社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2009－SLDM－139 (8)
2009－ＥＭＢ－ 12 (8)
2009／3／5

抽象モデルベース診断による協調システムの異常原因特定

沓名 拓郎† 佐藤 守一† 中條 直也†

† （株）豊田中央研究所
E-mail: †{kutsuna,shuichi-sato,chujo}@mosk.tytlabs.co.jp

あらまし 近年，自動車の高機能化に伴い，車載電子システムの大規模化が進んでいる．このような大規模化したシステムでは，システムに障害が起こった際に，その原因となった故障箇所を特定することが困難になるという問題が生じる．特定が難しくなる 1 つの要因として，システム内において，異常な入力を受け取った正常なコンポーネントの出力も異常となり，結果的に，システム内の複数箇所で異常が検知されてしまうことが挙げられる．本稿では，このような異常伝播の起こりうるシステムにおいて，根源的な原因となった箇所を自動特定するための診断技術について述べる．診断のためのフレームワークとして，人工知能分野で研究が進められているモデルベース診断の考え方を利用し，また，抽象モデルを用いることで，振る舞いの複雑なソフトウェアを含むシステムを扱えるようにする．
キーワード 異常個所特定，異常伝播，モデルベース診断

Fault Location in Collaborative Systems Using Abstracted Model Based Diagnosis

Takuro KUTSUNA†, Shuichi SATO†, and Naoya CHUJO†

† Toyota Central R&D Labs Inc.
E-mail: †{kutsuna,shuichi-sato,chujo}@mosk.tytlabs.co.jp

Abstract Automotive control systems are getting more large-scaled and complex each day. Such a system will cause a problem with locating faults during system failures. One reason for this problem is that system components which have received an abnormal input data from other components may also output abnormal data even if they are not in abnormal modes, and consequently many redundant faults are detected in the system. In this paper, we propose a diagnosis method to locate the origin of faults automatically in the system where the fault propagation may happen. We use the framework of model based diagnosis developed in the field of artificial intelligence, and furthermore, we introduce an abstract modeling technique in order to deal with complex software components.
Key words Fault Location, Fault Propagation, Model Based Diagnosis

1. はじめに

近年，車載電子システムの大規模化が進んでおり，走行系，安全系，ボデー系，情報系などを合わせて，1 つの車両に搭載される ECU（Electronic Control Unit）の数は少ない車種でも数十個，多い車種では 100 個を超える．今後，車両制御ロジックの複雑化やサービスの多様化に伴い，車載システムはますます大規模複雑化すると予想される．このような大規模複雑化の進んだシステムで問題となるのは，如何に信頼性の高いシステムを開発するかということである．一方で，市場投入後にハードウェア故障やソフトウェア欠陥（バグ）などによりシステムに障害が生じた際，その原因の特定が困難になるという問題も生じる．障害原因の特定は，ソフトウェアを含むシステムでは特に困難となる．なぜなら，多くのハードウェア故障とは異なり，ソフトウェアの欠陥は見た目では全く判断できないからである．さらに，ソフトウェアモジュール自体は正常であっても，想定していない異常なデータが入力された場合，そのモジュールから出ていくデータも異常になることがある．これにより，外部からはあたかも複数のソフトウェアモジュールで問題が生じたかのように見え，本当に問題のあったモジュールを特定することが難しくなる．以下では，この問題を異常伝播問題と呼ぶ．複数の ECU 間でデータをやりとりして協調動作するようなシステムや，統合化により 1 つの ECU 上で複数のタスクが並行動作するようなシステムが増加するにつれ，異常伝播問題が生じる可能性が高くなると考えられる．

本研究では，車載電子システムにおいて，障害発生時にその

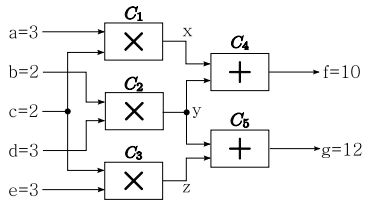

図 1　モデルベース診断の例

原因を自動特定するための診断手法を開発する．障害原因を自動特定するための方法として，人工知能の分野で研究が進められてきたモデルベース診断の考え方を応用する．またその中で，上で述べた異常伝播問題に対処するための方法を提案する．提案手法により，ソフトウェアを含む様々なシステムの障害診断を汎用的に扱うことが可能となる．第 2 節で，モデルベース診断の概要を説明し，第 3 節で，ソフトウェアを含む車載システムに対して，モデルベース診断を応用する方法について述べる．最後に，第 4 節でまとめと今後の課題について述べる．

2. モデルベース診断

モデルベース診断とは，1980 年代から人工知能分野で研究がすすめられてきた，システム診断のためのフレームワークである [1], [2], [4]．この節では，具体例を用いてモデルベース診断の考え方を概説する．

2.1 用　　語

モデルベース診断の説明に使用する用語を以下にまとめる．コンポーネントは，システムの構成要素であり，各コンポーネントについて，その振る舞いを定義する必要がある．モデルベース診断では，各コンポーネントが正常に機能しているかどうかを求めることが目的となる．System Description (SD) は，システムの構成から導かれる論理的な関係を示す式であり，Observations (OBS) は，システム内を流れるデータの観測結果を表す式である．また，Diagnosis (DIAG) は，システム内の各コンポーネントが正常か異常かを表した式である．

2.2 モデルベース診断の例

図 1 に示した例を用いて，モデルベース診断の考え方を説明する．図 1 は，全部で 5 つコンポーネントを持つシステムの例である．C_1, C_2, C_3 は 2 つの入力の積を出力するコンポーネントであり，C_4, C_5 は 2 つの入力の和を出力するコンポーネントである．システムは a, b, c, d, e を入力として持ち，f, g を出力する．また，x, y, z はシステムの内部を流れるデータである．今，内部データを除く a, b, c, d, e, f, g の値が図 1 に示すように観測されているとする．図 1 のシステムに対して，System Description (SD) を次のように記述する．

$$\begin{aligned}\mathrm{SD} \equiv & \{ok(C_1) \rightarrow x = a \times c\} \\ & \wedge \{ok(C_2) \rightarrow y = b \times d\} \\ & \wedge \{ok(C_3) \rightarrow z = c \times e\} \\ & \wedge \{ok(C_4) \rightarrow f = x + y\} \\ & \wedge \{ok(C_5) \rightarrow g = y + z\}\end{aligned} \quad (1)$$

ただし，$ok(C)$ とは，コンポーネント C が正常であることを示す．例えば (1) 式の 1 行目によって，コンポーネント C_1 が正常ならば，x の値は a と c の積と一致するという関係が表現されている．このように，System Description では，コンポーネントが正常な場合にシステム内で成り立つ関係を記述する．また，図 1 における Observations (OBS) を以下のように記述する．

$$\begin{aligned}\mathrm{OBS} \equiv & \{a = 3\} \wedge \{b = 2\} \wedge \{c = 2\} \wedge \{d = 3\} \\ & \wedge \{e = 3\} \wedge \{f = 10\} \wedge \{g = 12\}\end{aligned} \quad (2)$$

Diagnosis とは，各コンポーネントが正常か異常かを示したものであり，例えば，コンポーネント C_1, C_3 が異常で他が正常の場合，Diagnosis（DIAG）を次のように記述する．

$$\mathrm{DIAG} \equiv \neg ok(C_1) \wedge ok(C_2) \wedge \neg ok(C_3) \wedge ok(C_4) \wedge ok(C_5) \quad (3)$$

ただし，$\neg$ は否定記号であり，$\neg ok(C)$ は C が異常であることを意味する．以下では，異常なコンポーネントを括弧で括って Diagnosis を表現する．例えば，(3) 式の Diagnosis を $\{C_1, C_3\}$ と表す．

モデルベース診断では，System Description と Observations から，適切な Diagnosis を求める．具体的には，「どのように Diagnosis を仮定すれば，System Description と Observations の間に矛盾が生じないか」という考え方に基づき Diagnosis を求める．数学的には，次式を満たす Diagnosis を求める．

$$\mathrm{SD} \wedge \mathrm{OBS} \wedge \mathrm{DIAG} = \mathrm{True} \quad (4)$$

例えば，図 1 において Diagnosis を {}（すべてのコンポーネントが正常）と置くと，(1) 式と (2) 式より

$$f = x + y = a \times c + b \times d = 3 \times 2 + 2 \times 3 = 12 \neq 10$$

となり，矛盾が生じるため，(4) 式は成立しない．よって，{} は不適切な Diagnosis となる．また，Diagnosis を $\{C_1\}$ とすると，(1) 式の 1 つめの関係式 $x = a \times c$ が無効となり，x に適当な値を設定することで (4) 式を満たすことができる．よって，$\{C_1\}$ は適切な Diagnosis となる．同様にして，$\{C_1\}$ $\{C_4\}$ $\{C_2, C_5\}$ $\{C_2, C_3\}$，および，これらの上位集合が図 1 に対する適切な Diagnosis となる．Diagnosis のうち，$\{C_1\}$ $\{C_4\}$ $\{C_2, C_5\}$ $\{C_2, C_3\}$ のように，異常コンポーネントを 1 つでも減らすと (4) 式が不成立になるものを Minimal Diagnosis と呼ぶ．

上記の例ように，モデルベース診断では，1 組の System Description と Observations に対して，複数の Diagnosis が求ま

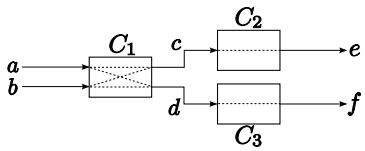

図 2　抽象モデルの例

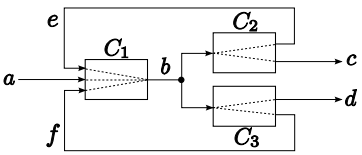

図 3　ループのあるシステムの例

ることがある．このような場合，複数のコンポーネントが同時に異常になる確率は小さいと考え，異常コンポーネントが少ない Diagnosis を診断結果として出力する．よって，図 1 の例では，C_1 または C_4 のどちらかが異常という診断結果を出力する．

3. 車載システムへの応用

この節では，ソフトウェアを含む車載システムに対して，モデルベース診断を適用するための方法について述べる．

3.1 コンポーネントについて

第 2. 節で述べたように，モデルベース診断では各コンポーネントに関して正常，異常の診断を行う．車載システムを対象とした場合，様々なレベルのコンポーネントが考えられる．例えば，車に搭載されている各 ECU をコンポーネントと見なし，それらが車載ネットワークでつながったものをシステム全体と捉えれば，ECU を診断対象とした診断システムとなる．また，統合化された ECU において，OS 上で複数のタスクが実行されている状況であれば，各タスクをコンポーネントと見なすことができる．この場合，タスクのレベルで診断を行うことになる．このように，コンポーネントの見方を変えることで，様々なレベルの診断システムを構築することが可能となる．

3.2 抽象モデルベース診断

通常のモデルベース診断では，コンポーネントの振る舞いの記述が必要となるが，コンポーネントがソフトウェアを含む場合，その振る舞いを正確に書き下すことは非常に難しい．この問題に対処するため，提案手法では振る舞いを抽象化したモデルを用いる [3], [5]．抽象モデルでは，「コンポーネントが正常で入力が正常ならば出力は正常」という関係を用いて System Description を求める．抽象モデルでは，コンポーネントの振る舞いを書き下す必要が無いため，ソフトウェアを含むコンポーネントを容易に扱うことができる．例えば，図 2 のシステムに対して，抽象モデルでは System Description を以下のように記述する．

$$\begin{aligned}\mathrm{SD} \equiv & \{ok(C_1) \wedge ok(a) \wedge ok(b) \rightarrow ok(c) \wedge ok(d)\} \\ & \wedge \{ok(C_2) \wedge ok(c) \rightarrow ok(e)\} \\ & \wedge \{ok(C_3) \wedge ok(d) \rightarrow ok(f)\} \end{aligned} \quad (5)$$

(5) 式の 1 行目は，コンポーネント C_1 が正常で，かつ，その入力である a, b が正常ならば，その出力 c, d は正常という関係を表している．

抽象モデルを用いる場合，Observations としてデータ値そのものではなく，各データに対し何らかの基準で正常か異常かを判断した結果を用いる．例えば，図 2 において c, e は異常，その他のデータは正常と判断した場合，Observations を次のように記述する．

$$\mathrm{OBS} \equiv ok(a) \wedge ok(b) \wedge \neg ok(c) \wedge ok(d) \wedge \neg ok(e) \wedge ok(f) \quad (6)$$

データが正常か異常かを判断するための方法として，以下のようなものが考えられる．

a） 設計者の知識に基づく方法

データが取りうる範囲を規定し，その範囲を外れた場合に異常とする方法や，データの発生周期を規定し，規定周期で発生しない場合に異常とする方法など．

b） 知識以外に基づく方法

蓄積データから，統計的学習やデータマイニング手法などで正常異常の判断基準を求める方法や，コンポーネントの中身を別途モデル化し，モデル検査等の形式手法を用いて正常異常の判断基準を求める方法など．

抽象モデルを用いる場合の Diagnosis の求め方は，第 2. 2 節で述べた通常のモデルベース診断の場合と同じである．つまり，得られた System Description と Observations に対して，(4) 式を満たす Diagnosis を求める．例えば，System Description と Observations がそれぞれ (5) 式と (6) 式で与えられる場合，(4) 式を満たす Diagnosis は $\{C_1\}$，および，その上位集合となる．よって，この場合は C_1 が異常と診断する．これは，C_1 で生じた異常によりデータ c が異常となり，それを入力とする C_2 の出力 e も異常になるという異常伝播問題に対し，その根源原因である C_1 を特定できることを意味する．

3.3 ループのあるシステムへの対応

抽象モデルを用いたモデルベース診断 [3], [5] では，各コンポーネントについて「コンポーネントが正常で入力が正常なれば出力は正常」という関係を利用し，System Description を導出する．しかし，システム内にデータがループする部分がある場合，抽象モデルベース診断は上手く機能しない場合がある．この節では，ループのあるシステムで生じる問題について述べ，その問題に対処するための改良方法について述べる．

3.3.1 ループのあるシステムで生じる問題

図 3 はループを持つシステムの例である．第 3. 2 節の方法では，このシステムの System Description は次のように求まる．

$$\begin{aligned}\mathrm{SD} \equiv & \{ok(C_1) \wedge ok(a) \wedge ok(e) \wedge ok(f) \rightarrow ok(b)\} \\ & \wedge \{ok(C_2) \wedge ok(b) \rightarrow ok(c) \wedge ok(e)\} \\ & \wedge \{ok(C_3) \wedge ok(b) \rightarrow ok(d) \wedge ok(f)\} \end{aligned} \quad (7)$$

今，次式の Observations が観測されたとする．

$$\mathrm{OBS} \equiv ok(a) \wedge \neg ok(b) \wedge ok(c) \wedge ok(d) \wedge \neg ok(e) \wedge ok(f) \quad (8)$$

このとき，(7) 式の SD と (8) 式の OBS を用いてモデルベース診断を行うと，{}（異常コンポーネント無し）が診断結果として求まる．(8) 式において，データ b, e に異常が観測されているにもかかわらず，診断結果が異常コンポーネント無しとなるのは明らかに不適である．この例のように，システム内にループが存在し，かつ，そのループ上のデータがすべて異常と観測された場合，第 3.2 節で述べた System Description ではモデルベース診断が上手く機能しない．

3.3.2 System Description の改良

第 3.2 節で述べた方法では，個々のコンポーネントにおいて成り立つ普遍的な関係を System Description として表した．ここでは，システム内に存在する個々のループにおいて成り立つ普遍的な関係を System Description に追加することで，ループのあるシステムを適切に扱うことができるようにする．System Description の導出方法を以下のように改良する．

System Description の導出手順（ループ対応版）

Step1. 第 3.2 節の方法で System Description を求める．

Step2. システム内のすべてのループについて「ループ上に存在するコンポーネントがすべて正常で，かつ，そのループに対する外部からの入力がすべて正常ならば，ループ上のすべてのデータは正常」という関係式を求め，System Description に追加する．ただし，同じデータを 2 度使用するループであっても，同じループを 2 度通らない限りは 1 つのループと見なす．

以下，図 3 のシステムを例に挙げ，上記の手順で導出した System Description を用いて，ループのあるシステムを扱う方法を説明する．図 3 のシステムには次の 3 つのループが存在する．

loop1. $C_1 \rightharpoonup b \rightharpoonup C_2 \rightharpoonup e \rightharpoonup C_1 \quad (a, f)$

loop2. $C_1 \rightharpoonup b \rightharpoonup C_3 \rightharpoonup f \rightharpoonup C_1 \quad (a, e)$

loop3. $C_1 \rightharpoonup b \rightharpoonup C_2 \rightharpoonup e \rightharpoonup C_1 \rightharpoonup b \rightharpoonup C_3 \rightharpoonup f \rightharpoonup C_1 \quad (a)$

ただし，行末の括弧内は各ループに対する外部からの入力を示す．各ループに対して，Step2 で導出される関係式は次のようになる．

loop1. $ok(C_1) \wedge ok(C_2) \wedge ok(a) \wedge ok(f) \rightarrow ok(b) \wedge ok(e)$

loop2. $ok(C_1) \wedge ok(C_3) \wedge ok(a) \wedge ok(e) \rightarrow ok(b) \wedge ok(f)$

loop3. $ok(C_1) \wedge ok(C_2) \wedge ok(C_3) \wedge ok(a) \rightarrow ok(b) \wedge ok(e) \wedge ok(f)$

これらを (7) 式に追加し，以下の System Description を得る．

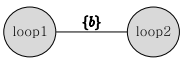

図 4 Elementary Loop の重複関係グラフ

$$\begin{aligned}
\mathrm{SD} \equiv\ & \{ok(C_1) \wedge ok(a) \wedge ok(e) \wedge ok(f) \rightarrow ok(b)\} \\
& \wedge \{ok(C_2) \wedge ok(b) \rightarrow ok(c) \wedge ok(e)\} \\
& \wedge \{ok(C_3) \wedge ok(b) \rightarrow ok(d) \wedge ok(f)\} \\
& \wedge \{ok(C_1) \wedge ok(C_2) \wedge ok(a) \wedge ok(f) \\
& \quad \rightarrow ok(b) \wedge ok(e)\} \\
& \wedge \{ok(C_1) \wedge ok(C_3) \wedge ok(a) \wedge ok(e) \\
& \quad \rightarrow ok(b) \wedge ok(f)\} \\
& \wedge \{ok(C_1) \wedge ok(C_2) \wedge ok(C_3) \wedge ok(a) \\
& \quad \rightarrow ok(b) \wedge ok(e) \wedge ok(f)\} \qquad (9)
\end{aligned}$$

(8) 式の OBS に対して，(9) 式の SD を用いてモデルベース診断を行った場合，求まる Diagnosis は $\{C_1\}$ または $\{C_2\}$ となる．これは，ループ上のすべてのデータが異常となっている部分に対して，そのループ上にあるコンポーネントのどれか 1 つが異常という結果を示しており，妥当な診断結果といえる．

3.3.3 ループ列挙アルゴリズム

前節で述べた改良法では，システム内に存在する全てのループを列挙する必要がある．以下に，システム内のループを列挙するためのアルゴリズムを提案する．

ループ列挙アルゴリズム

Step1. 与えられたシステムの Elementary Loop をすべて列挙する．ただし，Elementary Loop とは，同じデータを 2 度使用しないループのことである． BACKTRACK アルゴリズム [6] を用いることで，Elementary Loop を効率的に列挙することができる．

Step2. Elementary Loop の重複関係を無向グラフとして表す．ただし，各 Elementary Loop をノードとみなし，Elementary Loop の任意の組 $\{L_1, L_2\}$ に関して，L_1 と L_2 が共通データを持つ場合，対応するノード間にエッジを追加する．

Step3. Step2 で求めた無向グラフの連結部分グラフをすべて列挙する．

Step4. Step3 で求めた各連結部分グラフに対し，対応するループを求める．

以下，前節で採り上げた図 3 のシステムを例に挙げ，上記アルゴリズムを説明する．図 3 のシステムに対して，Step1 を適用すると，前節で述べた loop1 と loop2 が Elementary Loop として求まる．ここで，loop3 はデータ b を 2 度使用するため，Elementary Loop とはならないことに注意してほしい．次に，Step2 で Elementary Loop の重複関係を表す無向グラフを求めると，図 4 のグラフが求まる．ただし，エッジ上の $\{b\}$ は loop1 と loop2 がデータ b で重複していることを示す．Step3 で，図 4 のグラフの連結部分グラフを列挙すると，図 5 に示した (a)，(b)，(c) の 3 つのグラフが求まる．最後に，Step4 で図 5 の (a), (b), (c) に対応するループを求めると，それぞれ，

(a) (b) (c)

図 5　図 4 のグラフに対する連結部分グラフ

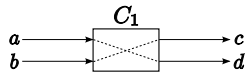

図 6　出力が特定の入力のみに依存する例

前節の loop1，loop2，loop3 が求まる．

3.4　抽象モデルベース診断の拡張

抽象モデルを用いるモデルベース診断では，以下のような拡張が可能である．システムの特徴を踏まえた拡張を行うことで，より詳細なモデル化が可能となり，診断能力を向上させることができる．

a）　出力が特定の入力のみに依存する場合

コンポーネント内部において，ある出力データの計算に，一部の入力データのみを利用している場合を考える．図 6 は，図 2 のコンポーネント C_1 において，出力 d の計算に入力 a のみを利用し，出力 c の計算に入力 b のみを利用している例である．このとき，(5) 式の System Description を次のように修正する．

$$\begin{aligned}\mathrm{SD} \equiv &\{ok(C_1) \wedge ok(a) \rightarrow ok(d)\}\\ &\wedge \{ok(C_1) \wedge ok(b) \rightarrow ok(c)\}\\ &\wedge \{ok(C_2) \wedge ok(c) \rightarrow ok(e)\}\\ &\wedge \{ok(C_3) \wedge ok(d) \rightarrow ok(f)\}\end{aligned} \tag{10}$$

比較のため，以下の Observations が得られた状況を考える．

$$\mathrm{OBS} \equiv \neg ok(a) \wedge ok(b) \wedge \neg ok(c) \wedge ok(d) \wedge ok(e) \wedge ok(f) \tag{11}$$

(11) 式の OBS と (5) 式の SD を用いたモデルベース診断では，診断結果は {}（異常コンポーネント無し）となる．これは，データ a が異常なためデータ c が異常になったという解釈が可能なためである．一方，(11) 式の OBS と (10) 式の SD を用いたモデルベース診断では，診断結果は $\{C_1\}$ となる．これは，データの依存関係からデータ a の異常によってデータ c が異常になることはないとわかるため，データ c が異常なのは C_1 の異常が原因と結論づけられるためである．このように，コンポーネント内部で入出力変数の依存関係がわかっている場合は，それを踏まえた System Description を用いることで，モデルベース診断の診断能力を上げることができる．

b）　データが複数の正常異常基準を持つ場合

ある 1 つのデータに関して，複数の正常異常基準が定義できる場合を考える．図 7 は，図 2 のデータ c, e に関して，それぞれ範囲に基づく基準と周期に基づく基準が定義される例である．ただし，$ok(x).range$ と $ok(x).period$ はそれぞれデータ x が範囲と周期に関して正常であることを表す．また，コンポーネント C_2 において，e は c に対し範囲と周期に関してそれぞれ独立に依存しているとする．このとき，(5) 式の System Description を次のように修正する．

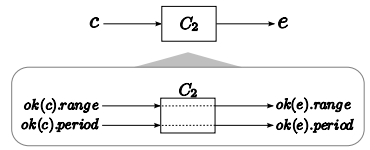

図 7　データが複数の正常異常基準を持つ例

$$\begin{aligned}\mathrm{SD} \equiv &\{ok(C_1) \wedge ok(a) \wedge ok(b)\\ &\quad \rightarrow ok(c).range \wedge ok(c).period \wedge ok(d)\}\\ &\wedge \{ok(C_2) \wedge ok(c).range \rightarrow ok(e).range\}\\ &\wedge \{ok(C_2) \wedge ok(c).period \rightarrow ok(e).period\}\\ &\wedge \{ok(C_3) \wedge ok(d) \rightarrow ok(f)\}\end{aligned} \tag{12}$$

前節と同様の理由により，(5) 式の代わりに (12) 式を SD とすることで，モデルベース診断の診断能力が向上する．

c）　正常異常基準が複数データで定義される場合

複数のデータを用いて定義される正常異常基準を考える．例として，図 2 のデータ c, d に関して，それぞれ単独では正常異常基準を求めることは難しいが，c を d で割った値 c/d について正常異常基準が定義できる場合を考える．c/d の値が異常のとき，c と d の少なくとも一方は異常であると考えられるため，このときの Observations を次のように記述できる．

$$\mathrm{OBS} \equiv \mathrm{OBS}_{others} \wedge \{\neg ok(c) \vee \neg ok(d)\}$$

ただし，OBS_{others} はその他の観測結果から得られる Observations である．例えば，図 2 のシステムにおいて，

$$\mathrm{OBS} \equiv ok(a) \wedge ok(b) \wedge \{\neg ok(c) \vee \neg ok(d)\} \wedge \neg ok(e) \wedge ok(f)$$

が得られたとき，(5) 式の System Description を用いたモデルベース診断で得られる診断結果は $\{C_1\}$ となる．

d）　正常異常の判断が難しいデータがある場合

必ずしもすべてのデータを，明確な基準で正常か異常か判断できるとは限らない．例えば，あるデータ x に関して，$x \leqq 10$ ならば正常，$x \geqq 30$ ならば異常と判断できるが，$10 < x < 30$ の場合，正常とも異常とも判断できない場合が考えられる．このような場合，次のように Observations を求めればよい．

（1）　$x < 10$ の場合，Obsesrvations に $ok(x)$ を追加

（2）　$x > 30$ の場合，Observations に $\neg ok(x)$ を追加

（3）　上記以外の場合，Observations には何も追加しない

このように，正常か異常か不明のデータに関しては，Observations に何も記述しない．例えば，図 2 のシステムにおいて，データ d の正常異常が不明で，その他のデータに関して

$$\mathrm{OBS} \equiv ok(a) \wedge ok(b) \wedge ok(c) \wedge ok(e) \wedge \neg ok(f)$$

が得られたとき，(5) 式の System Description を用いたモデルベース診断で得られる診断結果は $\{C_1\}$ または $\{C_3\}$ となる．

これは，不明部分の d が異常とすれば診断結果は $\{C_1\}$ ，d が正常とすれば診断結果は $\{C_3\}$ という両方の可能性が考えられるためである．

e） 時間の概念を取り入れる場合

抽象モデルを用いたモデルベース診断は，時間の概念を取り入れて拡張することができる．そのためには，各コンポーネントにおいて，ある時刻 t の入力データが，次の時刻 $t+1$ の出力に影響すると考える．例えば，図 2 のシステムに対して，時間の概念を取り入れた場合の System Descrition は次式のように表現できる．ただし，時刻として $t=0,1,2,3$ の 4 時刻を考慮する．

$$
\begin{aligned}
\mathrm{SD} \equiv & \{ok_0(C_1) \wedge ok_0(a) \wedge ok_0(b) \rightarrow ok_1(c) \wedge ok_1(d)\} \\
& \wedge \{ok_0(C_2) \wedge ok_0(c) \rightarrow ok_1(e)\} \\
& \wedge \{ok_0(C_3) \wedge ok_0(d) \rightarrow ok_1(f)\} \\
& \wedge \{ok_1(C_1) \wedge ok_1(a) \wedge ok_1(b) \rightarrow ok_2(c) \wedge ok_2(d)\} \\
& \wedge \{ok_1(C_2) \wedge ok_1(c) \rightarrow ok_2(e)\} \\
& \wedge \{ok_1(C_3) \wedge ok_1(d) \rightarrow ok_2(f)\} \\
& \wedge \{ok_2(C_1) \wedge ok_2(a) \wedge ok_2(b) \rightarrow ok_3(c) \wedge ok_3(d)\} \\
& \wedge \{ok_2(C_2) \wedge ok_2(c) \rightarrow ok_3(e)\} \\
& \wedge \{ok_2(C_3) \wedge ok_2(d) \rightarrow ok_3(f)\} \qquad (13)
\end{aligned}
$$

ここで，$ok_t(x)$ は x が時刻 t において正常であることを意味する．また，Observations として，各時点でデータが正常か異常かを観測した結果を用いる．図 2 のシステムに対する Observations の例を次式に示す．

$$
\begin{aligned}
\mathrm{OBS} \equiv & ok_0(a) \wedge ok_0(b) \wedge ok_0(c) \\
& \wedge ok_0(d) \wedge ok_0(e) \wedge ok_0(f) \\
& \wedge ok_1(a) \wedge ok_1(b) \wedge ok_1(c) \\
& \wedge ok_1(d) \wedge ok_1(e) \wedge ok_1(f) \\
& \wedge ok_2(a) \wedge ok_2(b) \wedge \neg ok_2(c) \\
& \wedge ok_2(d) \wedge ok_2(e) \wedge ok_2(f) \\
& \wedge ok_3(a) \wedge ok_3(b) \wedge ok_3(c) \\
& \wedge ok_3(d) \wedge \neg ok_3(e) \wedge \neg ok_3(f) \qquad (14)
\end{aligned}
$$

また，Diagnosis は各コンポーネントが各時刻で正常か異常かを表すものとする．ただし，最終時刻におけるコンポーネントの正常異常は Diagnosis に含めない．なぜなら，それ以降の出力が Observations として観測されないため，最終時刻でのコンポーネントの正常異常を判断することができないためである．(13) 式の SD と (14) 式の OBS に対し，(4) 式を満たす DIAG を求めると以下が求まる．

$$
\begin{aligned}
\mathrm{DIAG} = & ok_0(C_1) \wedge ok_0(C_2) \wedge ok_0(C_3) \\
& \wedge \neg ok_1(C_1) \wedge ok_1(C_2) \wedge ok_1(C_3) \\
& \wedge ok_2(C_1) \wedge ok_2(C_2) \wedge \neg ok_2(C_3) \qquad (15)
\end{aligned}
$$

(15) 式の結果は，C_1 は時刻 $t=1$ で異常となったが時刻 $t=2$ で正常に復帰し，C_3 は時刻 $t=2$ で異常となったと解釈される．時間の概念を組み込むことで，この場合の C_1 ように，ビット化けなどによる一時故障のタイミングまでを含めて，モデルベース診断で求めることができるようになる．

抽象モデルを用いたモデルベース診断に時間の概念を導入する場合，時刻の定義を注意深く決める必要がある．(13) 式の System Description からわかるように，ある時刻の入力の正常性が，次の時刻における出力の正常性を保証するように，時刻を定義する必要がある．仮に，各コンポーネントの出力が 100ms 前の入力に依存するのであれば，時刻は 100ms 毎に 1 単位進むように定義する必要がある．また，それとは別に，Observations が各時刻で計算できることが必要条件となる．例えば，時刻が 100ms 毎に 1 単位進む場合は，100ms 毎にデータの正常異常を判断する必要がある．

4. まとめと課題

本研究では，人工知能分野で研究がすすめられてきたモデルベース診断の考え方を利用し，車載システムの障害診断を行うための枠組みを提案した．ソフトウェアを多く含む複雑なシステムを扱うための手法として，抽象モデルを用いたモデルベース診断を用いた．また，ループを含むシステムも適切に診断できるように，手法の改良を行った．提案手法により，複雑化の進む車載システムにおいて，今後深刻な問題になると予想される異常伝搬問題に対処することが可能となる．抽象モデルを用いたモデルベース診断では，データの正常異常基準が重要な役割を果たすため，与えられたシステムからデータの正常異常基準を作成するための方法を今後開発していく必要がある．

文　献

[1] J. de Kleer and J. Kurien, “Fundamentals of model-based diagnosis,” Proceedings of the Fifth IFAC Symposium on Fault Detection, Supervision, and Safety of Technical Processes (Safeprocess), pp.25–36, 2003.

[2] J. de Kleer and B.C. Williams, “Diagnosing Multiple Faults,” Artificial Intelligence, vol.32, no.1, pp.97–130, 1987.

[3] G. Friedrich, M. Stumptner, and F. Wotawa, “Model-based diagnosis of hardware designs,” Artificial Intelligence, vol.111, no.1-2, pp.3–39, 1999.

[4] R. Reiter, “A Theory of Diagnosis from First Principles,” Artificial Intelligence, vol.32, no.1, pp.57–95, 1987.

[5] G. Steinbauer and F. Wotawa, “Detecting and locating faults in the control software of autonomous mobile robots,” 16th International Workshop on Principles of Diagnosis, pp.13–18, 2005.

[6] R. Tarjan, “Enumeration of the Elementary Circuits of a Directed Graph,” SIAM Journal on Computing, vol.2, pp.211–216, 1973.