基于统计分析的自适应Android系统性能异常检测

何钦尧

导师：陈渝

Background

Android

- Most popular smart phone operating system (compete with iOS)
- Open source, which cause fragmentation.
- Third-party modifications, devices with different hardware, different version exist in market.
- Performance degradation upon use with time lapse.
- Performance anomaly occurs frequently which need analysis.
- To solve performance issue, first find and understand anomaly.

Performance anomaly

- Anomaly —— different from normal
  - High latency
  - UI unresponsiveness
  - Disk IO
  - Memory swapness
  - Context switch

- Anomaly may not be too obvious to easy detected by human.
- May only exist in subtle statistics.

Related work

Related work

- Detecting large-scale system problems by mining console logs
  - Finding problem from log in large scale datacenter which is hard for manually inspection.
- Fingerprinting the Datacenter: Automated Classification of Performance Crises
  - Matching runtime statistics fingerprint with existing database, detecting similar problem and match possible solutions.
- AppInsight: Mobile App Performance Monitoring in the Wild
  - Inspecting calling relationship and execution path by injecting code to existing app.
- Measuring the Performance of Interactive Applications with Listener Latency Profiling
  - Injecting to callback listener record logs for Java GUI application

Methodology

Statistics from Linux kernel

- Android is based on Linux kernel
- /proc filesystem
- cpuinfo
- meminfo
- process
- io
- network

- We can have enormous runtime statistics either for the whole system or for a single process, and no need for modification on application.

Proc filesystem

```
$ ls
1       8700        config.gz   execdomains  irq         kpageflags  mtrr          slabinfo      timer_list
22576   8726        consoles    fb           kallsyms    loadavg     net           softirqs      timer_stats
22602   8727        cpuinfo     filesystems  kcore       locks       pagetypeinfo  stat          tty
22603   acpi        crypto      fs           keys        mdstat      partitions    swaps         uptime
22817   buddyinfo   devices     interrupts   key-users   meminfo     sched_debug   sys           version
2663    bus         diskstats   iomem        kmsg        misc        schedstat     sysrq-trigger vmallocinfo
69      cgroups     dma         ioports      kpagecgroup modules     scsi          sysvipc       vmstat
783     cmdline     driver      ipmi         kpagecount  mounts      self          thread-self   zoneinfo
```

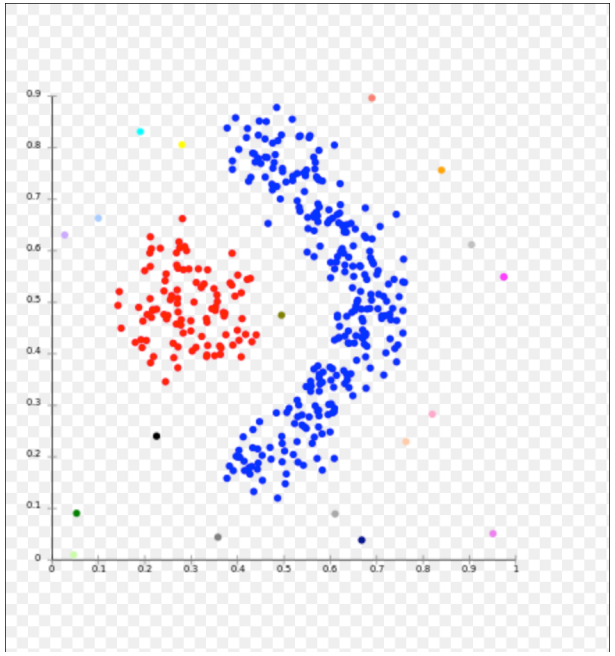
```
$ ls
attr        cmdline          environ  limits     mounts      oom_adj      root     stat     wchan
autogroup   comm             exe      map_files  mountstats  oom_score    sched    statm
auxv        coredump_filter  fd       maps       net         oom_score_adj schedstat status
cgroup      cpuset           fdinfo   mem        ns          pagemap      smaps    syscall
clear_refs  cwd              io       mountinfo  numa_maps   personality  stack    task
```

Performance anomaly detection

- When does a computer system run into an anomaly state.
- Large amount of statistics makes it hard for human observation.
- What is anomaly? High load? High latency? Hardware exception?
- Anomaly is rare in real system, difficult for direct prediction.
- Unsupervised learning from normal data —— anomaly detection.
- Find pattern adaptively from real time usage data without human intervention.

Anomaly(outlier) detection

- Gather normal feature pattern to form cluster.
- Outlier is predicted to be those far from any cluster.



UI latency prediction

- Find when a Android app encounter high UI latency with high-speed photography.
- Runtime statistics from Android OS kernel and framework, both by Linux kernel and code injection to framework and application.
- Can get labeled data about when we encounter UI latency and the runtime state.
- Enable further analysis of its reason.
- Maybe build a supervised machine learning model to predict latency directly from statistics.

Schedules

Schedules

- 2017.1 – 2017.3 build a tool which can gather and store runtime statistics from Linux kernel in proper format.
- 2017.3 – 2017.4 gather runtime statistics (both normal and anomaly) in experiment settings.
- 2017.4 – 2017.5 train unsupervised model for anomaly detection and online testing.
- 2017.5 – 2017.6 synthesize and thesis writing.

Thanks