3210

高信頼性設計のための Fault Tree Analysis 支援システムの開発 (ブロック図から FT 形式への変換)

古川 慈之^{*1}, 澤田 浩之^{*2}, 徳永 仁史^{*2}

Development of Computer-aided Fault Tree Analysis System for Reliability Design (Converting Block Diagrams into Fault Tree Forms)

Yoshiyuki FURUKAWA^{*1}, Hiroyuki SAWADA and Hitoshi TOKUNAGA^{*1} National Institute of Advanced Industrial Science and Technology
Namiki 1-2-1, Tsukuba, Ibaraki, 305-8564 Japan

This paper describes a method of converting block diagrams into fault tree forms for computer-aided fault tree analysis (FTA). This conversion method aims at achieving high quality and efficiency of FTA for reliability design in manufacturing industry. The method has been implemented as one of the capabilities of computer-aided FTA software "FTAid", which has been developed in a case study to realize a practical software product for the corporate business based on academic research products. The proposed method enables to construct fault tree (FT) automatically from a block diagram of the system to be analyzed. Moreover, the resulted FT includes the additional information to be utilized by the other capabilities of FTAid for supporting FTA. Therefore, the proposed method can contribute to achieving high quality and efficiency of FTA.

Key Words : Fault Tree Construction, Block Diagram, Fault Tree Analysis, Reliability Design

1. 緒 言

著者らは、製造業における高信頼性設計の実現に向けて、大学の研究成果を企業の実務で利用するためのソフトウェア開発を実施している。その対象は、製造業における工業製品の Fault Tree Analysis (FTA) 支援ソフトウェアである。ここで扱う FTA とは、設計者が製品に生じる可能性のある故障を網羅的に洗い出し、その原因にあらかじめ対処した設計によって製品の信頼性を確保することを目的に実施される手法である。このソフトウェア開発は、ジャトコ株式会社、東京大学、産業技術総合研究所の三者による共同研究の一部として実施されているものであり、それぞれの活動については既報⁽¹⁾⁽²⁾⁽³⁾の通りである。著者らの役割は、独自のソフトウェア開発基盤 MZ Platform⁽⁴⁾⁽⁵⁾を用いて、企業での実務用のソフトウェアを迅速に構築して提供し、その機能として FTA 支援手法の研究成果を導入することで、ソフトウェア開発手法と FTA 支援手法の両方について実証を行うことにある⁽²⁾。中心となる FTA 支援手法の研究成果は村上ら⁽⁶⁾による物理量次元インデクシング法である。開発したソフトウェア「FTAid (エフティイーエイド)」は 2011 年 7 月に正式版をリリースして企業での実利用を開始している。

FTAid が支援する FTA においては、望ましくない故障という事象をその原因となる要素に分解して Fault Tree (FT) を構築する作業が最も重要で難しい。FT を構築する際に、原因となる要素を網羅的に精度良くかつ効率的に挙げられるかどうかは FTA の効果に大きく影響するが、その成否は作業者の知識や経験などの能力に大きく依存することが一般的である。既報⁽²⁾でソフトウェア化の対象とした物理量次元を用いた整合性検証⁽⁶⁾は作成された時点での品質を評価するものであり、同じく物理量次元を用いた類似度検索⁽⁶⁾は蓄積された FT のデータを基に上位の事象から類推して下位への展開案を提示するものである。一方、対象とする製品に関する情報がモデル化されている場合、その情報を基に FT を構築する手法が存在する。そのような FT の構築方法の形式化と自動化に

^{*1} 産業技術総合研究所 (〒305-8564 茨城県つくば市並木 1-2-1)

^{*2} 正員, 産業技術総合研究所
E-mail: y-furukawa@aist.go.jp

については、1970年代から数多くの研究がなされている⁽⁷⁾。それらの多くは、対象となるシステムを構造的または機能的な視点でモデル化して、そのモデルを基にFTを作成するアプローチを取る。近年の例として、Majdaraら⁽⁸⁾が従来のモデル化アプローチの複数の手法、すなわち、グラフ構造のモデル定義と入出力関係表による機能定義を組み合わせたFT構築アルゴリズムを提案している。

本稿では、FTAidの機能の一つとして実装したブロック図からFT形式への変換方法について述べる。ここで対象とするFTの構築方法は、Majdaraら⁽⁸⁾の手法と同様のモデル化アプローチとして従来の手法の中にその要素技術を見ることができるが、FTAidで対象とするFTは一般的なFTA手法（例えば⁽⁹⁾）と異なるP-FTA⁽¹⁰⁾を対象としており、かつ既報⁽²⁾でソフトウェア化の対象とした物理量次元を用いた整合性検証と類似度検索用に追加の属性が含まれるため、それらに対応する部分で独自性がある。以降は、2章でブロック図と変換されるFT形式の定義と表記法について述べ、3章で変換手法について説明した後、4章で手法の位置づけと今後の課題について考察し、最後にまとめを述べる。

2. ブロック図とFT形式

2.1 ブロック図の定義と表記法

ここで対象とする「ブロック図 (Block Diagram)」は、あるシステムを構成する機能の関係を抽象的に表現するための図形式のモデルであり、「ブロック (Block)」、「変数 (Variable)」、「コネクタ (Connector)」から構成される。ブロックは機能の単位を表し、「名前 (Name)」と変数および「関係式 (Equation)」を有する。関係式は所有する変数の「名前 (Name)」を用いた方程式でブロックの機能を表す。変数は名前と「別名 (Alias)」,「単位 (Unit)」を有し、また変数の「種別」として「入力」、「出力」、「状態」の三種が存在する。以降では、種別が限定される場合に「入力変数 (Input Variable)」、「出力変数 (Output Variable)」、「状態変数 (State Variable)」という表記を用いる。変数の名前はブロックの関係式の記述に用いられるため、所属するブロックで重複しない必要がある。別名には変数の役割を示すわかりやすい表現を与えることができる。単位は変数が扱う物理量を示す。コネクタは、所属するブロックの異なる入力変数と出力変数を組み合わせるために接続するもので、接続される二つの変数の単位が同一である必要があり、また各変数に接続されるコネクタは一つ以下とする。

Fig. 1. Notation of block diagram

図1に本稿で扱うブロック図の表記法を示す。ブロック図は、ノードとエッジで構成されるネットワーク図として表記する。ブロックと入力変数および出力変数をノードとして表記し、コネクタを入力変数と出力変数を接

続する無向エッジとして表記する。入力変数は所属するブロックへの有向エッジで接続され、出力変数は所属するブロックからの有向エッジで接続される。ブロックのノードには名前と所有する状態変数の名前および関係式が表記される。また、入力変数と出力変数のノードには別名と名前が表記される。図 1 左側に示すように、FTAid では通常このような表記法を用いる。さらに、FTAid では図 1 右側に示す簡略化表示機能を提供している。簡略化表示では、コネクタと接続された二つの変数をまとめて単一の有向エッジとして表記する。

2・2 変換される FT 形式

図 2 に本研究で対象とする FT 形式の定義と表記法を示す。FT は、機器の故障等の望ましくない「事象 (Event)」をその原因となる事象へと段階的に分解した結果得られるツリー構造の表現である。そのため、上位の事象が下位の事象を原因とした結果を表現する。上位の事象と下位の事象は「論理ゲート (Logic Gate)」を介して接続され、下位に複数の事象が存在する際に、そのうち一つでも発生すれば上位の事象が発生する場合は「OR ゲート」、すべての下位の事象が発生した場合のみ上位の事象が発生する場合は「AND ゲート」を用いる。下位の事象が一つの場合でも OR ゲートの存在を仮定すればよい。ある事象の発生原因が下位の事象の単一論理ゲートによる組み合わせでは表現できない場合にも、OR ゲートと AND ゲートの組み合わせとなる中間的な事象を仮定すれば十分に表現できることが多い。図 2 に示すように、FTAid では頂上事象を左に配置して右側に下位の事象を配置する。

Fig. 2. Fault tree in FTAid

FT における事象は、内容を表現する「タイトル (Title)」とその詳細を説明する「注記 (Detail description)」を有することは一般的だが、FTAid で対象とする事象はさらに、対象となる物理量を示す「単位 (Unit)」とその物理量に関する「故障値状態 (Fault Status)」を有する。故障値状態は「+」か「-」のいずれかを取り、対象となる物理量が通常の値に対して大きいまたは小さい状態を表す。このような追加情報を有する FT は、物理量次元を基にした FT の整合性検証と類似度計算が可能となり、有用であることは既報²⁾の通りである。

3. ブロック図から FT 形式への変換

3・1 変換処理の流れ

FTAid においてブロック図から FT 形式に変換するための操作の流れは次の通りである。ブロック図として対象とするシステムのモデル化が完了したとして、ユーザは問題があると考えられる変数を画面上で選択する。続

いて、その変数が示す物理量の故障値状態を指定する。最後に FT 変換の実行ボタンを押すと、ブロック図が FT 形式に変換され、ツリー表示領域に表示される。

このとき行われる変換処理の流れを図 3 に示す。前述のように最初にユーザがいくつかの選択や指定をする以外は、ブロック図として表現されたモデルに応じて自動的に実行される。すなわち、ブロック図中でノードとして表現される入力変数・出力変数・ブロックを、選択された変数を出発点としコネクタなどのエッジから探索して FT を作成する。

Fig. 3. Flow of converting process

最初に、選択した変数から FT の頂上事象が作成される。処理中に作成された事象はすべてキューに追加され、順番に取り出されて処理が行われ、キューが空になった時点で処理が完了となる。

キューから取り出された事象が変数に関する事象である場合、その事象のタイトル・単位・故障値状態などの属性がブロック図の情報を基に設定される。変数に関する事象のタイトルは次のように設定する。キューに変数に関する事象が追加された時点で、必要な変数の名前または別名および故障値状態が得られていることを前提とする。これらを用いて、「(ブロックの名前)の(変数の名前/別名)が(大きい/小さい:故障値状態)」という趣旨の文字列を設定する。故障値状態を表現する文字列は物理量に応じて変えることも可能だが、ここでは統一した表現を用いる。続いて、変数はブロックに所属し、入力変数と出力変数はコネクタと接続する場合があるため、二方向の探索を行う。

コネクタ側への探索は、対象とする変数がコネクタを介して別の変数と接続している場合に行う。ただし、コネクタを介して別の変数に接続している場合でも、「コネクタ側完了フラグ」が設定されている場合にはこの探索

を行わない。このフラグは、ブロック図にループが含まれている場合でも、一度この探索が実施されたら確実にフラグを設定することで適切に展開を終了できる。コネクタ側への探索では、接続先の変数に対応する子事象の作成とキューへの追加を行う。この際、接続先の変数の故障値状態は接続元と同一に設定する。また、接続先の変数にコネクタ側完了フラグを設定する。

ブロック側への探索は、変数が常にブロックと接続されているため必ず行うが、コネクタ側の場合と同様に「ブロック側完了フラグ」が設定されている場合にはこの探索を行わない。ブロック側への探索では、対象とする変数を有するブロックに関する子事象の作成とキューへの追加を行う。この際、対象とする変数にブロック側完了フラグを設定する。

キューから取り出された事象がブロックに関する事象である場合、ブロック図の情報を基にその事象のタイトルを設定し、親事象の注記に関係式を設定する。このとき、ブロックに関する事象のタイトルは単純にブロック名を用いて「(ブロックの名前)に異常あり」という趣旨の文字列を設定する。ブロックは変数と異なり、物理量を示す単位とその故障値状態が設定されていない。これは、この変換方法においてブロックに関する事象は関係式が示す機能モデルの例外を表しているからである。つまり、もし望ましくない事象が発生していても関係式が満たされていれば、原因は変数の入出力に限定される。一方、ブロックに関する事象はその関係式が満たされていない場合を表現する。

続いて、親事象に対応する変数以外のすべての変数に関する兄弟事象を作成してキューに追加する。この際、兄弟事象に対応する変数にはブロック側完了フラグを設定する。また、兄弟事象に対応する変数には後述のように故障値状態を計算して設定する。

なお、処理の流れからもわかるように、現時点のブロック図で表現および FT に変換できる対象は、機能部品の関係式と変数の接続関係に限られ、冗長系の表現は扱えない。そのため、この変換処理で得られる FT は AND ゲートを持たず OR ゲートのみで構成される。

3・2 変数の故障値状態の計算

変数に関する事象を作成する際に、物理量の単位は直接ブロック図に含まれるためそのまま利用することができる。一方、故障値状態はブロック図に含まれていないため、何らかの方法で計算する必要がある。ブロックに対応する事象の処理で述べたように、ある変数に関する事象の子事象に対応するその他の変数は、関係式の示す機能モデルが満たされている場合の原因の候補である。すなわち、親事象に対応する変数の故障値状態が与えられれば、関係式から子事象に対応する各変数の故障値状態を計算することができる。なお、ここではブロックに属するすべての変数に関して単一の線形方程式が関係式として与えられている場合を想定する。

関係式 $f=0$ が式 (1) に示すように全ての変数に関して線形であれば、式 (2) から親事象に対応する変数 v_p を着目する子事象に対応する変数 v_c で偏微分して得られる符号を計算し、式 (3) に示すように親事象の故障値状態の符号 s_p に乗じることで子事象の故障値状態の符号 s_c が得られる。ただし、式 (2) の変形では v_p が v_c の関数で、かつその他の変数が一定とみなせることを利用している。

$$f(\mathbf{v}) = a_0 v_0 + \cdots + a_i v_i + \cdots + a_{n-1} v_{n-1} = 0 \quad (i: 0 \leq i < n \quad c, p \in i \quad c \neq p) \quad (1)$$

$$\frac{\partial f}{\partial v_c} = a_0 \frac{\partial v_0}{\partial v_c} + \cdots + a_i \frac{\partial v_i}{\partial v_c} + \cdots + a_{n-1} \frac{\partial v_{n-1}}{\partial v_c} = a_p \frac{\partial v_p}{\partial v_c} + a_c = 0 \quad \left(\frac{\partial v_i}{\partial v_c} = 0 \quad (i \neq c, p) \right) \quad (2)$$

$$s_c = s_p \cdot \text{sign} \left(\frac{\partial v_p}{\partial v_c} \right) = -s_p \cdot \text{sign} \left(\frac{a_c}{a_p} \right) \quad \text{ただし, } s_p = \begin{cases} 1 \\ -1 \end{cases}, \quad \text{sign}(x) = \begin{cases} 1 & (x \geq 0) \\ -1 & (x < 0) \end{cases} \quad (3)$$

もし関係式が与えられていないかまたは線形の方程式でない場合は、式 (1) に従う特定の線形の方程式が成立すると仮定して計算する。ただし、このとき各変数の係数をどのように設定するかで故障値状態の計算結果が変わってしまう。そのため、ここでは入力変数と状態変数が $a_i=1$ 、出力変数が $a_i=-1$ と係数の初期符号を仮定して、仮定した符号を反転させたい変数に対してはユーザが「符号反転属性」を設定することで対応する。なお、この符号反転属性は線形の関係式が指定されている場合には効果がないものとして扱う。

現時点では、親事象に対応する変数の故障値状態に応じて＋か－のいずれかの故障値状態を有する場合のみ想定する。つまり、より詳細な条件で故障値状態が＋や－になる状況は扱わない。このような複雑な場合分けが必要となる状況がどの程度実務で生じるかによって、今後の対応を検討する必要がある。よくある状況に対しては追加属性の設定等で対応することも可能だが、あまり生じない状況についてむやみに属性を増やすなどの対応をすると、ソフトウェアとしての操作性が低下するリスクが高まるので注意が必要である。

3・3 順展開の有無

扱うブロック図が表現する対象によって、FT 形式への変換処理を変更したい場合がある。例えば、逐次的な処理を実行するソフトウェアの動作を解析する場合には、いずれかの変数が異常値を示した場合に下流の処理を調べる必要はなく、上流側にあたる処理と変数を調べればよい。一方、油圧系の動作を解析する場合には、圧力などの物理量を変数として用いると、ブロック図として下流の機能として記述されている場合でも上流側の機能の不具合の原因となりうる場合があるため、全体を調べる必要がある。

このような違いに対応するために、FTAid では変換実行時にユーザが「順展開の有無」を選択できるようにしている。「順展開無し」が指定されている場合、処理中で対象とする変数が出力変数のとき、その変数に関する事象が作成されてキューに追加される際に、コネクタ側完了フラグを設定する。こうすることで、出力変数の事象がキューに追加されても、コネクタを介して下流の機能の変数に FT を展開（順展開）しない指定が可能となる。つまり、逐次的な処理を実行するソフトウェアの動作を解析するような場合は「順展開無し」を指定し、油圧系の動作を解析するような場合は「順展開有り」を指定すればよい。「順展開の有無」はモデル化に依存する指定なので、ユーザは状況に応じてよく検討して利用する必要がある。

3・4 変換例

図 4 に変換の一例を示す。これは図 1 で示したモデルに対し、出力変数「Out_A1」を選択し故障値状態「+」を指定して変換した例である。変換の際に順展開無しで実行した結果が図中右上側の FT であり、順展開有りで実行した結果が図中右下側の FT である。順展開無しの場合はブロック「Block_A」の出力変数「Out_A2」の事象の下位に事象が作成されていないのに対し、順展開有りの場合は「Out_A2」の事象の下位にブロック「Block_B」の変数に関する事象が作成されていることがわかる。また、二つの FT において各変数の故障値状態が頂上事象の故障値状態から計算されてすべて設定されていることがわかる。

Fig. 4. Conversion example

4. 議論と考察

本報告で提案する変換手法は既に FTAid の機能として実装されているが、実務における検証はまだ行っていない。参考までに、山本らによって実施された事前検討結果⁽¹¹⁾では、設計者が作成した FT と変換で構築した FT を比較すると、設計者が作成した FT では複数の現象をまとめた事象や省略した事象が含まれていたが、変換で構築した FT ではモデルに基づき同じ詳細度で省略なく FT を構築できたと述べられている。つまり、モデルの記述に問題がなければ、提案した処理に基づいて詳細度を揃えて省略なく FT を構築することが可能であると言える。ただし、この事前検討の段階ではモデル編集の GUI と関係式による計算が未実装であり、少なくともこれらに関する検証を実施する必要がある。

対応する本稿の内容としては、ブロック図の表示のわかりやすさに関する検証の必要性が挙げられる。これに関して図 5 に事例に基づいて作成したブロック図とその簡略化表示の例を示す。図からわかるように、対象とするシステムの機能ブロックの数が増えてくると、通常表示では変数ノードの数が多くなり見づらくなる傾向がある。これに対応するためにコネクタの簡略化表示の機能を導入しており、それによって接続された変数ノードの表示が省略されて一定の効果があると推察される。当然のことながら、これは表示の簡略化であり内部的に保持する情報は同一であるので、どちらの表示からでも FT 形式への変換結果は同一である。一方で、モデルの表示において何を表示して何を省略するかはユーザの要求に依存し、またそれによって作業の効率にも影響があるため、実務を想定したさらなる検討が必要である。

Fig. 5. Simplified view example

また、山本らの事前検討結果⁽¹¹⁾で挙げられた課題のうち、条件分岐の扱いおよび他の制御シミュレーションソフトウェアとの連携については、本稿でもまだこれらに対応していない。さらに、本稿で述べた内容においても、ブロックの関係式に単一の線型方程式を想定するなどの制限が含まれている。このように、ブロック図から FT 形式への変換はさらなる手法の拡充と検証が必要であると言えるが、最終的な目標は設計者の実務を支援することであるため、どの機能が本質的に実務を支援することになるかの視点が重要である。今後はこの視点に基づいた拡充と検証を進めることが必要と考える。

5. 結 語

本論文は、製造業における高信頼性設計の実現に向けて、大学での研究成果を企業の実務で利用するための FTA 支援ソフトウェア FTAid の開発で実装した機能の一つであるブロック図から FT 形式への変換手法について報告した。本報告では、FTA の対象となるシステムを FTAid 上のブロック図としてモデル化することで、FT 形式に自動変換できることを示した。この自動変換手法は既存技術と同様のモデル化アプローチに基づくが、FTAid が提供する FTA 支援機能に必要な FT 形式に対応する部分で独自性があると考えられる。このような FT 形式への自動変換が実現できたことで、既報で示した FTA 支援機能をさらに有効活用した精度良く効率的な FTA の実施につながり、高信頼性設計の実現に向けて貢献できたと考える。一方、自動変換を実現するための前提や手法の

適用範囲について拡充が望まれる部分が数多くあるため、実務を想定したさらなる検討を今後も進める予定である。

謝 辞

本論文の内容は、ジャトコ株式会社および東京大学と共同で実施したソフトウェア開発の事例に基づくものである。開発を実施するにあたり、ジャトコ株式会社の平岡洋二氏と山本克成氏をはじめとする多くの関係者からご意見とご協力をいただいた。また、東京大学の村上存教授と同大学院卒業生の吉永遼氏にはブロック図から FT 形式への変換に関して多くの助言とご協力をいただいた。ここに感謝の意を表する。

文 献

- (1) 村上存, 吉永遼, “高信頼性設計のための Fault Tree Analysis 支援システムの開発 (物理量次元インデクシングによる FTA の知識マネジメント)”, 日本機械学会第 21 回設計工学・システム部門講演会講演論文集, 2209, 2011.
- (2) 古川慈之, 澤田浩之, 徳永仁史, “高信頼性設計のための Fault Tree Analysis 支援システムの開発 (MZ Platform を用いた研究成果活用基盤としての実装)”, 日本機械学会第 21 回設計工学・システム部門講演会講演論文集, 2210, 2011.
- (3) 平岡洋二, 山本克成, “高信頼性設計のための Fault Tree Analysis 支援システムの開発 (設計現場での Fault Tree Analysis 支援システムの効果検証と実用性評価)”, 日本機械学会第 21 回設計工学・システム部門講演会講演論文集, 2211, 2011.
- (4) Sawada, H., Matsuki, N., Tokunaga, H., Furukawa, Y., "A Manufacturing Software Development and Operation Framework 'MZ Platform' and its Applications in Industry", In Proc of Advanced Engineering Design, 2004.
- (5) 古川慈之, 澤田浩之, 富澤拓志, 松木則夫, “MZ Platform: イベント駆動型コンポーネント指向開発環境を用いたエンドユーザ開発への試み”, 情報処理学会第 68 回全国大会講演論文集(1), pp.199-200, 2006.
- (6) 村上存, 菊地洋輔, “物理量次元インデクシングに基づく故障木解析の知識マネジメント”, 日本機械学会論文集 C 編, 75(756), pp.2171-2180, 2009.
- (7) Carpignano, A., Poucet, A., "Computer Assisted Fault Tree Construction: a Review of Methods and Concerns", Reliability Engineering and System Safety, 44, 265-278, 1994.
- (8) Majdara, A., Wakabayashi, T., "Component-Based Modeling of Systems for Automated Fault Tree Generation", Reliability Engineering and System Safety, 94, 1076-1086, 2009.
- (9) 小野寺勝重, 実践 FTA 手法, 日科技連, 2000.
- (10) 大城義孝, 村上裕一, 藤原敏, “P・FTA-D・QFD を用いた新製品開発の短期化と品質向上”, 日産技報, 44, pp.50-53, 1999.
- (11) 山本克成, 平岡洋二, “物理量次元インデクシングによる知識マネジメントの製品設計への応用 (FTA 支援ツールの開発と適応事例)”, 日本機械学会第 20 回設計工学・システム部門講演会講演論文集, 2202, 2010.