統計数理(2025)
第 73 巻 第 2 号 267–285

特集「生存時間解析と信頼性解析」
[研究詳解]

信頼性解析とツールの活用

岡村 寛之[†]・鄭 俊俊[†]・土肥 正[†]

(受付 2024 年 11 月 29 日；改訂 2025 年 2 月 28 日；採択 2 月 28 日)

要　　旨

本稿では，信頼性分野で使用されるツールについて紹介する．信頼性評価は，故障木やマルコフモデルに代表される確率モデルを用いたアプローチと，信頼性試験などから得られるデータをもとにモデルの同定を行う統計的手法に大別される．これらの手法はいずれも，信頼性を評価するために信頼性解析ツールが必要となる．本稿では，信頼性に関する基礎知識を紹介し，その後，具体的な信頼性解析ツールについて説明する．

キーワード：信頼性解析ツール，確率モデル，統計的手法，信頼性評価．

1. はじめに

近年，IoT や AI を用いたシステムがあらゆる分野に応用されつつある．特に自動運転など我々の命に関わるようなセーフティクリティカルシステムが複雑化の一途を辿り，その信頼性確保が急務な課題となっている．このような状況下，システムの信頼性を定量的に評価することは，製品開発のあらゆる段階において必要不可欠である．

信頼性評価は，大きく分けて二つのアプローチが存在する．一つは，故障木やマルコフモデルに代表される確率モデルを用いたアプローチである．このアプローチでは，システムの構造や部品の故障率をもとに，システム全体の信頼度や故障率を算出する．もう一つは，信頼性試験などから得られるデータをもとに，システムの信頼性モデルを同定する統計的手法である．このアプローチでは，過去のデータから将来の信頼性を予測することができる．これらの手法はいずれも，信頼性を評価するために信頼性解析ツールが必要となる．信頼性解析ツールは，複雑なシステムの信頼性モデルを構築し，様々な指標を算出することで，信頼性評価を支援する．近年では，商用ソフトウェアからオープンソースソフトウェアまで，様々なツールが開発・提供されている．

本稿では，信頼性分野で使用されるツールとその事例について紹介する．まず，信頼性に関する基礎知識として，確率モデルを用いた信頼性評価手法と統計的手法について概説する．その後，具体的な信頼性解析ツールについて，商用ソフトウェア，オープンソースソフトウェア，学術分野で開発されたツールなどを例に挙げ，それぞれの特徴や機能について説明する．

本稿の構成は以下の通りである．2 節では，信頼性に関する基礎知識について述べる．3 節では，確率モデルを用いた信頼性解析の基礎のうち，故障木を用いた静的モデルとその解析について言及する．特に，ツール化に必要なアルゴリズムを中心に概説する．4 節は，マルコフ連鎖を用いた信頼性解析の基礎と，数値的な計算アルゴリズムについて紹介する．5 節では，

[†]広島大学 大学院先進理工系科学研究科：〒 739–8527 広島県東広島市鏡山 1–4–1; okamu@hiroshima-u.ac.jp, jzheng@hiroshima-u.ac.jp, dohi@hiroshima-u.ac.jp

信頼性における統計解析の基礎と，相型分布と呼ばれる分布のパラメータ推定手続きを中心に紹介する．6 節では，信頼性に関するツールの紹介を行い，一つの信頼性評価事例を紹介する．最後に 7 節において，信頼性解析ツールの限界および今後の課題について言及する．

2. 信頼性の基礎知識

信頼性とは「ある期間において，所定の条件の下で，対象が意図した機能を遂行する能力」として定義されている（Barlow and Proschan, 1965）．ここで「ある期間」および「所定の条件」は，信頼性が時間の経過ならびに使用環境によって変化するものであり，信頼性を評価する際にはこれらの条件を明確に定義する必要がある．また，「意図した機能」とは，製品やシステムが期待される性能を発揮することが前提となり，一般的な故障を含むより広い概念として規定されている．信頼性は，製品やシステムが安全に運用されるために不可欠な要素であり，製品が備えるべき一つの重要な要素となっている．

信頼性工学では，信頼性を故障確率と関連付けて定量化することから始まる．故障確率とは，ある期間内に対象が故障する確率を表し，信頼性が高いほど故障確率は低くなる．確率変数 X を対象の寿命とし，$F(x)$ を寿命に対する累積分布関数（故障分布関数）とする．このとき，対象の信頼度は時刻 t において故障しない（寿命を迎えていない）確率として定義され，次のように表される．

$$R(t) = P(X > t) = 1 - F(t). \tag{2.1}$$

ここで，$R(t)$ は信頼度関数と呼ばれ，信頼度が時間の経過に伴って単調に減少することを表している．信頼度の時間変化を表す信頼度関数 $R(t)$ は，信頼性評価の基本的な指標として用いられる．また，故障分布関数の導関数 $f(t) = dF(t)/dt$ を用いて，故障率関数 $\lambda(t)$ が次のように定義される．

$$\lambda(t) = \frac{f(t)}{R(t)} = \frac{f(t)}{1 - F(t)}. \tag{2.2}$$

故障率は，時間の経過に伴う故障の発生頻度を示し，信頼性評価において重要な指標の一つである．例えば，故障率が一定である場合，時間経過による故障率の変化が見られないことを意味し，これは故障がランダムに発生する状況（偶発故障）を表す．この場合，故障分布は指数分布に従う．一方，対象が経年劣化する場合には，故障率は時間とともに増加する．また，初期不良のように時間経過とともに故障率が減少する場合もある．このように，故障率が時間に応じて増減する場合，故障分布は指数分布よりも一般的なワイブル分布などに従う．故障率と信頼度関数には以下の関係が成り立つ．

$$R(t) = e^{-\int_0^t \lambda(s)ds}. \tag{2.3}$$

上式から $\Lambda(t) = \int_0^t \lambda(s)ds = -\log R(t)$ となり，$\Lambda(t)$ を累積ハザード関数と呼び，これも信頼度評価によく用いられる．

一方で，信頼性工学では上記の時間要素を除いた信頼度あるいは故障確率（不信頼度）を用いることもよくある．その場合，信頼度は対象が特定の時点で機能する確率を表し，故障確率は文字通りその時点で既に故障している確率である．時間要素を除いた信頼度および故障確率は時刻 t を省略した記号で信頼度 R のように記述されることが多い．

信頼性工学における各種の分析では，信頼度関数 $R(t)$，信頼度 R および故障確率を求めることが主な目的となる．代表的な信頼性解析として故障木（FT: Fault Tree）解析や信頼性ブロック図（RBD: Reliability Block Diagram）解析がある（Barlow and Proschan, 1965; Birnbaum

et al., 1961)．故障木解析は，システムが故障に至る原因を木構造で表現し，各部品の故障や何らかのイベントがシステム全体の信頼性に与える影響を分析する．信頼性ブロック図解析もシステム故障と部品の故障の関係をブロック図で表現する手法であり，主に直列・並列と呼ばれるシステムの冗長性構造を視覚的に与える手法である．これらの手法は，システムを構成する部品とシステムの関係をモデル化することで，部品の信頼度からシステムの信頼度を同定する手法である．一般に，これらのモデルは静的モデルであり，部品の信頼度 R からシステムの信頼度 R を算出することを主な目的としている．ただし，Dynamic Fault Tree のように「故障の順番」の時間要素が含まれるものもある（Dugan et al., 1992)．

静的な解析と同様に，故障に至る過程をモデル化するアプローチであるが，時間による変化を考慮した手法として，マルコフモデルを用いた信頼性解析がある．マルコフモデルでは，システムを離散状態に分類し状態遷移をマルコフ過程で表現する．特に，信頼性解析では正常稼働状態と故障状態に分類し，故障状態に到達するまでの時間を考慮することで信頼度関数 $R(t)$ を求める．また，マルコフモデルと故障木を複合させることで，部品の信頼度関数 $R(t)$ からシステムの信頼度関数 $R(t)$ を求めるハイブリッド手法も存在する（Trivedi and Bobbio, 2017)．

他方，データから直接信頼度あるいは信頼度関数を同定するアプローチとして統計解析による手法がある．信頼性分野における統計解析では，主に信頼性試験や運用データから得られるデータをもとに故障分布を同定する．信頼性試験とは，部品やシステムを一定の条件下で運用し，故障や劣化の減少を観測する手法である．一般に，システムの信頼性試験や運用データを収集することは莫大なコストを必要とするため，信頼性試験では部品を対象に行われることが多い．実際に，信頼性試験から得られたデータから部品の故障分布を最尤推定法やベイズ推定で推定した後に，先に紹介した確率モデルを組み合わせることで，最終的にシステムの信頼度を評価することができる．

3. FT 解析

3.1 モデル表現

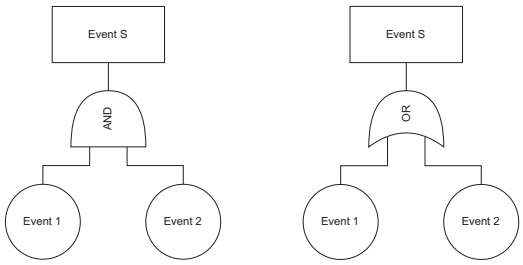
故障木(以下，FT)はシステム故障とその原因(部品の故障など)の関係を表すモデルとして利用され，システム故障に至る原因解析などを事前に抽出するために利用される．基本的な FT では原因と事象を AND ゲート(すべての原因が発生したら該当する事象が発生)，OR ゲート(少なくとも一つの原因が発生したら該当する事象が発生)あるいは k-out-of-n ゲート(n 個の原因のうち k 個が発生したら該当する事象が発生)でモデル化していく．FT は何らかの事象を表すノードと AND/OR/k-out-of-n ゲートから構成される木構造となっており，親ノードに対応する事象の発生条件をゲートと子ノードで表現する．例えば，図 1(a)は AND ゲートを用いて Event 1 と Event 2 が発生した時に Event S が発生する関係を表しており，図 1(b)は OR ゲートを用いて Event 1 と Event 2 の少なくともどちらか一方が発生した時に Event S が発生する関係を表している．実際の FT では複数の階層で構成されトップ事象(木の根ノードに対応する事象)の発生条件を原因事象(木の葉ノードに対応する事象)を用いて記述する．

FT の解析を行うために構造関数(structure function)を定義する．いま，FT における原因事象を表す変数 $x_i, i = 1, \ldots, n$ を

$$x_i = \begin{cases} 1 & (\text{原因 } i \text{ が発生している}) \\ 0 & (\text{原因 } i \text{ が発生していない}) \end{cases} \tag{3.1}$$

と定義する．この時，トップ事象の発生を表す変数 x_S が以下の式で与えられるものとする．

$$x_S = \varphi(x_1, x_2, \ldots, x_n). \tag{3.2}$$



(a) AND gate (b) OR gate

図 1. FT の例.

ここで，$\varphi(x_1, x_2, \ldots, x_n)$ は構造関数と呼ばれる論理関数であり，各原因事象の状態 x_i に応じてトップ事象の状態(0 または 1)を出力する．例えば，図 1(a)で，Event 1, Event 2 の状態(0 または 1)を表す変数 x_1, x_2 が与えられた時，Event S の状態 x_S は

$$x_S = x_1 \wedge x_2 \tag{3.3}$$

として与えられる．ここで $\wedge$ は論理積を表す．また，図 1(b)の場合は

$$x_S = x_1 \vee x_2 \tag{3.4}$$

となる．ここで，$\vee$ は論理和を表す．一般に AND ゲートは直列構造と呼ばれ，どちらか一方の部品が壊れるとシステム全体が壊れるような構造を表す．一方，OR ゲートは並列構造と呼ばれ，複数の部品が同時に壊れない限りシステム全体が壊れないような構造を表す(冗長設計)．k-out-of-n ゲートは AND ゲート，OR ゲートの中間に位置するゲートであり，n 個の部品のうち k 個以上が壊れた場合にシステム全体が壊れるような構造を表す．例えば，$k = 2, n = 3$ の場合，その構造関数は

$$x_S = (x_1 \wedge x_2) \vee (x_1 \wedge x_3) \vee (x_2 \wedge x_3) \tag{3.5}$$

となる．構造関数の論理演算による表記は簡単ではあるが実際のトップ事象の発生確率を求めるためには構造関数の多項式表現あるいは，後述する BDD(Binary Decision Diagram)を用いた手法が必要となる．

3.2 BDD による定量解析

FT では，各原因事象の発生確率からトップ事象の発生確率を算出することが必要となる．トップ事象の計算には構造関数の多項式表現が必要となる．これは，積事象を $x \wedge y \to xy$，和事象を $x \vee y \to 1 - (1 - x)(1 - y)$ と置き換えることで得られる．いま，構造関数 $\varphi(x_1, \ldots, x_n)$ から対応する多項式関数 $f_\varphi(x_1, \ldots, x_n)$ が得られたとする．$X_1, \ldots, X_n$ を原因事象の発生を表す指標確率変数，X_S をトップ事象の発生を表す指標確率変数とすると，トップ事象の発生確率は

$$P(X_S = 1) = f_\varphi(p_1, p_2, \ldots, p_n) \tag{3.6}$$

となる．ただし，ここでは原因事象が互いに独立であることを仮定している．ここで，$p_i = P(X_i = 1)$ は原因事象 X_i の発生確率である．しかしながら，繰り返し事象や k-out-of-n ゲートを含む FT では上記の単純な置き換えでは計算できない．繰り返し事象とは，同じ原因事象が葉ノードに複数回現れることを表す．図 2 は繰り返し事象を含む FT の例を示してい

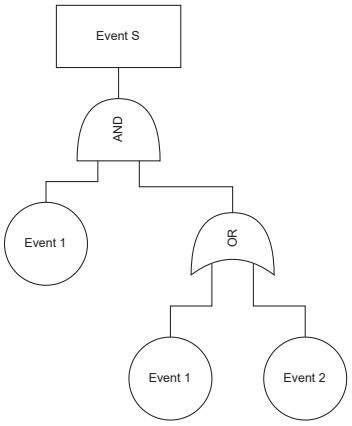


図 2. 繰り返し事象を含む FT の例.

る．この場合，Event 1 が葉ノードに複数回現れている．この FT に対する構造関数は

$$\varphi(x_1, x_2) = (x_1 \vee x_2) \wedge x_1 \tag{3.7}$$

となる．論理積，論理和を単純に $x \wedge y \to xy$，$x \vee y \to 1-(1-x)(1-y)$ で置き換えた多項式は

$$f_\varphi(x_1, x_2) = x_1^2 + x_1 x_2 - x_1^2 x_2 \tag{3.8}$$

となる．$x_1^2 = x_1$ を考慮して単純化すると，式(3.8)は $\varphi(x_1, x_2) = x_1$ となる．つまり，求めたい多項式は $f_\varphi(x_1, x_2) = x_1$ であるが，繰返し事象を含む FT に対しては，直並列の公式と多項式演算だけでなく，適切な単純化処理を行う必要がある．このような問題から，FT では構造関数を BDD(Binary Decision Diagram)で表現する手法が行われる（Rauzy, 1993）．BDD とは論理関数を非循環グラフにより表す手法であり，同じ論理値を出力するサブグラフを共有することで論理関数をコンパクトに表現することができる（Bryant, 1986）．

論理関数 $\varphi(x_1, \ldots, x_n)$ が与えられたとき，原因事象 1 の状態に着目するとシャノン分解から以下のように記述できる．

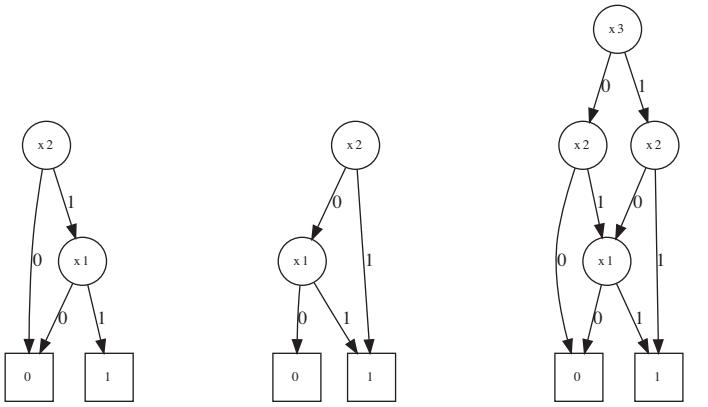
$$\varphi(x_1, \ldots, x_n) = (x_1 \wedge \varphi(1, x_2, \ldots, x_n)) \vee (\overline{x_1} \wedge \varphi(0, x_2, \ldots, x_n)). \tag{3.9}$$

ここで，$\overline{x_1} = 1 - x_1$ である．同様に $\varphi(1, x_2, \ldots, x_n)$, $\varphi(0, x_2, \ldots, x_n)$ の $x_2, \ldots, x_n$ についてもさらにシャノン分解を行うことができ，最終的には $\varphi(1, 0, \ldots, 1)$ のように具体的な値が与えられた時の論理値まで分解できる．BDD では，上記の部分的に値が与えられた $\varphi(1, x_2, \ldots, x_n)$ や $\varphi(0, x_2, \ldots, x_n)$ をノードとする二分木を構成することで論理関数を表現する．図 3(a)，図 3(b)，図 3(c)はそれぞれ式(3.3)，式(3.4)，式(3.5)の構造関数に対応する BDD 表現を示している．一番上にあるノードから各変数が取る値の枝を辿ることで最終的な構造関数の出力として 0 または 1 が得られることがわかる．

また，構造関数の多項式に対するシャノン分解は以下の再帰式となる．

$$f_\varphi(x_1, \ldots, x_n) = x_1 f_\varphi(1, x_2, \ldots, x_n) + (1 - x_1) f_\varphi(0, x_2, \ldots, x_n). \tag{3.10}$$

これと，式(3.6)の性質を利用すると BDD 表現からトップ事象の発生確率(故障確率や信頼度)を計算するアルゴリズムを構築することができる．以下の Algorithm 1 は BDD で表現された構造関数をつかってトップ事象の発生確率を計算する手続きを示している．



(a) AND ゲートの BDD (b) OR ゲートの BDD (c) 2-out-of-3 ゲートの BDD

図 3. 構造関数の BDD 表現.

Algorithm 1 Computation of Top Event Probability.

```
function prob(node)
  return visited[node] if haskey(visited, node)
  return 0 if is_zero(node)
  return 1 if is_one(node)
  p = get_prob(node)
  result = p * prob(get_one(node)) + (1-p) * prob(get_zero(node))
  visited[node] = result
  return result
end
```

ここで，visited, haskey はノードをキーとしたハッシュテーブルとハッシュテーブルに特定のキーがあるかどうか確認をする関数を表している．これは，一度計算した確率を何回も計算しないようにするためのキャッシュとして機能する．関数 get_prob は BDD ノードに対応する変数の発生確率を取得するための関数．is_zero, is_one は，それぞれ 0-終端，1-終端ノードかどうかを判別する関数，get_one, get_zero はそれぞれ，1-枝，0-枝の子ノードを取得するための関数である．

3.3 BDD による定性解析

静的モデルの構造関数の分析では，極小カットベクトル(Minimal Cut Vector)の導出が重要な役割を果たす．構造関数 $\varphi(x_1,\ldots,x_n)$ の入力値 $\boldsymbol{x}=(x_1,\ldots,x_n)$ を状態ベクトルと呼ぶことにすると，極小カットベクトルとは，$\varphi(\boldsymbol{x})=0$ となる状態ベクトルのうち，任意の x_i の値が 0 から 1 に変化した新しい状態ベクトル $\boldsymbol{x}'$ が $\varphi(\boldsymbol{x}')=1$ となるような状態ベクトル $\boldsymbol{x}$ を示す．また，極小カットベクトルの集合を極小カット集合(MCS: Minimal Cut Set)と呼ぶ．これは，現在正常に稼働している部品の一つでも故障すると，システムが故障するような部品の状態を意味している．FT が AND, OR, k-out-of-n ゲートで構成されるとき，このシステムは単調，つまり，部品の故障によってシステム状態がより故障しにくい状態には移行しない性質を持つ．システムが単調であるとき，MCS と構造関数は 1 対 1 に対応するため，通常の FT に対して MCS は一意に決定される．例えば，式(3.3)のシステムの場合，MCS は $\{(0,1),(1,0)\}$ となり，式(3.4)の場合は $\{(0,0)\}$ となる．また，式(3.5)の場合は，$\{(0,0,1),(0,1,0),(1,0,0)\}$ となる．このような，MCS の導出は信頼性の定性的な分析において重要な役割を果たす．

FT が繰り返し事象を含む場合，MCS の導出は容易ではない．ここでは，BDD を用いた MCS の導出アルゴリズムの紹介を行う（Rauzy, 1993）．基本的なアイデアは構造関数を表す BDD から，MCS の集合を表す BDD を再構成する．シャノン分解によって構造関数が

$$\varphi(x_1,\ldots,x_n) = (x_1 \wedge \varphi(1,x_2,\ldots,x_n)) \vee (\overline{x_1} \wedge \varphi(0,x_2,\ldots,x_n)) \tag{3.11}$$

のように分解されることを思い出す．システムが単調であることを仮定すると，$\varphi(0,x_2,\ldots,x_n)$ の $(x_2,\ldots,x_n)$ に対する MCS は $x_1 = 0$ とすることで $\varphi(x_1,\ldots,x_n)$ の MCS になっていることが容易にわかる．一方，$\varphi(1,x_2,\ldots,x_n)$ の $(x_2,\ldots,x_n)$ に対する MCS を求めたとき，$\varphi(0,x_2,\ldots,x_n)$ の $(x_2,\ldots,x_n)$ に対する MCS と同じ集合があった場合，明らかに MCS ではない．そのため，$\varphi(1,x_2,\ldots,x_n)$ の MCS から $\varphi(0,x_2,\ldots,x_n)$ の MCS を除いた集合を求め，その集合要素に対して $x_1 = 1$ を付け加えることで $\varphi(1,\ldots,x_n)$ の MCS を求めることができる．

以下の Algorithm 2 は，上記の考えに基づいて構造関数の BDD 表現から MCS を導出するアルゴリズムである（Rauzy, 1993）．

Algorithm 2 MCS.

```
function minsol(node)
  return node if is_zero(node) || is_one(node)
  x = minsol(get_zero(node))
  z = get_one(node) - x # set difference
  y = minsol(z)
  return create_node(x, y)
end
```

関数 `minsol` は再帰的に BDD の 0-枝と 1-枝を辿るようにして実行される．まず 0-枝のノードに対して `minsol` を実行し，MCS を得る．一方，1-枝については 0-枝に含まれる MCS を除いたノードに対して `minsol` を実行して MCS を取得し，最終的に，ぞれぞれ 0,1 の値を決定した際に得られた MCS を子ノードとするノードを `create_node` 関数を使って作成している．アルゴリズム内の `is_zero`, `is_one` は，それぞれ 0-終端，1-終端ノードかどうかを判別する関数．関数 `get_one` は 1-枝の子ノードを取得する．また演算子-は BDD 上の集合の差の演算を行う．

4. マルコフ解析

4.1 モデル表現

FT では原因と事象の関係を静的にモデル化している．そのため，時系列的な関係（例えば，Event 1 → Event 2 の順で発生したときだけ Event S が発生する）を表現することができない．そこで，より一般的なシステムの動特性を表現するためにマルコフ連鎖モデルがよく用いられる（Trivedi and Bobbio, 2017）．マルコフ連鎖とは離散状態の確率過程であり，将来の確率法則が現在の状態のみに依存して決定する性質をもつ．

マルコフ連鎖は離散時間あるいは連続時間の確率過程として定義することができるが，本稿では連続時間マルコフ連鎖を考える．いま，システムの状態が $\mathcal{S} = \{1,2,\ldots,n\}$ の離散状態で表現されているものとする．時刻 t におけるシステム状態を表すため，状態空間 $\mathcal{S}$ 上の連続時間マルコフ連鎖 $\{X(t);\ t \geq 0\}$ を考える．時刻 t における状態確率ベクトル（行ベクトル）を $\boldsymbol{x}(t) = [P(X(t) = i)]_{i=1,\ldots,n}$ として定義する．時刻 0 における状態確率ベクトル $\boldsymbol{x}(0)$ が与えられているもとで，任意時刻 t の状態確率ベクトル $\boldsymbol{x}(t)$ は次の微分方程式を解くことによって得られる．

$$\frac{d}{dt}\boldsymbol{x}(t) = \boldsymbol{x}(t)\boldsymbol{Q} \tag{4.1}$$

ここで $\boldsymbol{Q}$ は無限小生成行列と呼ばれ，(i,j) 要素が状態 i から状態 j への推移率で構成される正方行列である．

システムの動特性に基づいて信頼性尺度を算出する場合，システムの各状態に対する報酬ベクトルを与えることが多い．列ベクトル $\boldsymbol{r}$ の i 番目の要素 $[\boldsymbol{r}]_i$ に対して次の定義を行う．

$$[\boldsymbol{r}]_i = \begin{cases} 1 & \text{システムが故障している} \\ 0 & \text{システムが稼働している} \end{cases} \tag{4.2}$$

この時，時刻 t でシステムが故障している確率 $F_S(t)$ は

$$F_S(t) = \boldsymbol{x}(t)\boldsymbol{r} \tag{4.3}$$

となる．一方，無限時間システムを稼働させた場合の状態確率ベクトルは，ある一定の条件のもと，次の方程式を満足する定常分布 $\boldsymbol{\pi}$ で与えられる．

$$\boldsymbol{\pi}\boldsymbol{Q} = \boldsymbol{0}, \quad \boldsymbol{\pi}\boldsymbol{1} = 1. \tag{4.4}$$

ここで $\boldsymbol{1}$ は全ての要素が 1 の列ベクトル，$\boldsymbol{0}$ は全ての要素が 0 の列ベクトルである．故障確率と同様に，報酬ベクトル $\boldsymbol{r}$ を用いると，システムが定常状態において故障している確率は

$$F_S = \boldsymbol{\pi}\boldsymbol{r} \tag{4.5}$$

で与えられる．

4.2 計算アルゴリズム

マルコフ連鎖モデルでは，状態確率ベクトルの算出が必要となる．一般に状態確率ベクトルの計算は $\boldsymbol{x}(t)$ を求める計算（過渡解）と $\boldsymbol{\pi}$ を求める計算（定常解）で区別される．過渡解を求める計算は式(4.1)の常微分方程式を数値的に解く方法と，式(4.1)から得られる以下の行列指数表現から算出する手法がある．

$$\boldsymbol{x}(t) = \boldsymbol{x}(0)\exp(\boldsymbol{Q}t). \tag{4.6}$$

ここで exp は行列指数関数であり，任意の正方行列 $\boldsymbol{A}$ に対して

$$\exp(\boldsymbol{A}) = \boldsymbol{I} + \boldsymbol{A} + \frac{\boldsymbol{A}^2}{2!} + \cdots = \sum_{u=0}^{\infty} \frac{\boldsymbol{A}^u}{u!} \tag{4.7}$$

と定義される．ここで $\boldsymbol{I}$ は単位行列である．

行列指数関数の計算は一般に数値的に困難であるが，マルコフ連鎖の場合は一様化と呼ばれる効率的な計算手法が用いられる（Stewart, 1994）．一様化では，$\boldsymbol{P} = \boldsymbol{I} + \boldsymbol{Q}/q$ で得られる離散時間マルコフ連鎖の推移確率行列 $\boldsymbol{P}$ を用いる．ここで q は $\boldsymbol{Q}$ の対角要素の絶対値の最大値である．このとき式(4.7)を使うと，式(4.6)は

$$\boldsymbol{x}(t) = \boldsymbol{x}(0)\sum_{u=0}^{\infty} \frac{(qt)^u}{u!} e^{-qt}\boldsymbol{P}^u \tag{4.8}$$

となる．許容誤差 ε に対して以下を満たす $r_{\max}$ を定義する．

$$\sum_{u=0}^{r_{\max}} \frac{(qt)^u}{u!} e^{-qt} \geq 1 - \varepsilon. \tag{4.9}$$

ここで，$r_{\max}$ はパラメータ qt のポアソン分布の分位点となっていることに注意する．このとき，過渡解は

$$\boldsymbol{x}(t) \approx \tilde{\boldsymbol{x}}(t) = \boldsymbol{x}(0) \sum_{u=0}^{r_{\max}} \frac{(qt)^u}{u!} e^{-qt} \boldsymbol{P}^u \tag{4.10}$$

として計算され，確率の性質から $\|\boldsymbol{x}(t) - \tilde{\boldsymbol{x}}(t)\|_1 \leq \varepsilon$ が成り立つ．ここで $\|\cdot\|_1$ はベクトルの 1 ノルムを表す．

一方，定常解は式(4.4)の連立方程式を解くことによって得られる．無限小生成行列 $\boldsymbol{Q}$ は疎行列であることが多いため，Gauss-Seidel 法のような繰返し手法が使われる．具体的に無限小生成行列が上三角行列 $\boldsymbol{U}$，下三角行列 $\boldsymbol{L}$，対角行列 $\boldsymbol{D}$ によって次のように分解されるものとする．

$$\boldsymbol{Q} = \boldsymbol{U} + \boldsymbol{L} - \boldsymbol{D}. \tag{4.11}$$

これらの準備のもと，$\boldsymbol{\pi}$ が収束するまで次の更新を繰り返す．

$$\boldsymbol{\pi} \leftarrow \frac{\boldsymbol{\pi}\boldsymbol{U}(\boldsymbol{D} - \boldsymbol{L})^{-1}}{\boldsymbol{\pi}\boldsymbol{U}(\boldsymbol{D} - \boldsymbol{L})^{-1}\mathbf{1}}. \tag{4.12}$$

通常の Gauss-Seidel 法とは異なり，$\boldsymbol{\pi}\mathbf{1} = 1$ を保証するため正規化定数で更新値を割っていることに注意する（Bolch et al., 2006）．

5. 統計解析

5.1 故障分布

信頼性分野における統計解析は故障分布(寿命分布)の推定を行うことが主な目的になる．一般的に統計解析における推定はパラメトリックな方法とノンパラメトリックな方法がある．寿命分布の推定においても双方用いられるが，信頼性分野ではパラメトリックな方法がよく用いられる．寿命分布の推定は，信頼性試験やフィールドデータから得られた寿命データを用いて行われる．つまり，故障時間のデータが必要となるが，故障の観測がそもそも希な事象であり，十分なサンプルが得られないことも多い，そのため，十分なサンプル数を必要とするノンパラメトリックな方法は信頼性分野ではあまり用いられない．さらに，指標として平均故障時間などの要約統計量を求めることが多いため，パラメトリックな方法が適している．典型的に信頼性工学において故障分布として用いられるものを表 1 に示す（Meeker and Escobar, 1998）．

5.2 相型分布

ここでは表 1 に示した故障分布とは異なる枠組みの連続時間マルコフ連鎖に基づく相型分布について説明する．相型分布(Phase-Type Distribution, PH 分布)は，マルコフ過程の吸収時間として定義される確率分布である（Neuts, 1981）．この分布は，指数分布やその一般化として，さまざまな確率分布を近似する柔軟性を持ち，信頼性工学をはじめ多くの分野で利用されている．いま，m 個の過渡状態と 1 つの吸収状態を持つ連続時間マルコフ連鎖を考え，その無限小生成行列を $\boldsymbol{Q}$ とする．このとき，$\boldsymbol{Q}$ は次のように表される．

$$\boldsymbol{Q} = \begin{pmatrix} \boldsymbol{T} & \boldsymbol{\tau} \\ \mathbf{0} & 0 \end{pmatrix}. \tag{5.1}$$

ここで，$\boldsymbol{T}$ は $m \times m$ の行列であり，$\boldsymbol{\tau}$ は m 次元の列ベクトルである．このとき，相型分布の確率密度関数および分布関数は次のように表される．

表 1. 信頼性工学における主な故障分布.

分布名	特徴・用途
指数分布(Exponential)	故障率が一定．偶発故障のモデル化に適用.
ガンマ分布(Gamma)	段階的に進む劣化による故障をモデル化.
切断正規分布(Truncated Normal)	非負の寿命データをモデル化.
対数正規分布(Log-Normal)	故障時間が非対称分布を示す場合に適用.
切断ロジスティック分布(Truncated Logistic)	寿命データが非負かつ裾が重い場合に利用.
対数ロジスティック分布(Log-Logistic)	故障率が増減する場合や裾が重い分布に適用.
極値分布(ワイブル分布など) (Extreme Value/Weibull)	初期不良，偶発故障，経年劣化など広範な故障現象をモデル化可能.

$$f_{PH}(t) = \boldsymbol{\alpha} \exp(\boldsymbol{T}t)\boldsymbol{\tau}, \quad t \geq 0. \tag{5.2}$$

$$F_{PH}(t) = 1 - \boldsymbol{\alpha} \exp(\boldsymbol{T}t)\mathbf{1}, \quad t \geq 0. \tag{5.3}$$

ここで，$\boldsymbol{\alpha}$ は時刻 0 における状態を定義する確率ベクトル(行ベクトル)である．相型分布のパラメータは $(\boldsymbol{\alpha}, \boldsymbol{T}, \boldsymbol{\tau})$ であり，これらのパラメータを適切に選ぶことで，さまざまな確率分布を表現することができる．実際には $\boldsymbol{\tau} = -\boldsymbol{T}\mathbf{1}$ であるため，$(\boldsymbol{\alpha}, \boldsymbol{T})$ で十分であるが，ここでは記号の便宜上 $(\boldsymbol{\alpha}, \boldsymbol{T}, \boldsymbol{\tau})$ とする．特に，一つの過渡状態 $m = 1$ とした場合，相型分布は指数分布と一致する．

相型分布の重要な特性として閉包性が挙げられる（Maier and O'Cinneide, 1992）．これは，畳み込み，混合，最小値，最大値といった操作を施しても相型分布としての性質が保たれることを意味する．PH 分布の閉包性を具体例で示す．独立な相型分布に従う確率変数 X_A と X_B が，それぞれパラメータ $(\boldsymbol{\alpha}_A, \boldsymbol{T}_A, \boldsymbol{\tau}_A)$ および $(\boldsymbol{\alpha}_B, \boldsymbol{T}_B, \boldsymbol{\tau}_B)$ を持つとする．このとき，次の操作によって生成される確率変数 X_C も相型分布に従う．ただし，以下では記号の便宜上 $\mathbf{0}, \boldsymbol{O}$ は適切な次元の零ベクトルおよび零行列を表す.

(i) 畳み込み(Convolution)：$X_C = X_A + X_B$.

$$\boldsymbol{\alpha}_C = \begin{pmatrix} \boldsymbol{\alpha}_A & \mathbf{0} \end{pmatrix}, \quad \boldsymbol{T}_C = \begin{pmatrix} \boldsymbol{T}_A & \boldsymbol{\tau}_A\boldsymbol{\alpha}_B \\ \boldsymbol{O} & \boldsymbol{T}_B \end{pmatrix}, \quad \boldsymbol{\tau}_C = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\tau}_B \end{pmatrix}. \tag{5.4}$$

(ii) 混合(Mixture)：$X_C = \chi X_A + (1-\chi)X_B$．ただし，$\chi$ は確率 p で X_A を選ぶ指標確率変数.

$$\boldsymbol{\alpha}_C = \begin{pmatrix} p\boldsymbol{\alpha}_A & (1-p)\boldsymbol{\alpha}_B \end{pmatrix}, \quad \boldsymbol{T}_C = \begin{pmatrix} \boldsymbol{T}_A & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{T}_B \end{pmatrix}, \quad \boldsymbol{\tau}_C = \begin{pmatrix} \boldsymbol{\tau}_A \\ \boldsymbol{\tau}_B \end{pmatrix}. \tag{5.5}$$

(iii) 最小値(Minimum)：$X_C = \min(X_A, X_B)$.

$$\boldsymbol{\alpha}_C = \boldsymbol{\alpha}_A \otimes \boldsymbol{\alpha}_B, \quad \boldsymbol{T}_C = \boldsymbol{T}_A \oplus \boldsymbol{T}_B, \quad \boldsymbol{\tau}_C = \boldsymbol{\tau}_A \otimes \mathbf{1} + \mathbf{1} \otimes \boldsymbol{\tau}_B. \tag{5.6}$$

ただし，$\otimes$ はクロネッカー積，$\oplus$ はクロネッカー和を表す.

(iv) 最大値(Maximum)：$X_C = \max(X_A, X_B)$.

$$\boldsymbol{\alpha}_C = \begin{pmatrix} \boldsymbol{\alpha}_A \otimes \boldsymbol{\alpha}_B & \mathbf{0} & \mathbf{0} \end{pmatrix}, \quad \boldsymbol{T}_C = \begin{pmatrix} \boldsymbol{T}_A \oplus \boldsymbol{T}_B & \boldsymbol{\tau}_A \otimes \boldsymbol{I} & \boldsymbol{I} \otimes \boldsymbol{\tau}_B \\ \boldsymbol{O} & \boldsymbol{T}_B & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{T}_A \end{pmatrix}, \quad \boldsymbol{\tau}_C = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\tau}_B \\ \boldsymbol{\tau}_A \end{pmatrix}. \tag{5.7}$$

FT を使ったモデリングにおいて，AND ゲートで上位イベントが発生するためには，すべての下位イベントが発生する必要がある．つまり，上位イベントの発生時刻は下位イベントの最大値で表される．逆に，OR ゲートで上位イベントが発生するためには，いずれかの下位イベントが発生すればよいため，OR ゲートにおいて上位イベントが発生する時刻は下位イベントの最小値で表される．換言すると，FT において各原因事象の発生時間(各部品の故障時間)が相型分布に従う場合，上位イベントの発生時間も相型分布に従う．このような性質から，システム故障時間を相型分布で表現することの妥当性が示唆される．

相型分布は基礎となるマルコフ連鎖の構造により循環型と非循環型に分類される．マルコフ連鎖の状態遷移で，すべての過渡状態に高々 1 回しか到達しない場合を循環型，2 回以上到達する可能性がある場合を非循環型と呼ぶ．明らかに，循環型は非循環型の特別な場合であるため，分布の表現能力としては非循環型の方が高い．一方で，位相数(マルコフ連鎖の過渡状態数)を十分大きく取ることで，循環型は非循環型で近似できる．この性質から非循環型の相型分布は，理論的に任意の確率分布を任意の精度で近似できることが知られている（Asmussen and Koole, 1993）．さらに，先の畳み込み，混合，最小値，最大値の操作において，X_A, X_B が非循環型の相型分布であるならば明らかに X_C も非循環型の相型分布となることがわかる．

非循環型の相型分布に関する性質として，あらゆる非循環型の相型分布は標準形に変換することができることが知られている（Cumani, 1982）．以下は標準形の一つである CF1(Canonical Form 1)と呼ばれる形式である．

$$\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m), \tag{5.8}$$

$$\boldsymbol{T} = \begin{pmatrix} -\lambda_1 & \lambda_1 & & & \boldsymbol{O} \\ & -\lambda_2 & \lambda_2 & & \\ & & \ddots & \ddots & \\ & & & -\lambda_{m-1} & \lambda_{m-1} \\ \boldsymbol{O} & & & & -\lambda_m \end{pmatrix}, \quad \boldsymbol{\tau} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \lambda_m \end{pmatrix}. \tag{5.9}$$

ここで，α_i は状態 i に初期状態がある確率，λ_i は状態 i から状態 $i+1$ への遷移率を表し，$\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_m$ である．CF1 以外にも，Cox 分布型の標準形もある．これらの標準形は，同じ位相数を持つあらゆる非循環型の相型分布に対してパラメータ数が最小であることが示されている．具体的に CF1 の場合は，$\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ とすると，$(\boldsymbol{\alpha}, \boldsymbol{\lambda})$ の $2m$ 個のパラメータで表現され，これが位相数 m の非循環型の相型分布の最小パラメータ数となっている．ここで，$\boldsymbol{\alpha}\mathbf{1} = 1$ であるためパラメータ自由度は $2m-1$ であることに注意する．

このような性質から，FT モデルの故障分布が非循環型の相型分布で表現され，あらゆる非循環型が標準形に変換されることを考えると，故障データに基づいて $2m$ 個の標準形パラメータを推定することは，確率モデリングの観点からも妥当な故障分布推定が実施できていることがわかる．

5.3 パラメータ推定

統計解析による信頼性評価では，データから故障分布のパラメータを推定する必要がある．故障分布推定では，信頼性試験やフィールドデータから得られた故障データを用いるが，故障そのものが希な事象であるため信頼性試験やフィールドデータでは有限な時間で観測を打ち切った右打ち切りデータを扱うことが多い．

次のデータ構造を持つ打ち切りデータを考える．

$$\mathcal{D} = \{(t_1, \delta_1), (t_2, \delta_2), \ldots, (t_N, \delta_N)\}. \tag{5.10}$$

ここで t_i は観測された故障時刻，δ_i は打ち切り指示を表す指標である．$\delta_i = 0$ の場合，時刻 t_i で故障が観測されたことを意味し，$\delta_i = 1$ の場合，時刻 t_i で故障が観測されないまま試験が終わったことを意味する．このデータに対する故障分布の最尤推定を考える．以下は上記の $\mathcal{D}$ に対する尤度関数である．

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{N} f(t_i;\boldsymbol{\theta})^{\delta_i} R(t_i;\boldsymbol{\theta})^{1-\delta_i}. \tag{5.11}$$

ここで，$R(t;\boldsymbol{\theta})$ は信頼度関数に対応し，$f(t;\boldsymbol{\theta}) = -dR(t;\boldsymbol{\theta})/dt$ は密度関数である．$\boldsymbol{\theta}$ はパラメータベクトルである．最尤推定法では上記の尤度を最大にするパラメータ $\boldsymbol{\theta}$ を求める．

一方，相型分布を故障分布として信頼性解析を行う場合，分布パラメータが非常に多く上記の尤度あるいは対数尤度を直接最大にするパラメータを見つけることが難しい．そのため EM アルゴリズムを用いたアルゴリズムが提案されている（Asmussen et al., 1996; Olsson, 1996）．EM アルゴリズムは，不完全データに対する最尤推定手法である．不完全データとは，観測されたデータと観測されなかったデータの両方を含むデータであり，EM アルゴリズムは，観測されたデータからパラメータベクトルを推定するための反復的な手法である（Dempster et al., 1977; Wu, 1983）．ここで，$\mathcal{D}$ と $\mathcal{U}$ を観測されたデータと観測されなかったデータとする．EM アルゴリズムは E ステップと M ステップからなる．E ステップでは，事後分布 $p(\mathcal{U}|\mathcal{D};\boldsymbol{\theta}')$ に基づく期待対数尤度 $Q(\boldsymbol{\theta}|\boldsymbol{\theta}')$ を計算する．M ステップでは，期待対数尤度 $Q(\boldsymbol{\theta}|\boldsymbol{\theta}')$ を最大化するようなパラメータベクトル $\boldsymbol{\theta}$ を求める．これらのステップをパラメータベクトルが収束するまで繰り返す．

Olsson（1996）で与えられた打ち切りデータを用いた相型分布に対する EM アルゴリズムの式を示す．以下の確率変数を定義する．

- $T^{(k)}$: k 番目のデータの故障時刻．

実際のデータは打ち切りデータであるため，すべてのサンプルに対して故障時刻が観測されているとは限らないことに注意する．一方，相型分布を支配するマルコフ連鎖の状態が観測されていないため，次の未観測データ（確率変数）を導入する．

- $B_i^{(k)}$: $T^{(k)}$ のマルコフ連鎖が状態 i で開始したかどうか表す指標確率変数．
- $Z_i^{(k)}$: $T^{(k)}$ のマルコフ連鎖が吸収状態に到達するまでに状態 i に滞在する時間．
- $Y_i^{(k)}$: $T^{(k)}$ のマルコフ連鎖が吸収状態に到達する直前に状態 i に滞在するかどうか表す指標確率変数．
- $N_{i,j}^{(k)}$: $T^{(k)}$ のマルコフ連鎖が状態 i から状態 j に遷移する回数．

このとき，EM アルゴリズムの M ステップは以下のように表される．

$$\alpha_i \leftarrow \frac{\sum_{k=1}^{N} \mathrm{E}[B_i^{(k)} \mid \mathcal{D}_k;\boldsymbol{\theta}]}{N}, \quad i = 1,\ldots,m, \tag{5.12}$$

$$\tau_i \leftarrow \frac{\sum_{k=1}^{N} \mathrm{E}[Y_i^{(k)} \mid \mathcal{D}_k;\boldsymbol{\theta}]}{\sum_{k=1}^{N} \mathrm{E}[Z_i^{(k)} \mid \mathcal{D}_k;\boldsymbol{\theta}]}, \quad i = 1,\ldots,m, \tag{5.13}$$

$$\lambda_{i,j} \leftarrow \frac{\sum_{k=1}^{N} \mathrm{E}[N_{i,j}^{(k)} \mid \mathcal{D}_k;\boldsymbol{\theta}]}{\sum_{k=1}^{N} \mathrm{E}[Z_i^{(k)} \mid \mathcal{D}_k;\boldsymbol{\theta}]}, \quad i = 1,\ldots,m, \quad j = 1,\ldots,m, \quad i \neq j. \tag{5.14}$$

ここで $\mathcal{D}_k = (t_k, \delta_k)$ である．

E ステップにおける期待値を計算するため以下のベクトルならびに行列を定義する．

$$\boldsymbol{f}(t) = \boldsymbol{\alpha}\exp(\boldsymbol{T}t), \tag{5.15}$$

$$\overline{\boldsymbol{f}}(t) = \boldsymbol{\alpha}(-\boldsymbol{T})^{-1}\exp(\boldsymbol{T}t), \tag{5.16}$$

$$\boldsymbol{b}(t) = \exp(\boldsymbol{T}t)\boldsymbol{\tau}, \tag{5.17}$$

$$\overline{\boldsymbol{b}}(t) = \exp(\boldsymbol{T}t)\mathbf{1}, \tag{5.18}$$

$$\boldsymbol{H}(t) = \int_0^t \exp(\boldsymbol{T}s)\boldsymbol{\tau}\boldsymbol{\alpha}\exp(\boldsymbol{T}(t-s))du, \tag{5.19}$$

$$\overline{\boldsymbol{H}}(t) = \int_0^t \exp(\boldsymbol{T}s)\boldsymbol{\tau}\boldsymbol{\alpha}(-\boldsymbol{T})^{-1}\exp(\boldsymbol{T}(t-s))ds. \tag{5.20}$$

このとき，未観測データに対する各期待値は以下のように計算される．

$$\mathrm{E}[B_i^{(k)}|\mathcal{D}_k;\boldsymbol{\theta}] = \delta_k\frac{\alpha_i[\boldsymbol{b}(t_k)]_i}{\boldsymbol{\alpha}\boldsymbol{b}(t_k)} + (1-\delta_k)\frac{\alpha_i[\overline{\boldsymbol{b}}(t_k)]_i}{\boldsymbol{\alpha}\overline{\boldsymbol{b}}(t_k)}, \tag{5.21}$$

$$\mathrm{E}[Y_i^{(k,l)}|\mathcal{D}_k;\boldsymbol{\theta}] = \delta_k\frac{[\boldsymbol{f}(t_k)]_i\tau_i}{\boldsymbol{\alpha}\boldsymbol{b}(t_k)} + (1-\delta_k)\frac{[\overline{\boldsymbol{f}}(t_k)]_i\tau_i}{\boldsymbol{\alpha}\overline{\boldsymbol{b}}(t_k)}, \tag{5.22}$$

$$\mathrm{E}[Z_i^{(k)}|\mathcal{D}_k;\boldsymbol{\theta}] = \delta_k\frac{[\boldsymbol{H}(t_k)]_{i,i}}{\boldsymbol{\alpha}\boldsymbol{b}(t_k)} + (1-\delta_k)\frac{[\overline{\boldsymbol{b}}(t_k)\overline{\boldsymbol{f}}(0) + \overline{\boldsymbol{H}}(t_k)]_{i,i}}{\boldsymbol{\alpha}\overline{\boldsymbol{b}}(t_k)}, \tag{5.23}$$

$$\mathrm{E}[N_{i,j}^{(k)}|\mathcal{D}_k;\boldsymbol{\theta}] = \delta_k\frac{\lambda_{i,j}[\boldsymbol{H}(t_k)]_{j,i}}{\boldsymbol{\alpha}\boldsymbol{b}(t_k)} + (1-\delta_k)\frac{\lambda_{i,j}[\overline{\boldsymbol{b}}(t_k)\overline{\boldsymbol{f}}(0) + \overline{\boldsymbol{H}}(t_k)]_{j,i}}{\boldsymbol{\alpha}\overline{\boldsymbol{b}}(t_k)}. \tag{5.24}$$

上記の $\boldsymbol{H}(t)$ ならびに $\overline{\boldsymbol{H}}(t)$ は畳み込み積分が必要となるため計算コストが高い．Okamura et al.（2011, 2013）では一様化を適用した効率的な計算法を提案している．そのアルゴリズムでは，非循環型の相型分布に対して EM ステップの計算量が $O(mK)$ となっており，より大きな位相数の分布推定が可能となっている．

6. 信頼性解析ツール

6.1 FT ツール

FT は信頼性解析や安全性評価で頻繁に使用されており，商用およびフリーのソフトウェアがいくつか存在する．特に，機能安全規格（例：ISO 26262，IEC 61508）では，FTA（Fault Tree Analysis）を安全性分析の手法として推奨しており，一部の商用ソフトウェアはこれらの規格に対応している．商用ソフトウェアとしては，Reliability Workbench や RiskSpectrum によるソフトウェア群などが挙げられる．これらは製品開発における利用を想定しているため高い性能を持つ代わりにライセンス料が高い．

FT の研究においては，オープンソースやフリーソフトウェアの利用が想定される．フリーソフトウェアとしては，OpenFTA（`https://github.com/luyangshang/OpenFTA`）が広く知られている．OpenFTA は，静的 FT の構築と解析を支援するオープンソースのツールであり，定性的および定量的な解析を提供している．広島大学のグループは Julia 言語をベースとしたオープンソースの FT 解析ツール FaultTree.jl（`https://github.com/JuliaReliab/FaultTree.jl`）を開発しており，OpenFTA と同様に定量・定性解析を提供し，Jupyter Notebook 上の IJulia カーネルで利用することができるため，非常に柔軟な利用方法が可能である．

FT による解析例として，61 個の部品と 84 個のゲートからなるモデルを用いる（Rauzy, 1993）．実際に Julia 言語による記述例を付録 A に示す．表 2 は，FaultTree.jl により，BDD を構築する時間（BDD creation），MCS を導出する時間（MCS extraction），信頼度を算出する時間（Reliability）を計測した結果である．なお数値例に利用した環境は Apple M2, 16GB memory

表 2. FT 解析の計算時間.

	計算時間(秒)
BDD creation	1.07
MCS extraction	3.08
Reliability	0.43

である.

BDD の構築について，対象モデルは 546,399 個のノードと 1,092,794 個の枝から構成される BDD を構築した．モデルの規模としては非常に大きいが，BDD の作成については，1 秒程度となっており，実用的に十分な速度を有している．FT から BDD の構築には BDD に対する Apply 演算と呼ばれるアルゴリズムを利用するが，これが有効に機能したものと考えられる．次に MCS の算出については，3 秒程度の時間がかかった．実際に MCS のための BDD は 11,318 個のノードと 22,632 個の枝から構成され，MCS の要素は 46,188 個となった．こちらについてもかなり大規模であるが，実用的な時間での分析が行えていることがわかる．定量的な指標である信頼度については 0.5 秒程度とかなり速く計算が行えている．

6.2 マルコフ解析ツール

マルコフ連鎖モデルに対する解析は行列操作であるため MATLAB や Python の Numpy などの汎用的な数値計算ライブラリが利用されることが多い．一方で, マルコフ連鎖解析でよく使われる一様化手法や定常分布の計算に特化したツールやライブラリも存在する．これらのツールは, 特にマルコフ連鎖に関連する大規模な行列操作や, 状態遷移に伴う複雑な確率分布の計算を効率的に実行できるように設計されている．例えば, SHARPE (`https://sharpe.pratt.duke.edu`) はマルコフ連鎖モデルだけでなく，FT や信頼性ブロック図も含む包括的な信頼性解析ツールである．また，PRISM (`https://www.prismmodelchecker.org`) は確率モデル検査のためのツールで，マルコフ連鎖や確率的モデルの解析に広く使用されている．Julia 言語を用いたものでは MarkovChain.jl (`https://github.com/wangnangg/MarkovChains.jl.git`) や，広島大学グループで開発した NMarkov.jl (`https://github.com/JuliaReliab/NMarkov.jl.git`) がある，これはパラメータ感度解析を行うこともできる．

マルコフ連鎖モデルに基づく信頼性解析の例として，図 4 に示すような生産システムを考える．システムは 4 つのステーションからなり，各ステーションでは与えられた部品を加工して次のステーションに送る．各ステーションの容量(処理中＋バッファで加工待ちの部品数)は 5 とし，送り先のステーションの容量がいっぱいの場合は処理を中断して送り先の容量が空くのを待つこととする．ステーション 1 で加工された部品は，ステーション 2 とステーション 3 へ分解されて送られ，ステーション 2 とステーション 3 で加工された部品はステーション 4 に送られる．ステーション 4 ではステーション 2 とステーション 3 で加工された部品を組み立て出荷する．一方で，ステーション 2 とステーション 3 で加工された部品は検査され，不良品がある場合は廃棄される．ステーション 2 とステーション 3 で加工される部品はペアで利用されるため，一方で不良品がある場合，部品のバランスがおかしくなる．一方で大量の不良品が発生すると，ステーション 2 あるいはステーション 3 の容量がいっぱいになり，生産が中断される(デッドロックが起こる)．このシステムの信頼性としてデッドロックが発生する確率をマルコフ連鎖モデルにより解析する．

デッドロックの解析は，特定のイベントの順序が発生要因となる場合がある．つまり，時間にともなう状態変化を正確に捉える必要があるため，単純な FT による分析では対応できず，

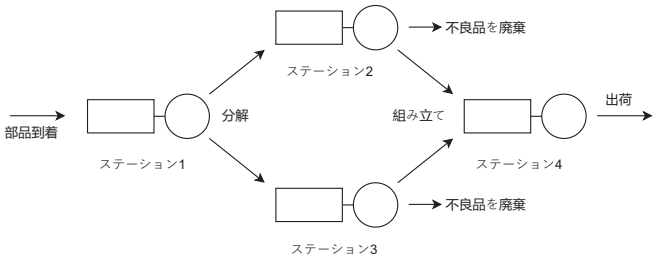


図 4. Fork-Join 型生産システム.

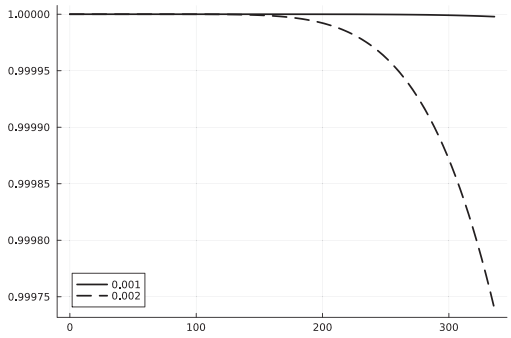


図 5. マルコフ解析による信頼度関数.

マルコフ連鎖によるモデル化および解析を必要とする．このような例としては，そのほかにも通信における輻輳解析や，ソフトウェアのバグ解析などが挙げられる．一方，FT は航空システムや電力システムなど，主に個々の部品の故障が独立してシステム故障に影響するシステムの解析を，より少ない計算量で実施できるため適している．

図 5 はステーション 1 への部品の到着が $\lambda = 5.0$ [1/hour]，全ステーションの処理率がすべて一律で $\mu = 10.0$ [1/hour]，ステーション 2 とステーション 3 の不良率が $f = 1/1000$ あるいは $f = 2/1000$ とした時の信頼度関数，つまり 366 時間(24 時間 × 14 日間)稼働させたときにデッドロックの発生する確率を NMarkov.jl パッケージを使って計算した結果を示している．このような評価を行うことで，不良率 $f = 1/1000$ 程度であれば，デッドロックが発生するのは希であり，実用上ほとんど問題ないが $f = 2/1000$ であれば，デッドロックの発生に対する対策が必要であることがわかる．

生産システムに対するマルコフ連鎖モデルの状態数は 7,776 で，NMarkov.jl によって生産システムの信頼度関数を計算するのに要した時間は $f = 1/1000$ の場合で 1.56 秒，$f = 2/1000$ の場合で 1.57 秒であった．ここで，生産システムに対するマルコフ連鎖モデルの無限小生成行列は確率ペトリネットツール gospn (`https://github.com/JuliaReliab/gospn`) によって生成された．数千状態という中規模モデルであるが，いずれの場合も計算時間が 2 秒以下と言う実用的な時間で解析が行えている．

6.3 統計解析ツール

データから分布パラメータを推定するための統計解析ツールとしては，R や Python の Scipy などの汎用的な統計解析ライブラリが利用されることが多い．特に，R の survival パッケー

表 3. 相型分布のパラメータ推定の計算時間.

位相数	計算時間(秒)
10	0.523
20	0.954
30	1.458
40	2.842
50	3.680
60	4.821
70	5.969
80	7.209
90	8.494
100	9.855

ジを利用すると，打ち切りデータから典型的な指数分布，ガンマ分布，ワイブル分布などのパラメータ推定を行うことができる．一方で，相型分布に特化した統計解析ツールも存在する．広島大学グループが開発した R の mapfit パッケージ(`https://github.com/okamumu/mapfit`)は，EM アルゴリズムに基づいた相型分布のパラメータ推定を実行することができる．現在，打ち切りデータに対する推定は censored ブランチにのみ実装されているが，今後のバージョンでメインブランチにマージされる予定である．

Olsson（1996）で使われている右打ち切りデータを用いた相型分布に対する EM アルゴリズムの実行例を示す．R での利用方法は以下のコマンドで利用できる．

```
# mapfit の censored ブランチのインストール
devtools::install_github("okamumu/mapfit", ref="censored")

# mapfit パッケージの読み込み
library(mapfit)

# x: 故障時刻，delta: 打ち切りかどうか
# 打ち切りデータ ... は適切なデータを入力する
data <- data.frame(x=c(....), delta=c(....))

# 100 フェーズの CF1 を指定して推定
result <- phfit.surv(ph = cf1(100),
                     x = data$x,
                     delta = data$delta)
# 推定されたモデル
result$model
```

表 3 は mapfit パッケージを用いて，右打ち切りデータ(サンプル数 100)から相型分布のパラメータを推定するのに要した時間を示している．なお，数値例に利用した環境は Apple M2, 16GB memory である．位相数が増加するにつれて計算時間が増加しているが，位相数が 100 の場合でも 10 秒程度で推定が完了しており，十分な速度で推定が行えることがわかる．

7. まとめと今後の課題

本稿では，信頼性解析として，FT 解析，マルコフ解析，相型分布を用いた統計解析の三つの手法を紹介し，それらを支援するツールの紹介を行った．大規模で複雑なシステムの信頼性解析ではこれらのツールによる支援が必要不可欠である．一方で，現在のツールでは超大規模なシステムに対する解析には限界がある．FT 解析においては，BDD の構築や MCS の抽出に

比較的多くの計算コストがかかるため近似手法も含めた解析手法とそれを実装したツールが求められる．また，マルコフ解析においては，対象となるシステムの状態爆発の問題がある．先の例で見たように比較的簡単なモデルであっても指数的に状態数が増加するため，マルコフモデルについては構築と計算の双方における効率化が求められる．相型分布に対する統計解析においては，大きな状態数を持つ相型分布に対しても十分な速度でパラメータ推定が行える．しかしながら一方で，過適合の問題があり，罰則項を導入するなどの対策が求められる．また，実務的な観点からは，これらのツールを統合化し，同じプラットフォームを通じて信頼性解析が行える環境の構築が求められる．

付録 A：FaultTree.jl による FT の記述例

```
using FaultTree
ft = FTree()

@repeated ft c[1:61]

g62 = c[1] & c[2]; g63 = c[1] & c[3]; g64 = c[1] & c[4]; g65 = c[1] & c[5];
g66 = c[1] & c[6]; g67 = c[1] & c[7]; g68 = c[1] & c[8]; g69 = c[1] & c[9];
g70 = g62 | c[10]; g71 = g63 | c[11]; g72 = g64 | c[12]; g73 = g65 | c[13];
g74 = g62 | c[14]; g75 = g63 | c[15]; g76 = g64 | c[16]; g77 = g65 | c[17];
g78 = g62 | c[18]; g79 = g63 | c[19]; g80 = g64 | c[20]; g81 = g65 | c[21];
g82 = g62 | c[22]; g83 = g63 | c[23]; g84 = g64 | c[24]; g85 = g65 | c[25];
g86 = g62 | c[26]; g87 = g63 | c[27]; g88 = g64 | c[28]; g89 = g65 | c[29];
g90 = g66 | c[30]; g91 = g68 | c[31]; g92 = g67 | c[32]; g93 = g69 | c[33];
g94 = g66 | c[34]; g95 = g68 | c[35]; g96 = g67 | c[36]; g97 = g69 | c[37];
g98 = g66 | c[38]; g99 = g68 | c[39]; g100 = g67 | c[40]; g101 = g69 | c[41];
g102 = g66 | c[42]; g103 = g68 | c[43]; g104 = g67 | c[44]; g105 = g69 | c[45];
g106 = ftkofn(ft, 3, g70, g71, g72, g73); g107 = ftkofn(ft, 3, g74, g75, g76, g77);
g108 = ftkofn(ft, 3, g78, g79, g80, g81); g109 = ftkofn(ft, 3, g82, g83, g84, g85);
g110 = ftkofn(ft, 3, g86, g87, g88, g89); g111 = ftkofn(ft, 3, g94, g95, g96, g97);
g112 = ftkofn(ft, 3, g98, g99, g100, g101); g113 = g90 & g92; g114 = g91 & g93;
g115 = g102 & g104; g116 = g103 & g105; g117 = g113 | c[46]; g118 = g114 | c[47];
g119 = g107 | g108 | c[52]; g120 = g109 | g110; g121 = g66 | g117 | c[48];
g122 = g68 | g118 | c[49]; g123 = g67 | g117 | c[50]; g124 = g69 | g118 | c[51];
g125 = ftkofn(ft, 2, g121, g123, g122, g124); g126 = g111 | g112 | g125 | c[53];
g127 = g115 & g120; g128 = g116 & g120; g129 = g62 | g127 | c[54];
g130 = g63 | g128 | c[55];
g131 = g64 | g127 | c[56]; g132 = g65 | g128 | c[57]; g133 = g62 | g129 | c[58];
g134 = g63 | g130 | c[59]; g135 = g64 | g131 | c[60]; g136 = g65 | g132 | c[61];
g137 = ftkofn(ft, 3, g133, g134, g135, g136); g138 = g106 | g119 | g137;
g139 = g62 | g66 | g117 | g129 | c[48]; g140 = g63 | g68 | g118 | g130 | c[49];
g141 = g64 | g67 | g117 | g131 | c[50]; g142 = g65 | g69 | g118 | g132 | c[51];
g143 = g139 & g140 & g141 & g142; g144 = g111 | g112 | g143 | c[53];
top = g126 & g138 & g144;

@parameters ft begin
  // setting parameters for c[1] to c[61]
  c[1] = 0.01
  ... 省略 ...
end

@time bdd = ftbdd!(ft, top)
@time begin
    result = mcs(ft, bdd)
    mincuts = extractpath(result)
end
@time fresult = prob(ft, bdd)
```

参 考 文 献

Asmussen, S. and Koole, G. (1993). Marked point processes as limits of Markovian arrival streams, *Journal of Applied Probability*, **30**, 365–372.

Asmussen, S., Nerman, O. and Olsson, M. (1996). Fitting phase-type distributions via the EM algorithm, *Scandinavian Journal of Statistics*, **23**(4), 419–441.

Barlow, R. E. and Proschan, F. (1965). *Mathematical Theory of Reliability*, John Wiley & Sons, New York.

Birnbaum, Z. W., Esary, J. D. and Saunders, S. C. (1961). Multi-component systems and their reliability, *Technometrics*, **3**(1), 55–77.

Bolch, G., Greiner, S., de Meer, H. and Trivedi, K. S. (2006). *Queueing Networks and Markov Chains*: *Modeling and Performance Evaluation with Computer Science Applications*, 2nd ed., John Wiley & Sons, New York.

Bryant, R. (1986). Graph-based algorithms for Boolean function manipulation, *IEEE Transactions on Computers*, **C-35**(8), 677–691.

Cumani, A. (1982). On the canonical representation of homogeneous Markov processes modelling failure-time distributions, *Microelectronics and Reliability*, **22**(3), 583–602.

Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society, Series B*, **B-39**, 1–38.

Dugan, J. B., Bavuso, S. J. and Boyd, M. A. (1992). Dynamic fault-tree models for fault-tolerant computer systems, *IEEE Transactions on Reliability*, **41**(3), 363–377.

Maier, R. and O'Cinneide, C. (1992). A closure characterisation of phase-type distributions, *Journal of Applied Probability*, **29**(1), 92–103.

Meeker, W. Q. and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*, John Wiley & Sons, New York.

Neuts, M. F. (1981). *Matrix-Geometric Solutions in Stochastic Models*: *An Algorithmic Approach*, Dover Publications, Mineola, New York.

Okamura, H., Dohi, T. and Trivedi, K. S. (2011). A refined EM algorithm for PH distributions, *Performance Evaluation*, **68**(10), 938–954.

Okamura, H., Dohi, T. and Trivedi, K. S. (2013). Improvement of EM algorithm for phase-type distributions with grouped and truncated data, *Applied Stochastic Models in Business and Industry*, **29**(2), 141–156.

Olsson, M. (1996). Estimation of phase-type distributions from censored data, *Scandinavian Journal of Statistics*, **23**, 443–460.

Rauzy, A. (1993). New algorithms for fault trees analysis, *Reliability Engineering & System Safety*, **40**(3), 203–211.

Stewart, W. J. (1994). *Introduction to the Numerical Solution of Markov Chains*, Princeton University Press, Princeton, New Jersey.

Trivedi, K. S. and Bobbio, A. (2017). *Reliability and Availability Engineering*: *Modeling, Analysis, and Applications*, Cambridge University Press, Cambridge, UK.

Wu, C. F. J. (1983). On the convergence properties of the EM algorithm, *Annals of Statistics*, **11**(1), 95–103.

Reliability Analysis and Utilization of Tools

Hiroyuki Okamura, Junjun Zheng and Tadashi Dohi

Graduate School of Advanced Science and Engineering, Hiroshima University

In this paper, we introduce tools commonly used in the field of reliability. Reliability evaluation can be broadly categorized into two approaches: one involves probabilistic models, such as fault trees and Markov models, while the other uses statistical methods to identify models based on data obtained from reliability tests. Both approaches rely on reliability analysis tools for evaluation. This paper provides fundamental knowledge about reliability and explains specific reliability analysis tools in detail.

Key words: Reliability analysis tool, probabilistic model, statistical method, reliability evaluation.