



コードクローン検出ツール 調査報告

野口 耕二郎



調査したクローン検出ツール

- CP-Miner

- Z. Li, S. Lu, S. Myagmar and Y. Zhou, "CP-Miner: finding copy-paste and related bugs in large-scale software code," in *IEEE Transactions on Software Engineering*, vol. 32, no. 3, pp. 176-192, March 2006.

- NICAD

- Chanchal K. Roy and James R. Cordy. 2008. NICAD: Accurate Detection of Near-Miss Intentional Clones Using Flexible Pretty-Printing and Code Normalization. ICPC '08 pages 172-181



クローンタイプ

- タイプ1
 - 空白，空行を除いて完全に一致
- タイプ2
 - 変数名や関数名が異なるが，それ以外は一致
- タイプ3
 - タイプ2の特徴に加え，1,2行の追加や削除がある



CP-Miner

- シーケンシャルパターンマイニングのアルゴリズムを用いてクローンを検出する
- タイプ3までのクローンを検出できる
- クローン特有の欠陥も検出できる
 - コピペした際の変数名の変更忘れによる欠陥
- 使用するためには以下のコード分析ツールを購入する必要ありか？
 - <http://patterninsight.com/>

CP-Miner

- シーケンシャルパターンマイニング

$$D = \{abced, abecf, agbch, abijc, akclc\}$$

サブシーケンス *abc* を含む赤字のシーケンスを抽出する

- コードをシーケンスに変換して上記のようなパターンマイニングを行うことでクローンを検出

STATEMENT	HASH
68 void __init prom_meminit(void)	35487793
69 {	
.....	
92 for(iter=0; iter<num_regs; iter++) {	67641265
93 prom_phys_total[iter].start_adr =	133872016
94 prom_reg_memlist[iter].phys_addr;	
95 prom_phys_total[iter].num_bytes =	133872016
96 prom_reg_memlist[iter].reg_size;	
97 prom_phys_total[iter].theres_more =	82589171
98 &prom_phys_total[iter+1];	
99 }	
.....	
111 for(iter=0; iter<num_regs; iter++) {	67641265
112 prom_prom_taken[iter].start_adr =	133872016
113 prom_reg_memlist[iter].phys_addr;	
114 prom_prom_taken[iter].num_bytes =	133872016
115 prom_reg_memlist[iter].reg_size;	
116 prom_prom_taken[iter].theres_more =	82589171
117 &prom_phys_total [iter+1];	
118 }	
.....	
143 }	

Fig. 2. An example of hashing statements.



NICAD

- コード変換ツールTxI^[1]を用いてクローンを検出するツール
- タイプ3までのクローンを検出できる
- 以下のWebページから無料でダウンロードして使用することができる
 - <https://www.txl.ca/txl-nicaddownload.html>

[1]:<https://www.txl.ca/>

NICAD

Table 3. Two Function Clones

Item No.	Sequence 1 (Original Segment)	Sequence 2 (Copied and Edited Segment)	Similarity
1	void	void	1
2	sumTimes	sumTimes	1
3	(int n) {	(int n) {	1
4	float sum=	float sum=	1
5	0.0;	0.0;	1
6	double product =	double product =	1
7	1.0;	1.0;	1
8	for (	for (	1
9	int i=1;	int i=1;	1
10	i<=n;	i<=n;	1
11	i++){	i++){	1
12	sum=	sum=	1
13	sum + i;	sum + (i * i);	0
14	product=	product=	1
15	product * i;	product * (i * i);	0
16	fun	fun	1
17	(sum, product);	(sum, product);	1
18	} }	} }	1
	Total Items = 18 Unique Items = 2 UPI = 11.11%	Total Items = 18 Unique Items = 2 UPI = 11.11%	

- Txlを用いてコードを右のような形に変換
- 2つのコードから最長共通部分列を検出
- 下にあるUPIを求め、閾値以下ならクローンとしている
- 本研究では閾値を30%としている

$$UPI = \frac{\text{No.of Unique Items} * 100}{\text{Total No.of Items}}$$

— クローンの間で一致しない要素の割合