doi:10.3772/j.issn.1002-0470.2016.06.001

一种基于日志结合分析的集群系统失效预测方法①

付晓毓②* ** *** 任 睿* ** 詹剑锋* ** 孙凝晖* **

(* 计算机体系结构国家重点实验室 北京 100190)

(** 中国科学院计算技术研究所 北京 100190)

(*** 中国科学院研究生院 北京 100049)

摘 要 研究了大规模超级计算机群系统的失效预测。针对现有的单一分析系统日志的方法不仅需要复杂的分布式数据挖掘技术,而且失效预测的召回率普遍不高的问题,提出了一种通过将记载集群系统负载变化的作业日志同系统日志结合起来分析来进行失效预测的方法。该方法的原理如下:首先,通过对原始系统日志和作业日志进行预处理和过滤,分别得到细粒度的二维的事件序列和作业序列;然后从中抽取出在系统日志的失效事件发生前作业日志所表现出的三种典型失效征兆;最后,利用失效征兆进行失效预测。在真实的 IBM BlueGene/P 系统的系统日志和作业日志上的实验结果表明,基于两种日志结合分析的方法能以较高的准确率和召回率实现细粒度的失效预测。

关键词 大规模集群系统, 系统日志, 作业日志, 日志分析, 失效预测

0 引 言

大规模集群系统是云计算和高性能计算的常用平台。随着越来越多传统的社交和经济活动被搬到互联网上,数据中心的计算任务变得日益复杂,超级计算机也正朝着百亿亿次级(exascale)的规模发展,相应地,大规模集群系统的失效(failure)则成了大规模集群系统的常态[1]。可靠性与功耗、并发、存储等问题一起成为了实现超大规模集群系统的巨大挑战。由于系统日志(system log)详细记录了集群系统发生状态变化的各种事件,能够长期地比较精确地追踪系统行为,因而对系统日志的分析是进行失效预测的关键。失效预测是一种利用系统的历史状态信息预测不久的将来系统是否会再次发生失效的有效的可靠性主动管理和失效预防机制。其中,通过分析海量的历史系统日志事件来预测失效事件是否会在未来某个时间点发生的失效预测方法成为近几年该领域研究的热点。然而,通过单一分析海量系统日志来进行失效预测往往需要复杂的并行数据挖掘技术,且失效预测的召回率普遍不高[2-5]。本研究考虑到集群系统负载的变化与集群系统失效往往有着密切的联系,提出了一种通过记载集群系统负载变化的作业日志(Job log)和系统日志结合起来处理、分析来实现失效预测的有效方法,并通过实验验证了该方法的有效性。

1 相关工作

在对集群系统进行主动失效管理的机制中,失效预测已经被国内外许多工作证明是实现集群系统可靠性的有效手段。文献[6]将失效预测的方法分为两大类:第一类是通过采集系统参数并分析系统参数周期性变化规律的特征采集分析方法;第二类

① 863 计划(2015AA015308)和 973 计划(2014CB340402)资助项目。

② 女,1985 年生,博士生;研究方向:分布式数据挖掘,集群系统可靠性和性能分析;联系人,E-mail: fuxiaoyu@ncic.ac.cn

(收稿日期:2016-03-14)

是通过追踪系统事件形成 RAS(reliability——可靠性,availability——可用性,and serviceability——可服务性)日志并分析日志的日志收集分析方法。文献[7]证明基于日志的方法更适用于实现大规模集群系统的可靠性。

在日志分析领域,日志预处理的工作[1,8]能够精简日志内容并去除冗余记录以利于后续的日志分析。文献[9]通过分析大量高性能计算(high performance computing,HPC)日志发现了一些简单的失效事件的分布特征。后续工作试图利用统计分析来揭示失效事件的行为模型以进行失效预测。例如,文献[10]研究了失效事件的统计特征以及失效事件和非失效事件之间的相关性来进行失效预测。文献[11]首先利用球面协方差模型和随机模型分别量化失效事件在时间上和空间上的相关性,然后对失效事件聚类得到失效事件发生的时间间隔,据此进行失效预测。文献[12]利用信号分析的概念模拟事件在非失效和失效状态下的行为以此进行失效预测。

文献[13]指出基于模型的方法无法适应 HPC 系统越来越高的复杂性,于是越来越多的工作采用基于数据挖掘的日志分析方法。文献[4,13]采用了将统计分析和关联规则挖掘相结合的方法得到失效模式进行失效预测。文献[2]采用了基于 Apriori 的算法对日志进行挖掘,并提出了基于事件关联图的失效预测方法。文献[3]也采用基于 Apriori 的算法挖掘事件序列,并利用动态窗口对日志进行划分。虽然以上方法在失效预测上可以得到 70% 以上的准确率,而召回率却不足 50%。文献[14]在提高召回率上采用了抽取事件因果关系的方法,但其依赖于前一阶段复杂的数据挖掘所得到的事件关联序列。由于挖掘非失效事件和失效事件的关联需要巨大的时空开销,现有方法往往采用分布式挖掘的方式处理[15,16]。因此,不论是基于数据挖掘还是基于简单统计的方法,仅仅依靠从 RAS 日志中获得失效的征兆进行失效预测显然是不足的。

Job 日志记载了集群系统上负载运行的详细信息,而许多工作也已经指出负载的类型、负载的数量和负载的变化与系统是否失效有很大的关联。例如,文献[17]指出变化的负载(负载的大小和类型、负载的周期、队列和调度方法以及所需的节点数)影响了失效事件的行为。文献[9]强调密集的负载下往往会有较高的失效事件发生率。文献[18]指出正确执行相同负载的系统往往会产生相同的日志。文献[19]指出需要在大量节点上运行的 Job 类型更可能引起失效事件的发生。文献[20]指出集群上负载的强度增大时更多的失效事件将会发生。所以,本文拟利用 Job 日志中提供的 Job 运行时信息,并结合 RAS 日志对集群系统进行失效预测。

2 基于 RAS 日志和 Job 日志的失效预测

基于 RAS 日志和 Job 日志的失效预测流程如图 1 所示。首先对原始日志进行预处理和过滤,然后从中抽取出有价值的征兆形成失效规则,最后进行失效预测。

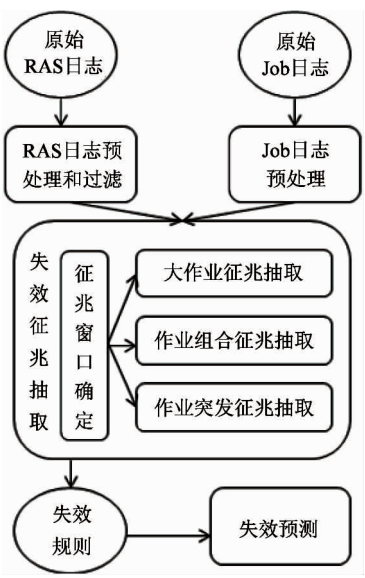

图 1 基于 RAS 日志和 Job 日志的失效预测流程

2.1 系统和日志简介

IBM Blue Gene/P (蓝色基因系列)是一个可扩展到 80 个机箱的低功耗大规模并行处理系统,其中每一个机箱有两个中间背板(midplane),总共包括了 512 个拥有四核 PowerPC450 处理器的计算节点,一个服务节点和四个网卡。有关更多系统结构的细节可以参考文献[21]。

RAS 日志:BlueGene/P 的核心监视控制系统(core monitoring and control system,CMCS)用来实时监视计算节点、I/O 节点和节点间的网络。CMCS 以日志事件记录的形式汇报监控信息。事件记录都被存储在后端的 DB2 数据库中。RAS 日志中事件记录的例子如图 2 所示。

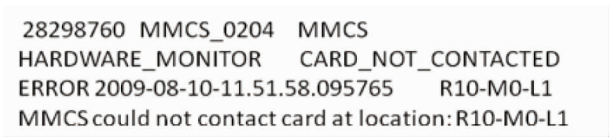

图 2 RAS 日志事件记录的例子

下面按照图 2 中各个域的顺序依次进行介绍。

RECID(Record ID)域是日志中事件记录的序列号。当有一个新的记录被记载到日志中时,序列号加 1。

MSG _ ID(Message ID)域指示了该事件记录的类型。

COMPONENT 域包括 APPLICATION,KERNEL,MC,MMCS,BAREMETAL,CARD,DIAGS,指示了监测和汇报该事件记录的软件部件类型。具体地,APPLICATION 指运行的作业;KERNEL 指操作系统内核;MC(Machine Control System)指机器控制系统;MMCS(Midplane Monitoring and Control System)指服务节点上的控制系统;BAREMETAL 指与服务相关的功能;CARD 指网卡控制器;DIAGS(Diagnostic functions)指计算节点或服务节点的监测功能。

SUBCOMPOENT 域指示了相应的软件部件 COMPONENT 记录和汇报事件的具体部件。

ERRCODE(Error Code)域指示了与事件类型 MSG _ ID 对应的具体的事件类型信息。

SEVERITY 域包括 INFO(Information),WARNING,ERROR 和 FATAL,按照严重程度递增来指示事件的等级。INFO 级事件提供了系统软件的一些运行时信息。WARNING 级事件通常指一些可自行恢复的小错误。ERROR 级事件指一些有危害但不影响应用继续运行的错误。FATAL 级事件指会导致应用或系统崩溃的失效事件。所以,失效预测的关键是能否提前预测该类型事件会在未来发生。

EVENT _ TIME 域指示了事件的发生时间。

LOCATION 域指示该事件发生在哪个节点。

MESSAGE 域给出了该事件的一个简略总结。

Job 日志:该日志是由作业调度器 Cobalt[22] 收集的。Job 日志记录的例子如图 3 所示。下面按照图 3 中各个域的顺序依次进行介绍。

Submission Time 域是该 Job 提交的时间。

Job _ ID 域是该 Job 的序列号。

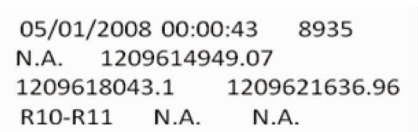

图 3 Job 日志作业记录的例子

Execution File 域是该 Job 的执行路径。

Queuing Time 是域该 Job 入等待队列的时间。

Starting Time 域是该 Job 开始执行的时间。

End Time 域是该 Job 运行完或被中断的时间。

Location 域是该 Job 执行时所在的节点。

User 域是该 Job 的用户名。

Project 域是该 Job 的工程名。

2.2 RAS 日志预处理和过滤

由于 RAS 日志的每条事件记录由 RECID 等 9 个域组成,所以有如下定义:

Event:由 9 元组(RECID,MSG _ ID,COMPONENT,SUBCOMPONENT,ERRCODE,SEVERITY,EVENT _ TIME,LOCATION,MESSAGE)表示。RAS 日志可以看作一个包含 n 个 Events 的序列,其中 $n \in N^*$。

Event ID:对于一个 Event,如果其对应 9 元组中的两个域(MSG _ ID,SEVERITY)与已产生的 Events 的都不同,则赋予该 Event 一个新的 Event ID。Event ID 表示了该 Event 的类型。

Log ID:对于一个 Event,如果其对应 9 元组中的 3 个域(MSG _ ID,SEVERITY,Location)与已产生的 Events 的都不同,则赋予该 Event 一个新的 Log ID。RAS 日志可以看作一个包含 m 个 Log ID 的序列,其中 m, $n \in N^*$, $m < n$。

通过上述定义,在日志预处理阶段可以首先将原始 RAS 日志解析成一个 Log ID 序列,其中每一个

Log ID 代表了一个具有细粒度 9 元组信息的 Event。原始 RAS 日志包含庞大的日志事件,而其中有些事件在短时间内被重复记录了很多次,导致产生了大量冗余。这里采用一个简单的统计算法来过滤掉重复的日志事件。对于一个日志事件,统计在该事件发生前与该事件 Log ID 相同的事件与该事件的时间间隔。若该时间间隔小于预设的阈值 σ_filter,则该事件被认为是重复事件并从 Log ID 序列中将其删除。

RAS 日志过滤后,利用一个滑动的时间窗口(比如一个小时)来划分 Log ID 序列。如图 4 所示,将发生时间落入某一时间窗口的所有 Log ID 组成一个日志事务(Log Transaction, LT),则原始 RAS 日志转化为一个 Log Transaction 序列。需要注意的是,虽然每一个 Log Transaction 里的事件间是有发生先后顺序的,但这里简便地认为它们都是在该 Log Transaction 对应的时间段内发生的。

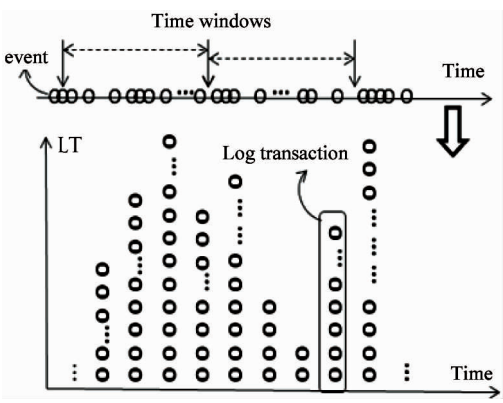

图 4 用时间窗口将事件日志划分成 Log Transaction 序列

2.3 Job 日志预处理

由于 Job 日志的每条日志记录是由 Job _ ID 等 9 个域组成,所以有如下定义:

Job:由 9 元组(Job ID, Submission Time, Execution File, Queuing Time, Starting Time, End Time, Location, User, Project)表示。Job 日志中可以包含 n 个 Jobs,其中 $n \in N^*$。

Job ID:对于日志中所有具有相同 Execution File 的 Jobs 赋予一个共同的 Job ID。则 Job 日志可以包含 m 个 Job ID,其中 m, $n \in N^*$, $m < n$。

Job 生存时间:每一个 Job 都会在 Job 日志的时间跨度内存在一小段自己的运行时间,我们称之为 Job 存在时间,用二元组(Starting Time, End Time)表示。

通过上述定义,在日志预处理阶段首先将原始 Job 日志中每一个 Job 分配一个 Job ID,其中每一个 Job ID 代表了一个具有细粒度 9 元组信息的 Job。由于 Job 日志精确记录了集群上 Job 的运行情况,所以不需要过滤。然后,利用一个滑动的时间窗口(比如一个小时)来划分 Job 日志。如图 5 所示,将存在时间落入某一时间窗口的所有 Job ID 组成一个 Job Transaction(JT),则原始 Job 日志转化为一个 Job Transaction 序列。需要注意的是,如果某一 Job 的存在时间跨越了多个时间窗口,则将该 Job 划分到对应的多个 Job Transaction 中。

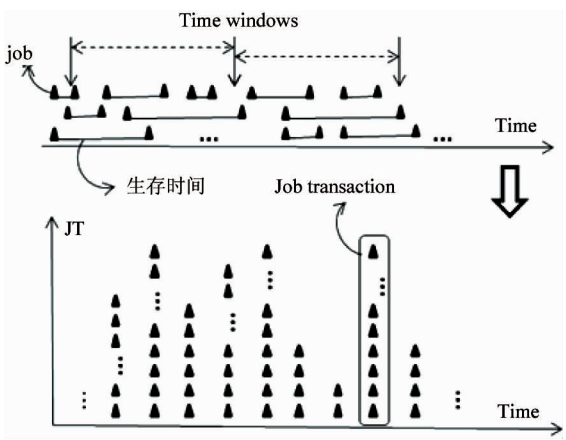

图 5 用时间窗口将作业日志划分成 Job Transaction 序列

2.4 失效征兆抽取

结合 Job 日志和 RAS 日志的失效预测基于 RAS 日志中失效事件发生前 Job 日志中负载作业所表现出的征兆。如果这种 Job 征兆先于某一失效事件发生的现象多次出现,则将其抽取出来形成失效规则用于预测失效。

2.4.1 征兆窗口的确定

要确定某一失效事件发生的 Job 征兆,一一判断生存时间处于该失效事件发生前的所有 Jobs 是没有必要的。如果 Job 征兆与对应失效事件的时间间隔大于有效预测窗口 Win_valid,失效预测是没有意义的。所以,只需要判断生存时间处于失效事

件有效预测窗口范围内的 Jobs。一方面,有效预测窗口参数不能选取太小,否则被排除的 Jobs 过多会不利 Job 征兆的抽取且管理员将没有足够的时间处理马上要发生的失效;另一方面,有效预测窗口参数不能选取太大,否则失效事件何时发生就很难判断。

由于 Job 征兆和对应的失效事件发生的时间间隔往往是不固定的,这里需要生成一个动态时间窗口,称为该失效事件的真征兆窗口。对于某一失效事件,其真征兆窗口从该失效事件发生的时间追溯到同一节点上的时间上最靠近的一个失效事件的发生时间。如果有效预测窗口和真征兆窗口发生冲突,则选取其中较小的一个作为该失效事件的征兆窗口。对于一个失效事件,其在日志中的每一次出现都对应一个征兆窗口。

2.4.2 三种征兆抽取

一些已有工作定性地指出了集群上负载的行为和失效事件之间存在关联。这些负载的行为包括运行负载的类型、运行负载的数量和运行负载的变化。

首先,文献[19]已经指出需要在大量节点上运行的 Job 类型更可能引起失效事件的发生,这里将其称为大作业征兆。在进行征兆抽取前,首先将生存时间处于某一失效事件征兆窗口内的所有作业进行 Job 过滤,即将与该失效事件不在同一节点发生的作业过滤掉。对于 h 个失效事件 $\{F_1,\cdots,F_h\}$,某一失效事件 F_i 和其征兆窗口内经 Job 过滤后的 m 个剩余作业 $\{J_1,\cdots,J_m\}$,作业 J_j 与失效事件 F_i 的相关性用提升度表示为

$$lift_1(J_j,\ F_i) = \frac{P(J_j,\ F_i)}{P(J_j)P(F_i)} \quad (1 \leqslant j \leqslant m,\ 1 \leqslant i \leqslant h) \tag{1}$$

其中,$P(F_i)$ 是失效事件 F_i 在 Log Transaction 中出现的概率,等于包含该失效事件的 Log Transaction 的个数除以 Log Transaction 的总个数。$P(J_j)$ 是负载作业 J_j 在 Job Transaction 中出现的概率,等于包含该负载作业的 Job Transaction 的个数除以 Job Transaction 的总个数。$P(J_jF_i)$ 等于负载作业 J_j 恰好会在失效事件 F_i 的征兆窗口内出现的总次数除以 Job Transaction 的总个数。如果 $lift > 1$,则 J_j 与 F_i 正相关,形成失效规则:

$$J_j \rightarrow F_i \tag{2}$$

其次,文献[17]已经指出集群上运行不同的作业类型会经历不同的失效事件,这里将其称为不同作业组合征兆。在判断不同作业的组合与失效事件的相关性前,首先将 $P(F_i \mid J_j) = P(J_j,\ F_i)/P(J_j)$ 小于置信度阈值0.6的作业 J_j 排除掉以减小组合规模。对于 h 个失效事件 $\{F_1,\cdots,F_h\}$ 中某一失效事件 F_i 和其征兆窗口内经 Job 过滤且未被排除的 n 个剩余作业 $\{J_1,\cdots,J_n\}$,不同作业的组合 $\{J_{j_1},\cdots,J_{j_r}\}$ 与失效事件 F_i 的相关性用提升度表示为:

$$lift_2((J_{j_1},\cdots,J_{j_r}),\ F_i) = \frac{P(J_{j_1},\cdots,J_{j_r},F_i)}{P(J_{j_1},\cdots,J_{j_r})P(F_i)} \quad (1 \leqslant j_1,\ \cdots,\ j_r \leqslant n,\ 1 < r \leqslant n \text{ 且 } j_1 \neq \cdots \neq j_r) \tag{3}$$

其中,$P(J_{j_1},\cdots,J_{j_r})$ 是负载作业组合 $\{J_{j_1},\cdots,J_{j_r}\}$ 在 Job Transaction 中出现的概率,等于同时包含这些负载作业的 Job Transaction 的个数除以 Job Transaction 的总个数。$P(J_{j_1},\cdots,J_{j_r},\ F_i)$ 等于作业组合 $\{J_{j_1},\cdots,J_{j_r}\}$ 恰好会在失效事件 F_i 的征兆窗口内出现的总次数除以 Job Transaction 的总个数。如果 $lift > 1$,则 $\{J_{j_1},\cdots,J_{j_r}\}$ 与 F_i 正相关,形成失效规则:

$$\{J_{j_1},\cdots,J_{j_r}\} \rightarrow F_i \tag{4}$$

另外,文献[20]已经指出集群上负载的强度增大时更多的失效事件将会发生,这里将其称为作业突发征兆。单位时间内作业个数的平均值 $\overline{N_{JT}}$ 等于所有 Job Transaction 中的作业个数的平均值。对于 h 个失效事件 $\{F_1,\cdots,F_h\}$ 中某一个失效事件 F_i 和其征兆窗口内 l 个 Job Transaction $\{JT_1,\cdots,JT_l\}$,某一个 JT_k 中作业的个数 N_{JT_k} 与 $\overline{N_{JT}}$ 之差除以 N_{JT_k} 为该 Job Transaction 的突发率,用 R_{JT_k} 表示。若 R_{JT_k} 满足

$$R_{JT_k} = \frac{N_{JT_k} - \overline{N_{JT}}}{N_{JT_k}} > \sigma\_r \quad (1 \leqslant k \leqslant l,\ 0 < \sigma\_r < 1) \tag{5}$$

则记录在 JT_k 中发生但未在 JT_{k-1} 中发生的作业集合 $\{J_1,\cdots,J_s\}$,其中 σ_r 是预设的突发率阈值。将满足式(5)条件的该 Job Transaction 表示为 $JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\}$。则 $JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\}$ 与失效事件 F_i 的相关性用提升度表示为

$$lift_3(JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\},F_i) =$$

$$\frac{P(JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\},F_i)}{P(JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\}P(F_i))} \quad (1 < s < N_{JT_k}) \quad (6)$$

$P(JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\})$ 是 $JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\}$ 在所有 Job Transaction 中出现的概率,等于该 Job Transaction 的个数除以 Job Transaction 的总个数。$P(JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\},F_i)$ 等于 $JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\}$ 恰好会在失效事件 F_i 的征兆窗口内出现的总次数除以 Job Transaction 的总个数。如果 $lift > 1$,则 $JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\}$ 与 F_i 正相关,形成失效规则:

$$JT_k\{R_{JT_k},\{J_1,\cdots,J_s\}\} \rightarrow F_i \quad (7)$$

2.5 基于失效征兆的细粒度失效预测

在仅仅依靠 RAS 日志进行失效预测的工作[2,3]中,失效预测算法在预测阶段需要不断分析系统收集的大量 RAS 日志,从中找到与失效规则匹配的事件序列,辅以置信度给出预测结果。而我们的方法实现了从简单的作业运行行为信息中得到失效征兆,将其作为失效规则存储在规则表里,用于在失效预测时进行匹配查询。由于每一个由 Log ID 表示的失效事件都含有细粒度的 9 元组信息,所以这里实现了能够锁定失效事件发生节点和失效事件类型等信息的精确的失效预测。

对于大作业征兆,只要集群系统中出现与失效规则匹配的需要很多个节点参与运行的大作业出现,则失效以置信度 $lift_1 \times P(F_i)$ 的概率发生。

对于不同作业组合征兆,只要集群上运行与失效规则匹配的作业组合,则失效以置信度 $lift_2 \times P(F_i)$ 的概率发生。

对于作业突发征兆,只要集群上运行的作业单位时间内个数和类型的变化与失效规则匹配,则失效以置信度 $lift_3 \times P(F_i)$ 的概率发生。

3 实 验

本部分将按照前一部分介绍的方法对 240 天的 Intrepid 系统的 RAS 日志和 Job 日志进行分析,并从中得到的失效规则进行失效预测。Intrepid 是美国阿贡国家实验室的一个由 40 个 Blue Gene/P 机箱组成的系统,为美国能源部服务。其总共包括了 40960 个计算节点,每秒峰值速度可达到 556 万亿次浮点运算次数。表 1 给出了两个日志的基本信息。两个日志的前 2/3 用来抽取失效征兆,后 1/3 用来失效预测。

表 1 RAS 日志和 Job 日志

内容	RAS Log	Job Log
大小	1.1GB	125M
记录个数	2083965	68936
天数	240	240
开始日期	2009-1-5	2009-1-5
结束日期	2009-8-31	2009-8-31

3.1 日志预处理和过滤

首先对原始 RAS 日志中具有相同(MSG _ ID, SEVERITY, Location)的事件记录赋予同一个 Log ID,对原始 Job 日志中具有相同 Execution File 的作业赋予同一个 Job ID,处理后得到 72187 个 Log ID 和 9664 个 Job ID。

对由原始 RAS 日志生成的 Log ID 序列进行去重时,重复记录的时间间隔若小于阈值,则删除这个重复记录。考虑到日志中重复事件的时间跨度分布,这里阈值选取 30 秒钟,去重后得到 1746568 个记录。

由于 93% 以上的作业是在 5 个小时内完成的,所以滑动时间窗口取 5 个小时。则 Log Transaction 和 Job Transaction 的个数均为 1152。其中,Job Transaction 的单位窗口内的作业均值 $\overline{N_{JT}}$ 为 131。以上实验结果如表 2 中所示。

表 2 RAS 日志和 Job 日志

内容	RAS Log	Job Log
Log ID/Job ID 个数	72187	9664
Event ID 个数	222	——
failure Log ID 个数	7259	——
原始记录个数	2083965	68936
去重后记录个数	1746568	——
LT/JT 个数	1152	1152
$\overline{N_{JT}}$	——	131

3.2 征兆抽取和失效预测

在基于失效征兆的细粒度的失效预测方法中,采用准确率和召回率两个指标来评价预测的效果。准确率 *precision* 是预测正确的失效事件个数除以预测出的失效事件总个数:

$$precision = \frac{TP}{TP + FP} \tag{8}$$

召回率 *recall* 是预测正确的失效事件个数除以失效事件总个数:

$$recall = \frac{TP}{TP + FN} \tag{9}$$

其中,*TP* 是预测正确的失效事件个数。*FP* 是预测发生但实际未发生的失效事件个数。*FN* 是未预测出但实际发生了的失效事件个数。

在确定征兆窗口时,为了保留更多的 Job 征兆又不影响失效预测的时效性,应合理选取有效预测窗口的大小。由于集群系统上的失效率往往呈现以周为单位的周期性,即周一到周五失效率比较高,而周末会降低[9]。所以,这里有效预测窗口参数 *Win_valid* 分别取 24h(1d)、48h(2d)、72h(3d)、96h(4d)和 120h(5d)并进行比较。在抽取作业突发征兆时,为了比较合理判断单位时间内作业突然增加对失效事件的影响,突发率阈值 σ_r 选取0.5。由于 BlueGene/P 系统节点规模较大,某些失效事件在 RAS 日志中仅仅出现了一两次,不足以满足失效征兆所必须的提升度条件,而已有工作[14]中指出具有相同 event ID 的失效事件具有相似的发生行为。所以,这里首先将具有相同 event ID 的失效事件分为一组(总共得到了 85 组),然后将每一组中的每一个失效事件的失效征兆作为该组中所有失效事件的失效征兆进行间接失效预测。

图 6 和图 7 列出了分别在 5 个不同有效预测窗口下直接利用 3 种失效征兆(用情景 1 表示)和间接利用 3 种失效征兆(用情景 2 表示)进行失效预测的准确率和召回率。首先,在直接失效预测中,3 种失效征兆下的准确率普遍都比较高,但召回率普遍都不高。其中,作业突发征兆下的准确率相对最高,大作业征兆下的召回率相对最低。其次,在情景 1 和情景 2 两种情景下,缩小有效预测窗口虽然对准确率影响不大,但是会显著减少失效征兆的抽取进而降低召回率。其中,基于不同作业组合征兆的失效预测召回率降得最多。另外,通过对不同有效预测窗口下准确率和召回率的对比,可以发现在窗口达到 72h 后,召回率的提高开始变得缓慢。如果对失效预测时效性要求比较高,那么设置 72h(3d)的有效预测窗口将能同时保证较高的准确率和召回率。最后,与直接失效预测相比,间接失效预测虽然可以提高召回率,但准确率也会随之降低。其中,作业突发征兆下的召回率提高最多,同时准确

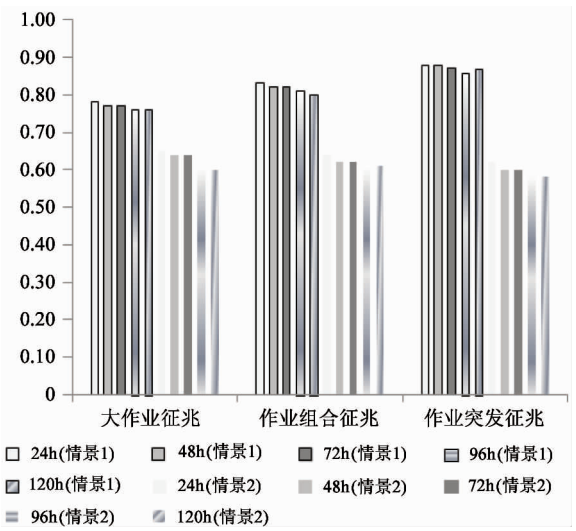

图 6 两种情景(情景 1 和情景 2)、5 种有效预测窗口和 3 种征兆下的失效预测准确率

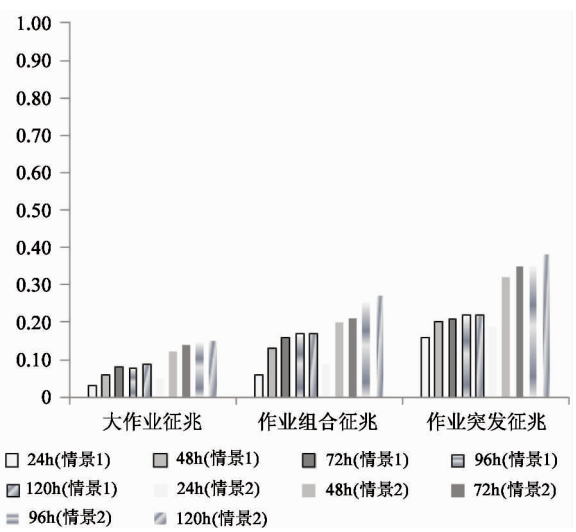

图 7 两种情景(情景 1 和情景 2)、5 种有效预测窗口和 3 种征兆下的失效预测召回率

率也降得最多。总之,失效预测要合理选取有效预测窗口;负载作业强度突然增加会导致失效事件更频繁发生;失效事件的失效征兆可以用来预测具有与该失效事件相同事件类型的其它失效事件,但要合理权衡准确率和召回率。

4 结 论

本文设计并实现了一种利用集群系统的系统日志和作业日志进行结合分析来实现失效预测的方法。经过预处理和过滤,将原始 RAS 日志和 Job 日志解析成具有细粒度信息的 Log Transaction 序列和 Job Transaction 序列。在动态地生成征兆窗口后,在 Log Transaction 中的所有失效事件的征兆窗口内从 Job Transaction 中抽取出负载作业所表现出的三种失效征兆:大作业征兆,不同作业组合征兆和作业突发征兆。最后,利用 3 种征兆实现失效预测。从对美国阿贡国家实验室 Intrepid 系统的 RAS 日志和 Job 日志进行结合分析和失效预测实验的结果中可以看出,我们的方法可以实现较高的准确率。并且,当间接利用 3 种失效征兆进行失效预测时,可以实现高于 50% 的召回率。与其它提高召回率的方法相比,我们不需要复杂的分布式数据挖掘技术。下一步的工作中,一方面将深入分析利用 3 种失效征兆间接失效预测时准确率降低的原因并提出改进的方法;另一方面将考察负载作业在系统失效时的其它行为并提出其它有效的失效征兆类型进行失效预测。

参考文献

[1] Liang Y, Zhang Y, Sivasubramaniam A, et al. Filtering failure logs for a BlueGene/L prototype. In: Proceedings of IEEE International Conference on Dependable Systems and Networks, Yokohama, Japan, 2005. 476-485

[2] Fu X, Ren R, Zhan J, et al. LogMaster: mining event correlations in logs of large-scale cluster systems. In: Proceedings of IEEE International Symposium on Reliable Distributed Systems, Irvine, USA, 2012. 71-80

[3] Gainaru A, Cappello F, Fullop J, et al. Adaptive event prediction strategy with dynamic time window for large-scale HPC systems. In: Proceedings of ACM SOSP Workshop on Managing Large-Scale Systems via the Analysis of System Logs and the Application of Machine Learning Techniques, NY, USA, 2011. 4-11

[4] Gujrati P, Li Y, Lan Z. A meta-learning failure predictor for Blue Gene/L systems. In: Proceedings of International Conference on Parallel Processing, Washington, USA, 2007. 40-48

[5] Zheng Z, Lan Z, Park B, et al. System log preprocessing to improve failure prediction. In: Proceedings of the International Conference on Dependable Systems and Networks, Lisbon, Portugal, 2009. 572-577

[6] Salfner F, Lenk M, Malek M. A survey of online failure prediction methods. *Journal of ACM Computting Surveys*, 2010, 42(3): 1-42

[7] Li Y, Zheng Z, Lan Z, et al. Practical online failure prediction for Blue Gene/P: Periodbased vs Event-driven. In: Proceedings of IEEE International Conference on Dependable Systems and Networks, Hong Kong, China, 2011. 259-264

[8] Chen C, Singh N, Yajnik S. Log analytics for dependable enterprise telephony. In: Proceedings of IEEE International Conference on Dependable Computing Conference, Sibiu, Romania, 2011. 94-101

[9] Schroeder B, Gibson G. A large-scale study of failures in high-performance computing systems. *IEEE Trans on Dependable and Secure Computing*, 2006, 7(4): 337-350

[10] Liang Y, Zhang Y, Sivasubramaniam A, et al. BlueGene/L failure analysis and prediction models. In: Proceedings of IEEE International Conference on Dependable Systems and Networks, Philadelphia, USA, 2006. 425-434

[11] Fu S, Xu C Z. Exploring event correlation for failure prediction in coalitions of clusters. In: Proceedings of ACM/IEEE International Conference on Supercomputing, Reno, USA, 2007. 1-10

[12] Gainaru A, Cappello F, Kramer W. Taming of the shrew: modeling the normal and faulty behavior of large-scale HPC Systems. In: Proceedings of IEEE International Conference on Parallel Distributed Processing Symposium, Shanghai, China, 2012. 1168-1179

[13] Gu J, Zheng Z, Lan Z, et al. Dynamic meta-learning for failure prediction in large-scale systems: A case study.

In: Proceedings of IEEE International Conference on Parallel Processing, Portland, USA, 2008. 157-164

[14] Fu X, Ren R, McKee S, et al. Digging deeper into cluster system logs for failure prediction and root cause diagnosis. In: Proceedings of IEEE International Conference on Cluster Computing, Madrid, Spain, 2014. 103-112

[15] Moens S, Aksehirli E, Goethals B. Frequent itemset mining for big data. In: Proceedings of IEEE International Conference on Big Data, Santa Clara, USA, 2013. 111-118

[16] Wu G, Zhang H, Qiu M, et al. A decentralized approach for mining event correlations in distributed system monitoring. *Elsevier Journal of Parallel and Distributed Computing*, 2013,73(3):330-340

[17] Martino C D, Cinque M, Cotroneo D. Assessing time coalescence techniques for the analysis of supercomputer logs. In: Proceedings of IEEE International Conference of Dependable Systems and Networks, Boston, USA, 2012. 1-12

[18] Oliner A, Stearley J. What supercomputers say: a study of five system logs. In: Proceedings of IEEE International Conference on Dependable Systems and Networks, Edinburgh, England, 2007. 575-584

[19] Zheng Z M, Yu L, Lan Z L, et al. 3-dimensional root cause diagnosis via co-analysis. In: Proceedings of ACM International Conference on Autonomic Computing, San Jose, USA, 2012. 181-190

[20] Yigitbas Y, Gallet M, Kondo D, et al. Analysis and modeling of time-correlated failures in large-scale distributed systems. In: Proceedings of IEEE International Conference on Grid Computing, Brussels, Belgium, 2010. 65-72

[21] Zheng Z, Yu L, Tang W, et al. Co-analysis of ras log and job log on Blue Gene/P. In: Proceedings of IEEE International Conference on Parallel Processing, Anchorage, USA, 2011. 1530-2075

[22] Tang W, Lan Z, Desai N, et al. Fault-aware utility-based job scheduling on Blue Gene/P systems. In: Proceedings of IEEE International Conference on Cluster Computing, New Orleans, USA, 2009. 1-10

A log co-analysis based failure prediction method for large-scale cluster systems

Fu Xiaoyu[*,**,***], Ren Rui[*,**], Zhan Jianfeng[*,**], Sun Ninghui[*,**]

([*] State Key Laboratory of Computer System and Architecture, Beijing 100190)

([**] Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)

([***] University of Chinese Academy of Sciences, Beijing 100049)

Abstract

The failure prediction for large-scale cluster supercomputer was studied. Aiming at the prolem that the existing prediction method only analyzing the single system log needs complex data mining techniques while its prediction recall rate is generally lower, this study presented an effective failure prediction method based on co-analysis of system logs and job logs that records the running workload information. The principle of the method is below: Firstly, the fine-grained two-dimensional event sequence and job sequence are produced through preprocessing and filtering of the two raw logs; Secondly, three failure symptoms are extracted from job logs before the occurrence of failure events; Finally, failure predictions are carried out by using these symptoms. The results of the experiments on real logs of the BlueGene/P system show that the proposed method can predict failures with a higher precision and a higher recall rate.

Key words: large-scale cluster system, system log, job log, log analysis, failure prediction