

開発履歴分析を用いたコードクローン内外における欠陥発生率の調査

中山 直輝, 吉田 則裕, 藤原 賢二, 飯田 元

出典：情報処理学会研究報告 Vol.2014-SE-186 No.2

コードクローンについて

コードクローンとは

ソースコード中に存在する互いに一致または類似したコード片

コードクローンの存在はソフトウェアの品質に悪影響を及ぼすと言われている

→このため、コードクローンに起因する欠陥を検出する手法が提案されている

コードクローンの例

```
File: sample1.c
17: if(len >= 9 && strncmp(buf, "EESOX", 9) == 0){
18:     buf += 9;
19:     len += 9;
20:
21:     if(len >= 5 && strncmp(buf, "term=", 5) == 0){
22:         .....
28:     } else
29:         ret = -EINVAL;
30: } else
31:     ret = -EINVAL;

File: sample2.c
62: if(len >= 9 && strcmp(buf, "CUMANA") == 0){
63:     buf += 9;
64:     len += 9;
65:
66:     if(len >= 5 && strncmp(buf, "term=", 5) == 0){
67:         .....
73:     } else
74:         ret = -EINVAL;
75: } else
76:     ret = -EINVAL;
```

コードクローンによる欠陥

コピー&ペーストしたプログラム内以外で欠陥が起こることがある

これまでの研究では、内外のどちらで多く欠陥が発生しており、開発者がどちらを優先すればよいのかの指標がなかった

コードクローンのどの部分を検査すれば効率的に欠陥対策できるのかわかる

優先度を決めるために 必要なもの

RQ1

コードクローン内外のどちらで欠陥が多く発生しているか.

RQ2

コードクローン外において特に欠陥が発生している範囲はどこか.

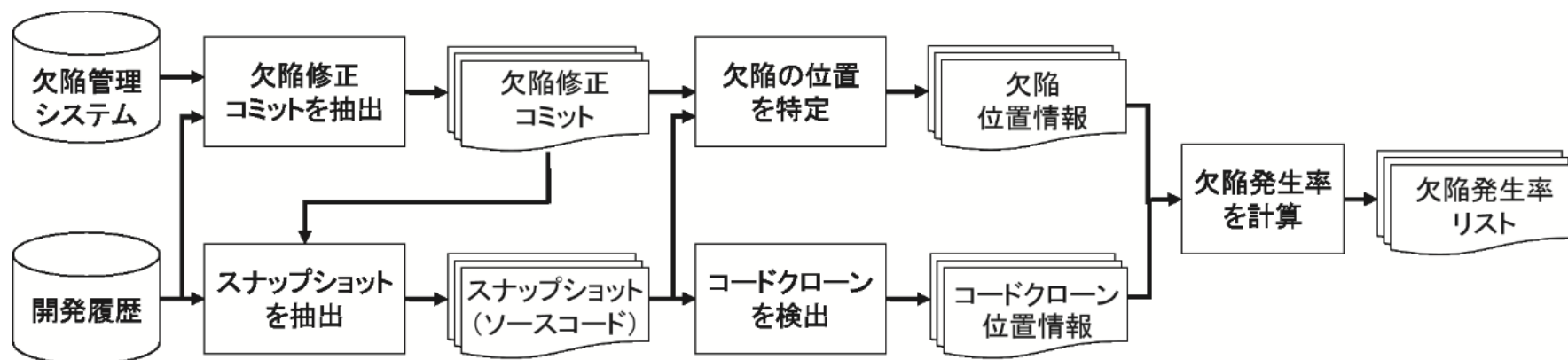
調査手法

調査対象 Cで開発された3つのOSS

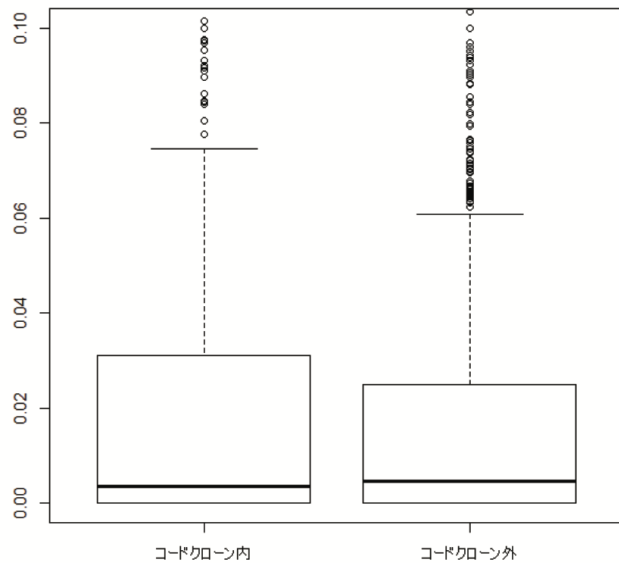
Table 1 Summary of studied projects.

	総スナップショット数	総 LOC	総ファイル数
gimp	82	851708	1018
evolution	41	721344	792
nautilus	60	251260	191

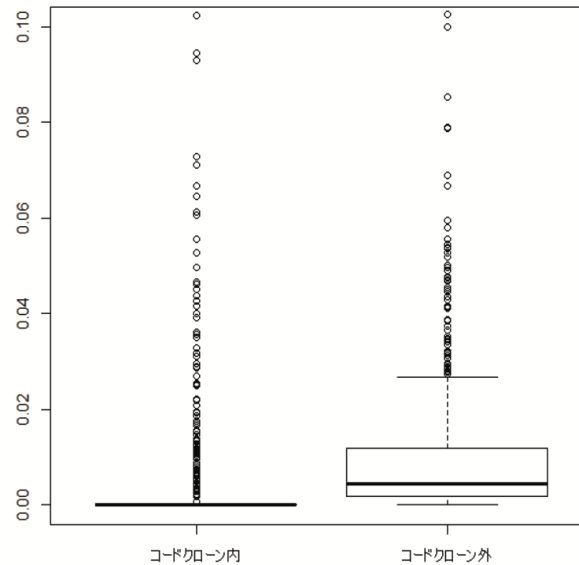
調査手法(概要)



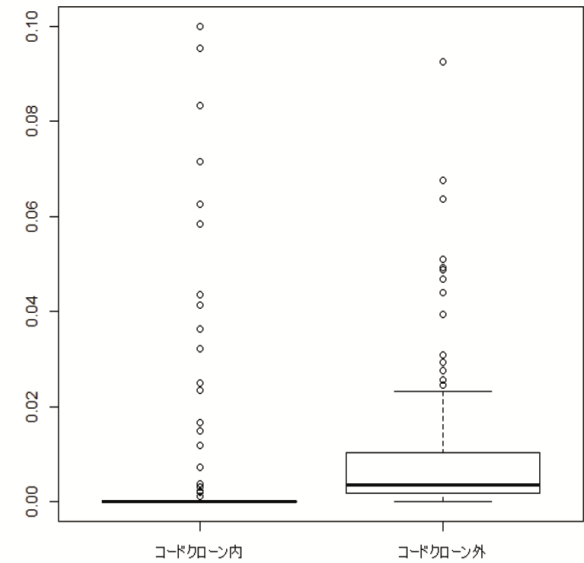
RQ1 コードクローン内外のどちらで欠陥が多く発生しているか.



gimp



evolution



nautilus

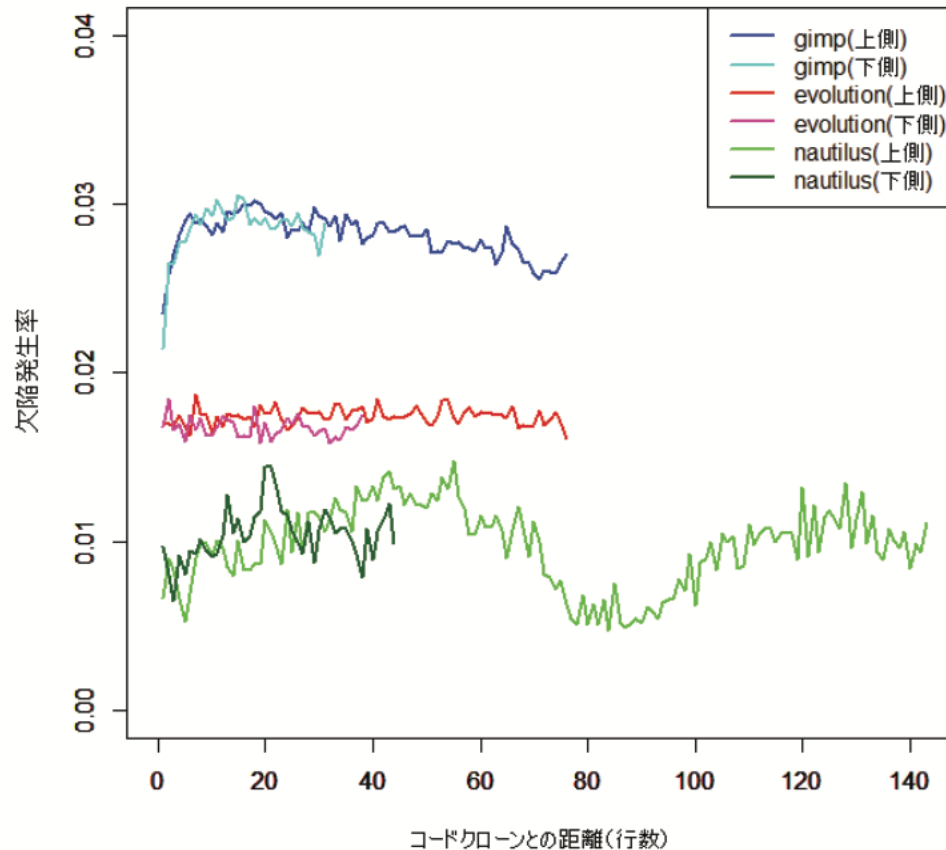
RQ1の結果

すべてのプロジェクトにおいてコードクローン外の中央値がコードクローン内の中央値に比べて高かった

	中央値（コードクローン内）	中央値（コードクローン外）	p 値（Wilcoxon の順位和検定）
gimp	0.003565	0.004619	p-value = 0.2783
evolution	0	0.003601	p-value < 2.2e-16
nautilus	0	0.004537	p-value < 2.2e-16

Wilcoxonの順位和検定の結果,
evolutionとnautilusでは欠陥発生率が
コードクローン内<コードクローン外 と言える

RQ2 コードクローン外において特に欠陥が多く発生している範囲はどこか。



RQ2の結果

上側ではコードクローンから遠ざかると欠陥発生率が低下している.

コードクローンの上下では欠陥発生率に差がない.

考察

RQ1

コードクローン外の欠陥発生率 > コードクローン内の欠陥発生率
→ 欠陥を探す際、コードクローン外も考慮する必要がある

RQ2

上側において 遠ざかるほど欠陥発生率が低下

上側と下側では差は見られなかった

→ コードクローンの上を検査する際、コードクローン近辺を優先的に検査すべき

課題

一般性のある知見を得るため、他OSSを対象とした調査を行う

他のコードクローン検出ツールを用いて再調査する

所感

コードクローンによって具体的にどの箇所で欠陥が発生するのか把握できた

考察という感じがしなかった