並列分散システムにおける異常動作の原因特定のためのログ解析*

佐伯 勇樹† 田浦 健次朗‡ 近山 隆§
東京大学工学部電子情報工学科† 東京大学大学院情報理工学系研究科‡ 東京大学大学院新領域創成科学研究科§

1 背景と目的

近年、大規模な計算を可能とする並列分散計算が注目を浴びているが、プログラムのバグや設定ミス、故障などたびたび異常が発生する。これらの異常は物理的に離れたノードや時間的に離れたイベントに影響を与えることもあり、その原因特定は既存のシステムでも完全に行うのは困難であるため、そのような異常の原因を特定するための手段を用意する必要がある。そこで、情報爆発時代に向けたIT基盤技術研究のための研究プラットフォーム"InTrigger"をテストベッドとし、機械学習を用いて、各ノードのCPU使用率や通信量などのログからノードの状態分類を行い、そのノードの接続情報からどのプロセスによる異常かを特定する手法の提案と実装、そしてその評価の報告をする。

2 関連研究

既存のネットワークモニタリングシステムとしてganglia [4] が上げられる。このツールではプロセス一つ一つのデータに関しては情報を得ることができず、また統計量だけからでは異常原因特定につながりにくいことが多々ある。またネットワーク上のWebサーバ数台による分散環境の状態を分析する方法として、Cohenらによって Tree-Augmented Naive Bayes (TAN) を用いた手法が提案されている。Cohenらはサーバの応答時間を正常と異常の分類基準とし、TAN [3] を用い、サーバのCPU時間、ネットワーク量などの統計量から、状態推定を行った [1]。そして調査したいシステムの各統計量をTANで2値に特徴づけ、クラスタリングする実験を行い [2]、異常状態の特徴を見つけることに成功した。問題としては数百といった大規模環境ではベクトルの次元数が大きくなりすぎることが挙げられる。

3 本研究の手法

本研究は、各マシンから得た統計量やプロセス情報でのクラスタリングを大規模環境で行う方法を提案し、その結果からマシンの異常状態の種類を調べ、その原因を状態分類とそのプロセス情報から推測する。

3.1 実験環境と解析を行うデータ

今回評価する並列分散システムは、前述のInTriggerを用いた。解析を行うデータとして、マシンの状態を表すCPU使用率、ディスクアクセス回数といった各種統計量などを /proc, vmstat, ps, netstat, iptraf から取得しログに記録する。

3.2 手法の概要

状態のクラスタリングは、ある一時点の1分間の統計量のベクトルをクラスタリングすることで行うが、各々のマシンは動いているプロセスにより統計量の変動に特徴があり、正常でも変則的な値をとったり、異常でも一見正常な値に見えることがある。そこで、Cohenらの2値変換の手法を大規模な環境向けに修正を加えて用い、異常原因となっている統計量を明確にする。まず、2値変換にはナイーブベイズ、TANの2種類の学習構造を事前に蓄えたログから学習し、適したものを用いる。

ここで用いるナイーブベイズとは、各々の変数がクラスに依存しており、ある時点での統計量ベクトル $\boldsymbol{x}$ が $\boldsymbol{m}$ であったときに、その状態クラスを最も確率 $P(s|\boldsymbol{x}=\boldsymbol{m})$ の高いクラス s に分類する手法である。その確率は次のようになる。

$$P(s|\boldsymbol{x}=\boldsymbol{m}) \propto \prod_{i=1}^{n} P(x_i = m_i|s)\, P(s)$$

ここで確率分布 $P(x_i = m_i|s)$ は正規分布を仮定し、学習データの平均、分散から求める。

TAN [3] とはナイーブベイズの学習構造において、各々の変数はクラス以外に最大1つの異なる変数に依存しているとする学習構造である。ただし依存関係に閉ループはないものとする。これは、各統計量間の依存関係の強さを表す相互情報量をエッジ、各統計量成分をノードとするグラフGの最大全域木を求める問題に帰着できる。

手順は次のようになる。

まずは学習用ログデータを、ssh応答が速いか遅いか、ping応答が速いか遅いか、loadaverageが変に大きいか正常な値であるかなど、閾値によって単純に正常クラスと異常クラスの2つに分ける。このクラス分類をナイーブベイズ、TANで学習する。

次いで調べたいログデータを2値変換する。統計量の2値変換に使う学習構造の選び方は、分類したい時点 t の近辺 w の正常、異常ラベルをよりはっきり推定できる学習構造をBrier Scoreという基準によって判定して選び、2値変換は選んだ学習構造での単純な確率比較によって、正常時にとる確率が高いなら +1、そうでない場合 −1 にする。

*Log Analysis for Finding Trouble Sources in Distributed Systems.
†Yuki Saeki, Department of Information and Communication Engineering, Faculty of Engineering, The University of Tokyo
‡Kenjiro Taura, Graduate School of Information Science and Technology, The University of Tokyo
§Takashi Chikayama, Graduate School of Frontier Sciences, The University of Tokyo

さらに調べたいシステムの状態群を各要素が2値のベクトルで表したら、これをクラスタリングする。ここではkmeans法を採用した。ある程度クラスタリングされたデータが集まった後は、各クラスタの中心を求めておき、以降の分類はどの中心に近いか、もしくはどの中心ともかけ離れている新たな異常であるかで状態分類することができる。状態分類を行う環境を50〜100台の並列分散環境に適応させるため、ここでは同一時点のそれぞれのマシンの状態を別々に集める。時刻tの状態データは分散環境のマシンの台数分集まることになり、それを調べたい区間で集め、まとめてkmeansクラスタリングを行う。また、NFSサーバと計算ノードは別々にkmeansを行うとする。

最後に状態分類をした後は、netstatコマンドなどのネットワーク接続情報、psコマンドの出力などのプロセス情報からその状態とかかわりの深いアクティブであるプロセスを抽出し、原因を判定する。具体的には、プロセスグループごとにプロセスをまとめ、同一プロセスグループIDを持つプロセスが極端に多い(50〜100以上)グループ、CPU使用率やメモリ領域に変動のあるグループ、STATがR、Dになっているプロセス数の多いグループ、ネットワーク通信量の多いグループなどである。

4 評価

今回の評価の一例として、84の計算ノードと1つのNFSサーバからなるhongoクラスタ環境で起こった異常を取り上げる。

1月6日から8日の間、あるユーザのプロセスの異常により、NFSサーバに大きな負荷がかかった。この区間でのデータを収集し、クラスタリングを行った。

クラスタリングに用いた統計量は、メモリの空き容量、割り込み回数、コンテキストスイッチ回数、CPUユーザ時間、CPUシステム時間、IO待ち時間、ディスク読み込み回数、ディスクアクセス回数、ネットワークデータ送信量、ネットワークデータ受信量の10項目で、正常と異常の大まかな分類基準としてloadaverageを閾値5.0で用いた。

まず学習データとして計算ノード一つとNFSサーバの正常と異常の混じった1日分のデータを学習させ、続いて、この学習データを用い、すべてのノードの3日間のデータを2値変換し、ノード群とNFSサーバとで別々にkmeans法を計算ノードではk=7で、NFSサーバではk=10で実行した。すると、次のことがわかった。

- 通常時は計算ノードが状態2か6、NFSサーバが状態4になった。
- 6日の15：30近辺で1〜69番の計算ノードが状態3に、0番の計算ノードが状態0と3をふらつく状態になった。
- 6日の19：00近辺でで1〜69番ノードは状態2に戻るが、0番ノードのみ状態4, 5をふらつく状態へと遷移した。
- 7日から8日で、0番ノードは状態4, 5をふらつく状態と状態1, 4, 5をふらつく状態を繰り返した。
- 7日から8日で、NFSサーバは状態0, 2, 3, 7, 8をふらついていたが、時間帯によって状態2が多い場合と状態8が多い場合が存在した。それぞれ0番ノードが状態4, 5をふらついていたときと状態1, 4, 5をふらついていたときに対応していた。

プロセス情報を調べると、6日15：30〜19：00に0〜69番ノードで同一名のプロセスが立ち上がっていたが、うち0番ノードのみそれ以降もそのプロセスが残り続け、30以上にforkしIO待ち状態になっていた。また状態の遷移に伴いIO待ちの数が変動していた。さらに、NFSサーバではNFSデーモンがそれに対応して大きく負荷をかけているのがわかった。

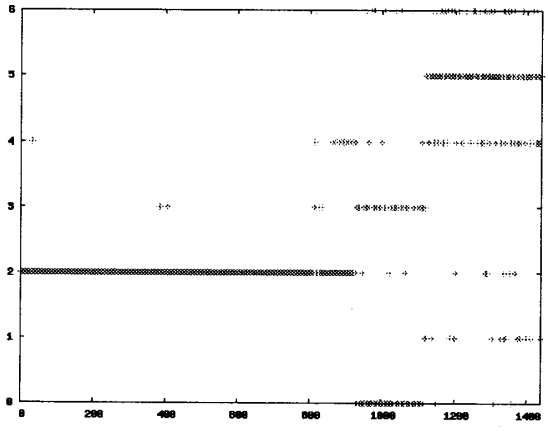

図. 0番ノードでの1月6日のクラスタリングの結果

5 まとめと今後の課題

各ノードごとの状態を別々に集めてクラスタリングすることにより、マシンの状態遷移と原因プロセスを追いかけることを可能にできたが、状態のふらつきが多いために、完全な自動判断にはまだ改良の必要がある。分類に用いる統計量の選択、原因プロセスの学習、十分な学習量をえることが今後の課題である。

参考文献

[1] I. Cohen, M. Goldszmidt, T. Kelly, J. Symons, J. S. Chase. Correlating instrumentation data to system states : A building bloc for automated diagnosis and control. In Proc. 6th USENIX OSDI, San Francisco, CA, Dec. 2004.

[2] I. Cohen, S. Zhang, M. Goldszmidt, J. Symons, T. Kelly, A. Fox. Capturing, Indexing, Clustering, and Retrieving System History. In SOSP'05.

[3] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers In Machine Learning, vol. 29, pp. 131 163, 1997.

[4] Ganglia Monitoring System. http://ganglia.sourceforge.net/