平成 29 年度 卒業論文

相乗りを考慮した
タクシー乗り場の待ち行列モデル

千葉大学工学部都市環境システム学科
14T0246M 松尾容典

指導教員：塩田茂雄
平成 30 年 2 月 1 日提出

目次

第 1 章

序論

1.1　研究背景

待ち行列理論では 1 つのサービス窓口に対して 1 つの客（団体）がサービスを受けるモデルを対象として扱われてきた．しかし実際のサービスにおいては共用が許されるケースも見受けられる．ジェットコースターは車両 1 台に対して複数の客が同時にサービスを受ける例や飲食店において 2 人用のテーブルに対して別々の客が相席をするような例が挙げられる．このうち後者のように共用の可否が客により異なるモデルについての研究はなされていない．よって本論文においては相乗りを考慮したタクシー乗り場の待ち行列モデル（以降相乗りモデル）を構築し，共用の可否が客により異なるモデルについて考察していく．

1.2　研究目的

本論文は相乗りモデルという共用の概念を取り入れた待ち行列モデルを確立し，これを解析し結果を考察することを目的としている．

1.3　本論文の構成

本論文は以下のような構成になっている．第 2 章では本論の前段階として待ち行列理論の概要，出生死滅過程を用いた解析，タクシー乗り場のモデルの解析，準出生死滅過程について述べる．第 3 章では第 2 章を元に相乗りモデルの状態遷移図，大域平衡方程式を導出して平均待ち時間や平均待ち行列長等の数値の計算方法とこのモデルの安定条件について述べる．第 4 章では大域平衡方程式の解析と時間駆動型シミュレーションの二通りの解析方法について述べる．第 5 章では第 4 章で述べた解析方法をもとに解析を行い，その結果の考察を，第 6 章では結論を述べる．

第 2 章

準備

第 2 章では本論に入る前に予備知識として待ち行列理論の概要，出生死滅過程を用いた解析，タクシー乗り場のモデルの解析，準出生死滅過程について述べる．

2.1　待ち行列理論の概要

2.1.1　待ち行列理論とは

待ち行列理論とは遊園地のアトラクション待ちの列やチケット売り場の行列などといった待ち行列について数理的に解析する理論である．待ち行列理論の歴史は古く [1]，1907 年に発表されたアーランの論文 [2] に端を発する．待ち行列理論を用いることで行列に並んだ客がサービスの利用を終了するまでにどれくらい時間がかかるのか（平均待ち時間），待ち行列には常に何人程度の客が並んでいるのか（平均待ち行列長），サービス窓口のキャパシティがオーバーしないためには客がシステムを訪れる頻度（到着率）と客にサービスを提供する時間（サービス時間）にどのような関係性が成立するのかといったことを数理的に解析することができる．

2.1.2　待ち行列のモデル化

では現実のシステムはどのようにモデル化すればよいのだろうか．まずメインとなる待ち行列システムについて述べると，待ち行列システムは待ち室と窓口の 2 つの要素により成り立つ．待ち室は遊園地でアトラクションに乗るまでにできる行列などで例えることができる．待ち室は駅において用いられる複数列による整列乗車のように複数の列を構成することもあるが一般的には 1 列のものが多い．窓口は遊園地のアトラクションがまさにそれである．窓口も待ち室と同様に複数の窓口をもつケースがある．複数台 ATM が設置してある ATM コーナーなどはまさに複数の窓口をもつケースである．また，窓口で受ける行為をサービスと呼ぶ．

このシステムを利用する個体のことを客と呼ぶ．客と呼称するものの客は人には限らず車やデータパケットといったものを指すこともある．そして客がシステムを訪れることを到着といい，サービスの利用が終了してシステムから立ち去ることを退去という．以上の要素を図示したものが図 2.1 である．

2.1.3　待ち行列の特徴の表現 (ケンドールの記号)

個々の待ち行列を特徴づけるものとして客の到着過程，客のサービス時間分布，窓口の数，待ち室の容量，客の到着処理方法（サービス規律）の 5 つが挙げられる．これらの特徴はケンドールの記号 [3] により簡潔に表すことが通例となっている．$A/B/c/K$ の順に客の到着過程，客のサービス時間分布，窓口の数，待ち室の

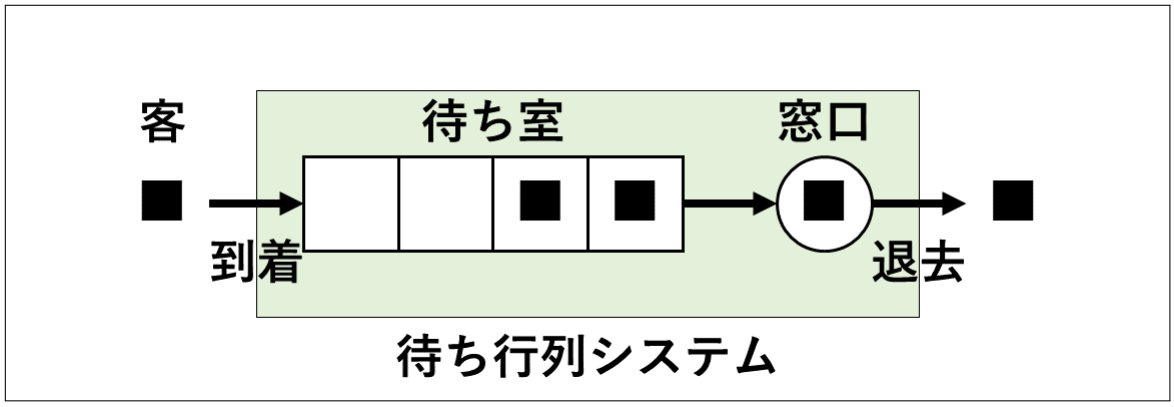

図 2.1　待ち行列モデルの構成要素

容量を表す．サービス規律はこの後ろに記載する．なお，以下に記載する到着過程やサービス時間分布などはあくまで一例でありほかにも多くの記号が存在する．詳しくは待ち行列理論の基礎と応用 [4] や待ち行列理論 [5] を参照されたい．

客の到着過程

客の到着過程は客の到着確率分布を記入することで表現している．ポアソン過程などの指数分布に従う到着過程はケンドールの記号の A の部分に M と記入する．もう 1 つ扱われることの多い到着過程として客の到着分布の独立性を考慮しない一般過程がある．こちらは G と記入する．

客のサービス時間分布

客のサービス時間分布についても客の到着過程と同様に指数分布で表される場合は B の部分に M と記入する．客のサービス時間分布を特に規定せず（すなわち一般分布）かつ独立性を保証しない場合には G と記入する．

窓口の数

窓口の数はシステム内の窓口の個数を c に記入する．なお，窓口の数が客の数に比べて非常に大きいモデルにおいては ∞ と記入する．

待ち室の容量

待ち室に入れる客の数を K に記入する．c, K ともに 1 の時はサービスを受けている客がいるときは待ち行列に加われないモデルを表す．また，待ち室の容量が特に制限されていない場合は ∞ と記入せずに何も書かず省略することが多い．

サービス規律

先着順に客のサービス処理を行う先着順サービスは FIFO や FCFS といった文字を $A/B/c/K$ の後ろに記入する．ほかに待ち室の先頭に並ぶサービス規律である非割り込み膠着順サービスは LIFO-NP や LCFS-NP と記入する．後着順サービスには今現在サービスを受けている客のサービスを一時中断して割り込むタイプのサービス規律も存在しこれを割り込み継続型後着順サービスといい LIFO-PR や LCFS-PR と記入する．なお，先着順サービスの場合は記入することを省略する場合が多い．

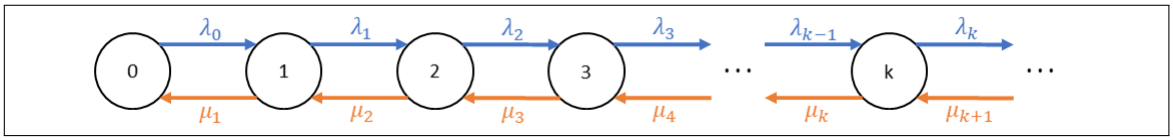

図 2.2　出生死滅過程の状態遷移図

2.1.4　リトルの公式

待ち行列理論において頻繁に用いられる公式としてリトルの公式というものがある．詳しくは待ち行列の数理とその応用 [6] を参照されたい．リトルの公式は客の到着率を λ，平均待ち行列長を L，平均待ち時間を W としたときに以下の式が成り立つというものである．

$$L = \lambda W \tag{2.1}$$

また，リトルの公式はシステム全体に限らず窓口のみを対象として用いることも可能である．窓口における待ち時間はサービス時間に等しく，例えば客のサービス時間分布が指数分布 μ に従うとき，サービス時間は $\frac{1}{\mu}$ となるため，

$$L = \frac{\lambda}{\mu} \tag{2.2}$$

となる．窓口のみを対象としたとき，L は窓口にどれぐらいの確率で客が存在するのかを表していて，これを利用率 ρ と表す．

2.2　出生死滅過程を用いた解析

2.2.1　出生死滅過程

出生死滅過程は客の到着過程がポアソン過程に従い，サービス時間分布が指数分布に従う待ち行列モデルを解析するために使われることが多い．出生死滅過程がどのような特徴を持つかを以下に示す．

- 系内客数が i であることを状態 i であるとする
- 一番客の数が少ない状態は状態 0 とする
- 過去の状態遷移に関わりなく現在の状態のみから未来の状態が決定する
- 状態 i の遷移先が毎回独立である
- 状態 i から一度のイベントで遷移できる先は状態 $i+1$ か状態 $i-1$ のどちらかに限る

以上の条件を満たしたものが出生死滅過程と呼ばれている．

ここで状態 i から状態 $i+1$ に遷移する確率を λ_i，状態 i から状態 $i-1$ に遷移する確率を μ_i とする．なお，状態 0 よりも小さい状態は存在しないため $\mu_0 = 0$ である．この確率をもとに状態遷移図を描くと図 2.2 のようになる．ここで出生死滅過程のシステムが定常である，つまり時刻に依存せず各状態が一意に定まるとすると各時刻において，ある状態から出力されるフローと入力されるフローは等しくなる．よって図 2.2 をもとにマルコフチェーン [7] を用いて式 (2.3),(2.4) のように方程式が書き下せる．ただし π_i は状態 i である確率を表

す．なお，この方程式は状態確率が平衡状態であるときに定まる式であり，平衡方程式と呼ばれている．

$$\lambda_0\pi_0 = \mu_1\pi_1 \tag{2.3}$$

$$(\lambda_k + \mu_k)\pi_k = \lambda_{k-1}\pi_{k-1} + \mu_{k+1}\pi_{k+1} \quad (k = 1, 2, \ldots) \tag{2.4}$$

ここで式 (2.3), (2.4) を $k = 0, 1, 2, \cdots, n-1$ まで両辺を足し合わせると式 (2.5) が導出される．

$$\begin{aligned}
\lambda_0\pi_0 &= \mu_1\pi_1 \\
(\lambda_1 + \mu_1)\pi_1 &= \lambda_0\pi_0 + \mu_2\pi_2 \\
(\lambda_2 + \mu_2)\pi_2 &= \lambda_1\pi_1 + \mu_3\pi_3 \\
&\vdots \\
+ \quad (\lambda_{n-1} + \mu_{n-1})\pi_{n-1} &= \lambda_{n-2}\pi_{n-2} + \mu_n\pi_n \\
\hline
\lambda_{n-1}\pi_{n-1} &= \mu_n\pi_n
\end{aligned} \tag{2.5}$$

式 (2.5) の n を k に置き換えて両辺を μ_k で割ると式 (2.6) が導出される．

$$\begin{aligned}
\pi_k &= \frac{\lambda_{k-1}}{\mu_k}\pi_{k-1} \\
&= \frac{\lambda_{k-1}\lambda_{k-2}}{\mu_k\mu_{k-1}}\pi_{k-2} \\
&= \frac{\lambda_{k-1}\lambda_{k-2}\cdots\lambda_0}{\mu_k\mu_{k-1}\cdots\mu_1}\pi_0
\end{aligned} \tag{2.6}$$

式 (2.6) を用いてすべての状態確率を足すと 1 になることを表すと式 (2.7) が導出される．

$$\sum_{k=0}^{\infty}\pi_k = \pi_0\left(1 + \sum_{k=1}^{\infty}\frac{\lambda_{k-1}\lambda_{k-2}\cdots\lambda_0}{\mu_k\mu_{k-1}\cdots\mu_1}\right) = 1 \tag{2.7}$$

よって式 (2.7) より式 (2.8) が導出される．なお，この出生死滅過程が定常状態を持つためには π_0 が収束することが条件である．

$$\pi_0 = \left[1 + \sum_{k=1}^{\infty}\frac{\lambda_{k-1}\lambda_{k-2}\cdots\lambda_0}{\mu_k\mu_{k-1}\cdots\mu_1}\right]^{-1} \tag{2.8}$$

2.2.2 M/M/1 モデル

M/M/1 モデルは待ち行列モデルの中でも最も基本となるモデルである．M/M/1 モデルは客がポアソン過程に従って到着し，指数分布時間でシステムから退去する窓口が一つのモデルである．ここで M/M/1 モデルの特徴を以下に書く．

- 系内客数が i 人であることを状態 i であるとする
- 一番客数が少ない状態は状態 0 とする
- ポアソン過程はすべての客の到着時刻が互いに独立な到着過程である
- 観測する時刻を微小時間 Δt で考えればたかだか Δt の間に客は一人到着するか退去するかのどちらかである

このように M/M/1 モデルは出生死滅過程であらわすことができるため，出生死滅過程のときと同様に計算ができる．M/M/1 モデルの客の到着率 λ はポアソン過程に従うため，それぞれの客が独立に平均 λ に従って到

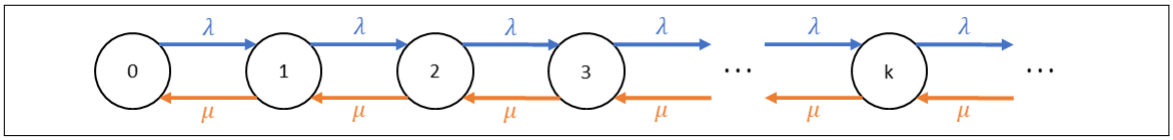

図 2.3　M/M/1 モデルの状態遷移図

着する．よって $\lambda_k = \lambda$ $(k = 0, 1, 2, \cdots)$ となる．一方で客の平均サービス時間も指数分布に従うため平均 $\dfrac{1}{\mu}$ に従って退去する．よって毎時刻確率 μ でシステムから退去するため到着率と同様に $\mu_k = \mu$ $(k = 0, 1, 2, \cdots)$ となる．以上の点をもとに出生死滅過程のときと同様に状態遷移図を描くと図 2.3 のようになる．ここで式 (2.5) に $\lambda_k = \lambda, \mu_k = \mu$ を代入し，両辺を μ で割ると式 (2.9) が導出される．

$$\begin{aligned} \pi_k = \frac{\lambda}{\mu}\pi_{k-1} &= \rho\pi_{k-1} \\ &= \rho^2\pi_{k-2} \\ &= \rho^k\pi_0 \end{aligned} \tag{2.9}$$

式 (2.9) を用いてすべての状態確率を足すと 1 になることを表すと式 (2.10) が導出される．

$$\sum_{k=0}^{\infty}\pi_k = \pi_0\left(1 + \sum_{k=1}^{\infty}\rho^k\right) = \pi_0\sum_{k=0}^{\infty}\rho^k = 1 \tag{2.10}$$

よって式 (2.10) より式 (2.11) が導出される．なお，この出生死滅過程が定常状態を持つためには π_0 が収束することが条件である．

$$\pi_0 = \left[\sum_{k=0}^{\infty}\rho^k\right]^{-1} \tag{2.11}$$

式 (2.11) 中の $\displaystyle\sum_{k=0}^{\infty}\rho^k$ は無限等比級数であり，$-1 < \rho < 1$ の区間において $\dfrac{1}{1-\rho}$ に収束することが知られている．よって M/M/1 モデルは $\rho < 1$ のときのみ定常状態確率を持つ．$\rho < 1$ のとき $\displaystyle\sum_{k=0}^{\infty}\rho^k = \frac{1}{\rho}$ を式 (2.11),(2.9) に代入すると式 (2.12),(2.13) が導出される．

$$\pi_0 = \left(\frac{1}{1-\rho}\right)^{-1} = 1 - \rho \tag{2.12}$$

$$\pi_k = (1-\rho)\rho^k \tag{2.13}$$

ここで平均待ち行列長 $E[L]$ は $k\pi_k$ の総和で表されるので無限等比級数に関する式 $\displaystyle\sum_{k=0}^{\infty}k\rho^k = \rho/(1-\rho)^2$ と式 (2.13) を用いて式 (2.14) が導出される．また，リトルの公式より平均待ち時間 $E[W]$ は式 (2.15) のように導出される．

$$E[L] = \sum_{k=0}^{\infty}k\pi_k = \frac{\rho}{1-\rho} \tag{2.14}$$

$$E[W] = \frac{E[L]}{\lambda} = \frac{1}{\mu(1-\rho)} \tag{2.15}$$

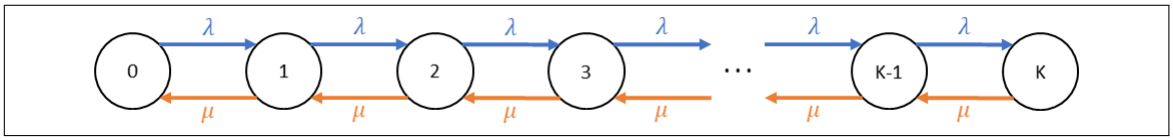

図 2.4　M/M/1/K モデルの状態遷移図

2.2.3　M/M/1/K モデル

M/M/1/K モデルは M/M/1 モデルの条件に加えて，待ち行列に形成可能な最大値（最大待ち行列長）が定められているという条件が加わったモデルである．M/M/1/K モデルがとりえる状態 i の最大値は K になっていて，状態 $K+1$ 以上の状態は存在しない．よって状態遷移図は図 2.4 のように描ける．これをもとに平衡方程式を書き下すと式 (2.16),(2.17),(2.18) が導出される．

$$\lambda\pi_0 = \mu\pi_1 \tag{2.16}$$

$$(\lambda+\mu)\pi_k = \lambda\pi_{k-1} + \mu\pi_{k+1} \quad (k = 1, 2, \ldots, K-1) \tag{2.17}$$

$$\mu\pi_K = \lambda\pi_{K-1} \tag{2.18}$$

ここで式 (2.16),(2.17) を $k = 0, 1, 2, \cdots, n-1 \leq K$ まで両辺を足し合わせると式 (2.19) が導出される．

$$\begin{aligned}
\lambda\pi_0 &= \mu\pi_1 \\
(\lambda+\mu)\pi_1 &= \lambda\pi_0 + \mu\pi_2 \\
(\lambda+\mu)\pi_2 &= \lambda\pi_1 + \mu\pi_3 \\
&\vdots \\
+ \quad (\lambda+\mu)\pi_{n-1} &= \lambda\pi_{n-2} + \mu\pi_n \\
\hline
\lambda\pi_{n-1} &= \mu\pi_n \quad (1 \leq n \leq K)
\end{aligned} \tag{2.19}$$

式 (2.19) の n を k に置き換えて両辺を μ で割ると式 (2.20) が導出される．

$$\begin{aligned}
\pi_k = \frac{\lambda}{\mu}\pi_{k-1} &= \rho\pi_{k-1} \\
&= \rho^2\pi_{k-2} \\
&= \rho^k\pi_0
\end{aligned} \tag{2.20}$$

式 (2.20) を用いてすべての状態確率を足すと 1 になることを表すと式 (2.21) が導出される．

$$\sum_{k=0}^{K}\pi_k = \pi_0\left(1 + \sum_{k=1}^{K}\rho^k\right) = \pi_0\sum_{k=0}^{K}\rho^k = 1 \tag{2.21}$$

$\pi_0\sum_{k=0}^{K}\rho^k$ は等比数列の総和になるので式 (2.22),(2.23) が導出される．

$$\pi_0\sum_{k=0}^{K}\rho^k = \begin{cases} \pi_0(K+1) & \rho = 1 \text{ のとき} \\ \dfrac{\pi_0(1-\rho^{K+1})}{1-\rho} & \rho \neq 1 \text{ のとき} \end{cases} \tag{2.22}$$

$$\pi_0 = \begin{cases} \dfrac{1}{K+1} & \rho = 1 \text{ のとき} \\ \dfrac{1-\rho}{1-\rho^{K+1}} & \rho \neq 1 \text{ のとき} \end{cases} \tag{2.23}$$

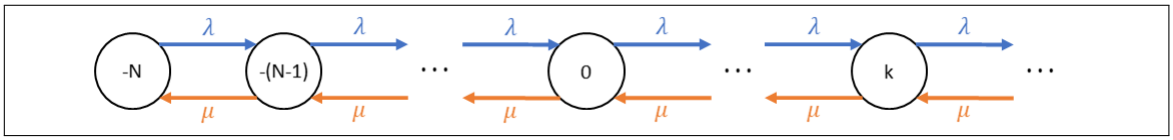

図 2.5　タクシー乗り場のモデルの状態遷移図 (M/M/1)

また式 (2.23) を式 (2.20) に代入することで式 (2.24) が導出される．

$$\pi_k = \begin{cases} \dfrac{1}{K+1} & \rho = 1 \text{ のとき} \\ \dfrac{(1-\rho)\rho^k}{1-\rho^{K+1}} & \rho \neq 1 \text{ のとき} \end{cases} \tag{2.24}$$

なお，式 (2.23) より M/M/1/K モデルは常に安定であることが分かる．ところで M/M/1/K モデルは状態 K のときに新たに客が到着しても待ち行列に加われないモデルなので客がこのモデルに到着したときにこの行列に加われない確率（呼損率）は π_K で表すことができる．

2.3　タクシー乗り場のモデルの解析

この節では相乗りを考慮しない一般的なタクシー乗り場のモデルについて述べる．より深くタクシー乗り場のモデルについて学ぶ場合はタクシースポット市場の差別化と社会的厚生 [8] や，Optimization and strategic behavior in a passenger-taxi service system[9] などを参照されたい．

2.3.1　M/M/1 モデルを用いた解析

乗客の到着過程はポアソン過程に従い，到着率は λ とする．また，タクシーの到着過程もポアソン分過程に従い，到着率は μ とする．乗客とタクシーがマッチングしたらシステムから即時に抜けるとすると，このシステムは乗客のみが列をなす，タクシーのみが列をなす，乗客とタクシーどちらもシステム内にいないという 3 パターンに分類することができる．よってこのモデルにおいて状態 π_i は以下のように定義する．

$$\pi_i \begin{cases} i \geq 1 & \text{乗客が } i \text{ 人並んでいてタクシーは並んでいない状態} \\ i = 0 & \text{乗客，タクシー共に並んでいない状態} \\ i \leq -1 & \text{乗客は並んでいなく，タクシーが } -i \text{ 台並んでいる状態} \end{cases}$$

また，タクシーの最大待ち行列長（駐車容量）を N とおく．N はタクシープール内で待機できるタクシーの台数を表しているため，N が自然数のときは N 台がタクシー乗り場内で乗客を待つことができ，$N = 0$ のときは乗客が待っていなければ即座にタクシープールを離れなければならないモデルを表している．また，タクシーが待機することができないようなモデルは実際のケースに落とし込むとタクシー乗り場のないただの道での乗客とタクシーのマッチングを表しているということになる [10].

以上の条件を踏まえると，このモデルの状態遷移図は図 2.5 のように描ける．図 2.5 と図 2.3 を比べると，このモデルは M/M/1 モデルの条件のうち最も小さい状態 i を 0 から $-N$ に変えただけであることが分かる．図 2.5 をもとに平衡方程式を書き下すと式 (2.25),(2.26) が導出される．

$$\lambda\pi_{-N} = \mu\pi_{-(N-1)} \tag{2.25}$$

$$(\lambda+\mu)\pi_k = \lambda\pi_{k-1} + \mu\pi_{k+1} \quad (k = -(N-1), -(N-2), \ldots) \tag{2.26}$$

ここで式 (2.25),(2.26) を $k = -N, -(N-1), -(N-2), \cdots, n-1$ まで両辺を足し合わせると式 (2.27) が導出される．

$$
\begin{aligned}
\lambda\pi_{-N} &= \mu\pi_{-(N-1)} \\
(\lambda+\mu)\pi_{-(N-1)} &= \lambda\pi_{-N} + \mu\pi_{-(N-2)} \\
(\lambda+\mu)\pi_{-(N-2)} &= \lambda\pi_{-(N-1)} + \mu\pi_{-(N-3)} \\
&\vdots \\
+ \quad (\lambda+\mu)\pi_{n-1} &= \lambda\pi_{n-2} + \mu\pi_n
\end{aligned}
$$

$$
\lambda\pi_{n-1} = \mu\pi_n \ \ (n = -(N-1), -(N-2), \ldots) \tag{2.27}
$$

式 (2.27) の n を k に置き換えて両辺を μ で割ると式 (2.28) が導出される．

$$
\begin{aligned}
\pi_k = \frac{\lambda}{\mu}\pi_{k-1} &= \rho\pi_{k-1} \\
&= \rho^2\pi_{k-2} \\
&= \rho^k\pi_0 \\
&= \rho^{k+N}\pi_{-N}
\end{aligned} \tag{2.28}
$$

式 (2.28) を用いてすべての状態確率を足すと 1 になることを表すと式 (2.29) が導出される．

$$
\sum_{k=-N}^{\infty}\pi_k = \pi_{-N}\sum_{k=-N}^{\infty}\rho^{k+N} = \pi_{-N}\sum_{k=0}^{\infty}\rho^k = 1 \tag{2.29}
$$

よって式 (2.29) より式 (2.30) が導出される．なお，この出生死滅過程が定常状態を持つためには π_{-N} が収束することが条件である．

$$
\pi_{-N} = \left[\sum_{k=0}^{\infty}\rho^k\right]^{-1} \tag{2.30}
$$

式 (2.30) 中の $\displaystyle\sum_{k=0}^{\infty}\rho^k$ は無限等比級数であり，$-1 < \rho < 1$ の区間において $\dfrac{1}{1-\rho}$ に収束することが知られている．よって M/M/1 モデルは $\rho < 1$ のときのみ定常状態確率を持つ．$\rho < 1$ のとき $\displaystyle\sum_{k=0}^{\infty}\rho^k = \frac{1}{\rho}$ を式 (2.30),(2.28) に代入すると式 (2.31),(2.32) が導出される．

$$
\pi_{-N} = \left(\frac{1}{1-\rho}\right)^{-1} = 1-\rho \tag{2.31}
$$

$$
\pi_k = (1-\rho)\rho^{k+N} \tag{2.32}
$$

定常状態確率が求まったため，乗客，タクシーそれぞれの平均待ち行列長，平均待ち時間が導出できる．乗客の平均待ち行列長を $E[L_1]$，タクシーの平均待ち行列長を $E[L_2]$ とすると式 (2.33), (2.34) のように導出される．

$$
\begin{aligned}
E[L_1] &= \sum_{k=0}^{\infty}k(1-\rho)\rho^{k+N} \\
&= \rho^N\sum_{k=0}^{\infty}k(1-\rho)\rho^k \\
&= \frac{\rho^{N+1}}{1-\rho}
\end{aligned} \tag{2.33}
$$

$$
\begin{aligned}
E[L_2] &= \sum_{k=-N}^{0} (-k)(1-\rho)\rho^{k+N} \\
&= \rho^N(1-\rho) \sum_{k=-N}^{0} (-k)\rho^k \\
&= \rho^N(1-\rho) \sum_{k=0}^{N} k\left(\frac{1}{\rho}\right)^k \\
&= \rho^N(1-\rho)\frac{\rho}{1-\rho}\left(\frac{N}{\rho^{N+1}} - \frac{\frac{1}{\rho^N}-1}{1-\rho}\right) \qquad (*) \\
&= N - \frac{\rho}{1-\rho}(1-\rho^N) \qquad (2.34)
\end{aligned}
$$

また，リトルの公式より平均待ち時間 $E[W]$ は式 (2.35), 式 (2.36) のように導出される．

$$E[W_1] = \frac{E[L_1]}{\lambda} \tag{2.35}$$

$$E[W_2] = \frac{E[L_2]}{\lambda} \tag{2.36}$$

なお，式 $(*)$ の式変換は以下に記しておく．

$$
\begin{aligned}
\sum_{k=0}^{N} k\left(\frac{1}{\rho}\right)^k &= \frac{1}{\rho} + \frac{2}{\rho^2} + \frac{3}{\rho^3} + \cdots + \frac{N-1}{\rho^{N-1}} + \frac{N}{\rho^N} \\
\frac{1}{\rho}\sum_{k=0}^{N} k\left(\frac{1}{\rho}\right)^k &= \frac{1}{\rho^2} + \frac{2}{\rho^3} + \cdots + \frac{N-2}{\rho^{N-1}} + \frac{N-1}{\rho^N} + \frac{N}{\rho^{N+1}}
\end{aligned}
$$

$$
\begin{aligned}
\left(1-\frac{1}{\rho}\right)\sum_{k=0}^{N} k\left(\frac{1}{\rho}\right)^k &= \frac{1}{\rho} + \frac{1}{\rho^2} + \frac{1}{\rho^3} + \cdots + \frac{1}{\rho^N} - \frac{N}{\rho^N+1} \\
&= \sum_{l=1}^{N}\left(\frac{1}{\rho}\right)^l - \frac{N}{\rho^{N+1}} \\
&= \frac{\frac{1}{\rho}\left(1-\frac{1}{\rho^N}\right)}{1-\frac{1}{\rho}} - \frac{N}{\rho^{N+1}} = -\frac{N}{\rho^{N+1}} + \frac{\frac{1}{\rho^N}-1}{1-\rho}
\end{aligned}
$$

よって

$$\sum_{k=0}^{N} k\left(\frac{1}{\rho}\right)^k = \frac{1}{\left(1-\frac{1}{\rho}\right)}\left(-\frac{N}{\rho^{N+1}} + \frac{\frac{1}{\rho^N}-1}{1-\rho}\right) = \frac{\rho}{1-\rho}\left(\frac{N}{\rho^{N+1}} - \frac{\frac{1}{\rho^N}-1}{1-\rho}\right)$$

なお，タクシーの呼損率は π_{-N} で表される．

2.3.2 M/M/1/K モデルを用いた解析

M/M/1/K モデルを用いた解析では M/M/1 モデル用いた時の条件に加えて，乗客の最大待ち行列長 K を導入する．すると状態遷移図は図 2.6 のように描ける．これをもとに平衡方程式を書き下すと式 (2.37),(2.38),(2.39)

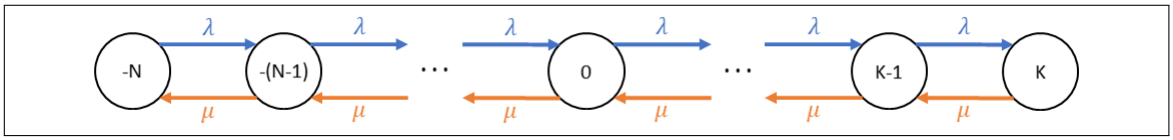

図 2.6　タクシー乗り場のモデルの状態遷移図 (M/M/1/K)

$$\lambda\pi_{-N} = \mu\pi_{-(N-1)} \tag{2.37}$$
$$(\lambda+\mu)\pi_k = \lambda\pi_{k-1} + \mu\pi_{k+1} \quad (k = -(N-1), -(N-2), \ldots, K-1) \tag{2.38}$$
$$\mu\pi_K = \lambda\pi_{K-1} \tag{2.39}$$

ここで式 (2.37), (2.38) を $k = -N, -(N-1), -(N-2), \cdots, n-1 \leq K$ まで両辺を足し合わせると式 (2.40) が導出される．

$$\begin{aligned}
\lambda\pi_{-N} &= \mu\pi_{-(N-1)} \\
(\lambda+\mu)\pi_{-(N-1)} &= \lambda\pi_{-N} + \mu\pi_{-(N-2)} \\
(\lambda+\mu)\pi_{-(N-2)} &= \lambda\pi_{-(N-1)} + \mu\pi_{-(N-3)} \\
&\vdots \\
+ \quad (\lambda+\mu)\pi_{n-1} &= \lambda\pi_{n-2} + \mu\pi_n \\
\hline
\lambda\pi_{n-1} &= \mu\pi_n \quad (k = -(N-1), -(N-2), \ldots, K)
\end{aligned} \tag{2.40}$$

式 (2.19) の n を k に置き換えて両辺を μ で割ると式 (2.41) が導出される．

$$\begin{aligned}
\pi_k = \frac{\lambda}{\mu}\pi_{k-1} &= \rho\pi_{k-1} \\
&= \rho^2\pi_{k-2} \\
&= \rho^k\pi_0 \\
&= \rho^{k+N}\pi_{-N}
\end{aligned} \tag{2.41}$$

式 (2.41) を用いてすべての状態確率を足すと 1 になることを表すと式 (2.42) が導出される．

$$\sum_{k=-N}^{K} \pi_k = \pi_{-}N \sum_{k=-N}^{K} \rho^{k+N} = 1 \tag{2.42}$$

よって式 (2.42) より式 (2.43) が導出される．なお，この出生死滅過程が定常状態を持つためには π_{-N} が収束することが条件である．

$$\pi_{-}N = \left[\sum_{k=-N}^{K} \rho^{k+N}\right]^{-1} \tag{2.43}$$

ここで $\displaystyle\sum_{k=-N}^{K} \rho^{k+N} = 1$ を計算すると，$\rho = 1$ のとき (2.44) が，$\rho \neq 1$ のとき式 (2.45) が導出される．

$$\sum_{k=-N}^{K} \rho^{k+N} = \sum_{k=-N}^{K} = \sum_{k=0}^{K+N} = K + N + 1 \tag{2.44}$$

$$
\begin{aligned}
\sum_{k=-N}^{K} \rho^{k+N} &= \sum_{k=-N}^{0} \rho^{k+N} + \sum_{k=0}^{K} \rho^{k+N} - \rho^{N} \\
&= \sum_{k=0}^{N} \rho^{k} + \sum_{k=0}^{K} \rho^{k+N} - \rho^{N} \\
&= \frac{1-\rho^{N+1}}{1-\rho} + \frac{1-\rho^{K+1}}{1-\rho} - \rho^{N} \\
&= \frac{2-\rho^{N+1}-\rho^{K+1}-(1-\rho)\rho^{N}}{1-\rho} \\
&= \frac{2-\rho^{K+1}-\rho^{N}}{1-\rho}
\end{aligned} \tag{2.45}
$$

よって式 (2.44),(2.45) を式 (2.43) に代入すると式 (2.46) が，式 (2.41) に代入すると式 (2.47) が導出される．

$$
\pi_{-N} = \begin{cases} \dfrac{1}{N+K+1} & \rho = 1 \text{ のとき} \\ \dfrac{1-\rho}{2-\rho^{K+1}-\rho^{N}} & \rho \neq 1 \text{ のとき} \end{cases} \tag{2.46}
$$

$$
\pi_{k} = \begin{cases} \dfrac{1}{K+N+1} & \rho = 1 \text{ のとき} \\ \dfrac{(1-\rho)\rho^{k+N}}{2-\rho^{K+1}-\rho^{N}} & \rho \neq 1 \text{ のとき} \end{cases} \tag{2.47}
$$

なお，式 (2.46) よりこのモデルは常に定常状態を持ち，乗客の呼損率は π_K，タクシーの呼損率は π_{-N} で表される．

2.4 準出生死滅過程

準出生死滅過程とは出生死滅過程を発展させた過程である．ここで準出生死滅過程を説明するにあたっては次のようなモデルを用いる．

- 客の到着過程はポアソン過程に従い，到着率は λ_1 とする
- 客は窓口で 0 と 1 の 2 つのサービスを受ける
- 客は必ず $0 \to 1$ の順にサービスを受ける
- 次の客は前の客がサービス 1 を終了したときに窓口に入れる
- サービス 1 から 2 に移行する到着過程はポアソン過程に従い，到着率は λ_2 とする
- サービス 1 を開始してから退去するまでの時間分布は指数分布に従い，平均 $1/\mu$ とする
- 窓口は 1 つとする
- 待ち行列は無限に並べるとする

以上の条件を元にすると，状態遷移図は図 2.7 のように描ける．これをもとに平衡方程式を書き下すと式 (2.48),(2.49),(2.50),(2.51) が導出される．

$$
-\lambda_1 \pi_{0,0} + \mu \pi_{1,1} = 0 \tag{2.48}
$$

$$
\lambda_2 \pi_{1,0} - (\lambda_1 + \mu)\pi_{1,1} = 0 \tag{2.49}
$$

$$
\lambda_1 \pi_{i-1,0} - (\lambda_1 + \lambda_2)\pi_{i,0} + \mu \pi_{i+1,1} = 0 \quad (i = 1, 2, \ldots) \tag{2.50}
$$

$$
\lambda_1 \pi_{i-1,1} + \lambda_2 \pi_{i,0} - (\lambda_1 + \mu)\pi_{i,1} = 0 \quad (i = 2, 3, \ldots) \tag{2.51}
$$

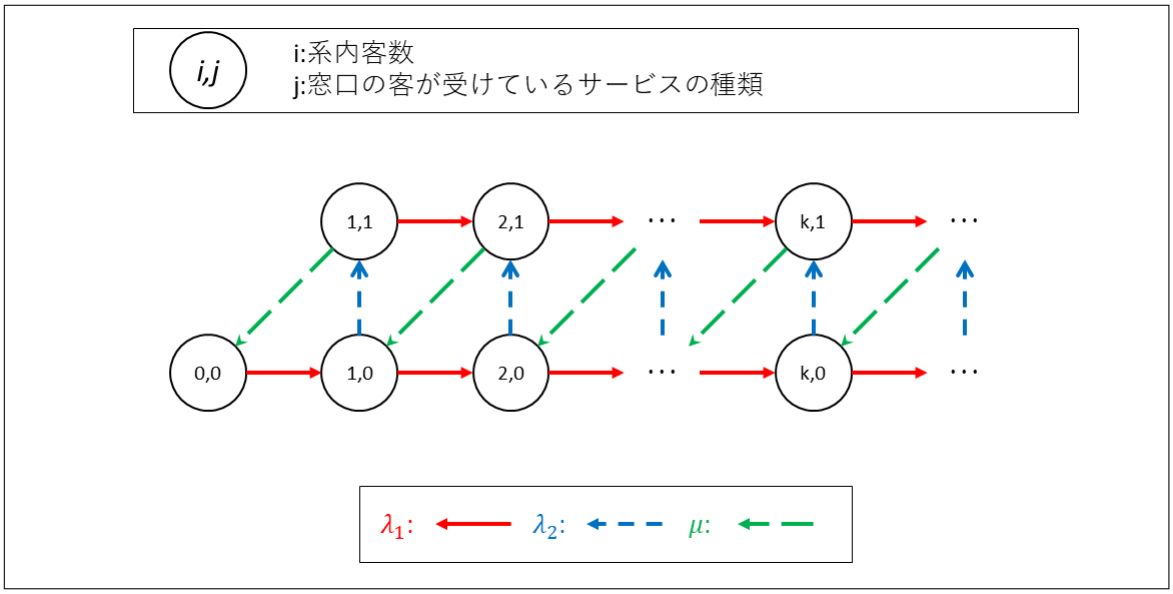

図 2.7　状態遷移図

以上の式を行列式を用いて書くと式 (2.52) が導出される．ただし，$\boldsymbol{\pi}, \boldsymbol{Q}, \boldsymbol{O}$ は以下のように定義する．

$$\boldsymbol{\pi}\boldsymbol{Q} = \boldsymbol{O} \tag{2.52}$$

$$\boldsymbol{\pi} = (\pi_{0,0}, \pi_{1,0}, \pi_{1,0}, \pi_{2,0}, \pi_{2,1}, \pi_{3,0}, \pi_{3,1}, \ldots)$$

$$\boldsymbol{Q} = \begin{pmatrix} -\lambda_1 & \lambda_1 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & -(\lambda_1+\lambda_2) & \lambda_2 & \lambda_1 & 0 & 0 & 0 & \cdots \\ \mu & 0 & -(\lambda_1+\mu) & 0 & \lambda_1 & 0 & 0 & \cdots \\ 0 & 0 & 0 & -(\lambda_1+\lambda_2) & \lambda_2 & \lambda_1 & 0 & \cdots \\ 0 & \mu & 0 & 0 & -(\lambda_1+\mu) & 0 & \lambda_1 & \cdots \\ 0 & 0 & 0 & 0 & 0 & -(\lambda_1+\lambda_2) & \lambda_2 & \cdots \\ 0 & 0 & 0 & \mu & 0 & 0 & -(\lambda_1+\mu) & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

$$\boldsymbol{O} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \end{pmatrix}$$

ここで以下のように表現すると $\boldsymbol{\pi}, \boldsymbol{Q}$ はより簡潔に書くことができる．

$$\pi_{0,0} = \boldsymbol{\pi}_0 \ , \ (\pi_{i,0}, \pi_{i,1}) = \boldsymbol{\pi}_i \quad (i = 1, 2, \ldots) \ , \ \boldsymbol{\pi} = (\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots)$$

$$\boldsymbol{Q}_{+1} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_1 \end{pmatrix} \ , \ \boldsymbol{Q}_0 = \begin{pmatrix} -(\lambda_1+\lambda_2) & \lambda_2 \\ 0 & -(\lambda_1+\mu) \end{pmatrix} \ , \ \boldsymbol{Q}_{-1} = \begin{pmatrix} 0 & 0 \\ \mu & 0 \end{pmatrix}$$

$$\boldsymbol{B}_{+1} = \begin{pmatrix} \lambda_1 & 0 \end{pmatrix} \ , \ \boldsymbol{B}_0 = -\lambda_1 \ , \ \boldsymbol{B}_{+1} = \begin{pmatrix} 0 \\ \mu \end{pmatrix}$$

$$\boldsymbol{\pi Q} = (\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots) \begin{pmatrix} \boldsymbol{B}_0 & \boldsymbol{B}_{+1} & \boldsymbol{O} & \boldsymbol{O} & \cdots \\ \boldsymbol{B}_{-1} & \boldsymbol{Q}_0 & \boldsymbol{Q}_{+1} & \boldsymbol{O} & \cdots \\ \boldsymbol{O} & \boldsymbol{Q}_{-1} & \boldsymbol{Q}_0 & \boldsymbol{Q}_{+1} & \cdots \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{Q}_{-1} & \boldsymbol{Q}_0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} = \boldsymbol{O} \tag{2.53}$$

式 2.53 のような式で表すことができるようなモデルを準出生死滅過程と呼ぶ．ところで図 2.7 の状態のうち，鉛直方向に増減する数値 i をレベル，垂直方向に増減する数値 j を相と呼び，$\boldsymbol{Q}$ に内包されている行列式の右下についている数字は +1,0,-1 の順にそれぞれレベルが 1 増える，変わらない，1 減るような遷移を表している．各定常状態確率の導出方法は M/M/1 を越えて―準出生死滅過程への招待―[11] に任せて，ここでは準出生死滅過程の安定条件について述べる．

準出生死滅過程では出生死滅過程の無限バッファ時の安定条件の式 2.54 にあたるものとしてドリフト条件の判別式 2.55 が存在する．

$$\lambda < \mu \tag{2.54}$$

$$\tilde{\boldsymbol{\pi}} \boldsymbol{Q}_{+1} \mathbf{1}^T < \tilde{\boldsymbol{\pi}} \boldsymbol{Q}_{-1} \mathbf{1}^T \tag{2.55}$$

$$\text{ただし} \quad \tilde{\boldsymbol{\pi}} \tilde{\boldsymbol{Q}} = \boldsymbol{O} \ , \ \tilde{\boldsymbol{\pi}} \mathbf{1}^T = 1 \ , \ \tilde{\boldsymbol{Q}} = \boldsymbol{Q}_{+1} + \boldsymbol{Q}_0 + \boldsymbol{Q}_{-1}$$

上で触れたように，$\boldsymbol{Q_{+1}}$ はレベルが 1 増える遷移を，$\boldsymbol{Q_{-1}}$ はレベルが 1 減る遷移を表しているのでイメージとしては出生死滅過程の λ と μ を準出生死滅過程用に置き換えたと考えるとわかりやすい．

第 3 章

本論

第 3 章では第 2 章で述べた既存のタクシー乗り場の待ち行列モデルを拡張して相乗りを考慮したタクシー乗り場の待ち行列モデルを構築していく．

3.1　条件設定

条件設定は以下のように行う．

- 相乗りを許さない乗客（タイプ 1）と相乗りを許す乗客（タイプ 2）を設定する
- それぞれ到着過程はポアソン過程に従い，到着率は λ_1, λ_2 とする
- タクシーの到着過程もポアソン過程に従い，到着率は μ とする
- 窓口となるタクシー乗り場は 1 つとする
- タクシーの最大待ち行列長（駐車容量）を M，乗客の最大待ち行列長を N とする

3.2　相乗りの表現方法

本研究ではまだペアを組んでいないタイプ 2 の乗客が待ち行列にいるときに別のタイプ 2 の乗客が到着した場合にその 2 人でペアを組み相乗りをすると定義する．その際，後から来たタイプ 2 の乗客がそのまま前からいたタイプ 2 の乗客のところに一緒に並ぶ．なお，タイプ 2 の乗客が到着したときに待ち行列にまだペアを組んでいないタイプ 2 の乗客がいない場合は待ち行列の一番後ろに加わる．また 1 つのペアは 2 人までしか相乗りできないとするため一度ペアリングをしたペアに新たにタイプ 2 の乗客が加わることはない．上記の一連の動きは図 3.1 を参照されたい．

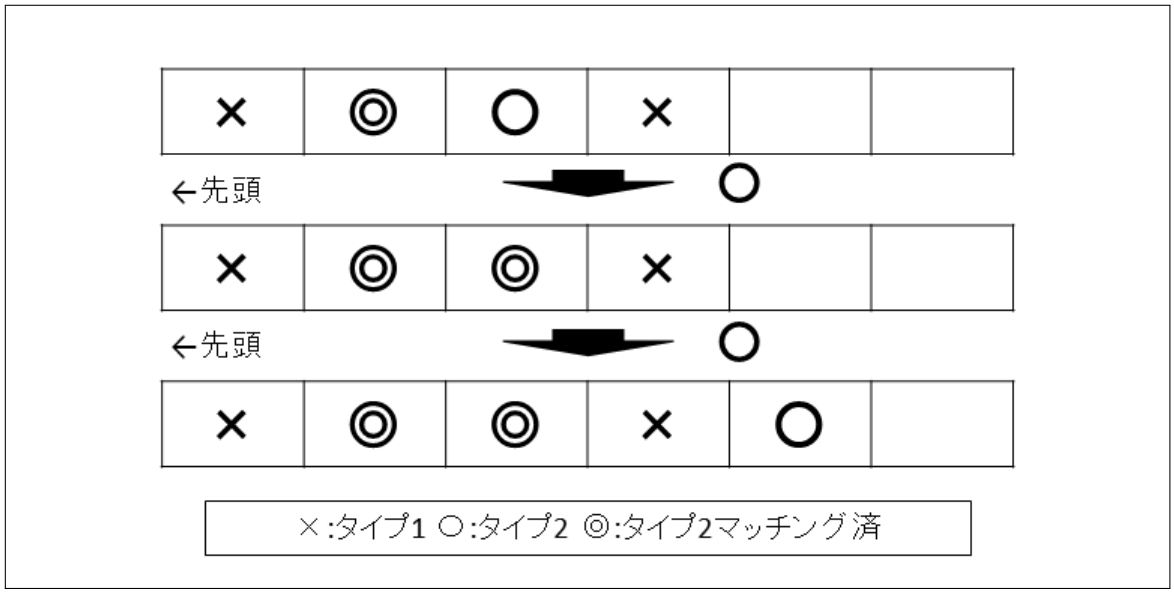

図 3.1　相乗りペアリング模式図

3.3　M/M/1 モデルを用いた解析

この節では M/M/1 モデルを発展させた形，すなわち $N = \infty$ の場合について解析を行う．ただし，ここでは簡略化するため $M = 0$ とする．3.1 節の条件を考慮して状態遷移図を描くと図 3.2 のようになる．ここで系内客数が i，まだペアを組んでいないタイプ 2 の乗客の並んでいる順番が j である状態を $\pi_{i,j}$ とすると図 3.2 をもとに平衡方程式を書き下すと式 (3.1),(3.2),(3.3),(3.4) が導出される．

$i = 0$ のとき

$$-(\lambda_1 + \lambda_2)\pi_{0,0} + \mu(\pi_{1,0} + \pi_{1,1}) = 0 \tag{3.1}$$

$i = 1, 2, \ldots$ のとき

$$-(\lambda_1 + \lambda_2 + \mu)\pi_{i,0} + \lambda_1\pi_{i-1,0} + \lambda_2 \sum_{k=1}^{i} \pi_{i,k} + \mu(\pi_{i+1,0} + \pi_{i+1,1}) = 0 \tag{3.2}$$

$$-(\lambda_1 + \lambda_2 + \mu)\pi_{i,j} + \lambda_1\pi_{i-1,j} + \mu\pi_{i+1,j+1} = 0 \quad (0 < j < i) \tag{3.3}$$

$$-(\lambda_1 + \lambda_2 + \mu)\pi_{i,i} + \lambda_2\pi_{i-1,0} + \mu\pi_{i+1,i+1} = 0 \tag{3.4}$$

更に式 (3.5) を定義することで式 (3.1)，(3.2)，(3.3)，(3.4) は式 (3.6)，(3.7) のように簡潔に書き下せる．

$$\pi(i) \coloneqq \sum_{j=0}^{i} \pi_{i,j} \tag{3.5}$$

$$(\lambda_1 + \lambda_2)\pi(0) = \mu\pi(1) \tag{3.6}$$

$$(\lambda_1 + \mu)\pi(i) + \lambda_2\pi_{i,0} = \lambda_1\pi(i-1) + \lambda_2\pi_{i-1,0} + \mu\pi(i+1) \quad (i = 1, 2, \ldots) \tag{3.7}$$

ただし，現状でこの方程式の解法は見つかっていないためシミュレーションを用いることでモデルの解析を行っている．シミュレーションについては 4.1 節を参照されたい．

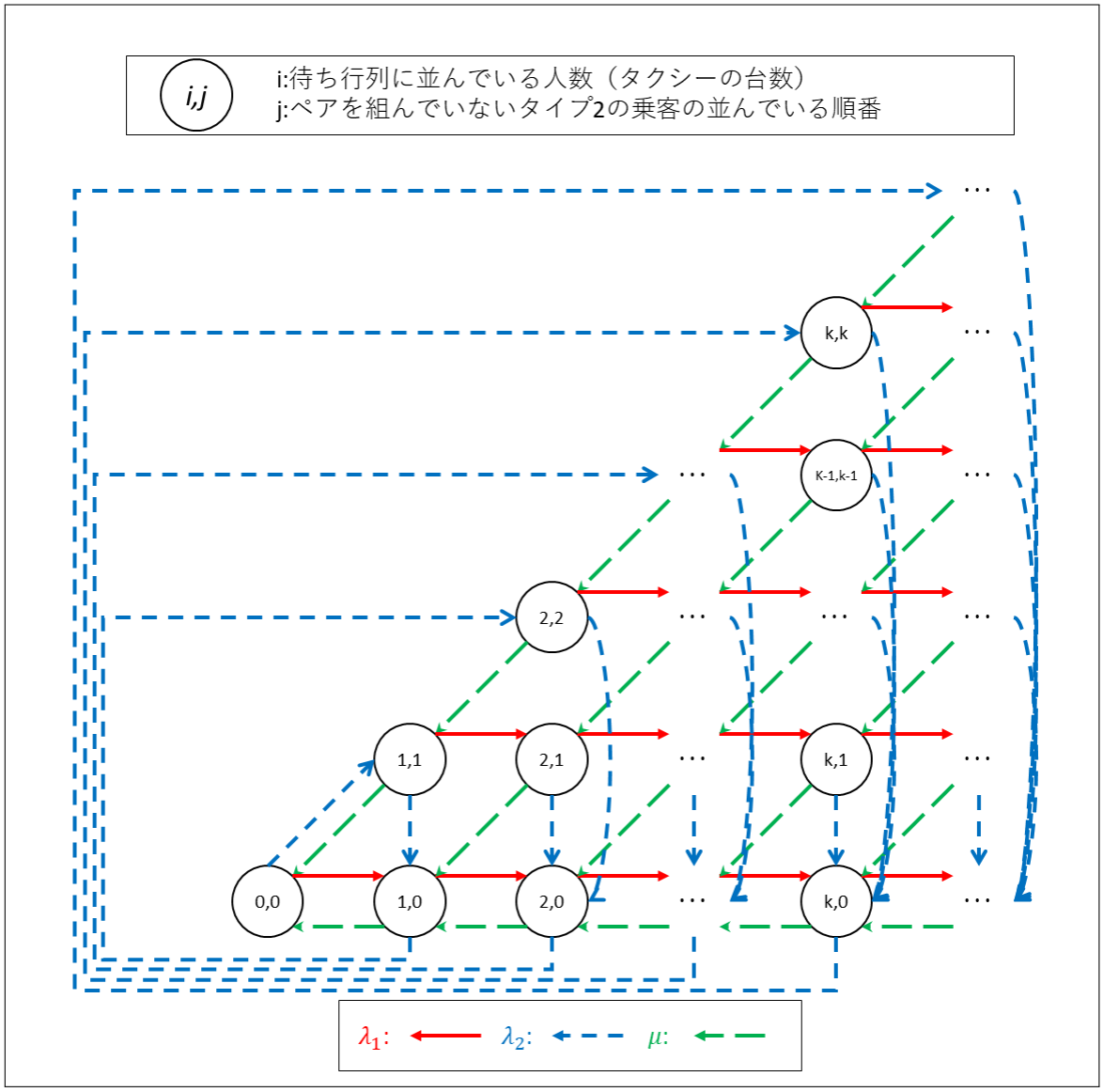

図 3.2　状態遷移図 ($N = \infty$ のとき)

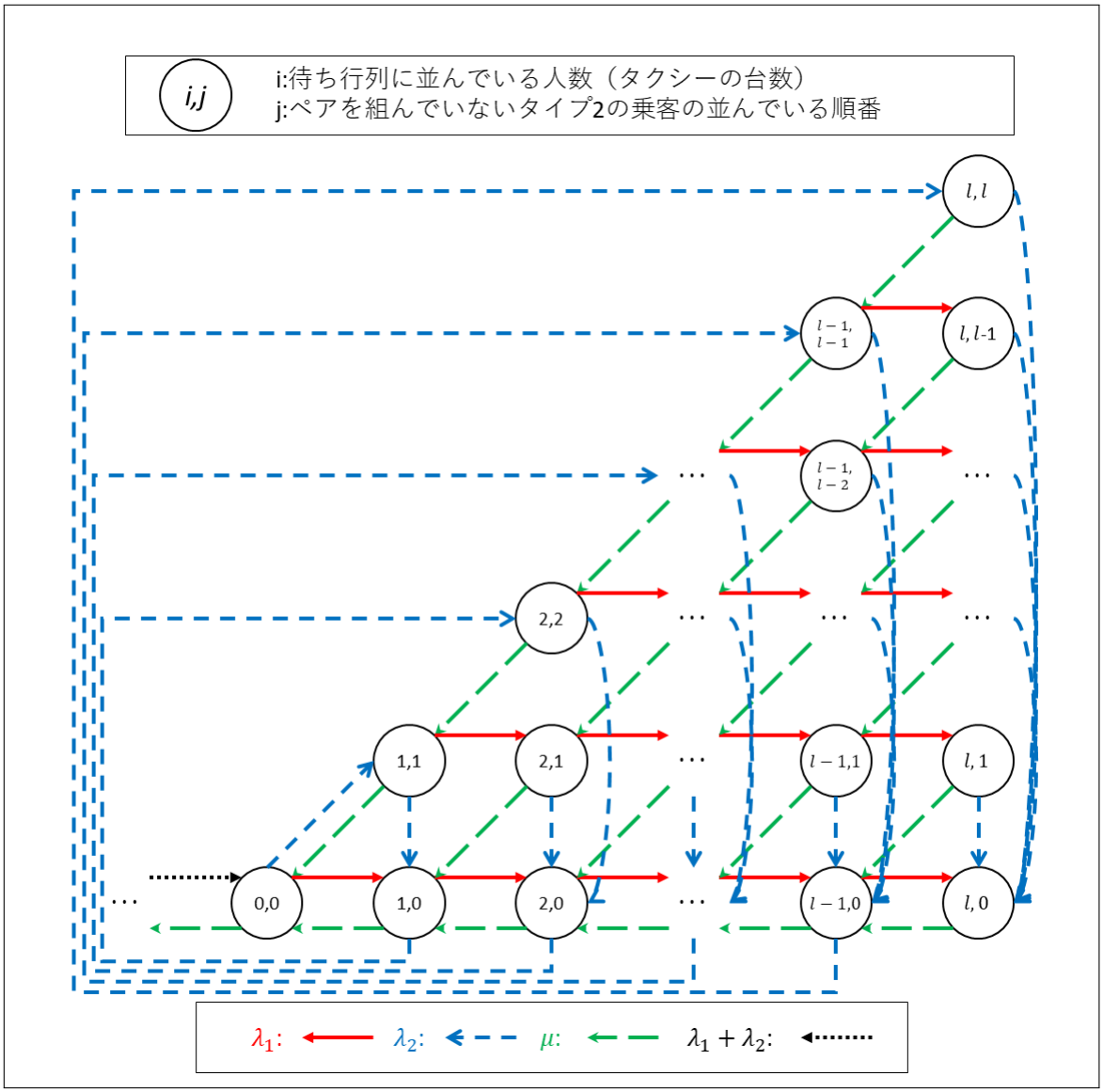

図 3.3　状態遷移図 ($N = l$，$i \geq 0$)

3.4　M/M/1/K モデルを用いた解析

この節では M/M/1/K モデルを発展させた形，すなわち N が有限の値 l である場合について解析を行う．このモデルでは前節とは異なり，M は自由な値をとるとする．3.1 節の条件を考慮して状態遷移図を描くと図 3.3，3.4 のようになる．このモデルではまだペアを組んでいないタイプ 2 の乗客の並んでいる順番が j であるとき，i の値によって以下のように定義すると図 3.4 をもとに平衡方程式を書き下すと式 (3.8)～(3.16) が導出される．

$$\pi_{i,j}\begin{cases} i \geq 1 & \text{乗客が } i \text{ 人並んでいてタクシーは並んでいない状態} \\ i = 0 & \text{乗客，タクシー共に並んでいない状態} \\ i \leq -1 & \text{乗客は並んでいなく，タクシーが } -i \text{ 台並んでいる状態} \end{cases}$$

$i = -M$ のとき

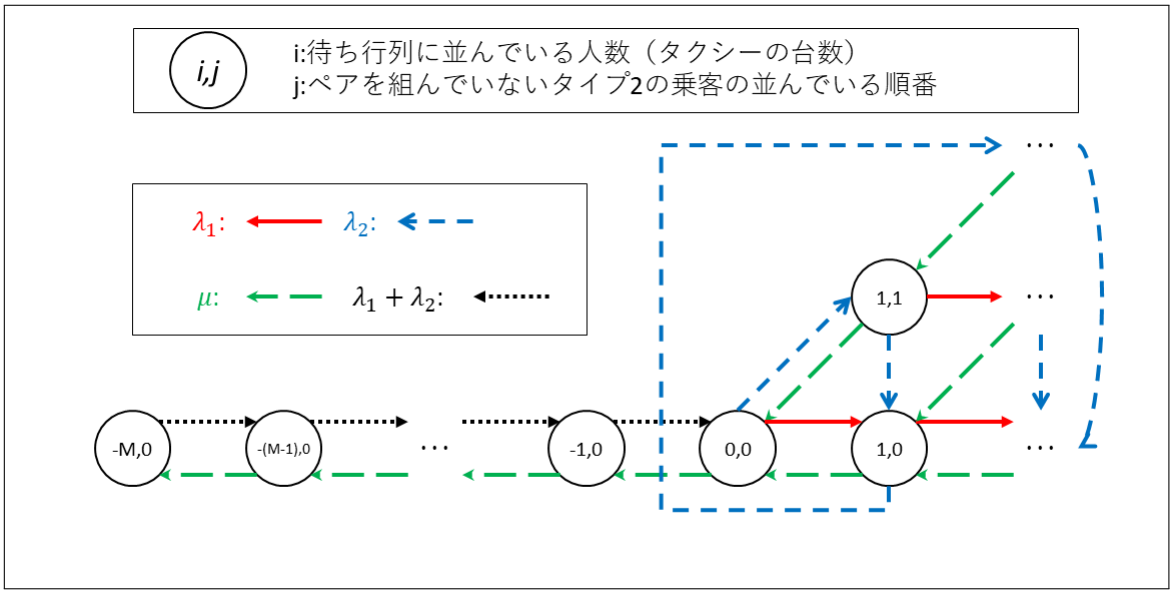

図 3.4　状態遷移図 ($N = l,\ i \leq 0$)

$$-(\lambda_1 + \lambda_2)\pi_{-M,0} + \mu\pi_{-(M-1),0} = 0 \tag{3.8}$$

$i = -(M-1), -(M-2), \ldots, -1$ のとき

$$-(\lambda_1 + \lambda_2 + \mu)\pi_{i,0} + (\lambda_1 + \lambda_2)\pi_{-(i+1),0} + \mu\pi_{-(i-1),0} = 0 \tag{3.9}$$

$i = 0$ のとき

$$-(\lambda_1 + \lambda_2 + \mu)\pi_{0,0} + (\lambda_1 + \lambda_2)\pi_{-1,0} + \mu(\pi_{1,0} + \pi_{1,1}) = 0 \tag{3.10}$$

$i = 1, 2, \ldots, l-1$ のとき

$$-(\lambda_1 + \lambda_2 + \mu)\pi_{i,0} + \lambda_1\pi_{i-1,0} + \lambda_2 \sum_{k=1}^{i} \pi_{i,k} + \mu(\pi_{i+1,0} + \pi_{i+1,1}) = 0 \tag{3.11}$$

$$-(\lambda_1 + \lambda_2 + \mu)\pi_{i,j} + \lambda_1\pi_{i-1,j} + \mu\pi_{i+1,j+1} = 0 \quad (0 < j < i) \tag{3.12}$$

$$-(\lambda_1 + \lambda_2 + \mu)\pi_{i,i} + \lambda_2\pi_{i-1,0} + \mu\pi_{i+1,i+1} = 0 \tag{3.13}$$

$i = -l$ のとき

$$-\mu\pi_{l,0} + \lambda_1\pi_{l-1,0} + \lambda_2 \sum_{k=1}^{l} \pi_{l,k} = 0 \tag{3.14}$$

$$-(\lambda_2 + \mu)\pi_{l,j} + \lambda_1\pi_{l-1,j} = 0 \quad (0 < j < l) \tag{3.15}$$

$$-(\lambda_2 + \mu)\pi_{l,l} + \lambda_2\pi_{l-1,0} = 0 \tag{3.16}$$

これらの式に確率の総和が 1 であることを表した式 (3.17) を加えた連立方程式を解くことで直接 $\pi_{i,j}$ を算出することができる．本論文ではこれらの連立方程式をプログラムで解析して各種数値を算出している．プログラムは 4.2 を参照されたい．

$$\sum_{i=-M}^{-1} \pi_{i,0} + \sum_{i=0}^{l} \sum_{j=0}^{i} \pi_{k,j} = 1 \tag{3.17}$$

3.5 各種数値の算出

本論文では乗客の最大待ち行列長が有限の状態で解析するため数値計算も乗客の最大待ち行列長が有限であることを前提に行う．π_i を式 (3.18) のように定義すると，乗客の平均待ち行列長 L_p とタクシーの平均待ち行列長 L_t は (3.19),(3.20) のように導出される．

$$\pi(i) \coloneqq \sum_{j=0}^{i} \pi_{i,j} \tag{3.18}$$

$$L_p = \sum_{i=0}^{l} i\pi_i \tag{3.19}$$

$$L_t = \sum_{i=-M}^{0} (-i)\pi_{i,0} \tag{3.20}$$

また，タイプ 2 の乗客がシステムに到着したときにまだペアを組んでいないタイプ 2 の乗客がいる確率（ペア成立率）π_{λ_2} は式 (3.21) のように導出される．

$$\pi_{\lambda_2} = 1 - \left(\sum_{i=0}^{l} \pi_{i,0} + \sum_{i=-M}^{1} \pi_{i,0} \right) \tag{3.21}$$

平均待ち時間についてはひとつひとつ考えていく必要がある．まず，最も簡単に計算できるタイプ 1 の乗客の平均待ち時間について考える．待ち行列の先頭に並んだ乗客は平均 $1/\mu$ でサービスを受ける．タイプ 1 の乗客はシステムに到着すると待ち行列の最後尾に並ぶためその乗客が到着した時点での待ち行列長 +1 人分 (自分のサービス時間分）待つため，タイプ 1 の乗客の平均待ち時間 W_1 は式 (3.22) のように導出される．

$$W_1 = \sum_{i=0}^{l-1} \frac{i+1}{\mu} \pi_i \tag{3.22}$$

なお $i \leq -1$ においてはタクシーが待機しているため待ち時間がないこと，$i = l$ のときはこれ以上待ち行列に乗客が並ぶことができず呼損するため上記の式内では反映されていないことに留意されたい．

次にタクシーの平均待ち時間について考える．待ち行列の先頭に並んだタクシーは平均 $1/(\lambda_1 + \lambda_2)$ で乗客とマッチングする．タクシーはシステムに到着すると待ち行列の最後尾に並ぶためそのタクシーが到着した時点での待ち行列長 +1 台分 (自分の乗客とのマッチングまでの時間分）待つため，タクシーの平均待ち時間 W_t は式 (3.23) のように導出される．

$$W_t = \sum_{i=-M}^{0} \frac{-i+1}{\lambda_1 + \lambda_2} \pi_{i,0} \tag{3.23}$$

タクシーの平均待ち時間においても $i \geq 1$ においては乗客が待機しているため待ち時間がないこと，$i = -M$ のときはこれ以上待ち行列に乗客が並ぶことができず呼損するため上記の式内では反映されていないことに留意されたい．

タイプ 2 の乗客の平均待ち時間は 2 通りに分けて計算する必要がある．

待ち行列内にまだペアを組んでいないタイプ 2 の乗客がいない場合

この場合はタイプ 1 の乗客と同様に平均待ち時間を計算できる．この場合のタイプ 2 の乗客の平均待ち時間を W_{2A} とすると式 (3.24) のように導出される．

$$W_{2A} = \sum_{i=0}^{l-1} \frac{i+1}{\mu} \pi_{i,0} \tag{3.24}$$

なおタイプ 1 の乗客の場合は j がどのような値の時もまとめて計算したが，タイプ 2 の乗客の場合は $j \neq 0$ の時は待ち行列内にまだペアを組んでいないタイプ 2 の乗客がいるため計算式に入れない点に留意されたい．

待ち行列内にまだペアを組んでいないタイプ 2 の乗客がいる場合

この場合はシステムに到着した乗客は j 番目に並ぶため，この場合のタイプ 2 の乗客の平均待ち時間を W_{2B} とすると式 (3.25) のように導出される．

$$W_{2B} = \sum_{i=1}^{l} \sum_{j=1}^{i} \frac{j}{\mu} \pi_{i,j} \tag{3.25}$$

待ち行列内にまだペアを組んでいないタイプ 2 の乗客がいる場合では待ち行列に並んでいる乗客が l 人であったとしても待ち行列に加わることが可能なので $i = l$ まで含めて計算していることに留意されたい．

よってタイプ 2 の乗客の平均待ち時間は式 (3.24),(3.25) を足し合わせた式 (3.26) のように導出される．

$$\begin{aligned} W_2 &= W_{2A} + W_{2B} \\ &= \sum_{i=0}^{l-1} \frac{i+1}{\mu} \pi_{i,0} + \sum_{i=1}^{l} \sum_{j=1}^{i} \frac{j}{\mu} \pi_{i,j} \end{aligned} \tag{3.26}$$

3.6 安定条件

この節では相乗りを考慮するタクシー乗り場の待ち行列モデルの安定条件を探る．そのためにペアになっていないタイプ 2 の乗客の並ばせる位置を操作して安定条件を探る．

3.6.1 タイプ 2 後回しモデル

通常の相乗りを考慮するタクシー乗り場の待ち行列モデルに以下の条件を加えたモデルを考える

- まだペアを組んでいないタイプ 2 の乗客は常に待ち行列の最後尾に並ばせる

すると状態遷移図は図 3.5 のように描ける．これをもとに平衡方程式を書き下すと式 (3.27),(3.28),(3.29),(3.30) のように導出される．

$$-(\lambda_1 + \lambda_2)\pi_{0,0} + \mu\pi_{1,0} + \mu\pi_{1,1} = 0 \tag{3.27}$$

$$\lambda_2\pi_{0,0} - (\lambda_1 + \lambda_2 + \mu)\pi_{1,1} + \mu\pi_{2,2} = 0 \tag{3.28}$$

$$\lambda_1\pi_{i-1,0} - (\lambda_1 + \lambda_2 + \mu)\pi_{i,0} + \lambda_2\pi_{i,i} + \mu\pi_{i+1,0} = 0 \quad (i = 1, 2, \cdots) \tag{3.29}$$

$$\lambda_2\pi_{i-1,0} + \lambda_1\pi_{i-1,i-1} - (\lambda_1 + \lambda_2 + \mu)\pi_{i,i} + \mu\pi_{i+1,i+1} = 0 \quad (i = 2, 3, \cdots) \tag{3.30}$$

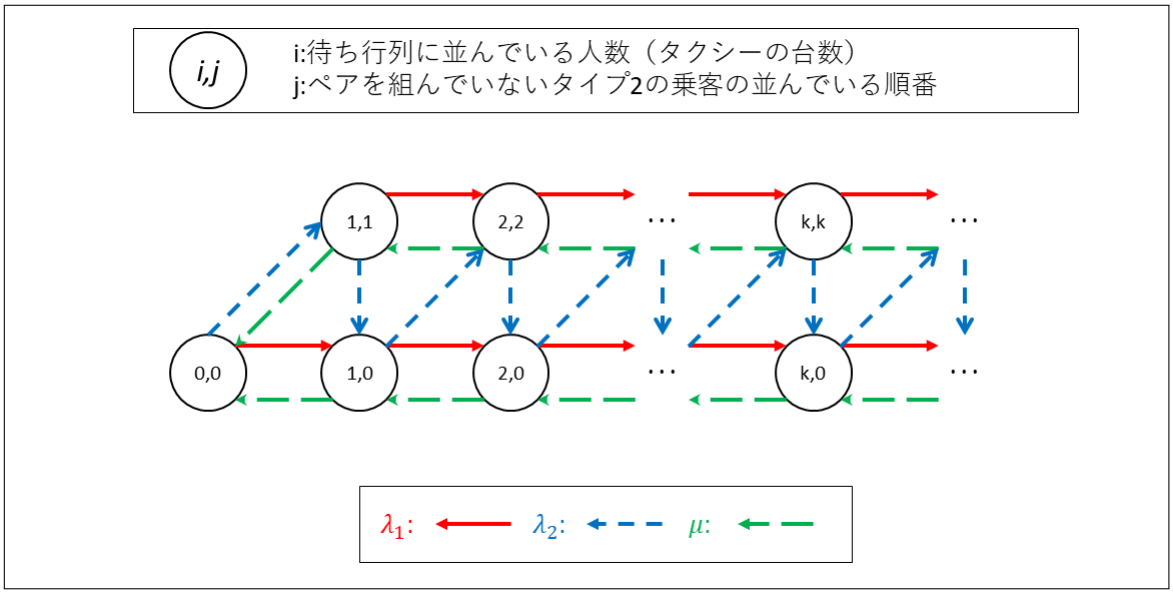

図 3.5　タイプ 2 後回しモデルの状態遷移図

式 (3.27),(3.28),(3.29),(3.30) を元に以下のような値を設定するとこのモデルは準出生死滅過程で表すことができる．

$$\pi_{0,0} = \boldsymbol{\pi}_0 \ , \ (\pi_{i,0}, \pi_{i,i}) = \boldsymbol{\pi}_i \quad (i = 1, 2, \ldots) \ , \ \boldsymbol{\pi} = (\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots)$$

$$\boldsymbol{Q}_{+1} = \begin{pmatrix} \lambda_1 & \lambda_2 \\ 0 & \lambda_1 \end{pmatrix} \ , \ \boldsymbol{Q}_0 = \begin{pmatrix} -(\lambda_1 + \lambda_2 + \mu) & 0 \\ \lambda_2 & -(\lambda_1 + \lambda_2 + \mu) \end{pmatrix} \ , \ \boldsymbol{Q}_{-1} = \begin{pmatrix} \mu & 0 \\ 0 & \mu \end{pmatrix}$$

$$\boldsymbol{B}_{+1} = \begin{pmatrix} \lambda_1 & \lambda_2 \end{pmatrix} \ , \ \boldsymbol{B}_0 = -(\lambda_1 + \lambda_2) \ , \ \boldsymbol{B}_{+1} = \begin{pmatrix} \mu \\ \mu \end{pmatrix}$$

$$\boldsymbol{\pi}\boldsymbol{Q} = (\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots) \begin{pmatrix} \boldsymbol{B}_0 & \boldsymbol{B}_{+1} & \boldsymbol{O} & \boldsymbol{O} & \cdots \\ \boldsymbol{B}_{-1} & \boldsymbol{Q}_0 & \boldsymbol{Q}_{+1} & \boldsymbol{O} & \cdots \\ \boldsymbol{O} & \boldsymbol{Q}_{-1} & \boldsymbol{Q}_0 & \boldsymbol{Q}_{+1} & \cdots \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{Q}_{-1} & \boldsymbol{Q}_0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} = \boldsymbol{O} \tag{3.31}$$

ドリフト条件を計算すると式 (3.32) のように導出される．

$$\tilde{\boldsymbol{Q}} = \boldsymbol{Q}_{+1} + \boldsymbol{Q}_0 + \boldsymbol{Q}_{-1} = \begin{pmatrix} -\lambda_2 & \lambda_2 \\ \lambda_2 & -\lambda_2 \end{pmatrix}$$

$$\begin{pmatrix} \tilde{\pi_0} & \tilde{\pi_1} \end{pmatrix} \begin{pmatrix} -\lambda_2 & \lambda_2 \\ \lambda_2 & -\lambda_2 \end{pmatrix} = \begin{pmatrix} -\lambda_2\tilde{\pi_0} + \lambda_2\tilde{\pi_1} & \lambda_2\tilde{\pi_0} - \lambda_2\tilde{\pi_1} \end{pmatrix} = \begin{pmatrix} 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} \tilde{\pi_0} & \tilde{\pi_1} \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \tilde{\pi_0} + \tilde{\pi_1} = 1 \ , \ \begin{pmatrix} \tilde{\pi_0} & \tilde{\pi_1} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} \end{pmatrix}\begin{pmatrix} \lambda_1 & \lambda_2 \\ 0 & \lambda_1 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \lambda_1 + \frac{1}{2}\lambda_2$$
$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} \end{pmatrix}\begin{pmatrix} \mu & 0 \\ 0 & \mu \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \mu$$
$$\lambda_1 + \frac{1}{2}\lambda_2 < \mu \tag{3.32}$$

式 (3.32) の両辺を μ で割ると，このモデルの安定条件は式 (3.33) で表される．

$$\rho_1 + \frac{\rho_2}{2} < 1 \tag{3.33}$$
$$\text{ただし} \rho_1 = \frac{\lambda_1}{\mu} \ , \ \rho_2 = \frac{\lambda_2}{\mu}$$

3.6.2 タイプ 2k 優先モデル

通常の相乗りを考慮するタクシー乗り場の待ち行列モデルに以下の条件を加えたモデルを考える

- タイプ 2 の乗客が到着したときに，ペアを組めなかった場合は待ち行列の k 番目に並ばせる
- ただし待ち行列に k 人乗客が並んでいない場合は待ち行列の最後尾に並ばせる

すると状態遷移図は図 3.6 のように描ける．これをもとに平衡方程式を書き下すと式 (3.34)〜(3.40) で導出される．

$$-(\lambda_1 + \lambda_2)\pi_{0,0} + \mu\pi_{1,0} + \mu\pi_1, 1 = 0 \tag{3.34}$$
$$\lambda_1\pi_{i-1,0} - (\lambda_1 + \lambda_2 + \mu)\pi_{i,0} + \sum_{m=1}^{i} \lambda_2\pi_{i,m} + \mu\pi_{i+1,0} + \mu\pi_{i+1,1} = 0 \quad (i = 1, 2, \ldots, k-1) \tag{3.35}$$
$$\lambda_1\pi_{i-1,0} - (\lambda_1 + \lambda_2 + \mu)\pi_{i,0} + \sum_{m=1}^{k} \lambda_2\pi_{i,m} + \mu\pi_{i+1,0} + \mu\pi_{i+1,1} = 0 \quad (i = k, k+1, \ldots) \tag{3.36}$$
$$\lambda_1\pi_{i-1,j} - (\lambda_1 + \lambda_2 + \mu)\pi_{i,j} + \mu\pi_{i+1,j+1} = 0 \quad (0 < j < i) \tag{3.37}$$
$$\lambda_2\pi_{i-1,0} - (\lambda_1 + \lambda_2 + \mu)\pi_{i,i} + \mu\pi_{i+1,i+1} = 0 \quad (i = 1, 2, \ldots, k-1) \tag{3.38}$$
$$\lambda_2\pi_{k-1,0} - (\lambda_1 + \lambda_2 + \mu)\pi_{k,k} = 0 \tag{3.39}$$
$$\lambda_2\pi_{i-1,0} + \lambda_1\pi_{i-1,k} - (\lambda_1 + \lambda_2 + \mu)\pi_{i,k} = 0 \quad (i = k+1, k+2, \ldots) \tag{3.40}$$

このモデルも準出生死滅過程の条件を満たすものになっている．安定条件を探るために必要な値だけを以下に列挙する．

$$\boldsymbol{Q}_{+1} = \begin{pmatrix} \lambda_1 & 0 & 0 & \cdots & \lambda_2 \\ 0 & \lambda_1 & 0 & \cdots & 0 \\ 0 & 0 & \lambda_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_1 \end{pmatrix} \ , \ \boldsymbol{Q}_{-1} = \begin{pmatrix} \mu & 0 & \cdots & 0 & 0 \\ \mu & 0 & \cdots & 0 & 0 \\ 0 & \mu & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \mu & 0 \end{pmatrix}$$

$$\boldsymbol{Q}_0 = \begin{pmatrix} -(\lambda_1 + \lambda_2 + \mu) & 0 & 0 & \cdots & 0 \\ \lambda_2 & -(\lambda_1 + \lambda_2 + \mu) & 0 & \cdots & 0 \\ \lambda_2 & 0 & -(\lambda_1 + \lambda_2 + \mu) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \lambda_2 & 0 & 0 & \cdots & -(\lambda_1 + \lambda_2 + \mu) \end{pmatrix}$$

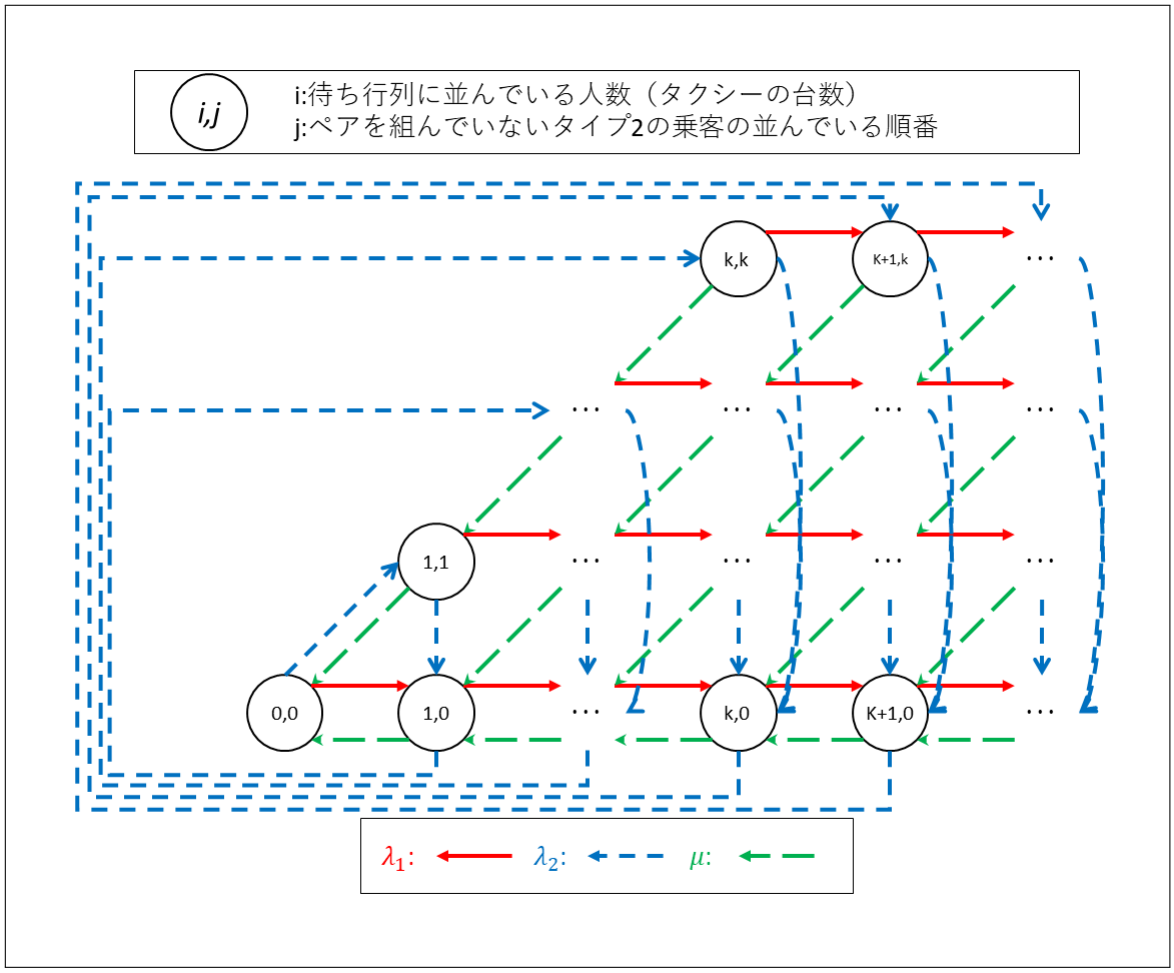

図 3.6　タイプ 2k 優先モデルの状態遷移図

$$\tilde{\boldsymbol{Q}} = \begin{pmatrix} -\lambda_2 & 0 & \cdots & 0 & \lambda_2 \\ \lambda_2+\mu & -(\lambda_2+\mu) & \cdots & 0 & 0 \\ \lambda_2 & \mu & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \lambda_2 & 0 & \cdots & \mu & -(\lambda_2+\mu) \end{pmatrix}$$

ただし，以上の行列式はいずれも $k+1$ 字の正方行列である．これをもとにドリフト条件を調べるため，まず $\tilde{\boldsymbol{\pi}}$ を求める．$\tilde{\boldsymbol{Q}}$ が $k+1$ 次の正方行列なので式 (3.41) で定義される．

$$\tilde{\boldsymbol{\pi}} = \begin{pmatrix} \tilde{\boldsymbol{\pi}}_0 & \tilde{\boldsymbol{\pi}}_1 & \tilde{\boldsymbol{\pi}}_2 & \ldots & \tilde{\boldsymbol{\pi}}_k \end{pmatrix} \tag{3.41}$$

$\tilde{\boldsymbol{\pi}}\tilde{\boldsymbol{Q}} = \boldsymbol{O}$ と $\tilde{\boldsymbol{\pi}}\boldsymbol{1}^T = 1$ を解くと式 (3.42),(3.43),(3.44),(3.45) で導出される．

$$\tilde{\boldsymbol{\pi}}_0 + \tilde{\boldsymbol{\pi}}_1 + \tilde{\boldsymbol{\pi}}_2 + \ldots + \tilde{\boldsymbol{\pi}}_k = 1 \tag{3.42}$$

$$-\lambda_2\tilde{\boldsymbol{\pi}}_0 + +(\lambda_2+\mu)\tilde{\boldsymbol{\pi}}_1 + \lambda_2\tilde{\boldsymbol{\pi}}_2 + \ldots + \lambda_2\tilde{\boldsymbol{\pi}}_k = 0 \tag{3.43}$$

$$-(\lambda_2+\mu)\tilde{\boldsymbol{\pi}}_n + \mu\tilde{\boldsymbol{\pi}}_n + 1 = 0 \quad (n = 1, 2, \ldots, k-1) \tag{3.44}$$

$$\lambda_2\tilde{\boldsymbol{\pi}}_0 - (\lambda_2+\mu)\tilde{\boldsymbol{\pi}}_k = 0 \tag{3.45}$$

ここで $\lambda_2/\mu = \rho_2$ とおき，式 (3.42) と式 (3.43) を λ_2 で割ったものを足すと式 (3.46) が導出される．

$$\begin{aligned} & \tilde{\boldsymbol{\pi}}_0 + \tilde{\boldsymbol{\pi}}_1 + \tilde{\boldsymbol{\pi}}_2 + \ldots + \tilde{\boldsymbol{\pi}}_k = 1 \\ +) \quad & -\lambda_2\tilde{\boldsymbol{\pi}}_0 + +(\lambda_2+\mu)\tilde{\boldsymbol{\pi}}_1 + \lambda_2\tilde{\boldsymbol{\pi}}_2 + \ldots + \lambda_2\tilde{\boldsymbol{\pi}}_k = 0 \end{aligned}$$

$$2\tilde{\boldsymbol{\pi}}_0 - \frac{1}{\rho_2}\tilde{\boldsymbol{\pi}}_1 = 1 \tag{3.46}$$

一方で式 (3.44),(3.45) より式 (3.47),(3.48) が導出される．

$$\begin{aligned} \tilde{\boldsymbol{\pi}}_1 &= \frac{1}{\rho_2+1}\tilde{\boldsymbol{\pi}}_2 \\ &= \frac{1}{(\rho_2+1)^2}\tilde{\boldsymbol{\pi}}_3 \\ &= \frac{1}{(\rho_2+1)^{k-1}}\tilde{\boldsymbol{\pi}}_k \\ &= \frac{1}{(\rho_2+1)^{k-1}}\frac{\rho_2}{\rho_2+1}\tilde{\boldsymbol{\pi}}_0 \\ &= \frac{\rho_2}{(\rho_2+1)^k}\tilde{\boldsymbol{\pi}}_0 \end{aligned} \tag{3.47}$$

$$\tilde{\boldsymbol{\pi}}_n = (\rho_2+1)^{n-1}\tilde{\boldsymbol{\pi}}_1 \tag{3.48}$$

よって式 (3.46) に式 (3.47) を代入することで式 (3.49),(3.50) が導出される．

$$\tilde{\boldsymbol{\pi}}_0 = \frac{(\rho_2+1)^k}{2(\rho_2+1)^k - 1} \tag{3.49}$$

$$\tilde{\boldsymbol{\pi}}_1 = \frac{\rho_2}{2(\rho_2+1)^k - 1} \tag{3.50}$$

ここで式 (3.48) を用いることで $\tilde{\boldsymbol{\pi}}$ は式 (3.51) が導出される．

$$\tilde{\boldsymbol{\pi}} = \frac{1}{2(\rho_2+1)^k - 1}\begin{pmatrix} (\rho_2+1)^k & \rho_2 & \rho_2(\rho_2+1) & \rho_2(\rho_2+1)^2 & \cdots & \rho_2(\rho_2+1)^{k-1} \end{pmatrix} \tag{3.51}$$

$\tilde{\boldsymbol{\pi}}$ を用いてドリフト条件式を解くと式 (3.52),(3.53) が導出される．

$$\tilde{\boldsymbol{\pi}}\boldsymbol{Q}_{+1}\mathbf{1}^T = \lambda_1 + \frac{(\rho_2+1)^k\lambda_2}{2(\rho_2+1)^k - 1} \tag{3.52}$$

$$\tilde{\boldsymbol{\pi}}\boldsymbol{Q}_{-1}\mathbf{1}^T = \mu \tag{3.53}$$

よってこのモデルの安定条件式 (3.54) が導出される．

$$\lambda_1 + \frac{(\rho_2+1)^k\lambda_2}{2(\rho_2+1)^k - 1} < \mu \iff \rho_1 + \frac{\rho_2}{2-(\rho_2+1)^{-k}} < 1 \tag{3.54}$$

3.6.3 上界と下界

ここで，通常のモデル，タイプ 2 後回しモデル，タイプ 2k 優先モデルの 3 つが全く同じ確率過程で表される場合系内客数はどうなるかを調べた．時刻 0 から 30 の間で図 3.7 のようにイベントを起こした．図 3.7 の挙動をもとに 3 つのモデルにおいて系内客数をグラフ化したものが図 3.8, 3.9,3.10 である．なお，図 3.9,3.10 には図 3.8 の結果を線で表記している． 通常のモデルとタイプ 2 後回しモデルを比べるとタイプ 2 後回しモデルの方が平均系内客数が少ないことが分かる．これは通常のモデルでは相乗りができなかったタイプ 2 の乗

時刻	イベント内容	通常のモデル						タイプ2後回し						タイプ2k優先					
		数	待ち行列					数	待ち行列					数	待ち行列				
0	イベント開始	0						0						0					
1	タイプ1到着	1	×					1	×					1	×				
4	タイプ2到着	2	×	○				2	×	○				2	○	×			
8	タイプ1到着	3	×	○	×			3	×	×	○			3	○	×	×		
12	サービス終了	2	○	×				2	×	○				2	×	×			
16	サービス終了	1	×					1	○					1	×				
19	タイプ2到着	2	×	○				1	◎					2	○	×			
22	サービス終了	1	○					0						1	×				
25	タイプ2到着	1	◎					1	○					2	○	×			
27	サービス終了	0						0						1	×				
29	サービス終了	0						0						0					
30	イベント終了	0						0						0					

タイプ1：× タイプ2：○ ペア済のタイプ2：◎

図 3.7　イベント表

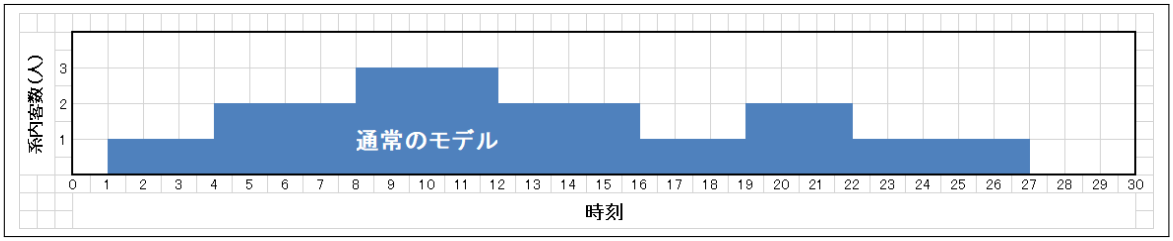

図 3.8　通常のモデルの系内客数変化

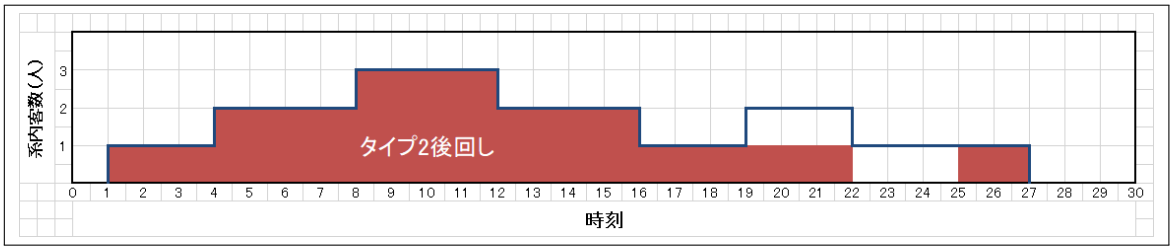

図 3.9　タイプ 2 後回しモデルの系内客数変化

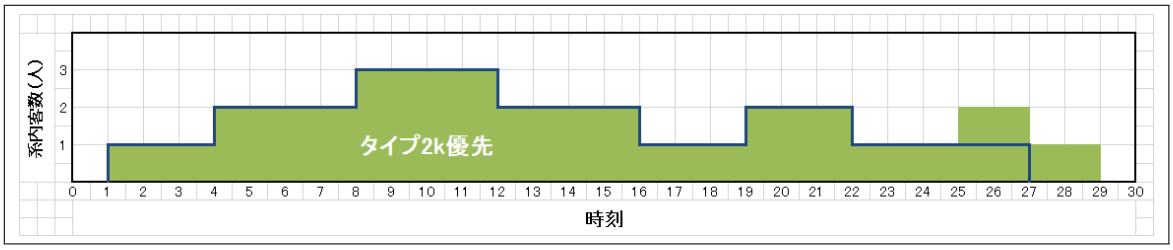

図 3.10　タイプ 2k 優先モデルの系内客数変化

客が相乗りできる可能性を増やせているため平均系内客数を減らせていることが分かる．このように乗客の到着と退去が同じ確率過程で表される場合において，タイプ 2 後回しモデルの系内客数は必ず通常のモデルの系内客数以下の値をとることが分かる．これは言い換えればタイプ 2 後回しモデルは常に通常のモデルの利用率以下の値の利用率をとるということであり，タイプ 2 後回しモデルの利用率が通常のモデルの利用率の上界となることが分かる．

一方で，通常のモデルとタイプ 2k 優先モデルを比べるとタイプ 2k 優先モデルの方が平均系内客数が多いことが分かる．これは通常のモデルでは相乗りができたタイプ 2 の乗客が相乗りできる可能性を減らしているため平均系内客数を増やしていることが分かる．このように乗客の到着と退去が同じ確率過程で表される場合において，タイプ 2k 優先モデルの系内客数は必ず通常のモデルの系内客数以上の値をとることが分かる．これは言い換えればタイプ 2k 優先モデルは常に通常のモデルの利用率以上の値の利用率をとるということであり，タイプ 2k 優先モデルの利用率が通常のモデルの利用率の下界となることが分かる．

ここでタイプ 2k 優先モデルの安定条件 (3.54) について，タイプ 2 k 優先モデルの k を限りなく大きくした場合，タイプ 2 k 優先モデルは通常のモデルと等価となる．このときの安定条件式を計算すると $\rho_1 + \rho_2/2 < 1$ となる．この条件式はタイプ 2 後回しモデルの安定条件と一致するため相乗りモデルの安定条件は式 (3.55) が導出される．

$$\rho_1 + \frac{\rho_2}{2} < 1 \tag{3.55}$$

3.6.4　安定条件への誘導

相乗りを考慮した待ち行列モデルでは安定条件が $\rho_1 + \rho_2/2 < 1$ であることから全ての乗客を相乗り可能にした場合に半分の利用率まで下げることができる．よって相乗りを考慮しないモデルの利用率が $1 \leq \rho < 2$ のときに，一定割合以上の乗客に相乗りを許容させることでこのシステムが定常状態をもつようにすることができる．$\rho_1 + \rho_2/2 < 1$ とするために必要なタイプ 2 の乗客の割合を k とすると，式 (3.56) を満たす k を探せばよいことになる．

$$(1-k)\rho + \frac{k\rho}{2} = \left(1 - \frac{k}{2}\right)\rho < 1 \tag{3.56}$$

式 (3.56) を整理すると k に関する安定条件式 (3.57) が導出される．

$$k > 2(1 - \rho^{-1}) \tag{3.57}$$

第 4 章

解析

第 4 章では第 3 章で述べた相乗りを考慮するタクシー乗り場の待ち行列モデルを実際に解析する手法を述べる．なお，それぞれのプログラムは python を用いて実行した．

4.1　シミュレーション

待ち行列のシミュレーションに用いられる手法は以下の 2 つが挙げられる．

時間駆動型（タイムドリブン）シミュレーション

時間駆動型シミュレーションは実際に時刻 t を $0 \leq t \leq t_{max}$ の区間で dt 毎に増やしていき，システムを観測するものである．各時刻 dt で乱数 rnd を発生させて，$0 \leq rnd < \lambda dt$ であれば客を発生させる．客が発生しなかった場合は $0 \leq rnd < \mu dt$ であれば客を離脱させる．それぞれの客に到着時刻やサービス開始時刻，退去時刻を与え，最終的にその値を用いて各種数値の算出を行う．詳しくはシミュレーションによる確率論 [12] を参照されたい．

イベント駆動型（イベントドリブン）シミュレーション

イベント駆動型シミュレーションは，まず乱数を用いて全ての客の到着時刻を発生させて到着時刻を予定させる．次に到着時刻と乱数を元にすべての客の退去時刻を発生させ退去時刻を予定させる．以上の操作を行うことによりある時刻における待ち行列長を導出することができるため各種数値の算出が行える．

本論文においては M/M/1 モデルを用いた相乗りを考慮するタクシー乗り場の待ち行列モデルを検討するにあたって時間駆動型シミュレーションを用いる．具体的な操作は図 4.1 を参照されたい．また，微小時間 $dt = 0.001$，シミュレーション時間 $t_{max} = 10000$ としてシミュレーションを行い，次節で述べる大域平衡方程式の解析との結果の比較を行う．実際に用いたプログラムコードは付録 A を参照されたい．

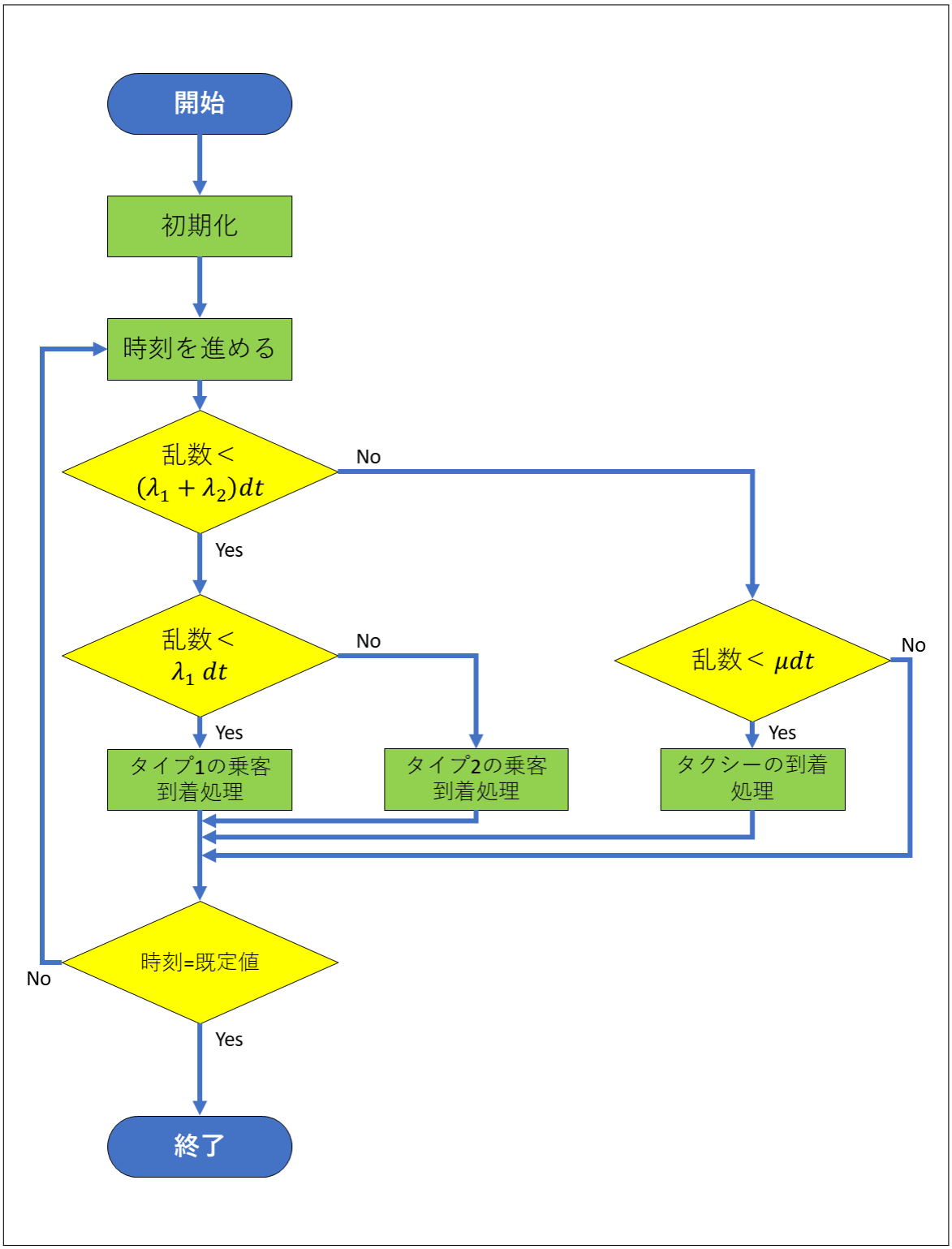

図 4.1　時間駆動型シミュレーションのフローチャート

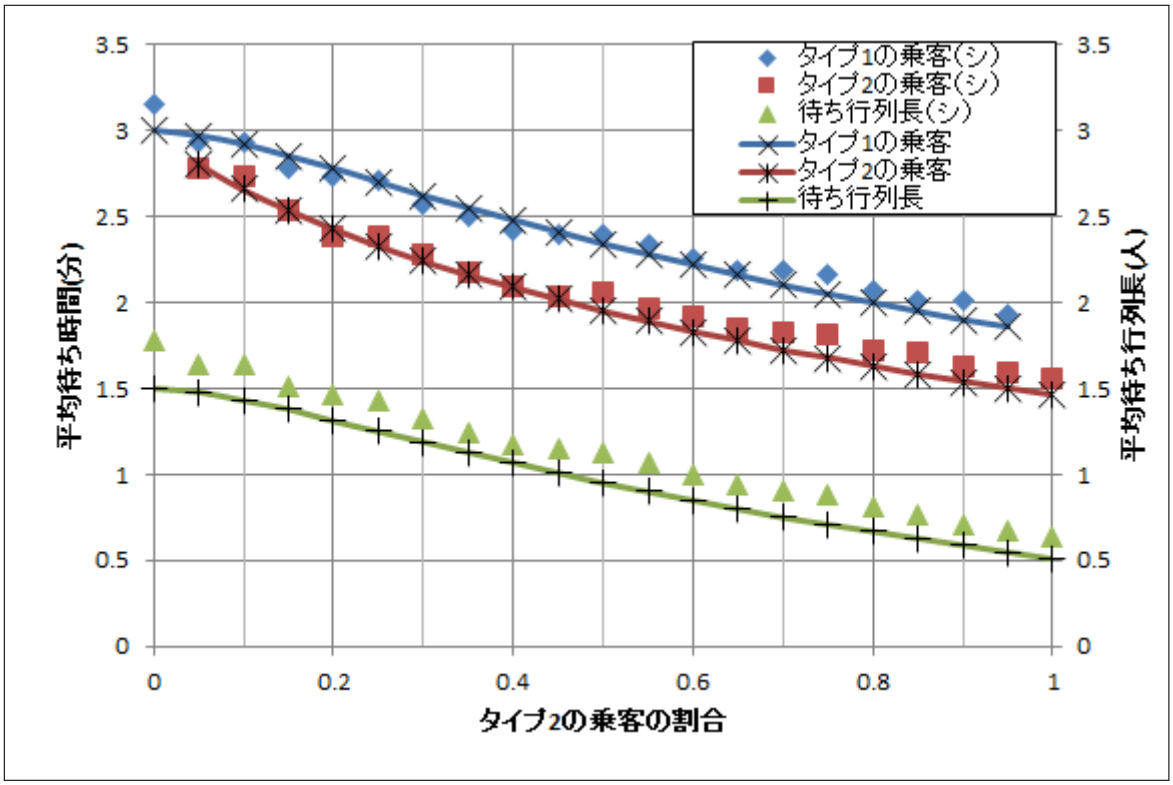

図 4.2　比較 $(\lambda_1 + \lambda_2, \mu) = (1/2, 5/6)$

4.2　大域平衡方程式の解析

大域平衡方程式の解析は実際に第 3 章で導出した方程式を計算させて，各種数値を導出した．実際に用いたプログラムコードは付録 B を参照されたい．

4.3　2 つの解析の比較

シミュレーションと大域平衡方程式の解析 2 つのプログラムが確かであることを確認するためそれぞれ $(\lambda_1 + \lambda_2, \mu) = (1/2, 5/6), (3/4, 5/6)$ としてタイプ 1 の乗客とタイプ 2 の乗客の比 $\lambda_2/(\lambda_1 + \lambda_2)$ を 0 から 1 まで 0.05 刻みで変化させてプログラムを実行し比較を行う．図 4.2 が $(\lambda_1 + \lambda_2, \mu) = (1/2, 5/6)$ のとき，図 4.3 が $(\lambda_1 + \lambda_2, \mu) = (3/4, 5/6)$ のときの結果である． シミュレーションの結果が点，大域平衡方程式の解析結果が点と平滑線で表されている．図 4.2,4.3 のどちらの図を見てもシミュレーション結果の方が大域平衡方程式の解析結果よりも数値がやや大きくなってはいるが，シミュレーション結果と大域平衡方程式の解析結果に大きな齟齬はないことからどちらのプログラムも正しいとして，理論値が計算できる大域平衡方程式の解析を用いて第 5 章では議論していくこととする．

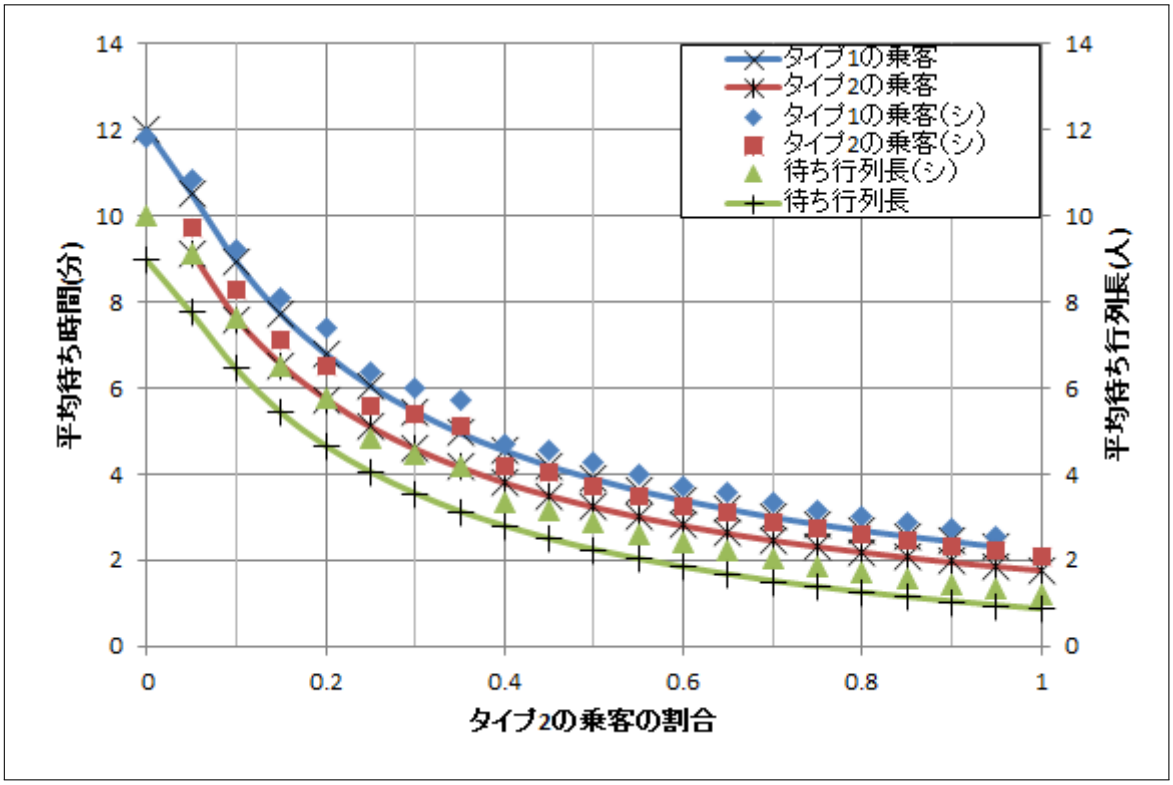

図 4.3　比較 $(\lambda_1 + \lambda_2, \mu) = (3/4, 5/6)$

第 5 章

解析結果・考察

この章では前章までで構築した相乗りを考慮するタクシー乗り場モデルの様々なケースを想定し大域平衡方程式の解析を用いてこのモデルがどのような特徴を持つのかを調べていく．

5.1　利用率による比較

まず，相乗りを考慮しないタクシー乗り場モデルを 3 パターン用意する．パラメータはタクシーの到着率 μ は 5/6（台/分）で固定して，乗客の到着率 λ を $1/2, 3/4, 19/24$（人/分）の 3 パターン分を用意する．この 3 パターンのモデルはそれぞれ利用率 $\rho = 0.6, 0.9, 0.99$ であることも確認されたい．ここで $\lambda_1 + \lambda_2 = \lambda$ として，それぞれのモデルにおいてタクシー乗り場を訪れる乗客のうち相乗りを許容する乗客の割合 $\lambda_2/(\lambda_1 + \lambda_2)$ を変化させて相乗りを許容する乗客の割合の変化が乗客の平均待ち時間や平均待ち行列長にどのような変化を及ぼすかを見ていく．

図 5.1 が $(\lambda_1+\lambda_2, \mu) = (1/2, 5/6)$ のモデル，図 5.2 が $(\lambda_1+\lambda_2, \mu) = (3/4, 5/6)$ のモデル，図 5.3 が $(\lambda_1+\lambda_2, \mu) = (19/24, 5/6)$ のモデルにおける平均待ち時間と平均待ち行列長の推移を示している．また，図 5.4 はそれぞれのモデルのタイプ 1 の乗客の平均待ち時間の推移の比較を，図 5.5 はそれぞれのモデルのタイプ 2 の乗客の平均待ち時間の推移の比較を，図 5.6 はそれぞれのモデルの平均待ち行列長の推移の比較を示している．

元々高利用率であったモデルであればあるほどすべての乗客が相乗りを許容する乗客になったときの平均待ち時間や平均待ち行列長の減少幅は顕著であることが分かる．また，相乗りを許容する乗客の割合が等しい場合について比較すると，元々高利用率であったモデルであればあるほど相乗りを許容する乗客が全く存在しないときと比べて平均待ち時間や平均待ち行列長が大幅に減少していることが分かる．例えば相乗りを許容する乗客が全体の 2 割のとき，$(\lambda_1 + \lambda_2, \mu) = (1/2, 5/6)$ のケースでは元々のタイプ 1 の乗客の平均待ち時間 3 分と平均待ち行列長 1.5 人に対してタイプ 1 の乗客の平均待ち時間 2.78 分，平均待ち行列長 1.31 人と大きな効果が得られていないが，$(\lambda_1 + \lambda_2, \mu) = (19/24, 5/6)$ のケースでは元々のタイプ 1 の乗客の平均待ち時間 24 分と平均待ち行列長 19 人に対してタイプ 1 の乗客の平均待ち時間 8.85 分，平均待ち行列長 6.37 人と大きな効果が得られていることが分かる．

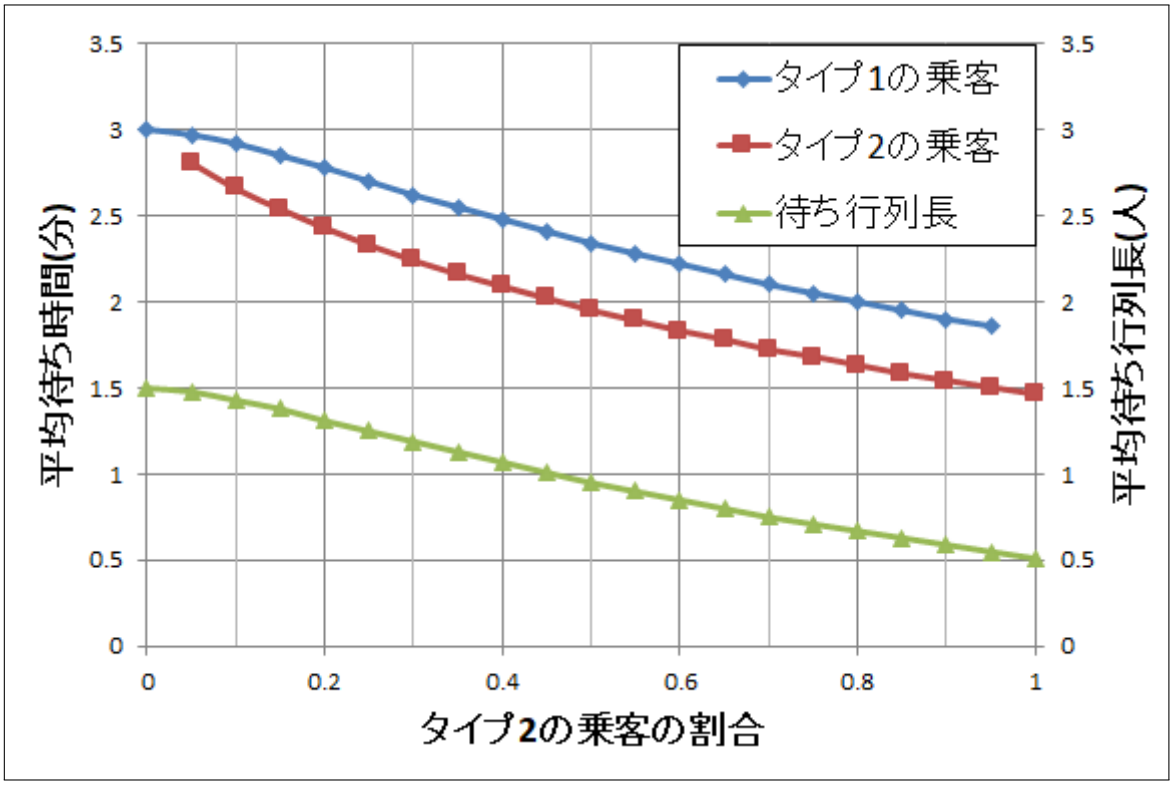

図 5.1　$(\lambda_1 + \lambda_2, \mu) = (1/2, 5/6)$ における平均待ち時間と平均待ち行列長の推移

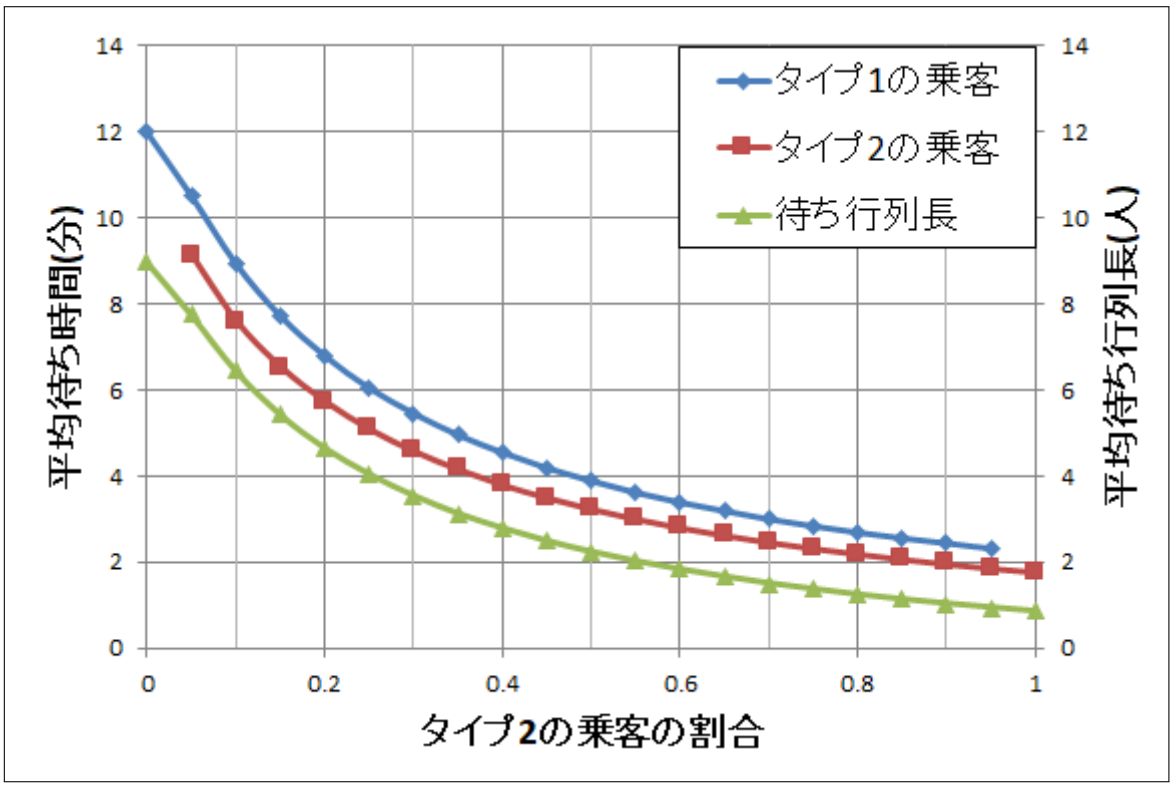

図 5.2　$(\lambda_1 + \lambda_2, \mu) = (3/4, 5/6)$ における平均待ち時間と平均待ち行列長の推移

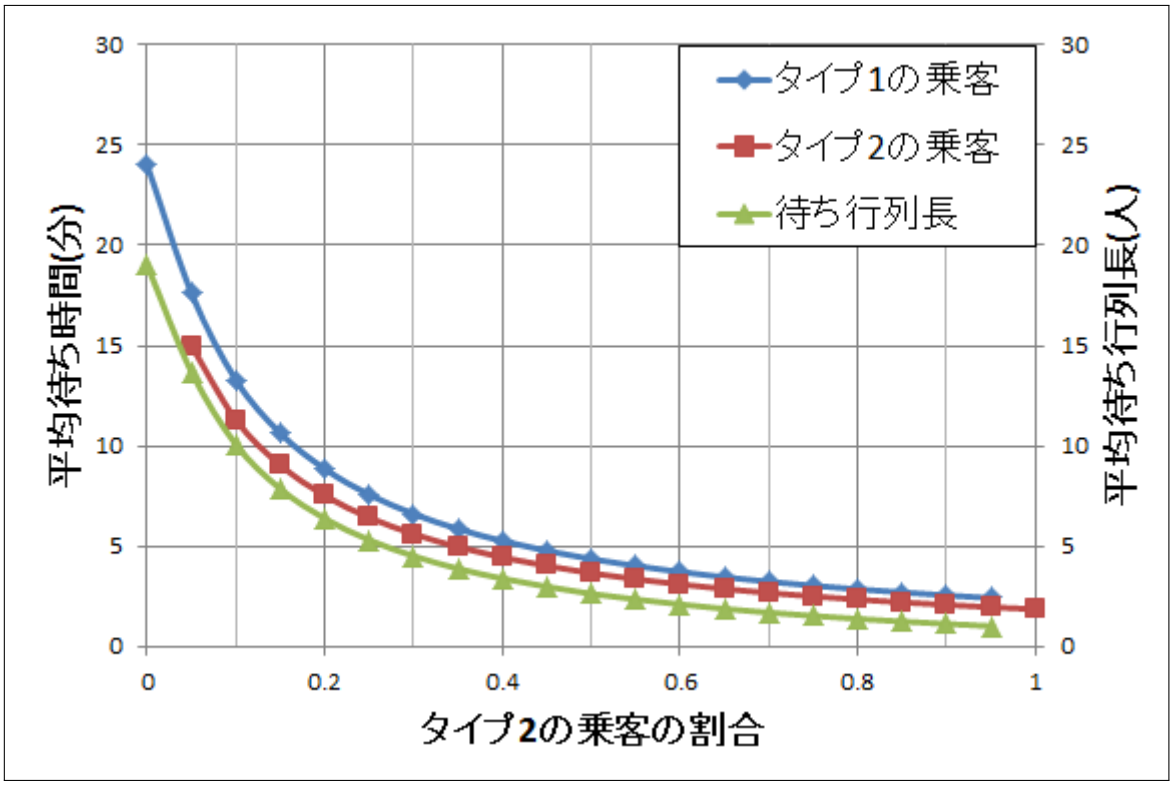

図 5.3　$(\lambda_1 + \lambda_2, \mu) = (19/24, 5/6)$ における平均待ち時間と平均待ち行列長の推移

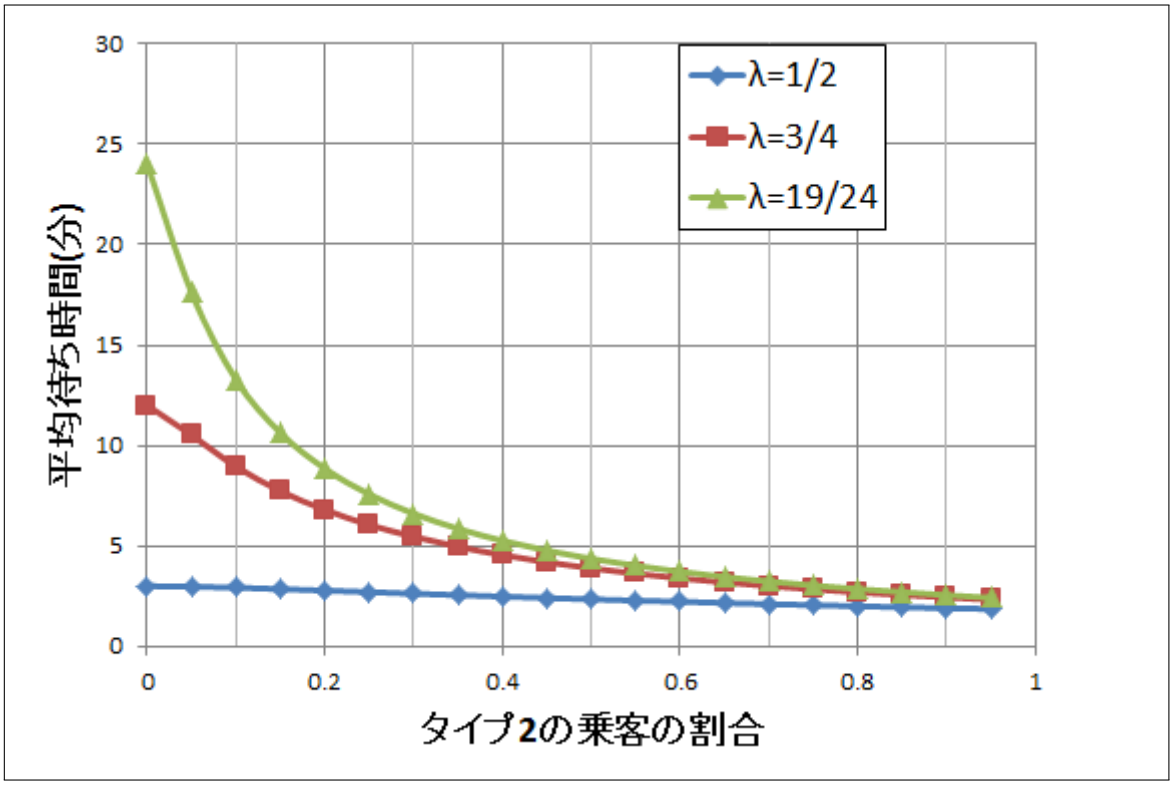

図 5.4　それぞれのモデルのタイプ 1 の乗客の平均待ち時間の推移の比較

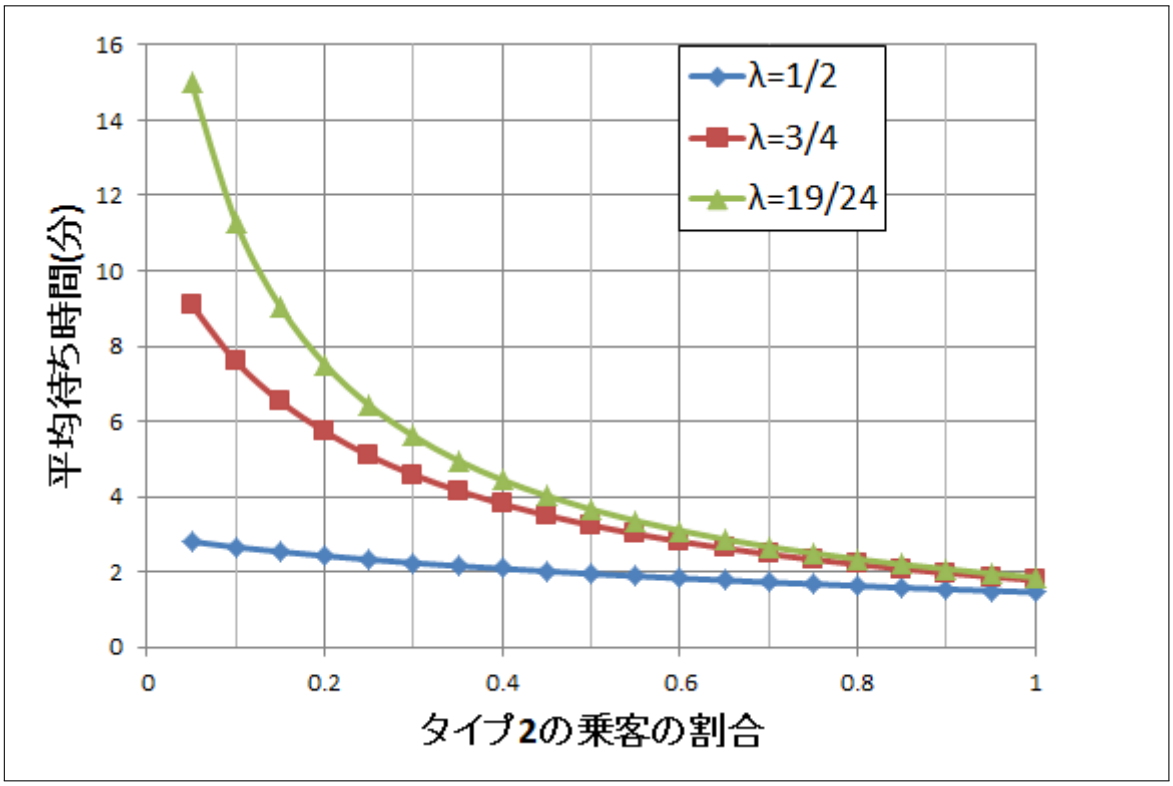

図 5.5　それぞれのモデルのタイプ 2 の乗客の平均待ち時間の推移の比較

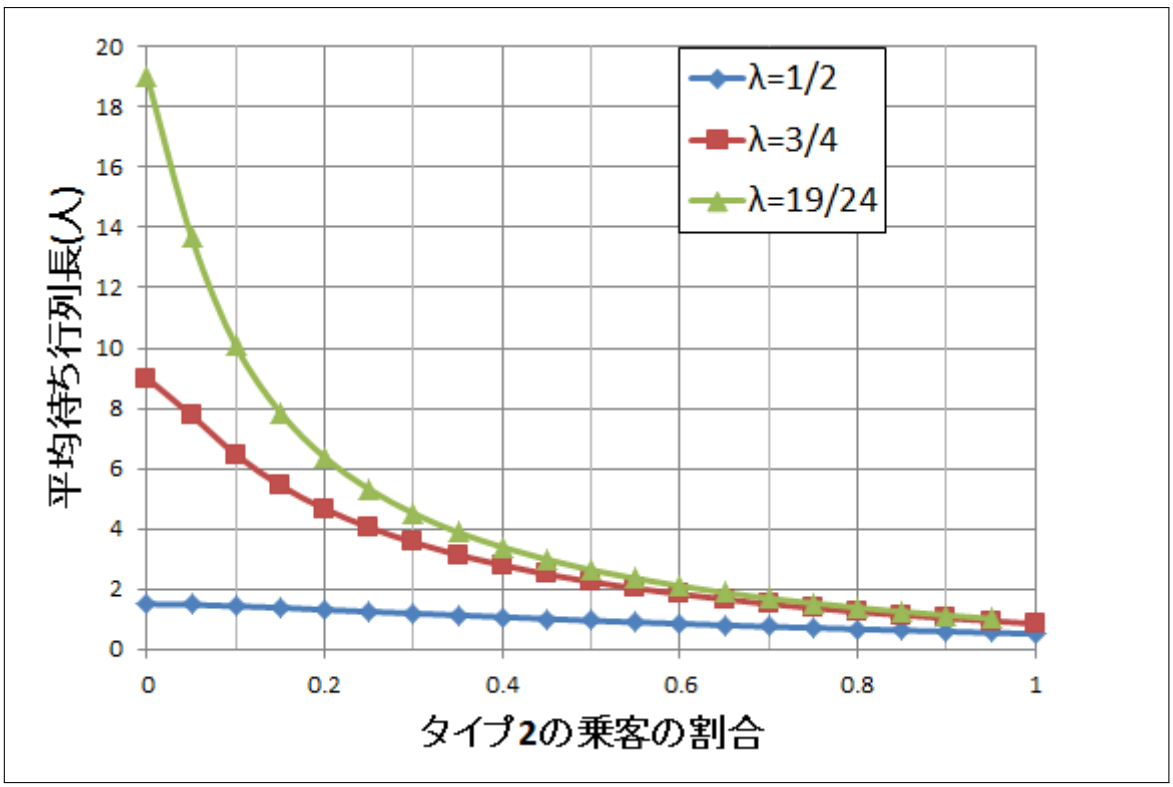

図 5.6　それぞれのモデルの平均待ち行列長の推移の比較

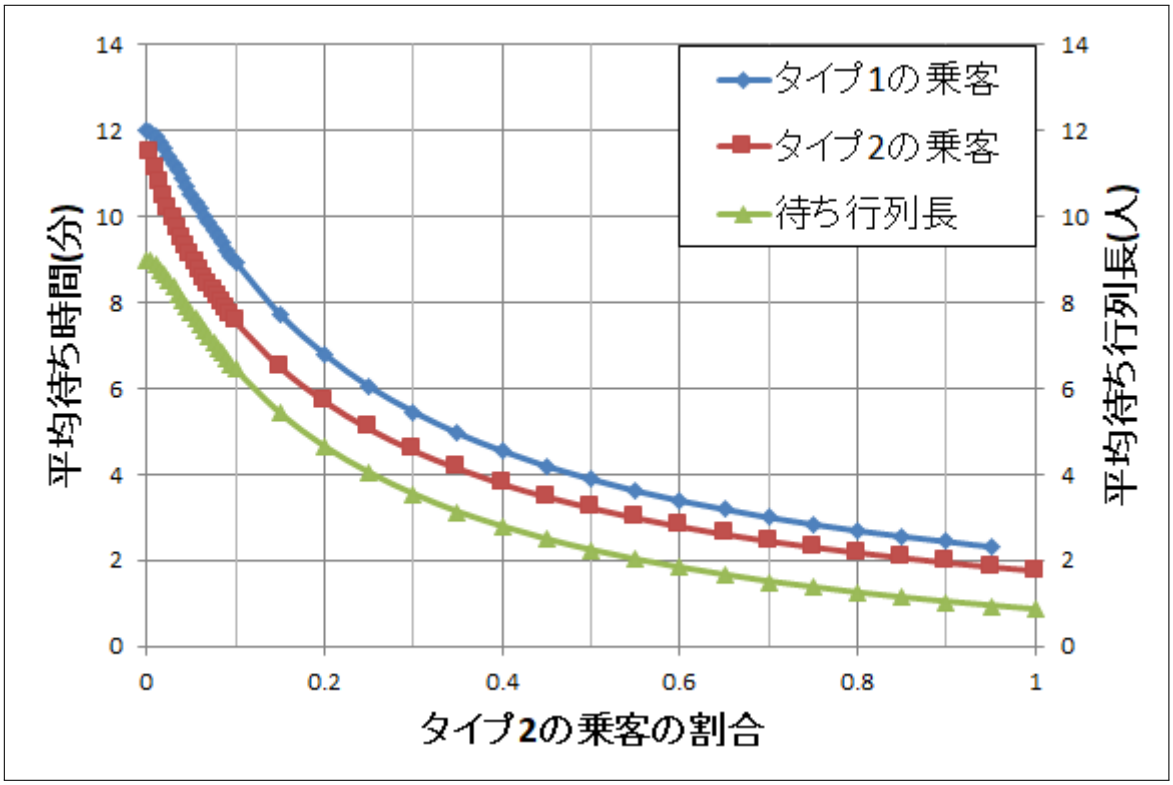

図 5.7　$(\lambda_1 + \lambda_2, \mu) = (3/4, 5/6)$ における平均待ち時間と平均待ち行列長の推移 (ver2)

5.2　ペア成立率と平均待ち時間，平均待ち行列長の関係性

ここで図 5.1,5.2,5.3 の 3 つを比較すると，図 5.2,5.3 のタイプ 1 の乗客の平均待ち時間と平均待ち行列長は指数関数状の変化をしているように見て取れるが，図 5.1 のタイプ 1 の乗客の平均待ち時間と平均待ち行列長は指数関数状の変化をしているようには見えない．実のところ図 5.1 のように指数関数状の変化をしない―指数関数状の変化をしはじめるタイプ 2 の乗客の割合が遅れてやってくるように見える―現象は図 5.2 のときにも生じており（図 5.7,5.8 を参照されたい），元のモデルの利用率に関わらず発生するものと思われる．相乗りを考慮するタクシー乗り場の待ち行列モデルには以上のような特性を持つことが分かった．この特性はペア成立率に依存していると推測できる．3 つのパターンのペア成立率を比較したグラフが図 5.9 である．ペア成立率が低い，図 5.9 でいえばペア成立率が 0.05 程度のときはただでさえ少ない相乗りを許容する乗客のほとんどがペアを組むことができないため相乗りを許容する乗客が全くいない時と比べても相乗りによる平均待ち行列長，平均待ち時間の減少の効果がほとんど得られない．このように相乗りが効果をあまり発しない区間が存在し，ペア成立率が一定の値になるまでは指数関数状の変化をしないと推測できる．

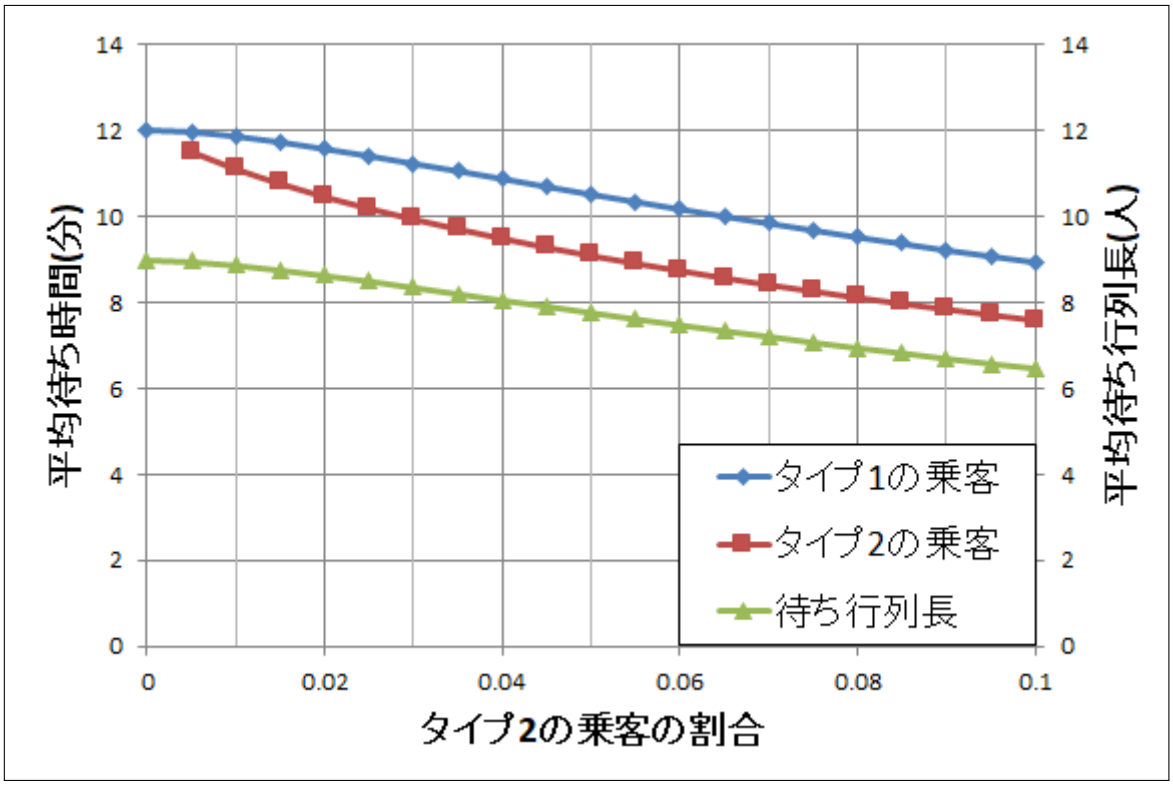

図 5.8　$(\lambda_1 + \lambda_2, \mu) = (3/4, 5/6)$ における平均待ち時間と平均待ち行列長の推移 $(0 < \lambda_2/(\lambda_1 + \lambda_2) < 0.1)$

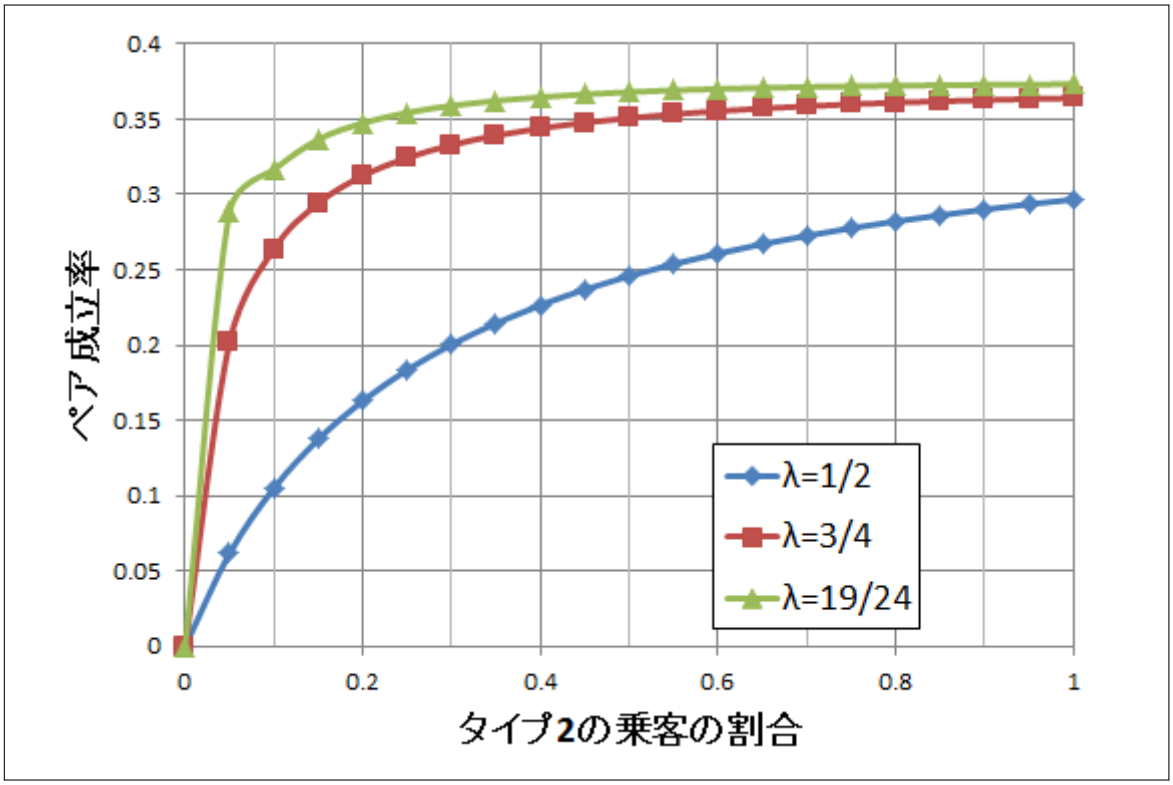

図 5.9　それぞれのモデルのペア成立率の比較

第 6 章

結論

本研究では大域平衡方程式の解析により相乗りを考慮するタクシー乗り場の待ち行列モデルの乗客の平均待ち時間，平均待ち行列長を解析できることが分かった．また安定条件の導出も行い，$\rho_1 + \rho_2/2 < 1$ で表されることが分かった．実際に乗客，タクシーの到着率を設定して，相乗りを許容する乗客の割合を変化させて平均待ち時間や平均待ち行列長を比較したところ，特に高利用率の場合，相乗りを許容する乗客の増加によって平均待ち時間や平均待ち行列長が大幅に短縮できることが分かった．なお，現時点において無限バッファモデルでの結果は，有限バッファモデルにおいてバッファサイズが十分大きいときの結果で代用することしかできないので無限バッファモデルを自己完結的に計算できるようにすることが今後の目標である．また，発展的な内容として以下のような研究をしていく所存である．

- M/G/1 モデル，G/G/1 モデルを元にした相乗りを考慮するタクシー乗り場のモデルの解析
- ペアできる人数を 2 人から増やしたモデルの解析
- 相乗りを許容する乗客を増加させる最適戦略の解析

付録 A

シミュレーションのプログラムコード

ソースコード A.1　シミュレーションのプログラムコード

```
# -*- coding: UTF-8 -*-
#↑このおまじないがないと日本語をファイル内に書き込めない
#ここから本体
import numpy.random as rnd
import matplotlib.pyplot as plt
import numpy as np
import copy

lmd1 = 3 / 8
lmd2 = 3 / 8
lmd = lmd1 + lmd2
mu = 5 / 6
t_max = 10000
dt = 0.001
c_max = t_max * 2

nstep = int(t_max / dt)
index = 1
findex = 1
oknum = 0
kekka = np.zeros(c_max * 8, dtype = float)
service = 0
support = np.array([0])
queue = np.zeros(c_max, dtype = int)
qtop = -1
kekka.shape = c_max, 8

sum1 = 0
sum2 = 0
sump = 0
wait1 = 0
wait2 = 0
pretime = 0.0
len = 0
lenave = 0.0

for it in range(0, nstep):
    t = it * dt
    pr = rnd.rand()
    if pr < lmd * dt:
        if pr < lmd1 * dt:
            kekka[index - 1][0] = index
            kekka[index - 1][1] = 1
            kekka[index - 1][2] = 0
            kekka[index - 1][3] = t
            if t >= t_max / 10:
                lenave += len * (t - pretime)
            pretime = t
            len += 1
            if service == 0:
                service = index
                kekka[index - 1][4] = t
                kekka[index - 1][6] = 0
                qtop = 0
            else:
                queue[qtop] = index
                qtop += 1
```

```
        else:
            kekka[index - 1][0] = index
            kekka[index - 1][1] = 2
            if oknum == 0:
                kekka[index - 1][2] = -1
                kekka[index - 1][3] = t
                if t >= t_max / 10:
                    lenave += len * (t - pretime)
                pretime = t
                len += 1
                oknum = index
                if service == 0:
                    service = index
                    kekka[index - 1][4] = t
                    kekka[index - 1][6] = 0
                    qtop = 0
                else:
                    queue[qtop] = index
                    qtop += 1
            else:
                kekka[index - 1][2] = oknum
                kekka[oknum - 1][2] = index
                kekka[index - 1][3] = t
                if oknum == service:
                    kekka[index - 1][4] = t
                oknum = 0
        index += 1
        continue
    if service != 0:
        if rnd.rand() < mu * dt:
            while(kekka[findex - 1][1] == 2) and (kekka[findex - 1][2] != -1) and (findex > kekka[findex -
              1][2]):
                findex += 1

            if kekka[findex - 1][1] == 1:
                kekka[findex - 1][5] = t
                kekka[findex - 1][7] = kekka[findex - 1][5] - kekka[findex - 1][3]
                if t >= t_max / 10:
                    lenave += len * (t - pretime)
                pretime = t
                len -= 1
                findex += 1
                service = queue[0]
                kekka[service - 1][4] = t
                kekka[service - 1][6] = t - kekka[service-1][3]
                if (kekka[service - 1][2] != 0) and (kekka[service - 1][2] != -1):
                    support[0] = kekka[service - 1][2]
                    kekka[support[0] - 1][4] = t
                    kekka[support[0] - 1][6] = kekka[support[0] - 1][4] - kekka[support[0] - 1][3]
                for i in range(0, qtop):
                    queue[i] = queue[i + 1]
                queue[qtop] = 0
                qtop -= 1
            elif (kekka[findex - 1][1] == 2) and (kekka[findex - 1][2] == -1):
                kekka[findex - 1][5] = t
                kekka[findex - 1][7] = kekka[findex - 1][5] - kekka[findex - 1][3]
                if t >= t_max / 10:
                    lenave += len * (t - pretime)
                pretime = t
                len -= 1
                oknum = 0
                findex += 1
                service = queue[0]
                kekka[service - 1][4] = t
                kekka[service - 1][6] = t - kekka[service-1][3]
                if (kekka[service - 1][2] != 0) and (kekka[service - 1][2] != -1):
                    support[0] = kekka[service - 1][2]
                    kekka[support[0] - 1][4] = t
                    kekka[support[0] - 1][6] = kekka[support[0] - 1][4] - kekka[support[0] - 1][3]
                for i in range(0, qtop):
                    queue[i] = queue[i + 1]
                queue[qtop] = 0
                qtop -= 1
            elif (kekka[findex - 1][1] == 2) and (findex < kekka[findex - 1][2]):
                kekka[findex - 1][5] = t
                kekka[findex - 1][7] = kekka[findex - 1][5] - kekka[findex - 1][3]
                support[0] = kekka[findex - 1][2]
                kekka[support[0] - 1][5] = t
                kekka[support[0] - 1][7] = kekka[support[0] - 1][5] - kekka[support[0] - 1][3]
                if t >= t_max / 10:
                    lenave += len * (t - pretime)
```

```
                pretime = t
                len -= 1
                oknum = 0
                findex += 1
                service = queue[0]
                kekka[service - 1][4] = t
                kekka[service - 1][6] = t - kekka[service-1][3]
                if (kekka[service - 1][2] != 0) and (kekka[service - 1][2] != -1):
                    support[0] = kekka[service - 1][2]
                    kekka[support[0] - 1][4] = t
                    kekka[support[0] - 1][6] = kekka[support[0] - 1][4] - kekka[support[0] - 1][3]
                for i in range(0, qtop):
                    queue[i] = queue[i + 1]
                queue[qtop] = 0
                qtop -= 1
            continue
for i in range(0, index - 1):
    if (kekka[i][5] != 0) and (kekka[i][3] >= (t_max / 10)):
        if kekka[i][1] == 1:
            sum1 += 1
            wait1 += kekka[i][7]
            waita += kekka[i][7]
        elif kekka[i][1] == 2:
            sum2 += 1
            wait2 += kekka[i][7]
            waita += kekka[i][7]
            if kekka[i][2] != -1:
                sump += 0.5


if (sum1 != 0) and (lmd1 != 0):
    wait1 /= sum1
else:
    wait1 = 0

if (sum2 != 0) and (lmd2 != 0):
    wait2 /= sum2
else:
    wait2 = 0

if kekka[sum1 + sum2 -1][5] != 0:
    lenave /= kekka[sum1 + sum2 - 1][5] - (t_max / 10)
else:
    lenave /= kekka[sum1 + sum2 - 2][5] - (t_max / 10)

print("␣␣␣␣␣␣lmd1␣␣␣␣␣␣␣lmd2␣␣␣␣␣␣␣␣␣mu)
print("%10.4f"%lmd1,␣end='␣')
print("%10.4f"%lmd2,␣end='␣')
print("%10.4f"%mu,␣end='␣')
print()
print("wait_type1 wait_type2    len_ave")
print("%10.2f"%wait1,␣end='␣')
print("%10.2f"%wait2,␣end='␣')
print("%10.2f"%lenave,␣end='␣')
print()
```

付録 B

大域方程式の解析のプログラムコード

ソースコード B.1　大域方程式の解析のプログラムコード

```
# -*- coding: UTF-8 -*-
#↑このおまじないがないと日本語をファイル内に書き込めない

#行列式を作るためのおまじない
import numpy as np

#定常状態確立の要素数を返す
def ele(x):
    sum = 0
    to = x + 1
    for i in range(1,to + 1):
        sum += i
    return sum

#-----ここから各種変数の定義----------------------------------------------------
#乗客のバッファサイズ
buffer = 200
#タクシーのバッファサイズ
carbuf = 0
#相乗りを認めない客の到着率
lam1 = 6 / 8
#相乗りを認める客の到着率
lam2 = 0 / 8
#タクシーの到着率
mu   = 5 / 6
#行列式の係数のカギ
key = 0
key2 = 0
key3 = 0
#平均系内客数
ELC = 0
ELT = 0
#平均待ち時間
EW1 = 0
EW2 = 0
EWT = 0
#ペアになっていないタイプ２が行列にいる確率
pailam2 = 1
#ele()の計算を一度きりにする
cuspat = ele(buffer)
#-----ここまで-----------------------------------------------------------------
#iの値でまとめた定常状態確率を格納する行列の生成
Psum = np.zeros(buffer + carbuf + 1,dtype = float)

#行列paiに乗算する行列の用意
jou = np.zeros([cuspat + carbuf,1],dtype = float)
jou[0,0] = 1.0

#行列paiを定常状態確立の要素分の正方行列として全要素0で初期化
pai = np.zeros([cuspat + carbuf,cuspat + carbuf],dtype = float)

#行列paiの一行目の全要素を1にする
for i in range (0,cuspat + carbuf):
    pai[0,i] = 1.0

key += 1

```

```
#各定常状態確率の平衡方程式の係数を入力する
Tsum = lam1 + lam2 + mu
for i in range (1,buffer):
    for j in range (0,i+1):
        if j == 0:
            pai[key,key]          = -Tsum
            pai[key,key - i]      =  lam1
            pai[key,key + i + 1] =  mu
            pai[key,key + i + 2] =  mu
            for k in range (1,i+1):
                pai[key,key + k] =  lam2
        elif j == i:
            pai[key,key]          = -Tsum
            pai[key,key - i - j] =  lam2
            pai[key,key + i + 2] =  mu
        else:
            pai[key,key]          = -Tsum
            pai[key,key - i]      =  lam1
            pai[key,key + i + 2] =  mu
        key += 1

for j in range (0,buffer + 1):
    b = buffer
    if j == 0:
        pai[key,key]          = -mu
        pai[key,key - b]      =  lam1
        for k in range (1,b + 1):
            pai[key,key + k] =  lam2
    elif j == b:
        pai[key,key]          = -(mu + lam2)
        pai[key,key - b - b] =  lam2
    else:
        pai[key,key]          = -(mu + lam2)
        pai[key,key - b]      =  lam1
    key += 1

if carbuf != 0:
    if carbuf != 1:
        pai[key,key]      = -Tsum
        pai[key,0]        =  mu
        pai[key,key + 1] =  lam1 + lam2
        key += 1
    for k in range (0,carbuf - 2):
        pai[key,key]      = -Tsum
        pai[key,key - 1] =  mu
        pai[key,key + 1] =  lam1 + lam2
        key += 1
    pai[key,0]       = -Tsum
    pai[key,1]       =  mu
    pai[key,2]       =  mu
    pai[key,cuspat] =  lam1 + lam2
    key += 1

#print("行列式の設定完了\n")

#方程式を解く
kekka = np.linalg.solve(pai,jou)

#iの値でまとめた定常状態確率の計算・表示
temp = 0
for i in range(0,buffer + 1):
    for j in range (0,i + 1):
        Psum[i] += kekka[temp]
        temp += 1
for i in range(0,buffer + 1):
    print("π(%3d)=%5.2f%%  "%(i,100*Psum[i]),end = "")
    if (i % 5) == 0:
        print()
if (buffer % 5) != 0:
    print()
print()

#タクシーの定常状態確率の表示
for k in range (1,carbuf + 1):
    print("π(%3d)=%5.2f%%  "%(-k,100*kekka[cuspat + k - 1]),end = "")
    if (k % 5) == 0:
        print()
if (carbuf % 5) != 0:
    print()
print()

```

```
print("================各種数値一覧================")
print("乗客のバッファサイズ      =%8d人"%buffer)
print("タクシーのバッファサイズ  =%8d台"%carbuf)
print("タイプ1の到着率           =%8.2f人／分"%lam1)
print("タイプ2の到着率           =%8.2f人／分"%lam2)
print("タクシーの到着率          =%8.2f台／分"%mu)
print("================計算結果表示================")

#呼損率の計算・表示
koson1 = Psum[buffer]
koson2 = kekka[cuspat - buffer - 1]
kosont = kekka[cuspat + carbuf - 1]
if lam1 != 0:
    print("タイプ1の呼損率           =%8.2f%%"%(100*koson1))
if lam2 != 0:
    print("タイプ2の呼損率           =%8.2f%%"%(100*koson2))
if mu   != 0:
    print("タクシーの呼損率          =%8.2f%%"%(100*kosont))

#平均系内客数を求める
for i in range (0,buffer + 1):
    ELC += i * Psum[i]
print("平均乗客行列長            =%8.2f人"%ELC)

#平均系内タクシー数を求める
if carbuf != 0:
    for k in range (1,carbuf + 1):
        ELT += k * kekka[cuspat + k - 1]
else:
    ELT = 0

#タクシー平均待ち時間を求める
if carbuf != 0:
    EWT += kekka[0] / (lam1 + lam2)
for k in range (1,carbuf):
    EWT += (k+1) * kekka[cuspat + k - 1] / (lam1 + lam2)
print("平均タクシー行列長        =%8.2f台"%ELT)

#平均待ち時間を求める
for i in range (0,buffer):
    EW1 += 1 / mu * (i + 1) * Psum[i]
for i in range (0,buffer):
    for j in range (0,i + 1):
        if j == 0:
            EW2 += 1 / mu * (i + 1) * kekka[key2]
        else:
            EW2 += 1 / mu * j * kekka[key2]
        key2 += 1
for j in range (0,buffer + 1):
    if j == 0:
        0
    else:
        EW2 += 1 / mu * j * kekka[key2]
    key2 += 1

if lam1 != 0:
    print("タイプ1の平均待ち時間     =%8.2f分"%EW1)
if lam2 != 0:
    print("タイプ2の平均待ち時間     =%8.2f分"%EW2)
print("タクシーの平均待ち時間     =%8.2f分"%EWT)

#ペアになっていないタイプ2が行列にいる確率
for i in range(0, buffer + 1):
    for j in range(0, i + 1):
        if j == 0:
            pailam2 -= kekka[key3]
        key3 += 1
for i in range(0,carbuf):
    pailam2 -= kekka[key3]
    key += 1
print("未ペアのタイプ2がいる確率=%8.2f%%"%(100*pailam2))
print("================プログラム終了================")
```

謝辞

本論文を作成するにあたり，指導教員である塩田茂雄先生に多大にご協力いただいことをここに記し，感謝の意を伝えさせていただきたい．また，塩田先生をはじめとする塩田研究室一同の皆様に関しましても常日頃お世話になり，そのうえで本論文が完成したことをここに記し，感謝の意を伝えさせていただきたい．

参考文献

[1] 高橋幸雄，待ち行列研究の変遷，オペレーションズ・リサーチ 経営の科学，vol.43，No.9，pp.495-499（1998）

[2] Agner K. Erlang，Probability and Telephone Calls，Nyt Tidsskr. Mat. seriesB，Vol.20，pp.33-3(1909)

[3] David G. Kendall, Stochastic Processes Occurring in the Theory of Queues and Their Analysis by the Method of the Imbedded Markov Chain，The Annals of Mathematical Statistics，vol.24，pp.338-354(1953)

[4] 塩田茂雄，河西憲一，豊泉洋，会田雅樹，待ち行列理論の基礎とその応用，共立出版，第 5 章（2014）. 川島幸之助（監修）

[5] 大石進一，待ち行列理論，コロナ社，第 5 章 (2003)

[6] 宮沢政清，待ち行列の数理とその応用，牧野書店，第 2 章 (2006)

[7] 尾畑伸明，確率モデル要論 確立論の基礎からマルコフ連鎖へ，牧野書店，第 6 章 (2012)

[8] 松島格也，小林潔司，坂口潤一，タクシースポット市場の差別化と社会的厚生，土木学会論文集，No.723，pp.41-53（2003）

[9] Ying Shi，Zhaotong Lian，European Journal of Operational Research，249，pp.1024-1032（2016）

[10] 松島格也，小林潔司，タクシー・サービスのスポット市場均衡に関する研究，土木計画学研究・論文集，No.16，pp.591-600（1999）

[11] 滝根哲哉，M/M/1 を越えて―準出生死滅過程への招待―，オペレーションズ・リサーチ 経営の科学，vol.59，No.4，pp.179-184（2014）

[12] 山本浩，森隆一，藤曲哲郎，シミュレーションによる確率論，日本評論社，第 6 章 (1993)