4)$$

【００５３】
上式で$1_{k,c}$は、k=cの場合1に等しく、k≠cの場合は0に等しい。T_cは、クラスc要求の到着の瞬間に実行中である要求の完了前の残余時間であり、Kは要求クラスの数であり、E[Dk]はクラスkの平均サービス需要であり、
【００５４】
【数８】

$$E[A^{c}_{k}]$$

【００５５】
は新たに到着した要求と現在サービス中の要求の両方を除くシステムにおいて待ち行列に入っているクラスkの要求の平均数である。
【００５６】
式4は、たとえばGI/GI/1/FCFS待ち行列におけるクラスcの要求の平均要求時間を提供できる。GIは一般的な独立分布を示しており、要求が相互に独立している特定のタイプの一般分布である。FCFSスケジューリングを有するGI/GI/1待ち行列の例示的な場合では、システムがn+1個の待ち行列に入っている要求（すなわちウェイティング中のn個の要求とサ

ービス中の要求）を有するときに到着するクラスcの要求の応答時間を説明する確率変数が以下のように提供される：

$R_c=T_c+D^1+D^2+\cdots+D^n+D_c$ (5)

上式で、T_cはサービス中の現在の要求の完了前の残余時間であり、$1\leqq i\leqq n$であるD^iはi番目に待ち行列に入っている要求のサービス需要であり、D_cは新たに到着した要求のサービス需要である。

【００５７】

到着待ち行列長が$A^c=n$であることが条件とされている予想をすると：

【００５８】

【数９】

$$E[R_c \mid A^c=n]=E[T_c \mid A^c=n]+\sum_{i=1}^{n}E[D^i \mid A^c=n]+E[D_c] \qquad (6)$$

【００５９】

上式で、C(i)は位置i(1≦i≦n)における要求のクラスである。

【００６０】

要求クラスの関数としてのサービス需要は以下のように表すことができる：

【００６１】

【数１０】

$$E[D^i \mid A^c=n]=\sum_{k=1}^{K}E[D_k]\mathbb{P}[C(i)=k \mid A^c=n] \qquad (7)$$

【００６２】

式7を式6に代入すると：

【００６３】

【数１１】

$$E[R_c \mid A^c=n]=E[T_c \mid A^c=n]+\sum_{i=1}^{n}\sum_{k=1}^{K}\mathbb{P}[C(i)=k \mid A^c=n]E[D_k] \qquad (8)$$

【００６４】

続いて平均応答時間は以下に基づいて決定されうる：

【００６５】

【数１２】

$$\begin{aligned}E[R_c]&=\sum_{n=1}^{+\infty}E[R_c \mid A^c=n]\mathbb{P}[A^c=n]\\&=E[T_c]+\sum_{k=1}^{K}E[D_k]\sum_{n=1}^{+\infty}\left(1_{k,c}+\sum_{i=1}^{n}\mathrm{P}[C(i)=k \mid A^c=n]\mathbb{P}[A^c=n]\right)\\&=E[T_c]+\sum_{k=1}^{K}E[D_k]\left(1_{k,c}+\sum_{n=1}^{+\infty}E[A_k^c \mid A^c=n]\mathbb{P}[A^c=n]\right)\\&=E[T_c]+\sum_{k=1}^{K}E[D_k]\left(1_{k,c}+E[A_k^c]\right)\end{aligned} \qquad (9)$$

【００６６】

上式で、

【００６７】

【数１３】

$$\sum_{i=1}^{n} \mathrm{P}\left[C(i) = k \mid A^{c} = n\right]$$

【００６８】

は、クラスcの要求による到着時に待ち行列内でウェイティング中のクラスk要求の平均数

【００６９】

【数１４】

$$E\left[A_k^c \mid A^c = n\right]$$

【００７０】

である。

【００７１】

クラス需要$E[D_k]$の推定は、非負最小二乗アルゴリズム(たとえば、Mathworks, Inc.によってMATLABにおいて提供されるlsgnonneg)などの回帰法を使用して式4から得ることができ、推定上の信頼区間は標準的な数式を使用して生成されうる。式4は、クラスごとの応答時間R_cおよび到着待ち行列長A_cが知られている場合、平均サービス需要$E[D_k]$を推定するための直線回帰において使用されうる。一般に、$E[T_c]$を直接測定するのは難しい場合がある。したがって、この量を推定するために近似が必要である。これを実現するために、これに限定されないが、標準的なポアソン到着近似$T_c=D_k$を含むことができる近似スキームが実装されうる。

【００７２】

例示的なGI/GI/1/FCFS待ち行列内に残余時間数式(T_c)についての正確な結果がないため、以下の残余時間についての再生理論数式が使用される:

【００７３】

【数１５】

$$E[T_c] = \frac{E[D_c]}{2}\left(1 + CV_c^2\right) \qquad (10)$$

【００７４】

上式で、CV_cはクラスcのサービス需要分布の変動係数である。これは明らかにGI/GI/1/FCFS待ち行列における近似であり、到着ストリームがポアソン過程の場合、正確な数式になる。CV_cの値が先験的に知られていない場合、研究中のシステムに適していると仮定される異なるCV_c値についての式4の値を求めることはまだ可能である。式4の最小二乗解法において最小平均残余誤差を生じさせる、したがって観測に最適である変形係数のグループであるセットがCV_c値の最良の推定値として選択される。これは、非指数サービス分布のための式4の可能なアプリケーションについての一般的なスキーマを概説する。

【００７５】

いくつかの実装形態では、一定のサンプルパスを観測する確率の解析式に基づいて確率変数の統計を推論するために最尤推定が使用されうる。平均サービス需要推定の範囲内で、すべての$1 \leqq i \leqq I$についてRiはi番目に観測または測定された応答時間を示させることによって最大尤度を公式化でき、上式でIは測定された応答時間の総数である。測定された応答時間のシーケンスを観測する確率$R^1,\cdots,Ri,\cdots,R^I$が最大になるように、それぞれのクラス$1 \leqq k \leqq K$平均サービス需要$E[D_k]$が求められる。公式にこれは平均サービス需要のセット$E[D_1],\cdots,E[D_k]$を発見することとして表すことができる。これは以下の最大化問題を解決する:

【００７６】
【数１６】

$$\max_{E[D_1],\ldots,E[D_k]} \mathrm{P}\left[R^1,\ldots,R^I,\ldots,R^I \mid E[D_1],\ldots,E[D_K]\right] \qquad (11)$$

【００７７】
すべてのクラスk について、$E[D_k] \geqq 0$であることを条件とする。応答時間を独立確率変数と仮定すると、式11における同時確率の数式は以下の積に簡約される:
【００７８】
【数１７】

$$\max_{E[D_1],\ldots,E[D_k]} \prod_{i=1}^{I} \mathrm{P}\left[R^i \mid E[D_1],\ldots,E[D_K]\right] \qquad (12)$$

【００７９】
積を和として同等に表現するために対数を取ると以下が提供される:
【００８０】
【数１８】

$$\max_{E[D_1],\ldots,E[D_k]} L\left(E[D_1],\ldots E[D_K]\right) \qquad (13)$$

【００８１】
上式で、ここでは独立変数は以下のように提供される尤度関数である:
【００８２】
【数１９】

$$L\left(E[D_1],\ldots E[D_K]\right) = \sum_{i=1}^{I} \log P\left[R^i \mid E[D_1],\ldots E[D_K]\right] \qquad (14)$$

【００８３】
課題は研究中の問題を表す尤度関数についての、および最大値を効率的に計算できる数式を得ることである。最適化はコストがかかりすぎる場合があるので、この数式は解析的に扱いやすいことも重要である。特に、平均サービス需要の推定は積形式待ち行列に基づく標準的な容量立案モデルの仕様についての著しいパラメータなので、本明細書は平均サービス需要の推定に焦点を合わせる。
【００８４】
いくつかの実装形態では、尤度関数を導くための1つの方法は、サブジェクトシステムのサービスおよび応答時間の両方を、それらの評価についての効率的な数式を享受する位相型分布によって特徴付ける。不要な複雑性を避けるために、指数分布のサービス需要の場合の手法が説明され、一般的なサービス需要に関するこれの外延が提供される。
【００８５】
たとえば、現在サービス中の要求を含む、$n_1,\cdots,n_k$要求がクラスごとに待ち行列に入っているときにシステムに到着する要求が考慮される。すべてのクラスについて指数サービス需要を仮定でき、クラスkの平均サービス率が$\mu_k=1/E[D_k]$で示される。k番目の要求についての応答時間の分布は、レートμ1でn1指数サービス需要の合計によって与えられ、レートμ2でn2指数サービス需要の合計によって与えられ、すべてのKクラスについて同様である。したがって、これはマルコフ連鎖における吸収時間として$n=n_1+\cdots+n_k$状態でモデル化でき、それぞれが要求サービスのための待ち時間を表している。非限定的な例示とし

て、K=2クラスがあり、到着時に見られる待ち行列がn_1=3およびn_2=2の場合、両方の到着した要求およびサービス中の1つの要求を含めて、以下の(D_0, D_1)表現で位相型分布によって吸収時間を提供できる:

【００８６】

【数２０】

$$D_0 = \begin{bmatrix} -\mu_1 & \mu_1 & 0 & 0 & 0 \\ 0 & -\mu_1 & \mu_1 & 0 & 0 \\ 0 & 0 & -\mu_1 & \mu_1 & 0 \\ 0 & 0 & 0 & -\mu_2 & \mu_2 \\ 0 & 0 & 0 & 0 & -\mu_2 \end{bmatrix} \quad (15)$$

$$D_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \mu_2 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (16)$$

【００８７】

D_0マトリクスは、吸収につながらない状態iから状態jへの遷移を表す(i, j)の位置に非対角要素を有し、D_1要素は吸収に関連する遷移である。D_0の対角はこのようなので、D_0プラスD_1は、経時的なアクティブ状態の進化を説明するマルコフ連鎖の無限小生成作用素である。

【００８８】

(D_0, D_1)表現により、要求が応答時間Riを受信する確率が効率的に決定できるようになる。吸収するための時間は位相型分布(D_0, D_1)の確率密度によって説明される。吸収マルコフ連鎖の基本的理論から、これは以下のように容易に提供される:

【００８９】

【数２１】

$$P\left[R^i \mid E[D_1],...,E[D_K]\right] = \vec{\pi}_e e^{D_0 R^i}\left(D_1\right)\vec{e} \quad (17)$$

【００９０】

上式で

【００９１】

【数２２】

$$\vec{e} = (1, 1, ..., 1)^T,\ \vec{\pi}_e = \vec{\pi}_e\ (-D_0)^{-1}D_1$$

【００９２】

は、吸収直後の要求の元の状態を表す要素

【００９３】

【数２３】

$$(\vec{\pi}_e)_j$$

【００９４】
を有する行ベクトルであり、$E[D_1],\cdots,E[D_k]$の知識はレート$\mu_1,\cdots,\mu_k$を知っていることに等しい。D_1内に吸収につながる単一の遷移があるため、式17において、常に
【００９５】
【数２４】

$$\vec{\pi}_e = (1, 0, ..., 0)$$

【００９６】
であり、式17を評価することの主要コストはマトリクス指数の計算であるとすぐに結論付けられる。これは、一意化技法を使用して、またはパデ展開(たとえばMATLABで提供される)によって、効率的な方法で推定することができる。上述の手法は、位相型として推定することができる非指数サービス需要に一般化されうる。
【００９７】
この概念は
【００９８】
【数２５】

$$(S_0^k, S_1^k)$$

【００９９】
にクラスkのサービス需要の位相型表現を示させることによって説明されうる。非限定的な例示として、アーラン-2分布について、これは以下のように提供される:
【０１００】
【数２６】

$$S_0^k = \begin{bmatrix} -\mu_k & \mu_k \\ 0 & -\mu_k \end{bmatrix}$$

【０１０１】
および
【０１０２】
【数２７】

$$S_1^k = \begin{bmatrix} 0 & 0 \\ \mu_k & 0 \end{bmatrix}$$

【０１０３】
これは確率ベクトル
【０１０４】

【数２８】

$$\vec{\pi}_e^k = (1, 0)$$

【０１０５】
を暗示する。
【０１０６】
指数ケースの同じ例として、およびサービスが例示的なアーラン-2分布を使用して分布されると、待ち行列内で費やされる応答時間は以下のように表現できる:
【０１０７】
【数２９】

$$D_0 = \begin{bmatrix} -S_0^1 & S_1^1 & 0 & 0 & 0 \\ 0 & -S_0^1 & S_1^1 & 0 & 0 \\ 0 & 0 & -S_0^1 & S_1^1 \vec{e}\,\vec{\pi}_e^2 & 0 \\ 0 & 0 & 0 & -S_0^2 & S_1^2 \\ 0 & 0 & 0 & 0 & -S_0^2 \end{bmatrix} \qquad (18)$$

$$D_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ S_1^2 \vec{e}\,\vec{\pi}_e^1 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (19)$$

【０１０８】
上式で
【０１０９】
【数３０】

$$\vec{\pi}_e^k = \vec{\pi}_e^k \left(-S_0^k\right)^{-1} S_1^k$$

【０１１０】
である。一般的に
【０１１１】
【数３１】

$$(S_0^1, S_1^1)$$

【０１１２】
と
【０１１３】
【数３２】

$$(S_0^2, S_1^2)$$

【０１１４】
の状態空間サイズは異なるので、

【０１１５】
【数３３】

$$\vec{e}\vec{\pi}_e^k$$

【０１１６】
項は位相型分布の適切な初期化を保証する。上述の数式は、尤度関数を計算するために式17を使用してすぐに値を求めることができる。唯一の違いは、これはより大きなマトリクスを含むことである。
【０１１７】
いくつかのパラメータによって指定された超指数分布などのサービス需要分布について、上述の手法は、平均を除くすべてのモーメントが先験的に固定されることか(たとえば、サービス処理の変動性を推測することによって)、これらの追加パラメータを未知の変数として最尤処理に統合することのどちらかを必要とする。これは、最初のMモーメント
【０１１８】
【数３４】

$$E[D_k^M]$$

【０１１９】
、1≦m≦Mによって位相型分布が一意に指定される場合、ログ
【０１２０】
【数３５】

$$\log P\left[R' \mid E[D_k], E[D_k^2],..., E[D_k^M], 1 \le k \le K\right] \qquad (20)$$

【０１２１】
の形式の条件付き確率の値を求めるために最尤推定問題の構造を変化させる。たとえば、3つのパラメータによって超指数分布が完全に決定されうる。したがって、最初の3つのモーメント
【０１２２】
【数３６】

$$E[D_k^M]$$

【０１２３】
は分布を完全に指定するのに十分である。
【０１２４】
扱うのが最も難しい高変動性分布がある場合、増加する数のクラスにおけるワークロードを分解するのがより実用的である。このように、それぞれのクラスのサービス分布を指数によって推定することができる。たとえば、ワークロードクラスが密度
【０１２５】
【数３７】

$$f(t) = p\mu_1 e^{-\mu_1 t} + (1-p)\mu_2 e^{-\mu_2 t}$$

【０１２６】
である超指数サービス需要を有し、到着レートがλの場合、ワークロードは指数サービス需要が
【０１２７】

【数３８】

$$f_1(t) = \mu_1 e^{-\mu_1 t}$$

【０１２８】
および
【０１２９】
【数３９】

$$f_2(t) = \mu_2 e^{-\mu_2 t}$$

【０１３０】
である2つのクラス、ならびに到着レート$\lambda_1 = p\lambda$および$\lambda_2 = (1-p)\lambda$によってそれぞれ推定することができる。これにより、より単純な指数仮定の下で焦点を平均サービス需要推定に合わせることができるようになる。したがって、サービス需要解析の肝心な点は、マルチクラス設定における指数分布サービス需要を確実に推定できることである。
【０１３１】
次に図6Aおよび6Bを参照すると、従来のCPU利用ベースの手法と比較した応答時間ベースの手法の精度が強調されている。図6Aは、本開示の応答時間ベースの手法を使用してそれぞれパラメータ化された例示的なM/M/1//N待ち行列モデルについての推定およびモデル予測精度結果を提供する。図6Bは、CPU利用ベースの手法を使用してそれぞれパラメータ化された例示的なM/M/1//N待ち行列モデルについての推定およびモデル予測精度結果を提供する。図6Bは、CPU利用ベースの手法に従ってパラメータ化されたモデルが、基本的なERPシステムの測定された応答時間(R_{MEAS})を著しく低く推定する応答時間推定(R_{CPU})を提供することを示している。この場合、完了した要求の数に対するそれぞれの要求の総CPU消費
【０１３２】
【数４０】

$$(T_{MEAS}^{CPU})$$

【０１３３】
と、データベースからのデータを使えるように設定するための総時間
【０１３４】
【数４１】

$$(T_{MEAS}^{DB})$$

【０１３５】
との合計の直線回帰によって決定される、
【０１３６】
【数４２】

$$D_{EST}^{CPU}$$

【０１３７】
と等しいE[D]でパラメータ化されている例示的待ち行列モデルに基づいてR_{CPU}値が決定される。
【０１３８】
図6Bは、CPU利用ベースの手法に従ってパラメータ化されたモデルが、基本的なERPシステムの測定された応答時間(R_{MEAS})を著しく高く推定する応答時間推定(R_{TWP})を提供することも示している。この場合、完了された要求の数に対してワーク処理において費やされた合計時間
【０１３９】
【数４３】

$$(T_{MEAS}^{TWP})$$

【０１４０】
の直線回帰によって決定される、
【０１４１】
【数４４】

$$D_{EST}^{TWP}$$

【０１４２】
と等しいE[D]でパラメータ化されている例示的待ち行列モデルに基づいてR_{TWP}値が決定される。図6Aに見られるように、図6Bで考慮される利用ベースのサービス需要推定と比較して、質の高い予測の生産において本開示の応答時間ベースの手法の方がより効率的である。
【０１４３】
本開示による応答時間ベースのサービス需要推定の精度は、たとえば60、600、および3600秒のシミュレーション期間を使用して待ち行列システムシミュレータによって生成される利用および応答時間トレースを参照して図示することもできる。このようなシミュレータは数値計算環境およびプログラミング言語(たとえば、Mathworks, Inc.によって提供されるMATLAB)を使用して書くことができ、標準的な性能測定ならびに要求によって到着時に見られる待ち行列長のログを取るためにプログラムできる。到着時間間隔および平均サービス需要は指数分布から生成でき、シミュレーションについて固定された利用レベルを一致させるためにそれらの平均が割り振られる。
【０１４４】
シミュレーションは、600秒の例示的期間にわたるシステム利用および応答時間測定に基づき、低、中、および高サーバ利用レベル(たとえば、$\rho \in \{0.1,0.5,0.9\}$)ならびに要求クラス$K \in \{1,2,5\}$で実行されている。応答時間測定は要求ごとに収集でき、利用は毎秒サンプリングできる。これは、容易に個々の要求の応答時間のログを取るが、きめ細かい、または要求ごとの解決で利用を調べられない、現代のシステムにおける一般的な状況である。需要推定精度は従来のCPU利用ベースの手法(UR)と比較され、本開示による方法は応答時間ベースの回帰(RR)および最大尤度(ML)を含む。
【０１４５】
それぞれのシミュレーションについて、クラスKの数および利用ρの異なる選択に対応して、以下によって提供される誤差関数に基づいて推定精度を評価することができる:
【０１４６】

【数４５】

$$\Delta = \sum_{k=1}^{K} \frac{1}{K} \left| \frac{E[D_{EST,k}] - E[D_{EXACT,k}]}{E[D_{EXACT,k}]} \right| \qquad (21)$$

【０１４７】

これはシミュレーションにおいて使用される正確な値$E[D_{EXACT,k}]$について推定されたサービス需要$E[D_{EST,k}]$のすべてのクラスのにわたる平均相対誤差である。シミュレーションごとに、それぞれのクラスの$E[D_{EXACT,k}]$値が[0,1]の範囲の一様分布でランダムに導かれる。

【０１４８】

図7A～7Cは、ランダムサービス需要で100のシミュレーションにわたってΔを平均化することによって決定された平均値E[Δ]を示している、ならびに同数のクラスKおよびサーバ利用ρを有している例示的なグラフを含む。図7Aは、単一の要求クラスに対応するグラフであり、サーバ利用レベル0.1、0.5、および0.9についての平均相対誤差値を示している。図7Bは2つの要求クラスに対応するグラフであり、サーバ利用レベル0.1、0.5、および0.9についての平均相対誤差値を示しており、図7Cは5つの要求クラスに対応するグラフであり、同様にサーバ利用レベル0.1、0.5、および0.9についての平均相対誤差値を示している。図7A～7Cにおいて、UR-1はすべての利用可能なサンプル上のCPU利用ベースの回帰についての平均相対誤差値を示しており、UR-50sは50個の連続サンプルにわたって測定を平均化しているCPU利用ベースの回帰についての平均相対誤差値を示しており、UR-30 reqは30個の要求がそれぞれの分割内に入るように30個の連続サンプルによって形成される分割上の平均を使用してCPU利用ベースの回帰についての平均相対誤差値を示している。RR-1 reqはすべての利用可能サンプルを使用して応答時間ベースの回帰についての平均相対誤差値を示しており、RR-30 reqは同じクラスの30個の要求の分割にわたる平均を使用する。MLは、すべての利用可能なサンプルを使用して応答時間ベースの最大尤度についての平均相対誤差値を示している。

【０１４９】

図8A～8Cは、ρ=0.5およびK=1、2、5についての100個のランダムモデルにわたるΔの累積分布関数(CDF)を示す例示的なグラフを含む。図8A～8Cは、作図を簡略化するために、および図示していないカーブは図7A～7Cと一致する定性的結論につながるので、UR-30 req、RR-30 req、およびMLについてのCDFだけを含む。

【０１５０】

単一の平均サービス需要E[D]だけが推定されるので、単一クラスのシナリオが最も単純なシナリオである。このケースではほとんどの方法がうまく実行され、異なる利用レベルについて誤差約5%を示す(図7A参照)。UR-1だけが、中負荷(ρ=0.5)でさえ30%を上回る誤差値を特徴とする質の低い結果をもたらす。この影響は、サンプル期間内の不十分な数のイベントの観点から説明でき、基数のうちのわずかだけを有する値のサブセットでサンプル平均が計算されるので(たとえば、1、2、または3つの要求でも)、直線回帰式を無効にする。したがって、UR-30 reqおよびUR-50 sにおいて使用される集約は、UR-1 sの誤差を取り除いてUR-30 req のためにE[Δ]を5%未満に下げることに極めて好ましい影響を与える。さらにRR法はすべての利用レベルに質の高い結果をもたらす。MLも同様にすべての利用レベルに質の高い結果をもたらす。図7Aの観測が誤差分布によって確認される。より具体的には、図8Aはすべての技法について、多くの方法に低い高誤差率を12%の約95パーセンタイルでもたらす。したがって、単一クラスのケースではUR-1 sを除く技法は正確な結果をもたらす。

【０１５１】

単一からマルチクラスワークロードに移動する場合、図7Bに示されるように誤差が増える。2つの要求クラスを考慮する場合、MLは最小誤差を有し、中負荷の状況（ρ=0.5)で8%未満の誤差をもたらす。UR-1 sの質と比較してML性能の質が強調され、同じ利用レベルに

ついて約35%の誤差値が見られる。利用サンプルの集約は単一クラスのシナリオと同じくらい効果的であるとは判明せず、UR-30 reqならびにUR-50 sはUR-1 reqとはそれほど異ならない。RR-1 reqはUR法よりもよい結果をもたらす。分布解析は、ρ=0.5について同様の結果を提示する。たとえば図8Bで、MLは17%の95パーセンタイルを有し、RR-30 reqおよびUR-30 reqはそれぞれ54%および61%の95パーセンタイルを有する。

【０１５２】

図7Cは、5クラスのシナリオに移動すると結果の質が著しく変更するという事実を示している。より具体的には、特に高負荷において一般的な精度が低下する（すべての図面について縦軸が異なる点に留意されたい）。これは、推定されるべき多数のサービス需要の結果と推測される。UR-1 s、UR-30 req、およびUR-50 sは一貫して大きな誤差を有し、UR法の間の最良の結果はρ=0.5のケースにおけるUR-1 sによって実現されている。URについての集約は、やはり精度の向上にはつながらない。RR-1 reqは、中程度の利用レベルにおいて同様の結果をもたらし、RR-30 reqは誤差値を悪化させる。少なくとも低および中程度の利用レベルで質の高い結果をもたらす唯一の方法はMLである(12%未満の誤差)。すべての方法にとって負荷の重いケースは困難に見える。図8CにおけるK=5のシナリオのCDFは、21%未満の誤差の生産の95%を実現するMLの質を強調する。RR-30 reqは111%の95パーセンタイルを有し、UR-30 reqについては162%である。UR-1 sについては(図7Cには示せず)56%である。したがって、総合的にMLがより堅牢である。

【０１５３】

図7A～7Cおよび8A～8Cでもたらされる例示的な結果は、異なる利用値についての600秒内のイベントのセットに対応する中程度のサイズのデータセット内のマルチクラス推定は困難な課題であることを示している。MLはこのような推定をほとんどすべての場合堅牢に実行する。MLの結果は、限定された情報(たとえば低利用)で極めて良い。すべての実験においてRR-1 reqはRR-30 reqよりも効果的に見え、低および中負荷において一般的にUR法に対抗する。

【０１５４】

次に図９を参照すると、本開示によって実行されうる例示的なステップを流れ図が示している。ステップ902でワークロードが生成され、ワークロードはコンピュータシステムによって処理されうる複数のクラスを含む複数のサービス要求を含む。コンピュータシステムは、アプリケーションを実行するアプリケーションサーバ、および要求を生成する1つまたは複数のクライアントシステムを含むことができる。ステップ904でコンピュータシステムにワークロードが適用され、ステップ906でワークロードの要求ごとに応答時間が測定される。ステップ908で、応答時間および基本的な待ち行列モデルに基づいてクラスごとに平均サービス需要が推定される。基本的な待ち行列モデルは、最後の待ち行列モデルの特徴の仮定を含むことができ、その特徴はスケジューリングおよびサービス需要分布のうちの少なくとも1つを備える。

【０１５５】

平均サービス需要を推定するステップは、測定された応答時間に基づいて直線回帰および最尤法解析のうちの1つを使用して基本的な待ち行列モデルにおける平均サービス需要を推定するステップを含むことができる。平均サービス需要を推定するステップは、複数の到着待ち行列長にさらに基づくことができ、その複数の到着待ち行列長はワークロードのそれぞれの要求に対応する。複数の到着待ち行列長のそれぞれの到着待ち行列長は、要求の発着時間を報告するログファイルから決定されうる。平均サービス需要を推定するステップは、複数の残余時間にさらに基づくことができる。複数の残余時間は、ワークロードのそれぞれの要求に対応できる。それぞれの残余時間は、処理されるべき要求の到着時に作業中の要求の処理を完了するために残っている時間に対応できる。

【０１５６】

ステップ910で、平均サービス需要およびワークロードの特徴に基づいて待ち行列モデルが生成される。ワークロードの特徴は、1秒あたり平均5つの要求を有する指数到着時間間隔などの、待ち行列内に到着する要求の到着時間間隔分布を含むことができるが、これ

に限定されない。待ち行列モデルを生成するステップは、平均サービス需要を使用して基本的な待ち行列モデルをパラメータ化するステップを含むことができる。ステップ912で、複数の入力を使用して待ち行列モデルを処理することによって、コンピュータシステムの性能が評価される。

【０１５７】

次に図10を参照すると、本開示の実装を実行するために使用されうる例示的なハードウェアコンポーネント1000の略図が提供されている。システム1000は、本明細書に記載された方法に関連して記載される動作のために使用されうる。たとえば、システム1000はアプリケーションサーバシステム106内に含まれてよい。システム1000は、プロセッサ1010、メモリ1020、記憶装置1030、および入力/出力装置1040を含む。それぞれのコンポーネント1010、1020、1030、および1040はシステムバス1050を使用して相互接続されている。プロセッサ1010はシステム1000内で実行するための命令を処理できる。ある実装形態では、プロセッサ1010はシングルスレッドプロセッサである。他の実装形態では、プロセッサ1010はマルチスレッドプロセッサである。プロセッサ1010は、ユーザインターフェースのためのグラフィック情報を入力/出力装置1040上に表示するためにメモリ1020または記憶装置1030内に格納された命令を処理できる。

【０１５８】

メモリ1020はシステム1000内に情報を格納する。ある実装形態ではメモリ1020はコンピュータ可読媒体である。ある実装形態ではメモリ1020は揮発性メモリユニットである。他の実装形態ではメモリ1020は非揮発性メモリユニットである。記憶装置1030はシステム1000に大容量記憶装置を提供できる。ある実装形態では記憶装置1030はコンピュータ可読媒体である。様々な異なる実装形態では、記憶装置1030はフロッピー（登録商標）ディスク装置、ハードディスク装置、光ディスク装置でもよく、テープ装置でもよい。入力/出力装置1040は、システム1000に入力/出力動作を提供する。ある実装形態では、入力/出力装置1040はキーボードおよび/またはポインティングデバイスを含む。他の実装形態では、入力/出力装置1040はグラフィカルユーザインターフェースを表示するためのディスプレイユニットを含む。

【０１５９】

記載した機能は、デジタル電子回路内、またはコンピュータハードウェア、ファームウェア、ソフトウェア内に実装されてもよく、それらの組合せ内に実装されてもよい。装置は、プログラム可能プロセッサによって実行するために、機械可読記憶装置内などの情報担体内に有形に実装されたコンピュータプログラム製品内に実装されてよく、方法ステップは、入力データ上で動作して出力を生成することによって、記載された実装形態の機能を実行するために命令のプログラムを実行しているプログラム可能プロセッサによって実行されてよい。記載した機能は、データ記憶システムからデータおよび命令を受信するために、ならびにデータ記憶システムにデータおよび命令を伝送するために結合された少なくとも1つのプログラム可能プロセッサ、少なくとも1つの入力装置、ならびに少なくとも1つの出力装置を含む、プログラム可能システム上で実行可能な1つまたは複数のコンピュータプログラム内に有利に実装されてよい。コンピュータプログラムは、一定の活動を実行するために、または一定の結果をもたらすために、コンピュータ内で直接または間接に使用されうる命令のセットである。コンピュータプログラムは、コンパイラ型またはインタープリタ型言語を含むどのような形式のプログラミング言語でも書き込むことができ、スタンドアロン型プログラムとして、あるいはモジュール、コンポーネント、サブルーチン、またはコンピューティング環境での使用に適した他のユニットを含むどのような形式でも展開されうる。

【０１６０】

命令のプログラムの実行に適したプロセッサには、例として汎用および専用マイクロプロセッサの両方、ならびに単一プロセッサ、またはどのような種類のコンピュータの複数のプロセッサのうちの1つもある。一般的に、プロセッサは読出し専用メモリまたはランダムアクセスメモリ、あるいはその両方から命令およびデータを受信することになる。コ

ンピュータの不可欠な要素は命令を実行するためのプロセッサ、ならびに命令およびデータを格納するための1つまたは複数のメモリである。一般的にコンピュータは、データファイルを格納するための1つまたは複数の大容量記憶装置も含むか、それと通信するために動作可能なように結合されており、このような装置は内蔵ハードディスクおよびリムーバブルディスクなどの磁気ディスク、光磁気ディスク、ならびに光ディスクを含む。コンピュータプログラム命令およびデータを有形に実装するのに適した記憶装置には、例としてEPROM、 EEPROM、およびフラッシュメモリ装置などの半導体メモリ装置、内蔵ハードディスクおよびリムーバブルディスクなどの磁気ディスク、光磁気ディスク、ならびにCD-ROMおよびDVD-ROMディスクを含むすべての形式の非揮発性メモリがある。プロセッサおよびメモリはASICs（特定用途向け集積回路）で代用されてもよく、その中に組み込まれてもよい。

【０１６１】

ユーザとの対話を提供するために、ユーザに情報を表示するためのCRT（ブラウン管）またはLCD（液晶ディスプレイ）モニタなどのディスプレイ装置、ならびにユーザがコンピュータに入力を提供できるキーボードおよびマウスまたはトラックボールなどのポインティングデバイスを有するコンピュータ上に機能が実装されうる。

【０１６２】

機能は、データサーバなどのバックエンドコンポーネントを含むコンピュータシステム、あるいはアプリケーションサーバまたはインターネットサーバなどのミドルウェアコンポーネントを含むコンピュータシステム、あるいはグラフィカルユーザインターフェースまたはインターネットブラウザを有するクライアントコンピュータなどのフロントエンドコンポーネントを含むコンピュータシステム、あるいはそれらのどのような組合せ内にも実装されうる。システムのコンポーネントは通信ネットワークなどのどのような形式または媒体のデジタルデータ通信によっても接続されうる。通信ネットワークの例にはLAN、WAN、ならびにインターネットを形成するコンピュータおよびネットワークがある。

【０１６３】

コンピュータシステムはクライアントおよびサーバを含むことができる。クライアントおよびサーバは一般的に相互に離れており、通常は記載されたネットワークのようなネットワークを通じて対話する。クライアントとサーバの関係は、それぞれのコンピュータの上で稼動しており互いにクライアントとサーバの関係を有する、コンピュータプログラムによって生じる。

【０１６４】

さらに、図面に示した論理フローは、望ましい結果を達成するために示した特定の順序、または連続順序を必ずしも必要としない。さらに、記載したフローから他の諸ステップが提供されてもよく、そこから諸ステップが削除されてもよく、および記載したシステムに他のコンポーネントが追加されてもよく、そこから削除されてもよい。したがって、他の実装形態は添付の特許請求の範囲内である。

【０１６５】

本開示のいくつかの実装形態を説明してきた。しかし、本開示の趣旨および範囲から逸脱することなしに様々な改変形態が作成されてよいことが理解されよう。したがって、他の実装形態は添付の特許請求の範囲内である。

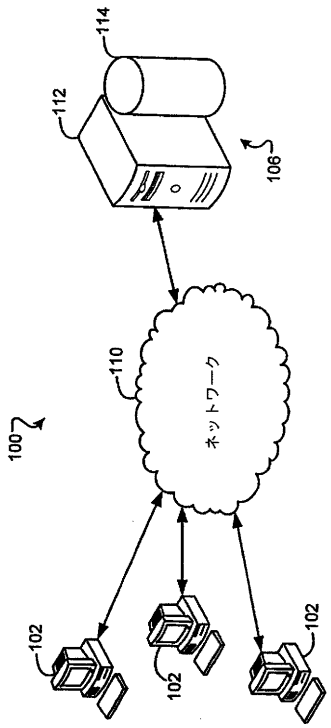
【符号の説明】

【０１６６】

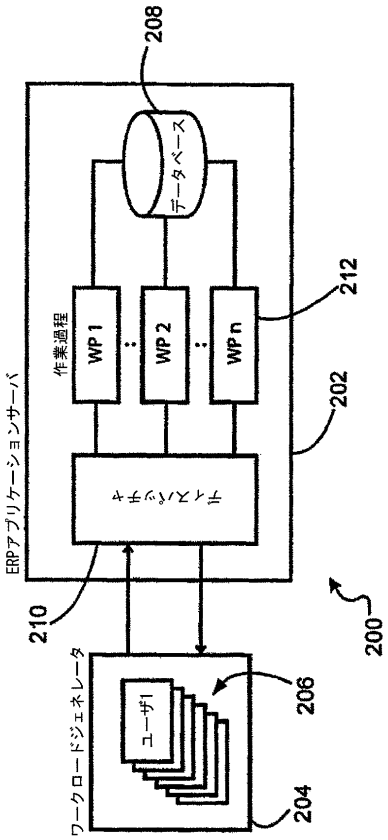
100 例示的システム
102 クライアントコンピュータ
106 バックエンドサーバシステム
110 ネットワーク
112 アプリケーションサーバ
114 データベース
200 アプリケーションサーバシステム

202 アプリケーションサーバシステム
204 ワークロードジェネレータ
206 ユーザ
208 データベース
210 ディスパッチャ
212 作業過程
300 M/M/1//Nモデル
302 遅延サーバ
304 ERPシステム
306 ウェイティングバッファ
308 作業過程およびデータベースシステム
400 データ収集
402 モデリング仮定(たとえばスケジューリングおよびサービス分布)
404 需要推定
406 モデル生成およびソリューション
1000 例示的なハードウェアコンポーネント
1010 プロセッサ
1020 メモリ
1030 記憶装置
1040 入力/出力装置
1050 システムバス

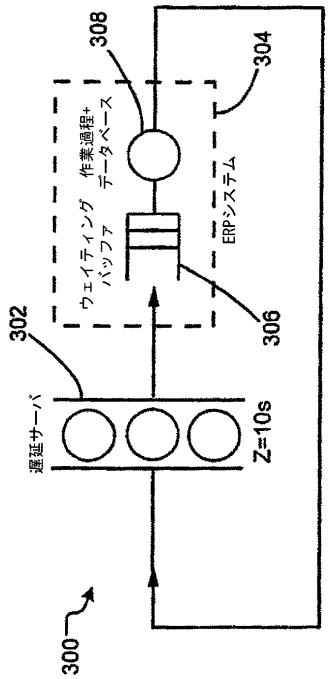
【図１】



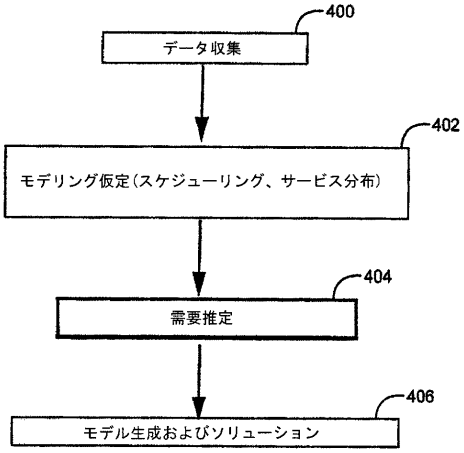
【図２】



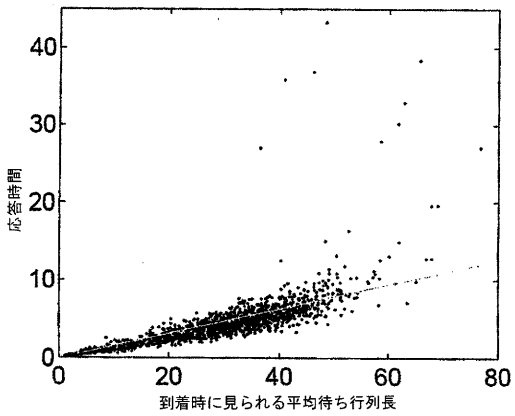
【図３】



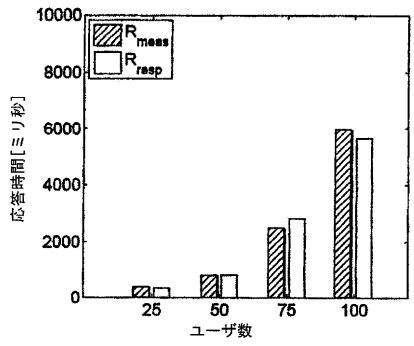
【図４】



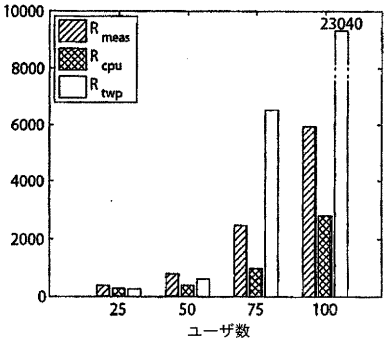
【図５】



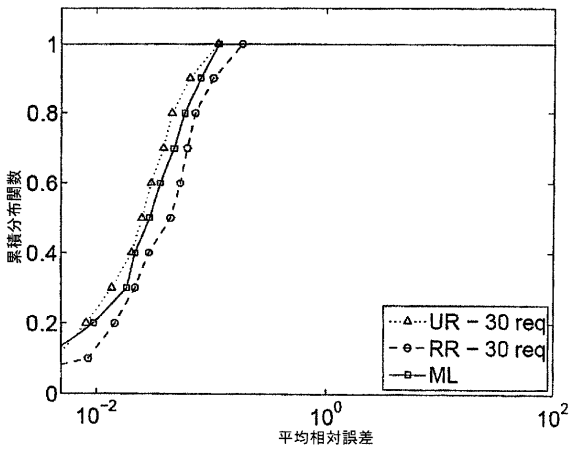
【図６Ａ】



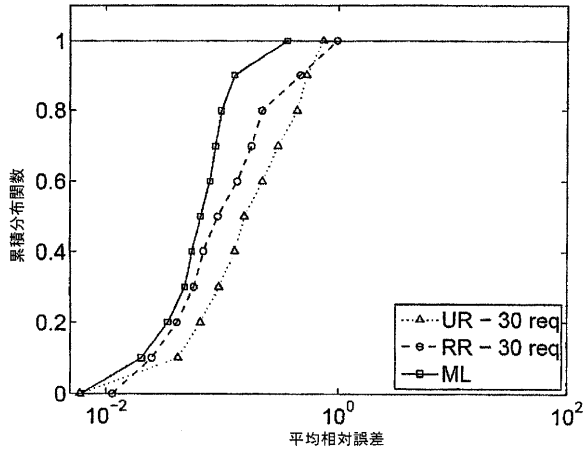
【図６Ｂ】



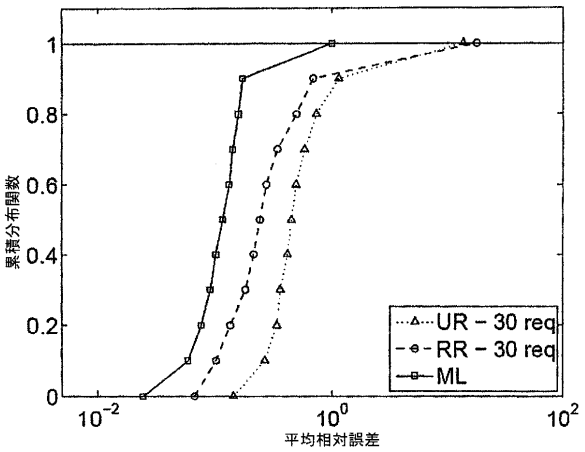
【図８Ａ】



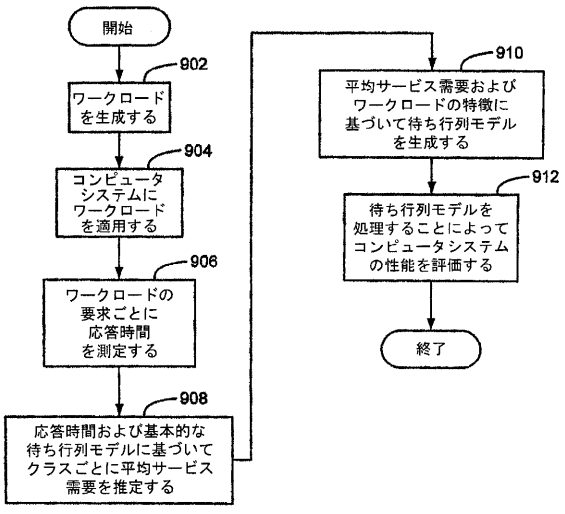
【図８Ｂ】



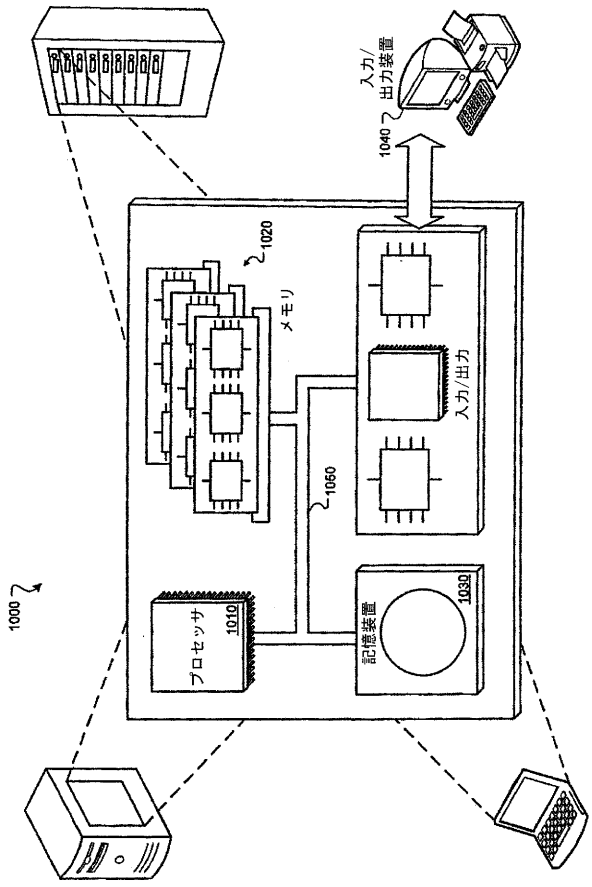
【図８Ｃ】



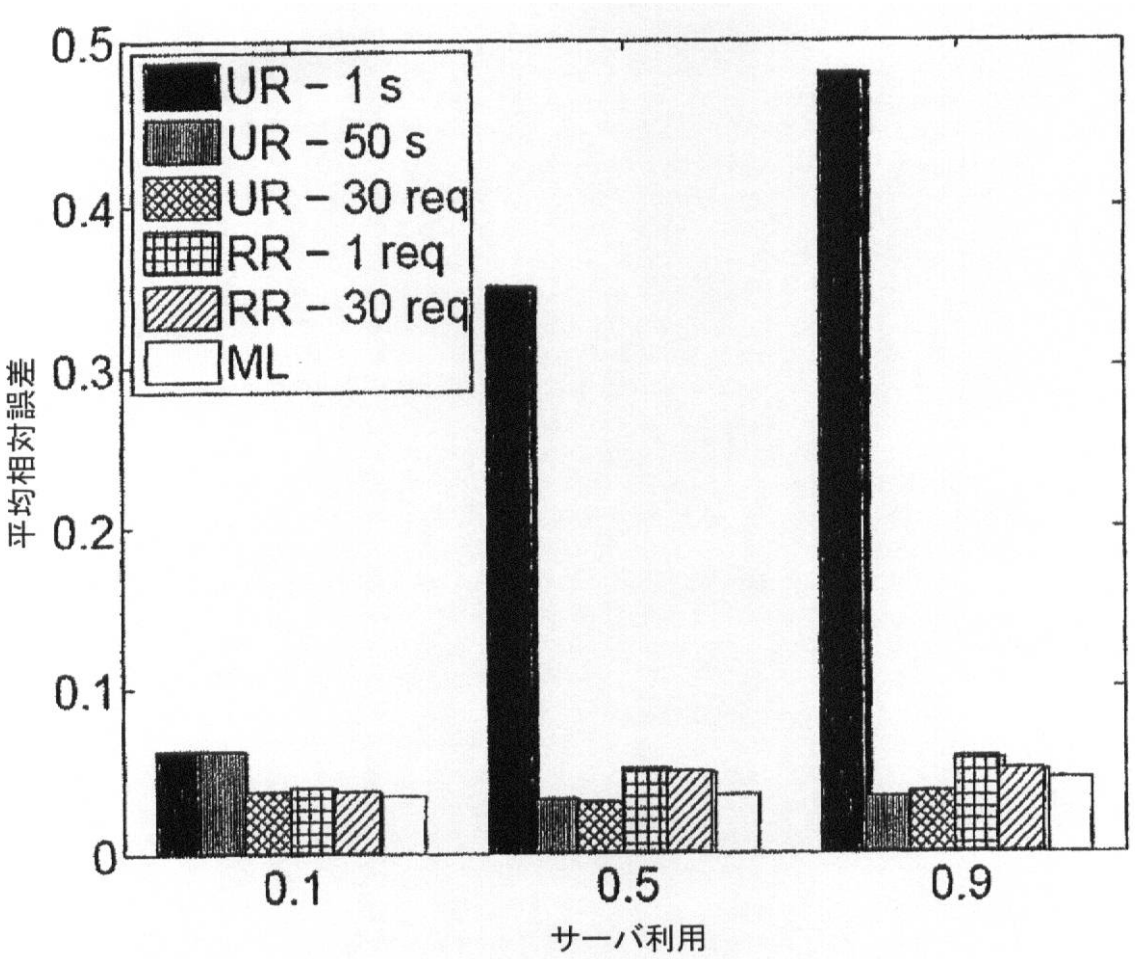
【図９】



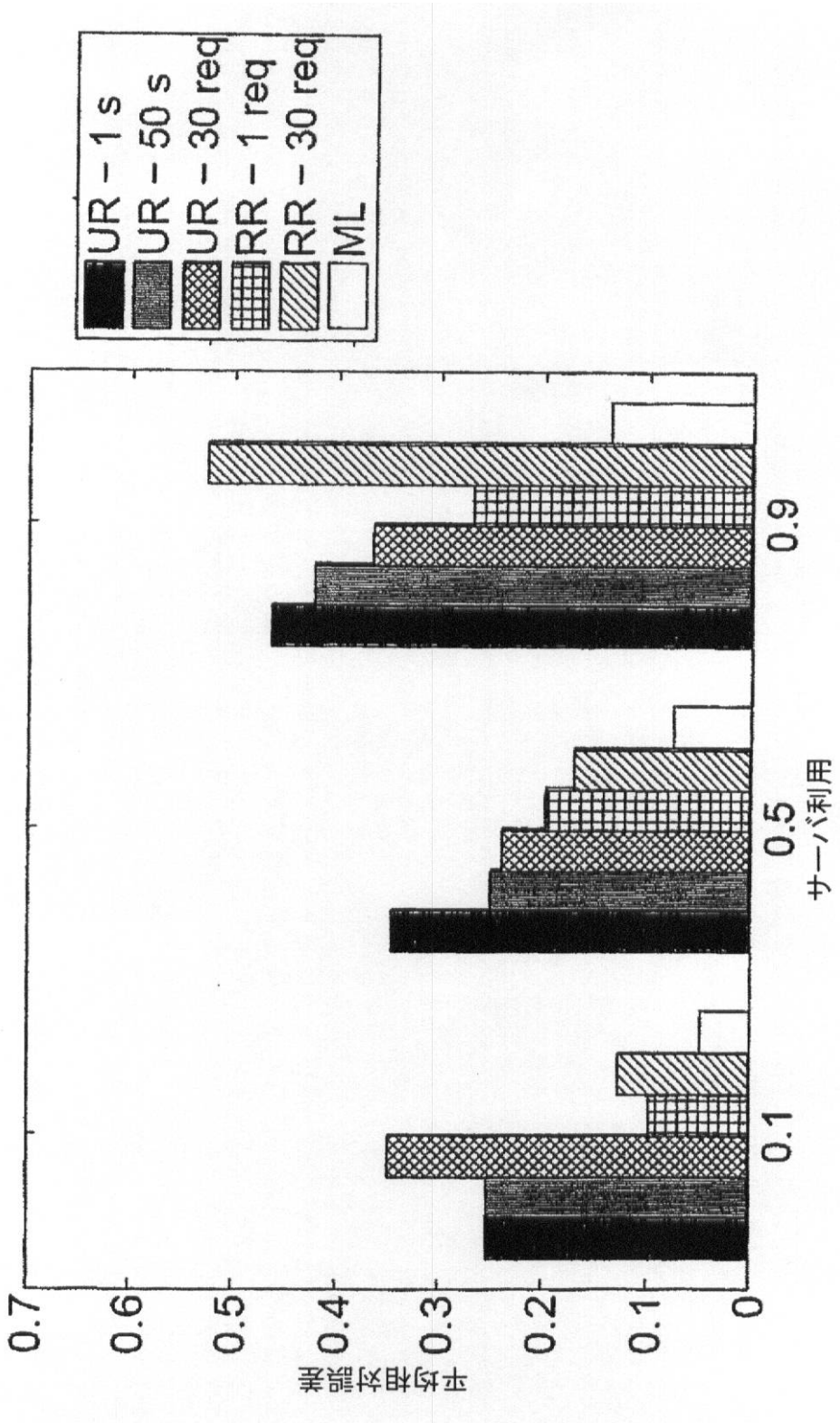
【図１０】



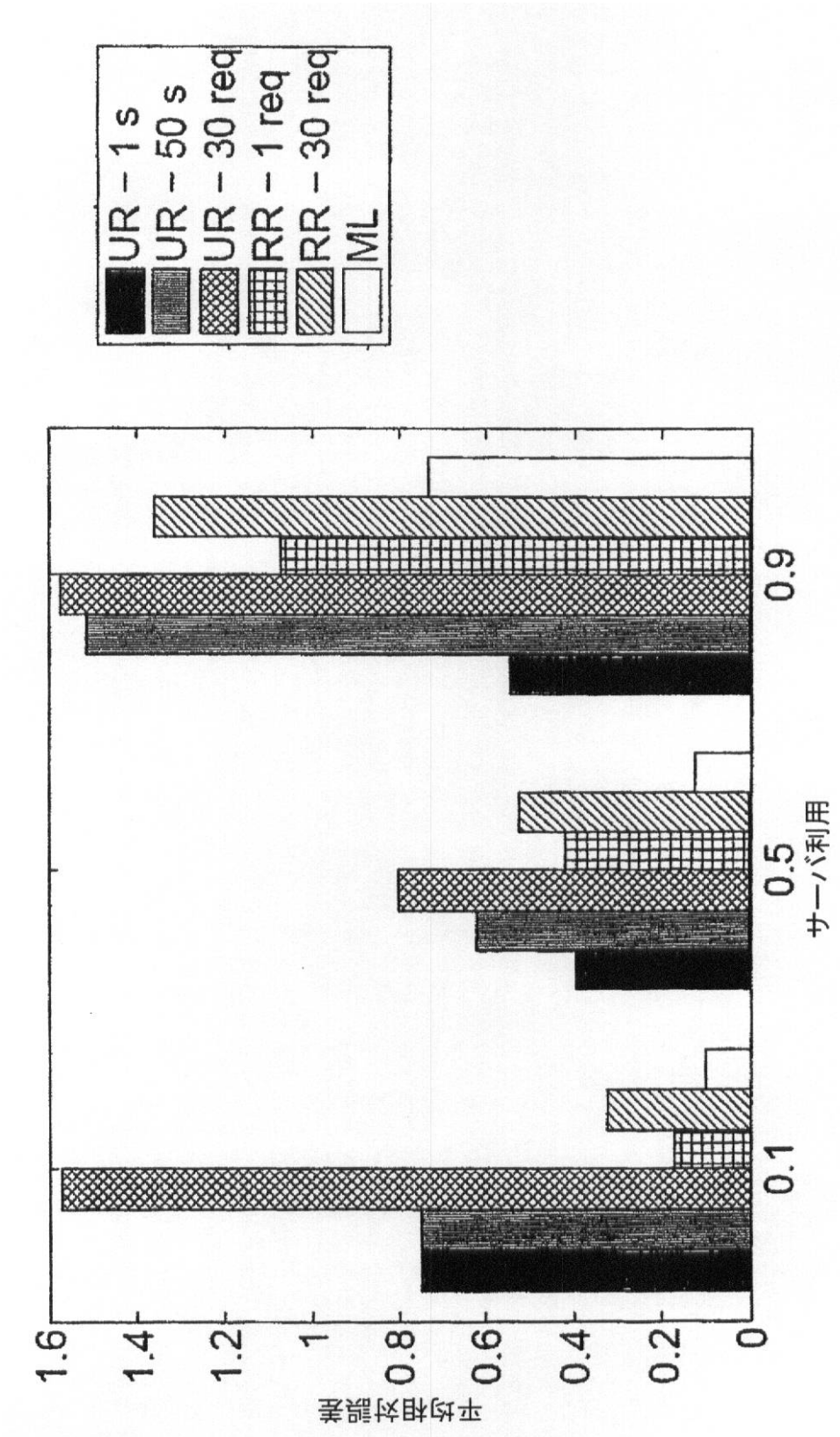
【図７Ａ】


0.5
0.4
0.3
0.2
0.1
0
UR－1 s
UR－50 s
UR－30 req
RR－1 req
RR－30 req
ML
平均相対誤差
0.1
0.5
0.9
サーバ利用

【図７Ｂ】


UR – 1 s
UR – 50 s
UR – 30 req
RR – 1 req
RR – 30 req
ML
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
平均相対誤差
0.1
0.5
0.9
サーバ利用

【図７Ｃ】


UR－1 s
UR－50 s
UR－30 req
RR－1 req
RR－30 req
ML
平均相対誤差
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0.1
0.5
0.9
サーバ利用

フロントページの続き

(74)代理人 100110364
弁理士 実広 信哉
(72)発明者 シュテファン・クラフト
イギリス・ＢＬＢＴ１・３ＳＧ・ベルファスト・セント・ジョージズ・ハーバー・４１
(72)発明者 セルヒオ・パチェコーサンチェス
イギリス・ＢＴ３７・０ＱＢ・ニュータウンアビー・（番地なし）・エスアーペー・リサーチ・シーイーシー・ベルファスト
(72)発明者 ジュリアーノ・カザーレ
イギリス・ＡＴＢＴ７・１ＪＸ・ベルファスト・クロムウェル・ロード・６８
(72)発明者 スティーヴン・ドーソン
イギリス・ＡＴＢＴ５・５ＬＵ・ベルファスト・ブルームフィールド・ロード・４６
Ｆターム(参考) 5B042 HH07 MA14 MC25 MC33 MC35 MC40

【外国語明細書】

ESTIMATING SERVICE RESOURCE CONSUMPTION BASED ON RESPONSE TIME

BACKGROUND

[0001] Performance models can be implemented to predict the scalability of software and hardware systems, either by analytical methods or by simulation. More specifically, requests to software and hardware systems place demands on system resources that are employed to respond to the requests. Although modeling and evaluation techniques exist to obtain performance predictions of such systems, there are few resource consumption estimation methods that can provide good estimates of the service demands. Service demands are important parameters in specifying performance models. Consequently, the accurate estimation of service demands is desired for defining models that are both representative and robust.

[0002] Traditionally, central processor unit (CPU) utilization has been implemented to estimate service demands of software and hardware systems. The use of CPU utilization, however, suffers from several deficiencies. For example, CPU utilization measurement requires access to an operating system executed on the hardware, and also requires specialized CPU sampling instrumentation. The CPU sampling instrumentation can interfere with normal system activities and can provide inaccurate CPU utilization measurements. In some scenarios, such as in virtualized environments, accurate CPU utilization sampling is more difficult, because filtering of hypervisor overheads is required, for example. In other scenarios, to be modeled systems are owned by third parties, such as web service providers, which do not provide CPU utilization data for their servers.

SUMMARY

[0003] Implementations of the present disclosure provide computer-implemented methods for generating a queuing model of a computer system. In some implementations, the method includes defining a workload comprising a plurality of service requests, each service request corresponding to a class of a plurality of classes, applying the workload to a computer system that receives and processes service requests, measuring a response time of the computer system for each request of the workload,

estimating a mean service demand for each class based on the response times and a base queuing model that represents the computer system, and generating the queuing model based on the mean service demands and characteristics of the workload. Characteristics of the workload can include, but are not limited to, an inter-arrival time distribution of requests arriving into the queue such as exponential inter-arrival times having mean of 5 requests per second.

[0004] In some implementations, the method further includes determining a plurality of arrival queue-lengths corresponding to each request of the workload, wherein estimating a mean service demand is further based on the plurality of arrival queue-lengths. Each arrival queue-length of the plurality of arrival queue-lengths can be determined from log files that report a time of arrival and departure of requests.

[0005] In some implementations, the method further includes determining a plurality of residual times corresponding to each request of the workload, wherein estimating a mean service demand is further based on the plurality of residual times. Each residual time can correspond to a time remaining to complete processing of an in-process request upon arrival of a to-be-processed request.

[0006] In some implementations, estimating a mean service demand includes estimating mean service demands in the base queuing model using one of linear regression and maximum likelihood method analyses based on the measured response times.

[0007] In some implementations, generating the queuing model includes parameterizing the base queuing model using the mean service demands.

[0008] In some implementations, the base queuing model includes assumptions on characteristics of the queuing model, the characteristics comprising at least one of scheduling and a service demand distribution.

[0009] In some implementations, the computer system includes an application server that executes an application, and one or more client systems that generate the requests.

[0010] In some implementations, the method further includes evaluating a performance of a computer system by processing the queuing model.

[0011] The present disclosure also provides a computer-readable storage medium coupled to one or more processors and having instructions stored thereon which, when

executed by the one or more processors, cause the one or more processors to perform operations in accordance with implementations of the methods provided herein.

[0012] The present disclosure further provides a system for implementing the methods provided herein. The system includes a computer system that receives and processes service requests, one or more processors, and a computer-readable storage medium coupled to the one or more processors and having instructions stored thereon which, when executed by the one or more processors, cause the one or more processors to perform operations in accordance with implementations of the methods provided herein.

[0013] It is appreciated that methods in accordance with the present disclosure can include any combination of the aspects and features described herein. That is to say that methods in accordance with the present disclosure are not limited to the combinations of aspects and features specifically described herein, but also include any combination of the aspects and features provided.

[0014] The details of one or more implementations of the present disclosure are set forth in the accompanying drawings and the description below. Other features and advantages of the present disclosure will be apparent from the description and drawings, and from the claims.

DESCRIPTION OF DRAWINGS

[0015] FIG. 1 is a schematic illustration of an exemplar system architecture in accordance with implementations of the present disclosure.

[0016] FIG. 2 is a functional block diagram of an exemplar enterprise resource planning (ERP) system.

[0017] FIG. 3 is a functional block diagram of an exemplar queuing model corresponding to an exemplar ERP application.

[0018] FIG. 4 summarizes a response time based approach in accordance with implementations of the present disclosure.

[0019] FIG. 5 is a graph illustrating a linear relation between measured response times and measured queue-lengths.

[0020] FIGs. 6A and 6B are graphs illustrating a comparison of response time predictions based on a response time approach in accordance with implementations of the present disclosure, and a traditional utilization based approach, respectively.

[0021] FIGs. 7A-7C are graphs illustrating exemplar mean relative errors at different server utilization levels.

[0022] FIGs. 8A-8C illustrate exemplar cumulative distribution functions (CDFs) corresponding to the mean relative errors of FIGs. 7A-7C.

[0023] FIG. 9 is a flowchart illustrating exemplar steps that can be executed in accordance with implementations of the present disclosure.

[0024] FIG. 10 is a schematic illustration of exemplar computer systems that can be used to execute implementations of the present disclosure.

[0025] Like reference symbols in the various drawings indicate like elements.

DETAILED DESCRIPTION

[0026] With particular reference to FIG. 1, an exemplar system 100 is illustrated. The exemplar system 100 of FIG. 1 can be provided as an enterprise resource planning (ERP) system including a plurality of client computers 102 that communicate with one ore more back-end server systems 106 over a network 110. The network can be provided as a large computer network, such as a local area network (LAN), wide area network (WAN), the Internet, a cellular network, or a combination thereof connecting any number of mobile clients and servers. In some implementations, the clients 102 may be directly connected to the server system 106 (without connecting through the network, for example).

[0027] The client computers 102 represent various forms of processing devices including, but not limited to, a desktop computer, a laptop computer, a handheld computer, a personal digital assistant (PDA), a cellular telephone, a network appliance, a camera, a smart phone, an enhanced general packet radio service (EGPRS) mobile phone, a media player, a navigation device, an email device, a game console, or a combination of any two or more of these data processing devices or other data processing devices. The server system 106 includes an application server 112 and a database 114, and is intended to represent various forms of servers including, but not limited to a web server, an

application server, a proxy server, a network server, and/or a server farm. In general, the server system 106 accepts user requests for application services and provides such services to any number of client devices 102 over the network 110. In some implementations, the server system 106 can provide a central point through which service-providers can manage and access data related to web services.

[0028] In operation, multiple clients 102 can communicate with server system 106 through the network 110. In order to run an application, such as a browser-based application, for example, each client 102 can establish a corresponding session with the server system 106. Each session can involve two-way information exchange between the server system 106 and each individual client 102. This two-way information exchange can include requests generated at the client 102 that are communicated to the server system 106. The server system 106 receives the requests, queues multiple requests, executes processes based on the requests, and provides a response to the requesting client 102.

[0029] Implementations of the present disclosure are directed to the accurate estimation of service demands on computer software and hardware systems, such as the system 100 discussed above with reference to FIG. 1. Although ERP systems are used to illustrate implementations of the present disclosure, applicability of the present disclosure is not limited to ERP systems. More specifically, the present disclosure is applicable to any computer software and hardware system including, but not limited to any type of multi-tier program application, modeling of hardware (e.g., a disk drive, or memory), and/or network links.

[0030] Implementation of the present disclosure provide estimation methods that use measured response times of requests to estimate service demands for a multi-class workload. The response time is provided as the end-to-end time for completing a request, and includes both the service time (e.g., the time the request is being attended to or service) and delays due to resource contention (e.g., waiting time due to buffering). The workload (i.e., a body of requests) can include several classes (i.e., transactions types). Implementations of the present disclosure address the estimation of service time, also referred to as service demand, for each class based on response time measurements.

[0031] By way of non-limiting example, a real-world industrial ERP application is used to illustrate implementations of the present disclosure showing that the disclosed response time based approach results in improved performance predictions. The response time based approach of the present disclosure is more widely applicable, because response times are often directly logged by executing applications, and/or can be obtained by an external observer. Furthermore, response times may be easily available for systems that are owned by third parties, such as web service providers, which do not publicly expose utilization data for their servers.

[0032] The service demand estimation methods of the present disclosure can be implemented with, but are not limited to, systems with first-come first-served (FCFS) scheduling, and are based on linear regression and maximum likelihood estimation approaches. The response time based linear regression uses exact equations that relate the mean response time of requests to a queue length seen on arrival of a request to the system, both of which can be easily obtained from application logs. The maximum likelihood approach considers the entire distribution of the measured response times, and can achieve increased estimation accuracy. Using phase-type distributions, the present disclosure provides a computational method to evaluate likelihood values and to estimate resource consumption.

[0033] This information is used to parameterize a queuing model that models the underlying system. As used herein, parameterizing includes a process of determining and defining the parameters necessary for a complete or relevant specification of a model, a queuing model in the present case. A queuing model is a model that approximates a real queuing system so the queuing behavior can be analyzed using steady-state performance measures. Queuing models can be represented using Kendall's notation which is provided as: A / B / S / K / N / Disc; where A is the inter-arrival time distribution, B is the service time distribution, S is the number of servers, for example, K is the system capacity, N is the calling population, and Disc is the service discipline assumed. In some instances, K, N and Disc are omitted, so the notation becomes A / B / S. Standard notation for distributions A and/or B include M for a Markovian (exponential) distribution, Eκ for an Erlang distribution with κ phases, D for Degenerate (or

Deterministic) distribution (constant), G for General distribution (arbitrary), and PH for a Phase-type distribution.

[0034] Numerical results, discussed in further detail below, illustrate the improved accuracy of the response time based approach as compared to utilization based approaches. Furthermore, an intrinsic advantage of the response time based approach of the present disclosure is that response times at a server depend on all of the latency degradations incurred by requests within the system. Consequently, the response times are inherently more comprehensive descriptors of performance, because the response times account for bandwidth, memory, and input/output (I/O) contention delays. The response time based approach of the present disclosure readily accounts for these delays, which are otherwise ignored by traditional approaches.

[0035] Referring now to FIG. 2, a simplified, exemplar application server system 200 is illustrated. The simplified architecture of FIG. 2 is presented as a non-limiting example that is used to illustrate implementations of the present. The system 200 includes an application server system 202 and a workload generator 204. The workload generator 204 functions as a plurality of client devices (e.g., clients 102) that transmit requests to the application server system 202, placing service demands on resources of the application server system 202. The system 200 can include an ERP application of one or more application programs, such as those included in the SAP Business Suite, provided by SAP AG of Walldorf, Germany. The exemplar system 200 of FIG. 2 is provided in a two-tier configuration and includes an application server and a database server installed on a common virtual machine. In the example of FIG. 2, and for purposes of the present illustration, no other virtual machines run on the physical machine. It is appreciated, however, that the present disclosure is not limited to such arrangements.

[0036] The system 200 is stress-tested using a workload of operations (e.g., sales and distribution), which can include, but are not limited to standard business transactions (e.g., the creation and/or listing of sales orders and invoices). The workload generator 204 can be of a closed-type where requests are issued by a fixed group of N users 206. Upon the completion of a request submitted by a user, an exponentially-distributed think time (e.g., with mean time Z equal to 10s) expires before submitting a new request to the system. All users 206 can cyclically submit the same sequence of requests to the ERP

application. After submission of the requests, the application server processes the requests from the multiple, concurrent users 206. This processing can be achieved using information extracted from a database 208. The application server uses a dispatcher 210 that buffers the requests in a queue and transfers the requests to work processes 212. The work processes 212 can be provided as independent operating system threads serving the requests in parallel. The work processes 212 can be executed as either dialog processes, which execute interactive programs, or update processes, which perform database changes.

[0037] The parameterization of a basic queuing model that describes the performance of an ERP application (e.g., of FIG. 2) can be significantly improved if the measured response times (R_{MEAS}) are used to estimate resource consumption of requests. The performance model considered for this task includes, but is not limited to, a basic M/M/1/ /N model 300, an exemplar structure of which is illustrated in FIG. 3. The exemplar model 300 of FIG. 3 includes a delay server 302 and an ERP system 304 that includes a waiting buffer 306 and a work processes and database system 308. The M/M/1/ /N model 300 is an M/M/1 queue with requests generated by a finite population of N users. This can also be seen as a closed queuing network composed of a delay station, which models user think times, followed by a queuing station on a closed-loop. The waiting buffer 306 represents an admission control in the ERP system 304, and the server models the resource consumption of the requests when executed in the work process.

[0038] It is common that the utilization of the database server is less than 5%. Consequently, queuing effects at the database tier are negligible and the total time to provision data from the database (T_{MEAS}^{DB}) can be used as a component of the service demand in the work process. Both think times and service demands in the queuing model can be assumed to be exponentially distributed. This assumption has been verified as a good approximation of actual service demand distribution in an ERP system. The coefficient of variation (CV) of the measured service demands (i.e., the ratio between standard deviation and mean) ranges over different validations in CV = [1:02; 1:35], whereas for the theoretical exponential is CV = 1. Consistently with the model assumptions, all think times for validations have been generated using an exponential distribution as well.

[0039] When parameterizing a queuing model, a base queuing model is initially determined. The base queuing model is a stochastic model that represents the arrival and departure of requests from resources of the computer system (e.g., CPU). The base queuing model is provided as an abstraction of the analyst in order to capture the phenomena that affect performance of requests served by the resource modeled. Characteristics that are defined for such a queuing model to describe the computer system include the resource scheduling disciplines and their service time distributions, as well as interconnections with other resources and/or the statistical characteristics of request arrivals at each resource. Based on this information, the base queuing model can be provided using standard Markov chain theory, for example. A final or target queuing model can be subsequently parameterized based on estimated mean service demands that can be determined by processing the base queuing model and measured response times. The estimation of the mean service demand provided in the present disclosure enables the paramterization of the service time distribution for one or more resources modeled in the base queuing model.

[0040] The mean service demand (E[D]) of the requests at the server is determined such that the response times predicted by the model accurately match accurately the measured response times (R_{MEAS}) of the real system for all possible numbers of users N. Due to the large role of caching, which affects the behavior of the ERP system very differently at light and heavy loads, the service demands are specified as a function of the number of users N in the model, (i.e., $E[D] \equiv E[D](N)$). This estimation approach is routinely used in modeling complex software systems.

[0041] Referring now to FIG. 4, the present disclosure provides direct parameterization of queuing models based on the response time measurements (R_{MEAS}). More specifically, service demands are estimated to best match the distribution or moments of the measured response times. This requires an important paradigm shift in the modeling methodology as illustrated in FIG. 4. The modeling methodology of FIG. 4 includes data collection 400, modeling assumptions (e.g., scheduling and service distribution) 402, demand estimation 402, and model generation and solution 406. A solution of the model can be provided by values for the mean response times, the mean

throughput, and the probability of observing a certain number of requests in the queue at a random instant.

[0042] Because the response times depend on several system properties related to scheduling or service demand distribution, assumptions on scheduling or general form of the distribution are taken prior to starting the service demand estimation activity. The assumptions can include, but are not limited to, the type of scheduling (e.g., FCFS scheduling), the number of workload classes to be modeled, and the distribution of the service demands of each request (e.g., exponential distribution). Consequently, the returned service demands depend on the characteristics of the model in which they will be used. Preliminary assumptions on the characteristics of the final model are provided to determine the best possible service demand estimates relative to the target model. Because the goal of the modeling activity is to obtain good agreement between experimental observations and model predictions, the response time based approach of the present disclosure has a stronger focus on achieving this goal by returning the best parameters under model assumptions.

[0043] As discussed above, the illustrative ERP system can be modeled as a FCFS queue with exponentially distributed service demands. For example, an exponential distribution can describe the times between events in a process in which events occur continuously and independently at a constant average rate. Under these assumption, a request that finds the system with n waiting requests and one request in service receives a response time distribution given by the distribution of the sum of random variables, provided as:

$$R = T + D^1 + D^2 + \ldots + D^n + D^{n+1} \qquad (1)$$

where T is the residual time before completion of the request in service, D^i for $1 \leq i \leq n$ is the service demand of the ith queued request, and D^{n+1} is the service demand of the newly arrived request. Note that, by definition of exponential distribution, T is equal to D and thus the distribution of the random variable R is the convolution of n + 1 exponential random variables.

[0044] By simple derivations it can be shown that the mean response time is found to satisfy the following relation:

$$E[R] = E[D](1 + E[A]) \qquad (2)$$

where E[D] is the mean service demand and E[A] is the mean queue length observed by a request upon time of arrival. Equation 2 can be used to estimate the mean service demand of the requests given the knowledge of the mean number of requests in the ERP system at the time of arrival of a new request. The value A can be obtained from the log files of experiments that have been performed, which report the time of arrival and departure of requests. An estimate of E[D] (D_{EST}^{RSP}) can be obtained by linear regression of Equation 2 using a sequence of samples of E[R] and E[A] obtained by averaging R_{MEAS} and the measured A values for groups of X consecutive requests (e.g., X = 20). An exemplar graphical illustration of a measured linear relation between E[R] and E[A] in an ERP system is provided in FIG. 5.

[0045] From Equation 2, E[D] can be evaluated by linear regression as:

$$E[R_{MEAS}^{i}] = E[D](1 + E[A_{MEAS}^{i}]) \quad (3)$$

where the index i stands for the ith value collected in the measurement activity and $E[D_{MEAS}^{i}]$ and $E[A_{MEAS}^{i}]$ are the response time and arrival queue-length, respectively, averaged on X consecutive i samples (e.g., X = 20).

[0046] Using the above-described equations as a basis, the present disclosure provides service demand estimation algorithms for multi-class requests based on linear regression and maximum likelihood techniques, each of which is described in further detail herein using an exemplar queuing scenario. Solely for purposes of exemplar illustration, it is assumed that a user has provided an input trace with a total of I samples to the service demand estimation algorithms. This can include, but is not limited to, a sequence of measured response times (R_c) for requests of class c, and the corresponding queue-length (A_k^c) of class-k requests seen upon arrival by each class c request. The number of workload classes is denoted by K.

[0047] In one implementation, the linear relation given in Equation 3 does not change if the assumptions on exponential service are dropped, or multiple classes are considered. Accordingly, a variant of the expression developed for approximating FCFS queues in non-product-form queuing networks with multiple classes can be provided as:

$$E[R_c] = E[T_c] + \sum_{k=1}^{k} E[D_k]\left(1_{k,c} + E[A_k^c]\right) \quad (4)$$

where $1_{k,c}$ is equal to 1 when k = c and is equal to 0 when k ≠ c. T_c is the residual time before completion of the request in execution at the instant of arrival of the class c request, K is the number of request classes, E[Dk] is the mean service demand of class k, and E[A_k^c] is the mean number of requests of class k queuing in the system excluding both the newly arrived request and the request currently in service.

[0048] Equation 4 can provide the mean request time of a request of class c in a GI/GI/1/FCFS queue, for example. GI indicates general independent distribution, which is a specific type of general distribution where requests are independent of each other. In the exemplar case of a GI/GI/1 queue with FCFS scheduling, the random variable describing the response time of a request of class c arriving when the system has n+1 queued requests (i.e., n requests waiting plus the request in service) is provided as:

$$R_c = T_c + D^1 + D^2 + \ldots + D^n + D_c \qquad (5)$$

where T_c is the residual time before completion of the current request in service, D^i for $1 \leq i \leq n$ is the service demand of the ith queued request, and D_c is the service demand of the newly arrived request.

[0049] Taking expectations conditioned on the arrival queue-length being $A^c = n$ provides:

$$E[R_c \mid A^c = n] = E[T_c \mid A^c = n] + \sum_{i=1}^{n} E[D^i \mid A^c = n] + E[D_c] \qquad (6)$$

where C(i) is the class of the request in position i ($1 \leq i \leq n$).

[0050] The service demands as a function of the request class can be expressed as:

$$E[D^i \mid A^c = n] = \sum_{k=1}^{K} E[D_k] \mathrm{P}[C(i) = k \mid A^c = n] \qquad (7)$$

[0051] Substituting Equation 7 into Equation 6 provides:

$$E[R_c \mid A^c = n] = E[T_c \mid A^c = n] + \sum_{i=1}^{n} \sum_{k=1}^{K} \mathrm{P}[C(i) = k \mid A^c = n] E[D_k] \qquad (8)$$

[0052] The mean response time can subsequently be determined based on:

$$
\begin{aligned}
E[R_c] &= \sum_{n=1}^{+\infty} E[R_c \mid A^c = n]\mathrm{P}[A^c = n] \\
&= E[T_c] + \sum_{k=1}^{K} E[D_k] \sum_{n=1}^{+\infty} \left(1_{k,c} + \sum_{i=1}^{n} \mathrm{P}[C(i) = k \mid A^c = n] \mathrm{P}[A^c = n] \right) \\
&= E[T_c] + \sum_{k=1}^{K} E[D_k] \left(1_{k,c} + \sum_{n=1}^{+\infty} E[A_k^c \mid A^c = n] \mathrm{P}[A^c = n] \right) \\
&= E[T_c] + \sum_{k=1}^{K} E[D_k] \left(1_{k,c} + E[A_k^c] \right)
\end{aligned}
\tag{9}
$$

where $\sum_{i=1}^{n} \mathrm{P}[C(i) = k \mid A^c = n]$ is the mean number $E[A_k^c \mid A^c = n]$ of class k requests waiting in the queue upon arrival by a request of class c.

[0053] Estimates of the class demands $E[D_k]$ can be obtained from Equation 4 using regression methods such as the non-negative least-squares algorithm (e.g., the `lsqnonneg` provided in MATLAB by The Mathworks, Inc.) and confidence intervals on the estimates can be generated using standard formulas. Equation 4 can be used in linear regression for estimating the mean service demands $E[D_k]$, if the per-class response times R_c and the arrival queue-lengths A_c are known. In general, $E[T_c]$ can be difficult to directly measure. Consequently, approximations are needed to estimate this quantity. To achieve this, approximation schemes can be implemented, which can include, but are not limited to the standard Poisson arrival approximation $T_c = D_k$.

[0054] Due to the lack of exact results for the residual time expression (T_c) in the exemplar GI/GI/1/FCFS queues, the following renewal-theory expression for the residual time is used:

$$
E[T_c] = \frac{E[D_c]}{2}\left(1 + CV_c^2\right) \tag{10}
$$

where CV_c is the coefficient of variation of the service demand distribution of class c. This is clearly an approximation in GI/GI/1/FCFS queues, which becomes an exact expression when the arrival stream is a Poisson process. If the CV_c value is not known *a priori*, it is still possible to evaluate Equation 4 for different CV_c values that are assumed feasible for the system under study. The set that produces the minimum mean residual error in the least-squares solution of Equation 4, which is thus the group of coefficient of variations that best fits the observations, is selected as the best estimate of the CV_c

values. This outlines a general schema for possible application of Equation 4 to non-exponential service distributions.

[0055] In some implementations, maximum likelihood estimation can be used for inferring statistics of random variables based on analytical expressions of the probability of observing a certain sample path. Within the scope of mean service demand estimation, maximum likelihood can be formulated by letting R^i denote the ith observed, or measured response time, for all $1 \leq i \leq I$, where I is the total number of measured response times. The mean service demand $E[D_k]$ for each class $1 \leq k \leq K$ is sought, such that the probability of observing the sequence of measured response times $R^1, \ldots, R^i, \ldots, R^I$ is maximal. Formally, this can be expressed as finding the set of mean service demands $E[D_1], \ldots, E[D_K]$. This solves the following maximization problem:

$$\max_{E[D_1],\ldots,E[D_k]} \mathrm{P}\left[R^1,\ldots,R^i,\ldots,R^I \mid E[D_1],\ldots,E[D_K]\right] \tag{11}$$

subject to $E[D_k] \geq 0$, for all classes k. Assuming the response time as independent random variables, the joint probability expression in Equation 11 is simplified into the product:

$$\max_{E[D_1],\ldots,E[D_k]} \prod_{i=1}^{I} \mathrm{P}\left[R^i \mid E[D_1],\ldots,E[D_K]\right] \tag{12}$$

and taking the logarithm to equivalently express the products as a summation provides:

$$\max_{E[D_1],\ldots,E[D_k]} L(E[D_1],\ldots E[D_K]) \tag{13}$$

where the argument is now the likelihood function provided as:

$$L(E[D_1],\ldots E[D_K]) = \sum_{i=1}^{I} \log P\left[R^i \mid E[D_1],\ldots E[D_K]\right] \tag{14}$$

[0056] The challenge is to obtain an expression for the likelihood function that is representative of the problem under study, and for which the maximum can be computed efficiently. It is also important that this expression is analytically tractable, since optimization can be otherwise too expensive. In particular, the focus here is on the estimation of the mean service demand, because this is a significant parameter for the specification of standard capacity planning models based on product-form queuing models.

[0057] In some implementations, one method for deriving the likelihood function characterizes both the service and response times of the subject system by phase-type distributions, which enjoy efficient analytical expressions for their evaluation. To avoid unnecessary complexity, an approach for the case of exponentially-distributed service demands is set forth, and an extension of this to the case of general service demands is provided.

[0058] For example, a request that arrives to the system when $n_1, \ldots, n_K$ requests are queued for each class, including the request currently in service, is considered. Exponential service demands can be assumed for all classes, and the mean service rate of class k is denote by $\mu_k = 1/E[D_k]$. The distribution of response times for the kth request is given by the sum of n_1 exponential service demands with rate μ_1, n_2 exponential service demands with rate μ_2, and so forth for all K classes. Consequently, this can be modeled as the time to absorption in a Markov chain with $n = n_1 + \ldots + n_K$ states, each one representing the waiting time due to a request service. By way of non-limiting example, if there are $K = 2$ classes and the queue seen on arrival is $n_1 = 3$ and $n_2 = 2$, including both the arrived request and the one in service, the time to absorption can be provided by the phase-type distribution with the following (D_0, D_1) representation:

$$D_0 = \begin{bmatrix} -\mu_1 & \mu_1 & 0 & 0 & 0 \\ 0 & -\mu_1 & \mu_1 & 0 & 0 \\ 0 & 0 & -\mu_1 & \mu_1 & 0 \\ 0 & 0 & 0 & -\mu_2 & \mu_2 \\ 0 & 0 & 0 & 0 & -\mu_2 \end{bmatrix} \tag{15}$$

$$D_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \mu_2 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{16}$$

[0059] The D_0 matrix has off-diagonal elements in position (i, j) representing a transition from state i to state j that does not lead to absorption, while D_1 elements are transitions associated to absorption. The diagonal of D_0 is such that D_0 plus D_1 is an

infinitesimal generator of a Markov chain describing the evolution of the active state over time.

[0060] The (D_0, D_1) representation enables the probability that a request receives a response time R^i to be efficiently determined. The time to absorption is described by the probability density of the phase-type distribution (D_0, D_1). From the basic theory of absorbing Markov chains this is readily provided as:

$$P\left[R^i \mid E[D_1],\ldots,E[D_K]\right] = \vec{\pi}_e e^{D_0 R^i}\left(D_1\right)\vec{e} \qquad (17)$$

where $\vec{e} = (1, 1, ..., 1)^T$, $\vec{\pi}_e = \vec{\pi}_e\ (-D_0)^{-1}D_1$ is a row vector with elements $(\vec{\pi}_e)_j$ representing the initial state of a request immediately after absorption, and knowledge of $E[D_1]$, ..., $E[D_k]$ is equivalent to knowing the rates μ_1, ..., μ_K. Because there is a single transition in D_1 leading to absorption, it is immediately concluded that, in Equation 17, it is always $\vec{\pi}_e = (1, 0, ..., 0)$ and the main cost of evaluating Equation 17 is the computation of the matrix exponential function. This can be approximated in an efficient manner using a uniformization technique, or by Padé expansion (e.g., provided in MATLAB). The above-described approach can be generalized to non-exponential service demands that can be approximated as phase-type.

[0061] This concept can be illustrated by letting (S_0^k, S_1^k) denote the phase-type representation of the service demands of class k. By way of non-limiting example, for the Erlang-2 distribution, this is provided as:

$$S_0^k = \begin{bmatrix} -\mu_k & \mu_k \\ 0 & -\mu_k \end{bmatrix}; \text{ and}$$

$$S_1^k = \begin{bmatrix} 0 & 0 \\ \mu_k & 0 \end{bmatrix},$$

which implies a probability vector $\vec{\pi}_e^k = (1, 0)$.

[0062] For the same example of the exponential case, and if the service is distributed using the exemplar Erlang-2 distribution, the response time spent in the queue can be expressed as:

$$D_0 = \begin{bmatrix} -S_0^1 & S_1^1 & 0 & 0 & 0 \\ 0 & -S_0^1 & S_1^1 & 0 & 0 \\ 0 & 0 & -S_0^1 & S_1^1 \vec{e}\,\vec{\pi}_e^2 & 0 \\ 0 & 0 & 0 & -S_0^2 & S_1^2 \\ 0 & 0 & 0 & 0 & -S_0^2 \end{bmatrix} \quad (18)$$

$$D_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ S_1^2 \vec{e}\,\vec{\pi}_e^1 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (19)$$

where $\vec{\pi}_e^k = \vec{\pi}_e^k \left(-S_0^k\right)^{-1} S_1^k$. The $\vec{e}\,\vec{\pi}_e^k$ terms ensure proper initialization of the phase-type distribution, because, in general, the state space sizes of (S_0^1, S_1^1) and (S_0^2, S_1^2) are different. The above-described expression can be immediately evaluated for computing the likelihood function using Equation 17. The only difference is that this involves larger matrices.

[0063] For service demand distributions specified by several parameters, such as the hyper-exponential distribution, the above-described approach requires either all moments be fixed except the mean *a priori* (e.g., by guessing the variability of the service process), or to integrate these additional parameters as unknown variables in the maximum likelihood processing. This changes the structure of the maximum likelihood estimation problem to evaluating conditional probabilities in the form of:

$$\log P\left[R' \mid E[D_k], E[D_k^2], \ldots, E[D_k^M], 1 \le k \le K\right] \quad (20)$$

if the phase-type distributions are uniquely specified by their first M moments $E[D_k^M]$, $1 \le m \le M$. For example, the hyper-exponential distribution can be fully determined by three parameters. Consequently, the first three moments $E[D_k^M]$ are sufficient to completely specify the distribution.

[0064] In the presence of high-variability distributions, which are the most challenging to address, it is more practical to decompose the workload in an increased number of classes. In this manner, the service distribution of each class can be approximated by an exponential. For example, if a workload class has hyper-exponential

service demands with density $f(t) = p\mu_1 e^{-\mu_1 t} + (1-p)\mu_2 e^{-\mu_2 t}$, and arrival rate λ, then the workload can be approximated by two classes with exponential service demands $f_1(t) = \mu_1 e^{-\mu_1 t}$ and $f_2(t) = \mu_2 e^{-\mu_2 t}$, and arrival rates $\lambda_1 = p\lambda$ and $\lambda_2 = (1-p)\lambda$, respectively. This enables the focus to be on the mean service demand estimation under simpler exponential assumptions. Accordingly, a point of interest of the service demand analysis is to be able to reliably estimate exponentially-distributed service demands in a multi-class setting.

[0065] Referring now to FIGs. 6A and 6B, the accuracy of the response time based approach is highlighted as compared to a traditional CPU utilization based approach. FIG. 6A provides estimates and model prediction accuracy results for the exemplar M/M/1/ /N queuing models parameterized respectively using the response time based approach of the present disclosure. FIG. 6B provides estimates and model prediction accuracy results for the exemplar M/M/1/ /N queuing models parameterized respectively using the CPU utilization based approach. FIG. 6B illustrates that the model parameterized in accordance with a CPU utilization based approach provides a response time estimate (R_{CPU}) that grossly underestimates the measured response time (R_{MEAS}) of the underlying ERP system. In this case, R_{CPU} values are determined based on the exemplar queuing model being parameterized with E[D] being equal to D_{EST}^{CPU}, which is determined by linear regression of the sum of the total CPU consumption of each request (T_{MEAS}^{CPU}) and the total time to provision data from the database (T_{MEAS}^{DB}) against the number of completed requests.

[0066] FIG. 6B also illustrates that the model parameterized in accordance with a CPU utilization based approach provides a response time estimate (R_{TWP}) that grossly overestimates the measured response time (R_{MEAS}) of the underlying ERP system. In this case, R_{TWP} values are determined based on the exemplar queuing model being parameterized with E[D] being equal to D_{EST}^{TWP}, which is determined by linear regression of the total time spent in the work process (T_{MEAS}^{TWP}) against the number of completed requests. As seen in FIG. 6A, the response time based approach of the present disclosure is more effective in producing high-quality predictions, as compared to the utilization based service demand estimation considered in FIG. 6B.

[0067] The accuracy of the response time based service demand estimation in accordance with the present disclosure can also be illustrated with reference to utilization and response time traces generated by a queuing system simulator using simulation periods of 60, 600 and 3600 seconds, for example. Such a simulator can been written using a numerical computing environment and programming language (e.g., MATLAB provided by The Mathworks, Inc.) and can programmed to log standard performance measures, as well as the queue-length seen upon arrival by requests. Inter-arrival times and mean service demands can be generated from exponential distributions and their means are assigned to match utilization levels that are fixed for the simulations.

[0068] The simulations are based on system utilization and response time measurements over an exemplar period of 600 seconds, and have been run with low, medium and high server utilization levels (e.g., $\rho \in \{0.1, 0.5, 0.9\}$), as well as with request classes $K \in \{1, 2, 5\}$. Response time measurements can be collected on a per-request basis, while utilization is sampled every second. This is the typical situation in modern systems that easily log response times of individual requests, but cannot probe utilization with fine-grain or even per-request resolution. The demand estimation accuracy is compared for the traditional, CPU utilization-based approach (UR), and the methods in accordance with the present disclosure including response-time based regression (RR) and maximum likelihood (ML).

[0069] For each simulation, corresponding to a different choice of the number of classes K and of the utilization ρ, the estimation accuracy can be evaluated based on an error function provided by:

$$\Delta = \sum_{k=1}^{K} \frac{1}{K} \left| \frac{E[D_{EST,k}] - E[D_{EXACT,k}]}{E[D_{EXACT,k}]} \right| \qquad (21)$$

which is the mean relative error over all classes of the estimated service demands $E[D_{EST,k}]$ with respect to the exact value $E[D_{EXACT,k}]$ used in the simulations. For each simulation, the $E[D_{EXACT,k}]$ value of each class is randomly drawn with uniform distribution ranging in [0, 1].

[0070] FIGs. 7A-7C include exemplar graphs illustrating the mean value $E[\Delta]$ determined by averaging Δ over 100 simulations with random service demands and having the same number of classes K and server utilization ρ. FIG. 7A is a graph

corresponding to a single request class and illustrates the mean relative error values for server utilization levels of 0.1, 0.5 and 0.9. FIG. 7B is a graph corresponding to two request classes and illustrates the mean relative error values for server utilization levels of 0.1, 0.5 and 0.9, and FIG. 7C is a graph corresponding to five request classes and also illustrates the mean relative error values for server utilization levels of 0.1, 0.5 and 0.9. In FIGs. 7A-7C, UR – 1s indicates the mean relative error value for CPU utilization based regression on all available samples, UR – 50s indicates the mean relative error value for CPU utilization based regression averaging measurements over 50 consecutive examples, and UR – 30 req indicates the mean relative error value for CPU utilization based regression using averages on partitions formed by 30 consecutive samples such that 30 requests fall into each partition. RR – 1 req indicates the mean relative error value for response time based regression using all available samples, while RR – 30 req uses averages over partitions of 30 requests of the same class. ML indicates the mean relative error value for response time based maximum likelihood using all available samples.

[0071] FIGs. 8A-8C include exemplar graphs that illustrate the cumulative distribution function (CDF) of Δ over 100 random models for $\rho = 0.5$ and $K = 1, 2, 5$. FIGs. 8A-8C only include the CDF's for UR – 30 req, RR – 30 req and ML for ease of plotting, and because the curves not shown lead to qualitative conclusions in line with FIGs. 7A-7C.

[0072] The single-class scenario is the simplest scenario, because only a single mean service demand E[D] is estimated. Most methods perform well in this case showing an error around 5% for the different utilization levels (see FIG. 7A). Only UR – 1 s delivers a low quality result featuring error values of more than 30% even at medium load (ρ = 0.5). This effect can be explained in terms of an insufficient number of events in the sample period, which invalidates the linear regression formula, because the sample averages are computed on a subset of values that has too small of a cardinality (e.g., even 1, 2 or 3 requests). Consequently, the aggregations used in UR – 30 req and UR – 50 s have a remarkably positive effect on removing the UR – 1 s errors, lowering E[Δ] to below 5% for UR – 30 req. Furthermore, the RR methods provide high quality results for all utilization levels. ML also provides similar high-quality results for all utilization levels. The observations of FIG. 7A are confirmed by the error distribution. More

specifically, FIG. 8A provides low probabilities of high errors for most of the methods, approximately with a 95th-percentile of 12% for all techniques. Accordingly, for the single class case, the techniques, except UR – 1 s, provide accurate results.

[0073] When moving from single to multi-class workloads, the errors grow as illustrated in FIG. 7B. ML has the smallest errors when considering two request classes and provides errors of less than 8% in medium load situations ($\rho = 0.5$). The quality of the ML performance is highlighted in comparison to UR – 1 s, where error values of almost 35% are seen for the same utilization level. An aggregation of utilization samples does not prove to be as effective as in the single-class scenario, and UR – 30 req, as well as UR – 50 s are not very different from UR – 1 req. RR – 1 req provides better results than the UR methods. The distribution analysis presents similar results for $\rho = 0.5$. For example, in FIG. 8B, ML has a 95th-percentile of 17%, whereas RR – 30 req and UR – 30 req have 95th percentiles of 54% and 61% respectively.

[0074] FIG. 7C shows the fact that the quality of the results significantly changes when moving to a five class scenario. More specifically, the general accuracy decreases (note that the vertical scale is different for all figures) and especially in high load. This is expected as a result of the larger number of service demands to be estimated. UR – 1 s, UR – 30 req, and UR – 50 s have consistently large errors, with the best result among UR methods being achieved by UR – 1 s in the $\rho = 0.5$ case. Aggregation for UR again does not lead to accuracy increases. RR – 1 req delivers similar results in medium utilization levels, while RR – 30 req deteriorates the error value. The only method that delivers high-quality results, at least at low and medium utilization levels, is ML (less than 12% error). The heavy load case seems difficult for all methods. The CDF of the $K = 5$ scenario in FIG. 8C emphasizes the quality of ML, which achieves producing 95% of errors below 21%. RR – 30 req has a 95th-percentile of 111%, while for UR – 30 req it is 162%. For UR – 1 s (not shown in FIG. 7C) it is 56%. Consequently, ML is overall more robust.

[0075] The exemplar results provided in FIGs. 7A-7C and 8A-8C indicate that multi-class estimation within medium sized data sets, corresponding to sets of events in 600 seconds for different utilization values, is a challenging problem. ML robustly performs such estimations in almost all cases. The ML results are extremely good with limited

information (e.g., low utilization). RR – 1 req appears more effective than RR – 30 req in all experiments, and is generally competitive with the UR methods at low and medium load.

[0076] Referring now to FIG. 9, a flowchart illustrates exemplar steps that can be executed in accordance with the present disclosure. In step 902, a workload is generated, the workload including a plurality of service requests including a plurality of classes that can be processed by a computer system. The computer system can include an application server that executes an application, and one or more client systems that generate the requests. The workload is applied to the computer system in step 904, and a response time is measured for each request of the workload in step 906. In step 908, a mean service demand is estimated for each class based on the response times and a base queuing model. The base queuing model can include assumptions on characteristics of a final queuing model, the characteristics comprising at least one of scheduling and a service demand distribution.

[0077] Estimating a mean service demand can include estimating mean service demands in the base queuing model using one of linear regression and maximum likelihood method analyses based on the measured response times. Estimating a mean service demand can be further based on a plurality of arrival queue-lengths, the plurality of arrival queue-lengths corresponding to each request of the workload. Each arrival queue-length of the plurality of arrival queue-lengths can be determined from log files that report a time of arrival and departure of requests. Estimating a mean service demand can be further based on a plurality of residual times. The plurality of residual times can correspond to each request of the workload. Each residual time can correspond to a time remaining to complete processing of an in-process request upon arrival of a to-be-processed request

[0078] A queuing model is generated in step 910 based on the mean service demands and characteristics of the workload. Characteristics of the workload can include, but are not limited to, an inter-arrival time distribution of requests arriving into the queue such as exponential inter-arrival times having mean of 5 requests per second. Generating the queuing model can include parameterizing the base queuing model using the mean

service demands. In step 912, a performance of a computer system is evaluated by processing the queuing model using a plurality of inputs.

[0079] Referring now to FIG. 10, a schematic illustration of exemplar hardware components 1000 that can be used to execute implementations of the present disclosure is provided. The system 1000 can be used for the operations described in association with the methods described herein. For example, the system 1000 may be included in the application server system 106. The system 1000 includes a processor 1010, a memory 1020, a storage device 1030, and an input/output device 1040. Each of the components 1010, 1020, 1030 and 1040 are interconnected using a system bus 1050. The processor 1010 is capable of processing instructions for execution within the system 1000. In one implementation, the processor 1010 is a single-threaded processor. In another implementation, the processor 1010 is a multi-threaded processor. The processor 1010 is capable of processing instructions stored in the memory 1020 or on the storage device 1030 to display graphical information for a user interface on the input/output device 1040.

[0080] The memory 1020 stores information within the system 1000. In one implementation, the memory 1020 is a computer-readable medium. In one implementation, the memory 1020 is a volatile memory unit. In another implementation, the memory 1020 is a non-volatile memory unit. The storage device 1030 is capable of providing mass storage for the system 1000. In one implementation, the storage device 1030 is a computer-readable medium. In various different implementations, the storage device 1030 may be a floppy disk device, a hard disk device, an optical disk device, or a tape device. The input/output device 1040 provides input/output operations for the system 1000. In one implementation, the input/output device 1040 includes a keyboard and/or pointing device. In another implementation, the input/output device 1040 includes a display unit for displaying graphical user interfaces.

[0081] The features described can be implemented in digital electronic circuitry, or in computer hardware, firmware, software, or in combinations of them. The apparatus can be implemented in a computer program product tangibly embodied in an information carrier, e.g., in a machine-readable storage device, for execution by a programmable processor; and method steps can be performed by a programmable processor executing a

program of instructions to perform functions of the described implementations by operating on input data and generating output. The described features can be implemented advantageously in one or more computer programs that are executable on a programmable system including at least one programmable processor coupled to receive data and instructions from, and to transmit data and instructions to, a data storage system, at least one input device, and at least one output device. A computer program is a set of instructions that can be used, directly or indirectly, in a computer to perform a certain activity or bring about a certain result. A computer program can be written in any form of programming language, including compiled or interpreted languages, and it can be deployed in any form, including as a stand-alone program or as a module, component, subroutine, or other unit suitable for use in a computing environment.

[0082] Suitable processors for the execution of a program of instructions include, by way of example, both general and special purpose microprocessors, and the sole processor or one of multiple processors of any kind of computer. Generally, a processor will receive instructions and data from a read-only memory or a random access memory or both. The essential elements of a computer are a processor for executing instructions and one or more memories for storing instructions and data. Generally, a computer will also include, or be operatively coupled to communicate with, one or more mass storage devices for storing data files; such devices include magnetic disks, such as internal hard disks and removable disks; magneto-optical disks; and optical disks. Storage devices suitable for tangibly embodying computer program instructions and data include all forms of non-volatile memory, including by way of example semiconductor memory devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks. The processor and the memory can be supplemented by, or incorporated in, ASICs (application-specific integrated circuits).

[0083] To provide for interaction with a user, the features can be implemented on a computer having a display device such as a CRT (cathode ray tube) or LCD (liquid crystal display) monitor for displaying information to the user and a keyboard and a pointing device such as a mouse or a trackball by which the user can provide input to the computer.

[0084] The features can be implemented in a computer system that includes a back-end component, such as a data server, or that includes a middleware component, such as an application server or an Internet server, or that includes a front-end component, such as a client computer having a graphical user interface or an Internet browser, or any combination of them. The components of the system can be connected by any form or medium of digital data communication such as a communication network. Examples of communication networks include, e.g., a LAN, a WAN, and the computers and networks forming the Internet.

[0085] The computer system can include clients and servers. A client and server are generally remote from each other and typically interact through a network, such as the described one. The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other.

[0086] In addition, the logic flows depicted in the figures do not require the particular order shown, or sequential order, to achieve desirable results. In addition, other steps may be provided, or steps may be eliminated, from the described flows, and other components may be added to, or removed from, the described systems. Accordingly, other implementations are within the scope of the following claims.

[0087] A number of implementations of the present disclosure have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the present disclosure. Accordingly, other implementations are within the scope of the following claims.

1. A computer-implemented method of generating a queuing model, comprising:

defining a workload comprising a plurality of service requests, each service request corresponding to a class of a plurality of classes;

applying the workload to a computer system that receives and processes service requests;

measuring a response time of the computer system for each request of the workload;

estimating a mean service demand for each class based on the response times and a base queuing model that represents the computer system; and

generating the queuing model based on the mean service demands and characteristics of the workload.

2. The method of claim 1, further comprising determining a plurality of arrival queue-lengths corresponding to each request of the workload, wherein estimating a mean service demand is further based on the plurality of arrival queue-lengths.

3. The method of claim 2, wherein each arrival queue-length of the plurality of arrival queue-lengths is determined from log files that report a time of arrival and departure of requests.

4. The method of claim 1, further comprising determining a plurality of residual times corresponding to each request of the workload, wherein estimating a mean service demand is further based on the plurality of residual times.

5. The method of claim 1, wherein estimating a mean service demand comprises estimating mean service demands in the base queuing model using one of linear regression and maximum likelihood method analyses based on the measured response times.

6. The method of claim 1, wherein generating the queuing model comprises parameterizing the base queuing model using the mean service demands.

7. The method of claim 1, further comprising evaluating a performance of a computer system by processing the queuing model using a plurality of inputs.

8. A computer-readable storage medium coupled to one or more processors and having instructions stored thereon which, when executed by the one or more processors, cause the one or more processors to perform operations comprising:

defining a workload comprising a plurality of service requests, each service request corresponding to a class of a plurality of classes;

applying the workload to a computer system that receives and processes service requests;

measuring a response time of the computer system for each request of the workload;

estimating a mean service demand for each class based on the response times and a base queuing model that represents the computer system; and

generating the queuing model based on the mean service demands and characteristics of the workload.

9. The storage medium of claim 8, wherein the operations further comprise determining a plurality of arrival queue-lengths corresponding to each request of the workload, wherein estimating a mean service demand is further based on the plurality of arrival queue-lengths.

10. The storage medium of claim 9, wherein each arrival queue-length of the plurality of arrival queue-lengths is determined from log files that report a time of arrival and departure of requests.

11. The storage medium of claim 8, wherein the operations further comprise determining a plurality of residual times corresponding to each request of the workload,

wherein estimating a mean service demand is further based on the plurality of residual times.

12. The storage medium of claim 8, wherein estimating a mean service demand comprises estimating mean service demands in the base queuing model using one of linear regression and maximum likelihood method analyses based on the measured response times.

13. The storage medium of claim 8, wherein generating the queuing model comprises parameterizing the base queuing model using the mean service demands.

14. The storage medium of claim 8, further comprising evaluating a performance of a computer system by processing the queuing model using a plurality of inputs.

15. A system comprising:

a computer system that receives and processes service requests;

one or more processors; and

a computer-readable storage medium coupled to the one or more processors and having instructions stored thereon which, when executed by the one or more processors, cause the one or more processors to perform operations comprising:

defining a workload comprising a plurality of service requests, each service request corresponding to a class of a plurality of classes;

applying the workload to the computer system;

measuring a response time of the computer system for each request of the workload;

estimating a mean service demand for each class based on the response times and a base queuing model that represents the computer system; and

generating a queuing model based on the mean service demands and characteristics of the workload.

16. The system of claim 15, wherein the operations further comprise determining a plurality of arrival queue-lengths corresponding to each request of the workload, wherein estimating a mean service demand is further based on the plurality of arrival queue-lengths.

17. The system of claim 16, wherein each arrival queue-length of the plurality of arrival queue-lengths is determined from log files that report a time of arrival and departure of requests.

18. The system of claim 15, wherein the operations further comprise determining a plurality of residual times corresponding to each request of the workload, wherein estimating a mean service demand is further based on the plurality of residual times.

19. The storage medium of claim 15, wherein estimating a mean service demand comprises estimating mean service demands in the base queuing model using one of linear regression and maximum likelihood method analyses based on the measured response times.

20. The storage medium of claim 15, wherein the computer system comprises an application server that executes an application, and one or more client systems that generate the requests.

1 Abstract

Implementations of the present disclosure provide computer-implemented methods including defining a workload comprising a plurality of service requests, each service request corresponding to a class of a plurality of classes, applying the workload to a computer system that receives and processes service requests, measuring a response time of the computer system for each request of the workload, estimating a mean service demand for each class based on the response times and a base queuing model that represents the computer system, and generating the queuing model based on the mean service demands and characteristics of the workload.

2 Representative Drawing

Fig. 1

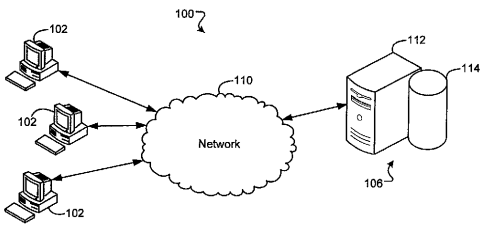


FIG. 1

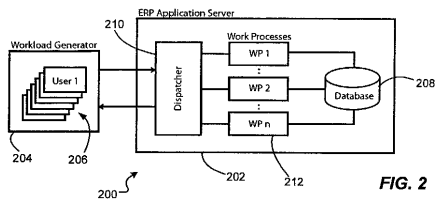


FIG. 2

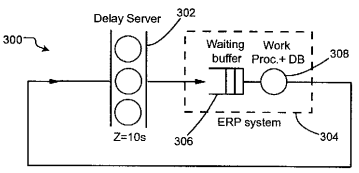


FIG. 3

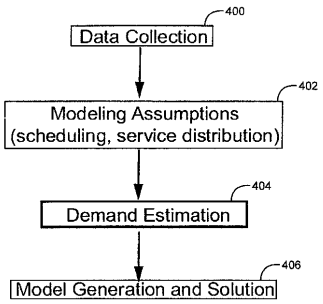


FIG. 4

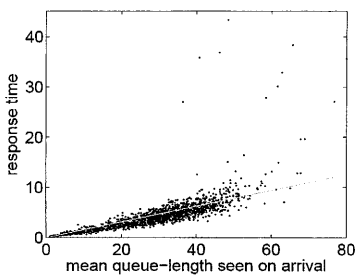


FIG. 5

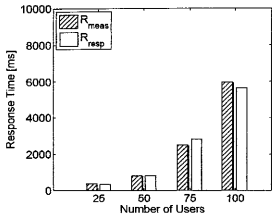


FIG. 6A

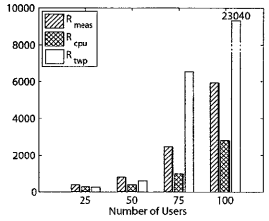


FIG. 6B

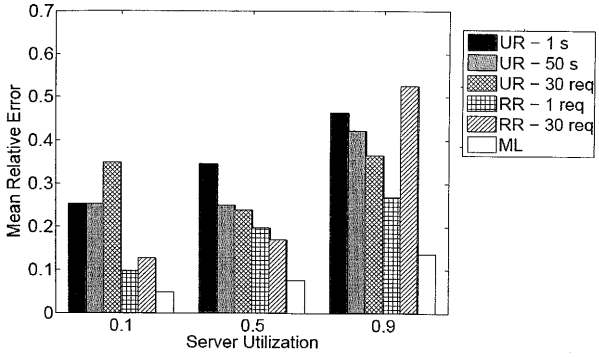


FIG. 7B

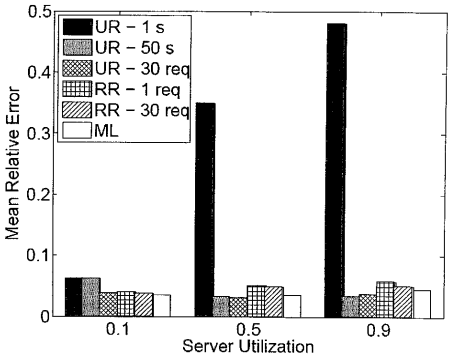


FIG. 7A

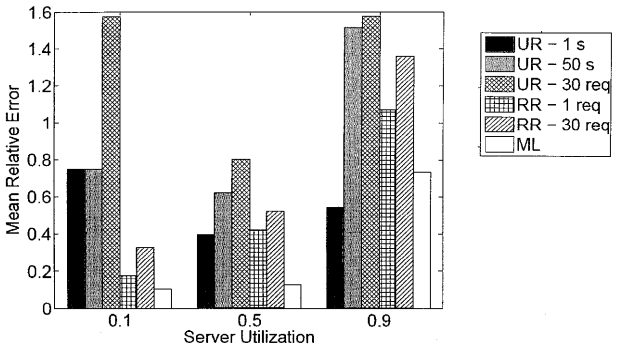


FIG. 7C

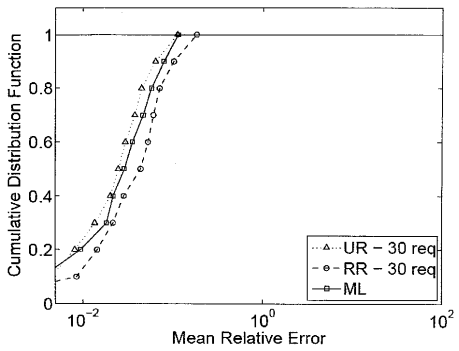


FIG. 8A

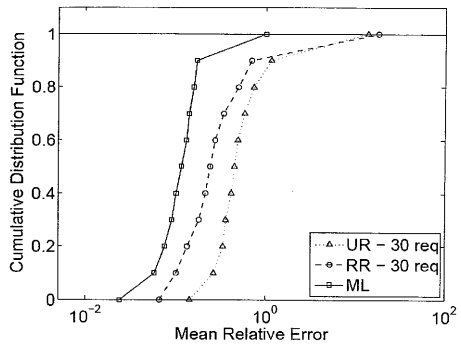


FIG. 8C

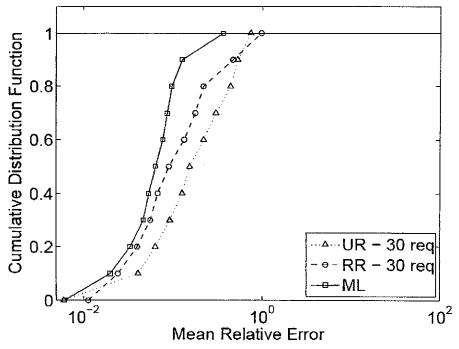


FIG. 8B

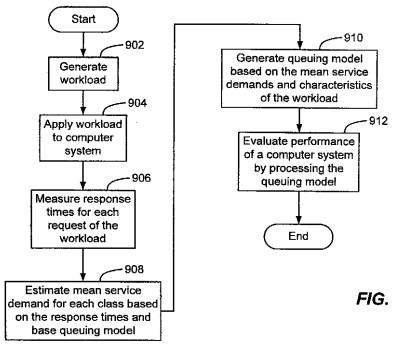


FIG. 9

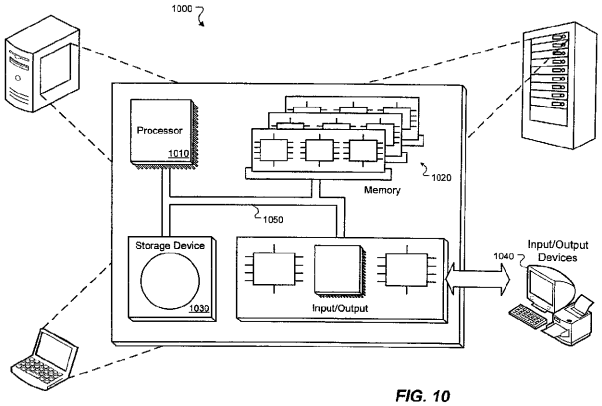

1000
Processor
1010
1020
Memory
1050
Storage Device
1030
Input/Output
1040
Input/Output
Devices

FIG. 10