2008－IOT－1（5）
2008/5/8

ルールベース障害検出システムの実装例と評価

猪鹿倉 知広　登内 敏夫

NEC サービスプラットフォーム研究所

〒211-8666 神奈川県川崎市中原区下沼部 1753

E-mail: t-igakura@bx.jp.nec.com　tonouchi@cw.jp.nec.com

あらまし　現在、多数の機器をネットワークで接続し、さらに多数の利用者が同時に利用するというシステムが増加している。このようなシステムの規模は大規模化し、その管理コストも増大し続けている。このようなシステムを障害から守るためには、障害が発生したことを少しでも早く発見し、その原因を分析し、管理者に通知する仕組みが必要となる。この目的を達成するために、われわれは現在、携帯電話網を対象としたルールベース障害管理システムの開発を行っている。ルールベースとは、障害とみなすのに参照すべきパラメータやその値に対する閾値、さらに原因を分析するために必要なパラメータの種類、閾値、解析方法などを示したルールを持ち、そのルールに基づいて管理対象の状態を監視し、障害を発見するシステムである。このルールは人手によって作成するため一度で完璧なものを作成できるわけはなく、試行錯誤の末に完成されることがほとんどである。そのため、ルールの試験と試験結果の評価、その評価結果に基づいたルールの再設計というループを実行し続けることでルールを最適なものに近づけていく。本稿では、この試験結果の評価を支援することで、効率的な試験結果評価を可能にし、その結果最適なルール設計を効率的に行うことを可能にすることを目的とする。本稿では、ルールに不備があり、実際には異なる障害原因であるにもかかわらず、ルールの分析結果が同じになってしまうという状況を大量のルール試験結果の中から抽出することでルール評価支援を行う方式を提案し、提案方法によりルール結果の問題点の検出を効率的に行えることを確認した。

キーワード　ルールベース、障害管理、障害発見、原因分析、クラスタリング

Construction and Evaluation of Rule-Based Fault Management System

Tomohiro IGAKURA　Toshio TONOUCHI

NEC　Service Platforms Research Lab.

1753 Simonumabe, Nakahara-ku, Kawasaki-city, Kanagawa, 211-8666 Japan

E-mail: t-igakura@bx.jp.nec.com　tonouchi@cw.jp.nec.com

Abstract　Systems constructed by a lot of equipment connected with network have been increasing. Management cost for this kind of systems is also increasing. In order to reduce influences of fault upon service quality, it is necessary for management systems to discover quickly that fault occurs, to analyze the cause of the fault, and to notify the cause of the fault to a manager. For this purpose, we developed the rule based fault management system. It is the system which watches the state of the managed systems, and discovers fault in accordance with rules which show thresholds of performance parameters and analytical methods etc. Because these rules are written by human, they may not be perfect from first time. They are refined through trial and error. It means that in order to write appropriate rules, it is necessary to continue to writing, testing, evaluating, and redesigning rules. This paper proposes a method supporting the "evaluating rules" in order to efficiently refine appropriate rules, and confirms efficiency of this method by an experiment with real data.

Keyword　Rule-Based, Fault Management, Failure Detection, Root Cause, Clustering

1. はじめに

現在、携帯電話網など多数の機器をネットワークで接続し、さらに多数の利用者が同時に利用するというシステムが増加している。このようなシステムの規模は大規模化し、その管理コストも増大し続けている。このようなシステムを障害から守るためには、障害が発生したことを少しでも早く発見し、その原因を分析し、管理者に通知する仕組みが必要となる。この目的を達成するために、われわれは現在、携帯電話網を対象としたルールベース障害管理システムの開発を行っている。

ルールベースとは、障害とみなすのに参照すべきパラメータやその値に対する閾値、さらに原因を分析するために必要なパラメータの種類、閾値、その他解析

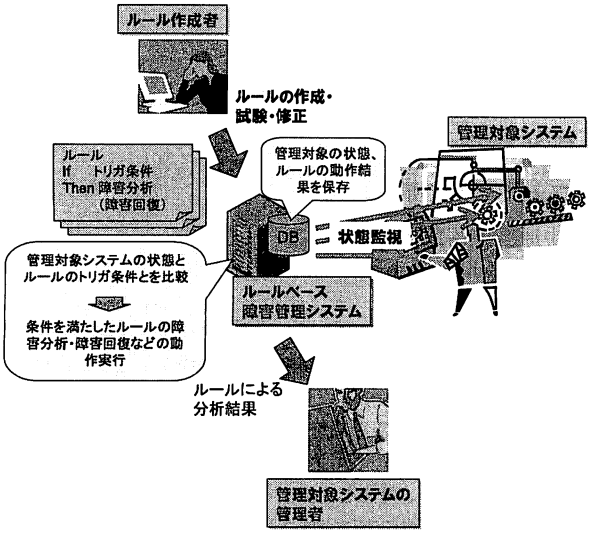

図 1 ルールベース障害管理システム

方法などを示したルールを持ち、そのルールに基づいてシステムのパラメータを監視し、障害を発見・分析するものである[2][4]。そのため、構成などが未知、多種・雑多な機器が多く接続されているシステムでは適切なルールを充分用意することが困難なため適用困難になる可能性があるが、反面、構成が整理されているシステムにおいては、他の障害発見方式よりも信頼性があるといえる。また、ルールベース障害管理の信頼性はどのようなルールが整備されているかに完全に依存する。

このルールは、多くの場合、システムの構成・動作に通暁した人間が作成する[3]。基本的には本人が障害を発見するシーケンスを模して、その手順どおりに動作するルールを作成する。しかし、実際には、人間が障害を探すときには複数のパラメータを見比べて「なんとなく」異常と正常を切り分けていたものに、特定の（複数でも可ではあるが）パラメータに対して特定の閾値を与えて、異常と正常を線引きする必要がある。さらに、原因分析するためには、どの機能がどの機器、プログラム、通信路を使用するのかなどといった、原因分析を行うために必要な知識をルールとして落とし込む必要がある。

そのため作成されるルールは複雑になり、ルール作成後に、そのルールが意図通りにくみ上げられたかを確認する作業が必須となる。ルールが適切に作成されているかを確認するためには、ルールを試験用データを用いて仮想実行させ、その結果、ルールが適切な時刻・機器に対して動作し、動作結果が意図したものとなっているか、それ以外のところではルールが動作しない、もしくは動作したとしてもその結果障害なしと判断されるかどうかを確認する必要がある。実際には一度で適切なルールを作成できることはめったになく、ルールの作成→ルールの試験→ルールの評価→ルールの修正という、ループを繰り返すことで適切なルールに近づいていく。

ルールが意図と異なる動作をしないか確認するためには多彩な状況を含んだ大量のデータに対してルールを動作させる必要がある。しかし、大量のデータに対してルールを動作させれば、そのルールの動作回数も膨大になり、その結果が正しいのか否か評価するための作業も膨大になる。そこで、最適なルールを効率的に作成するためには、このルール試験結果の評価を効率化することが必要となる。

本稿では、このループにおけるルールの評価を支援する機能について提案する。ルールの動作の中から特にルールの動作結果が疑わしいものをピックアップし、そのピックアップされたものの動作が正しいのか否かを、まず評価することで、ルールの問題点を発見するのが効率化される。この目的を達するための技術として、クラスタリングを用いた試験結果の分析とその分析結果を用いたルール評価支援機能の詳細を述べ、その方式の検証を実際のデータに基づいて行う。

2. ルールベース障害管理システム

ルールベース障害管理システムとは、管理対象となるシステムに発生する障害を発見し、障害の原因を分析することを目的とするシステムである。ルールは障害が発生したと思われる状態を条件(トリガ条件)としてもち、さらに障害の原因などを特定するための分析アルゴリズムを持つ。基本的な動作は、管理対象システムの状態を監視し、トリガ条件に記述された状態になったことを検知すると、ルールに記述された分析アルゴリズムにより障害の原因を分析し、分析結果を管理者に提示する(図 1)というものである。

ルールは多くの場合、管理対象システムの動作を熟知した管理者の知識を基に人手により作成される。

2.1. 管理対照システム:携帯電話網

われわれはこのような障害管理システムを特に携帯電話網向けに開発している。携帯電話網は Radio Network Controller(RNC)と呼ばれる機器と、その管理下にある NodeB から構成されている。NodeB はさらに 1 個以上のセルからなり、このセルが直接携帯端末と無線の通信を行う。セルの通信状況を RNC に伝えるため、セルから RNC に定期的に携帯端末とセルの通信についての品質情報(PM データ)が報告される。この PM データは、アクセス数、接続に成功した数、失敗の理由(アドミッションコントロール(受付制御)など)ごとの数がセルごとに RNC に報告される。また、内部で障害などが発生したとき、たとえば何らかの理由で再起動したときなど、自律メッセージが発行される。自律

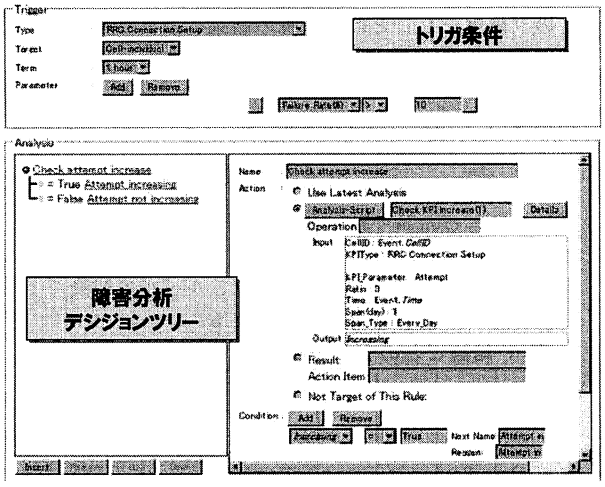

図 2 ルール構成

メッセージは RNC と NodeB から発行される。

2.2. ルール構成

ルールの記述方式にはさまざまな方法が考えられる。われわれが選択した形式では、ルールを起動するトリガ条件を持ち、さらに障害を分析するための分析アルゴリズムをデシジョンツリー[1]を用いて記述する(図 2)。トリガ条件は特定の自律メッセージの発行、もしくは、セルごと・もしくは RNC 全体での PM データに対する条件を記載する。トリガ条件ではデシジョンツリーと異なり、複雑な計算処理は制限されている。その理由は、トリガ条件の計算は膨大な数のセルすべてに対して毎時間行う必要があるため、トリガ条件に複雑な計算を行うと、特に多数のルールを設定した場合に処理能力不足に陥ってしまうためである。

デシジョンツリーでは、予め用意された PM データ、自律メッセージと対象とした解析関数を用いて分岐条件を記述していくことができる。

3. 障害検出ルール

3.1. 作成したルール

今回評価対象とするルールは PM データを用いて品質が劣化した事をトリガとし、その品質劣化について、障害ではなくアクセスの増大や無線環境の悪い場所でのアクセスが多いだけなのか、セルに障害が起きたのか、RNC などより上位におきた障害の影響なのか、といった品質低下の根本原因を分析する。

トリガとする品質劣化は端末の接続成功率の低下したこととしている。端末の接続成功率は端末が接続要求を出した数に対して、最終的に接続が完了するのに失敗した数の割合である。接続成功率の低下は、セル、RNC、その他接続するためのネットワークなどに障害が発生した場合にも、もちろん低下するが、それ以外にも、アクセス数が過剰になることで処理能力を超えた時に接続率が低下することもある。これらの接続率低下原因の分類をルールのデシジョンツリー中で行う。

3.2. ルール試験

ルールを手作業で作成している以上、必ずしも一度で意図通りのルールを作成できているとは限らず、ルールを試験し、その結果を評価することでルールが意図通り動作するのか、しないのであればどこに問題があるのかチェックする必要がある。

ルール試験は管理対象システムの過去のデータを用いてルールを仮想実行することである。試験はまず、ルールが対象とする事象が発生したときの PM データを用いて、ただしくルールが動作するかどうかをチェックする。さまざまなセルの長期間の PM データ、自律メッセージを対象にしてルールを適用させることで、多彩な状況に対して、それぞれルールがどのように動作するのか確認することができる。

つまり、ルール試験後には、その結果を精査し、ルールの動作の結果がルール作成者の意図通りのものになっているのかどうか、つまり、ルールが動くべきときに正常に動作し正しく障害の原因を特定できているのか、それ以外の時にはルールは動作しない、もしくはルールの動作結果として障害なしとの結果がでることを確認することが必要になる(図 3)。この作業をルール評価と呼ぶ。

3.3. ルール評価

ルール評価とはルール試験結果を分析し、ルールの意図通りに動作しているかどうか評価することである。特に多数のセルの長期間のデータに対してルール試験を行う場合、ルールの動作回数は多くなるため、そのなかにルールの意図と異なるものが含まれるか否かの評価が難しくなる。そこで、ルールの試験結果を分析し、ルールが正しく動作しているかどうか評価するのがルール評価である。

本稿で提案するのは評価方式そのものではなく、評

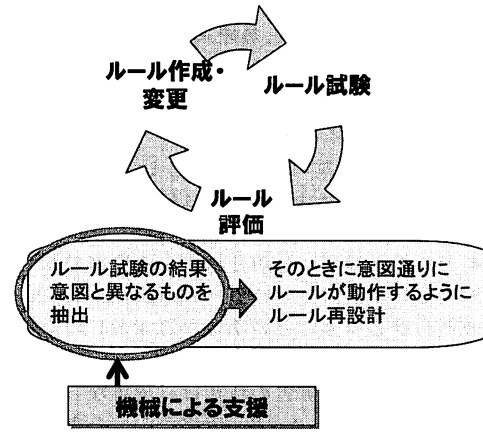

図 3 ルール評価

価を支援するための方式である。機械の支援によりルール試験結果を分析・分類することでルールの不備、たとえばデシジョンツリーに必要な分岐がかけているため分離すべき障害を抽出できていないなどの発見を容易にすることを目的とする。

4. ルール評価支援

4.1. 評価支援方式

ルール評価支援方式のための分析方法としてクラスタリングを用いた。

本方式で用いたクラスタリングは、階層的クラスタ解析と呼ばれるもので、多数のシステムパラメータを持つルール実行結果を、パラメータの差分の二乗和を距離として用い、距離が近いものを結びつけ、最終的に一つのツリーにするものである。ある閾値を設定すれば、ツリーから距離が近いルール実行結果を集めたグループが生成される。作成されたグループの中で、ルールの発火時刻や発火対象システムなどの実行結果が不一致のルールの動作に問題がある可能性が高いのではないかと判断して、ルール作成者に提示する。ルール作成者は提示された時刻・対象とその結果をまず検証することで、ルールの不正を発見しやすくなる。クラスタリングにおいて、パラメータ毎に重み付けを設定することができるが、本システムでは重み付けの調整は行っていない。

ルール試験検証支援システムにおけるクラスタリングの見方は、ルールが動作した時刻・セルの組に対して、全体をクラスタリングによりグループ化し、ある実行結果が集中しているグループにおいて、別の実行結果のものが混入しているかどうか、同じ実行結果のものが他のグループ、もしくはどのグループにも所属しないでいるかどうかをチェックし、この条件に当てはまるものを優先的に、ルールの結論が実際のシステムの状態とあっているかどうかをチェックする(図4)。

4.2. 検証

評価に用いたデータは、無線網における5地域(RNC一台が管理する範囲を地域と呼ぶ)において、それぞれ独立して本方式を適用した物を用いる。それぞれ、地域①～⑤と呼称する。

評価に用いる品質パラメータとして採用したパラメータは15種類。うち、8種類は接続時におけるさまざまな処理の失敗率を用いている。のこりは、全失敗回数のうちの7種類に分類された失敗原因の割合をパラメータとして使用している。15種類とも、失敗率、全失敗における原因ごとの割合など0～1の数値にしている。

ルールが分類する障害の種類は「障害なし」も含めて7種類。それぞれA～F、障害なし、と表記する。この「障害なし」とは、分析の結果、障害ではないと判断された(失敗率は閾値を超えたが、状況から判断してセルやRNCなどの障害が原因ではないと判断された)ものである。

このルールの動作結果に対して、グループ化の閾値を0.4として、グループを生成した。まず、各グループに含まれる結論の数、及び、一つの結論がいくつのグループに分かれているかを調べる。これにより、クラスタリングによるグループ化で、ルールの結論ごとにどれくらい分けられるか評価する。

表 1 結論ごとのグループ数

	A	B	C	D	E	F	障害なし	全体
地域①	3			1	1		7	12
地域②	5					1	8	11
地域③	2						6	8
地域④			2			1	6	4
地域⑤		1	4				7	10

表1に各地域における結論がいくつのグループから構成されているか示す。結論ごとのグループ数の合計と、全体のグループ数が一致しないのは同じグループに異なる結論が含まれてしまっているからである。

例えば、地域①では、全体で12個のグループができた。結論Aはグループα、β、γの3つに、結論Dはδ、結論Eはε、障害なしはζ～μとなる。地域①では、同じグループないには一つの結論しか含まれない。そのため、結論A、D、E、障害なしのグループ数の和は全体のグループ数12と等しくなる。しかし、地域②では、結論Aがグループα、β、γ、δ、εの5つ、結論Fがζ、障害なしが、β、ε、ζ、η、θ、ι、κ、λの8つのグループに属する。このうち、β、

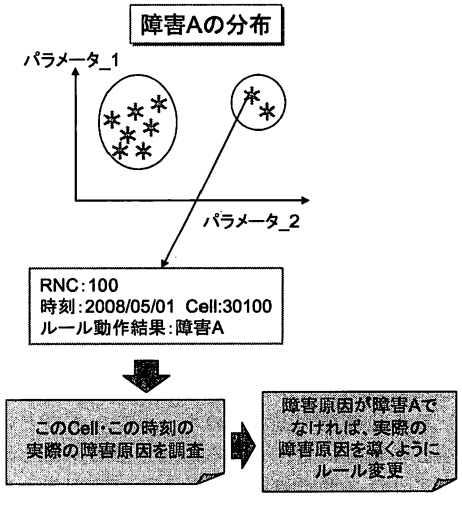

図 4 ルール評価支援方式

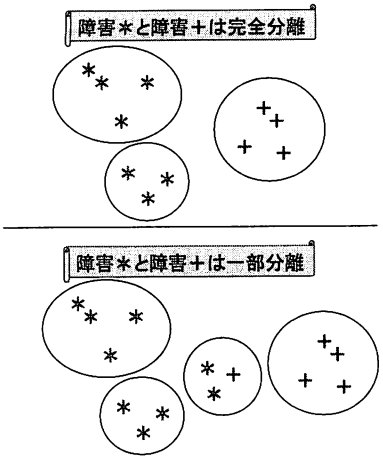

図 5 完全分離と一部分離

ε、ζは同じグループ内に異なる結論のものを含んでいることになる。このように、グループ化はルールの結論とは独立して、システムパラメータのみを用いて分割しているので、ルールの動作結果と必ずしも一致していないことがわかった。

次に、ひとつのグループに含まれる結論の数を示す(表 2)。

表 2 グループに含まれる結論の数

	最大	平均
地域①	1	1
地域②	2	1。3
地域③	1	1
地域④	3	1。3
地域⑤	3	1。2

地域①では、結論 A、D、E、障害なしそれぞれが属するグループには、他の結論が含まれておらず、最大、平均ともに 1 である。しかし地域②においては、グループβ、εは結論 A と障害なし、ζには、結論 F と障害なしと、それぞれ二つの結論を含む。つまり、地域②では 1 グループに含まれる結論が最大 2 となる。

次に、これらの結果を評価するために、本ルール作成支援技術の利用シーンについて考える。本技術はルール作成時に行われ、ルールの結論が間違っているものを探し出すために使用される。その目的のために、望ましいのは、結論ごとのグループ数が全て 1 であり、なおかつ、グループに含まれる結論の数が 1 であることが望ましい。そうであるならば、同じ結論に至ったものが別のグループに分けられているときに、これは、別の結論になるべきであると判断されるからである。実際にはグループ化の閾値によって、この値は変わり、また、クラスタリングに用いられるパラメータの種別によっても異なる。基本的に、結論ごとのグループ数を 1 に近づけると、グループに含まれる結論の数は増加し、グループに含まれる結論の数を 1 に近づけると結論ごとのグループ数が増える。

本技術の使用手順は、まず、ルールのひとつの結論に注目し、その結論がただしく一つなのか、それとも異なる結論が未分岐のまま混在しているのか調べることから始める。一つの結論が複数のグループに分かれていれば、それは、グループ毎に本来異なる結論であるべきものが分かれていることを疑うことになる。例えば、結論 D と E を分離できていなかった場合、地域①の結果を見ると、同じ結論であるのに、二つのグループに分かれてしまう。そこで、この二つのグループのルールの動作結果について、そのときの RNC やセルの状態、さらに、実際に問題が起きていたなら、管理作業が発生していたはずなのでその報告書などから、結論が間違っているか否かを検証する。実際には、それぞれ、D と E と異なる障害になっているため、調査の結果異なる結論に分かれているべきであることがわかり、ルールに対して、D と E を分離するロジックを追加する必要があるとわかる。ルール作成者は、このことを元にルールを改良し、再びルール試験を行う。

次に、仮にルールの原因分析が不十分で、結論を分離できていなかったと仮定して、本方式で分離できるかどうかを確認する。例えば地域①ならば結論が A、D、E、障害なしの 4 種なので、そこから 2 つを取り出す組み合わせ=6 通りのうち区別できるものを求める。ここで二種類の分離可能数というものを導入する。すなわち、完全分離数と一部分理数である(図 5)。完全分離とは結論 A、B を例にとった場合、結論 A を含むグループと結論 B を含むグループが一切一致していないものをさす。完全分離数とは、全組み合わせのうち、完全分離が成り立っている組み合わせの数を言う。地域①では、結論ごとに全て異なるグループになっているので全ての組み合わせで完全分離が成り立っているので、完全分離数は組み合わせの数と等しく 6 となる。一部分離とは、結論 A を含み、かつ結論 B を含まないグループと、結論 B を含み、かつ結論 A を含まないグループがともに存在することをさす。一部分理数とは、全組み合わせのうち一部分離が成り立っているものをさす。地域②では、結論 A と結論 F が完全分離であり、結論 A と障害なしが一部分離である。結論 F と障害なしはどちらも成り立っていない。完全分離であるということは一部分理も成り立っているので、完全分離数が 1、一部分理数が 2 となる。

表 3 結論の分離可能数

	組み合わせ数	完全分離数	一部分離数
地域①	6	6	6
地域②	3	1	2
地域③	1	1	1
地域④	3	0	1

地域⑤	3	0	1

一見、完全分離が成り立っている必要がありそうだが、実際には、一部分離が成り立っていれば、そのグループ間を比較することで結論を分離すべきだということがわかる。例えば、グループα、β、γが在り、αに結論Aのみ、βに結論Bのみが含まれていれば、グループαとβを比較することで、結論AとBに分けるべきであるということがわかる。とはいえ、グループ数を増やしていけば完全分離・一部分理は必ず成り立つので、必ずしも正当な評価とはいえない。

そこで、例えば、結論AとBがルール上で分離されておらず結論A'と出てしまうような場合において、結論A'のグループを二つとりだし、そのグループ間を比較することを考える。A'のグループはα、β、γであり、αに結論Aのみ、βに結論Bのみが含まれるとする。γならば両方の結論が含まれているので、γとα、γとβを比較しても必ずしも結論を分けるべきとはわからない。しかし、αとβを比較すれば、そのグループからどれを取り出しても結論Aと結論Bに分かれているので、A'に結論AとBが含まれていることがわかり、結果、ルールにAとBを分ける新たな分岐が必要であると知ることができる。

このような分離するための処理が成功するまでの比較回数の期待値を計算する。上の例では、α、β、γのうち二つを取り出す3通りの組み合わせのうち、αとβの組み合わせ一通りのみ成功なので、$1\times\frac{1}{3}+2\times\frac{2}{3}\times\frac{1}{2}+3\times\frac{2}{3}\times\frac{1}{2}=2$となり、成功までの期待回数は2となる。また、組み合わせは3通りしかないので、多くとも3度行えば、かならず成功にたどり着くので、最大値は3となる。これらの期待回数、及び最大回数を表4に示す。表中では障害なしをNoと表記する。

表4 分離期待回数

	期待回数	最大回数
地域①	A-D:1.75 A-E:1.75 A-No:2.09 D-E:1 D-No:3.62 E-No:3.62	A-D:4 A-E:4 A-No:25 D-E:1 D-No:22 E-No:22
地域②	A-F:2.67 A-No:2.95 F-No:×	A-F:11 A-No:38 F-No:×
地域③	A-No:2.23	A-No:17
地域④	C-F: × C-No:3.67 F-No:×	C-F: × C-No:17 F-No:×
地域⑤	B-C: × B-No: × C-No:2.41	B-C: × B-No: × C-No:28

例えば、地域①において、障害AとDを分離するロジックが組み込まれていない場合、A-Dの結論を含むグループを平均1.75回、最大でも4回比較することで、障害Aと障害Dに分けるロジックがルールに必要であるとわかる。評価支援方式を用いず、直接A-Dの結果を一つずつ比較した場合、この障害A、障害Dそれぞれの発生回数を用いて同様の計算を行うと平均5.70回、最大1139回の作業が必要とわかる。このことから、評価支援機能により大きな効果を得られていることがわかる。

まとめると、一部分離が成り立っていれば、分離できる。今回の実験の範囲で得られるデータの範囲では地域ごとの組み合わせの合計、16のうち、一部分理が成り立っている11つまり69%について本方式によりルール試験結果の分離を効率化できることになる。また、これらについては、期待回数として1〜3。67回、最大でも38回の比較作業により、新たに分岐すべきものが発見できる。これは、評価支援機能を用いなかった場合の、例えば、地域①の結論A-Dにおける最大1139と比べると大きく効率化されていることがわかり、本評価支援機能の効果を確認できた。

5. まとめ

本稿では、携帯電話網を対象としたルールベース障害管理システムの開発において、特にルールの作成→ルールの試験→ルールの評価→ルールの修正という、ルール作成のループにおける特にルールの評価を支援する機能について提案し、その効果を検証した。支援方式として、ルール試験結果をクラスタリングにより分類し、ルールの動作結果により同じ障害原因であると分析されたもののうち、パラメータが外れているものを抽出することで、実際にはルールを変更し別の障害原因であると分析すべきものを発見することを支援する。

この方式を実際のルールからデシジョンツリーの分岐を削除したものを適用し、異なる障害原因であるにもかかわらず、ルール上は同じ原因であったと分析されてしまうものを発見するための評価作業を効率化できることを確認した。

今後は、利用するパラメータの選択方式、クラスタリング以外の分析方式との比較などを行い、より精度のよいルール評価支援方式を研究していく。

文　献

[1] M.Chen, A. X. Zheng, J. Lioyd, M. I. Jordan, E. Brewer, "Failure Diagnosis Using Decision Trees", 1st International Conference on Autonomic Computing(ICAC'04)

[2] Shinji Nakadai, Masato Kudo, Koichi Konishi, "Rule-Based CIM Query Facility for Dependency Resolution," 15th IFIP/IEEE International Workshop on Distributed Systems: Operation and Management(DSOM 2004)

[3] http://www.ilog.com/products/jrules/

[4] http://www-306.ibm.com/software/tivoli/products/enterprise-console/