ISS-P-60

履歴管理のためのグラフ情報検索

高杉　雄大†　三浦 孝夫†
† 法政大学理工学部創生科学科

1. システム開発と履歴管理

システム開発管理を効果的に行うためには,処理過程で生じる情報を正しくモデル化し,的確で効率よく検索する必要がある．文書などの変更管理機構(バージョン管理機構)では,プログラムソースコードや管理文書などシステム開発の変更履歴を記録追跡する．一般には,管理ログ情報を全件走査する必要から低水準のコマンドを用いることが多く,効率よい処理ができない．Linux システムで利用されたGIT システムはこのような情報を管理する機構である．GiT ではソースコードが共有リポジトリとして保存される.GiT の log はファイル名,履歴のログメッセージ,自分のID,親のIDの要素を有している．

バージョン管理システムログ操作は固有であり管理権限から汎用性を与えない。本稿では制限のない汎用的な履歴情報検索,並びに安定した機能の提供を目的とする．履歴管理において扱う情報ではグラフでモデルできることが多い。グラフの頂点は個別の履歴情報を表し，有向辺は頂点間の関係と更新順序を表す．方向は更新順序と不可逆を示す．辺<A,B>では“A→B”順序はあるが,“B→A”はない．始点とは連結関係の古い点(親)を指し,終点は新しい点(子)を指す.

2. 履歴情報管理とグラフ操作

グラフ構造による頂点検索や構造検索を行うためにグラフ操作を提案する．本稿では,情報検索のため有向グラフに対して以下の10機能を導入する.

コミットデータを利用して検索 –機能をそれぞれ下記のコマンドで実現		
	getRoot()	#根 id を表示
情報の中身・成分を知る機能		
	-getInfo(x)	#id x の属性を表示
最新の情報を知る機能		
	-getLeaf(x)	#id x の子孫葉ノードを全表示
情報の更新回数		
	-getDes(x,y)	#id x の子孫ノードをレベル y まで
情報のアジリティ		
	-getAns(x)	#id x の先祖をすべて表示
2つの情報を比べて優位性の有無を問う機能		
	-getLCS(x,y)	#x,y の最短経路
情報のなかから精度の高いものを敏速に収集・処理をする機能		
動作する上で必要な情報を探して保持する機能		
	-getPar(x)	#id x の親 id
	-getCh(x)	#id x の子すべて
	-getMerge(x)	#id x のMERGE 子すべて
	-getMergePar(x)	#id x のMERGE親

3. 情報検索の試作

本稿では履歴管理のためのグラフ構造管理を試作し,検索機能の有用性を評価する．テストに用いるデータは kernel/git/apw/checkpatch.git の Linux 3.2 から Linux3.3-rc1 までの変更履歴 1852 点を使用する．グラフ検索機能が履歴検索の結果と見比べて整合性を判断する．結果を得るための手順（ステップ数）で評価をする．各操作では、GiTLoG から連続的な更新が多いため，レベル差以上の変動がある．最短経路検索において表示される件数が多いほど内容の違いが多いと推測できる.

ランダムなデータを与えて10件の検索命令を実行し,各コマンドの実行命令内部のステップ数を表す．実際のログ状況の結果から大半の命令は一定の（単純な）動作をするが、getLCS() では平均 66.1 ステップ分散 59.52 ステップとなる．実行ステップ数の数値の大きさに対して,レベルの差が大きな影響を与える。getLCS()以外の検索する箇所が一律なので実行ステップ数が一定である.

コマンド	平均	分散
getInfo()	1	0
getRoot()	1	0
getLeaf()	1	0
getDes()	2	0
getAns()	1	0
getPar()	1	0
getCh()	1	0
getLCS(x,y)	66.1	59.52
getMerge()	1	0
getMergePar()	1	0

４．結論

履歴管理のための汎用操作を提案した．実際のログに適用し、ステップ数より検索機能の性能を示した．getDES や getLCS の結果の表示により履歴の深さを意識しない検索が可能となったが実行効率の変動がかなり大きいことが判った.

[参照文献]

index ： kernel/git/apw/checkpatch.git;
https://git.kernel.org/cgit/linux/kernel/git/apw/checkpatch.git