CSE-605 Checkpoint 1

Rex

March 12, 2015

Contents

1 TODO Introduction

We will measure the predictability of Android. How far ordinary Android system between Real-time system.

2 TODO Android Components

- Intents and Intent Filters We will evaluate intent delivery mechanism of Android. The intents will appear for almost all multiple processes experiments.
- Activities
- App Widgets

  We will not directly evaluate activities and app widgets because they're only related with UI, it's hard to produce convincible result because there're too many elements out of control, like the GPU power, the screen resolution.

- Services

  We will not directly evalute services as well. The reason is same as activities.

- Content Providers

  We will evalute content providers related **Garbage Collector**, **Synchronization**, and **Scheduler**. Because it's shared data mechanism Android provied.

- Processes and Threads We will evaluate processes and threads directly. They're both our targets and mechanisms to use.

3 Intents/Receivers

3.1 Single Process

To directly evaluate intents and receivers on single process deosn't make sense. But this can provide **baseline for our experiments**. We can compare the results of same experiment on single process and multiple processes. Then we can get reasoning of different behaviour.

3.2 Multiple Processes

We can use process(es) to generate bunch of intents, then use other process(es) to receive the intents. In addition, there're background process(es) with some computation to produce pressure for Android system. So we can evaluate the order and time of intent delivery. It can provide some pressure for **Garbage Collector**, **Synchronization**, and **Scheduler**.

3.2.1 Garbage Collector

We can use one sender and one receiver to test Garbage Collector, So we can evaluate how garbage collector works:

- frequency of grabage collection and memory pressure
- running time of grabage collection and memory pressure

The memory pressure should contains different types:

	big objects	medium objects	small objects
long live time	x	x	x
short live time	x	x	x

So where pressure come from?

To evalute the behavior of Android system, it need some pressure for different components. So that we can infer the predictability of different conponents, and the interaction between different components. The pressure can come from:

- Other background process(es) with computation
- Computation inside senders

- Computation inside receivers

We will divide our experiments into three phases.

Phase 1. we only have pressure from background process(es). It's easier to implement and tune for different `Memory Pressure Types`.

Phase 2. we'll add additional computation to senders and receivers to compare whether computation source affect Android's performance.

Phase 3. we combined the different pressure together to get final evaluation.

3.2.2 Synchronization

This task need other Android components. Because we can not pass an object as extra of an intent, we need serialize the object first. So there no directly synchronized mechanism between sender and receiver, but we can pass some metadata to let receivers use something need synchronization like **Content Provider**. We'll discuss in next section.

3.2.3 Scheduler

We can use multiple background processes to provide pressure for scheduler. Then we use the order of intent delivery to evaluate scheduler and intent delivery mechanism. More details can be found in Section Processes/Threads.

4 TODO Content Providers

5 TODO Processes/Threads

6 TODO AlarmManager

It's a critical factor for real-time system.

7 Parcelable/Serializable

According to this blog, parcelable mechanism have 10 times better performance than serializable mechanism. But parcelable need developers to implement writeToParcel and createFromParcel manually. So parcelable can save the overhead to iterate all fields of object. But we can compare the two mechanisms by how much pressure they generate to garbage collector.

The approach is to pass same amount of objects from one process to another process (either the same process or alien), then we compare the different behaviors of garbage collector. It's possible to evaluate scheduler as well.