第五章　日志分析

日志记录分析系统

- **健全的日志记录和分析系统是系统正常运行与优化以及安全事故响应的基础**

- 充足性
- 可用性
- 安全性

维护内部安全

- **对日志并没有充分的认识，安全工作更多地投入以特征码和预定义规则为基础的设备**

- 了解日志的概念
- 了解日志的配置和分析方法

目录

日志分析介绍

1. 日志文件的特点及日志分析的目的
2. 日志的分类
3. 网络日志分析相关术语
4. 网络日志分析流程
5. 日志分析面临的挑战

- 日志就是按照一定的规则将操作系统、应用程序、网络设备中发生的事件记录下来，对系统管理、网络安全策略实施状况的评估及其他安全防御系统的评估都是必不可少的证据。起到预防和阻止网络犯罪的作用。

日志文件的特点

1.多样性

2.可读性差

- 多样性：操作系统、服务、网络设备及应用软件的不同使得日志在格式和存储方式上存在较大的差异
- 可读性差 ：大多数计算机系统日志是以二进制形式存储的，并且各个系统日志格式不一致，使得获取有用信息十分困难。

日志文件的特点

3.日志记录的数据量很大

4.不容易获取

- 日志记录的数据量很大：计算机系统日志记录了诸如 Web服务器、数据库日志、防火墙日志以及系统中关于进程及文件等所有的信息，产生的大量日志数据使得其存储和分析存在较大的困难
- 不容易获取：各系统开发商和网络设备生产商产生的日志在格式和存储方式上各不相同，同时日志会随着系统运行状态的变化而变化，致使日志的获取更加困难。

日志文件的特点

5.不同日志之间存在某种必然的联系。

6.容易被篡改

- 不同日志之间存在某种必然的联系：一个系统事件可能被记录在多个系统日志中，只有将这些相关日志综合起来分析，才能准确的获取该用户的活动意图及活动情况。
- 容易被篡改 ：一方面系统日志的产生和保存方式可以通过修改注册表和syslog的配置文件进行修改或停止日志进程，另一方面可以通过修改、删除或伪造来误导网络管理员获取不实信息。

日志分析目的

1．对用户行为进行审计

2．监控恶意行为

3．对入侵行为的检测

- 对用户行为进行审计：系统日志通过监控账户使用情况及用户的行为来防止行为的滥用。
- 监控恶意行为：日志能够反映非法用户在网络系统中的各种恶意行为，通过日志可以有效监控恶意用户的行为，检测不合法行为。
- 对入侵行为的检测：通过及时收集、保全和分析不安全策略的行为及系统和网络行为的原始信息对黑客入侵系统的线路进行追踪。

日志分析目的

4．系统资源的监控

5．帮助恢复系统

6．评估造成的损失

- 系统资源的监控：获得计算机系统中内存、硬盘、进程、网络、文件、外围设备等的使用情况，并发现资源的异常占用及磁盘等硬件资源错误。
- 帮助恢复系统：系统遭到破坏前的状态信息以及入侵过程和结果信息都被记录在系统日志中，可以帮助迅速的定位系统故障原因，进而恢复系统功能。
- 评估造成的损失：可以确定入侵行为的范围并进一步评估损失，以便采取相应的应对措施。

日志分析目的

7．计算机犯罪的取证

8．调查报告的生成

- 计算机犯罪的取证：记录系统和网络行为的原始信息，通过及时收集、管理和分析系统日志，可以帮助追踪攻击者入侵网络系统的路径，对计算机犯罪活动进行取证
- 调查报告的生成：可以根据日志获取的入侵工具、过程、结果以及攻击者的身份获得详尽的调查报告。

日志的分类

- **UNIX/Linux系统日志**
- **Windows系统日志**
- **网络设备日志**
- **应用系统日志**

- 从日志产生的来源角度分类，日志主要分为三大类：操作系统日志（UNIX/Linux，Windows等）、网络设备日志（路由交换设备、防火墙等安全设备）、应用服务日志（Web等各种网络应用）

UNIX/Linux系统日志

1．登录时间日志

2．进程统计日志

3．错误日志

```
Nov 24 00:23:36 zjd-virtual-machine AptDaemon: INFO: Quitting was requested
Nov 24 00:24:05 zjd-virtual-machine anacron[1136]: Job `cron.daily' terminated
Nov 24 00:25:49 zjd-virtual-machine kernel: [  604.692962] pcnet32 0000:02:01.0 eth0: link up
Nov 24 00:25:49 zjd-virtual-machine kernel: [  604.693073] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> (eth0): carrier now ON (device state 20)
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> (eth0): device state change: unavailable -> disconnected (reason 'carrier-changed') [20 30 40
Nov 24 00:25:51 zjd-virtual-machine avahi-daemon[966]: Joining mDNS multicast group on interface eth0.IPv6 with address fe80::20c:29ff:fe0e:e8ba.
Nov 24 00:25:51 zjd-virtual-machine avahi-daemon[966]: New relevant interface eth0.IPv6 for mDNS.
Nov 24 00:25:51 zjd-virtual-machine avahi-daemon[966]: Registering new address record for fe80::20c:29ff:fe0e:e8ba on eth0.*.
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Auto-activating connection 'Wired connection '.
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) starting connection 'Wired connection '
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> (eth0): device state change: disconnected -> prepare (reason 'none') [30 40 0]
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) Stage 1 of 5 (Device Prepare) scheduled...
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) Stage 1 of 5 (Device Prepare) started...
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) Stage 2 of 5 (Device Configure) scheduled...
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) Stage 1 of 5 (Device Prepare) complete.
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) Stage 2 of 5 (Device Configure) starting...
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> (eth0): device state change: prepare -> config (reason 'none') [40 50 0]
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) Stage 2 of 5 (Device Configure) successful.
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) Stage 3 of 5 (IP Configure Start) scheduled.
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) Stage 2 of 5 (Device Configure) complete.
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) Stage 3 of 5 (IP Configure Start) started...
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> (eth0): device state change: config -> ip-config (reason 'none') [50 70 0]
Nov 24 00:25:51 zjd-virtual-machine NetworkManager[1005]: <info> Activation (eth0) Beginning DHCPv4 transaction (timeout in 45 seconds)
Nov 24 00:25:52 zjd-virtual-machine NetworkManager[1005]: <info> dhclient started with pid 3326
Nov 24 00:25:52 zjd-virtual-machine dhclient: Internet Systems Consortium DHCP Client 4.1-ESV-R4
Nov 24 00:25:52 zjd-virtual-machine dhclient: Copyright 2004-2011 Internet Systems Consortium.
Nov 24 00:25:52 zjd-virtual-machine dhclient: All rights reserved.
Nov 24 00:25:52 zjd-virtual-machine dhclient: For info, please visit https://www.isc.org/software/dhcp/
```

UNIX/Linux系统日志

登录时间日志子系统	登录时间日志通常会与多个程序的执行产生关联，一般情况下，将对应的记录写到/var/log/wtmp 和/var/run/utmp中。为了使系统管理员能够有效地跟踪谁在何时登录过系统，一旦触发login等程序，就会对wtmp和utmp文件进行相应的更新。
进程统计日志子系统	主要由系统的内核来实现完成记录操作。如果一个进程终止，系统就能够自动记录该进程，并在进程统计的日志文件中添加相应的记录。该类日志能够记录系统中各个基本的服务，可以有效地记录与提供相应命令在某一系统中使用的详细统计情况。
错误日志子系统	其主要由系统进程syslogd（新版Linux发行版采用rsyslogd服务）实现操作。它由各个应用系统（例如Http、Ftp、Samba等）的守护进程、系统内核来自动利用syslog 向/var/log/messages文件中进行记录添加，用来向用户报告不同级别的事件。

Windows系统日志

1．系统日志

- 系统中各种驱动程序在运行中出现的重大问题
- 操作系统的多种组件在运行中出现的重大问题
- 应用软件在运行中出现的重大问题
- （事件中的重大问题主要包括重要数据的丢失、错误等，甚至是系统产生的崩溃行为。）

Windows 7 系统日志

级别	日期和时间	来源	事件 ID	任务类别
信息	02/11/2015 08:59:08	Service Control Manager	7036	无
信息	02/11/2015 08:54:06	Service Control Manager	7036	无
信息	02/11/2015 08:50:29	Service Control Manager	7036	无
信息	02/11/2015 08:48:58	Service Control Manager	7036	无
信息	02/11/2015 08:46:45	Service Control Manager	7036	无
信息	02/11/2015 08:39:48	Service Control Manager	7036	无
信息	02/11/2015 08:35:11	Service Control Manager	7036	无
信息	02/11/2015 08:34:27	Service Control Manager	7036	无
信息	02/11/2015 08:34:07	Service Control Manager	7036	无
信息	02/11/2015 08:33:56	Service Control Manager	7036	无

Windows系统日志

2．安全日志

- 各种对系统进行登录与退出的成功或者不成功信息
- 对系统中的各种重要资源进行的各种操作

Windows 7 安全日志

级别	日期和时间	来源	事件 ID	任务类别
信息	02/11/2015 08:31:54	Microsoft Windows 安全审核。	4672	特殊登录
信息	02/11/2015 08:31:54	Microsoft Windows 安全审核。	4624	登录
信息	02/11/2015 08:31:54	Microsoft Windows 安全审核。	4648	登录
信息	02/11/2015 08:30:47	Microsoft Windows 安全审核。	4624	登录
信息	02/11/2015 08:30:34	Microsoft Windows 安全审核。	4672	特殊登录
信息	02/11/2015 08:30:34	Microsoft Windows 安全审核。	4624	登录
信息	02/11/2015 08:30:23	Microsoft Windows 安全审核。	4672	特殊登录
信息	02/11/2015 08:30:23	Microsoft Windows 安全审核。	4624	登录
信息	02/11/2015 08:30:19	Microsoft Windows 安全审核。	5056	系统完整性
信息	02/11/2015 08:30:17	Microsoft Windows 安全审核。	4672	特殊登录
信息	02/11/2015 08:30:17	Microsoft Windows 安全审核。	4624	登录

Windows系统日志

3．应用程序日志

- 主要记录各种应用程序所产生的各类事件

Windows 7 应用日志

级别	日期和时间	来源	事件 ID	任务类别
信息	02/11/2015 08:39:48	Security-SPP	903	无
信息	02/11/2015 08:38:01	Ci	4137	CI 服务
信息	02/11/2015 08:37:27	LoadPerf	1000	无
信息	02/11/2015 08:37:27	LoadPerf	1001	无
信息	02/11/2015 08:34:48	Security-SPP	902	无
信息	02/11/2015 08:34:48	Security-SPP	1003	无
信息	02/11/2015 08:34:47	Security-SPP	1066	无
信息	02/11/2015 08:34:28	SecurityCenter	1	无
信息	02/11/2015 08:34:06	Security-SPP	900	无
信息	02/11/2015 08:33:28	LMS	2000	LMS
信息	02/11/2015 08:33:05	IntelDalJhi	0	无

网络设备日志

1．PIX 防火墙日志

2．交换机日志

3．路由器日志

- 通常网络设备包括路由交换设备、防火墙、入侵检测及UPS系统等。

PIX 防火墙日志

- **该防火墙主要基于专用操作系统，同时采取实时的嵌入式系统来形成支撑。**

主要记录的事件：

- AAA（认证、授权和记账）事件。
- Connection（连接）事件。
- SNMP事件。
- Routing errors（路由错误）事件。
- Failover（故障转移）事件。
- PIX系统管理事件。

防火墙日志

```
[06/16/11 13:16:37] ufw allow in proto tcp from 192.168.0.100 port 22 to 192
[06/16/11 13:16:53] ufw --force delete 1
[06/16/11 13:17:41] ufw allow in log-all proto tcp from 192.168.0.100 port 22
[06/16/11 13:18:37] ufw allow in log-all proto tcp from 192.168.0.105 port 22
[06/16/11 13:18:46] ufw --force delete 1
[06/16/11 13:19:14] ufw default allow incoming
[06/16/11 13:19:29] ufw default reject incoming
[06/16/11 13:19:55] ufw --force delete 1
[06/16/11 13:20:16] ufw allow in proto tcp from 192.168.0.1 port 22 to 192.1
[06/16/11 13:23:15] ufw --force delete 1
[06/16/11 13:23:42] ufw allow in proto tcp from 192.168.0.105 port 22 to any
[06/16/11 13:27:06] ufw --force delete 1
[06/16/11 13:27:19] ufw allow in proto tcp from 192.168.0.105 port 22 to any
[06/16/11 13:27:42] ufw allow in proto tcp from 192.168.0.105/24 port 22 to
[06/16/11 13:27:47] ufw --force delete 1
[06/16/11 13:31:26] ufw allow in from 192.168.0.105/24 port 22 to any port 2
[06/16/11 13:31:50] ufw --force delete 1
[06/16/11 13:31:56] ufw --force delete 1
[06/16/11 13:32:00] ufw --force delete 1
[06/16/11 13:32:42] ufw allow in from 192.168.0.0/24 port 22 to any port 22
```

交换机日志

- 中高端交换机以及各种路由器，一般情况下都会采取一定的方式记录设备自身的运行状态，并且将系统在运行中产生的一些异常情况记录下来。另外，在兼容性方面，上述网络设备通常都提供了对Syslog RFC 3164的支持，并对该协议所明确的各种日志处理机制提供支持，因此可以通过Syslog协议来实现不同设备之间多种日志的相互转发。

```
<date> <time> <facility>.<severity> <program[<pid>]>: <message>
-------------------------------------------------------------------
2011 Nov 15 14:39:04 user.information awplus system: Warm Boot....
2011 Nov 15 14:39:04 user.information awplus evlog: Event log initialized
2011 Nov 15 14:39:04 user.information awplus file: File System initialized
2011 Nov 15 14:39:04 user.information awplus ssh: SSH server disabled
.....
```

路由器日志

```
1 Mar 26 10:47:24.955 VTTY: Console port: waiting connection on tcp port 5000
  1 (FD 10)
2 Mar 26 10:47:26.073 C3600_BOOT: starting instance (CPU0 PC=0xffffffffbfc000
  00,idle_pc=0x0,JIT on)
3 Mar 26 10:47:26.074 CPU0: CPU_STATE: Starting CPU (old state=2)...
4 Mar 26 10:47:26.197 ROM: Microcode has started.
5 Mar 26 10:47:37.620 ROM: trying to read bootvar 'NO_CP0'
6 Mar 26 10:47:37.620 ROM: trying to read bootvar 'NO_RANDOM_NUM'
7 Mar 26 10:47:37.620 ROM: trying to read bootvar 'RANDOM_NUM'
8 Mar 26 10:47:37.620 ROM: trying to read bootvar 'OFFSET'
9 Mar 26 10:47:40.975 CPU0: IO_FPGA: write to unknown addr 0x30006, value=0x0
  , pc=0x60698844 (size=1)
10 Mar 26 10:47:40.985 CPU0: IO_FPGA: write to unknown addr 0x30006, value=0x0
  , pc=0x60698844 (size=1)
11 Mar 26 10:47:40.996 CPU0: IO_FPGA: write to unknown addr 0x30006, value=0x0
  , pc=0x60698844 (size=1)
12 Mar 26 10:47:41.007 CPU0: IO_FPGA: write to unknown addr 0x30006, value=0x0
  , pc=0x60698844 (size=1)
13 Mar 26 10:47:41.009 ROM: unhandled syscall 0x00000047 at pc=0x605ecfb8 (a1=
  0x6293b590,a2=0x00000000,a3=0x00000000)
14 Mar 26 10:47:41.203 ROM: trying to read bootvar 'RANDOM_NUM'
```

应用系统日志

- **在系统的工作过程中，对应用程序的某些重要事件进行记录形成的日志**

- 例如：Apache、FTP、Samba、NFS、DHCP、NFS及微软IIS日志等。

应用系统日志

```
::1 - - [23/Mar/2014:15:07:00 -0700] "GET /images/apache_feather.gif HTTP/1.1" 200 4128
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/producer_consumer.png HTTP/1.1" 200 8
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/log_anatomy.png HTTP/1.1" 200 19579
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/consumer-groups.png HTTP/1.1" 200 268
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/log_compaction.png HTTP/1.1" 200 4141
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /documentation.html HTTP/1.1" 200 189893
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/log_cleaner_anatomy.png HTTP/1.1" 200
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/kafka_log.png HTTP/1.1" 200 134321
::1 - - [23/Mar/2014:15:07:04 -0700] "GET /images/mirror-maker.png HTTP/1.1" 200 17054
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /documentation.html HTTP/1.1" 200 189937
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /styles.css HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/kafka_logo.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/producer_consumer.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/log_anatomy.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/consumer-groups.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/log_cleaner_anatomy.png HTTP/1.1" 304
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/log_compaction.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/kafka_log.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:08:07 -0700] "GET /images/mirror-maker.png HTTP/1.1" 304 -
::1 - - [23/Mar/2014:15:09:55 -0700] "GET /documentation.html HTTP/1.1" 200 195264
```

网络日志分析相关术语

日志分析产品	通过日志代理、标准协议、文件导入等方式采集信息系统中的日志数据，并进行集中存储和分析的安全产品。
日志数据源	产生日志数据的原始来源
日志管理中心	对采集到的日志数据进行集中处理、存储、分析的功能模块。
审计日志	日志分析产品自身审计产生的日志数据。
日志记录	对采集到的原始日志数据进行预处理之后，根据一定规则生成并保存在日志管理中心的日志数据。
授权管理员	具有日志分析产品管理权限的用户，负责对日志分析产品的系统配置、安全策略和日志数据进行管理。
可信主机	赋予权限能够管理日志分析产品的主机。

网络日志分析流程

日志的采集和存储

1．日志数据的采集

- 标准协议接收：日志分析产品应能接收从日志数据源发送的基于syslog、snmp trap、ftp或其他标准协议的日志数据。
- 代理方式采集：日志分析产品应能通过日志代理方式采集日志数据源的日志数据。
- 日志文件导入：日志分析产品应能导入通用格式的日志文件。

日志的采集和存储

2．日志数据的预处理

- 数据筛选：日志分析产品应能基于既定策略对采集的日志数据进行过滤，有选择地生成日志记录。
- 数据转换：日志分析产品应能将各种不同格式的原始日志数据转换为统一的数据格式，且转换时不能造成关键数据项丢失。

日志的采集和存储

3．日志记录的生成

- a)事件发生的日期和时间；
- b)事件主体；
- c)事件客体；
- d)事件描述；
- e)事件类型；
- f)事件级别；
- g)日志数据源的IP地址或名称。

日志的采集和存储

4.日志数据的存储

- 安全保护：日志分析产品应采取安全机制，保护日志记录免遭未经授权的读取、删除或修改。
- 防止日志记录丢失：
- a)日志记录应存储于非易失性存储介质中；
- b)当日志记录的存储容量达到阈值时，发出报警信息；
- c)在日志记录的存储空间耗尽前，采用自动转储的方式将日志记录自动备份到其他的存储空间。

日志的分析和处理

1．日志记录的处理

- 数据整合：日志分析产品应能检查日志记录是否重复或无效，并进行数据的整合以保证数据的有效性、一致性，以及减少冗余信息。
- 数据拆分：若日志记录的单一字段包含多种信息并且这多种信息间具有分隔符，日志分析产品应能将此单一字段拆分成便于分析的若干字段存储。

日志的分析和处理

2．日志记录的分析

- 事件辨别
- 事件定级

日志的分析和处理

2．日志记录的分析

- 事件统计：

a)事件主体；

b)事件客体；

c)事件类型；

d)事件级别；

e)事件发生的日期和时间；

f)日志数据源的IP地址或名称；

g)事件的其他属性或属性的组合

日志的分析和处理

2．日志记录的分析

- 潜在危害分析
- 异常行为分析
- 关联事件分析

日志的分析和处理

2．日志记录的分析

- 日志记录数据挖掘：

具体要求：

a)提取同一类型的事件的共同性质

b)提取单个事件和其他事件之间依赖或关联的知识

c)提取反映同类事件共同性质的特征和不同事件之间的差异型特征，揭示隐含事件的发生；

d)发现其他类型的知识，揭示偏离常规的异常现象；

e)根据数据的时间序列，由历史的和当前的数据去推测未来的数据

日志的显示和告警

1．日志查询

- 事件主体
- 事件客体
- 事件类型
- 事件级别
- 事件发生的日期和时间
- 日志数据源的IP地址或名称
- 事件的其他属性或属性的组合。

日志的显示和告警

2．统计报表

- 日志分析产品应能够根据事件统计结果生成统计报表，并能以通用格式输出。

日志的显示和告警

3．分析报告

- a)日志记录分析结果；
- b)对信息系统或信息系统中单个资源的风险等级提供评估结果；
- c)对日志记录分析结果提供补救建议；
- d)根据日志数据挖掘收集到的知识，提供预测性的信息。

日志的显示和告警

4．告警机制

- a)用户指定的事件，如高危险级别的事件等；
- b)潜在危害分析结果表明信息系统存在潜在危害；
- c)异常行为分析结果表明信息系统存在入侵者的行为或合法用户的异常行为；
- d)日志记录分析结果表明信息系统或信息系统中某一资源存在风险；

日志分析面临的挑战

过多的数据

没有足够的数据

各种各样的记录

假警报

重复数据

难以获得数据

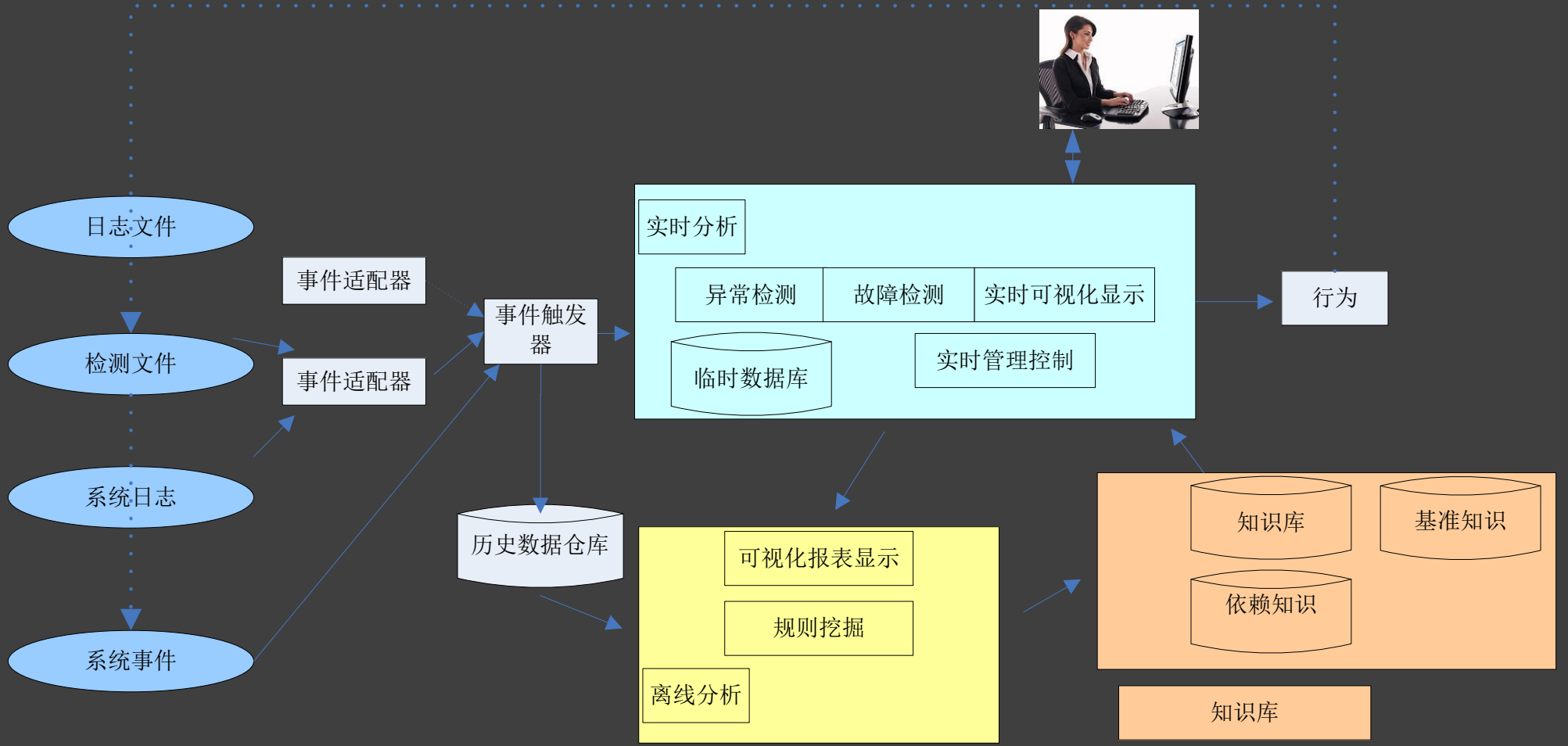
日志分析模型与方法

1.日志分析方法

2.日志分析工具

- 日志数据分析管理系统通常是IT服务管理的核心系统架构。
- 常用于日志分析的数据挖掘算法有关联分析、序列分析及聚类分析。

日志分析系统体系结构


日志文件
检测文件
系统日志
系统事件
事件适配器
事件适配器
事件触发器
实时分析
异常检测
故障检测
实时可视化显示
临时数据库
实时管理控制
行为
历史数据仓库
可视化报表显示
规则挖掘
离线分析
知识库
基准知识
依赖知识
知识库

日志分析方法

1.关联分析

2.序列分析

3.聚类分析

- 关联分析将提取的日志间的特征依照某一规则进行关联，从而得到新的证据链。
- 序列分析描述犯罪行为随时间变化的规律或趋势。
- 聚类分析是将相似的对象划分到同一个类中从而得到多个类的无监督学习过程。

$I=\{i_1,i_2,\cdots\cdots,i_m\}$

关联分析

- 设所有项目的集合为$I=\{i_1,i_2,...i_m\}$，事物数据库为D，一个项目子集为事物$T(T\subseteq I)$，项集A则是一个由项目构成的集合，$A\subseteq T$。如果项集A包含 k个项目，则称为 k项集。项集的支持度support(x)为项集A在事物数据库D中出现的次数占D中总事物的百分比，定义规则的置信度为 $confidence(X\Rightarrow Y)=P(Y\mid X)=\sup port(X\cup Y)\mid\sup port(X)$
- 用户设定一个最小支持度阈值，当项集的支持度超过设定值时，则该项集即是大项集或频繁项。当置信度和支持度分别满足用户设定的阈值时则构成关联规则。

关联分析

- 以Apriori算法为例，介绍频繁项集的具体实现过程：
- Step1.单遍扫描数据库 D计算各个项的支持度，得到频繁1项集构成的集合L_1 。
- Step2.利用迭代得到的 （ k-1 ）项集，产生新的候选 k项集L_k ，设C_k是L_k的超集。这个步骤包括连接和剪枝两步。

- 连接：频繁(k-1)项集集合L_{k-1} 中每个项集中的元素按照字典顺序排序。将包含的前(k-2)个项相同的任意两个 (k-1)项集$F_{k-1}^{`}$ 和$F_{k-1}^{``}$连接得到一个候选 k项集F_k 。
- 剪枝：如果候选k项集的某(k-1)项子集是不频繁的，则从C_k中删除这个候选项集。

关联分析

- Step3.单遍扫描数据库D，然后通过计算得到C_k中各个项集的支持度。
- Step4.剔除C_k中不满足最小支持度的项集得到由频繁k项集构成的集合。
- Step5.重复迭代step2至step4，当新的频繁项集的集合不再产生时则迭代停止，此时得到的即是所有满足最小支持度的频繁项集。

序列分析

- 利用时间对日志进行关联以完成日志的序列分析，来达到网络攻击的预测及防范。
- 序列分析算法对包含候选序列的集合进行计数得到支持度，然后得到满足支持度要求的大序列，主要包括WINEPI、AprioriAll等。

序列分析

- 以WINEPI 为例说明其具体实现过程：
- Step1.设序列事物集 W的最小支持度为minsup，最小置信度为minconf 以及滑动窗口的最大长度为 len，设定窗口宽度 len=1，在W中寻找序列，使其满足支持度大于minsup且长度为1；
- Step2.递增滑动窗口宽度，对于任意满足 $S \in W$ 的序列 $S = \{s_1, s_2, ...s_n\}$ 使窗口的左边界与 s_1 重合，获得长度等于宽度len 的子序列 $sub_1 = \{s_1, ...s_{len}\}$
- Step3.向右每滑动一步窗口得到一个长度为len 的子序列 sub_i，直至滑动到右边界 s_n，一共得到

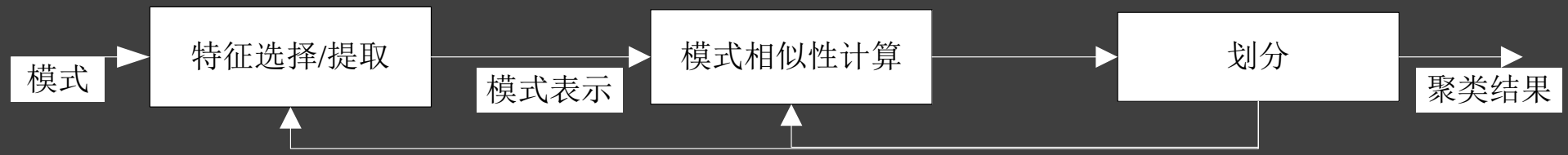
  n-len+1个子序列。
- Step4.计算所有子序列的支持度和置信度，对于任意 sub_i，支持度为 $\sup port(s_i \cup ...s_{i+len+1})$，置信度为 $\sup port(s_i \cup ...s_{i+len+1}) / \sup port(s_i \cup ...s_{i+len-2})$。然后得到满足支持度和置信度大于minsup 和minconf ，长度为len的序列。
- Step5.重复step2到step4，当 len=maxlen时则得到满足置信度和支持度要求的序列集合。

聚类分析

- 聚类(Clustering)就是将数据集划分成同一组对象间的相似度最大，不同组中对象间的相似度最小化的多个组(group)或簇(cluster)的过程。聚类分析是数据分析中的一种重要技术，应用十分广泛，如数据挖掘，机器学习，统计学，模式识别，电子商务，生物学等。

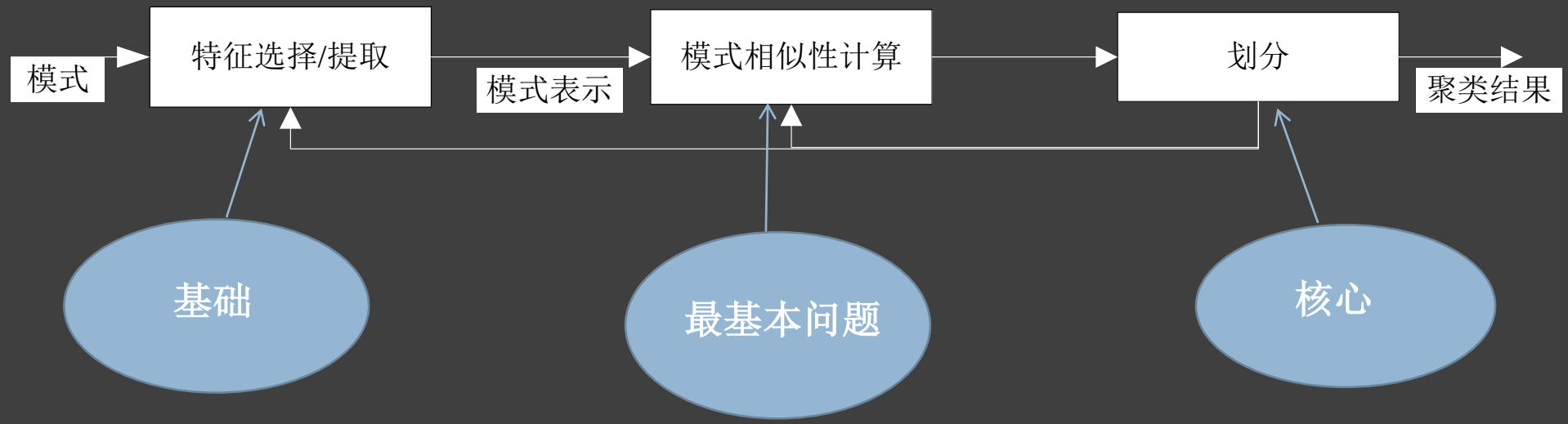
聚类分析

- 聚类分析过程



聚类分析

- 聚类分析过程



日志分析工具

1．AWStats

2．Webalizer

AWStats ：AWStats是在sourceforge上发展很快的一个基于Perl的WEB日志分析工具，系统本身可以运行在GNU/Linux上或安装了Active Perl的Windows上，解决了跨平台问题，且分析的日志直接支持Apache格式(combined)和经过修改的IIS格式。

Webalizer

Webalizer是一个可以通过web服务器浏览其以HTML文件形式呈现的分析结果的WEB服务器日志分析程序。它是用C语言编写的程序，所以具有很高的运行效率，支持标准的网站日志格式及几种其他的格式，支持多种语言，不限制日志文件的大小。但由于缺乏维护，即使拥有Windows平台版也乏人问津。

日志分析工具

3．Wusag

4．Analog

- Wusag ：Wusag是一个基于某种模式生成报表并可同时分析多个日志文件，功能强大的日志分析器，用于UNIX和Windows等各种操作系统的版本。它以二进制文件的形式发布，提供了用户配置命令及友好的界面，但不提供免费服务。
- Analog ：Analog是由Stephen Turner和剑桥大学统计学实验室编写的一款日志分析工具，能够运行在UNIX，Macintosh和dos命令下，具有较高可配置性且与HTML2.0完全兼容。

日志分析工具

5．Web Trends

- Web Trends是美国著名的日志分析软件，能处理超过15GB的日志文件，并在日志文件的基础上，可形成多种报告形式，如DOC、HTML、XLS和ASCALL文件格式，但很贵，并且运行速度很慢。

日志分析工具

- 国外的日志统计分析软件最主要的问题是不适合中国的国情，而国内的日志分析软件很少，主要有网站哨兵、Web-IAM等，但都是商业软件，价格非常昂贵，并对日志文件格式有要求。其他软件大部分是基于Perl 或其他脚本语言的，且不能支持所有的日志文件格式。

日志文件的异常检测

1.基于监督学习的异常检测

2.基于无监督学习的异常检测

- 基于数据挖掘的方法进行日志分析，主要的方法有分类和聚类，或者是基于监督的和基于无监督的。

基于监督学习的异常检测

- **结构化的日志事件**
- **无结构的日志事件**
- **非平衡数据的分类**

- 基于监督学习的方法就是分类。对于系统异常检测问题，一般就两类：“正常”和“异常”。

结构化的日志事件

- 假如x是一个结构化的系统日志事件，且有如下若干属性：

```
process = 'rtvscan.exe'
cputime > 99%
memory > 256M
...
```

结构化的日志事件

- 在朴素贝叶斯里面，给定 y,x被表示成若干独立不相干的属性集合，因此

$$p(y \mid x) = \frac{p(y) \bullet p(process = 'rtvscan.exe' \mid y) \bullet p(cputime > 99\% \mid y) \ldots}{p(x)}$$

- 最终分类的结果看p(y=1|x) 和p(y=0|x) 的值谁更大。如果 p(y=1|x) >p(y=0|x) ，那么就断定 x是异常的。

无结构的日志事件

- 无结构化的日志事件分析方法主要来源于传统的文本挖掘技术。文本挖掘和信息检索主要研究如何处理文本数据。文本数据的模型建立方法千差万别。这里所谓模型的建立，在分类问题上，就是指如何构造分类的属性空间(feature space)。

无结构的日志事件

- 以下图显示一段PVFS2运行日志为例。

```
[D 04/27 05:04] [alt-aio]: pthread_create completed: id: 0, thread_id: 0x9527208
[D 04/27 05:04] issue_or_delay_io_operation: lio_listio posted 0x952aaf0 (handle
92233720368547752 63, ret 0)
[D 04/27 05:04] flowproto-multiqueue trove_write_callback_fn, error_code: 0, flow: 0x9528538.
```

- 这个事件可以表示成为一堆词：alt-aio, pthread_create, completed, id, 0, thread_id, issue_or_delay_io_operation, lio_listio, …
- 用朴素贝叶斯分类函数可以表示为

$$p(y \mid x) = \frac{p(y)\,p('aly-aio' \mid y)\,p('pthread\_create' \mid y)\,p('completed' \mid y)\ldots}{p(x)}$$

无结构的日志事件

- 直接使用朴素贝叶斯的方法通常还比较粗糙。通常的处理手法是预先过滤掉频率极低的词。这些低频词可以当作分类模型的噪声数据。
- 对于贝叶斯分类模型来说，所有的属性都是离散的，那么解决方法就是将这些具体数值离散化。
- 朴素贝叶斯的方法虽然十分简单粗糙，但是在很多日志分析里面符合人类的直观分析经验。

非平衡数据的分类

- 异常检测的分类数据通常都是非平衡的。非平衡的意思就是绝大部分样本是“正常”，只有极少数是“异常”。很多分类算法都是从整体分类的结果评估精度，所以极少数“异常”样本会被忽略。
- 例如上节所述的朴素贝叶斯，SVM分类器。。。

非平衡数据的分类

- 非平衡数据下的分类问题大致有两类解决办法。
- 第一类是通过采样的方法让训练数据集平衡，以SMOTE算法为代表。具体的采样方案可以是under-sampling和over-sampling。
- 第二类办法是加入样本的权重值，计算带权值的分类模型的分类函数。

非平衡数据的分类

- 样本加权的优点是，不用改变训练数据集，不会增大或者丢失原始训练数据。但是它的缺点是权值的调整通常是很困难的。
- 在很多分类模型里面，权值并不直接反应其类别的样本的实际重要性，它还依赖于分类模型和数据集的分布。如果用户对于原始数据集的分布不了解，通常唯一的办法是不断改变权值来测试SVM的效果(cross-validation)。

基于无监督学习的异常检测

- **离群点的检测：主要是子空间聚类分析。**

- 无监督的学习的异常检测指对不需要提供标注的训练数据集进行的检测方法。
- 大致方法是通过数据挖掘里面的聚类分析找出远离簇的数据点。
- 基于此类方法的异常检测都有一个大前提假设：异常的日志事件出现的概率远小于正常的日志事件。

基于无监督学习的异常检测

- 在数据挖掘技术里面，离群点分析依赖聚类算法。

先调用常见的聚类算法，比如K-means、DBScan等算法先找到聚簇。然后遍历每个数据点，计算其到最近一个聚簇的距离。如果这个距离值远大于其他大部分点，那么这个数据点即可被判定为离群点。

子空间聚类分析

- 利用PCA的方法寻找子空间的大前提假设：大部分运行日志事件都是正常情况，只有极少数的事件是异常。
- PCA试图找到最大个子空间去表达全部原始数据。因为异常数据点极少，那么这个表达全体数据集的子空间也近似于表达全部“正常”数据点的子空间。
- 在数据挖掘领域，也有一些自动的方法去寻找合适的子空间聚类，例如GPCA， K-subspace，CLIQUE，SUBCLU等算法。

子空间聚类分析

- 子空间的聚类分析大致可以分为两种情况。
- 方法一是假定子空间的坐标轴是跟原始坐标轴并行的，即垂直投影可以获取的子空间。
- 问题提出：

2^d

2^d

子空间聚类分析

- 子空间的聚类分析大致可以分为两种情况。
- 方法一是假定子空间的坐标轴是跟原始坐标轴并行的，即垂直投影可以获取的子空间。
- 问题提出：

假设原始数据空间是d维的，那么任意{1,2,…d}的子集都可以构成一个子空间，总共有 2^d 个子空间的可能性。当d很大的时候，拿每个子空间来做投影变换再做聚类分析，至少要做 2^d 次，显然是很低效的做法。

子空间聚类分析

- 子空间的聚类分析大致可以分为两种情况。
- 方法一是假定子空间的坐标轴是跟原始坐标轴并行的，即垂直投影可以获取的子空间。
- 问题提出：

假设原始数据空间是d维的，那么任意{1,2,…d}的子集都可以构成一个子空间，总共有 2^d 个子空间的可能性。当d很大的时候，拿每个子空间来做投影变换再做聚类分析，至少要做 2^d 次，显然是很低效的做法。

如何解决？？

子空间聚类分析

- 大部分针对此类问题的算法都利用一种叫做downward-closure的性质:

给定任意一个空间，包含一个聚簇，所有的子空间，也必然包含该聚簇。

子空间聚类分析

- 从欧氏距离计算也可以很好解释这一性质。
- 假设当前空间 T是d 维度的，任意两个点 $x=(x_1,x_2,...x_d)^T$ 和 $y=(y_1,y_2,...y_d)^T$ 如果在一个聚簇内，既满足

$$\sqrt{(x_1-y_1)^2+...+(x_d-y_d)^2}\le\varepsilon$$

- 那么在其任意子空间 $\{i_1...i_p\}\subseteq\{1,...d\}$ 内，也存在

$$\sqrt{(x_{i1}-y_{i1})^2+...(x_{ip}-y_{ip})^2}\le\sqrt{(x_1-y_1)^2+...(x_d-y_d)^2}\le\varepsilon$$

子空间聚类分析

- 子空间的搜索算法大致有分自底向上法和自顶向下法。
- 自底向上的方法是先找低维度空间找聚簇。如果一个低维子空间内不存在任何聚簇，那么其超空间肯定也不存在这样的聚簇，因此算法就不用再搜索其超空间。
- 自顶向下的方法则是先找出存在聚簇的空间，排除这些空间（因为其子空间肯定也存在聚簇），然后继续寻找不存在聚簇的空间的子空间集。这些算法和基于关联规则挖掘与基于Apriori性质的频繁子集挖掘类似。

子空间聚类分析

- 方法二是严格按照线性代数描述的线性子空间(Linear Subspace)。
- 例如原始 d维空间内的一个数据点 $x=(x_1,x_2,...x_d)^T$，p 是一个pxd 矩阵，，那么 x投影到子空间内就是 p_x。p_x是一个 p维的数据点。因为矩阵 p 可以有无穷个，所以这种线性子空间的数量是无穷个。

子空间聚类分析

- GPCA(Generalized Principal Analysis)是一个典型的线性子空间寻找算法。
- 如果一个数据点 落在某一个子空间，则这个数据点满足：

$$p_n(x) = \prod_{i=1}^{n} b_i^T x = 0$$

- 显然数据点不可能完全恰好落在某个平面，问题则变成一个最小化的问题，寻找k元系数$b_1,...b_n$，使其能够最小化

$$\sum_{x \in D} | p_n(x) |^2 = \sum_{x \in D} | \prod_{i=1}^{n} b_i^T x |^2$$

基于事件模式的系统故障溯源

1.从日志到事件

2.事件模式挖掘

3.日志事件的依赖性挖掘

4.基于依赖关系的系统故障溯源

- 通过数据挖掘技术分析系统运行日志，来揣测不同设备和系统模块之间的依赖关系。

基于事件模式的系统故障溯源

- 当信息系统出现问题的时候
- 传统的解决方法

依赖相关领域专家的知识和经验来寻找问题的根源。

现代计算机系统

产生巨量的日志数据。日志系统的数据描述了每一个组件的状态，记录系统的操作流程，比如启动服务，停止服务，网络应用，软件配置修改，软件执行错误等。

从日志到事件

- 文本日志转换为系统事件的3种主要方法

方法	优点	缺点	应用场景
日志解析	非常准确	需要用户理解日志，不能适用于不同的系统日志，需要开发日志解析程序	需要准确生成系统事件，比如监控系统
分类	准确并能适用于不同的系统日志	需要用户提供训练数据，领域专家标注数据可能费时费力	容易获取标注的训练数据
聚类	不需要人工干预并且能适用于不同的系统日志	不是非常准确	可以容忍部分错误或干扰事件

事件模式挖掘

1．序列模式

2．完全依赖模式

3．部分周期依赖模式

4.其他常见模式

- 事件挖掘综合利用数据挖掘技术、机器学习、统计和数据库技术来发现隐含的模式、未知的趋势以及事件之间的关系。
- 被发现的事件可以用于未来的决策，不同的应用需要不同的事件模式来解决问题

序列模式

- 序列模式挖掘主要用于在序列数据中挖掘频繁子项集。一个典型的例子是从大量的销售数据中挖掘感兴趣的模式。常用的算法是Apriori算法。

完全依赖模式

- 完全依赖模式是用来发现非频繁但是有依赖关系的事件模式。为避免最小支持度阈值带来的问题，假设测试被应用到依赖测试。

部分周期依赖模式

- **周期模式的特征让分析人员可以获取数据的内在特征。**

- 实际应用中有几个问题需要解决：
- 周期性行为可能会中断
- 不精确的时间信息
- 周期是事先未知的
- 一个固定的支持度是不能适应去捕获所有模式的周期
- 事件可能丢失或随机事件可能混入周期模式
- 。。。。。。

其他的常见事件模式

- 相互依赖模式，t—模式，频繁情节事件，突发事件，稀有事件等。

日志事件的依赖性挖掘

1.相关性评估

- 离散型的日志事件相关性挖掘目标不是数值上的相关性，而是对于事件发生的时间前后做相关性的分析。在数理统计与概率论中，相关性的定义通常是指两个或者多个随机变量之间分布的关系。

日志事件的依赖性挖掘

- 例如现在有两个随机变量 X和 Y，常用的Pearson相关性度量：

$$corr(x,y)=\frac{\mathrm{cov}(X,Y)}{\sigma_x\sigma_y}=\frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_x\sigma_y}$$

- 决定相关性大小的是 $E[(X-\mu_X)(Y-\mu_Y)]$ ，通过排序不等式：
- 顺序和 $\geq$ 乱序和 $\geq$ 逆序和

相关性评估

- 反映到的联合分布的时候就是：当X和Y同时取大取小的时候，$E[(X-\mu_X)(Y-\mu_Y)]$的值是最大的（顺序和），当X取大但Y取小的时候（乱序和），$E[(X-\mu_X)(Y-\mu_Y)]$较小。所以，当$E[(X-\mu_X)(Y-\mu_Y)]$较大的时候，意味着X和Y大多数时候是同时取大取小，因此这个时候和就有直观意义上的相关性。
- 在离散型的日志事件里面，定义的随机变量可以看作只有两个值，0和1。0表示这个事件没有发生，1表示这个事件有发生。

日志事件的依赖性挖掘

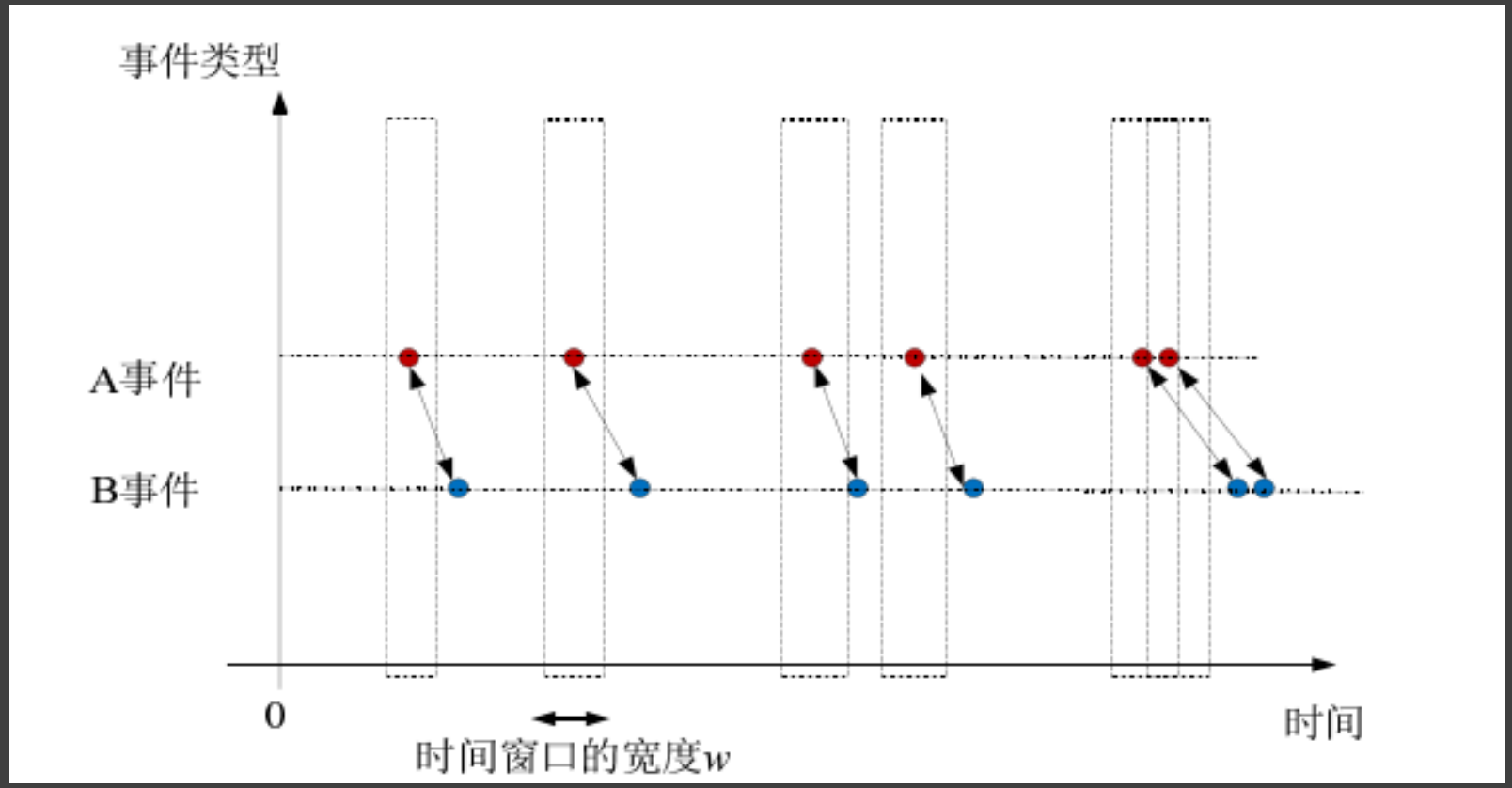
- 最早分析序列数据上的事件依赖模式的论文出之于1995年的KDD。算法寻找的目标就是寻找频繁出现在一起的事件类型集合。该算法的大致思想是将序列数据按照事件分成多个可以重叠的时间窗口。如果事件A 和事件B 经常出现在同一个时间窗口，则可以认定事件A和事件 B是频繁出现的事件类型集合，从而揣测他们可能具有一定依赖性。

日志事件的依赖性挖掘

- 改进算法
- 人们后来逐步用lift和Peason相关度量等其它指标（除支持度和置信度）来评判频繁集合。
- 在某些算法中要求用户给定时间窗口的长度。如果时间窗口宽度过小，有可能会丢掉真正的依赖性模式；如果时间窗口过大，有可能找到大量无意义的相关事件模式。

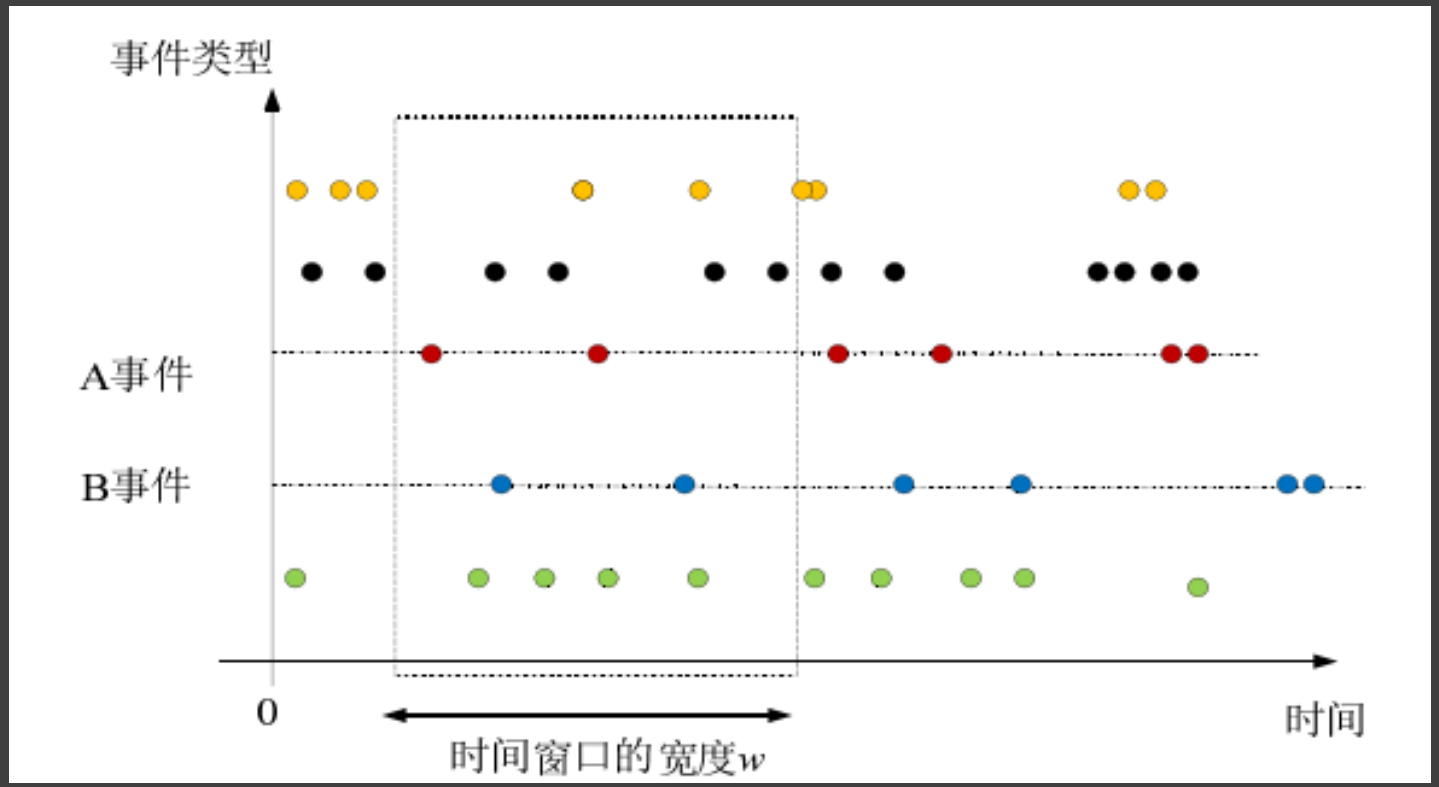
日志事件的依赖性挖掘

- 时间窗口的宽度参数w过小



日志事件的依赖性挖掘

- 时间窗口的宽度参数w过大



日志事件的依赖性挖掘

- 寻找合理时间窗口的首要问题是判定一个给定的时间窗口是否合理，其次再是如何寻找。
- 其他文献中还介绍了基于时间间隔的算法，无人工参数的时间窗口寻找的方法，结果显示不同的事件依赖关系的时间窗口差别很大。小的时间窗口几乎可以当作同时发生，大的时间窗口要相隔一个小时才发生，因此人工定义时间窗口往往是不准确的。

基于依赖关系的系统故障溯源

- 例如node2依赖于node1。假设我们有2个状态{A,B}，那么P(node2=A|node1=B)表达了当node1状态是B的时候，ndoe2状态是A的条件概率。在已知的系统依赖图基础上，就可以如此创建出贝叶斯网络，然后观察对于其中出现的某些组件的日志事件数据得到那些组件的当前状态。当故障出现之后，利用其前后的日志数据得到该贝叶斯网络中某些节点的当前状态，再通过此贝叶斯网络估计其它未观测到的组件最有可能的状态，即可知道故障根源的组件。

事件总结

1.事件总结相关背景

2.基于事件发生频率变迁描述的事件总结

3.基于马尔科夫模型描述的事件总结

4.基于事件关系网络描述的事件总结

- 日志事件总结是一种辅助的数据挖掘方法，其目的是为了通过对给定系统日志进行总结，粗略地展现出日志中事件的关联。

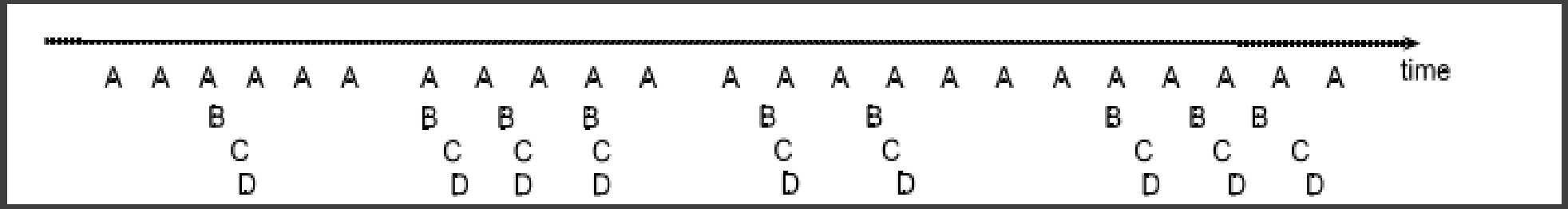
事件总结相关背景

- **针对传统的模式挖掘算法的缺点，Kiernan等人提出了事件总结挖掘算法的四个基本要求**

- 1．简洁并准确。
- 2．能够从宏观描述数据。
- 3．能够描述局部数据的特点。
- 4．尽量少的算法参数。

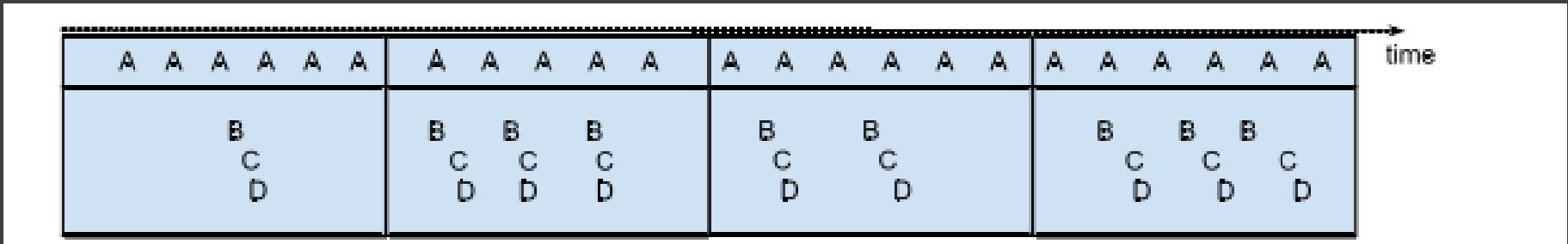
事件总结相关背景

- 例子：假设一份日志中记录了四种事件（A,B,C,D）的发生情况，下图用二维空间描述了这个日志中记录的事件发生情况，其中x轴为时间，不同行（y轴）表示不同类型的事件。



基于事件发生频率变迁描述的事件总结

- 对于前述给定数据，该方法首先把整个事件序列从时间维度分割成了4段，对于每一段，通过事件发生的频率进行了聚类。
- 例如对于示例数据，该方法把每一段聚成了两个簇，其中事件A单独为一个簇，事件B，C和D被聚成一个簇。这样的聚类结果是由于在每一段中，事件A的发生频率和其他事件相差较大，而事件B、C和D发生的频率几乎总是相近。

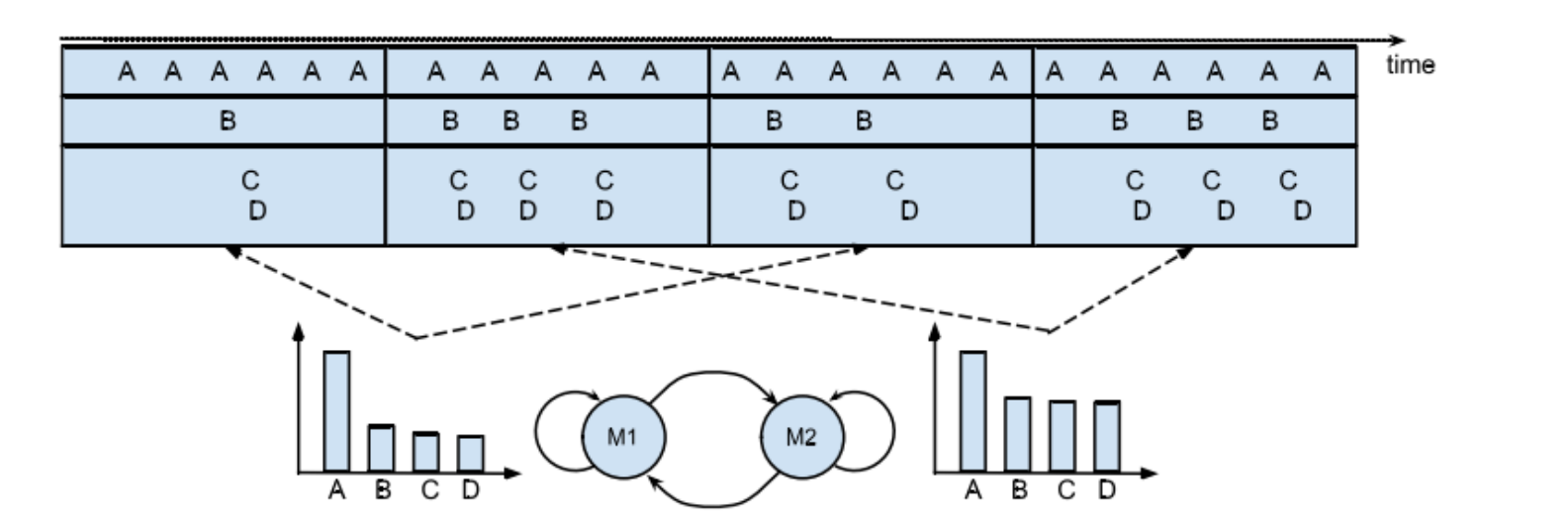


基于事件发生频率变迁描述的事件总结

- 为了对事件序列进行分段，Kiernan等人提出了基于最小描述长度(Minimum Description Length，MDL)原则的事件总结模型。
- 该模型通过二段编码的原则分别对事件段以及事件段中的各个事件发生频率进行编码，描述了需要使用多少比特来表示事件模型以及需要使用多少比特来基于模型描述事件。
- 然后，Kiernan等人提出基于动态规划的方法来寻找基于该模型的最佳事件总结描述。

基于马尔科夫模型描述的事件总结

- 图中整个时间序列被分为了4段，其中第一段与第三段事件发生频率的分布情况相似，因此它们可以使用同一个状态(M1)描述，同理，第二段与第四段事件的发生频率也相似，因此它们也可以用同一个状态(M2)描述。



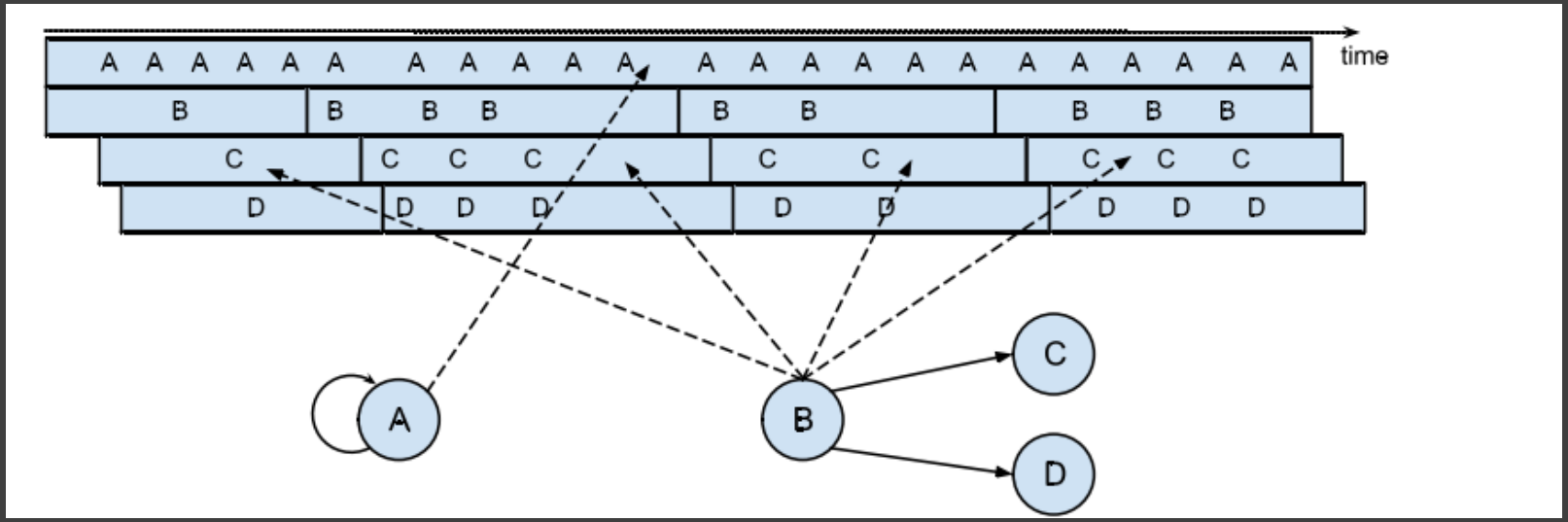
基于事件关系网络描述的事件总结

- **前两种方法的弊端**

- 通过马尔科夫模型描述的事件总结方法从一定程度上保留了事件日志的时序信息，但是仍然丢失了具体信息，因此不能很好地满足“能够描述局部数据的特点”
- 引入了过多的数学模型来抽象地对事件进行总结，这些模型对于系统分析员来说过于晦涩，因此普适性不高。

基于事件关系网络描述的事件总结

- 事件关系网络是IBM公司提出的用于事件管理的通用模型，它使用通俗易懂的状态转换方式来描述事件的变迁。相较于描述事件单纯频率的变化，该方法通过分段来表示事件频率的变化，并通过事件关系网络来表示时间的具体关联，从宏观和局部两个角度对事件进行了总结。



小结

- 本章介绍了日志的概念与特点，日志分析的目的；日志文件的分类；日志分析的流程。重点给出了日志分析的模型和方法，根据实际网络管理的需求重点讨论了日志文件的异常检测以及基于事件模型的系统故障溯源。介绍了事件总结，事件总结方法从宏观的角度展现整个日志中事件的关系，详细介绍了三种常见的事件总结方法。