産学連携による携帯端末用ソフトウェアに対する品質改善の取り組み

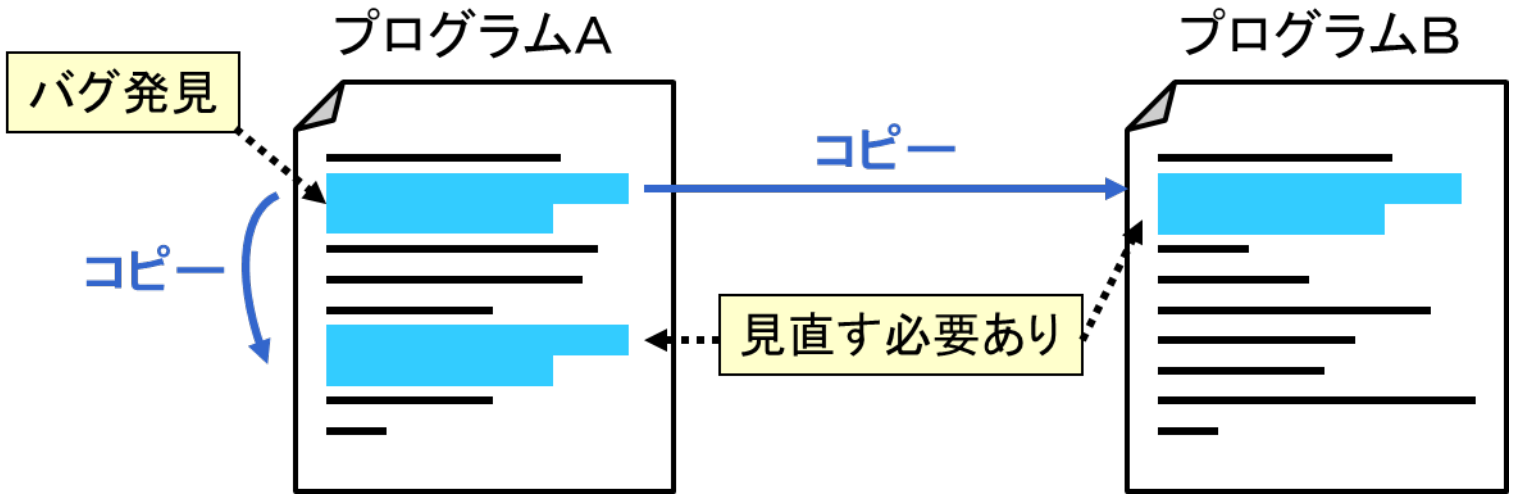
吉田則裕（奈良先端大），崔 恩瀞，肥後芳樹，楠本真二，井上克郎（阪大）

発表の流れ

- コードクローン研究と産学連携
- 産学連携の経緯
- 既存技術の試用
- コードクローン検査ツールの開発
- 携帯端末用ソフトウェアへの適用
- 今回の産学連携から得られた知見

コードクローン

- 同一，または類似したコード片
- コピー＆ペーストなどが原因で生じる
- ソフトウェア保守を困難にする要因の一つ
  - コードクローンとなっているコード片の一つを修正すると，他のコード片も修正の検討を行う必要がある

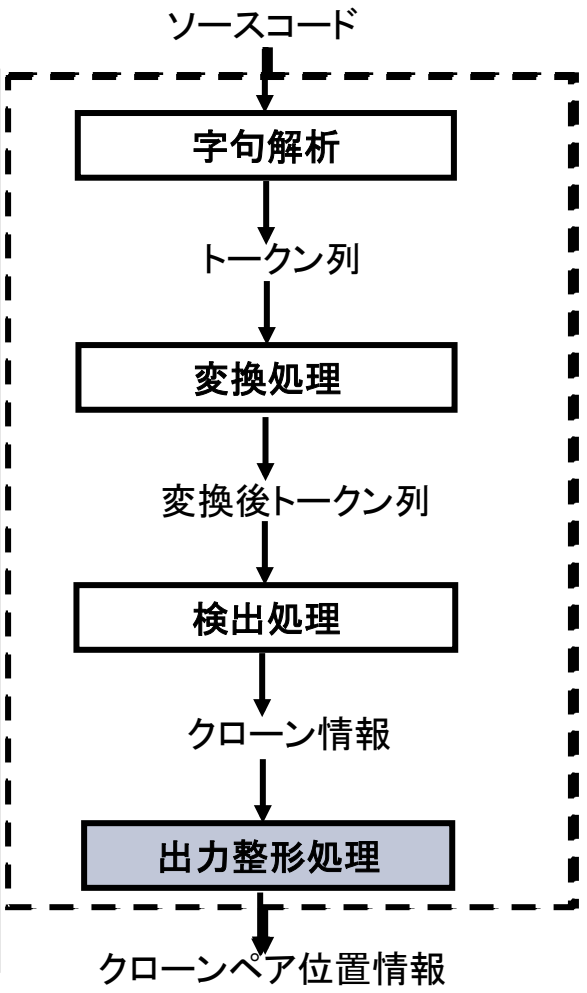


コードクローンの研究

- ソースコード中のコードクローンを検出する手法
  - CCFinder，CP-Miner，Scorpio等
- 集約支援手法
- コードクローンの一貫性検査に基づくバグ検出手法
  - CP-Miner等

CCFinder [Kamiya2002]の処理概要

```
1. static void foo() throws RESyntaxException {
2.    String a[] = new String [] { "123,400", "abc", "orange 100" };
3.    org.apache.regexp.RE pat = new org.apache.regexp.RE("[0-9,]+");
4.    int sum = 0;
5.    for (int i = 0; i < a.length; ++i)
6.       if (pat.match(a[i]))
7.          sum += Sample.parseNumber(pat.getParen(0));
8.    System.out.println("sum = " + sum);
9. }
10. static void goo(String [] a) throws RESyntaxException {
11.    RE exp = new RE("[0-9,]+");
12.    int sum = 0;
13.    for (int i = 0; i < a.length; ++i)
14.       if (exp.match(a[i]))
15.          sum += parseNumber(exp.getParen(0));
16.    System.out.println("sum = " + sum);
17. }
```



[Kamiya2002] T. Kamiya, et al.: CCFinder: A Multilinguistic Token-based Code Clone Detection System for Large Scale Source Code, IEEE Trans. Softw. Eng., 2002.

クローン研究における産学連携の意義 (1/2)

- 産業界における適用事例が強く求められている
  - 数多くの検出手法が提案されているが，産業界の適用事例はそれほど多くない
  - 国際会議等でも高く評価されることがある
    - 産業界への適用に重きが置かれる流れにある
  - 企業の開発プロジェクトでなければできない適用実験がある
    - 提示したコードクローンが有用かどうかは開発者でなければ，判断が難しい

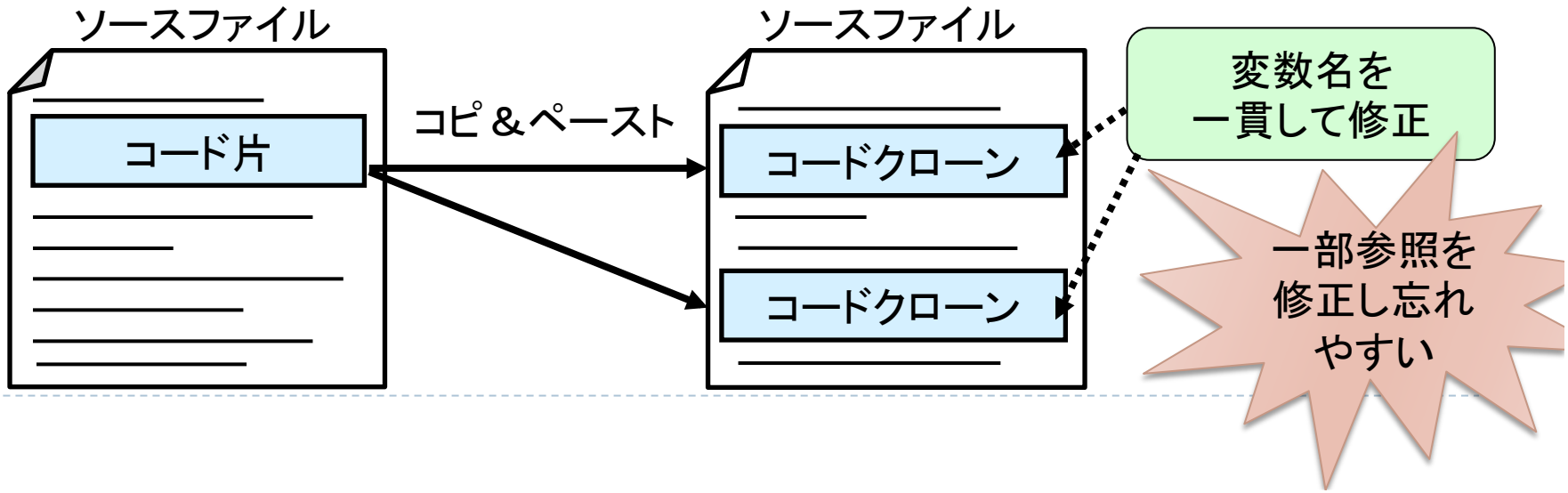
クローン研究における産学連携の意義 (2/2)

- コードクローン検出を応用できる新しいトピックを開拓したい
  - 同じ目的設定で精度改善することも重要だが，応用範囲の拡大も同時にやりたい
  - 企業側のニーズに合った研究をしたい

今回の産学連携の概要

- 相手企業
  - 携帯端末向けソフトウェアを開発する企業A
  - 再利用・修正することで，各国・地域向け版を開発
    - 全体で数億行

- 研究内容
  - コピーアンドペースト後の修正漏れを検出するツールの開発・適用



コピー&ペースト後の修正漏れ例

```
127:  o_count = v_count;
128:  o_var = varse;
129:  o_names = v_names;
130:
131:  v_count += STORE_INCR;
132:  varse = (char **) malloc (v_count*sizeof(char *));
133:  v_names = (char **) malloc (v_count*sizeof(char *));
134:
135:  for (indx = 3; indx < o_count; indx++)
135:      varse[indx] = o_var[indx];
137:
138:  for (; indx < v_count; indx++)
139:      varse[indx] = NULL;
        ......
161:  o_count = a_count;
162:  o_ary = arrays;
163:  o_names = a_names;
164:
165:  a_count += STORE_INCR;
166:  arrays = (char **) malloc (a_count*sizeof(char *));
167:  a_names = (char **) malloc (a_count*sizeof(char *));
168:
169:  for (indx = 1; indx < o_count; indx++)
170:      varse[indx] = o_ary[indx];
171:
172:  for (; indx < v_count; indx++)
173:      lists[indx] = NULL;
```

変数名v_countを
a_countに置換

置換忘れ

産学連携の経緯

企業A

携帯端末用ソフトウェアの開発を行なっている

品質保証部門 B氏

・テスト工程では，取りきれない残存バクが多くなっている
・コピーアンドペーストにより開発した部分に残存バグが多い

コンタクト

著者らの研究グループ

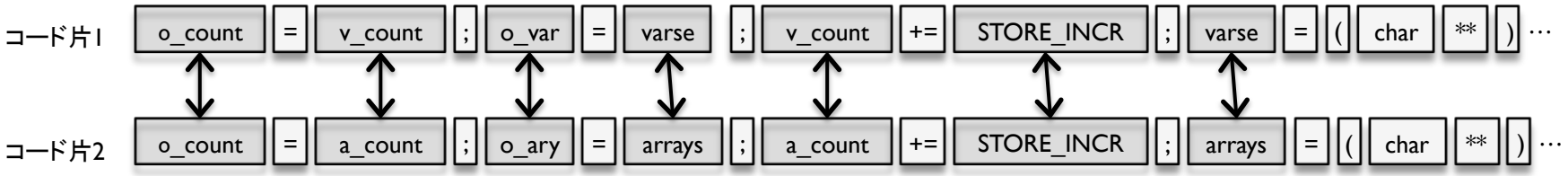
コードクローンに関する研究を行なっている

産学連携のスケジュール（抜粋）

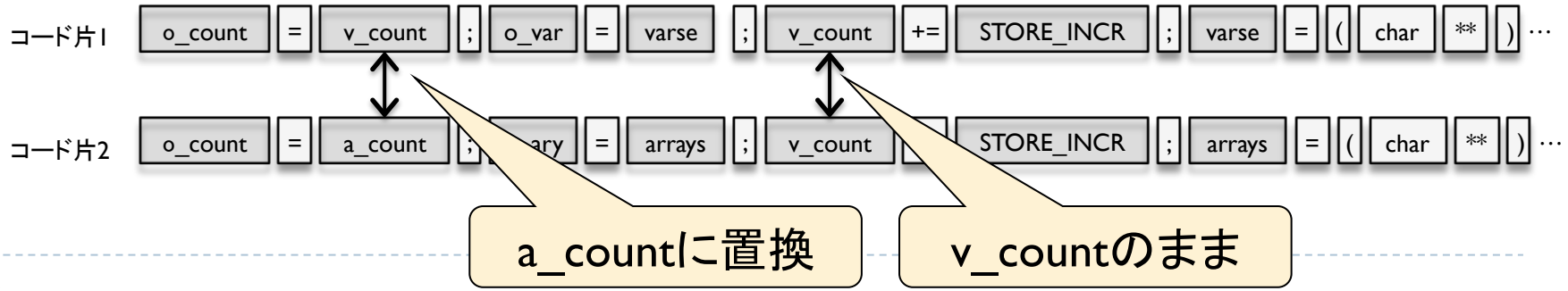
- 1ヶ月目
  - 企業：問題提起
  - 大学：技術紹介
- 6ヶ月目
  - 企業：既存ツールの適用，および問題点の発見
- 10ヶ月目
  - 大学：著者らが企業Aを訪問し，新ツールの仕様について議論
- 12ヶ月目
  - 企業：B氏らが阪大に滞在し，新ツールを開発
  - 企業：B氏らが携帯端末用ソフトウェアに対して適用実験

CP-Minerが行う識別子名の一貫性検査

- まずは，識別子名の対応関係を検査する手法を既存研究として，企業Aに紹介
  - コードクローン間で，識別子が1対1に対応する → バグなし
    - 完全に一致するか，もしくは一貫した置換が行われている



  - コードクローン間で，識別子が1対1に対応しない → バグあり



一貫性検査ツール[Yii2008]の適用

- 我々の研究グループが以前開発した一貫性検査ツール
  - CCFinderの検出したコードクローンの一貫性を検査し，バグを含むものを抽出
    - CP-Minerと同様に，識別子名の一貫性を検査する
  - 誤検出の削減
    - 3種類以上の識別子に置換された場合をフィルタリング
  - 手法の有効性調査のために開発されたプロトタイプ

[Yii2008] Yii Yong Lee, Yasuhiro Hayase, Makoto Matsushita, Katsuro Inoue: "Token Comparison Approach to Detect Code Clone-related Bugs", 電子情報通信学会技術研究報告, SS2007-64, Vol.107, No.505, pp.43-48, 2008

スケーラビリティの問題

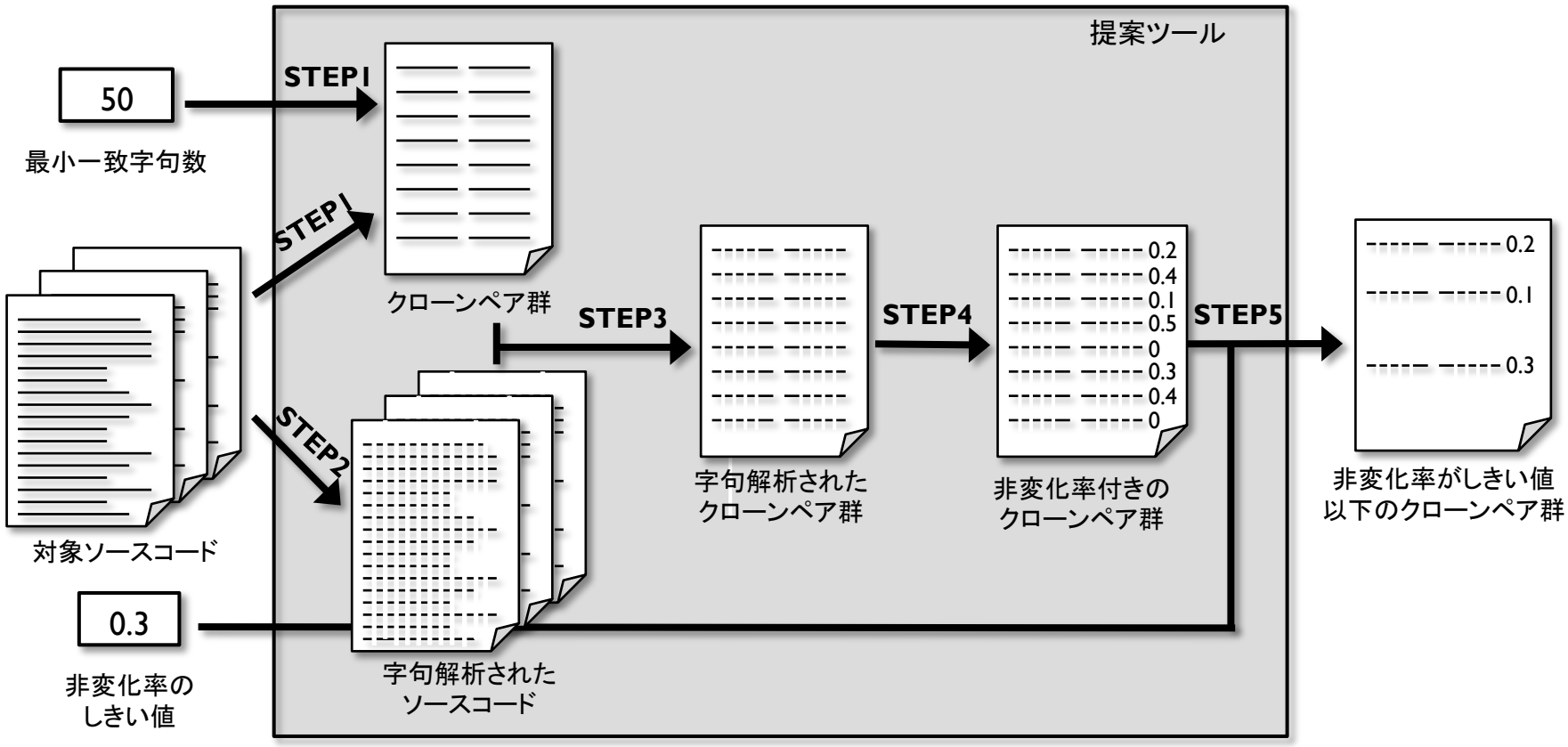
- 予備実験として，企業Aのコードに適用したところ，数百万行程度のソースコードを分析できなかった.
  - オープンソースプロジェクトと比較して，クローンの量が多い
  - 最低でも，数百万行程度のコードには適用する必要がある
    - 全体では，数億行の資産がある
  - できるだけ高速に分析できる方が良い
- 全ソースコードの字句解析結果をメモリに格納したあとで一貫性を検査していた.

新ツールの開発

- 企業AのB氏ら2名により，新ツールを開発
  - 1つのクローンセット（同値類）の一貫性検査が終わるたびに，メモリを解放することで，スケーラビリティを向上
  - B氏らが阪大に滞在し，著者らのアドバイスを踏まえながら開発
    - コミュニケーションを密にとりながら，短期間で開発を行うことができた
  - A社のニーズに合わせたツールの開発

新ツールの適用実験

- 企業Aが開発したソフトウェアに適用した.



実験対象と結果

- ソフトウェアX
  - 用途：通信
  - 言語：C
  - 行数：約400万行

63個の修正漏れ候補を抽出

開発者による検証

そのうち，25個が未知のバグであったため，修正が行われた

- ソフトウェアY
  - 用途：アプリケーション
  - 言語：C
  - 行数：約14万行

5個の修正漏れ候補を抽出

開発者による検証

そのうち，1個が未知のバグであったため，修正が行われた

スケーラビリティ

- 使用した計算機
  - CPU Intel Core i5
  - メモリサイズ4GB

← 4GBで数百万行のソースコードを分析可能

- ソフトウェアX
  - 用途：通信
  - 言語：C
  - 行数：約400万行

計443秒（うちCCFinderが300秒）

- ソフトウェアY
  - 用途：アプリケーション
  - 言語：C
  - 行数：約14万行

計40秒（うちCCFinderが4秒）

適用結果についての企業Aの感想

- 修正漏れ候補の数が現実的
  - 63個もしくは5個であり，人手による確認作業が可能である
- スケーラビリティは十分
  - コードクローン検出時間とあわせても，8分以内

今回の産学連携の特徴，得た知見 (1/3)

- 今回は珍しいタイプの産学連携を行った
  - タイプA（手法の提案は大学，開発，実験は企業）
    - 企業：ニーズの提供，データ提供，適用実験，ツールの開発
    - 大学：手法の考案
  - タイプB（手法の提案，開発は大学，実験は企業）
    - 企業：ニーズの提供，データ提供，適用実験
    - 大学：手法の考案，ツールの開発
  - タイプC（手法の提案，開発，実験まで大学）
    - 企業：ニーズの提供，データ提供
    - 大学：手法の考案，ツールの開発，適用実験

珍しいケース

今回の産学連携の特徴，得た知見 (2/3)

- 企業側が開発・実験を行うことで効率的な連携ができた
  - 企業側への技術移転が短期間でできた
  - 企業側の都合に合わせたツールの開発ができた
    - 企業側による適用実験につながった
    - 大学でツールを開発すると，大学側が企業に赴いて適用実験をしなければいけなくなり，オーバーヘッドが大きい

大学側がソースコードの閲覧・実験を行わなかったため，
見落としている問題がある可能性もある

今回の産学連携の特徴，得た知見 (3/3)

- 産学連携の目的が比較的はっきりしていたため，効率的に研究を行うことができた
  - コピー＆ペーストが引き起こすバグの検出に特化した連携を行った.
  - 目的が曖昧な産学連携は，経験上あまりうまくいかないケースが多い.
  - 本研究は昨年度行った内容であり，今年度分は目的を再設定して進行中である

まとめ

- 産学連携により，携帯端末向けソフトウェアを対象としたコードクローンの一貫性検査を行った．
  - ソフトウェアXでは，25個の未知のバグを発見できた
- 企業側が開発・実験を行うことで効率的な連携ができた
  - 大学側がソースコードの閲覧・実験を行わなかったため，見落としている問題がある可能性がある
- 産学連携の目的が比較的はっきりしていたため，効率的に研究を行うことができた