B-024

オブジェクトのサイズと寿命を考慮した Android 世代別 GC の性能向上に関する一考察

Performance Improvement of Android Generational GC Based on the Relation between Sizes and Lifetimes

森 竜佑† 栗原 駿† 福田 翔貴† 小口 正人‡ 山口 実靖†
Ryusuke Mori Shun Kurihara Shoki Fukuda Masato Oguchi Saneyasu Yamaguchi

1. はじめに

Android には Low memory killer という機能があり，メモリ不足になるとプロセスを強制終了する．強制終了されたプログラムを再利用するときにはプロセスの再生成が必要になり，より多くの時間を要する．よって，プロセスメモリサイズを小さくして強制終了の回数を低減することが重要である．

Android は ART（Android Runtime）上で動作する．ART のメモリ管理機能の 1 つに GC があり，GC がメモリを解放する．よって，ART の GC の動作を制御することによりプロセスのメモリサイズを制御することができる．これにより Low memory killer の回数も抑制できる．

ART に実装されている GC の一つに世代別 GC がある．世代別 GC ではオブジェクトに年齢という概念を取り入れ，年齢ごとにオブジェクトを分類し GC を行う．ART における世代別 GC では，メモリ領域を新世代領域と旧世代領域に分類し，新世代領域内 GC で一定回数回収されなかったオブジェクトは旧世代領域へと Promote(昇進)する．この Promote の条件を調整することにより，GC の性能を向上できると考えられる．また過去の研究[1]にて，オブジェクトの分類の改善により GC の性能を向上できる可能性が示されており，文献[2]にて世代別 GC の Promote 条件を厳しくする手法が 2 つ提案されている．本稿では，文献[2]の手法を著明な実アプリケーション（YouTube）で評価し，有効性の検証を行う．

2. ART における世代別 GC

ART には Generational Semi Space(GSS)という世代別 GC が実装されている．GSS ではオブジェクトを新世代領域に入るものと旧世代領域に入るものの 2 グループに分けて管理し，範囲の異なる 2 つの GC を行う．

生成されたばかりのオブジェクトは新世代領域に置かれ，bps GC(bump pointer space GC)によって高速にかつ積極的に回収される．2 回の bps GC を生き残ったオブジェクトは旧世代領域へと Promote される．そして，Promote したオブジェクトの累積バイト数が PromotedThreshold（初期設定にて 4 MB）を超えるたびに whole GC が行われる．whole GC の範囲は全領域であり，GC 時間が長いため，頻繁に whole GC を行うことはアプリケーションの性能上は好ましくないと考えられる．ただし，旧世代領域に Promote されてから参照されなくなったオブジェクトは bps GC の調査範囲外となってしまい，whole GC を行うまで回収できなくなってしまうため，ヒープサイズの縮小のためには whole GC を積極的に実行するか，Promote 条件を厳しくして bps GC の調査範囲外となるオブジェクトを減らすことが重要である．

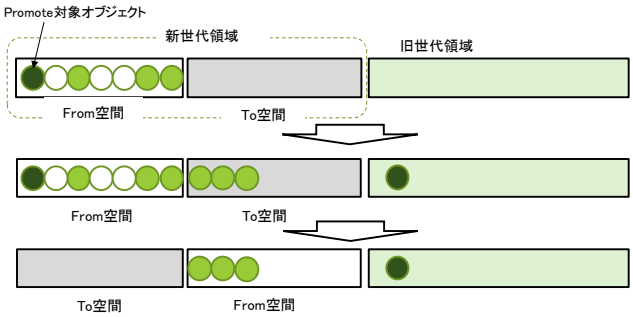

図 1 Generational Semi Space

スマートフォンアプリケーションにおけるオブジェクト特性

本章にて，一般公開されている既存のアプリケーションにおける生成オブジェクトのサイズや寿命などの特性の傾向について述べる．

文献[3]にて，Android のアプリケーションにおいて生成されるオブジェクトの特性の調査結果が示されている．この文献では，ART の動作観察システムを構築し，アプリケーションランタイム内におけるオブジェクトの生成と GC による回収の観察を行い，特性と寿命の関係の調査を行っている．

同報告では，オブジェクトの寿命は短いものがもっとも多く，寿命が延びるに従い該当オブジェクトの数が単調に減少することが示されている．寿命が 0 のオブジェクトの数が最も多く，全体の約 80%を占めることが示されている．ただし，同報告における年齢はそのオブジェクトが経験した GC の回数であり，寿命とはそのオブジェクトが回収されるまでに経験した GC の回数の意味である．

オブジェクトのサイズと寿命の関係に関しては，オブジェクトサイズが小さいほど，寿命が 0 である確率が高く平均寿命が短い傾向があるこが示されている．

これらの傾向より，オブジェクトがゴミになりやすいか否かを推定することが可能であり，GC 性能の向上が可能であると予想される．

3. Promote 抑制手法

本章にて，文献[2]にて提案された 2 つの手法を紹介する．いずれの手法も ART の GSS の Promote 条件を厳しくし，ヒープサイズの削減を目指す手法であり，一つはオブジェクトの特性を考慮せずに Promote を抑制する手法であり，単純

† 工学院大学大学院 工学研究科 電気・電子工学専攻
Electrical Engineering and Electronics, Kogakuin University Graduate School
‡ お茶の水女子大学 理学部 情報科学科
Department of Information Sciences, Ochanomizu University

Promote 抑制手法と呼ぶ．もう一つは，オブジェクトのサイズを考慮して Promote を抑制する手法であり，オブジェクトサイズを考慮した Promote 抑制手法と呼ぶ．

4.1 単純 Promote 抑制手法

本節にて，単純 Promote 手法の紹介をする． 前述のように，ART の GSS GC は GC を 2 回経験したオブジェクトを旧世代領域に昇進させる．本手法では，GC を 2 回以上経験したオブジェクトは確率 p で Promote させ，1-p で Promote させない．ある GC 実行にて Promote させないと判定されたオブジェクトも，次回以降の GC で再度判定の対象となり，同様に確率 p で Promote させ，確率 1-p で Promote させない処理をうける．

4.2 オブジェクトサイズを考慮した Promote 抑制手法

オブジェクトサイズを考慮した Promote 抑制手法では，GC を 2 回以上経験したオブジェクトのうちサイズが m [byte]未満のオブジェクトは確率 p で Promote させ，確率 1-p で Promote させない様に制御する．サイズが m 以上のオブジェクトは，通常通り GC を 2 経験すると必ず Promote させる．

前章で述べたとおり，サイズが小さいオブジェクトは短命である確率が高いという傾向が確認されている．よって，サイズが小さなオブジェクトに対して Promote することを抑制することにより，近い将来にゴミになるオブジェクトが旧世代領域に Promote してしまい GC による回収を長期間回避してしまう現象を減少できると期待できる．

5.性能評価

本章にて，前章で紹介した手法の YouTube アプリケーションにおける評価を行う．

5.1 評価方法

YouTube アプリケーションを起動し，動画検索でヒットしたリストを表示し続け，GC の動作を観察した．動画検索で用いたワードは “a” “b” “c” “d” “e” “f” “g” であり，検索結果を表示し終えたら次のワードで検索を行う． “g” の検索結果を表示し終えたら測定終了とした．またこれらの処理は全て自動化しており，人間の操作による誤差は含まれていない．

前章で述べた，m と p の値はそれぞれ 16，0.66 に設定した． 1 回の GC で Promote するオブジェクトが少なくなることを考慮し，PromotedThreshold は 1 MB に変更した．測定に用いた端末は Nexus7(2013)， OS のバージョンは Android 6.0.1，CPU は Snapdragon S4 Pro 1.5GHz，メモリは 2GB である．

5.2 評価結果

図 2 に平均ヒープサイズを示す．平均とは，GC 発生直後のヒープサイズを調査し，それらの平均である．結果より，オブジェクトサイズを考慮した Promote 抑制手法と単純 Promote 抑制手法はいずれも通常手法と比較してヒープサイズを縮小できていることを確認できる．

図 3 に各 GC 処理中に起こった STW（Stop The World）時間の平均を示す．結果より，オブジェクトサイズを考慮した Promote 抑制手法と単純 Promote 抑制手法いずれも STW 時間の増加が確認できる．しかし，オブジェクトサイズを考慮した Promote 抑制手法と単純 Promote 抑制手法では前

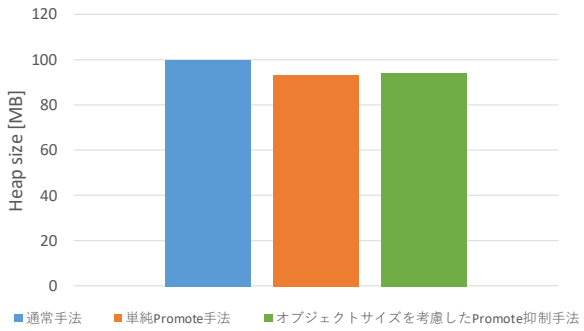

図 2 平均ヒープサイズ

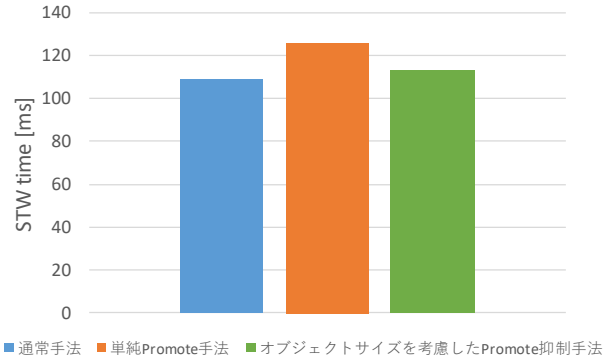

図 3 平均 STW 時間

者の方が STW 時間の増加が少なく，アプリケーション性能において優れていることがわかる．

4. おわりに

本稿では，Promote の抑制によるアプリケーションヒープサイズ縮小手法を紹介し，著名な動画再生アプリケーションを用いた性能評価を行った．評価の結果，Promote 抑制によりヒープサイズの縮小が可能であることと，オブジェクトサイズを考慮した Promote 抑制手法の方が STW 時間の増加が少ないことを確認した．

今後はより多くのアプリケーションでの評価と，STW の低減方法に関する考察を行っていく予定である．

謝辞

本研究は JSPS 科研費 26730040, 15H02696, 17K00109 の助成を受けたものである。

本研究は、JST、CREST JPMJCR1503 の支援を受けたものである。

参考文献

[1]濱中 真太郎　栗原 駿　福田 翔貴　小口 正人　山口 実靖, ”Android アプリケーションのオブジェクトと GC の関係に関する一考察”, FIT2016, 2016 - 09

[2] 森 竜佑　　小口 正人　山口 実靖, ” スマートフォンアプリケーションの特性を考慮した世代別 GC の Promote 条件に関する一考察”, 第 19 回 CDS 研究会, 2017-5

[3] Shintaro Hamanaka, Shun Kurihara, Shoki Fukuda, Masato Oguchi, Saneyasu Yamaguchi, ” A Study on Object Lifetime in GC of Android Applications” , CANDAR 2016 ,2016-11