6J-3

# AndroidOSにおけるLRUに基づく終了プロセス選定の改善

野村　駿†　　永田　恭輔†　　中村　優太†　　山口　実靖†

工学院大学大学院†

## 1. はじめに

Android はスマートフォン，タブレット PC，音楽プレイヤーなど様々なデバイスの OS として採用されており，その重要性が高まっている．

Android には，low memory killer と呼ばれる独自のメモリ管理システムが搭載されており，独自のルールに従ってメモリの管理を行なっている．low memory killer では，メモリの空き容量を確保するためにプロセスを強制終了する．このプロセスの強制終了により，ユーザが再度同じアプリケーションを使用する場合に，プロセスの再起動が必要となりユーザの待ち時間を増加させることがある．

本研究では，強制終了するプロセスの選定の改善によりプロセス再起動にともなうユーザの待ち時間を低減することを目的として，新しい選定手法の提案とその評価を行う．

## 2. low memory killer

low memory killer は，メモリの空き容量が閾値以下まで下がった場合に起動され，*adj* と *minfree* の関係に基づいてプロセスを選定し強制終了するプログラムである． low memory killer が起動されると，強制終了するプロセスの選定のための第一次判定として起動している全てのプロセスの *adj* を比較し，*adj* の数値がより高いプロセスから順に強制終了する．最高 *adj* のプロセスが複数存在する場合，第二次判定としてメモリ使用量を比較し，メモリ使用量のより多いプロセスを強制終了する．

*adj* と *minfree* の関係は，/init.rc で定義されている．*adj* はプロセスの状態，種類により定まり，プロセスの状態が変化するたびに数値が増減する． *adj* が小さいプロセスほど強制終了されづらい．*minfree* は，プロセス強制終了実行の閾値となる空きページ数であり，*adj* によってランク分けされている．

Improvement of kill process selection based LRU in AndroidOS

Shun Nomura†, Kyosuke NAGATA†, Yuta Nakamura†, Saneyasu YAMAGUCHI†

† Electrical Engineering and Electronics, Kogakuin University Gtaduate School

## 3. アプリケーションの起動手順

Android のアプリケーションを新規に起動する場合，次の手順に従って起動される．①ユーザがアプリケーションのアイコンをタッチする．ホームアプリケーション(Launcher)が起動要求のインテントを ActivityManager に送信する．②ActivityManager が Zygote にプロセス生成要求を送信する．③Zygote が自分自身を fork し，子プロセスを生成する．④新しいプロセスが初期化される．⑤アプリケーションのライフサイクルに従って `onCreate()`，`onStart()`，`onResume()`が呼び出される[1].

また，起動済みであるがバックグラウンド状態にあるプロセスの再開は，次の手順に従って行われる．①ユーザがアプリケーションのアイコンにタッチするとホームアプリケーションが再開要求のインテントを ActivityManager に送信する．②それを受けた ActivityManager は対象のバックグラウンドアプリケーションプロセスに再開要求を送信する．③バックグラウンドアプリケーションプロセスは再開要求を受けてアプリケーションプロセスとして起動しフォアグラウンド状態となる．本論文では前者を「新規起動」，後者を「再開」と呼び，1 度起動したプロセスが low memory killer によって強制終了されることなく再利用された場合は「再開」，強制終了された後に再利用された場合は「新規起動」の手続きが行われる．通常，「再開」よりも「新規起動」に要する時間の方が長いため，後に再利用するプロセスの強制終了はユーザの待ち時間を増加させてしまう．

## 4. 提案手法

本章では low memory killer の強制終了プロセス選定手法を改変し，プロセス開始時間の低減を行う手法を提案する．

### 4.1 LRU 手法

本節で，強制終了アプリケーション選定の第一次判定を LRU にて行う LRU 手法を提案する．

本手法では，ユーザが起動したアプリケーションを LRU により管理し，LRU 順にて新しい(最後の起動からの時間が短い)ものの *adj* を下げ強制終了の対象となりにくくする．後述の評価実験の実装では *adj* を URL 順/2 とする．ただし，

変更前の *adj* が変更後の *adj* より低い場合は，変更前の *adj* を優先する．

### 4.2 起動時間手法

本節で，アプリケーションの新規起動時間を考慮して選定する起動時間手法を提案する．本手法では，アプリケーションの新規起動時間を計測し，履歴内の新規起動時間の長いアプリケーションを強制終了されにくくする．同様に実装では *adj* を順位/2 とした．アプリケーションの新規起動時間は，アプリケーションのライフサイクルにおける `onCreate()` の呼び出しから `onResume()` の呼び出しまでの時間とする．

### 4.3 LRU×メモリ手法

本節で，強制終了プロセス選定の第二次判定をプロセスのメモリ使用量と LRU 順の積で行う LRU×メモリ手法を提案する．アプリケーションが LRU の履歴内に存在しない場合は，LRU 順として(履歴長+1)を用いる．

### 4.4 LRU_起動時間併用手法

本節で，LRU 手法と起動時間手法を組み合わせて第一次判定を行う LRU_起動時間併用手法を提案する．

本手法では，LRU 手法と起動時間手法を並列して動作させ，両値の小さい方をそのプロセスの *adj* とする．

### 4.5 LRU_LRU×メモリ手法

本節で，第一次判定を LRU 手法，第二次判定を LRU×メモリ手法で行う LRU_LRU×メモリ手法を提案する．

## 5. 評価

本章にて，標準 low memory killer(以下標準手法と呼ぶ)および提案手法のアプリケーションの合計起動時間の評価を行う．

### 5.1 評価方法

標準手法および提案手法が実装された Android スマートフォンにおいて，複数のアプリケーションを順に起動し，その起動に要した時間を測定した．実験は，表 1 に示す仕様のスマートフォンを用いて行った．

表 1 実験環境

| | |
|---|---|
| Device | Nexus S |
| OS | Android4.0.3 |
| CPU | Cortex A8 （ Hummingbird ） Processer 1GHz |
| Memory | 512MB |

### 5.2 評価シナリオ

本稿では，起動するアプリケーションの順番を定めたものを “シナリオ” と呼ぶ．本実験では，ユーザから許可を得て取得したユーザの実際の 1 日のアプリケーション使用履歴であるシナリオを 2 つ(シナリオ A，B)用意し，標準手法と全ての提案手法の起動時間の評価を行った．

シナリオ A はアプリケーションを 154 個，シナリオ B はアプリケーションを 397 個起動しており，アプリケーション起動とアプリケーション起動の間にホームアプリケーション(Launcher)を起動し，実際の使用順に近いものとしている．

### 5.3 評価結果

シナリオ A，B における評価結果を図 1，2 に示す．各図の縦軸は，シナリオのホームアプリ（Launcher）以外の全アプリケーションの起動時間の合計であり，少ないほど優れている．

図より，両シナリオにおいて標準手法より全ての提案手法が合計起動時間が短く，標準手法よりも優れた手法であることが確認できた．

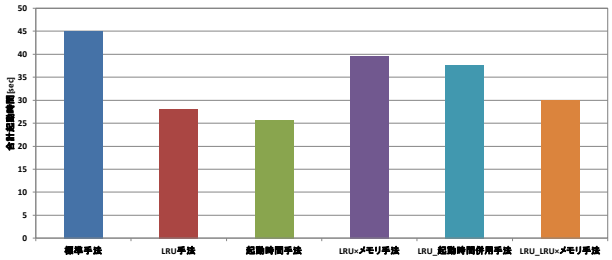


図 1 シナリオ A における評価結果

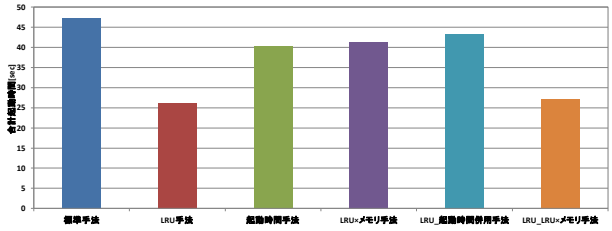


図 2 シナリオ B における評価結果

## 6. おわりに

本論文では low memory killer における強制終了プロセスの選定手法に着目し，LRU による選定手法や，アプリケーションの起動時間を考慮した選定手法，を提案し，評価した．評価した結果，提案手法が標準手法よりも優れていることが確認された．

今後は，`onResume()` 開始後に発生する処理に要する時間も考慮し考察していく予定である．

**謝辞**

本研究は JSPS 科研費 24300034，25280022 の助成を受けたものである。

**参考文献**

[1] Kyosuke Nagata, Saneyasu Yamaguchi, "An Android Application Launch Analyzing System ," 8th ICCM: 2012 International Conference on Computing Technology and Information Management, (2012/04/24)