メッセージパターン学習による障害発生検知

渡辺幸洋† 松本安英†

大規模なクラウドコンピューティング環境の運用においては，運用プロセスの自動化による省力化は必須である．しかしながら，従来の手作業によるフィルタ定義を伴う障害検知手法では，大規模化するシステムの障害検知を自動的に行うことはできない．本稿では、従来の手法の課題であったメッセージパターンの自動生成を可能にするため，メッセージログからメッセージパターン学習についてベイズ推定を用いて実現する方式を提案する．さらに，この方式により効率の良い学習ができ，高い精度で障害検知を行えることを実験によって示す．

Trouble Detection with Message Pattern Learning

Yukihiro Watanabe† and Yasuhide Matsumoto†

In operations and managements of large-scale IT systems such as cloud computing environment, an automatic operation process is required to manage them by less operators and costs. Even though the detection of troubles by monitoring message patterns is widely used for the rapid troubleshooting, it is difficult to define the message patterns and to keep them updated manually. In this paper, we propose a method for automatic generation of the message patterns using Bayesian methods, focusing on correlations between message patterns and troubles. Moreover, we developed a prototype system to evaluate the effectiveness of our method. It analyzes message log recorded in the case of past troubles, generates the message patterns for the case automatically. The result of the evaluation in experimental environments is performed and the result shows that our method can detect the occurrence of troubles efficiency.

1. はじめに

近年のクラウドコンピューティングへの流れの中で，大規模化のスケールメリットによる運用コストの低減が期待されている．仮想化を活用した計算機資源の集約によって，担当者1人あたり 1,000 台以上のサーバを運用することも可能になるといわれている．しかし，現状では運用管理業務の大規模化への対応は不十分であり，担当者1人あたり 140 台程度である[1].

運用管理において障害対応は多くの時間を費やす業務の1つであり，効率的な運用管理を行うためには，障害対応の迅速化・自動化が求められる．特に，障害の検知から発生している障害の特定までの初動時間を短縮することは，システムが提供するサービスの可用性・信頼性を維持するために重要である．これまで，障害の検知を自動化するために，システムが発するメッセージを監視するいくつかの技術や製品が開発されている．しかしながら，これらの製品は，監視すべきメッセージパターンを人が定義しなければならず，実運用で発生する障害にともなう”想定外”のパターンを検知することができない．さらに，監視対象に新たなメッセージパターンを追加するためには，人によるメッセージの相関などの分析が必要で，手間のかかるものであった．

† 株式会社富士通研究所 クラウドコンピューティング研究センター
Research Center for Cloud Computing, Fujitsu Laboratories Co.Ltd.

我々は，人による事前定義を行わずとも，過去の障害記録とメッセージログから障害を特徴付けるメッセージパターンを抽出し，障害の再発を迅速に検知するとともに，発生した障害の種類と，過去に行った対処を管理者に提示する障害検知システムを開発した．本稿では，我々の開発した障害検知システムの方式と，実際の運用管理業務で取得したログから生成したメッセージパターンによって，障害検知を試みた評価実験の結果を紹介する．

2. クラウドコンピューティング環境の運用管理業務における課題

従来の運用管理では，担当者は，機器が出力する運用管理メッセージを監視し，障害と思われる事象に気づいたときに対応を行ってきた．

クラウドコンピューティング環境の運用の特徴として、コストを抑制するために多数の機器を少人数で運用することが挙げられる．このため，一人の担当者が多数の機器を監視する必要がある．ここで，従来の運用管理の方法に従ってメッセージを管理用コンソールに表示して障害の発生を判断しようとした場合，多数の機器から出力された，正常異常織り交ぜた多数のメッセージが，コンソールに一度に出力されることになる（メッセージ・フラッド）．このため，障害が発生しているのかどうかを，メッセージコンソール画面から判断することは難しく，障害の発生に気づくのが遅れ

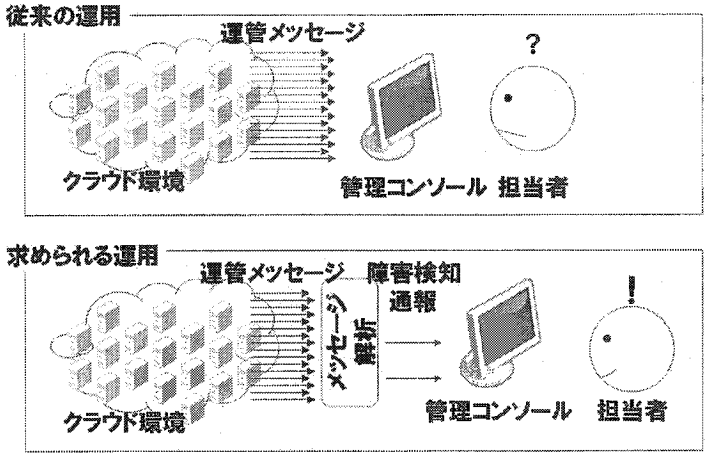

図 1　クラウド環境での障害検知

る懸念がある．この課題を解決するためには，多数発生するメッセージを解析し，障害の発生を捉えて担当者に通知することで，担当者が生のメッセージの集合を解釈する必要なしに障害発生に気づく事を可能とし，効率的な運用を行う仕組みが必要である（図 1）．メッセージを解釈する手法のひとつに，ある障害が発生する時に特徴的に見られるメッセージの出現パターンを検知するものがある．本稿では，この「メッセージパターン」による障害の自動検知について述べる．

3. 関連研究

メッセージパターンによる障害の自動検知技術は，手作業によってメッセージパターンを定義する障害検知と，学習によってメッセージパターンを生成するマイニングベースの障害検知に分類できる．

3.1 パターン定義による障害検知

運用管理担当者などにより，障害発生時のメッセージの出現パターンをモデル化し，定義するものである．簡易なものとしては多数のメッセージから重要なメッセージを抽出するフィルタがある．このタイプのメッセージ監視は，フィルタの定義が比較的容易である反面，メッセージの有無のみで障害発生を検知するので，誤検知が多くなる傾向にある．一方で，複数のメッセージの出現順序などの関係を定義してメッセージを監視するものがある．例としては，IBM の Log Trace Analyzer(LTA)[2]などが挙げられる．これらの複雑なメッセージパターンを定義して行われるメッセージ監視では，単純なフィルタを用いたものよりも障害の発生をより正確に検知できる反面，監視すべきパターンが複雑になり，その定義が難しいという問題がある．

3.2 パターン学習による障害検知

前述のように，監視すべきパターンを人が定義することは難しいため，パターンの生成からトラブル発生の検知までを自動的に行うための研究も多数行われている．これらの研究は，障害の発生を予測する予兆検知と，障害発生後の原因箇所切り分けにも用いられている．Hellerstain らは，正常時のメッセージログを統計的に解析してメッセージ出現パターンをモデル化し，観察されるメッセージパターンが正常時からはずれた場合に障害発生の予兆であると判断する研究を行っている[3]．Hamely らは，ベイズ推定を用いてメッセージの出現頻度を解析し，ディスク障害を予知できる可能性を示唆している[4]．さらに，Salfner らは，隠れマルコフモデルを用いてメッセージの出現パターンと障害発生の関係を解析し，障害発生を予測する方式を提案している[5]．しかし，これらの研究は，主に「正常か異常か」を見分ける(予知する)ためのものであり，「どのような障害が起こったか」については知ることができない．これに対し，Duan らは，メッセージパターンを学習することで，発生した障害が既知のものか未知のものかを判断する研究を行っている[6]．

上記はいずれも障害の発生を検知したり，障害の種類を特定したりするための研究であるが，複数機器からなるシステムでのメッセージログを使った障害検知の例はない．我々の研究に近いものとしては，Zheng らが行った研究がある[7]．この論文で筆者らは，クラスタを構成する複数の機器の設定情報から障害の根本原因箇所を切り分ける試みを行っている．しかしこの方式では，管理対象の全ての機器で情報収集を行う必要があり，メッセージ監視型の障害検知には適さない．我々は，障害の発生検知と，発生した障害の種類の特

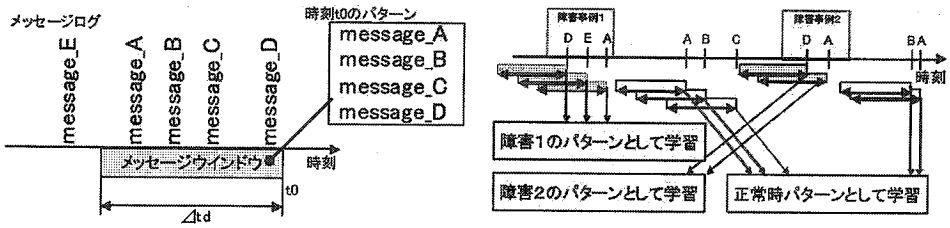

図 2 メッセージパターン学習

定を同時に行うことを目標として技術開発を行っている.

4. トラブルの特徴となるメッセージパターンの抽出

4.1 アプローチ

我々が着目したのは，メッセージログとトラブル対処の記録の関連である．これら二種類のデータは，いずれもシステム運用管理の現場で入手できる情報であり，障害の発生検知と切り分けに利用できれば，効果的な運用管理が実現できる.

しかし，メッセージログそれ自体には，障害に関係のあるメッセージ以外にも様々なメッセージが混ざっており，そのままでは障害検知のためのメッセージパターンを作成できない．また一方で，トラブル対処記録には障害の発生時刻や解消時刻が記述されているが，知識として再利用しようとした場合，症状や原因箇所を特定するための切り分け手順の記述が曖昧な場合も多い.

ある時点における過去一定期間のメッセージのパターンは，その時点におけるシステムの状態を説明するための変数であると捉えることができる．また，ほとんどのトラブル対処記録には，トラブルの発生時刻・解消時刻が記載されている．従って，メッセージパターンとトラブル発生の有無を関連づけて学習していくことで，トラブルを特徴づけるメッセージ･パターンを抽出することが可能になる.

運用時には，システムから出力されるメッセージを監視し，過去一定期間のメッセージと，ログから抽出したメッセージパターンとを比較し，トラブルの発生を管理者に通知する.

過去事例のトラブルごとにメッセージパターンを抽出しておくことにより，単にトラブルが発生したことだけでなく，検知したメッセージパターンと関連づけられたトラブルを管理者に通知することができる.

4.2 メッセージパターンの学習

4.2.1 概要

図 2 にメッセージパターン学習の概要を示す.

ある時刻 t0 において，一定時刻前(t0-⊿td)から t0 までに記録されたメッセージの集合を，t0 におけるメッセージパターンと定義する．(t0-⊿td)から t0 までの時刻をメッセージウインドウ，⊿td をウインドウ幅と定義する．ここで，障害対処記録から各障害事例の開始時刻と終了時刻を取り出し，ある障害が発生していた時刻のメッセージパターンを，それぞれの障害の「障害パターン」として学習し，そうでない時刻のメッセージパターンを「正常時パターン」として学習することで，それぞれの障害に特有なメッセージパターンを得ることができる.

4.2.2 学習

パターン学習にはベイズ推定を用いている．長い期間のメッセージログには，同一のメッセージパターンが複数回現れる．ここで，m 種類の障害についての対処記録と，n 種類のメッセージパターンを含むメッセージログがあるものとする．障害 i について，メッセージパターン j が観察されたとき，障害 i が発生している確率 $P'(S_j|Y_i)$は,

S_j：メッセージパターン j が観察されたという事象

Y_i：障害 i が発生しているという事象

N_i：障害 i が発生していないという事象

$P(S_j)$：メッセージパターン j が観察された確率

$P(Y_i|S_j)$：障害 i であったときにメッセージパターン j が観察された確率

$P(N_i|S_j)$：障害 i でないときにメッセージパターン j が観察された確率

を用いて，以下の式で表される.

$$P'(S_j|Y_i)=\frac{P(S_j)P(Y_i|S_j)}{P(S_j)P(Y_i|S_j)+P(S_j)P(N_i|S_j)}$$

右辺の P が事前確率，左辺の P'が事後確率である.

4.2.3 検知

学習したメッセージパターンは，運用管理時のメッセージと比較される．観察されたメッセージパターンと学習したn種類のメッセージパターンのいずれかが一致した場合，学習結果である$P(S_j|Y_i)$を評価し，障害の種類ごとに定めた任意の閾値α_iに対して

$$P(S_j|Y_i) > \alpha_i$$

となった場合に，障害iが発生したと判断する．

4.2.4 ノイズとなるメッセージへの対策

運用管理ログには，障害と共起関係にないメッセージも多くふくまれている．このため，メッセージログに含まれるh種類のメッセージからパターン学習を行う場合，学習の対象となるメッセージパターンの数は

$$\sum_{r=1}^{h} {}_hC_r$$

であるから，h が増えると膨大になる．また，互いに非同期にメッセージを発する多数の機器からなるシステムのログには，全く同じパターンが現れる可能性はとても低く，学習したパターンのほとんどが無駄になる．このような環境において，効率的にメッセージパターンによる障害検知を行うため，以下の手順でメッセージログから障害に関係のないメッセージを除いた上で，パターン学習を行う．

まず，メッセージログ内に現れるそれぞれのメッセージkについて，障害iとの関係を調べる．障害iについて，メッセージkが観察されたとき，障害iが発生している確率$P'(M_k|Y_i)$は，

M_k：メッセージkが観察されたという事象

Y_i：障害iが発生しているという事象

N_i：障害iが発生していないという事象

$P(M_k)$：メッセージkが観察された確率

$P(Y_i|M_k)$：障害iであったときにメッセージkが観察された確率

$P(N_i|M_k)$：障害iでないときにメッセージkが観察された確率

を用いて，以下の式で表される．

$$P'(M_k|Y_i) = \frac{P(M_k)P(Y_i|M_k)}{P(M_k)P(Y_i|M_k)+P(M_k)P(N_i|M_k)}$$

右辺のPが事前確率，左辺のP'が事後確率である．

ここで障害iについて，各メッセージとの関連を考えると，

$$P(M_k|Y_i)=0$$

となるようなメッセージkは，障害iとの関連が全くないことから，検知すべきメッセージパターンの構成要素として考慮する必要がない．

$$P(M_k|Y_i) \neq 0$$

となるようなメッセージkの集合が，障害iに関連のあるメッセージである．従って，この集合の部分集合から構成されるメッセージパターンについてのみ，障害の発生確率を算出すればよい．

これにより，現実的な数のパターンについて，パターンが観察された場合に障害である確率を求めることが可能になる．

4.3 特徴

4.3.1 本方式のメリット

本方式には，従来の手法にはない，以下のメリットがある．

(1) **監視すべきメッセージパターン定義が不要であるため，運用管理の手間が少なくて済む．**

(2) **従来のパターンマイニング的手法で実現している「正常か異常か」の判定だけでなく，発生した障害が，過去のどの障害と対応しているかを知ることができる．このため，該当するトラブルの対処記録を参照し，効果的な障害対応を行うことが可能である．**

5. 評価

5.1 評価指標

これまでに述べた障害検知手法の性能を測定するために，以下の3つの指標を導入する．これらの指標は本来，情報検索の性能を評価するためのものであるが，事象の回数を統計的に処理する類似の研究でも使われているものである[5].

- **精度*p***：全ての障害検知件数に対して，正しく障害を検知できた件数の割合．
- **再現率*r***：全ての障害発生件数に対して，正しく障害を検知できた件数の割合．
- ***F*値**：精度と再現率の調和平均．F値は0から1の間の値を取り，精度pおよび再現率rがともに1のとき，最大値の1になる．F値大きいほど検知性能が良いことを示す．

これらの値については，以下の真偽表を用いて次のように表すことができる．

表 1 分割表

		実際	
		障害発生	障害なし
検知	検知あり	真陽性(TP)	偽陽性(FP)
	検知なし	偽陰性(FN)	真陰性(TN)

表 2 各指標の定義

評価指標	定義
精度 p	$p = \frac{TP}{TP+FP}$
再現率 r(= 真陽性率 tpr)	$r = tpr = \frac{TP}{TP+FN}$
F 値	$F = \frac{2*p*r}{p+r}$

5.2 評価手法

5.2.1 評価指標の算出方法

ある期間のメッセージログと障害対処記録を使って，メッセージパターンを学習させた後，別のメッセージログを入力として，障害検知を行う．障害検知の結果は，ある時刻 t において観察されたメッセージパターンに対応する，障害発生確率（スコア）s の時系列グラフとして表現される．グラフの例を図 3 に示す．

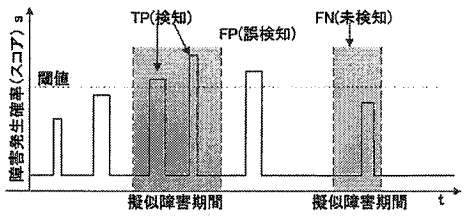

図 3 検知結果（メッセージ検知スコアグラフ）

このグラフにおいて，ある障害についての検知スコアが閾値を越えたとき，その障害が発生していた場合には，表 1 における真陽性 (TP) である，と判断する．スコアが閾値を越えたにも関わらず，その障害が発生していない場合には偽陽性(FP)である．

障害の発生期間中，一度も検知スコアが閾値を越えない場合，障害の見落としがあったものとして偽陰性（FN）と判断する．

検知用メッセージログの全てについて上記の処理を行った後，TP，FP，FN となった事象を数え，精度 p，再現率 r，および F 値を求める．

5.3 評価用データ

本方式の評価にあたっては，実際に社内システムの運用管理業務を行っている部門で記録したメッセージログと障害対処記録から評価用のデータを作成した．基となるデータの諸元を表 3 に示す．

表 3 評価用データ 諸元

項目	説明
システム規模	サーバ 270 台，ネットワーク機器 62 台
ログ取得期間	2008/12/18～2009/2/1
メッセージログサイズ	130770 行(17.9MB)
障害対処記録件数	36 件

障害対処記録には類似する障害事例が少ないので，繰り返し学習の回数を確保するために，上記のログを加工して評価用ログを作成した．加工の方法を図 4 に示す．加工にあたっては，正常期間と障害期間から乱数を用いて一定の割合でメッセージを取り出して順序を混ぜ合わせ，メッセージが発生した時刻をシフトすることで，評価用のメッセージログを作成する．今回の評価においては，本手法により，１つの障害事例から１００事例分の評価用メッセージログを得た．

5.4 評価実験

作成した評価用のメッセージログを用いて，メッセージパターン作成と障害検知の性能を評価した．評価用メッセージログから１つ以上の障害事例の期間を含む部分を学習用として取り出し，これを用いてメッセージパターンを学習した後，別の１つの事例を含む部分を取り出して，この部分のログから障害の検知を試みた．

5.4.1 パラメータ

メッセージパターンの学習を制御する変数として，ウインドウ幅⊿td と学習回数を挙げる．

学習と検知の際のウインドウ幅を，3 分から 20 分ま

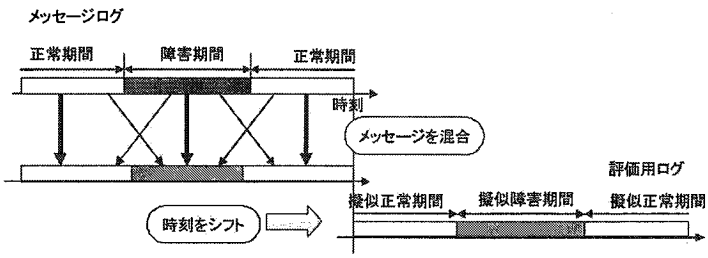

図 4 評価用データ作成方法

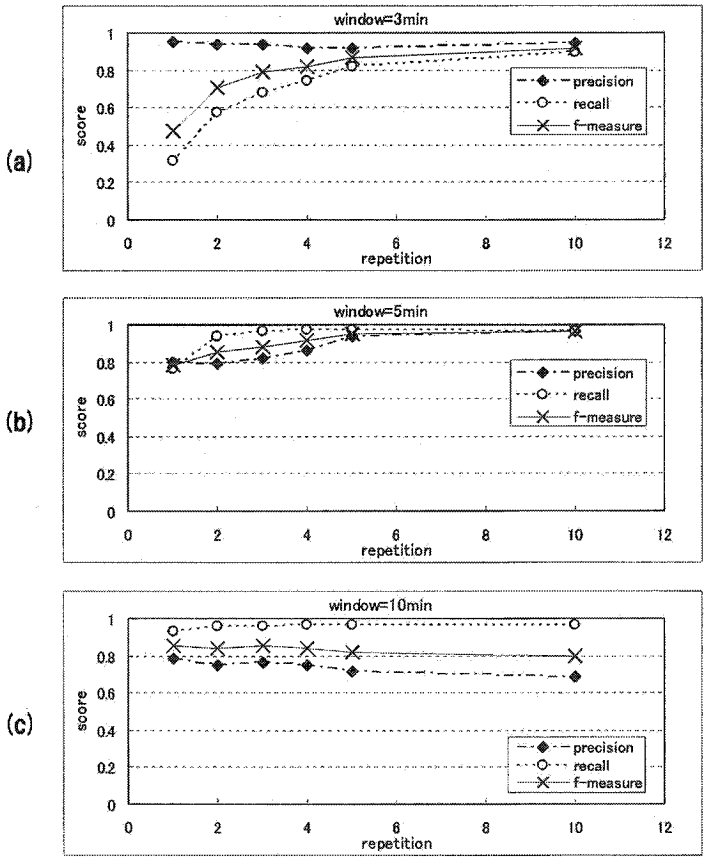

図 5　ウインドウ幅と学習回数が検知性能に及ぼす効果

で変化させつつ,繰り返し学習回数を 1 回から 10 回まで変化させた．今回の評価においては，障害発生と判定するための閾値αは 0.8 とした．

検知結果を示すスコアグラフから TP，FP，FN をカウントし，精度，再現率，およびF値を算出した．

5.5　実験結果

ウインドウ幅を３分，５分，１０分の場合の，繰り返し学習による検知性能の改善効果の測定結果を図 5 に示す．実験用データによる結果のばらつきを防ぐため，同じ条件で５回の実験を行った結果を平準化してある．

5.6　考察

5.6.1 検知性能特性

図 5 のいずれのグラフでも，繰り返し学習回数が増えると精度 p と再現率 r は一定の値に収束する．ただし，収束する値はウインドウ幅によって異なる．

ウインドウ幅が小さい場合（図 5 (a)），障害に特有なメッセージを含む短いパターンを多数学習する．このため，少ない学習でも，検知時にパターンが一致した場合，高い確率で障害が発生している（真陽性 : TP）場合が多く，表２の式より，高い精度 p を得ることができる．一方で，障害に特有なメッセージの前後のメッセージの組み合わせは多岐にわたるため，実際の検知ではマッチせず，障害発生を見逃すケース（偽陰性 : FN）が起きる．このため，少ない学習回数の再現率 r は低くなる（図 6）．繰り返し学習を行うと，上記組み合わせの多くの部分を学習していくため，再現率 r は急速に改善していく．

一方，ウインドウ幅が大きい場合（図 5 (c)），障害を特徴づけるメッセージの多くを含むパターンが学習される．このため，一度の学習でも，他の事例を検知可能なパターンを学習結果として得ることができる．この結果，障害の発生を見落とすこと（偽陰性 : FN）が減り，再現率 r が高くなる．しかし同時に，ウインドウ幅が大きくなることは，障害期間外の，障害と関係のないメッセージも多く学習することを意味する．

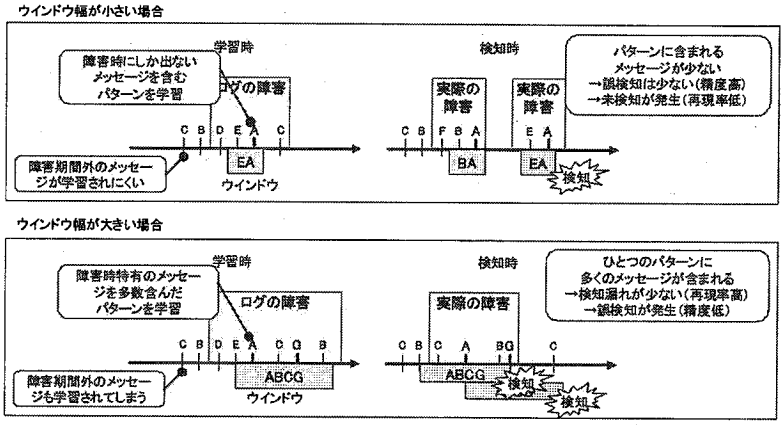

図 7　ウインドウ幅により検知特性が変化する原因

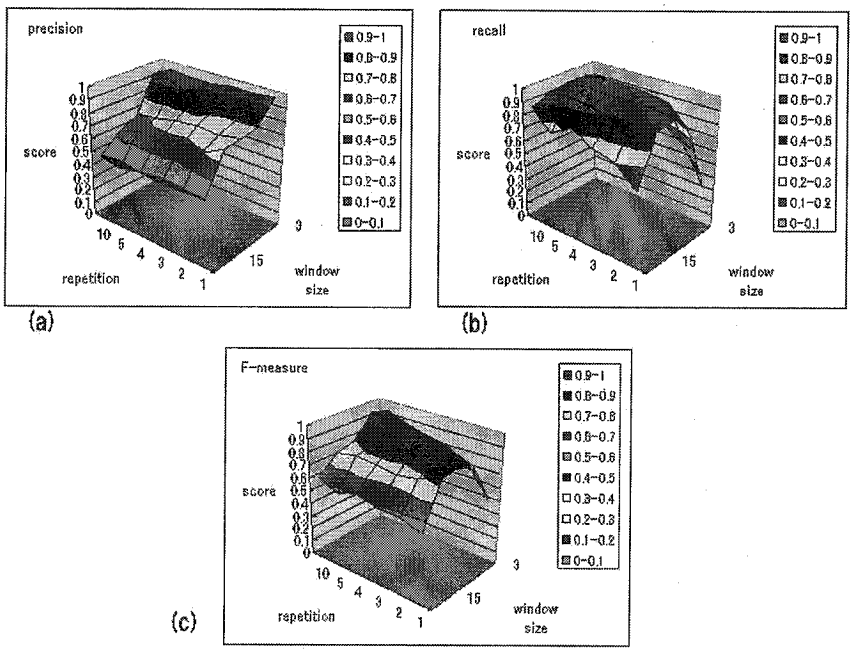

図 6　検知特性

この条件で繰り返し学習を行うと，障害と関係のないメッセージを含む多数のパターンを学習することで，誤検知（偽陽性：FP）が増え，精度 p は低下すると考えられる．

上記(a)と(c)の中間に，少ない学習回数でも高い再現率 r を持ち，繰り返し学習を行っても精度 p が低下しない領域がある（図 5 (b)）．この近傍に，今回のログの障害を最も効率良く検知できる条件があると考えられる．

最適な学習回数とウインドウ幅を求めるために，ウインドウ幅と学習回数のパラメータが，障害検知性能に対して及ぼす影響について調べた結果を，図 6 に示す．左上(a)が精度 p，右上(b)が再現率 r，そして下(c)が F 値のグラフである．各図中左下の軸が学習回数，右下の軸がウインドウサイズを表し，Z 軸が性能値を示している．(a)では，ウインドウ幅が 3 分の時，学習回数が 1 回でも精度 p が 0.9 以上であり，ウインドウ幅が 20 分のときには，学習回数をどんなに上げても 0.5 を超えることはない．この事から，精度 p には，学習回数よりもウインドウ幅の方が大きく寄与していることがわかる．再現率 r についての結果(b)では，学習回数が 1 回の時は，ウインドウ幅が大きくても小さく

ても再現率が悪化することが分かる．学習回数が 10 回のときには，ウインドウ幅による違いは減り，0.9 前後の高い値で落ち着く．

精度 p と再現率 r を総合的に見て，障害検知のフィルタとしての性能（F 値）を評価した図が(c)である．p と r の調和平均である F 値は，ウインドウ幅が 10 分，学習回数が 10 回の時に最大となり，その値は 0.96 である．

6. まとめ

運用管理の課題である迅速な障害発生検知および発生した障害の種別特定のために，障害発生時のメッセージログを学習して障害に特有なメッセージパターンを抽出し，運用中のシステムが発するメッセージと比較することで障害発生を検知するための方式を考案した．実際のログから作成した擬似障害ログを用いて，パターン学習の回数と，学習時のウインドウ幅が検知性能に及ぼす影響を評価し，最適となる学習回数とウインドウ幅の組み合わせが得られることを示した．

パターン学習による障害検知は，同様のトラブルが複数回起きたときに，学習により検知性能が向上していく．同一または類似した構成のシステムを多数用意するクラウドコンピューティングの特性を考えた場合，1 つのシステムで発生したトラブルが，他の多くのシステムでも発生することが考えられる．このため，クラウド環境では，従来の個々に固有な構成を持つシステムよりも，トラブル事例が早く蓄積され，トラブル検知精度が向上することが期待できる．

今後は障害の種類や特性によって，最適な学習回数やウインドウ幅がどのように変化するか評価した上で，多様なシステム障害の発生を柔軟かつ精度よく検知するための改善を行う予定である．

参考文献

1) Michael, A., Armando, F., Rean, G.,Anthony D. J., Randy H. K. Andrew K., Gunho, L., David A. P, Ariel A.,Ion S., and Matei, Z. 2009. Above the Clouds: A Berkeley View of Cloud Computing. UCB/EECS-2009-28 http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html

2) LTA for multievent software problem analysis http://www.ibm.com/developerworks/autonomic/library/ac-ltaanalyze/?S_TACT=105AGX90&S_CMP=content

3) Hellerstein, J. L., Zhang, F., and Shahabuddin, P. 2001. A statistical approach to predictive detection. Comput. Netw. 35, 1 (Jan. 2001), 77-95. DOI= http://dx.doi.org/10.1016/S1389-1286(00)00151-1

4) Hamerly, G. and Elkan, C. 2001. Bayesian approaches to failure prediction for disk drives. In Proceedings of the Eighteenth international Conference on Machine Learning (June 28 - July 01, 2001). C. E. Brodley and A. P. Danyluk, Eds. Morgan Kaufmann Publishers, San Francisco, CA, 202-209.

5) Salfner, F. and Malek, M. 2007. Using Hidden Semi-Markov Models for Effective Online Failure Prediction. In Proceedings of the 26th IEEE international Symposium on Reliable Distributed Systems (October 10 - 12, 2007). SRDS. IEEE Computer Society, Washington, DC, 161-174.

6) Duan, S. and Babu, S. 2008. Guided Problem Diagnosis through Active Learning. In Proceedings of the 2008 international Conference on Autonomic Computing (June 02 - 06, 2008). International Conference on Autonomic Computing. IEEE Computer Society, Washington, DC, 45-54. DOI= http://dx.doi.org/10.1109/ICAC.2008.28

7) Ziming Zheng; Yawei Li; Zhiling Lan, "Anomaly localization in large-scale clusters," Cluster Computing, 2007 IEEE International Conference on , vol., no., pp.322-330, 17-20 Sept. 2007 http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4629246&isnumber=4629185