Vol.2011-EVA-36 No.3
2011/12/2

詳細が未知の部分を含むシステムの性能評価モデル作成手法の提案

木村大地[†]　沼田絵梨子[†]　榊啓[†]　矢野尾一男[†]

情報処理システムの性能評価モデルには，待ち行列等のシステムの動作を模倣したモデルによって挙動を予測するホワイトボックス的なアプローチと，システムの実測値から機械学習等によってモデルを構築し挙動を予測するブラックボックス的なアプローチがある．本稿では，これらのアプローチを組み合わせ，詳細が未知の部分を含むシステムの性能評価モデルを作成する手法を提案した．実際に運用しないと分からない未知の部分はブラックボックスとして扱うことで，予測の精度を担保し，動作の詳細が既知の部分はホワイトボックスとして扱い，その部分を他のスペックのものに置き換えた場合の性能への影響評価を可能とする．各部分の性能が計測できる場合と，各部分の性能の計測が困難で，全体の性能のみ計測できる場合について，それぞれ論じた．前者の場合では，ホワイトボックスとして評価される部分の置き換えに対して性能評価ができたことを示した．後者の場合では，システム全体の性能の実測値を用いた強化学習によって，ブラックボックスとして評価される部分の実測値を用いずに，間接的にブラックボックスモデルの構築ができたことを示した．

Method for Making Performance Model about System which Includes Unknown Part

D. Kimura[†], E. Numata[†], H. Sakaki[†], K. Yanoo[†]

There are two approaches for performance evaluation of an IT system; white-box approach imitates mechanism of the system such as queuing theory and black-box approach employs machine learning using the measurements of the system. In this paper, we propose a method for making a performance model about a system which includes an unknown part by combining these two approaches. Treating unknown parts, of which we cannot know the details without operation, as black-box ensures the accuracy of the prediction. Treating well-known parts as white-box enables to evaluate the performance influence of replacing these parts with different ones. We have applied our method to a case in which we can measure the performance of each part and a case in which we can measure only the performance of the “whole” system. In the former case, our method can evaluate the performance for replacing white-box part. In the later case, our method can make a black-box for unknown part indirectly by reinforcement learning using the measurement of the “whole” system, where we do not use the measurement of each part.

1. はじめに

情報処理システムが基幹的なインフラとしての役割を担う場面が増えてきている．それに伴い，情報処理システムが期待通りの性能を出せるかどうかを評価することの重要性はますます高まってきている．

情報処理システムの性能を評価するモデルは，ホワイトボックス的なアプローチとブラックボックス的なアプローチの 2 種類がある．前者は，待ち行列に代表されるようなシステムの動作を模倣したモデルを作成することによって挙動を予測する．後者は，システムの実測値から機械学習や統計処理によってモデルを構築することで挙動を予測する．これら 2 つのアプローチは排他的なものではなく，組み合わせることによってより高精度な性能予測が可能になるものと期待される [1)].

本稿では，これら 2 つのアプローチを組み合わせることにより，詳細が未知の部分を含むシステムの性能評価モデルを作成する手法を提案する．

2. ホワイトボックスモデルとブラックボックスモデル

上記のように，性能評価のモデルは 2 種類ある．

ホワイトボックスモデル(WB)の長所は，システム動作のメカニズムが明示的にモデル化されているため，パラメタ（ここでは，システムのスペック等）の変更に対するモデルの感度（ここでは，モデルが予測する性能の変化）が明らかであることである．短所は，システムのメカニズムのモデル化には，システムに対する知識が求められることである．精度を上げるためには，システムの細部までモデル化する必要があるが，システムの細部についての情報が必ずしも明らかであるとは限らない．また，システムの細部が分かっていても，動作を細かく反映したモデルを作成しようとすると膨大な時間がかかる場合もある．モデルがもたらす効果と作成にかかる時間はトレードオフの関係にある．さらに，モデルがシステムの動作を正しく模倣していない，すなわち，モデルを作成する仮定である種の間違いを含んでしまう可能性がある．

ブラックボックスモデル(BB)の長所は，実測値，すなわち，実際のシステムの入出力だけを用いてモデルを作成するため，システムの細部についての知識を必要としない．また，実際のシステムの入出力がモデルに反映されているので，WB のように間違いを原理的には含み得ない．短所は，そのモデルの作成に用いられた実測値の変化範囲内でしかモデル化できない（実際に計測していないシチュエーションは予測できない）．すわなち，用いた実測値で変化していないパラメタ（システムのスペック等）については，そのパラメタの変更に対するモデルの感度（予測される性能の変化）が分からないことを意味する．

[†] ＮＥＣ　サービスプラットフォーム研究所
Service Platforms Research Labs., NEC

Vol.2011-EVA-36 No.3
2011/12/2

3. 提案手法

WB と BB を組み合わせることによって，詳細が未知の部分を含むシステムの性能評価モデルを作成する手法を提案する．具体的には，システムの各部分について，動作の詳細が既知の部分（あるいは，モデル化が容易な部分，といっても良い）は WB，動作の詳細が未知の部分は BB として評価する．利点は，実際に運用しないと分からない部分は BB にすることで，予測の精度を担保し，そうでない部分については WB とすることで，その部分を他のスペックのものに置き換えた場合の性能への影響を評価できるようにすることである．

以下では，次の 2 ケースについて論じる．

1. 各部分の性能が計測できる場合
2. 各部分の性能の計測が困難で，全体の性能のみ計測できる場合

上記 2 ケースの違いは，主に BB の取り扱い方に現れる．前者のケースでは，未知の部分の性能が直接計測できるので，その計測した値を用いて BB を構築すればよい．後者のケースでは，未知の部分を BB とした場合にその性能は直接計測できないので，システム全体の性能を用いて間接的に BB を構築する必要がある．

3.1 各部分の性能が計測できる場合

各部分の性能が計測できる場合について，WEB3 層システムへの適用例を述べる．図 1 に概略を示す．

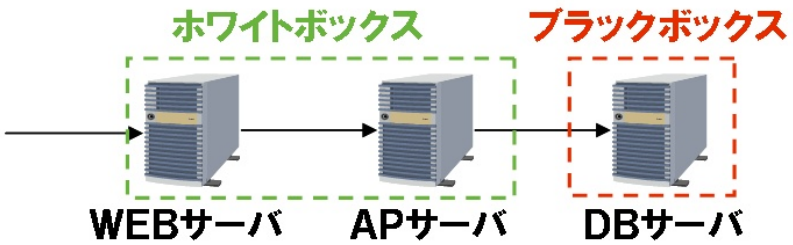


図 1 提案手法の WEB3 層システムへの適用例

ここでは，WEB，AP サーバの性能については WB として評価し， DB サーバの性能については BB として評価する．このように評価することによって，DB サーバが比較的複雑な振る舞いを示しても性能予測の精度は担保される一方で，WEB，AP サーバをスペックの異なるマシンへ置き換えた場合（パラメタを変更した場合）の性能評価を可能とする．

実際に WEB3 層システムを構築し，提案手法による性能評価の検証を行った．AP サーバを異なるスペックのマシンに置き換え，性能が予測できるかを確かめた．

本稿では，AP サーバの CPU 使用率を WB として予測し，DB サーバのディスク使用率を BB として予測した結果を示す．構築したシステムでは，クライエントからリクエストを受け取ると，AP サーバに実装したアプリケーションは，リクエストによって指定された回数だけ乱数を用いた計算を行った後，DB サーバから指定されたファイルをダウンロードする．指定された計算回数が多いほど，AP サーバの CPU にかかる負荷は大きくなる．図 2 に，AP サーバの CPU 負荷の違いによる DB サーバのディスク使用率を示す．

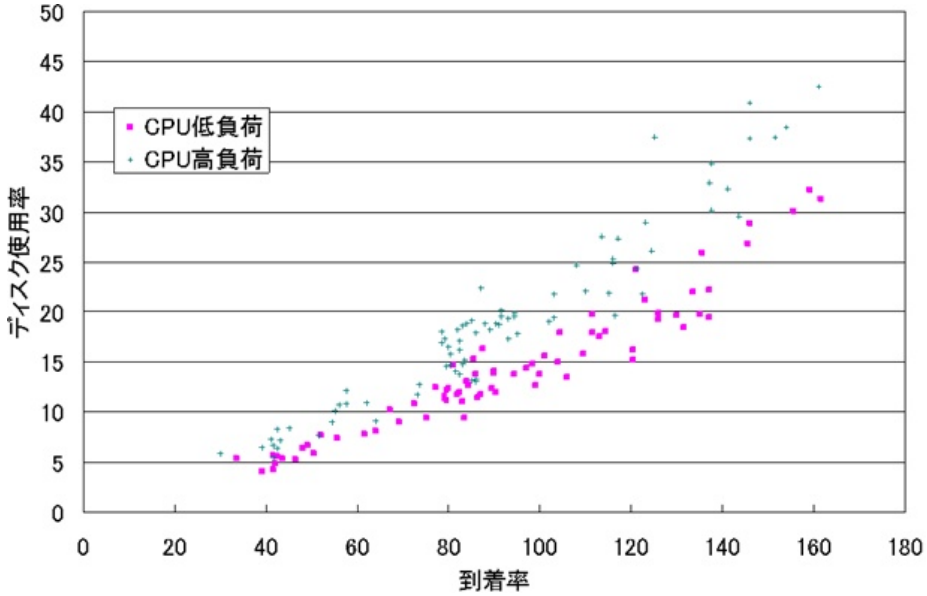


図 2 AP サーバの CPU 負荷の違いによる DB サーバのディスク使用率

横軸はリクエストの到着率，縦軸は DB サーバのディスク使用率を表す．図 2 に示されるように，このシステムでは，AP サーバの CPU 負荷の大きさが，DB サーバのディスク使用率と相関があると考えられる．ここでは，リクエストの到着率と AP サーバの CPU 使用率を入力変数，DB サーバのディスク使用率を出力変数として，（AP サーバを置き換える前に）計測したデータから BB として評価式を導いた．具体的には，上記の入出力変数についての回帰式を薄板平滑化スプラインによって導いた．他方で，AP サーバの CPU 使用率は，1 リクエストあたりの CPU 使用率を求め，到着率の比例

式として WB として評価した．

上記のシステムで，AP サーバを異なるスペックのマシンに置き換えた場合の性能を評価した．AP サーバの CPU 使用率を WB として予測するに際し，「異なるスペック」を WB の言葉で表現すると，「1 リクエストあたりの CPU 使用率」というパラメタが変更されるということになる．WB の予測値と実測値を比較した結果を図 3 に示す．図に示されるように WB としてよく予測できていることが分かる．

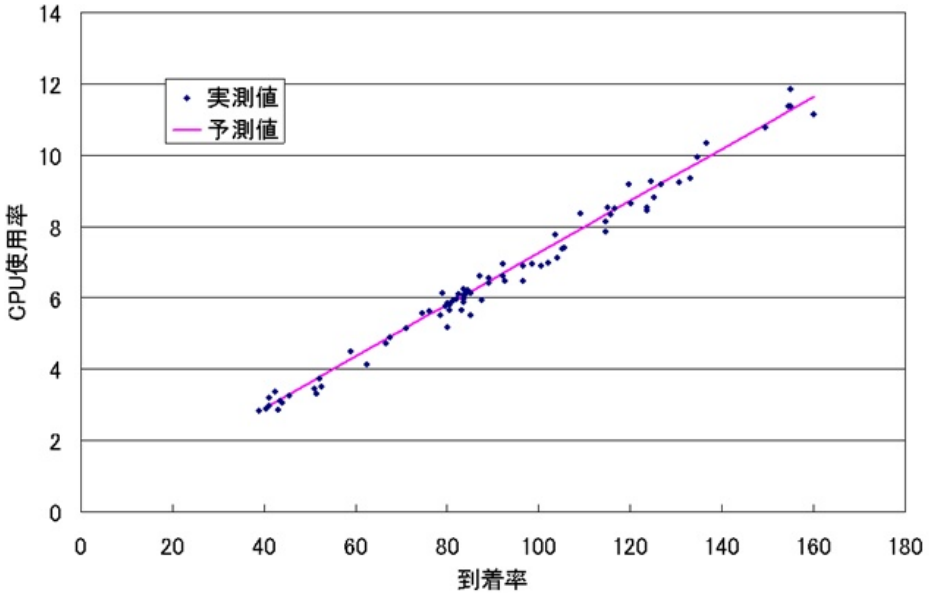


図 3　AP サーバの CPU 使用率

DB サーバのディスク使用率の予測には，既に構築している BB を用いる．BB への入力は，前記の予測した AP サーバの CPU 使用率とリクエストの到着率である．BB の予測値と実測値を比較した結果を図 4 に示す．到着率が 120 以下ではおおよそ予測できている．到着率が 120 以上で予測が外れてしまうのは，BB の入力変数として，到着率と AP サーバの CPU 使用率以外の要因がこの領域では影響を持ってくるためと考えられる．このように，WB で評価した部分の置き換えに対して，性能が評価できたことが示された．

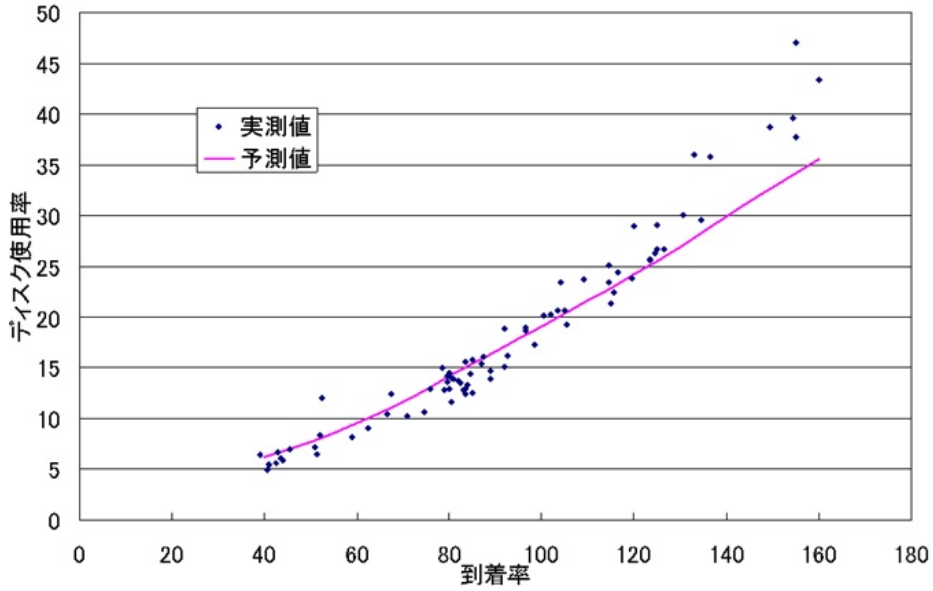


図 4　DB サーバのディスク使用率

3.2　各部分の性能の計測が困難で，全体の性能のみ計測できる場合

各部分の性能が計測できない場合には，前節のような未知の部分の性能の実測値を直接用いて BB を構築する方法は適用できない．そこで，システム全体の性能を用いて間接的に BB を構築するために，強化学習の枠組みを用いた．

強化学習では，学習するエージェントの挙動に応じてエージェントに報酬が与えられる．エージェントは報酬を最も獲得できるように挙動を学習する．通常の教師有り学習では直接的に正解情報が与えられるのに対して，強化学習では報酬という間接的な情報のみが与えられる．強化学習は，学習させたい課題に対してそもそも正解情報が分かっていない場合に有効な方法である．強化学習の 1 つの適用例として，倒立振り子問題を説明する．倒立振り子問題は，図 5 のように振り子を取り付けられた車が，振り子を倒さないように車の速度を調整するという問題である．

Vol.2011-EVA-36 No.3
2011/12/2

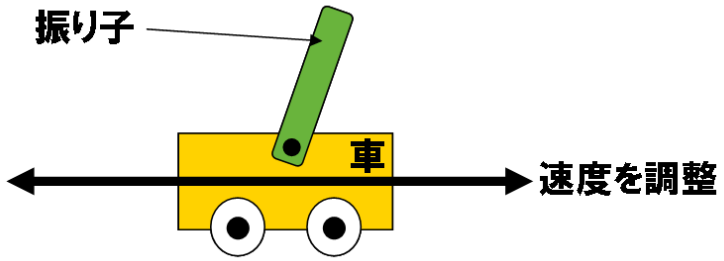


図 5　倒立振り子問題

車の速度をどう調整すればよいかという正解情報は，複雑な運動方程式を解かない限りわからない．すなわち，正解情報を用意するのは簡単ではない．そこで，車の速度を調整するエージェントに，強化学習によって正解情報を学習させる．エージェントは，振り子を倒さずに安定させることができると報酬を獲得することができるとする．例えば，報酬 R は次のような関数で定義される．

$$R=\begin{cases}1:\text{一定時間、振り子を維持した場合}\\0:\text{振り子を倒した場合}\end{cases}$$

エージェントは上記の報酬がもらえるように車の速度を調整，すなわち，正解情報を学習する．

上記の強化学習を WB と BB を組み合わせた性能評価モデルに適用する．ここでは，未知の部分の正確な挙動という「正解情報」が分からないので，システム全体の性能の実測値と評価モデルの予測値の誤差を「報酬」に用いることで，未知の部分に対応する BB（強化学習のエージェントに相当する）を学習させる．例えば，報酬 R は次のような関数で定義される．

$$R=\begin{cases}1:\text{予測値と実測値の誤差が一定範囲より小さい}\\0:\text{予測値と実測値の誤差が一定範囲より大きい}\end{cases}$$

BB は上記の報酬がもらえるように挙動を調整，すなわち，正解情報を学習する．

本稿ではシミュレーションに提案手法を適用して結果を検証した．シミュレーションは，図 6 に示すような 3 つのリソースが直列に結合されているシステムを模倣している．

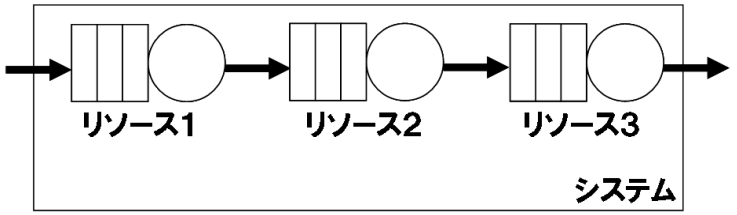


図 6　シミュレーションするシステム

システムには，平均到着率 λ でリクエストが到着し，各リソースは FIFO でリクエストを処理する．リソース 1 及び 3 の平均サービス時間は平均 D_1 及び D_3 で既知の値である．リソース 2 の平均サービス時間 D_2 はリソース 2 のスループット X_2 に依存し，その具体的な関数形 $D_2=D_2(X_2)$ は未知である．このシステムでは，リクエストの到着率やシステム全体の TAT は計測可能である．しかし，各リソースのスループットやサービス時間などは計測できない．

上記のシステムに対する性能評価モデルを次のように構築する．リソース 1，3 は平均サービス時間 Di(i=1,3)が既知なので，G/M/1 の待ち行列として，WB で評価する．リソース 2 は同じく G/M/1 の待ち行列ではあるが，平均サービス時間 D_2 が未知なので，この D_2 を BB として評価する．

D_2 は直接計測できないので，強化学習に則って，計測可能であるシステム全体の TAT を用いて D_2 を調整する．ここでは，D_2 の調整手段として図 7 のようなニューラルネットワークを用いる．

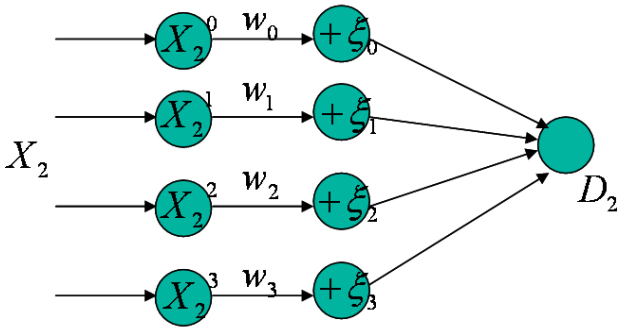


図 7　ニューラルネットワーク

Vol.2011-EVA-36 No.3
2011/12/2

ここで，w_i はシナプス荷重，ξ_i は平均 0，標準偏差 Q_i のガウシアンノイズである．D_2 はリソース２のスループット X_2 の関数なので，X_2 がニューラルネットワークの第１層に入力される．第 1 層では X_2 のべき乗が計算される．第 2 層でノイズ ξ_i が加算された後，最終層で全てが合計されて D_2 の予測値となる．このニューラルネットワークは下記のような式で表現することができる．

$$D_2(X_2) = (w_0 {X_2}^0 + \xi_0) + (w_1 {X_2}^1 + \xi_1) + (w_2 {X_2}^2 + \xi_2) + \cdots$$

すなわち，X_2 のべき乗でフィッティングするような BB である．シナプス荷重を変更することにより，D_2 の予測値は調整される．報酬がより多く獲得できるようなシナプス荷重の学習則は次のように導出できる [2]（導出の詳細は A.1 節を参照）．

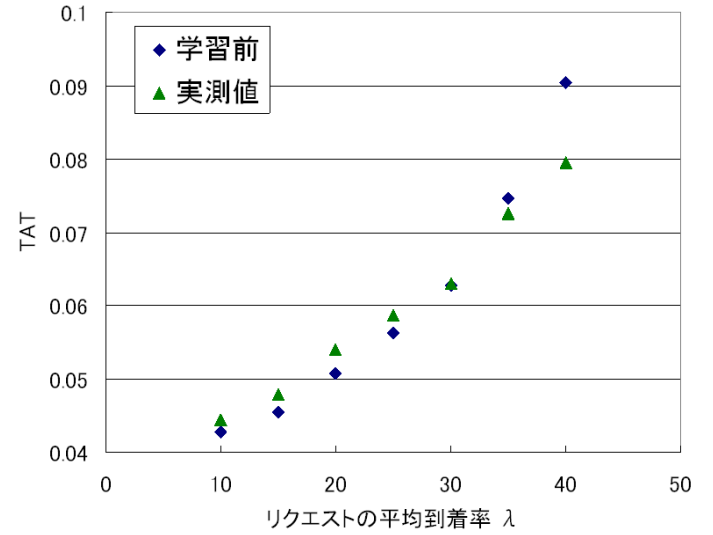
$$\Delta w_i = \varepsilon \frac{\partial \langle R \rangle}{\partial w_i} = \frac{\varepsilon}{Q_i} \left\langle R \cdot {X_2}^i \cdot \xi_i \right\rangle$$

ここで，ε は学習率，<…>は期待値を表す．報酬 R は，システム全体の TAT の性能評価モデルによる予測値と実測値の誤差を用いて次のように定義する．予測値と実測値の誤差を E_ξ とし，ノイズを 0 にした場合の性能評価モデルによる予測値と実測値の誤差を E_0 としたとき，

$$R = \begin{cases} 1 & (E_\xi < E_0) \\ 0 & (E_\xi \geq E_0) \end{cases}$$

とする．すなわち，ノイズがあった場合のほうが誤差が小さければ報酬が獲得でき，ノイズがない場合のほうが誤差が小さければ報酬が獲得できない．

学習前のシステム全体の TAT および D_2 の予測値と実測値を図 8 に示す．ここで，D_2 (X_2)の初期値はスループットに依存しない定数（w_0=0.018, w_i=0 for i > 0）とおいた．このように D_2 をスループットに依存しない定数とおくと，システム全体の TAT について予測値と実測値がずれている．強化学習による学習後の結果を図 9 に示す．図に示されるように，強化学習によってシステム全体の TAT の予測値と実測値が正しく予測できている．また，TAT を正しく予測するように，すなわち，報酬を獲得できるように D_2 を調整した結果，真の値とほぼ同じになっていることがわかる．このように，システム全体の性能を用いた強化学習によって，詳細が未知の部分についての BB の構築ができたことが示された．



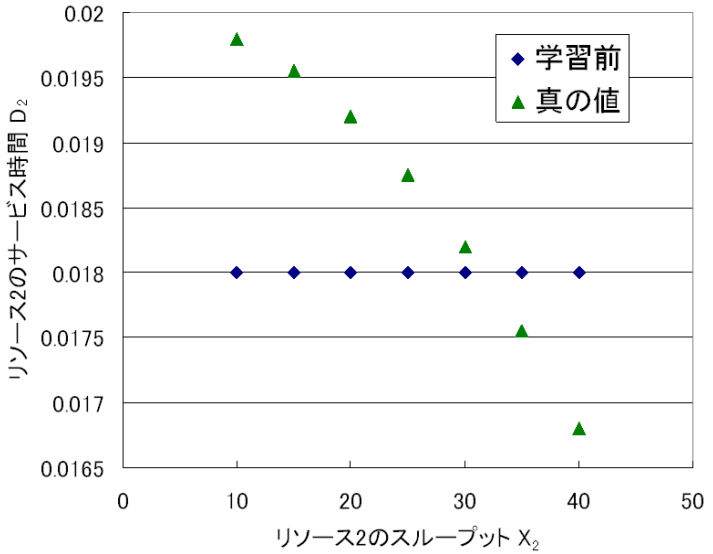
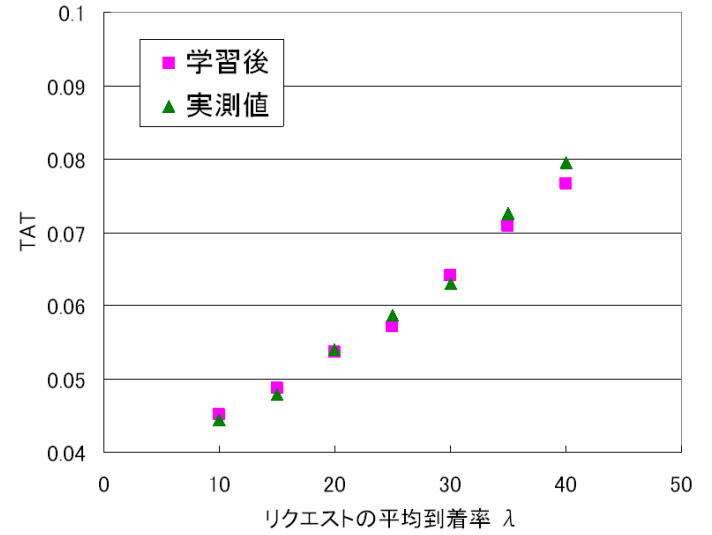


図 8　学習前



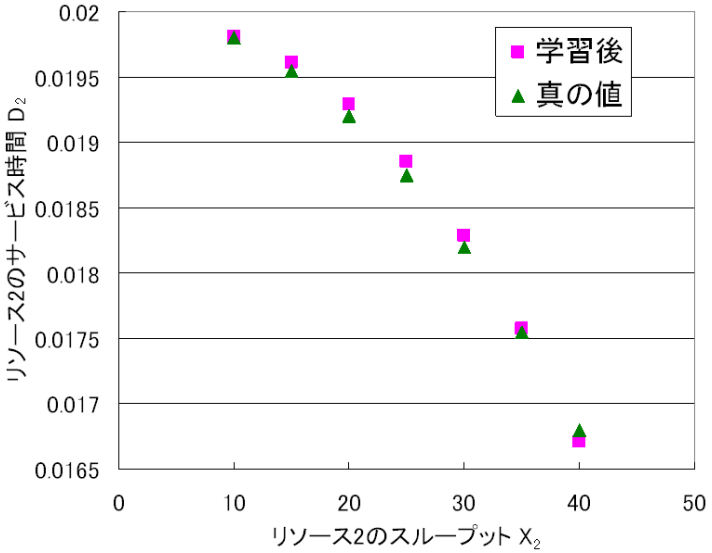


図 9　学習後

4. 関連研究

WB と BB は完全に区別できるものではない [3]．WB では，モデルに含まれるパラメタを決定するために何らかの実測値が用いられる．BB では，いくつかの事前知識によって機械学習の方式を選択する．しかしながら 4)では，これら 2 つのアプローチはそれぞれ独立して研究されてきており，WB と BB の融合がよりよい性能評価モデルをもたらすと主張している．

WB と BB を融合した手法の 1 つとして，WB に含まれるパラメタの推定に，実測値にカルマンフィルタなどの統計処理・機械学習を適用した結果を用いる方法が提案されている．例えば，各リソースのサービス時間 [5][6][7][8]やリクエストの到着率分布 [9]，また，性能評価以外では信頼性評価 [10]への適用例がある．別の融合手法として，WB の入出力を用いて機械学習により BB を構成し，その BB を基に実測値を用いて改めて機械学習を行ない BB を改善する方法が提案されている [11][12]．最初から実測値を用いて機械学習を行なうと，学習中は予測精度が低いという問題が起こりうるが，WB の入出力によってあらかじめ訓練しておくことにより予測精度をある程度確保するという効果が見込まれる．これは，特にオンラインで予測結果が求められる場合に有効である．

提案手法は，既知の部分を WB，未知の部分を BB としてモデル化している．先行研究は未知のパラメタという定数を推定しているが，提案手法は未知の部分という関数を推定しているといえる．この意味において，提案手法は先行研究よりも柔軟な適用が可能である．また，本稿ではオフライン型の強化学習を用いたが，オンライン型の強化学習に拡張することは容易であり，動的な性能評価へも適用可能であろう．

5. まとめ

本稿では，WB と BB を組み合わせることによって，詳細が未知の部分を含むシステムの性能評価モデルを作成する手法を提案した．各部分の性能が計測できる場合と，各部分の性能の計測が困難で，全体の性能のみ計測できる場合について，それぞれ論じた．前者の場合では，WB として評価される部分の置き換えに対して性能評価ができたことを示した．後者の場合では，システム全体の性能の実測値を用いた強化学習によって，BB の構築ができたことを示した．

後者の場合については，未知の部分が 2 つ以上ある場合にも適用できるかを検証する必要がある．報酬という全体に対する評価のみを用いて BB を構築するので，2 つ以上の BB を含む場合では，正しく構築できない可能性がある．

また，階層性がある性能評価モデルへ提案手法が適用可能かどうかも検討する必要があるだろう．5)では，階層的待ち行列モデルにカルマンフィルタの手法を用いてサ

ービス時間を推定している．既に述べたように，強化学習を用いた提案手法はカルマンフィルタよりも柔軟に適用することが可能であり，オンラインへの拡張も容易である．これらの実証による提案手法の適用評価が今後の課題である．

参考文献

1) Murray Woodside, Greg Franks, Dorina C. Petriu, “The Future of Software Performance Engineering,” Proc. Future of Software Engineering (FOSE'07), 2007.
2) Daichi Kimura, Yoshinori Hayakawa, “Reinforcement learning of recurrent neural network for temporal coding,” Neurocomputing, vol. 71, pp. 3379-3386, 2008.
3) Heiko Koziolek, “Performance evaluation of component-based software systems: A survey,” Performance Evaluation, Vol. 67, pp. 634-658, 2010.
4) Murray Woodside, “Performance Data and Performance Models”, Proc SPEC Int. Performance Evaluation Workshop (SIPEW2008), 2008.
5) Murray Woodside, Tao Zheng, Marin Litoiu, “Service System Resource Management Based on a Tracked Layered Performance Model,” Proc. IEEE International Conference on Autonomic Computing, pp. 175-184, 2006.
6) Qi Zhang, Ludmila Cherkasova, Evgenia Smirni, “A Regression-Based Analytic Model for Dynamic Resource Provisioning of Multi-Tier Applications,” Proc. Fourth International Conference on Autonomic Computing (ICAC'07), pp. 27-36, 2007.
7) Tao Zheng, Marin Litoiu, Murray Woodside, “Integrated Estimation and Tracking of Performance Model Parameters with Autoregressive Trends,” Proc. International Conference on Performance Engineering (ICPE’11), pp. 157-166, 2011.
8) Hamoun Ghanbari, Marin Litoiu, Cornel Barna, Murray Woodside, Tao Zheng, Johnny Wong, Gabriel Iszlai, “Tracking Adaptive Performance Models Using Dynamic Clustering of User Classes,” Proc. International Conference on Performance Engineering (ICPE’11), pp. 179-188, 2011.
9) Diwakar Krishnamurthy, Jerry Rolia, Min Xu, “WAM – The Weighted Average Method for Predicting the Performance of Systems with Bursts of Customer Sessions,” HP Labs. Technical Reports, HPL-2008-66, 2008.
10) Ilenia Epifani, Carlo Ghezzi, Raffaela Mirandola, Giordano Tamburrelli, “Model Evolution by Run-Time Parameter Adaptation,” Proc. the 31st International Conference on Software Engineering (ICSE '09), pp. 111-121, 2009.
11) Gerald Tesauro, Nicholas K. Jong, Rajarshi Das, Mohamed N. Bennani, “A Hybrid Reinforcement Learning Approach to Autonomic Resource Allocation,” Proc. 2006 IEEE International Conference on Autonomic Computing, pp. 65-73, 2006.
12) Eno Thereska, Gregory R. Ganger, “IRONModel: Robust Performance Models in the Wild,” Proc. the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems (SIGMETRICS’08), pp. 253-264, 2008.

付録

付録 A.1 学習則の導出

図 10 で示すようなニューラルネットワークの学習則を以下のように導出する [2].

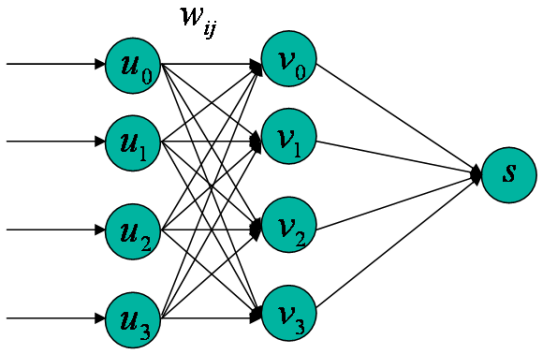


図 10　ニューラルネットワーク

入力層のニューロンの内部状態 u_i は外部から与えられる値である．中間層および出力層のニューロンの内部状態 v_i , s は次のように定義される．

$$v_i = \sum_j w_{ij} f_j(u_j) + \xi_i$$

$$s = \sum_i v_i$$

ここで，w_{ij} はシナプス荷重，f_j は u_j に基づいた出力関数，ξ_i は平均 0，標準偏差 Q_i のガウシアンノイズである．s の誤差を直接観測できないような場合には，シナプス荷重の学習則としてバックプロパゲーションを用いることはできない．ここでは，強化学習を用いる．

中間層のニューロンの内部状態にノイズが含まれるので，各ニューロンは確率的に様々な状態を取りうる．そのような確率的に取りうる状態を σ で表すと，その σ に対して報酬 R(σ)が定義される． σ が実現する確率を T_σ とすると報酬の期待値は

$$\langle R \rangle = \sum_\sigma R(\sigma) T_\sigma$$

で表される．学習則は，上記の<R>に基づいた勾配降下法に従って，学習率 ε を用いて w_{ij} を下記のように更新すればよい．

$$w_{ij}^{New} = w_{ij}^{Old} + \varepsilon \frac{\partial \langle R \rangle}{\partial w_{ij}}$$

<R>の勾配は次のように計算される．確率 T_σ は次のように書ける．

$$T_\sigma = \prod_i P(v_i^\sigma \mid u_0^\sigma, u_1^\sigma, \cdots)$$

ここで，P(A|B)は事象 B の元で事象 A が起こる条件付確率，u_i^σ，v_i^σ は状態 σ の元での各ニューロンの内部状態である．ところで，

$$v_i^\sigma = \sum_j w_{ij} f_j(u_j^\sigma) + \xi_i^\sigma$$

より，ノイズ ξ_i^σ の取る値によって状態 σ が決まる．またノイズはガウス分布に従うので，

$$P(v_i^\sigma \mid u_0^\sigma, u_1^\sigma, \cdots) = P(\xi_i^\sigma) = \frac{1}{\sqrt{2\pi Q_i}} \exp\left[-\frac{{\xi_i^\sigma}^2}{2Q_i} \right]$$

$$T_\sigma = \frac{1}{(2\pi)^{N/2} \prod_i \sqrt{Q_i}} \exp\left[-\sum_i \frac{{\xi_i^\sigma}^2}{2Q_i} \right]$$

$$= \frac{1}{(2\pi)^{N/2} \prod_i \sqrt{Q_i}} \exp\left[-\sum_i \frac{1}{2Q_i} \left\{ v_i^\sigma - \sum_j w_{ij} f_j(u_j^\sigma) \right\}^2 \right]$$

ここで，N は中間層のニューロンの数である．よって，

$$\frac{\partial \langle R \rangle}{\partial w_{ij}} = \sum_\sigma T_\sigma R(\sigma) \frac{1}{Q_i} \left\{ v_i^\sigma - \sum_j w_{ij} f_j(u_j^\sigma) \right\} f_j(u_j^\sigma)$$

$$= \frac{1}{Q_i} \left\langle R \xi_i f_j(u_j) \right\rangle$$

図 7 のニューラルネットワークに上式を適用するには，$u_j = X_2$, $f_j(x) = x^j$, $w_{ij} = 0$ (for $i \neq j$) とすればよい．