7T-02

機械学習による一般トラヒック型待ち行列システムの性能評価

二井 克 †　　宇都宮　陽一 ‡　　奥田　隆史 †

愛知県立大学　情報科学部　情報科学科 †　愛知県立大学　大学院　情報科学研究科 ‡

1　はじめに

IoT（Internet of Things，モノのインターネット）が急速に普及している．IoT で想定しているインターネットに接続されるモノ (以後 **IoT デバイス**) にはパソコンやスマホなどの情報通信機器だけではなく，各種センサを備えた自動車，工場設置機器，医療機器なども含まれる [1]．シスコシステムズは 2020 年には 500 億台の IoT デバイスが接続されると予測している [2]．

これらの IoT デバイスから効率的にデータを収集し活用するためには，データを処理するためのシステム（以後 **IoT データ処理システム**）をどのような形態で設計するかが重要である．IoT データ処理システムへ到着するデータ（以後**到着データ**）は多様かつ大量である．このため到着データの到着間隔にはバースト性が存在する．また到着データのサイズも多様である．したがって IoT データ処理システムの設計には，到着が一般到着間隔分布（GI），処理が一般処理時間分布（G）に従うサーバーが s 個の一般トラヒック型待ち行列 (以後 **GI/G/s システム**) の性能評価が不可欠である．

GI/G/s システムには，アーラン B 式や C 式 [5] に相当するような厳密解は存在しない [6, 7]．そのため性能評価には離散シミュレーションならびに，その結果をまとめた数表（Queueing Tables）[8] が利用されてきた．しかしながら離散シミュレーションには多大な計算時間を要する．また全ての条件に対応する数表を作成することは現実的ではない．

そこで本研究では，教師付き機械学習 [9] を用いた GI/G/s システムの性能評価をおこなうことを提案する．以下，第 2 節では想定する GI/G/s システムについて，第 3 節では機械学習を用いた性能評価について述べる．第 4 節では数値例を示す．第 5 節でまとめる．

2　想定する GI/G/s システム

本研究では，図 1 に示す IoT データ処理システムを想定する．この IoT データ処理システムを，複数の IoT デバイスから発生するデータの処理をおこなう GI/G/s システムとして捉える．ここで到着データは到着率 λ，平方変動係数 Ca^2 の一般分布で到着する．到着データは到着順 （FIFO）で，処理率 μ，平方変動係数 Cs^2 の一般分布で処理される．サーバー数は s 個とし，各サーバーの性能は同一とする．バッファサイズは有限または無限とする．

なお到着間隔および処理時間の分布は，平方変動係数 C^2 の値に応じて決定する．具体的には $C^2 = 0$ のとき一定分布，$0 < C^2 < 1, C^2 = k^{-1}$ のとき k 次のアーラン分布，$C^2 = 1$ のとき指数分布 (記号 M)，$C^2 > 1$ のとき 2 次の超指数分布に従うものとして扱う [10]．

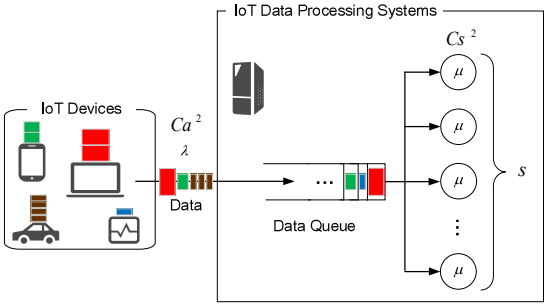

図 1　IoT データ処理システム

3　機械学習による GI/G/s システムの性能評価

本節では，教師付き機械学習による GI/G/s システムの性能評価について説明する．本研究で利用する機械学習はニューラルネットワーク (以後 **NN**) とする．NN を利用する理由は開発環境が整っていることと，情報処理速度を高めることが可能だからである [11]．

提案する性能評価手法の手順は以下の通りである．

Step1: 教師データを作成する．

Step2: NN を構築する．

Step3: 教師データを用い構築 NN を学習させる．

Step4: 学習完了した NN に，評価対象の GI/G/s システムを規定するパラメータを入力し，性能評価をおこなう．

ここで Step1 で作成する教師データの構成は**構成 (1)** と**構成 (2)** がある．構成 (1) は厳密解を有する特殊な GI/G/s システムのパラメータとその評価値 [12] から成る．構成 (2) は GI/G/s システムのパラメータとシミュレーションにより求めるその評価値から成る．なお構成 (1) と構成 (2) は同時に使用しない．

4　数値例

本研究では，表 1 の数値で規定される GI/G/s システムの性能評価をおこない平均システム内時間 W を算出する．ただしバッファサイズは無限とし，利用率は $\rho(= \lambda/s\mu)$ とする．

NN は入力，中間，出力の 3 層から成り，各層のニューロン数は 4，5，1 とする．入力ニューロン数 4 は Ca^2, Cs^2, s, ρ，出力ニューロン数 1 は W に対応する．NN の実装は Python ライブラリ chainer[14] を使用する．

構成 (1) は表 2 の数値とそれを用いて計算した厳密解 W を利用して作成する．構成 (2) は表 2 の数値とそれにより規定される GI/G/s システムのシミュレーションにより得られる評価値 W を利用して作成する（一つの W を得るためのシミュレーションは 100 万回実施する）．シミュレーションには離散シミュレーションパッケージ Csim20[13] を使用する．

構築 NN の学習は損失が十分小さくなるように 6000 回おこなう．

A Performance Evaluation of General Traffic Systems by Using Machine Learning

†Suguru NII, Takashi OKUDA

‡Yoichi UTSUNOMIYA

†Department of Information Science and Technology, Faculty of Information Science and Technology, Aichi Prefectural University

‡Graduate School of Information Science and Technology,Aichi Prefectural University

表 1　性能評価する GI/G/s システム

項目	記号	数値
平方変動係数	Ca^2, Cs^2	$0.1, 0.25, \cdots$ $1.0, 2.5, \cdots, 10.0$
到着率	λ	2.0, 8.0
処理率	μ	1.0
サーバー数	s	10
利用率	ρ	0.2, 0.8

表 2　教師データ作成時に使用する数値

教師データ	待ち行列システム	Ca^2	Cs^2	λ	μ	s	ρ
構成 (1)	M/G/1	1.0	0.1, 0.5	0.1, 0.3, 0.5	1.0	1	0.1, 0.3, 0.5
	GI/M/s	0.1, 0.5	1.0	0.1, 0.2, 0.3, 0.5		1, 2, 10	
構成 (2)	GI/G/s	0.1, 0.5, 1.0		0.6, 1.0, 3.0, 5.0			

利用率 $\rho = 0.2$ および $\rho = 0.8$ の場合の，GI/G/s システムの W を，図 2，3 にそれぞれ示す．図において，構成 (1)，(2) を教師データとして学習した NN により求めた W の推測値をそれぞれ W_{M_1}，W_{M_2} として表す．$W_{M_n}(n = 1, 2)$ との比較のため，すべてシミュレーションにより求めた値を真値 W(以後 W_S) として示す．x 軸は到着分布の平方変動係数 Ca^2，y 軸は処理分布の平方変動係数 Cs^2，z 軸は平均システム内時間 W である．

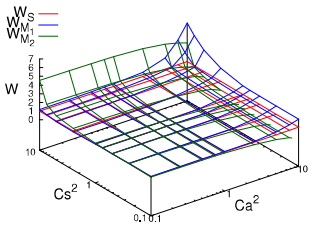

図 2　W ($\rho = 0.2$)

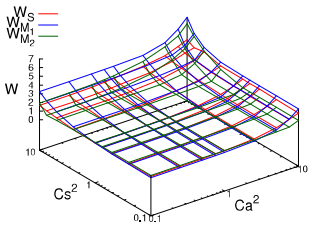

図 3　W ($\rho = 0.8$)

最初に W_{M_n} と W_S の関係を考察する．$\rho = 0.2$ の場合，W_{M_n} は，C^2 が 2.5 以下であれば W_S とおおむね一致した．しかし C^2 が 2.5 を超えると W_{M_1} は W_S より大きくなった．また Ca^2 が大きくなるほど W_{M_2} は W_S より小さくなり，Cs^2 が小さくなるほど W_{M_2} は W_S より大きくなった．一方 $\rho = 0.8$ の場合は，W_{M_n} は全ての領域において W_S に近いものとなった．

次に W_{M_n} と W_S の差分を図 4，5 に示す．x 軸，y 軸は図 2，3 と同様である．z 軸は W_{M_n} と W_S の差分である．W_{M_1} の差分と W_{M_2} の差分を比較すると，$\rho = 0.2$ のときは W_{M_1} の差分の方が全体的に小さいことがわかる．特に W_{M_2} の差分は，Cs^2 が 2.5 を超えると非常に大きくなった．一方 $\rho = 0.8$ のときは W_{M_2} の差分の方が全体的に小さいことがわかる．

最後に W_{M_n} と W_S を求めたときの計算時間を表 3 にまとめる．なお使用した PC のスペックは OS:Linux, CPU:Intel Xeon 2.7GHz, Memory:16GB である．表 3

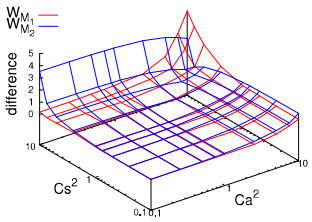

図 4　W_{M_n} の差分 ($\rho = 0.2$)

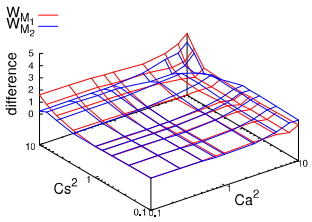

図 5　W_{M_n} の差分 ($\rho = 0.8$)

表 3　W を求めるのに要した計算時間

W の種類	項目	計算時間 ($\rho = 0.2$)	計算時間 ($\rho = 0.8$)
W_S	-	122 秒 27	153 秒 75
W_{M_1}	教師データ作成	0 秒 31	0 秒 31
	学習	17 秒 18	16 秒 96
	合計	17 秒 49	17 秒 27
W_{M_2}	教師データ作成	36 秒 71	36 秒 71
	学習	24 秒 62	24 秒 20
	合計	61 秒 33	60 秒 91

より W_{M_1} を求めたとき，計算時間は 60 秒以内に収まった．また W_{M_2} を求めたとき，計算時間は W_S を求めたときの約半分となった．このため，$\rho = 0.2$ の場合は構成 (1) を，$\rho = 0.8$ の場合は構成 (2) を用いれば短い時間に小さな誤差で W を求めることができるといえる．

5　おわりに

本研究では，機械学習による GI/G/s システムの性能評価をおこなった．その結果，$\rho = 0.2$ のときは構成 (1) を，$\rho = 0.8$ のときは構成 (2) を使用した機械学習による性能評価が有効であることがわかった．

現在 IoT データ処理システムとして有望視されているのは，フォグシステムで前処理をした後，クラウドシステムで処理するというタンデム型のシステムである．このシステムでは到着データの到着間隔やデータサイズに応じて動的にサーバーの数や性能を更新していくような仕組みが必要である．今後の課題は，このようなシステムの性能評価を機械学習によりおこなうことである．

参考文献

[1] 坂村，『コンピュータがネットと出会ったら モノとモノがつながりあう世界へ』，KADOKAWA，2015．[2]CiscoSystems，“Fog Computing and the Internet of Things:Extend the Cloud to Where the Things Are”，https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf，参照，Dec，2017．[3] 水越他，“通信トラフィック監視システムの試作とバーストトラフィックの検出 ”，情報処理学会研究報告インターネットと運用技術（IOT） ，vol.2004，no.77，pp.31-36，2004．[4] 米倉他，“バーストトラヒックを入力とするネットワークにおける平均系内時間の近似解析”，電子情報通信学会論文誌 B，vol.J85-B，no.7，pp.1021-1030，2002．[5] 村上，『わかりやすい情報交換工学』，森北出版，2009．[6] 大林他，“待ち行列理論による抽象化を用いたモデル検査手法の検討”，FIT2012（第 11 回情報科学技術フォーラム），B-030，pp.239-240．[7] 宮沢，『待ち行列の数理とその応用』，牧野書店，2013．[8]Seelen et al.，*Tables for Multi-Server Queues*，North-Holland，1985．[9] 清水，『はじめての深層学習プログラミング』，技術評論社，2017．[10] 秋丸他，『情報通信トラヒック–基礎と応用–』，オーム社，1990．[11] 市川，『階層型ニューラルネットワーク 非線形問題への応用』，共立出版，1993．[12] 奥田他，“ニューラルネットワークによる一般トラヒックモデル GI/G/s の性能評価”，電子情報通信学会論文誌 B，vol.J76-B1，No.10，pp.730-733，1993．[13]Mesquite Software，http://www.mesquite.com/，参照 Nov.2017．[14]Preferred Networks，https://chainer.org/，参照 Dec.2017．