

フロッキングを考慮した 直列型待ち行列モデルの数値解析

筑波大学 社会工学系

逆瀬川 浩孝

1. はじめに

いくつかの窓口が直列型に配置され、客はこれらの総ての窓口を順に通過し、退去するという、いわゆる直列型待ち行列モデルについて考える。このモデルを解析する為には、一つの窓口からの客の退去過程を知る必要がある。良く知られているように、 $M/M/s$ あるいは $M/G/\infty$ の退去過程は、到着と同じポアソン過程になるため、このような窓口のみの直列型モデルならば、本質的には単段モデルと同様に取り扱うことができる。また、それらのモデルの拡張として、いわゆる Kelly システム、あるいは BCMP 型窓口モデルを個々の窓口としても同様なことが言える。これらのモデルは、見かけ上は直列型でありながら、窓口相互が互いに独立に機能しているため、単段モデルの拡張にはなっていない、ということと言える。

直列型モデルを特徴付けるのは、窓口相互の干渉の存在である。例えば、各窓口での待合室の大きさと有限の値に制限すると、あるサービスを終了した客は、次の待合室に入るとする時、そこには他の客をみたさぬといへば先に進むことはできる。このため、窓口の次の客へのサービスに妨害が生じなくなる。これを窓口のフロッキングと呼ぶ。このようなフロッキングを供う待ち行列モデルについては、ポアソン到着、指数サービスの極く基本的なモデルについては、解析的に解くことはできない。二段モデルについては、母関数、あるいは差分方程式にもとづく数値的な解法が与えられる。それより大きなモデルに対しては、マルコフ連鎖モデルに帰着させ、平衡方程式を解くしかないようである。これは、与えられたモデルに対して、その平衡方程式を構成し、それをもとに平衡確率を求め、諸特性量を計算する為の計算機プログラムや試作例を報告する。マルコフモデル化する為には、アーラン分布を相分解するように、補助変数を導入する必要がある。平衡方程式の次元は、モデルの大きさと共に、指数的に増大するので、大きなモデルに対しては、この方法は余り有効ではなく、何らかの近似法を考えたほうがいい。しかし、近似による前に、どの程度のモデルならば近似を使わなくても解けるのか、ということは知らなくては必要があり

また、近似法の基礎として、このモデルに対して最適解を用意しておく必要もある。更に、マルコフ・モデル化することの利点は、この平衡解を反復法で解くことにより、この過程で、一時解をシミュレートすることからなる点にある。このようにいくつかの点で、試作プログラムは役に立つと思われる。

2. モデルの記述

K 段からなる直列型待ち行列モデルを考える。各段は C_k 個の窓口と、最大 $N_k - C_k$ の待合室からなり、そのサービス時間分布は、 λ_k -ア-ラン型、平均サービス時間を μ_k^{-1} とする。客の到着は第1段のみで、到着間隔は λ_0 -ア-ラン分布に従い、その平均は λ^{-1} である。 $k (< K)$ 段で、ある客のサービスが終了した時点で、 $k+1$ 段に $N_{k+1} < \infty$ の客がいる時、その客は先に進み、 k 段の窓口を占拠したまま、移動する状態になるまで待つ (k 段窓口をブロックするという)。 $k (> 1)$ 段である客のサービスが終了したときその客は k 段を退去した時点で、 $k-1$ 段にブロックされた窓口があれば、そのうちの一つを占拠してその客は k 段に進み、その窓口のブロックを解除する。ブロックされた窓口が2つ以上ある場合、もし、系内滞在時間の分布を問題にするのであれば

ば、どの窓口のフロップを先に解除するか、という規律の問題となる。ここには系内容数について調べる必要はない。うにフロップに選んでよい。ある窓口のフロップが解除されることによ、その直前の段の窓口のフロップが解除される場合も同様に扱う。第一段に N_1 人の客が $t=0$ 時に到着した客は拒否され、失われる。

3. 解析

モデルのマルコフ表現は、次のように確率変数を導入することによ、可能となる。 R_0 を到着過程の残りフェース数とする。 Q_k を k 段の系内容数、 $R_{k1}, \dots, R_{kc_k}$ を k 段の窓口のサーベヤの残りフェース数を大きさの順に並べたものとする。客が来ないか、フロップされた窓口の残りフェース数は 0 である。

$$(R_0(t), (Q_k(t), R_{k1}(t), \dots, R_{kc_k}(t))_{k=1,2,\dots,K})$$

は連続時マルコフ連鎖である。その状態集合を $\mathcal{A}_K$ 、生成行列を $A = (a(s, t))$ とすると、 $a(s, t)$ は次の式で与えられる。 $t = s$ 。

$$s = (r_0, (q_k, r_{k1}, \dots, r_{kc_k})_{k=1,\dots,K}) \equiv (r_0, (n_k)_{k=1,\dots,K})$$

$$c_k(k) = \min(q_k, c_k)$$

$$\xi_0(s) = s(r_0 \rightarrow r_0 - 1) \quad (r_0 > 1)$$

$\therefore \therefore \rho(a \rightarrow l) \text{ は } \wedge \gamma \vdash \text{ル } \rho \text{ の } a \text{ 成分 } r_2 + \varepsilon \text{ 以上 } l_1 = \frac{1}{2}$

之より得る。他は不変の $\wedge \gamma \vdash \text{ル } \varepsilon$ 表す a とする

$$\xi_0(s) = \rho(r_0 \rightarrow l_0) \quad (r_0 = 1)$$

$$\xi_k(s) = \rho(q_k \rightarrow q_{k+1})$$

$$\eta_k(s) = \begin{cases} \xi_k(s) & (c_k \leq q_k \leq N_k) \\ \rho(\underline{m}_k \rightarrow (q_{k+1}, l_k, r_{k1}, \dots, r_{k, l(k)})) & (q_k < c_k) \end{cases}$$

$$\zeta_k(s) = \begin{cases} \rho(\underline{m}_k \rightarrow (q_{k-1}, r_{k1}, \dots, r_{k, l(k)-1})) & (q_k \leq c_k) \\ \rho(\underline{m}_k \rightarrow (q_{k-1}, l_k, r_{k1}, \dots, r_{k, l(k)-1})) & (c_k < q_k \leq N_k) \end{cases}$$

$$\bar{\zeta}_k(s) = \xi_k \zeta_k(s)$$

$$\xi_{kj}(s) = \rho(r_{kj} \rightarrow r_{kj-1})$$

$$\alpha_j(\underline{m}_k) = \begin{cases} \# \{i; r_{ki} = r_{kj} + 1, i \leq j\} & (r_{kj} \neq r_{kj+1}) \\ 0 & (\exists u \wedge \gamma \vdash) \end{cases}$$

と定まる。

$$a(s, \xi_0(s)) = l_0 \lambda \quad (r_0 > 1 \text{ 又は } r_0 = 1 \text{ かつ } q_1 = N_1)$$

$$a(s, \xi_1 \xi_0(s)) = l_0 \lambda \quad (r_0 = 1 \text{ かつ } c_1 \leq q_1 < N_1)$$

$$a(s, \eta_1 \xi_0(s)) = l_0 \lambda \quad (r_0 = 1 \text{ かつ } q_1 < c_1)$$

$$a(s, \xi_{kj}(s)) = \alpha_j(\underline{m}_k) l_k \mu_k \quad \left(\begin{array}{l} r_{kj} > 1 \text{ 又は } j \neq l(k) \text{ 又は} \\ j = l(k) \text{ かつ } q_{k+1} = N_{k+1} \end{array} \right)$$

$$a(s, \zeta_k \eta_{k+1}(s)) = \alpha_{l(k)}(\underline{m}_k) l_k \mu_k \quad (r_{k, l(k)} = 1 \text{ かつ } r_{k-1, l(k-1)} > 0)$$

$$a(s, \zeta_{i-1} \bar{\zeta}_i \dots \bar{\zeta}_k \eta_{k+1}(s)) = \alpha_{l(k)}(\underline{m}_k) l_k \mu_k$$

$$(q_i > 0, r_{i, l(i)} = 0 \ (i = j, \dots, k) \text{ かつ } r_{j-1, l(j-1)} > 0, r_{k, l(k)} = 1)$$

状態集合が確定し、生成行列が与えられるば、原理的には平衡確率を計算する：とはできるが、与えられるべきパラメータに平衡方程式を構成する：とは大変なことで、その為の計算機プログラムを作ることを考える。それにはいくつかの補助的な道具が必要である。

(1) 状態集合 S_K の大きさ、あるいは平衡方程式の次元、を求める漸化式。

$\underline{n}_k = (q_k, r_{k1}, \dots, r_{k c(k)})$ は次の条件を満たし、正しい数ベクトルである。

$$l_k \geq r_{k1} \geq \dots \geq r_{k c(k)} \quad , \quad 0 \leq q_k \leq N_k$$

$$r_{k c(k)} \geq 1 \quad (q_{k+1} < N_{k+1} \text{ 又は } k=K)$$

$$r_{k c(k)} \geq 0 \quad (q_{k+1} = N_{k+1} \text{ かつ } k < K)$$

$q_k = i$, $r_{k c(k)} \geq \delta$ ($\delta = 0, 1$) の場合、このように組合せの数

$$\binom{l_k + i - \delta}{i} \quad (q_k = i < c_k)$$

$$\binom{l_k + c_k - \delta}{c_k} \quad (q_k = i \geq c_k)$$

によつて与えられる。よって

$$\beta_k(\delta) = \sum_{i=0}^{c_k-1} \binom{l_k + i - \delta}{i} + (N_k - c_k) \binom{l_k + c_k - \delta}{c_k}$$

とおくと、次の漸化式によつて S_K の大きさを計算する：ことができる。

$$f(0,0) = f(0,1) = l_0$$

$$f(k,\delta) = \beta_k(\delta) f(k-1,1) + \binom{l_k + c_k - \delta}{c_k} f(k-1,0) \quad (k=1,2,\dots)$$

$$=: z \quad f(K,1) = |S_K| \quad z \text{ である。}$$

(2) S_K の要素を整理させるアルゴリズム、すなわち、

S_K から $\{0, 1, \dots, f(K,1)-1\}$ の写像 g_K

$S = (r_0, (q_k, r_{k1}, \dots, r_{kA(k)})_k)$ から z へと変換する。

$$1. \quad g_K \leftarrow 0, \delta \leftarrow 1, k \leftarrow K$$

$$2. \quad q_k \leq 0 \text{ ならば } \delta \leftarrow 1 \text{ 以上 } q_k \text{ へ}$$

$$3. \quad g_K \leftarrow g_K + f(k-1,1) \left\{ \sum_{j=1}^{q_k-1} \binom{l_k - \delta + \min(j, c_k)}{l_k - \delta} \right\}, m \leftarrow 0, j \leftarrow \delta$$

$$4. \quad \text{もし } \min(q_k, c_k) \leq 1 \text{ ならば } \delta \text{ へ}$$

$$5. \quad i \leftarrow \min(q_k, c_k)$$

$$6. \quad l_{ki} \leq j \text{ ならば } \delta \text{ へ}$$

$$7. \quad m \leftarrow m + \binom{l_k + i - 1 - j}{i-1}, j \leftarrow j+1, i \leftarrow i-1. \text{ もし } i \geq 2 \text{ ならば } 6 \text{ へ}$$

$$8. \quad m \leftarrow m + l_{k1} - j. \text{ もし } q_k = N_k \text{ ならば } \delta = 0. \text{ もし } 4 \text{ へ}$$

$$\delta \leftarrow 1, g_K \leftarrow g_K + m \cdot f(k-1, \delta)$$

$$9. \quad k \leftarrow k-1. \text{ もし } k \geq 1 \text{ ならば } 2 \text{ へ. もし } 4 \text{ へ}$$

$$g_K \leftarrow g_K + r_0 - 1$$

(3) m 番目の状態へのフトルを生成するアルゴリズム、すな

わち、 g_K の逆写像。

$$0. \quad k \leftarrow K, \delta \leftarrow 1$$

$$1. \quad l \leftarrow \left\lfloor \frac{m}{f(k-1,1)} \right\rfloor, l_k \leftarrow \sum_{j=0}^{c_k-1} \binom{l_k + j - \delta}{j}, a_k \leftarrow l_k + (N_k - c_k) \binom{l_k + c_k - \delta}{c_k}$$

2. $\text{if } l \geq a_k \text{ then } g_k \leftarrow N_k, m \leftarrow m - a_k f(k-1, 1), l \leftarrow \left\lfloor \frac{m}{f(k-1, 0)} \right\rfloor$
 $j \leftarrow c_k \text{ and } l \geq 6$
3. $m \leftarrow \text{mod}(m, f(k-1, 1))$.
 $\text{if } l \geq l_k \text{ then } g_k \leftarrow c_k + \left\lfloor \frac{l - l_k}{\binom{l_k + c_k - \delta}{c_k}} \right\rfloor, l \leftarrow \text{mod}(l - l_k, \binom{l_k + c_k - \delta}{c_k})$
 $j \leftarrow c_k \text{ and } l \geq 6$
4. $\text{if } l \leq 0 \text{ then } m_k \leftarrow (0, \dots, 0) \text{ and } l \geq 11$
 $\text{if } \text{not } \text{if } \text{then } j \leftarrow 1, l \leftarrow l - 1 \text{ and } l \geq 5$
5. $l < \binom{l_k + j - \delta}{j} \text{ then } g_k \leftarrow j \text{ and } l \geq 6$
 $\text{if } \text{not } \text{if } \text{then } l \leftarrow l - \binom{l_k + j - \delta}{j} \text{ and } l \geq 5$
6. $r_{k, j+1} = \dots r_{k, c_k} \leftarrow 0, i \leftarrow j$
7. $l < \binom{l_k + i - 1 - \delta}{i-1} \text{ then } r_{ki} \leftarrow \delta, i \leftarrow i - 1 \text{ and } l \geq 7$.
 $\text{if } \text{not } \text{if } \text{then } h \leftarrow 0 \text{ and } l \geq 8$
8. $l \leftarrow l - \binom{l_k + i - 1 - h - \delta}{i-1}, h \leftarrow h + 1$
9. $\text{if } l \geq \binom{l_k + i - 1 - h - \delta}{i-1} \text{ then } \text{if } 8 \text{ then } \text{if } \text{not } \text{if } \text{then}$
 $r_{ki} \leftarrow h + \delta, i \leftarrow i - 1 \text{ and } i > 1 \text{ then } 9$
10. $r_{k1} \leftarrow l + h + \delta$
11. $k \leftarrow k - 1, \text{if } k \geq 1 \text{ then } 1$.
 $\text{if } \text{not } \text{if } \text{then } r_0 \leftarrow m + 1$

次に、与えられた平衡方程式から、平衡状態確率 $\pi = (\pi_0, \dots, \pi(f(k, 1) - 1))$ を計算するアルゴリズムを考えよう。 g_k によ、2 整列された状態の並べ方による生成行列を、改めて

$$A = (a(j, k)) \quad , \quad a(j, j) = -\sum_{k \neq j} a(j, k)$$

とおけば

$$\begin{cases} \underline{x} A = 0 \\ \sum_j x_j = 1 \end{cases} \quad \begin{cases} \underline{x} = (x_0, \dots, x_{f(k,1)-1}) \\ x_j \geq 0 \quad (\forall j) \end{cases}$$

は唯一つの解 π を持つから、この連立方程式を解けば π が求まる。連立方程式の数値解法として有名な消去法と、この問題に對して適用するこゝは得策ではない。何故なら、このようにしてこの方程式の次元は数千のオーダーになり、消去法の基礎となる積和演算の積み重ねに危険が伴うのと、演算時間の次元の増大に伴い、2 指數的に増大するからである。

よって、総ての C について、

$$\pi(CA + I) = \pi$$

が成り立つことから、 π は行列 $CA + I$ の固有値 1 に對する固有ベクトルになる。特に $CA + I$ が確率行列になれば、これは、良く知られてゐるように、この固有値 1 は絶対値最大で、しかもこの正相結合性の場合には、単根になる。このような条件を満たす固有ベクトルは反復法によつて求まることを示す。その収束は、 $\rho =$ 固有根の絶対値をばう $\rho - \delta$ とし、幾何的であることが言える。 $\rho =$ 固有根の大きさは C の与え方によつて異なる。

$$C = (\max |a(j, j)|)^{-1}$$

とするのが良い。反復法を用いる場合の計算時間は、行列の次元の自乗に比例するから、 $n \times n$ 行列 $A+I$ はスパースなので、行列の次元に比例すると考えよう。数値実験の例では、数千の次元の固有ベクトルを 10^{-8} の精度で求めるのに要する計算時間は、総てを倍精度で計算しても1分以内、反復回数は数百回のオーダーであらうから、単純反復法でも十分に実用的であることが確かめられた。次元が大きくなると時の問題は、計算を多数回繰り返す時の累積するであろう計算誤差である。これについては、まだ今後の課題として残されている。

直列型待ち行列モデルの効率の尺度として、最大流出率、maximum throughput rate が良く知られている。これを計算する為には、到着過程の代りに、才一段に最初から長さ無限大の待ち行列が存在するような、いわゆる倉庫付きモデルを考え、その出力率を計算すれば良い。このモデルの平衡確率は上述の考え方と全く同様に求めることができる。到着過程がPoisson、倉庫を除く才一段には常に C_1 人の客がいる、という条件による変数箇所は、(1) $a f(0,1)=0, f(0,0)=1$ と、(2), (3) で $g_1 \equiv C_1$ としなすければ何ら問題ないと思われる。

4. 数値例

いふれも、最大流出量を評価基準にしたいところのモデルの比較を行つたものがある。

4.1 二段モデル

第一段、第二段とも、窓口の構成、サービス能力と同等にする。各段のサービス能力、すなわち窓口数とサービス率と掛け合せたもの、を一定にしたまま、窓口数を増減した場合と、第一段の待合室の大きさを増減した場合について、最大流出率の変化を調べる。 $c_1 = c_2 = c$, $l_1 = l_2 = l$, $c\mu_1 = c\mu_2 = 1$
 $c = 1(1)5$, $N_2 - c = 0(1)4$, $l = 1(1)3$ について調べるもの。
 表.1 である。バイパスを作為にサービス効率を平均的に低下させるより、バッファを置く効果の方が、スループットの向上に寄与し、その傾向は、サービス分布の変動係数の減少と共に顕著になることがわかる。なお、 $c=5$, $N_2=9$ の時、方程式の次元数は $l=2$ に 360 $l=3$ に 93 と 3675 になる。

4.2 三段モデル

ジョブジョブ型モデルで、最大流出率を最大にするようにサービス窓口を順序付ける問題を考える。各段とも窓口数が 2、サービス能力が 1、待合室の大きさが 2 (第一段を除く) であり、サービス分布はそれぞれ異なる。 E_2, E_3, E_4 の 3

種類のある場合、そのサービス時間分布の並べ方によつて最大流出率のとおりに変化するのは示したのと表に示してある。どう並べたか、結果にそれ程の差はみられぬ、というのからOR的結論であるが、その若干の違ひを調べると、変動係数の大きいものを中間に置くと最も流れが良くなる、という。単一窓口直列型モデルの場合の結果の一部逆転してゐることに注目される。その理由については説明されたことである。なお、この表を判るようには、単一窓口直列型モデルで成立してゐる容量の可逆性も、この場合には成立してゐない。

5. おわりに.

フローキニフ起因する複数窓口直列型待ち行列モデルの解析の殆んどは手付かずの状態であるのに対して、アーラン型分布を持つモデルを数値的に解くプログラムを作り、数値解法の可能性を考えた。このプログラムでは、方程式の次元を a 、生成行列の非零要素の数 b とすると、主要なテーブル用の領域が $20a + 8b$ バイト必要である。 a, b の比率はモデルの段数、サービス数に依存する行列のスパースの割合を表わす尺度である。平均的に $b \approx 7a$ と考えれば、1MBのメモリを持つ計算機を使うと、次元が一万ぐらい迄のモデルならば解けることになる。その範囲のモデルとしては、

に於てに並べると、表.3 のようなものがある。方程式の次元
を下げる場合には、モデル全体を同時に解く代りに、全体を
いくつかのサブシステムに分けて計算を繰り返すことなど
工夫が考えられる。それについては今後の課題である。

Table.1 窓口数と待合室設置の効果

(1) M サービス

SERVER	BUFFER				
	0	1	2	3	4
1	0.6667	0.7500	0.8000	0.8333	0.8571
2	0.7500	0.8000	0.8333	0.8571	0.8750
3	0.7907	0.8269	0.8525	0.8714	0.8861
4	0.8161	0.8447	0.8655	0.8815	0.8940
5	0.8339	0.8576	0.8753	0.8891	0.9002

(2) E_2 サービス

SERVER	BUFFER				
	0	1	2	3	4
1	0.7273	0.8214	0.8682	0.8957	0.9137
2	0.7889	0.8487	0.8834	0.9054	0.9204
3	0.8217	0.8657	0.8935	0.9121	0.9252
4	0.8427	0.8775	0.9009	0.9171	0.9288
5	0.8576	0.8865	0.9067	0.9212	0.9318

(3) E_3 サービス

SERVER	BUFFER				
	0	1	2	3	4
1	0.7619	0.8587	0.9006	0.9235	0.9378
2	0.8106	0.8750	0.9086	0.9282	0.9409
3	0.8389	0.8866	0.9146	0.9319	0.9435
4	0.8574	0.8954	0.9194		

Table.2 窓口の最適配置

(l_1, l_2, l_3)	$C=2$	$\bar{C}=2$
(3, 2, 4)	0.74209	0.67627
(4, 2, 3)	0.74208	0.67627
(2, 4, 3)	0.74191	0.67695
(2, 3, 4)	0.74190	0.67658
(3, 4, 2)	0.74186	0.67695
(4, 3, 2)	0.74183	0.67658

Table.3 平衡方程式 α 次元 α 大 α j

($l_0=1$; $l_j=1$, $N_j=N$, $c_j=c$ for all j)

l	c	N	K						
			2	3	4	5	6	7	
1	1	1	5	13	34	89	233	610	
•	•	2	11	41	153	571	2131	7953	
•	2	2	12	51	219	942	4053	17439	
2	1	2	29	169	985	5741	33461		
•	•	3	55	433	3409	26839			
•	2	2	48	396	3276	27108			
•	2	3	102	1170	13428				
3	2	2	130	1720	22780				

M/M/1(5) E /4(5) M/1(5) M/1(5) a 結合 7555