

共有ストアをもつ 待ち行列ネットワーク型性能仕様モデルの定性的改善

澤 高根 (上智大学 大学院 理工学研究科 機械工学専攻 博士前期課程)
伊藤 潔 (上智大学 理工学部 一般科学研究室 情報科学部門)

共有ストアをもつ待ち行列ネットワーク型性能仕様モデルを対象とし、モデルの性能パラメータに対して定性推論を用いて、対象モデルの性能に悪影響を及ぼすものを検出し、改善プランを提示する手法について検討する。対象のシステムには、過大な負荷がかかっているつまり非定常（過負荷）状態にある、あるいはその可能性をもつ一状況を想定する。稼働率あるいは待ち行列長が過大なサーバ（ボトルネックサーバ）が存在したり、共有ストアが空あるいは一杯の時に入出力ポートで詰まってしまう（共有ストアのアンダフローボトルネックあるいはオーバフローボトルネック）と対象システムに悪影響を及ぼす。このような状態を解消するために、性能パラメータの測定平均値に対して定性推論を用いてチューニングする手法を述べる。

Qualitative Reasoning-Based Parameter Tuning for Performance Specification Model in the form of Queueing Network with Shared Store

Takane Sawa, Kiyoshi Itoh

Faculty of Science and Technology, Sophia University
Kioi-cho 7-1, Chiyoda-ku, Tokyo 102, Japan
itohkiyo@hoffman.cc.sophia.ac.jp

On the basis of heuristics and knowledges obtained from evaluation experts, the authors have developed a new "qualitative reasoning"-based expert system which can identify the bottlenecks, analyze the sources, and provide qualitative improvement plan option for a new type of Queueing Network (QN). The objective QN is the Queueing Network with Shared Store (QNSS). QNSS has the different types of bottlenecks in comparison with the ordinary QN. Primary types of bottlenecks in QNSS are "bottleneck servers" which have high utilization rates and / or high queue lengths, "shared stores in overflow bottlenecks" which have a large number of messages, and "shared stores in underflow bottlenecks" which have a small number of messages.

1 はじめに

共有ストアをもつ待ち行列ネットワーク型性能仕様モデルとその性能パラメータに対して定性推論([APT86],[BOB85],[DEX85],[MIZ89],[NIS88],[NIS89],[RAJ84])を用いて、対象システムの性能に悪影響を及ぼすものを検出し、改善プランを提示する手法を述べる。

共有ストアには一時的にメッセージを保存できる。また、この共有ストアにメッセージを搬入／搬出するトランザクションは、各々別々の経路を通る。このようなメッセージの一時保存ができるシステムの性能を分析するシステムとして、通常の待ち行列ネットワーク([KLE75],[GEL85])に共有ストアを導入したネットワーク(Queueing Network with Shared Store: QNSS)を用いる。図1に対象にしたQNSSの一例を示す。

対象のシステムには過大な負荷がかかっている — 即ち非定常(過負荷)状態にある、又はその可能性をもつ — 状況を想定する。稼働率又は待ち行列長が過大なサーバ(サーバのボトルネック)が存在したり、共有ストアが空の時に出力ポートで搬出待ちのためトランザクションが詰まってしまったり(共有ストアのアンダフローボトルネック)、逆に共有ストアがいっぱいの時に入力ポートで搬入待ちのため詰まってしまったり(共有ストアのオーバフローボトルネック)すると、対象システムに悪影響を及ぼす。システムの設計が誤っている場合、設計した通りにシステムが稼働しない場合、又は構成要素の能力の低下が起こった場合等である。

システムが定常状態の時は、状態方程式を立式可能である。非定常状態、即ち過負荷状態のシステムでは、

構成要素の各性能パラメータの関係を表す非線形の連立不等式を立式可能である。しかし、システム内の構成要素の数が増える事で方程式の数は膨大になり、これを解く事は非常に手間と時間がかかり効率的でない。そこで、ボトルネックを効率的に解消するためにQNSSの性能パラメータの測定平均値に対して定性推論を用いてチューニングする方法を考察する。

通常の待ち行列ネットワーク(Queueing Network:QN)のボトルネック改善を行うものとしてBDESとBIES([SAW89],[ITO89],[ITO90],[ITO91])がある。BDESは対象システム内に存在する1つ以上のボトルネックを検出し、評価者がその内の1つのボトルネックを選択すると、BDESは定性推論を用いてそのボトルネックを改善する択一的に選択可能な1つ以上の定性改善プランを提示する。BIESはBDESを受けて定量推論を用い定量改善プランを提示する。本稿で用いるQNSSはBDESで対象にしたQNに、共有ストアを導入したものである。BDESで用いた知識と、新たに導入した共有ストアに関する知識を合わせてQNSSのボトルネック改善を行う。本稿ではBDESの知識は省略する。

実際のシステムでボトルネックが起きた時、原因を解明し改善策を出し、その改善パラメータを用いて初期状態から動かし直す事は、ほとんどない。即ち、ボトルネックの状態でパラメータを改善パラメータに切り替えてボトルネックが解消する事が多い。そこで、QNSSのボトルネック改善はReal-time-tuningで行う。

QNSSのモデル化にあたり、様々な実現方法があると思われる。例えば、共有ストアをサーバの一種として

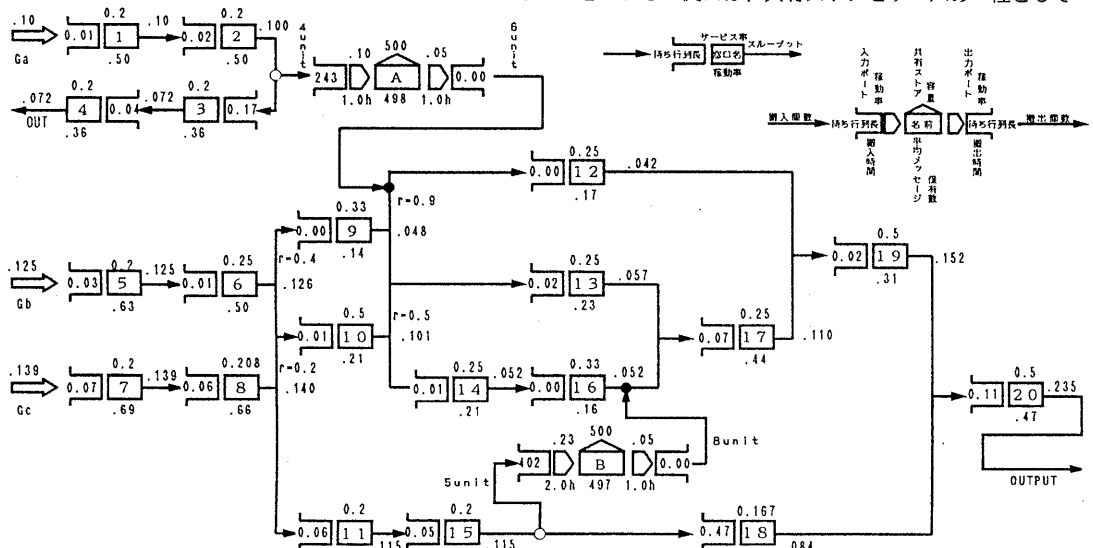


図1 共有ストアをもつ待ち行列ネットワーク QNSS10

考えてモデル化する事も可能であろうが、本稿ではサーバと共有ストアを別のものとして考える。

2節では様々な定義を、3節では個々のボトルネック改善策を、4節では共有ストア全体のボトルネック改善策を、5節では共有ストア及びサーバを含む部分形状のボトルネック改善策を述べる。

2 定義

2.1 共有ストアをもつ待ち行列ネットワークの定義

通常のQNは、サーバのみの組み合わせからなり、各サーバの状態は、主に上流のサーバの状態に影響された。本稿ではQNに共有ストアを加えたQNSSを用いる。

QNSS中を流れるトランザクションが共有ストアの入力ポートにきて、メッセージを共有ストアに搬入し、トランザクションは次のサーバに流れる。このメッセージは共有ストアに保管される。また、別の経路からのトランザクションが共有ストア出力ポートにきてメッセージを共有ストアから搬出し次のサーバへ流れる。

即ち、これまでのQNと違う点は、トランザクションとメッセージとは異なるものであり、その上、搬入と搬出のトランザクションが異なってもよい点である。

また、共有ストアには容量がある。容量を越える時には、共有ストア入力ポートにきたトランザクションはその待ち行列に並んで、共有ストアに空きができるまで待ち、空きができたならメッセージを搬入する。

逆に、共有ストアが空の場合、共有ストア出力ポートにきたトランザクションはその待ち行列に並び、共有ストアにメッセージがたまるのを待つ。

ここではトランザクションが持つメッセージの量を考えない。また、入力ポート及び出力ポートを通る経路は1つずつとする。

2.2 性能パラメータの定義

2.2.1 サーバの性能パラメータの定義

以下に単一のサーバの5つのパラメータを定義する。

λ : 単位時間当たりにサーバに到着するトランザクションの平均個数(到着率: arrival rate)

μ : 単位時間当たりにサーバで処理できるトランザクションの平均個数(サービス率: servicing rate)

t : 単位時間当たりにサーバから出てくるトランザクションの平均個数(スループット: throughput)

ρ : サーバが実際にトランザクションを処理する時間の割合(稼働率: utilization rate)

q : サーバの前で処理を待つトランザクションの平均個数(待ち行列長: queue length)

図2と図3に示す通り、 λ と μ が与えられるとその大関係によりそのサーバの性能を表す ρ 、 t が決まる。

図2は、到着率がサービス率より小さい場合、即ち、平均到着時間間隔(到着率の逆数)が平均サービス時間

(サービス率の逆数)より大きい場合、サーバはよどみなくトランザクションを処理できる。サーバから出るトランザクションの量(スループット)は、サーバへの到着率に等しくなる。稼働率は、

$$\rho = \text{平均サービス時間} / \text{平均到着時間間隔} \\ = (1/\mu) / (1/\lambda) = \lambda / \mu \quad \text{となる。}$$

図3は、到着率がサービス率より大きい場合、即ち、平均到着時間間隔が平均サービス時間より小さい場合、サーバはよどみなくトランザクションを処理できない。即ち、あるトランザクションに対するサービスを行う間に次のトランザクションがそのサーバに到着する。このサーバからのスループットは、サーバのサービス率に等しくなる。稼働率は、サーバが常に働いている事になるので、 $\rho \equiv 1$ となる。

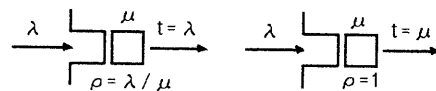


図2. $\lambda < \mu$ の場合

図3. $\lambda \geq \mu$ の場合

2.2.2 共有ストアの性能パラメータの定義

以下に共有ストアの13のパラメータを定義する。図4に共有ストアの表記方法を示す。

cap : 共有ストアに保管できるメッセージの最大数 (共有ストア容量: capacity of shared store)

ns : 共有ストアにあるメッセージの平均数(平均メッセージ保有数: average number of messages in shared store)

ρ_s : メッセージが共有ストア容量に占める割合 (共有ストア占有率: occupation rate of messages)

λ_{in} : 入力ポートに単位時間当たりに到着するトランザクション数(入力ポート到着率: arrival rate at entrance)

λ_{out} : 出力ポートに単位時間当たりに到着するトランザクション数(出力ポート到着率: arrival rate at exit)

$1/\mu_{in}$: 1つのトランザクション当たりの共有ストアへのメッセージの搬入時間(搬入時間: times for a transaction to take in messages)

$1/\mu_{out}$: 1つのトランザクション当たりの共有ストアへのメッセージの搬出時間(搬出時間: times for a transaction to take out messages)

nb_{in} : 1つのトランザクション当たりのメッセージの搬入個数(搬入個数: number of messages for a transaction to take in)

nb_{out} : 1つのトランザクション当たりのメッセージの搬出個数(搬出個数: number of messages for a transaction to take out)

q_{in} : 入力ポートに並ぶ単位時間当たりのトランザクション数(入力ポート平均待ち行列長 : queue length at entrance)

q_{out} : 出力ポートに並ぶ単位時間当たりのトランザクション数(出力ポート平均待ち行列長 : queue length at exit)

ρ_{in} : 入力ポートが実際にメッセージを処理する時間の割合(入力ポート稼働率 : utilization rate at entrance)

ρ_{out} : 出力ポートが実際にメッセージを処理する時間の割合(出力ポート稼働率 : utilization rate at exit)

共有ストア占有率 ρ_s は、以下のように定義される。

$$\rho_s = ns / cap$$

また、共有ストアにメッセージの搬入、搬出をしたトランザクション数をそれぞれ c_{in} 、 c_{out} とすると、

$$\rho_{in} = (1/\mu_{in}) * c_{in} / \text{総シミュレーション時間}$$

$$\rho_{out} = (1/\mu_{out}) * c_{out} / \text{総シミュレーション時間}$$

と表せる。

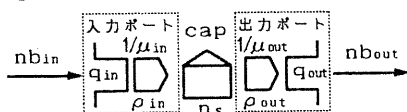


図4 共有ストアの表記

2.3 ボトルネックの定義

2.3.1 サーバのボトルネック

QNSS内で稼働率が1に極めて近いサーバは、ボトルネック(bottleneck)である。このボトルネックサーバの待ち行列(queue)は無限大に成長する恐れがある。ボトルネックサーバが存在する場合、QNSSは非定常(過負荷)状態(unstable or overloaded state)である。

稼働率が1に近くないが過大なサーバはボトルネックとなる可能性がある。サーバへの到着時間間隔やサーバのサービス時間が大きくなればつきを持つ場合、稼働率が1に近くなくても待ち行列長が時には急激に増加する危険性がある。フェールセーフ(failure to safety)の考え方を導入して、稼働率が0.7以上のサーバにボトルネックの可能性があるとして経験的に診断した。

また、稼働率が過大ではないが待ち行列長が過大であるサーバには——測定がそれほど長時間行われていないかもしれない——測定終了以降にトランザクションが過大に到着して稼働率が大きくなる危険性がある。この過大な待ち行列長は後に徐々に解消されるかもしれないが、ボトルネックの発生の事前防止というフェールセーフの考え方で、測定時に平均待ち行列長が1以上のサーバにボトルネックの可能性があるとして診断した。以下では、サーバのボトルネックになる可能性のある

状態を単にサーバのボトルネックと呼ぶ。

2.3.2 共有ストアボトルネック

2.3.2.1 オーバフローボトルネック

従来のサーバのみで構成されるQNでは、サーバの稼働率が0.7以上、または、平均待ち行列長が1以上であるものをボトルネックと定義した。QNSSでは、通常のサーバは従来と同様の基準でボトルネックと診断し、共有ストアボトルネックに新しい判断基準を設けた。

共有ストア内ではメッセージはどんな処理も受けず、ただ保管されるだけなので、サービス率 μ も存在せず、共有ストア稼働率という概念も存在しない。そこで、共有ストアの状況は、共有ストア占有率で判断する。

共有ストアの過負荷(OVERFLOW)とは、共有ストアがメッセージで一杯になり、入力ポートにきたトランザクションがメッセージを搬入する事ができず、搬入ポートで待ち行列に並ぶ状態である。このボトルネックを、オーバフローボトルネックと呼ぶ。

共有ストア占有率が1に近くないが過大な共有ストアは、一時的に共有ストア占有率が1に近くなる可能性があり、そのために入力ポートの平均待ち行列長が異常に増える可能性がある。共有ストアの過負荷状態を避けるために、フェールセーフの考え方を導入して、共有ストアの占有率が0.7以上の場合に、オーバフローボトルネックの可能性があると診断する。以下では、オーバフローボトルネックになる可能性のある状態を単にオーバフローボトルネックと呼ぶ。

2.3.2.2 アンダフローボトルネック

共有ストアボトルネックは、サーバのボトルネックのように過負荷のみを扱えばいいというわけではなく、共有ストアが空(UNDERFLOW)の場合もある。これは、共有ストアに十分なストックがないために、搬出のために共有ストアに訪れたトランザクションが、共有ストアに入力ポート側からメッセージが搬入されるまで、出力ポートで待ち行列に並ぶ状態である。このボトルネックをアンダフローボトルネックと呼ぶ。

共有ストア占有率が0に近くないが過小な共有ストアは、一時的に共有ストア占有率が0に近くなる可能性があり、そのために出力ポートの平均待ち行列長が異常に増える可能性がある。更に、共有ストア占有率が過小であるという事は、共有ストア容量を無駄にする事でもあり、非合理的である。また、占有率が過小でなくても、搬出時間が大きい事等により、出力ポート待ち行列長が過大である共有ストアは、長い時間が経過すると平均待ち行列長が無限大に成長する可能性がある。よって、共有ストアのメッセージ不足状態を避けるために、フェールセーフの考え方を導入して、共有ストア占有率が0.3以下の場合に、アンダフローボトル

ネックの可能性があると診断する。以下では、アンダフローボトルネックになる可能性のある状態を単にアンダフローボトルネックと呼ぶ。

2.3.2.3 入力・出力ポートのボトルネック

共有ストアに関する2つのボトルネックは、共有ストア容量に対してのメッセージの出し入れが過大、過小であるために起こった。入力・出力ポートボトルネックは、現在の共有ストア容量に関係なく、入力・出力ポートそのものがボトルネックになるという点で前出の2つのボトルネックと異なる。また、入力ポートボトルネックが原因となってアンダフローボトルネックが起きたり、出力ポートボトルネックが原因となってオーバフローボトルネックが起きたりする事もある。

入力ポートでは、搬入経路を流れてきたトランザクションが、運んできたメッセージを共有ストアに運び込む作業をする。当然この作業には一定の時間がかかる。その作業時間は搬入時間と呼ばれ、搬入個数と同様に搬入経路で値が決まる。搬入時間、またはトランザクションの到着率が過大であると、到着率が搬入時間の逆数より大きくなる(搬入時間が到着時間間隔より大きくなってしまふ)。すると、前のトランザクションが運んできたメッセージを降ろしている間に次のトランザクションが到着する事になり、そのトランザクションは入力ポート待ち行列に並ぶ事になる。即ち、常に待ち行列長が1以上となり、ボトルネックの可能性が生まれる。同様の事が出力ポートでも考えられる。

入力・出力ポート稼働率が1に近くないが過大な共有ストアは、一時的に稼働率が1に近くなる可能性があり、そのために入力・出力ポート待ち行列長が異常に増える可能性がある。フェールセーフの考え方を導入して、入力・出力ポートの稼働率が0.7以上の場合に、入力・出力ポートにボトルネックの可能性があると診断する。また、入力・出力ポートの待ち行列長が過大である共有ストアは、長い時間が経過すると平均待ち行列長が無限大に成長する可能性がある。そこで、入力・出力ポート待ち行列長が1以上の場合もボトルネックの可能性があると診断する。以下では、入力・出力ポートボトルネックになる可能性のある状態を単に入力・出力ポートボトルネックと呼ぶ。

3 ボトルネック改善

3.1 サーバのボトルネック改善

サーバのボトルネックの改善策は次の二通りがある。

- 1) サーバの到着率を減少。2) サービス率を増加。

3.2 共有ストアボトルネック改善

3.2.1 オーバフローボトルネックの改善

共有ストア占有率が0.7以上の場合のボトルネック改善策は、以下のように5つに分類される。

- 1) メッセージの搬入個数を減少。
- 2) 入力ポートのトランザクションの到着率を減少。
- 3) 共有ストア容量を増加。
- 4) 出力ポートのトランザクションの到着率を増加。
- 5) メッセージの搬出個数を増加。

3.2.2 アンダフローボトルネックの改善

共有ストア占有率が0.3以下の場合のボトルネック改善策は、以下のように5つに分類される。

- 1) メッセージの搬入個数を増加。
- 2) 入力ポートのトランザクションの到着率を増加。
- 3) 共有ストア容量を減少。
- 4) 出力ポートのトランザクションの到着率を減少。
- 5) メッセージの搬出個数を減少。

3.2.3 入力ポートのボトルネック改善

- 1) メッセージの搬入時間を減少。
- 2) 入力ポートのトランザクションの到着率を減少。

3.2.4 出力ポートのボトルネック改善

- 1) メッセージの搬出時間を減少。
- 2) 出力ポートのトランザクションの到着率を減少。

4 単一共有ストアボトルネック改善

QNSS内にはボトルネックを引き起こすための様々なパラメータが存在する。複数のボトルネックからシステムの改善策を模索しようとする、式の数が多くなり解くのに時間がかかってしまう。そこで、局所的な改善策を求め、それがシステム全体で矛盾がないようにしていく必要がある。この節では、単一共有ストアについて複数のパラメータからボトルネックが検出された時の、単一共有ストア全体での改善策を求める。

入力ポートボトルネックと出力ポートボトルネックとは直接的関係はないと思われる。そこで、入力ポート側(共有ストアも含む)と出力ポート側(共有ストアも含む)に分け、各々の改善策(4.1, 4.2節)を求める。そして、共有ストア全体での改善策として、入力ポート側と出力ポート側の改善策を組み合わせる。例えば、出力ポート側改善策が3通り、出力ポート側改善策が4通りある場合、共有ストア全体の改善策は各々の組み合わせとして12通り存在する。

但し、入力ポート側と出力ポート側の改善策で矛盾を起こす場合がある。これについては4.3節で述べる。

4.1 入力ポート側ボトルネック改善策

表1に入力ポート平均待ち行列長(q_{in})と入力ポート稼働率(ρ_{in})と共有ストア占有率(ρ_s)のとり得る状態を列挙する。

ここで、 q_{in} の '+' は平均待ち行列長が1以上、 '-' を1未満、 ρ_{in} の '+' を稼働率0.7以上、 '-' を0.7未満、 ρ_s の 'O' を共有ストア占有率0.7以上、 'U' を0.3以下、 '-' を0.3より大きく、0.7未満と定義する。即

ち、'+'の時入力ポートボトルネックとなり、'O'の時共有ストアのオーバーフローボトルネック、'U'の時共有ストアのアンダーフローボトルネックとなる。

表1. $q_{in}, \rho_{in}, \rho_s$ の関係

	q_{in}	ρ_{in}	ρ_s
①	+	+	O
②	+	+	-
③	+	+	U
④	+	-	O
⑤	+	-	-
⑥	+	-	U
⑦	-	+	O
⑧	-	+	-
⑨	-	+	U
⑩	-	-	O
⑪	-	-	- , U

複数のパラメータからボトルネックが検出された場合、各々の改善策のうち全てに共通する改善策を選ぶ。以下に、表の各場合の改善策を列挙する。

① $q_{in} \geq 1, \rho_{in} \geq 0.7, \rho_s \geq 0.7$ の場合

1) 入力ポートのトランザクションの到着率を減少。

② $q_{in} \geq 1, \rho_{in} \geq 0.7, 0.3 < \rho_s < 0.7$ の場合

1) メッセージの搬入時間を減少。

2) 入力ポートのトランザクションの到着率を減少。

③ $q_{in} \geq 1, \rho_{in} \geq 0.7, \rho_s \leq 0.3$ の場合

1) メッセージの搬入時間を減少。

④ $q_{in} \geq 1, \rho_{in} < 0.7, \rho_s \geq 0.7$ の場合

1) 入力ポートのトランザクションの到着率を減少。

2) 共有ストア容量を増加。

3) 出力ポートのトランザクションの到着率を増加。

4) メッセージの搬出個数を増加。

⑤ $q_{in} \geq 1, \rho_{in} < 0.7, 0.3 < \rho_s < 0.7$ の場合

q_{in} だけのボトルネックなので、3.2.3節の改善策を用いる。

⑥ $q_{in} \geq 1, \rho_{in} < 0.7, \rho_s \leq 0.3$ の場合

1) メッセージの搬入時間を減少。

⑦ $q_{in} < 1, \rho_{in} \geq 0.7, \rho_s \geq 0.7$ の場合

1) 入力ポートのトランザクションの到着率を減少。

⑧ $q_{in} < 1, \rho_{in} \geq 0.7, 0.3 < \rho_s < 0.7$ の場合

ρ_{in} だけのボトルネックなので、3.2.3節の改善策を用いる。

⑨ $q_{in} < 1, \rho_{in} \geq 0.7, \rho_s \leq 0.3$ の場合

1) メッセージの搬入時間を減少。

⑩ $q_{in} < 1, \rho_{in} < 0.7, \rho_s \geq 0.7$ の場合

ρ_s だけのボトルネックなので、3.2.1節の改善策を用いる。

⑪ $q_{in} < 1, \rho_{in} < 0.7, \rho_s < 0.7$ の場合

$\rho_s \leq 0.3$ の場合は出力ポート側の改善策を用いる。 $0.3 < \rho_s < 0.7$ の場合はボトルネックでないので改善しない。即ち、入力ポート側の改善策は提示しない。

4.2 出力ポート側ボトルネック改善策

表2に出力ポート平均待ち行列長(q_{out})と出力ポート稼働率(ρ_{out})と共有ストア占有率(ρ_s)のとり得る状態を列挙する。ここで、'+','-','O','U'の意味は、4節で説明したものと同じである。

表2. $q_{out}, \rho_{out}, \rho_s$ の関係

	q_{out}	ρ_{out}	ρ_s
①	+	+	O
②	+	+	-
③	+	+	U
④	+	-	O
⑤	+	-	-
⑥	+	-	U
⑦	-	+	O
⑧	-	+	-
⑨	-	+	U
⑩	-	-	O, -
⑪	-	-	U

複数のパラメータからボトルネックが検出された場合、各々の改善策のうち全てに共通する改善策を選ぶ。以下に、表の各場合の改善策を列挙する。

① $q_{out} \geq 1, \rho_{out} \geq 0.7, \rho_s \geq 0.7$ の場合

1) メッセージの搬出時間を減少。

② $q_{out} \geq 1, \rho_{out} \geq 0.7, 0.3 < \rho_s < 0.7$ の場合

1) メッセージの搬出時間を減少。

2) 出力ポートのトランザクションの到着率を減少。

③ $q_{out} \geq 1, \rho_{out} \geq 0.7, \rho_s \leq 0.3$ の場合

1) 出力ポートのトランザクションの到着率を減少。

④ $q_{out} \geq 1, \rho_{out} < 0.7, \rho_s \geq 0.7$ の場合

1) メッセージの搬出時間を減少。

⑤ $q_{out} \geq 1, \rho_{out} < 0.7, 0.3 < \rho_s < 0.7$ の場合

q_{out} だけのボトルネックなので、3.2.4節の改善策を用いる。

⑥ $q_{out} \geq 1, \rho_{out} < 0.7, \rho_s \leq 0.3$ の場合

1) 出力ポートのトランザクションの到着率を減少。

2) 共有ストア容量を減少。

3) 入力ポートのトランザクションの到着率を増加。

4) メッセージの搬出個数を増加。

⑦ $q_{out} < 1, \rho_{out} \geq 0.7, \rho_s \geq 0.7$ の場合

1) メッセージの搬出時間を減少。

⑧ $q_{out} < 1, \rho_{out} \geq 0.7, 0.3 < \rho_s < 0.7$ の場合

ρ_{out} だけのボトルネックなので、3.2.4節の改善策を用いる。

⑨ $q_{out} < 1, \rho_{out} \geq 0.7, \rho_s \leq 0.3$ の場合

- 1) 出力ポートのトランザクションの到着率を減少。
 ⑩ $q_{out} < 1, \rho_{out} < 0.7, \rho_s > 0.3$ の場合
 $\rho_s \geq 0.7$ の場合は入力ポート側の改善策を用いる。
 $0.3 < \rho_s < 0.7$ の場合は、ボトルネックでないので改善しない。即ち、出力ポートの改善策は提示しない。
 ⑪ $q_{out} < 1, \rho_{out} < 0.7, \rho_s \leq 0.3$ の場合
 ρ_s だけのボトルネックなので、3.2.2節の改善策を用いる。

4.3 例外事項

表3に入力ポート側ボトルネック改善策と、出力ポート側ボトルネック改善策の組合わせて表すと矛盾が起きてしまう場合を示す。

表3. $q_{in}, \rho_{in}, \rho_s, \rho_{out}, q_{out}$ の関係

	q_{in}	ρ_{in}	ρ_s	ρ_{out}	q_{out}
①	+	+	U	-	+
②	+	+	U	-	-
③	+	-	O	+	+
④	-	-	O	+	+

表3の①で矛盾が起きる時の例を挙げる。これは、入力ポート側改善策(4.1節)の③と出力ポート側改善策(4.2節)の⑥の組み合わせである。このうち出力ポート側改善策の3)入力ポートのトランザクションの到着率を増加、を用いると出力ポート側からみるとアンダフローが解消されて q_{out} も減るが、 q_{in} が過大であるので更に到着率を増やしても改善効果が現れなく、 q_{in} の待ち行列長が更に増大する可能性がある。このため3)の改善策は用いない。

以下に、表の各場合の改善策を示す。

- ①入力ポート側改善策の③と、出力ポート側改善策の⑥の3)を除いたものを組合わせる。
 ②入力ポート側改善策の③と、出力ポート側改善策の⑥の2)を除いたものを組合わせる。
 ③入力ポート側改善策の④の3)を除いたものと、出力ポート側改善策の⑥を組合わせる。
 ④入力ポート側改善策の④の4)を除いたものと、出力ポート側改善策の⑥を組合わせる。

5 共有ストアを含む部分形状の改善

4節で単一共有ストアの改善策と、その中で様々な改善パラメータを挙げた。5節で各々の改善パラメータと単一共有ストアを含む部分形状での改善策を考察する。

5.1 改善パラメータ

共有ストア改善策を以下に示す。

- i) 入力ポートのトランザクションの到着率を減少 ($d\lambda_{in}=-$)
 ii) 入力ポートの到着率を増加 ($d\lambda_{in}=+$)
 iii) 出力ポートの到着率を減少 ($d\lambda_{out}=-$)
 iv) 出力ポートの到着率を増加 ($d\lambda_{out}=+$)

- v) メッセージの搬入時間を減少 ($d1/\mu_{in}=-$)
 vi) メッセージの搬出時間を減少 ($d1/\mu_{out}=-$)
 vii) 倉庫容量を増加 ($dcap=+$)
 viii) 倉庫容量を減少 ($dcap=-$)
 ix) メッセージの搬入個数を減少 ($dnb_{in}=-$)
 x) メッセージの搬入個数を増加 ($dnb_{in}=+$)
 xi) メッセージの搬出個数を減少 ($dnb_{out}=-$)
 xii) メッセージの搬出個数を増加 ($dnb_{out}=+$)

5.2 改善パラメータと単一共有ストアを含む部分形状

各改善パラメータと単一共有ストアを含む部分形状での改善策を考察する。単一共有ストアを含む部分形状について、搬入ポート、搬出ポートを通る経路でタイプ分けしたものを図5の④~⑥に示す。例えば、④のaとcは上流で合流しないという点で⑤と異なる。12種類の改善パラメータのうち3種類を以下に挙げる。

- i) $d\lambda_{in}=-$: このパラメータは入力ポート側改善策にしか現れなく、その時にとり得る出力ポート側改善策は、1) $d\lambda_{out}=-$, 2) $d(1/\mu_{out})=-$, 3) 改善しないのどれかである。出力ポート側改善策とその形状としての改善策の関係を表4に示す。

表4 $d\lambda_{in}=-$ の時の出力ポート側改善策と形状改善策

④	1) $d\lambda_{out}=-$ 2) $d(1/\mu_{out})=-, d\lambda_{as}=-, (d\lambda_{bs}=-; d\lambda_{ds}=+)$ 3) $d\lambda_{as}=-, (d\lambda_{bs}=-; d\lambda_{ds}=+)$
⑤	1) $d\lambda_{as}=-$ 2) $d(1/\mu_{out})=-, (d\lambda_{bs}=-; d\lambda_{ds}=+)$ 3) $d\lambda_{bs}=-; d\lambda_{ds}=+$
⑥	1) $d\lambda_{as}=-; d\lambda_{ds}=-; d\lambda_{bs}=+; d\lambda_{cs}=+$ 2) $d(1/\mu_{out})=-, (d\lambda_{as}=-; d\lambda_{cs}=+), (d\lambda_{bs}=-; d\lambda_{ds}=+)$ 3) $(d\lambda_{cs}=+; d\lambda_{as}=-), (d\lambda_{bs}=-; d\lambda_{ds}=+)$
⑦	1) $d\lambda_{as}=-; d\lambda_{cs}=-$ 2) $d(1/\mu_{out})=-, d\lambda_{as}=-$ 3) $d\lambda_{as}=-$
⑧	1) $d\lambda_{as}=-; (d\lambda_{bs}=-; d\lambda_{cs}=+), (d\lambda_{es}=+; d\lambda_{fs}=-)$ 2) $d(1/\mu_{out})=-, (d\lambda_{bs}=-; d\lambda_{cs}=+), (d\lambda_{as}=-, (d\lambda_{es}=-; d\lambda_{fs}=+))$ 3) $(d\lambda_{bs}=-; d\lambda_{cs}=+), (d\lambda_{as}=-, (d\lambda_{es}=-; d\lambda_{fs}=+))$
⑨	1) $d\lambda_{as}=-; d\lambda_{es}=-$ 2) $d(1/\mu_{out})=-, d\lambda_{as}=-$ 3) $d\lambda_{as}=-$
⑩	1) $d\lambda_{as}=-; (d\lambda_{bs}=-; d\lambda_{cs}=+), (d\lambda_{gs}=+; d\lambda_{hs}=-)$ 2) $d(1/\mu_{out})=-, (d\lambda_{bs}=-; d\lambda_{cs}=+), (d\lambda_{as}=-, (d\lambda_{gs}=-; d\lambda_{hs}=+))$ 3) $(d\lambda_{bs}=-; d\lambda_{cs}=+), (d\lambda_{as}=-, (d\lambda_{gs}=-; d\lambda_{hs}=+))$

この表で、 $d\lambda_{as}=-$ は図のaからの入力量を減少する事を、 $d\lambda_{bs}=-$ はbへの分岐確率を減少する事を表す。また、', 'は'かつ'を、'; 'は'または'を意味する。

- ii) $d\lambda_{in}=+$: このパラメータは出力ポート側改善策にしか現れなく、その時にとり得る入力ポート側改善策は、1) $d(1/\mu_{in})=-$, 2) 改善しないのどちらかである。入力ポート側改善策とその形状としての改善策の関係を表5に示す。

表5 $d\lambda_{in}=+$ の時の入力ポート側改善策と形状改善策

④	1) $d(1/\mu_{in})=-, d\lambda_{as}=-, (d\lambda_{bs}=-; d\lambda_{ds}=+)$ 2) $d\lambda_{as}=-, (d\lambda_{bs}=-; d\lambda_{ds}=+)$
⑤	1) $d(1/\mu_{in})=-, (d\lambda_{bs}=-; d\lambda_{ds}=+)$ 2) $d\lambda_{bs}=-; d\lambda_{ds}=+$
⑥	1) $d(1/\mu_{in})=-, (d\lambda_{as}=-; d\lambda_{cs}=-), (d\lambda_{bs}=-; d\lambda_{ds}=+)$ 2) $(d\lambda_{as}=-; d\lambda_{cs}=-), (d\lambda_{bs}=-; d\lambda_{ds}=+)$

①	1) $d(1/\mu \text{ in}) = -, d\lambda a = +$
2)	$d\lambda a = +$
②	1) $d(1/\mu \text{ in}) = -, (d\lambda a = +, (d r e = +; d\lambda f = -); d\lambda b = +; d r c = -)$
2)	$(d\lambda a = +, (d r e = +; d\lambda f = -); d\lambda b = +; d r c = -)$
③	1) $d(1/\mu \text{ in}) = -, d\lambda a = +$
2)	$d\lambda a = +$
④	1) $d(1/\mu \text{ in}) = -, (d\lambda a = +, (d r g = +; d\lambda h = -); d\lambda b = +; d r c = -)$
2)	$(d\lambda a = +, (d r g = +; d\lambda h = -); d\lambda b = +; d r c = -)$

iii) $d\lambda \text{ out} = -$: このパラメータは出力ポート側改善策にしか現れなく、その時にとり得る入力ポート側改善策は、1) $d\lambda \text{ in} = -$, 2) $d(1/\mu \text{ in}) = -$, 3) 改善しないのどれかである。入力ポート側改善策とその形状としての改善策の関係を表6に示す。

表6 $d\lambda \text{ out} = -$ の時の出力ポート側改善策と形状改善策

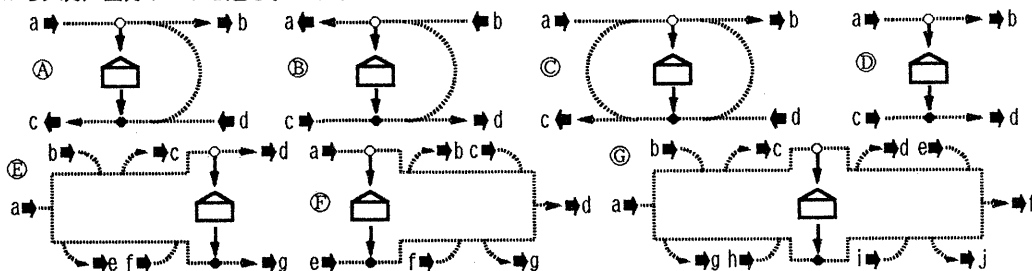
①	1) $d\lambda a = -$
2)	$d(1/\mu \text{ in}) = -, (d r b = +; d\lambda d = -)$
3)	$d r b = +; d\lambda d = -$
②	1) $d\lambda c = -$
2)	$d(1/\mu \text{ in}) = -, d\lambda c = -, (d\lambda b = -; d r d = +)$
3)	$d\lambda c = -, (d\lambda b = -; d r d = +)$
③	1) $d\lambda a = -; d\lambda d = -; d r b = +; d r c = +$
2)	$d(1/\mu \text{ in}) = -, (d r b = +; d\lambda d = -), (d r c = +; d\lambda a = +)$
3)	$(d r b = +; d\lambda d = -), (d r c = +; d\lambda a = +)$
④	1) $d\lambda a = -, d\lambda c = -$
2)	$d(1/\mu \text{ in}) = -, d\lambda c = -$
3)	$d\lambda c = -$
⑤	1) $d\lambda a = -; (d\lambda b = -; d r c = +), (d r e = +; d\lambda f = -)$
2)	$d(1/\mu \text{ in}) = -, (d r e = +; d\lambda f = -), (d\lambda a = -, (d\lambda b = +; d r c = -))$
3)	$d r e = +; d\lambda f = -; (d\lambda a = -, (d\lambda b = +; d r c = -))$
⑥	1) $d\lambda a = -, d\lambda e = -$
2)	$d(1/\mu \text{ in}) = -, d\lambda e = -$
3)	$d\lambda e = -$
⑦	1) $d\lambda a = -; (d\lambda b = -; d r c = +), (d r g = +; d\lambda h = -)$
2)	$d(1/\mu \text{ in}) = -, (d r g = +; d\lambda h = -), (d\lambda a = -, (d\lambda b = +; d r c = -))$
3)	$d r g = +; d\lambda h = -; (d\lambda a = -, (d\lambda b = +; d r c = -))$

6 おわりに

QNSSでの定性的な改善推論を用いてチューニングする手法を述べた。まだ、単一共有ストアを含めた部分形状での改善までしか行っていないが、複数の共有ストア間の改善を考える必要がある。サーバ間のボトルネック改善には、BDESで用いた定性推論を用いる。

今後はこの知識を用いて、QNSSのパラメータをチューニングするエキスパートシステムを開発していく。

更に、トランザクションが初期値としてもメッセージの量、メッセージの最大量(容量)や、複数の経路から入力/出力ポートを通るQNSSも考える。



注1 共有ストアの入力・出力ポートは省略

注2 ○, ●はそれぞれ共有ストアへの搬入、搬出箇所を表す

注3 点線上にサーバが存在してもなくてもよい

注4 分岐・合流は存在してもなくてもよい

図5 単一共有ストアを含む部分形状の分類

参考文献

- [APT86]Apte, C. et al.: Using qualitative reasoning to understand financial arithmetic, Proc. AAAI'86, (1986).
- [BOB85]Bobrow, D.G., et al. ed.: Qualitative reasoning about physical systems, MIT Press, (1985).
- [DEK85]De Kleer, J.: How circuits work, in [BOB85], (1985).
- [GEL85]Gelenbe, E. et al.: Analysis and Synthesis of Computer Systems, Academic Press, (1985).
- [ITO89]Itoh, K. et al.: Knowledge-Based Parameter Tuning for Queueing Network Type System - A New Application of Qualitative Reasoning, CAPT'89, (October, 1989).
- [ITO90]伊藤, 本位田, 沢村, 志田: 定性推論と定量推論を導入した待ち行列ネットワークのボトルネック診断と改善法, 人工知能学会誌, Vol.5, No.1, (1990).
- [ITO91]伊藤, 本位田: 定性推論のパラメータチューニングへの応用, 情報処理学会誌, 32.2, (Feb., 1991)
- [KLE75]Kleinrock, L.: Queueing Systems, John Wiley & Sons, Inc., (1975).
- [MI289]溝口, 古川, 安西(編): 定性推論, 共立出版(1989).
- [NIS88]西田: 定性推論に関する最近の研究動向, 情報処理, Vol.29, NO.9, No.11, (1988).
- [NIS89]西田: 定性推論の基礎, 人工知能学会誌, Vol.4, NO.5, 522-527 (1988).
- [RAJ84]Rajagopalan, R.: Qualitative modeling in the turbojet engine domain, Proc. AAAI'84, (1984).
- [SAW89]沢村, 本位田, 志田, 伊藤: 知識工学的手法を用いた待ち行列ネットワークのボトルネック診断, 情報処理学会論文誌, 30.8, 990-1002 (August, 1989).