入門講座

やさしい待ち行列 (4) ——ネットワークとコントロール

高橋　幸雄

最終回の今回は、待ち行列がネットワーク状につながった待ち行列ネットワークと、そのときに重要となるコントロールの問題について考えてみましょう。

前回お話ししたような待ち行列は、単独で存在するものばかりではなく、あちこちに互いに関連しながらできることがよくあります。たとえば組立工場では組み立てられていく半製品が工程ごとに待ち行列を作っていますし、コンピュータの中でもジョブがCPUの前やディスクの前などで行列を作って自分の順番を待っています。最近、急速に技術革新が行われている高速通信でも、ネットワークを通じて情報が送られるとき各ノードのバッファで待ちが入ります。このような複数の待ち行列が関連してシステムを構成しているものを、待ち行列ネットワークと呼んでいます。

このような待ち行列ネットワークでは、どの待ち行列が混んで、どれがすいているか、ボトルネックはどこか、ある待ち行列が混みだすと、それが他の待ち行列にどのように伝播するか、などいろいろな問題があり、さらにこれらの問題を解消するためにどのようなコントロールが可能なのか、といった様々な問題を考えていかなければなりません。

これらはモデル解析という点からもたいへん難しく、本講座のレベルをはるかに超えていますが、ここではその一部をちらっとご紹介しましょう。まずは、単独の待ち行列のコントロールの問題からです。

1. 最短順サービス

前回、待ち時間を小さくするには、i) 到着客の数を減らすかサービス時間を短くして利用率 ρ を小さくすること、ii) 到着間隔やサービス時間の分布のランダムネスを小さくすること、iii) そして可能ならば複数窓口にすること、が有効であることをお話ししました。もうひとつ、サービスの順番を変えることによって平均待ち時間を小さくすることもできます。

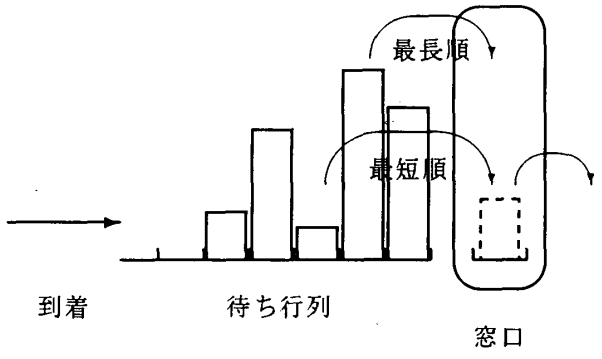

図 1: 最短順と最長順

ここでも簡単のためM/M/1モデル、つまり窓口は1つで、客はパラメータ λ のポアソン過程にしたがって到着し、平均 $1/\mu$ の指数分布にしたがう時間だけサービスをうける、というモデルで考えてみましょう。

いま、待ち行列で待っている客それぞれの要求サービス時間が、前もってわかっているものとします。ある客のサービスが終了したときに、つぎにサービスする客は、到着順に関わらず、待っている客の中でサービス時間が最も短い客を選ぶ、というサービス規律を“最短順”、最も長い客を選ぶサービス規律を“最長順”と呼びます (図 1)。ではクイズ、

> **クイズ 1**
> M/M/1 モデルにおいて、先着順、最短順、最長順の3つのサービス規律では、平均待ち時間はどれが一番短くなるでしょう。

何をどう考えたらよいのか、わかりにくかったかもしれませんが、つぎのように考えると理解しやすいと思います。

図 2 を見てください。いま、客 F のサービスが終了した時刻 t において 3 人の客 A, B, C が待ち行列で待っているものとしましょう。この 3 人が先着順でサービスされるなら、時刻 t 以降のこれら 3 人の待

たかはし ゆきお　東京工業大学 大学院 情報理工学研究科 数理・計算科学専攻
〒 152 東京都目黒区大岡山 2-12-1

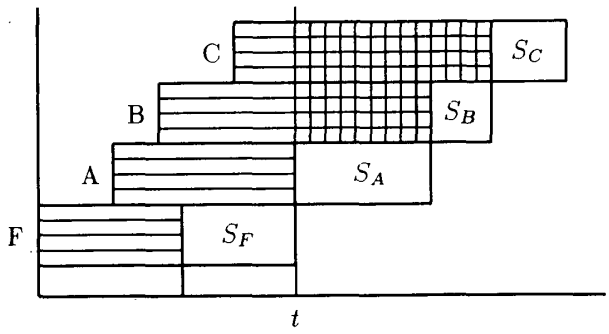

図 2: 待ち時間の比較

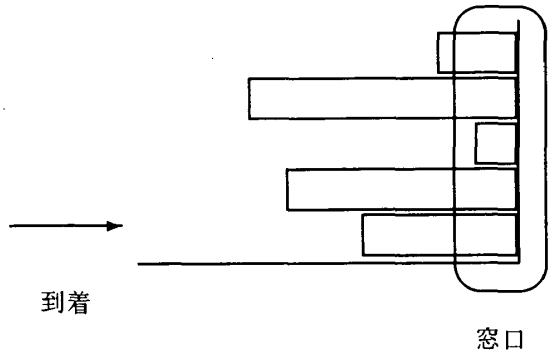

図 3: プロセッサーシェアリング

ち時間の合計 (図の格子のハッチ部分の面積) は

$$W_t(\text{先着順}) = 2S_A + S_B$$

となります。ここで S_A, S_B, S_C はそれぞれ客 A, B, C のサービス時間です。

3 人のサービス時間の長さが $S_B < S_C < S_A$ という順序であったとすると、最短順のときは B, C, A の順にサービスされます。するとこのときの t 以降の 3 人の待ち時間の合計は

$$W_t(\text{最短順}) = 2S_B + S_C$$

となりますが、これが上の W_t(先着順) よりも小さいことはサービス時間の大小関係から明らかです。最長順が最も大きくなることも、同様に確かめられます。

3 人のサービスの順番をこのようにいろいろ変えても、他の客の待ち時間には影響しませんので、待ち時間の合計は、したがって平均待ち時間も、最短順のときがもっとも短く、最長順がもっとも長くなります。ここでの議論にはポアソン到着や指数分布といった仮定は何も使っていませんので、この性質はすべての標準型待ち行列モデルに対して成り立ちます。

では、最短順ではどのくらい平均待ち時間 W_q が短くなるかというと、残念ながらこれは解析的には求められません。そこでシミュレーションで確かめてみました。表 1 がその結果です。ここでは平均サービス時間を 1 としています。

表 1: サービス順による平均待ち時間の違い

ρ	.5	.8
割り込み最短順	0.58	1.78
最短順	0.72	1.88
先着順	1.00	4.00
最長順	1.44	9.10

これをみると、利用率 ρ が 0.8 のときは、最短順では W_q が先着順のときの半分以下になることがわかります。ρ が 0.5 のときは、これほどの効果はありませんが、それでも待ち時間をかなり短くしています。

表 1 の割り込み最短順というのは、新たな客が到着したとき、その客のサービス時間がサービス中の客の残りサービス時間より短かければ、サービス中の客のサービスを一時中断し、到着した客のサービスを先に行う (割り込み、図 5 参照)、というサービス規律です。これは最短順をさらに徹底したもので、これが平均待ち時間を最も小さくすることが知られています。

2. 計算機とサービス規律

前節の最短順は、客のサービス時間があらかじめわかるときしか使えません。しかし実際には、客のサービス時間はサービスをしてみないとわからないのがふつうです。そこで計算機の世界では、サービス時間がわからないときでも実質的に最短順サービスに近づけるよう、いろいろな工夫がなされています。

2.1 プロセッサーシェアリング

そのひとつは、プロセッサーシェアリングといって、図 3 のように待ち行列は作らず、そのときに系内に n 個のジョブがあったとすると、サービス能力を $1/n$ ずつに分けて n 個のジョブを少しずつ同時にサービスするものです。たとえば、あるジョブが、そのサービス中、ずっと $n-1$ 個のジョブと一緒だったとすると、そのジョブのサービスには本来のサービス時間の n 倍かかるということです。

こうすると、最短順ほどではありませんが、サービス時間の短いジョブは長いジョブよりも早めにサービスが終了され、全体として平均系内滞在時間が小さくなることが期待されます。

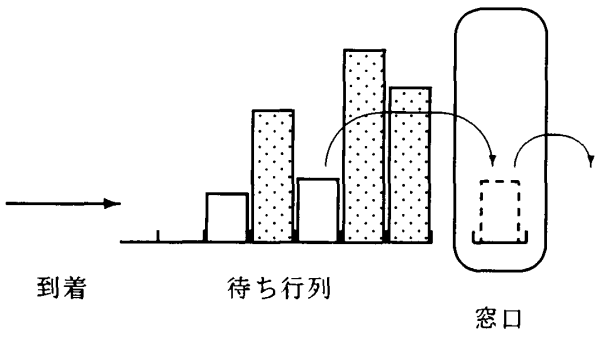

図 4: 非割り込み 優先権

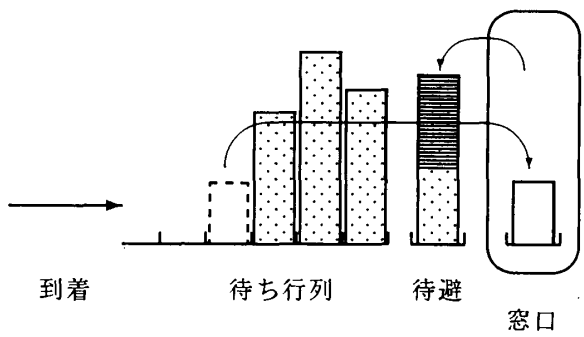

図 5: 割り込み継続型 優先権

2.2 優先権をつける

計算機に投入するジョブを、バッチジョブと会話型ジョブの2種類に分類することがよくあります。

会話型ジョブは、エディタを使うなど、ちょっと計算機で試してみて、その結果をもとにまた修正する、といったごく短い計算時間のジョブで、結果がすぐに返ってくることが要求されます。一方、バッチジョブは計算時間が長く、結果が返ってくるのに多少時間がかかることは仕方がありません。そこで、この違いを利用して、会話型ジョブに優先権を与えます。

優先権の与え方には何通りかありますが、代表的なのは非割り込み優先権と割り込み継続型優先権です。

非割り込み優先権というのは、ある客のサービスが終了したときに、図4のように待ち行列にバッチジョブ (網掛け) と会話型ジョブ (白抜き) の両方が待っているときは、会話型ジョブを優先してサービスするものです。図4では、待ち行列の先頭と2番目のバッチジョブを3番目に並んでいる会話型ジョブが追い越して、つぎにサービスを受けます。一般に会話型ジョブの方がバッチジョブよりも短いので、これは最短順と同様の効果があり、平均待ち時間が短くなるのです。

割り込み継続型優先権では、つぎにサービスされるジョブとして会話型ジョブが優先されることに加えて、会話型ジョブが到着したときにバッチジョブがサービス中であれば、そのバッチジョブのサービスを一時的に中断し、到着した会話型ジョブをサービスします (図5)。ではクイズ、

クイズ 2

計算機でジョブの平均ターンアラウンドタイム (平均系内滞在時間) を短くするには、先着順 (FCFS)、プロセッサーシェアリング (PS)、非割り込み優先権 (NP)、割り込み継続型優先権 (PR) のうち、どのサービス規律をとるとよいでしょう。

このクイズに答えるため、つぎのような待ち行列モデルを考えてみましょう。

客にはタイプ1 (会話型ジョブ) とタイプ2 (バッチジョブ) があり、タイプ1の客はパラメータ λ_1 のポアソン過程にしたがって到着し、パラメータ μ_1 の指数分布にしたがう時間だけサービスを受けます。タイプ2の客も同様で、パラメータはそれぞれ λ_2, μ_2 です。記号を少し導入しておきましょう。

$$\rho_1 = \mu_1^{-1}\lambda_1, \quad \rho_2 = \mu_2^{-1}\lambda_2 \tag{1}$$

$$\rho = \rho_1 + \rho_2, \quad b = \mu_1^{-1}\rho_1 + \mu_2^{-1}\rho_2 \tag{2}$$

すると、クイズのサービス規律のもとでのタイプ1とタイプ2の客の平均系内滞在時間 W_1 と W_2 は、表2のように与えられることが知られています。そして、全体の平均系内滞在時間は

$$W = \frac{\lambda_1 W_1 + \lambda_2 W_2}{\lambda_1 + \lambda_2} \tag{3}$$

で求められます。W_1 と W_2 の間には

$$\rho_1 W_1 + \rho_2 W_2 = \text{一定} \tag{4}$$

という関係がありますので、W_1 と W_2 の両方を小さくすることは不可能です。しかし $\lambda_1 > \lambda_2$ という関

表 2: サービス規律と平均系内滞在時間の公式

	W_1	W_2
FCFS	$\frac{b}{1-\rho} + \mu_1^{-1}$	$\frac{b}{1-\rho} + \mu_2^{-1}$
PS	$\frac{\mu_1^{-1}}{1-\rho}$	$\frac{\mu_2^{-1}}{1-\rho}$
NP	$\frac{b}{1-\rho_1} + \mu_1^{-1}$	$\frac{b}{(1-\rho_1)(1-\rho)} + \mu_2^{-1}$
PR	$\frac{\mu_1^{-1}\rho_1}{1-\rho_1} + \mu_1^{-1}$	$\frac{b}{(1-\rho_1)(1-\rho)} + \frac{\mu_2^{-1}}{1-\rho_1}$

FCFS : 先着順　PS : プロセッサーシェアリング
NP : 非割り込み優先権　PR : 割り込み継続型優先権

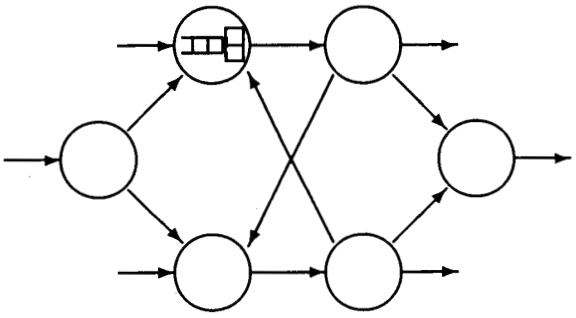

図 6: 開待ち行列ネットワーク

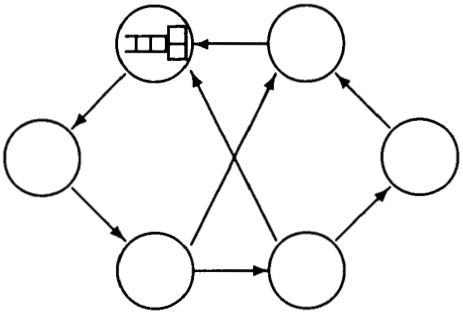

図 7: 閉待ち行列ネットワーク

係を利用して、W_2 の犠牲のもとに W_1 を小さくすることによって、全体の平均系内滞在時間 W を小さくすることは可能です。(この原理は最短順でも同じです。つまりある種の不公平を導入しているのです。)

具体的に数値をあてはめて、平均系内滞在時間はどれが短いかみてみましょう。結果は表 3 の通りです。

表 3: サービス規律による平均系内滞在時間の違い

$\lambda_1 = 2,\ \lambda_2 = 1,\ \mu_1 = 20,\ \mu_2 = 2$

	W_1	W_2	W
FCFS	.6875	1.1375	.8375
PS	.1250	1.2500	.5000
NP	.3333	1.2083	.6250
PR	.0556	1.2639	.4583

先着順 (FCFS) と較べて、プロセッサーシェアリング (PS) と割り込み継続型優先権 (PR) が優れていることがわかります。実際、多くの汎用計算機では割り込み継続型優先権が、また UNIX 系の計算機ではプロセッサーシェアリングが採用されています。

3. 待ち行列ネットワーク

計算機や通信システム、生産システムなどでは、客がシステムの中で何度もサービスをうけて出ていきます。そのため待ち行列もたくさんできてきます。このような複雑なシステムをモデル化するには、待ち行列がネットワーク状につながった“待ち行列ネットワーク”モデルが必要です。

待ち行列ネットワークは、客がネットワークの外から入ったり外へ出たりするかどうかによって、開ネットワークと閉ネットワーク、およびそれらの混合型に大別することができます。

開待ち行列ネットワークでは、図 6 のように、客は外から入ってきて、あちこちのノード (個別の待ち行列) を訪問してサービスを受けた後、ネットワークの外へ出ていきます。図 8 の直列型待ち行列システムは生産ラインの性能評価に欠かせないものですが、これは典型的な開ネットワークです。

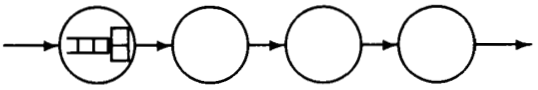

図 8: 直列型待ち行列システム

一方、図 7 の閉待ち行列ネットワークでは、客はネットワークの外から入ったり外へ出たりしません。したがって常に一定の数の客がネットワークの中を動き回っていることになります。計算機システムの代表的なモデルである図 9 のセントラルサーバモデルは、典型的な閉ネットワークです。

待ち行列ネットワークは、単独の待ち行列モデルがいくつもつながっているのですから、大変複雑なモデルになり、そう簡単に解析することはできません。ただ、ジャクソン型ネットワークという M/M/c 待ち行列がつながったような特殊なケースについては、積形式解という大変きれいな結果が得られています。

4. ジャクソン型ネットワーク

つぎのような待ち行列ネットワークをジャクソン型ネットワークといいます[1]。

N 個のノードがあり、ひとつひとつのノードは窓

[1] 本文の仮定はもっとゆるめられます。たとえば、窓口の数は複数でもよいし、サービス率はそのときのノードにいる客の数に依存してもかまいません。また、外からの客の到着率も、ネットワーク内の客の総数に依存してかまいません。ただし、待合い室の容量は無限でなければなりません。有限のときはブロッキングという面倒な状況が発生します。

また、ジャクソン型をさらに一般化した BCMP 型ネットワークというのもあります。

口と待ち行列から成っています。ここでは簡単のためすべて単一窓口であるものとしましょう。ノード j の窓口では、パラメータが μ_j の指数分布にしたがったサービスが行われます。ノード i でサービスを終了した客は、確率 r_{ij} でノード j へ進みます。

さらに、開ネットワークでは、外部からノード j への客の到着は、パラメータ λ_j のポアソン過程に従います。また、ノード j でサービスを終了した客は、確率 r_{j0} でネットワークを去ります。

むろん、客の到着、サービス、ノード選択などの確率現象は、みな互いに独立であるものと仮定します。

4.1 開ネットワーク

$\theta_j,\ j=1,2,\ldots,N,$ を方程式

$$\theta_j \;=\; \lambda_j + \sum_{i=1}^{N} \theta_i r_{ij}, \quad j=1,2,\ldots,N \qquad (5)$$

の解とします。この θ_j は、ノード j への到着率とみなせます。そこで $\rho_j = \dfrac{\theta_j}{\mu_j}$ と置くことにしましょう。すると、十分時間がたったとき、ノード $j,\ j=1,2,\ldots,N,$ に n_j 人の客がいるという状態の確率は

$$p(n_1,\ldots,n_N) \;=\; \prod_{j=1}^{N} \frac{1}{1-\rho_j}\rho_j^{n_j} \qquad (6)$$

で与えられることが知られています。

(6)から、各ノードはあたかも互いに独立な M/M/1 待ち行列モデルであるかのように考えられます[2]。したがって、各ノードにおける平均滞在時間やネットワークに入ってからそこを去るまでの平均滞在時間などは、簡単に計算することができます。この意味で、ジャクソン型の開ネットワークは容易に解析することができるといってよいでしょう。

4.2 閉ネットワークと積形式解

条件はほとんど同じでも、閉ネットワークの場合には状況がかなり異なります。ネットワーク内に K 人(一定)の客がいるものとしましょう。

$\theta_j,\ j=1,2,\ldots,N,$ を方程式

$$\theta_j \;=\; \sum_{i=1}^{N} \theta_i r_{ij}, \qquad j=1,2,\ldots,N \qquad (7)$$

の解とします。ただしこの方程式から求められた θ_j は、定数倍の自由度があります。そこで、ひとつの θ_j (たとえば θ_1) の値は任意に定めることができます。

[2]厳密にいうと、互いに独立なのは平衡状態における同時分布だけで、ノード内客数の変化などの過程は互いに独立にはなりません。

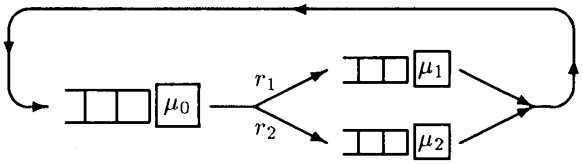

図 9: セントラルサーバモデル

十分時間がたったときに、ノード $j,\ j=1,2,\ldots,N,$ に n_j 人の客がいる確率は、$\sum_{j=1}^{N} n_j = K$ のとき、

$$p(n_1,\ldots,n_N) \;=\; \frac{1}{G}\prod_{j=1}^{N}\left(\frac{\theta_j}{\mu_j}\right)^{n_j} \qquad (8)$$

となることが知られています。ここで G は (8) の和が1となるような正規化定数です。この式は各ノードに対応する項の積の形をしていますので、"積形式解"と呼ばれています。

開ネットワークのときはどんな n_j の組み合わせに対しても (6) の式が成り立ちました。閉ネットワークの場合、(8) が成り立つのは $\sum_{j=1}^{N} n_j = K$ となるときだけです。したがって、各ノードは独立ではありません。そのため正規化定数 G を求めること、さらには各ノードでの平均客数やスループットといったネットワークの特性量を求めることは、簡単にはできません。このためたたみこみ法や平均値解析法といった閉ネットワーク専用の解析手法が開発されています。

4.3 セントラルサーバモデル

閉待ち行列ネットワークの例として、計算機の典型的なモデルであるセントラルサーバモデルをみてみましょう。これは図9のように、CPU に対応する1つのノードと、ディスクに対応する複数の並列ノードからなっているモデルです。ここでは簡単のため、ディスクは2台しかないものとします。

CPU で平均 μ_0^{-1} の指数分布にしたがう時間サービスされたジョブは、確率 r_1 でディスク1へ、確率 r_2 でディスク2へ進みます。ディスク1と2ではそれぞれ平均 μ_1^{-1}, μ_2^{-1} の指数分布にしたがう時間サービスを受け、また CPU へ戻ります。

このモデルでは、システム内にあるジョブの数 K は一定であるものと仮定されています。実際の計算機では、ジョブの多重度が設定されていることが多く、CPU にアクセスできるジョブの数は一定に制限されています。各ジョブは何回か CPU で処理されるとシステムの外に出ていきますが、すると入れ替わりにシ

ステムの外で待っていたジョブが入ってきます。このように、ジョブ数一定という仮定は不自然ではなく、むしろ理にかなったものなのです。

具体的な数字を入れて、各ノードでの平均ジョブ数などを求めてみましょう。平均値解析法という計算方法を用いると簡単に求められるのですが、紙数の関係で詳細は省略いたします。

例　　$\mu_0 = 1$　　$\mu_1 = .6$　　$\mu_2 = .4$

$r_1 = .6$　　$r_2 = .4$

とします。このとき、システム内の客数が 10 人のときと 100 人のときの、ノード j における平均客数 L_j と、ノード j を 1 回通過するのにかかる時間の平均 W_j は、表 4 のようになります。

表 4: バランスしたセントラルサーバモデルの例

	$K = 10$		$K = 10$	
j	L_j	W_j	L_j	W_j
0	3.333	4.000	33.333	34.000
1	3.333	6.667	33.333	56.667
2	3.333	10.000	33.333	85.000

ここでクイズです。

クイズ 3

上のセントラルサーバモデルで、パラメータがつぎのように少し変わったら、システム内総ジョブ数 $K = 100$ のとき、平均ノード内ジョブ数 L_j はどのように変わるでしょうか。

ケース 1　　$r_1 = .5$　　$r_2 = .5$

ケース 2　　$\mu_1 = .8$　　$\mu_2 = .6$

これは、ぜひ、答えをみる前にご自分で考えてみてください。たとえば、ケース 1 では r_1 が少し減って r_2 が増えたのだから、上の例から類推すると・・・。

じつは、表 5 のようになります。

どうですか、全く予想はあたりましたか。これはわざと意地悪な出題をしたのです。はじめの例では、たまたま各ノードの負荷がバランスしていたので、L_i もバランスしていたのですが、これはとても特殊な場合なのです。たいていはどこか 1ヶ所がボトルネックとなって、そこだけに客が集まってしまいます。ケース 1 ではディスク 2 がボトルネックとなり、ケース 2 では CPU ノードがボトルネックになっています。

表 5: クイズ 3 のセントラルサーバモデル

	ケース 1		ケース 2	
j	L_j	W_j	L_j	W_j
0	4.000	5.000	95.000	95.000
1	2.000	5.000	3.000	5.000
2	94.000	235.000	2.000	5.000

少々パラメータをいじってみても、ジョブが確率的にルート選択している限り、ボトルネックができるという性質を回避できません。これを避けるには、強力なコントロールを導入するしかありません。

現在、構築が始まっている超高速情報ネットワークでは、ノードの混雑状況を情報発信源に知らせることによって、混雑をコントロールしようとしています。その効果を評価し、信頼性と効率性とを兼ね備えたネットワークを構築することは、いま、緊急の課題です。

5. 新しい待ち行列理論を求めて

待ち行列ネットワークに関して、残念ながら待ち行列理論はまだまだ未熟です。ジャクソン型の仮定が少し崩れただけでも、その挙動はまだほとんどわかっていません。ましてや、コントロールが入ったり、動画データのように従来のポアソン到着とは全く異なる到着パターンが出現すると、それをどう扱うか、という段階から考え直さなければなりません。

現在、情報通信ネットワークが急速に発展していますが、その性能評価をタイムリーに行うためにも、待ち行列理論は大いに期待されています。従来の解析手法にとらわれず、新しい発想で新しい理論構築を行う時期にきたのだと思います。若い方々にも、ぜひ、研究に参加して、新しい待ち行列理論を一緒に作り上げていってほしいものです。

この 4 回シリーズもいよいよ今回で終わりですが、長いことお付き合いくださいましてありがとうございました。参考文献はつぎの補遺をご覧ください。

このシリーズを執筆するにあたり、待ち行列研究部会および高校生のための OR 研究部会のメンバーの方々から貴重なご意見をたくさん頂戴いたしました。また、今回のシミュレーションを含め、原稿作成に全面的に協力してくれた大学院生の大原久樹君はじめ、東京工大の高橋・牧本研究室の諸君にもお世話になりました。この場を借りて深く感謝いたします。