

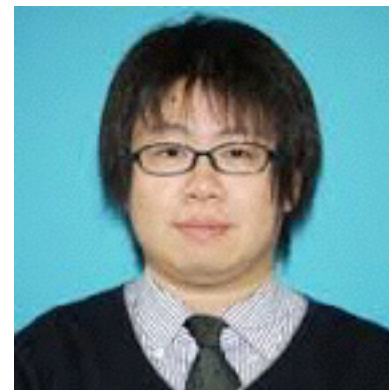
Guilty or Not Guilty: Using Clone Metrics to Determine Open Source Licensing Violations



Akin Monden
Graduate School of
Information Science
at NAIST



Satoshi Okahara
System Engineer
at TIS



Yuki Manabe
Graduate School of
Information Science
and Technology
at Osaka Univ.



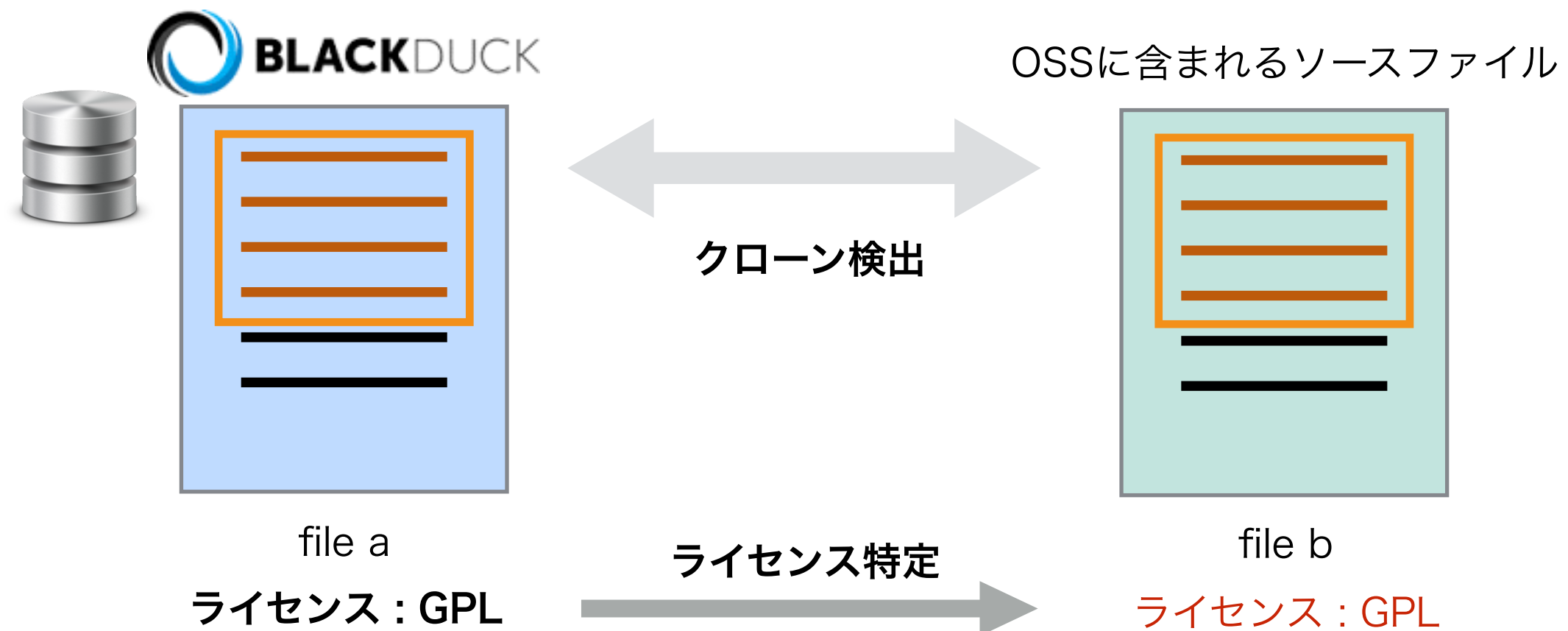
Kenichi Matsumoto
Graduate School of
Information Science
at NAIST

背景

- ✧ 企業のソフトウェア開発では、生産性向上のために、OSSを再利用する
 - 開発者は、オンラインでOSSのソースコードに容易にアクセスできるようになってきた
- ✧ 一方で、OSS再利用におけるライセンス違反も広まってきた
 - OSSを再利用するには、OSSライセンスを遵守しなければならない
- ✧ 企業のOSS再利用におけるライセンス違反を解消するため、OSSライセンス管理サービスが展開されている

Protex (Black Duck社)

- 大量のKnowledge Baseを用いて、ソースファイルのライセンスを特定
- 複数含まれているライセンスの条項から矛盾している条項を検出



他に、Protexと似たものでPalamidaというツール(Palamida社)もある

問題

- ✧ たまたま同じコードになってしまったコードクローンを考慮していない
 - ProtexやPalamidaは、コードクローン技術を用いている
 - たまたま同じコードになってしまったものは、流用ではないないため、罪ではない
- ✧ 流用によるコードクローンか、たまたまクローンになってしまったのか判別する技術が必要である

研究の目的

- ✧ 流用によるコードクローン(Guilty), と自然にできてしまったコードクローン(Not Guilty)を判別する
- ✧ アプローチ
 - クローンが長いものは, 流用によるコードクローンである可能性が高い
 - クローンメトリクスを使って判別できないか
 - 流用によるものか, そうでないかは, プロのプログラマーだったら目視で判別できるらしい

コードクローンメトリクス

- ✧ MLC (Maximum Length of Clones)

- 最大コードクローンのトークン数
 - ★ トークン数が長い（クローンが長い）

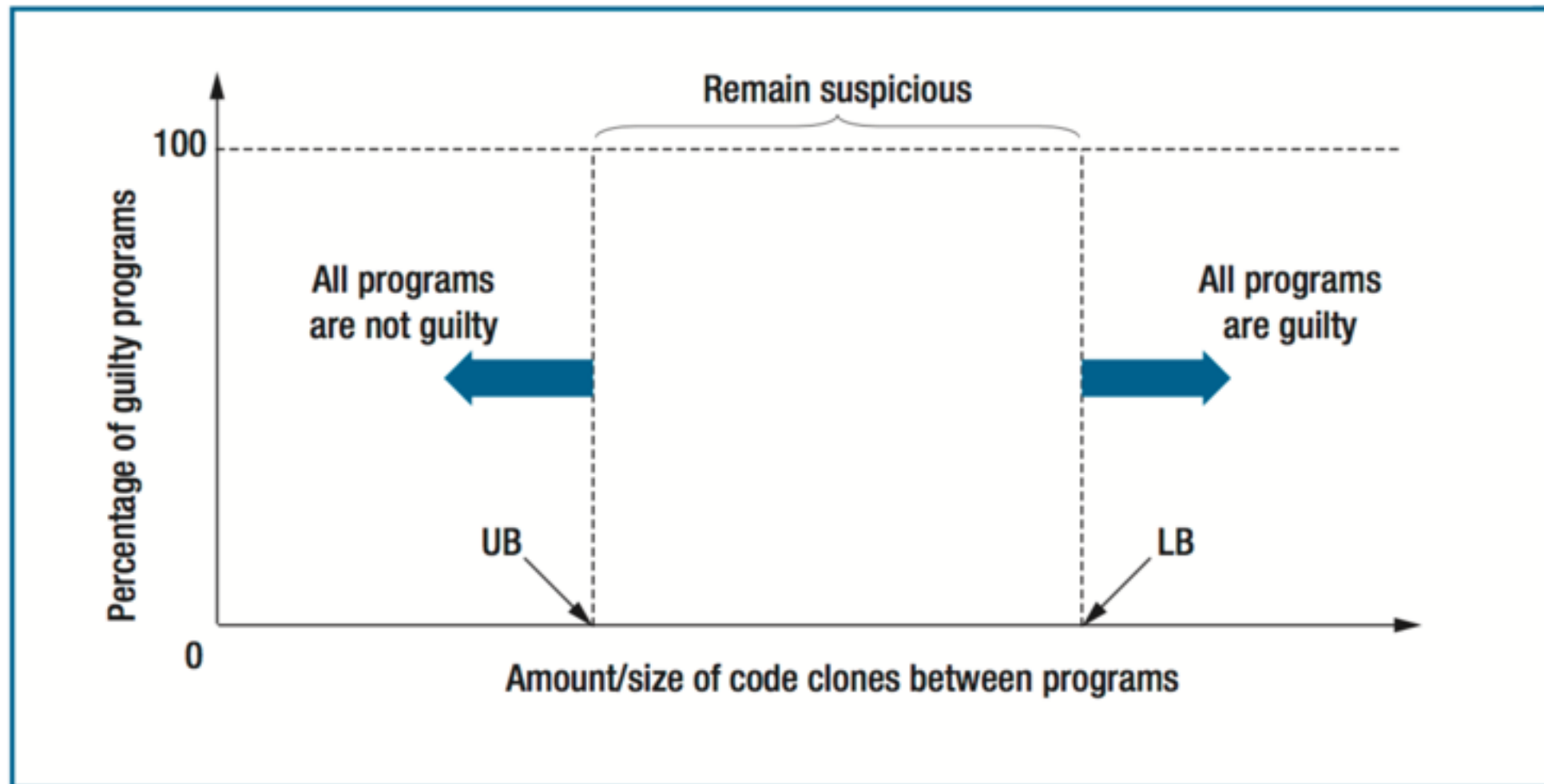
- ✧ LSim(Clone-Based Local Similarity)

- MLCのトークンを含むファイルの含有率

$$LSim(A, B) = \frac{2 \times MLC(A, B)}{|a| + |b|} \times 100, \quad (1)$$

本研究で取り組む課題

- ✧ クローンメトリクスにおけるGuilty, Not Guiltyの境界値（閾値）はどれくらいか



実験

✧ 実験対象

1. 様々なドメインからOSSを50個抽出
2. 1225(50C2)のOSSペアを作成し, CCFinderで最大コードクローンを検出

✧ 実験方法

- 1. プロ開発者が, 目視でクローンが流用によるものか, そうでないのかを判別していく
 - ★ 1225の内訳 (流用あり:121, 流用なし:648, 対象外:456)
- 2. 流用ありクローン集合と, 流用なしクローン集合で, Precisionが100%になる閾値を各メトリクスで調査する

実験結果：MLCの閾値

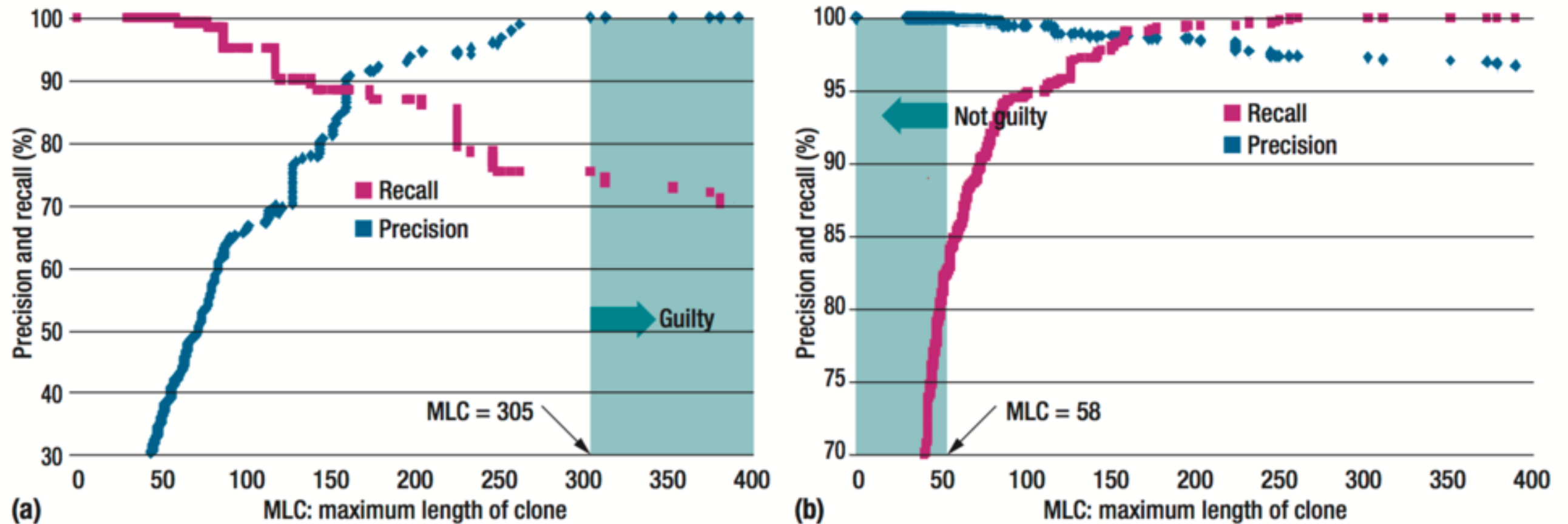


FIGURE 2. Resulting MLC values. (a) Precision and recall for identifying guilty products by MLC. We identified an MLC of 305 as the lower bound for a guilty verdict. (b) Precision and recall for identifying not guilty products by MLC. We identified an MLC of 58 as the upper bound for a not guilty verdict—that is, all pairs are not guilty if they contain clones less than or equal to 58 tokens.

流用あり判別の閾値: MLCが**305**以上だとPrecisionが100%

流用なし判別の閾値: MLCが**58**以下だとPrecisionが100%

実験結果：LSimの閾値

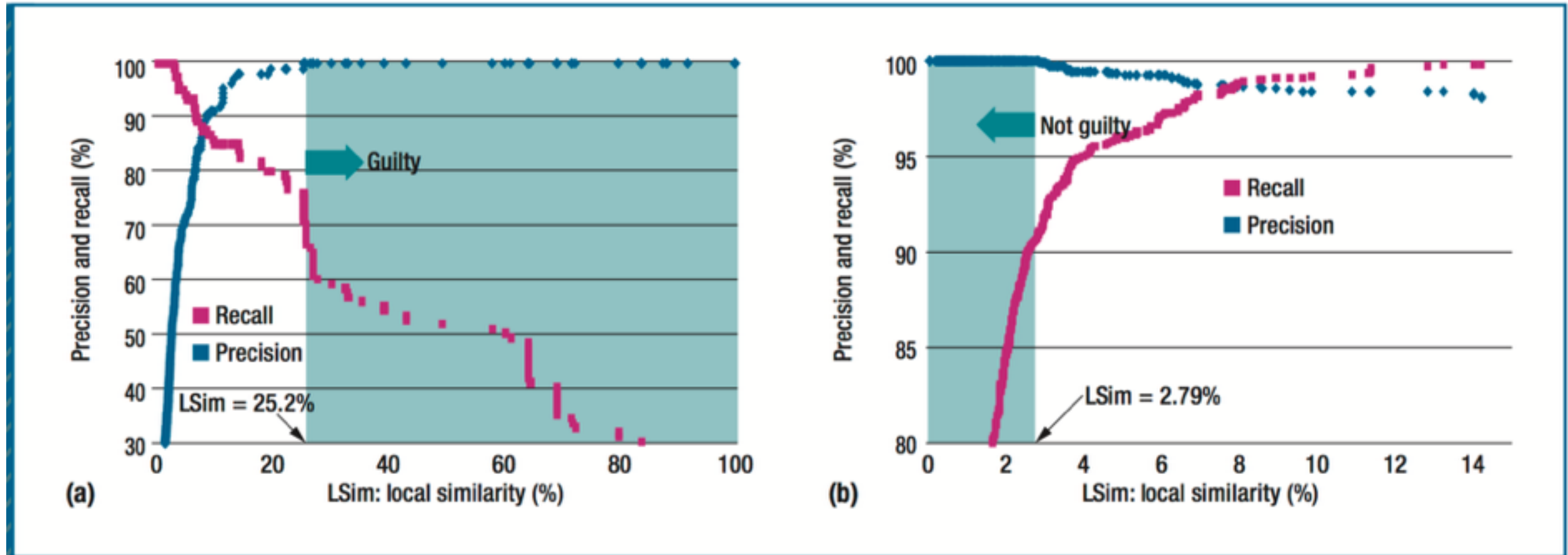


FIGURE 4. Resulting LSim values. Identification of (a) guilty and (b) not guilty product pairs. A product is guilty if its clone coverage is greater than 25.2 percent and not guilty if coverage is less than or equal to 2.79 percent.

流用あり判別の閾値: Lsim **25.2%**以上

流用なし判別の閾値: Lsim **2.79%**以下

Lsimが有用な例

＊ 短いコードクローンが流用によるものかどうかの判別

```
/* Splice in "list" into "head" */
static inline void glame_list_splice(struct glame_list_head *list,
struct glame_list_head *head)
{
    struct glame_list_head *first = list->next;

    if (first != list) {
        struct glame_list_head *last = list->prev;
        struct glame_list_head *at = head->next;

        first->prev = head;
        head->next = first;

        last->next = at;
        at->prev = last;
    }
}
```

```
/*
 * Splice in "list" into "head"
 */
static INLINE void list_splice(struct list_head *list, struct list_head
*head)
{
    struct list_head *first = list->next;

    if (first != list) {
        struct list_head *last = list->prev;
        struct list_head *at = head->next;

        first->prev = head;
        head->next = first;

        last->next = at;
        at->prev = last;
    }
}
```

このクローンにおけるメトリクス値・・・MLC : 59, Lsim 0.32

流用あり判定における閾値・・・MLC : 305以上, **Lsim 0.252**以上

考察

- ✧ MCLの閾値調査結果について
 - 長いコードクローンは流用によるもの、短いコードクローンは偶然作成されたものである可能性が高い
- ✧ Lsimの閾値調査結果について
 - MCLを用いた場合より、精度（Recall）が低いことが分かった
 - MCLで判別できなかったものをLsimで判別したところ30件中11件を判別することができた
- ✧ 121件の流用によるクローンを101件(84%)判別することができたため、併用すれば有用であると言える

まとめ

- ❖ コードクローンが流用されたものか, 偶然作成されたものかどうかをメトリクスを用いて判別できることを示した.
 - 流用によるコードクローンのうち84%(101/121)判別に成功した
- ❖ 今後の課題
 - 他に利用できるメトリクスはないか模索する