A Test Case Recommendation Method Based on Morphological Analysis, Clustering and the Mahalanobis-Taguchi Method

Hirohisa Aman[1)]

Takashi Nakano[2)]

Hideto Ogasawara[2)]

Minoru Kawahara[1)]

1) Ehime University, Japan
2) Toshiba Corporation, Japan

Overview

Purpose

To **recommend similar but different** test cases in order to reduce the risk of **overlooking regressions**

Method

Quantify the **similarity** between test cases through the **morphological analysis**, and categorized them (**clustering**)

Once **a test case is selected by a test engineer**, the proposed method **automatically recommends additional test cases** based on the results of clustering

Result

The proposed method is about **six times more effective** than the random test case selection; it would be useful in making a regression test plan

Outline

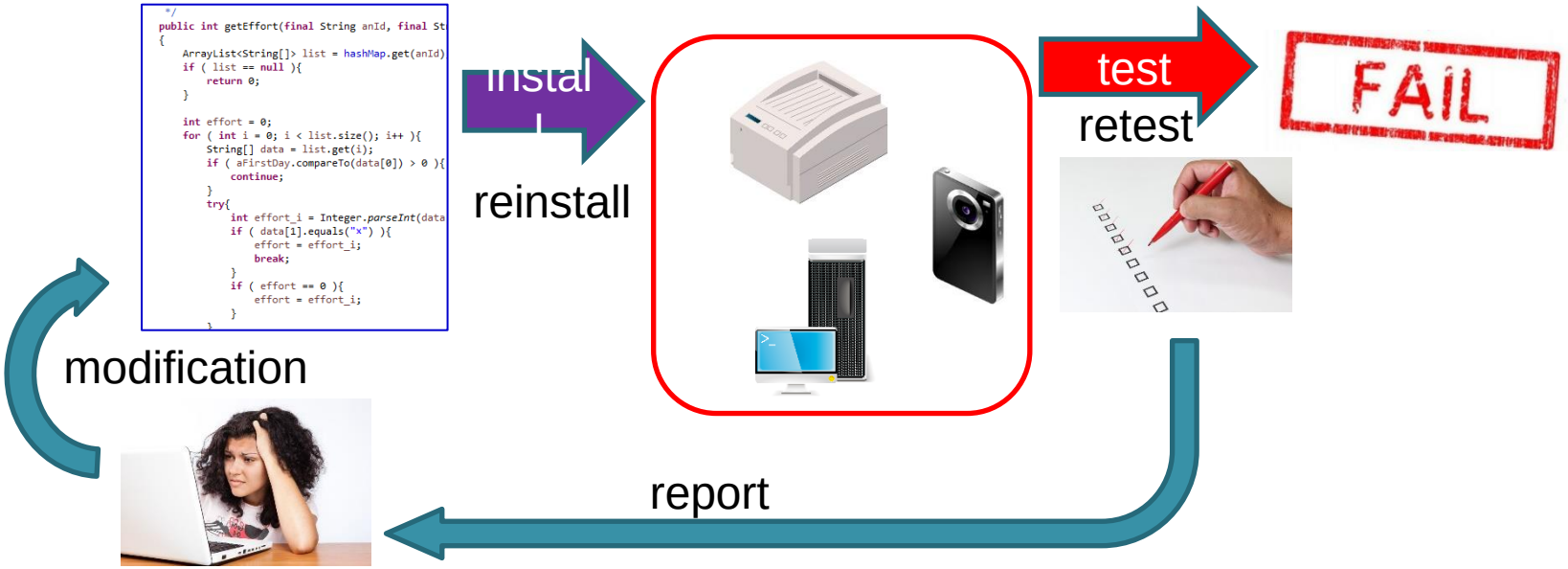
- Background, Motivation & Situation
- Test Case Recommendation
  - Morphological Analysis
  - Test Case Clustering
  - Test Case Prioritization
- Empirical Study
- Related Work
- Conclusion & Future Work

Outline

- **Background, Motivation & Situation**
- Test Case Recommendation
  - Morphological Analysis
  - Test Case Clustering
  - Test Case Prioritization
- Empirical Study
- Related Work
- Conclusion & Future Work

Background: Regression Testing

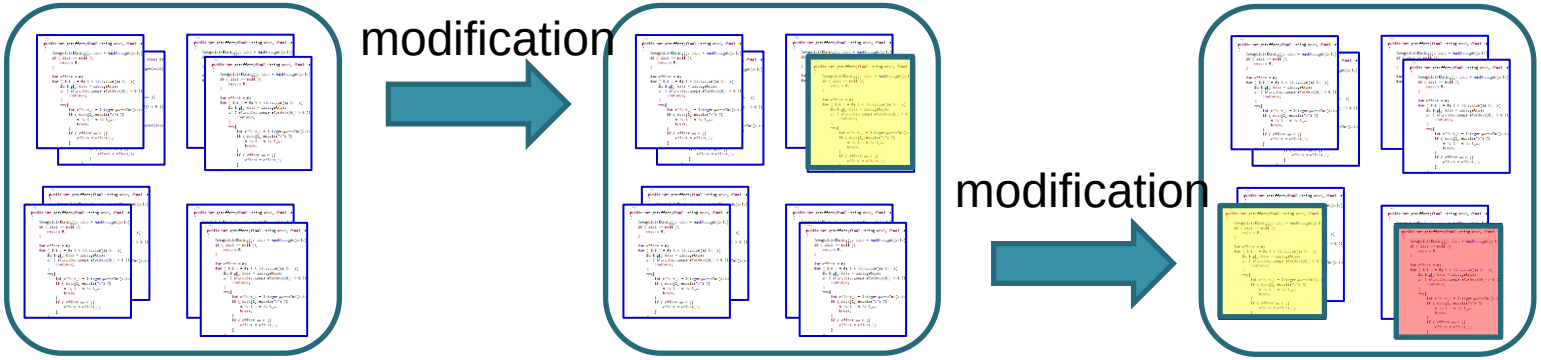
- In fact, it is **difficult** to **always make a one-shot release** of **a perfect product** which has no need to be modified in the future



- Program modifications may cause **other failures** (regressions)

Motivation: Unexpected Failures & Testing Cost

- We may encounter **unexpected failures** in **unexpected functions** after modifications

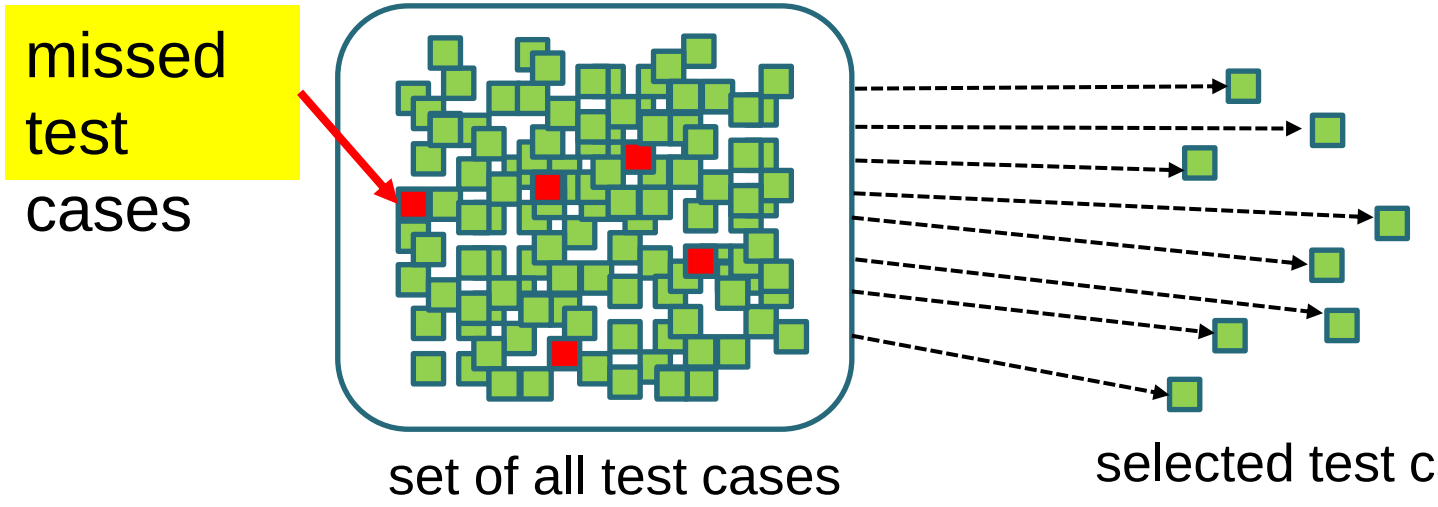


Unexpected failure in another function which **seemed to be independent** of the modified functions!

- While it is ideal to **rerun all test cases** every time, we have the restriction of **cost**…

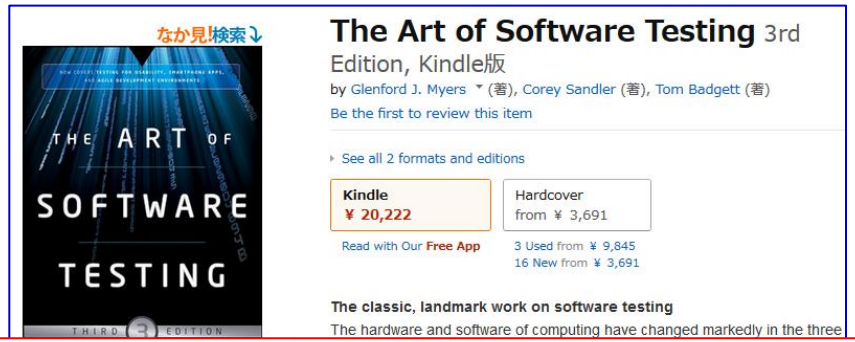
Motivation: Risk of Overlooking regressions

- We have **a lot of test cases**, and it's **unrealistic** to rerun **all** of them **whenever** a modification is made
- We have to **select test cases**, but there is the **risk of overlooking** regressions since we might miss rerunning important test cases



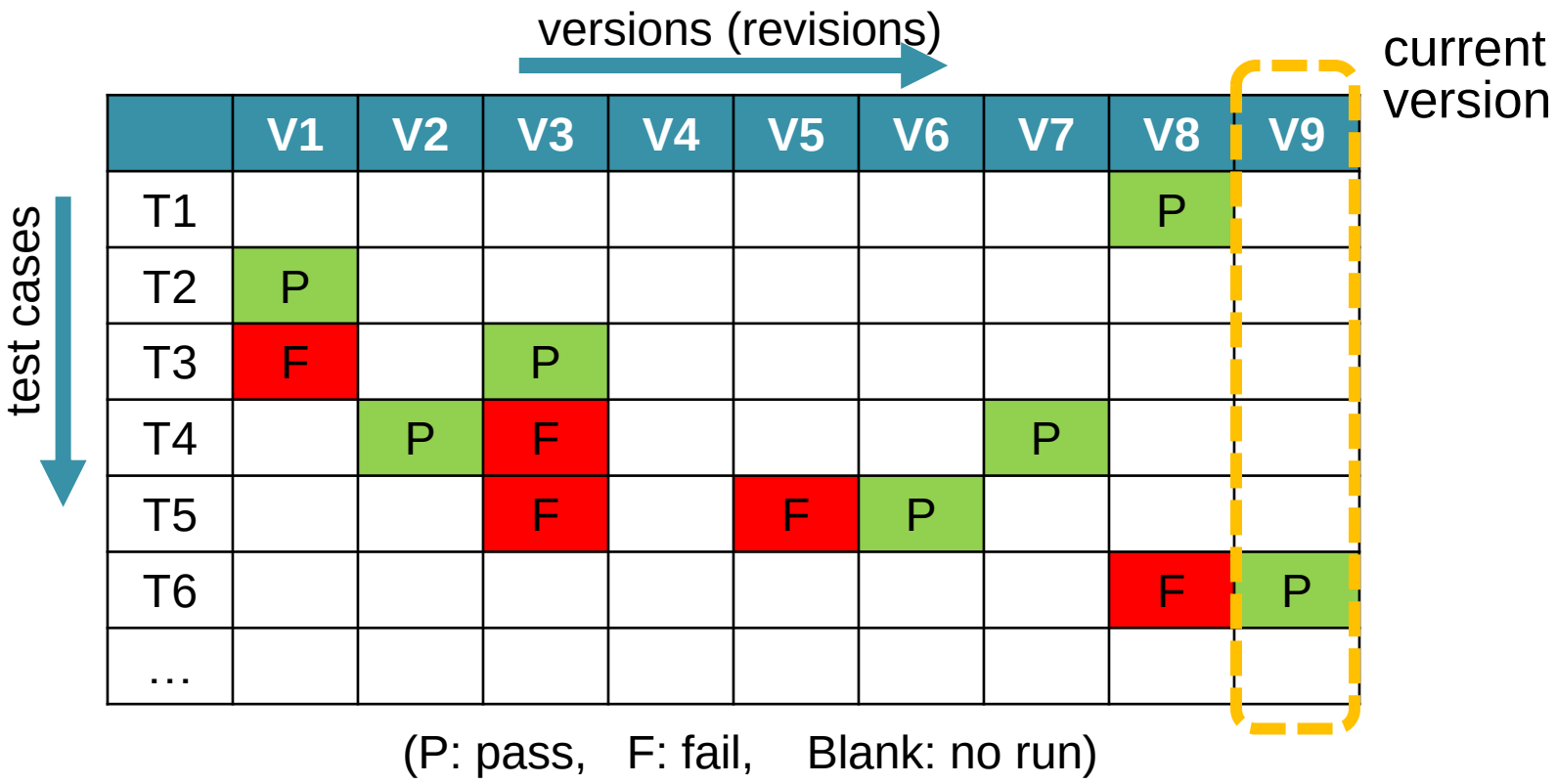
Motivation: Automated Recommendation in Use

- When you look at a book on Amazon.com



Can we recommend appropriate test cases in an automated way?

Our Available Data

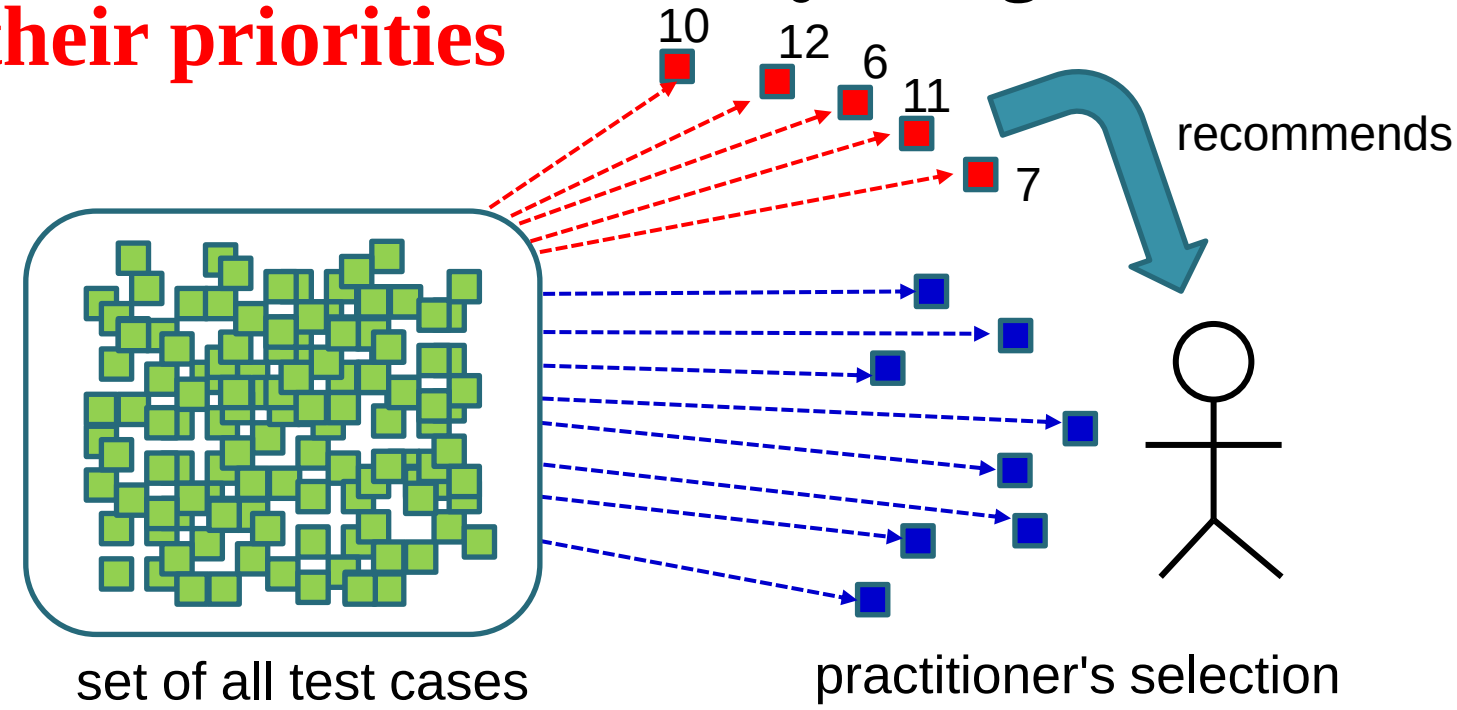


Outline

- Background, Motivation & Situation
- **Test Case Recommendation**
  - Morphological Analysis
  - Test Case Clustering
  - Test Case Prioritization
- Empirical Study
- Related Work
- Conclusion & Future Work

Scenario for Our Test Case Recommendation

1. For each version, a practitioner **decides** on a set of test cases to rerun (R_0)
2. We **recommend** another set of test cases **similar** to the ones in R_0 in regards to **their priorities**

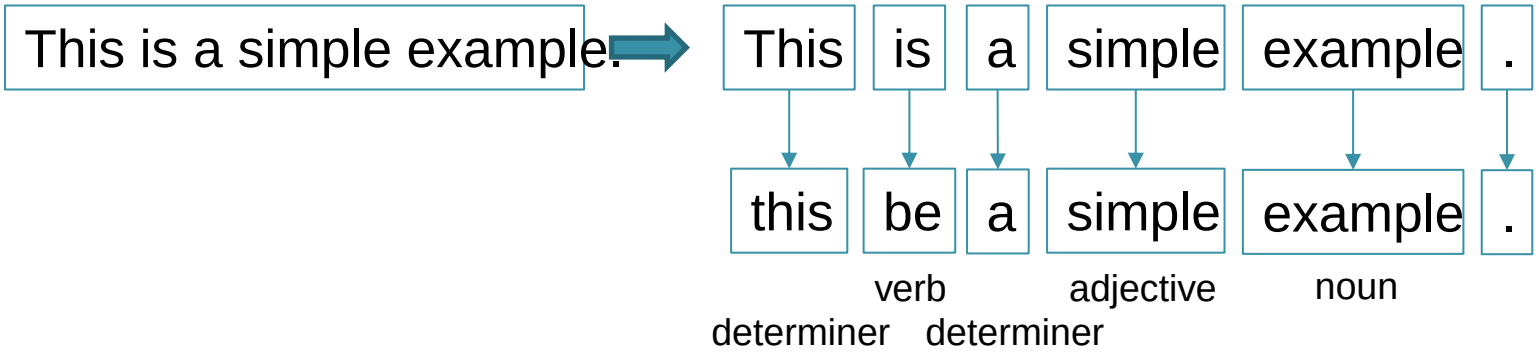


Outline

- Background, Motivation & Situation
- **Test Case Recommendation**
  - **Morphological Analysis**
  - Test Case Clustering
  - Test Case Prioritization
- Empirical Study
- Related Work
- Conclusion & Future Work

Morphological Analysis

- A **morphological analysis** is used to **analyze texts written in a natural language**
- It divides text strings into **component words** and detects their **parts of speech** (noun, verb, …)



- There are many applications of it like **machine translations**

Analysis of Our Test Case

- Our test case is **written in Japanese**
- A test engineer performs his/her test according to the test case

An example of a test case (translated into English)

A project creation:
Enter a name of project, and check if we can successfully create a new project on the system.
The length of project's name should be around 10 characters.

- We used **MeCab** (one of the most popular morphological analysis tool for Japanese), and **extracted a set of words** (nouns, adjectives and verbs)

Similarity between Test Cases

- We compute the similarity between test cases t_i and t_j by using the **Jaccard index**:

$$J(t_i, t_j) = \frac{|W_i \cap W_j|}{|W_i \cup W_j|}$$

  - W_i: the set of words in test case t_i
  - W_j: the set of words in test case t_j

- This is a simple but useful index; **it has been widely used** in the natural language processing world

Example

- Suppose our sets of words are

W_1	button, click, chronological, date, display, download, file, log, order
W_2	archive, button, click, chronological, date, download, file, order

$W_1 \cap W_1$	button, click, chronological, date, download, file, order
$W_1 \cup W_2$	archive, button, click, chronological, date, display, download, file, log, order

7

10

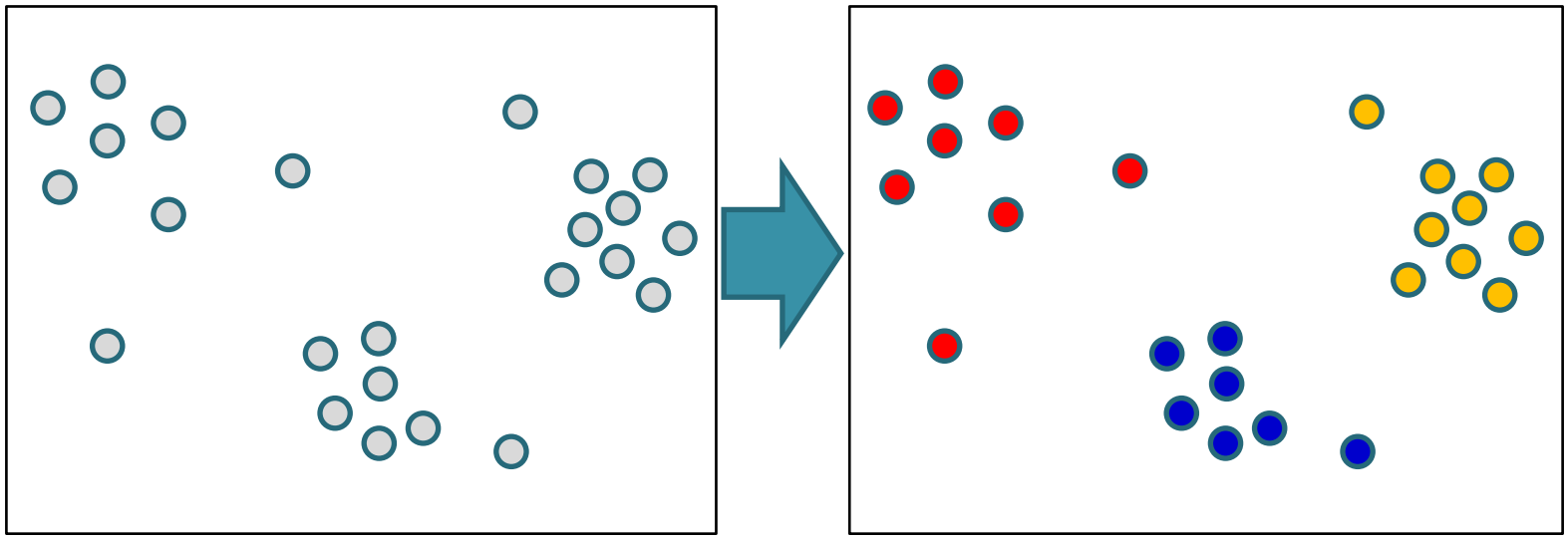
$$J(t_1, t_2) = 0.7$$

Outline

- Background, Motivation & Situation
- **Test Case Recommendation**
  - Morphological Analysis
  - **Test Case Clustering**
  - Test Case Prioritization
- Empirical Study
- Related Work
- Conclusion & Future Work

Clustering

- **Clustering** is the task of **grouping a set of objects together** (making a **cluster**)
- Objects belonging to the same group are **more similar** to each other than they are to objects of other groups



Test Case Clustering

- Define the distance between test cases

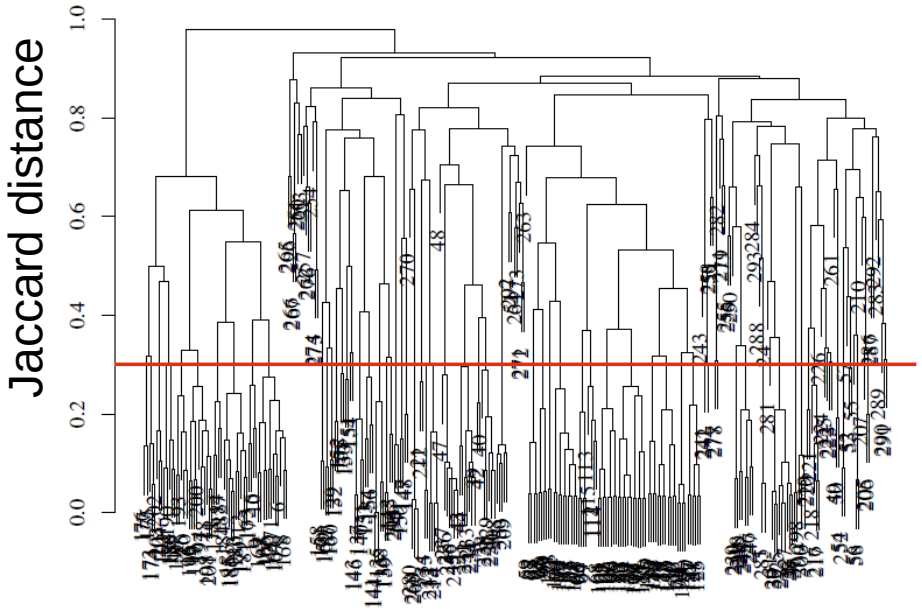
$$d(t_i, t_j) = 1 - J(t_i, t_j)$$

This is referred to as **Jaccard distance**

- Then, perform a clustering
  - We used **hclust** function in **R** (a popular statistical computing environment)
  - The function performs a hierarchical cluster analysis with the complete linkage method

Dendrogram (tree diagram)

- We can obtain the results of clustering



cut level

we will group test cases whose distances are less than the cut level in the same cluster

- We empirically set **0.3** as the **cut level**: we consider that two test cases are similar when their Jaccard index ≥ 0.7 $(= 1 - 0.3)$

Outline

- Background, Motivation & Situation
- **Test Case Recommendation**
  - Morphological Analysis
  - Test Case Clustering
  - **Test Case Prioritization**
- Empirical Study
- Related Work
- Conclusion & Future Work

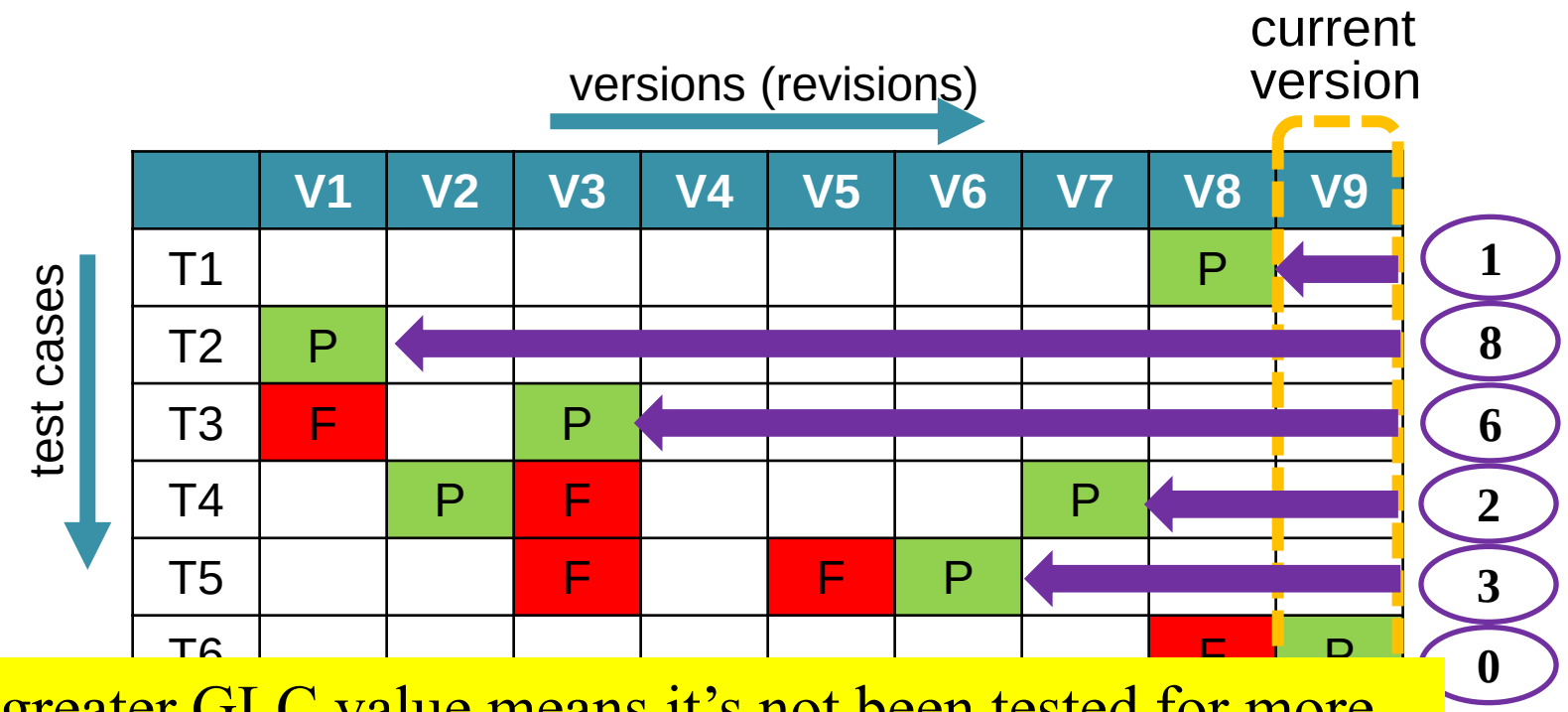
Test Case Prioritization

- After our test case clustering, we select test cases to rerun
- Within a cluster, we **prioritize** certain test cases

- We have empirically used two criteria:
  I. **Gap between the Last run version and the Current version** (**GLC**)
  II. **Failure Rate** (**FR**)

Priority of a Test Case: Type-I

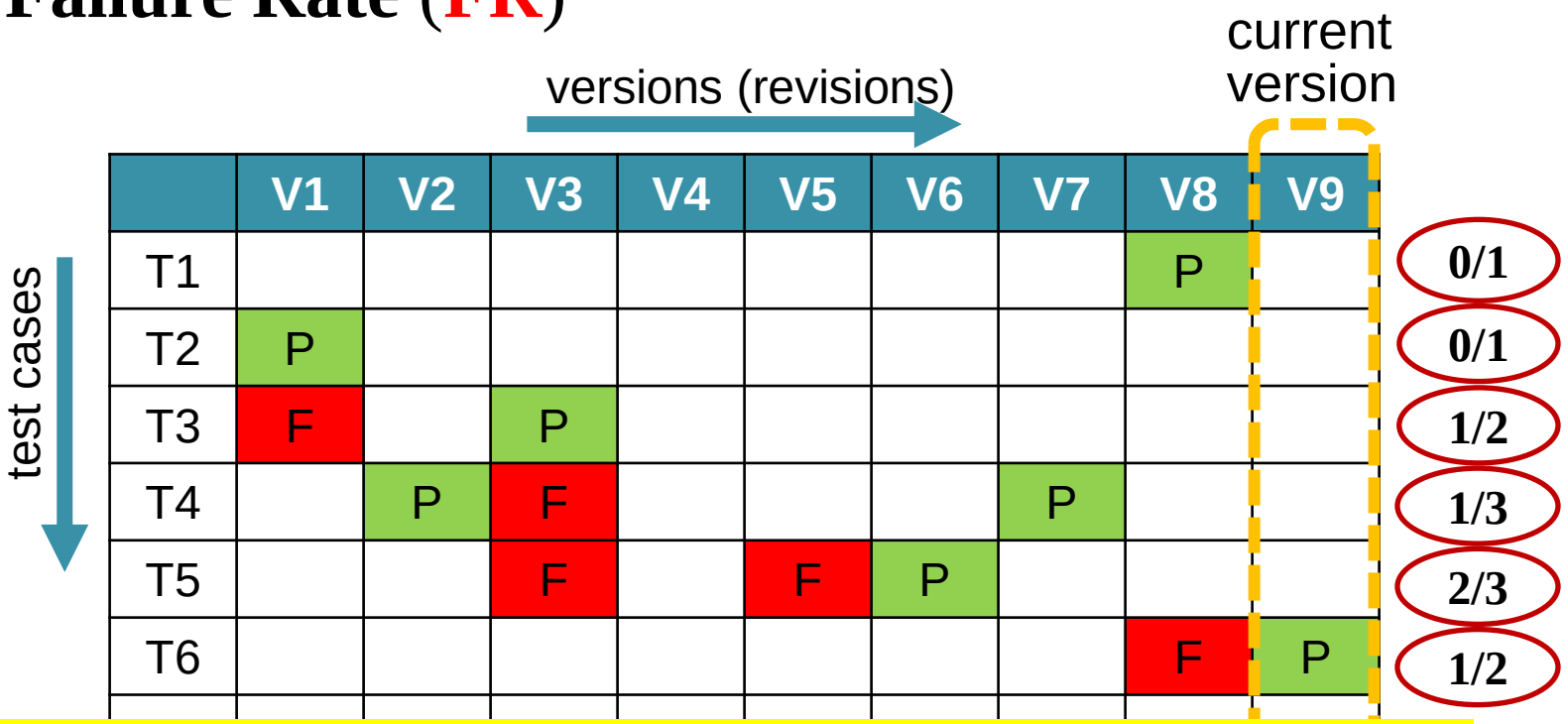
Gap between the Last run version and the Current version (GLC)



A greater GLC value means it's not been tested for more versions. Ignoring such a test case has a higher risk of overlooking regressions.

Priority of a Test Case: Type-II

Failure Rate (FR)



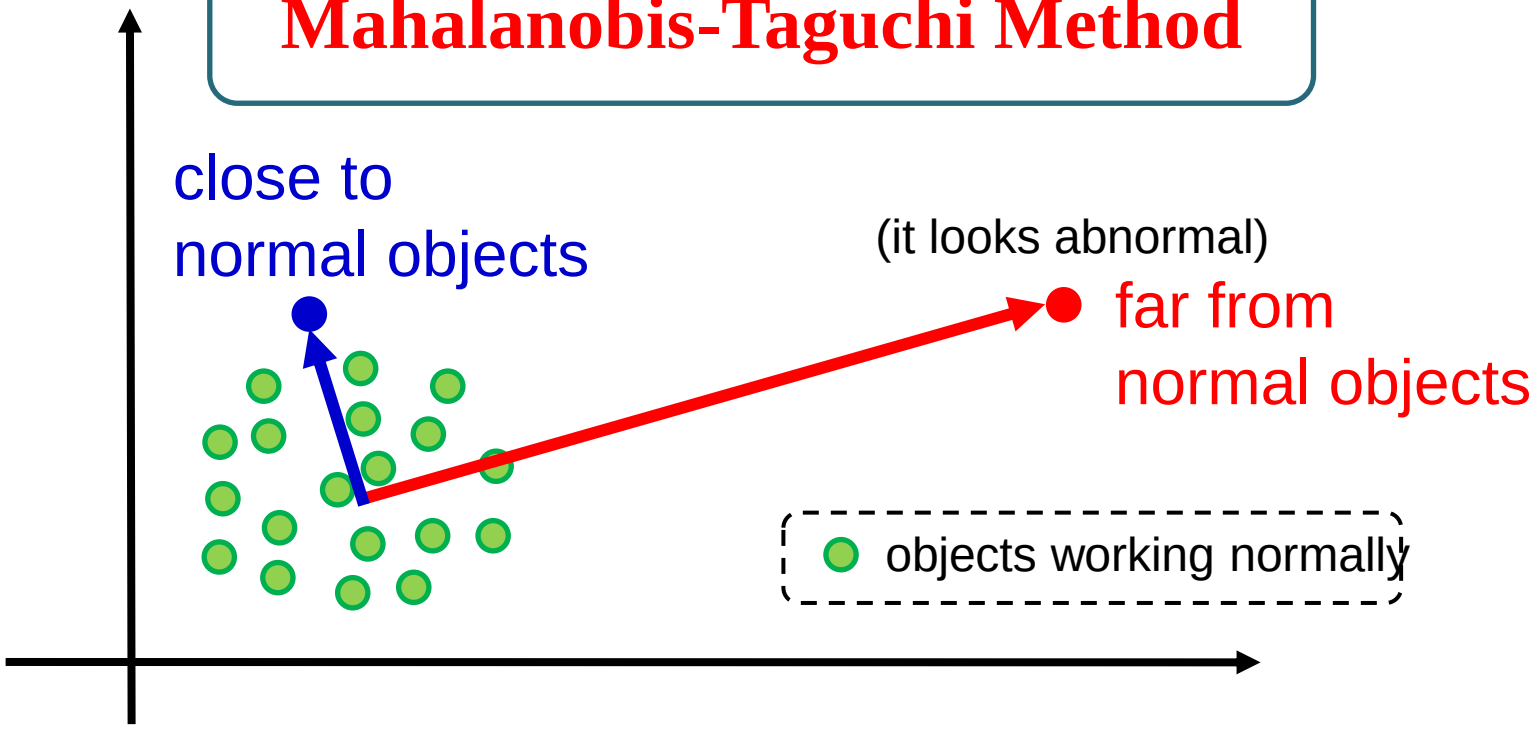
A higher FR value means a better track record for finding a failure in the past.
Such a test case may test a part which is fault-prone and we might expect a higher ability to find a regression.

How should we combine them?

We have to **consistently combine** two **different criteria** for all test cases

To implement such an integration, we adopt the notion of the

Mahalanobis-Taguchi Method



What is Mahalanobis distance?

- A distance **normalized** by the **dispersion** of data: the distance between $\boldsymbol{x}$ and $\boldsymbol{a}$

$$d_M(\boldsymbol{x}, \boldsymbol{a}) = (\boldsymbol{x} - \boldsymbol{a})^T S_A^{-1} (\boldsymbol{x} - \boldsymbol{a})$$

where S_A is the variance-covariance matrix

  - cf. **Euclidean distance**

$$d_E(\boldsymbol{x}, \boldsymbol{a}) = (\boldsymbol{x} - \boldsymbol{a})^T (\boldsymbol{x} - \boldsymbol{a})$$

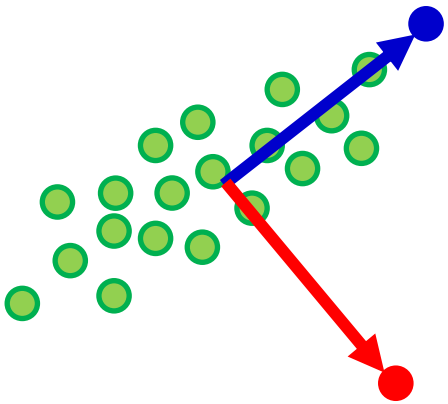
An Intuitive Interpretation

- One-dimensional Mahalanobis distance

$$d_M(x,a) = \frac{(x-a)^2}{\sigma_A^2}$$

It's the **Euclidian distance divided by the variance** of data

- This notion is generalized to the multi-dimensional form



Their **Euclidian distances are the same**, but the **red one is clearly farther** from the center

Mahalanobis distance can capture such a difference

Example: Test Case Evaluation

(P: pass, F: fail, Blank: no run)

	V1	V2	V3	V4	V5	V6	V7	V8	V9	GLC	FR
T1								P		1	0/1
T2	P									8	0/1
T3	F		P							6	1/2
T4		P	F				P			2	1/3
T5			F		F	P				3	2/3

calculating Mahalanobis distance

	GLC	d_{GLC}	FR	d_{FR}	$d_{GLC\&FR}$
T1	1	0.11	0	0.00	0.12
T2	8	7.11	0	0.00	7.81
T3	6	4.00	1/2	4.00	11.42
T4	2	0.44	1/3	1.78	3.03
T5	3	1.00	2/3	7.11	10.67

Outline

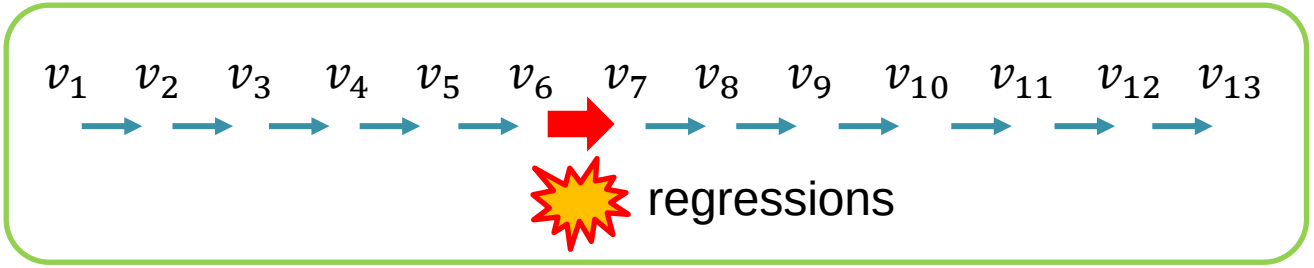
- Background, Motivation & Situation
- Test Case Recommendation
  - Morphological Analysis
  - Test Case Clustering
  - Test Case Prioritization
- **Empirical Study**
- Related Work
- Conclusion & Future Work

Empirical Study: Dataset

- We prepared **300 test cases** for an information system: $t_1, t_2, \cdots, t_{300}$
- The system to be tested has **13 versions**: $v_1, v_2, \cdots, v_{13}$
- All test cases are **written in Japanese** and test engineers manipulate the system according to those test cases

Dataset & Aim

- While there were **regressions**, the original test activity **overlooked** them



  - When the system was upgraded **from v_6 to v_7**, there were regressions; if we reran **more test cases at or later than v_7**, we might have prevented the overlooking

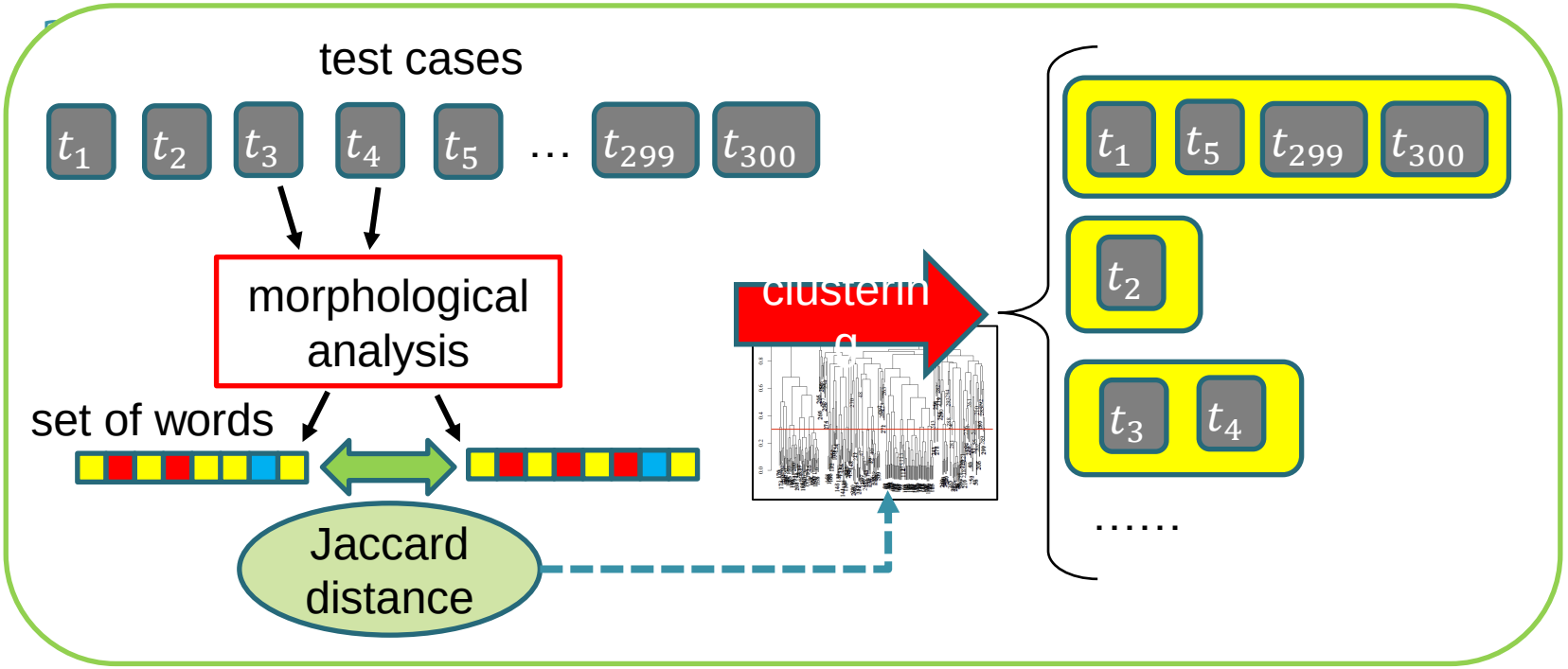
We will examine if the proposed method can recommend appropriate test cases

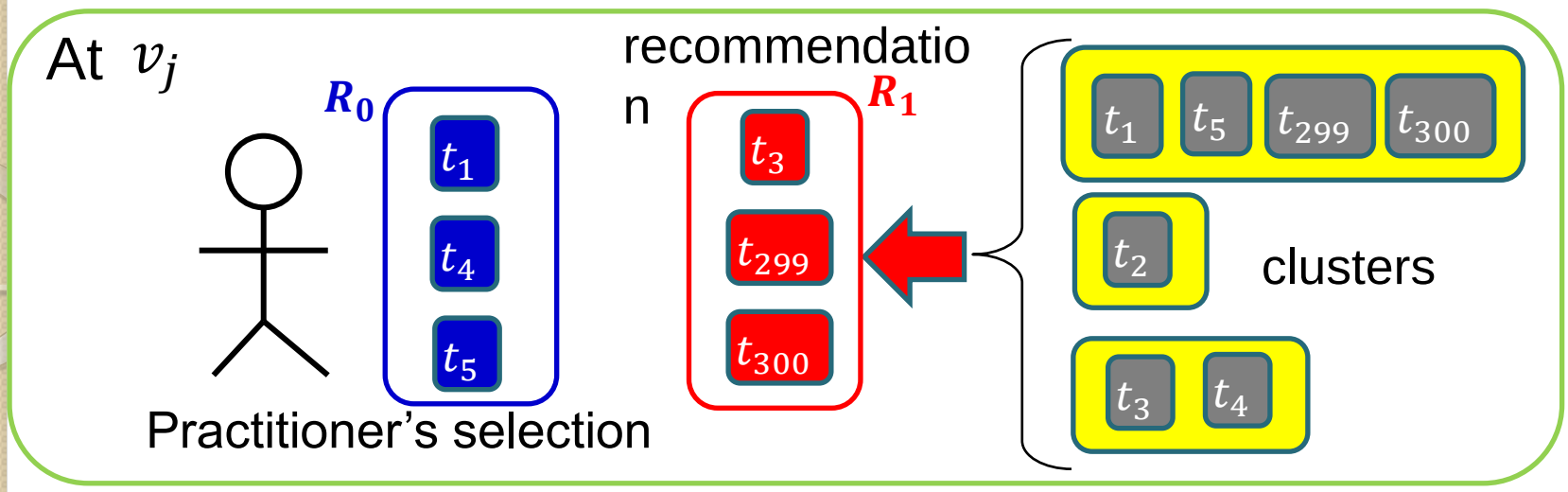
Procedure

1. Perform a morphological analysis on each of the 300 test cases
2. Categorize test cases into clusters
3. Iterate the following for each version v_j:
   a. $R_0 \leftarrow$ test cases selected by practitioners (the original test plan)
   b. $R_1 \leftarrow$ test cases recommended by using R_0 with the clustering results (Step2)
   c. Examine how many test cases in R_1 can detect regressions

Procedure

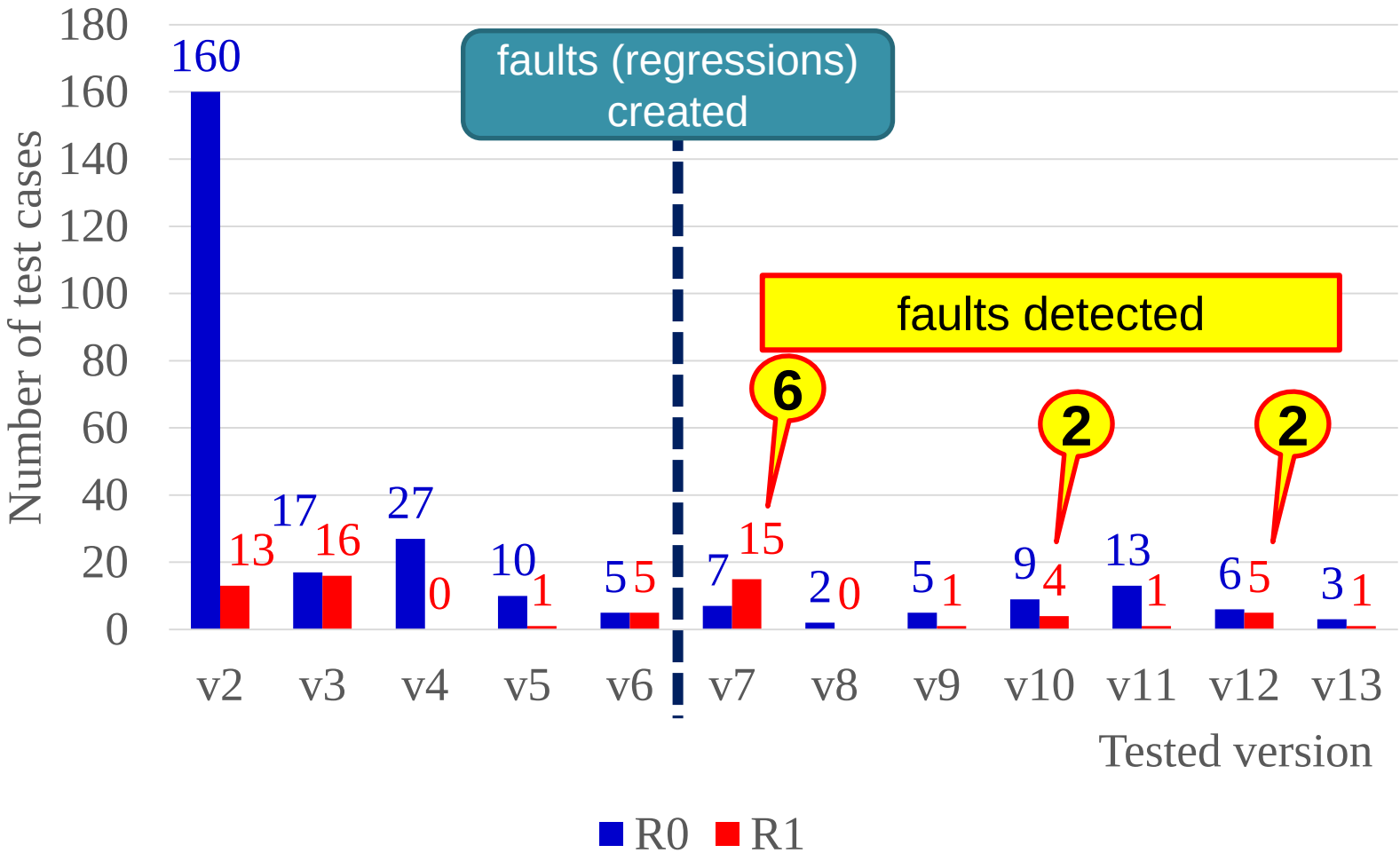
1. Perform a **morphological analysis** on each of the 300 test cases
2. **Categorize** test cases into clusters



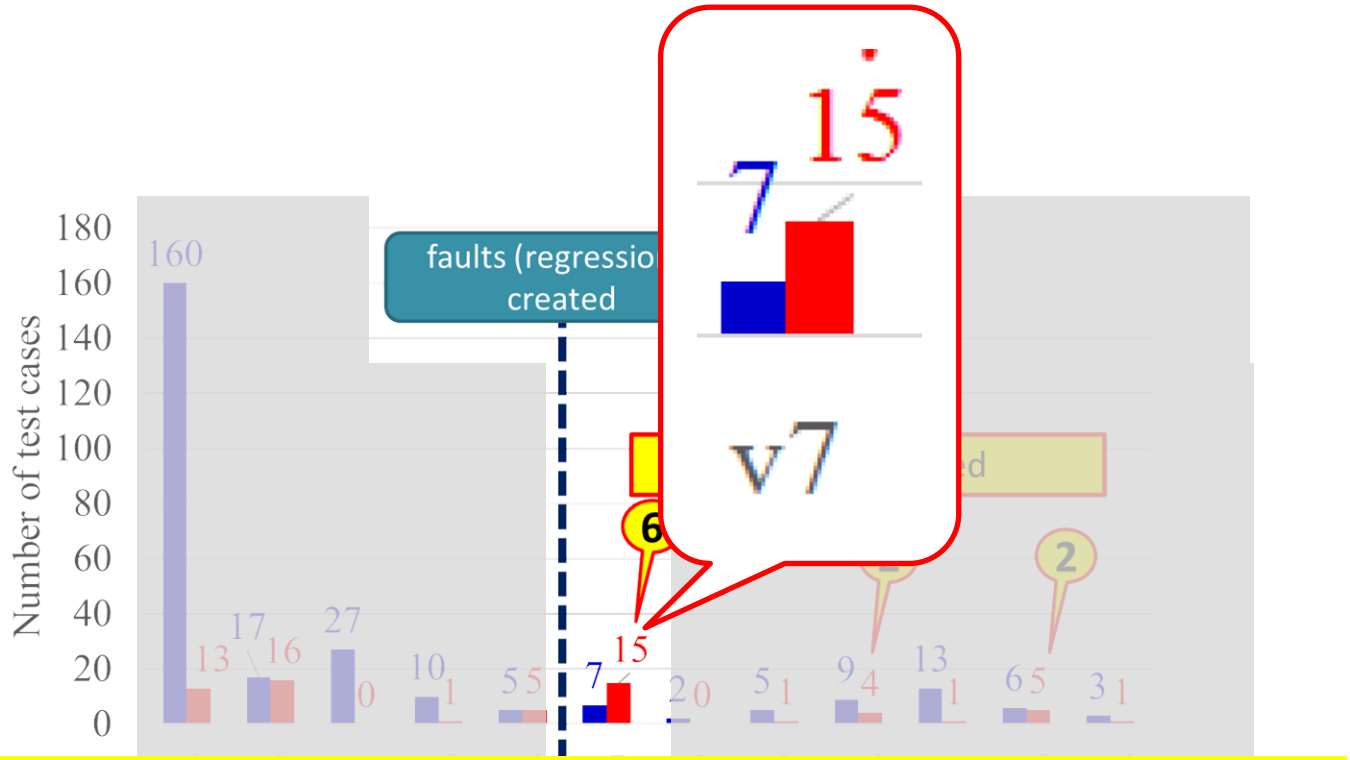


3. Iterate the following for each version v_j:
   a. $\boldsymbol{R_0}$ ← test cases **selected by practitioners** (the original test plan)
   b. $\boldsymbol{R_1}$ ← test cases **recommended** by using R_0 with the clustering results (Step2)
   c. **Examine** how many test cases in $\boldsymbol{R_1}$ **can detect regressions**

Results: Manual Selections(R_0) vs Recommendations(R_1)

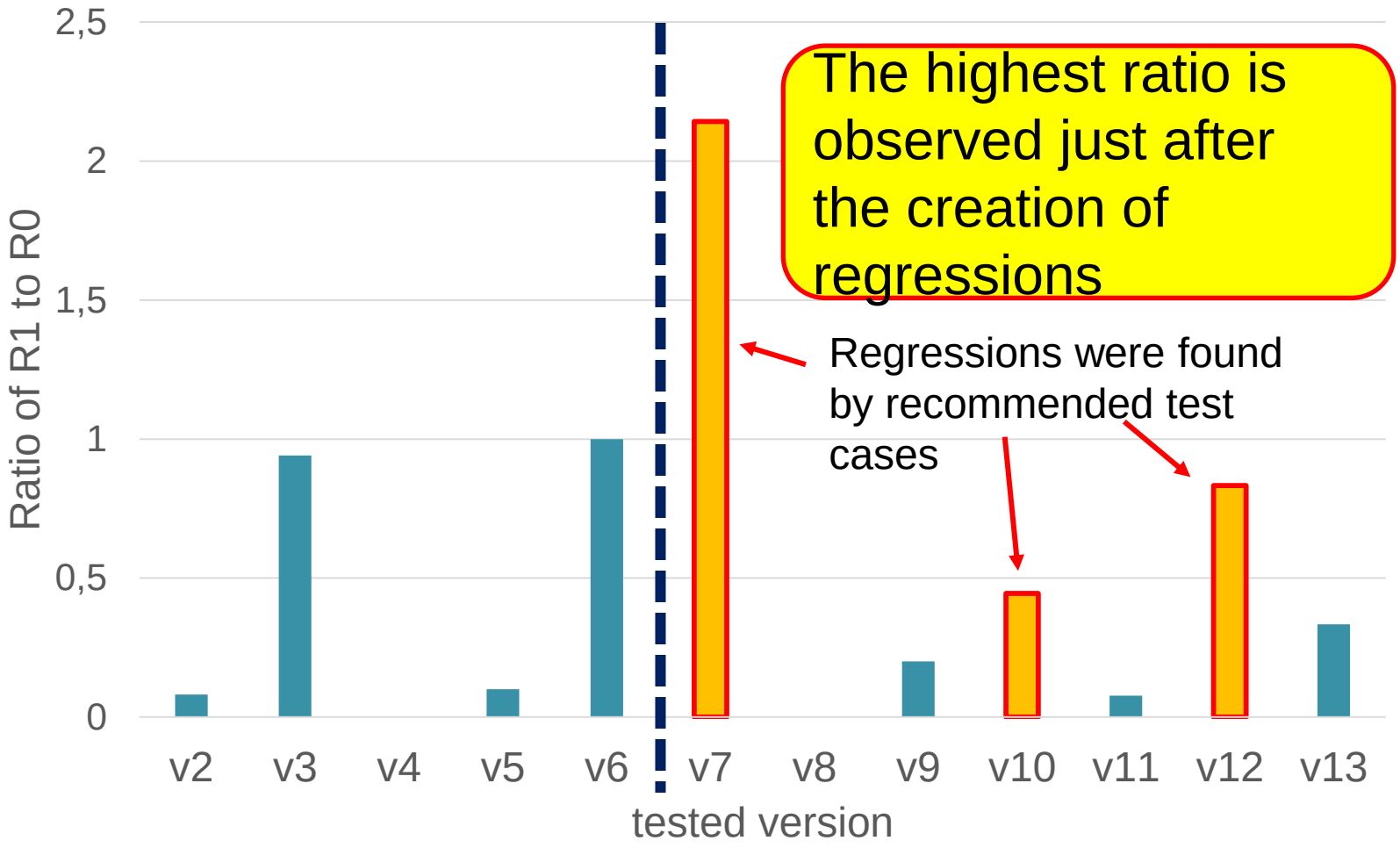


Discussion: Recommendation at v_7 (just after faults were created)



More test cases are recommended than the practitioners' selections; it is obviously a different feature from other versions

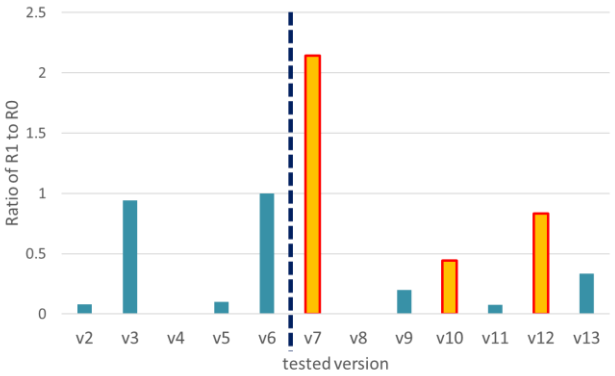
Ratio of Recommendations to Manual Selections: $|R_1| / |R_0|$



What does such a high ratio mean?

- For a set of manually selected test cases, a **higher ratio** shows that there are **more test cases** which are **similar** but **not selected**

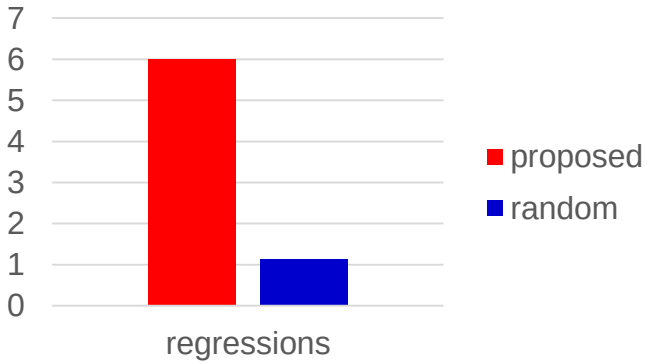
overlooking regressions



- The ratio would be useful in detecting the **insufficiency of a test plan**

Effectiveness of Recommendation

- At v_7, the proposed method recommended **15 test cases**
- If we had also rerun those **recommended** test cases, **6** would have succeeded in finding regressions
- On the other hand, if we had selected 15 test cases **randomly**, the expectation of finding regressions is about **1.1**



About 5-6 times more effective than random selection

Effectiveness of Prioritization

- If **many test cases** are recommended, we may need to **prioritize** them because of cost or time for testing
- We can do this by using the **Mahalanobis-Taguch(MT) method**

rank	detecting defect
1	Yes
2	No
3	Yes
4	No
5	Yes
6	Yes
7	Yes
8	Yes

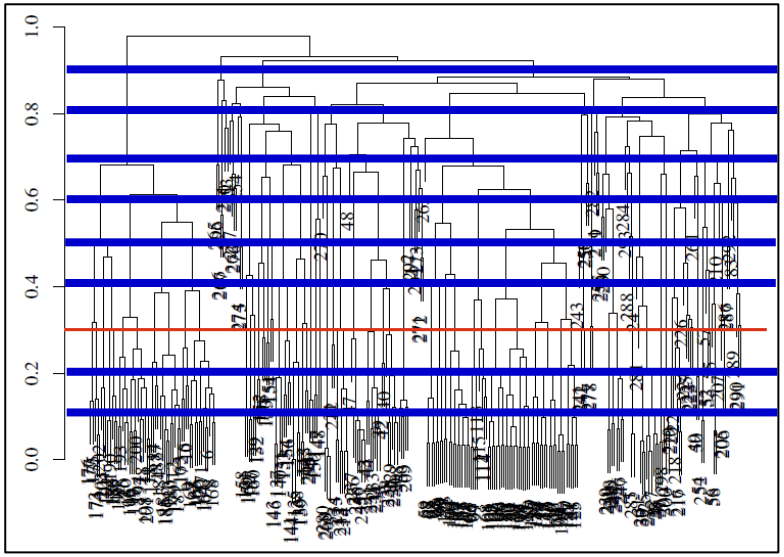
rank	detecting defect
9	No
10	No
11	No
12	No
13	No
14	No
15	No

All defects are **detected** by the test cases **with higher priorities**

MT method works well

Cut Level when Clustering

- While we set **0.3** as the cut level based on our experience, it has room for discussion

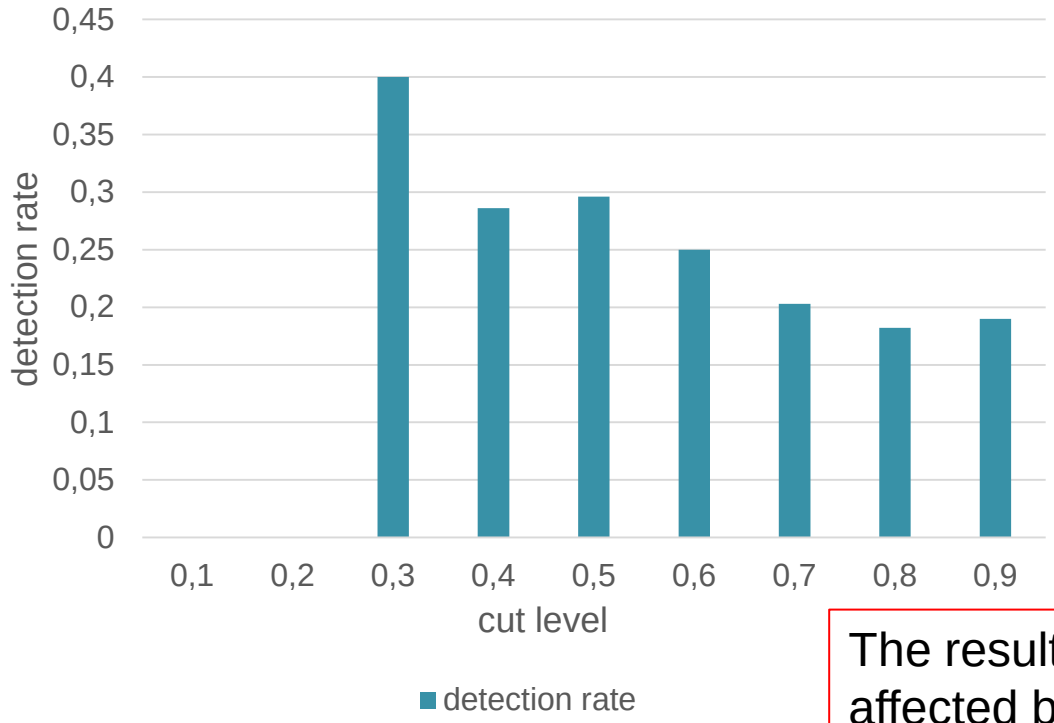


- We performed additional experiments at v_7 using **other cut levels** (**0.1—0.9**)

Defect Detection Rate vs Cut Level

- detection rate

$$= \frac{\text{number of test cases detecting defects}}{\text{number of recommended test cases}}$$



A model using higher cut level recommends more test cases, but includes more false-positive ones too

The results would be highly affected by how to describe test cases, so further analysis is our future work

Threats to Validity (1/2)

- Since our study covers a part of regression testing for a **single product**, we **cannot say** our results are **generalizable**

- However, we believe that this study contributes to **stirring up the utilization of the morphological analysis** in the regression testing world

Threats to Validity (2/2)

- There might be a **large variety of vocabulary** among test cases because they are written by different engineers, in natural language (Japanese) : **different engineers might use different words to describe the same thing**

- It would be better to perform **data preprocessing to link a word with another word** which has the same meaning; a further analysis of vocabulary is our future work

Outline

- Background, Motivation & Situation
- Test Case Recommendation
  - Morphological Analysis
  - Test Case Clustering
  - Test Case Prioritization
- Empirical Study
- **Related Work**
- Conclusion & Future Work

Related Work (1/3)

- **Code analysis-based** test case prioritization
  - Jeffrey et al.[3] and Mirarab et al.[4] proposed ways of prioritizing test cases through the **program slicing** analysis or the **code coverage** analysis
- **Test history-based** test case prioritization
  - Kim et at.[5] prioritized test cases by using the notion of the **exponentially smoothed moving average** on the test history
  - Aman et al.[6],[7] formulated a test case prioritization as a **0-1 programming problem**

Related Work (2/3)

- **Clustering-based** test case prioritization
  - Sherrif et al.[8] classified test cases through an analysis of **source code change history**
  - Carlson et al.[9] and Leon et.[10] categorized test cases by using the **code coverage** data or the **execution profiles**
  - Arafeen et al.[11] focused on the **requirement specification** and categorized related test cases

Related Work (3/3)

- **Content-based** test case prioritization
  - Ledru et al.[12] used a **string distance** (**character level** distance) and selected the **farthest test cases** from the set of already-run test cases
  - Thomas et al.[13] leveraged the **topic modeling method**: they extracted topics from test cases and quantified the membership degrees of each test case to those topics
- While our approach has a similar aspect to [13], we tried to propose **another, easier method** of test case clustering by focusing on words

Outline

- Background, Motivation & Situation
- Test Case Recommendation
  - Morphological Analysis
  - Test Case Clustering
  - Test Case Prioritization
- Empirical Study
- Related Work
- **Conclusion & Future Work**

Conclusion & Future Work

- **Conclusion**
  - A **morphological analysis** method has been applied in **test case recommendation**
  - Once a test engineer decides to rerun a test case t_0, the proposed method **recommends other test cases whose contents are similar** to t_0
  - An empirical study showed the proposed method is useful in **preventing the overlooking of regressions**
- **Future Work**
  - we plan to perform a further analysis on **features of test cases** from the perspective of **natural language analysis**

Answers to the Survey

- How did you get in contact with the industrial partner?
  - ✓**After a discussion at a workshop, I approached the industrial partner about the collaboration**
- How did you collaborate with the industrial partner?
  - ✓**The industrial partner gave me real data (confidential parts were masked), and I analyzed the data and discussed the results**
- How long have you collaborated with the industrial partner?
  - ✓**5 years**
- What challenges did you experience when collaborating with the industrial partner?
  - ✓**to prove how our research results would successfully work in the field**