Automated Assessment of Programming Assignments

Vreda Pieterse

University of Pretoria

Abstract

We discuss our experience when using our own tool for the automatic assessment of programming assignments of a large class. Our observations corroborates similar experiences reported in the literature.

We identified success factors and concerns related to automatic assessment through the analysis of our experiences in relation with literature on this topic. Our reflection on this experience informs our proposed actions to improve future usage of our system.

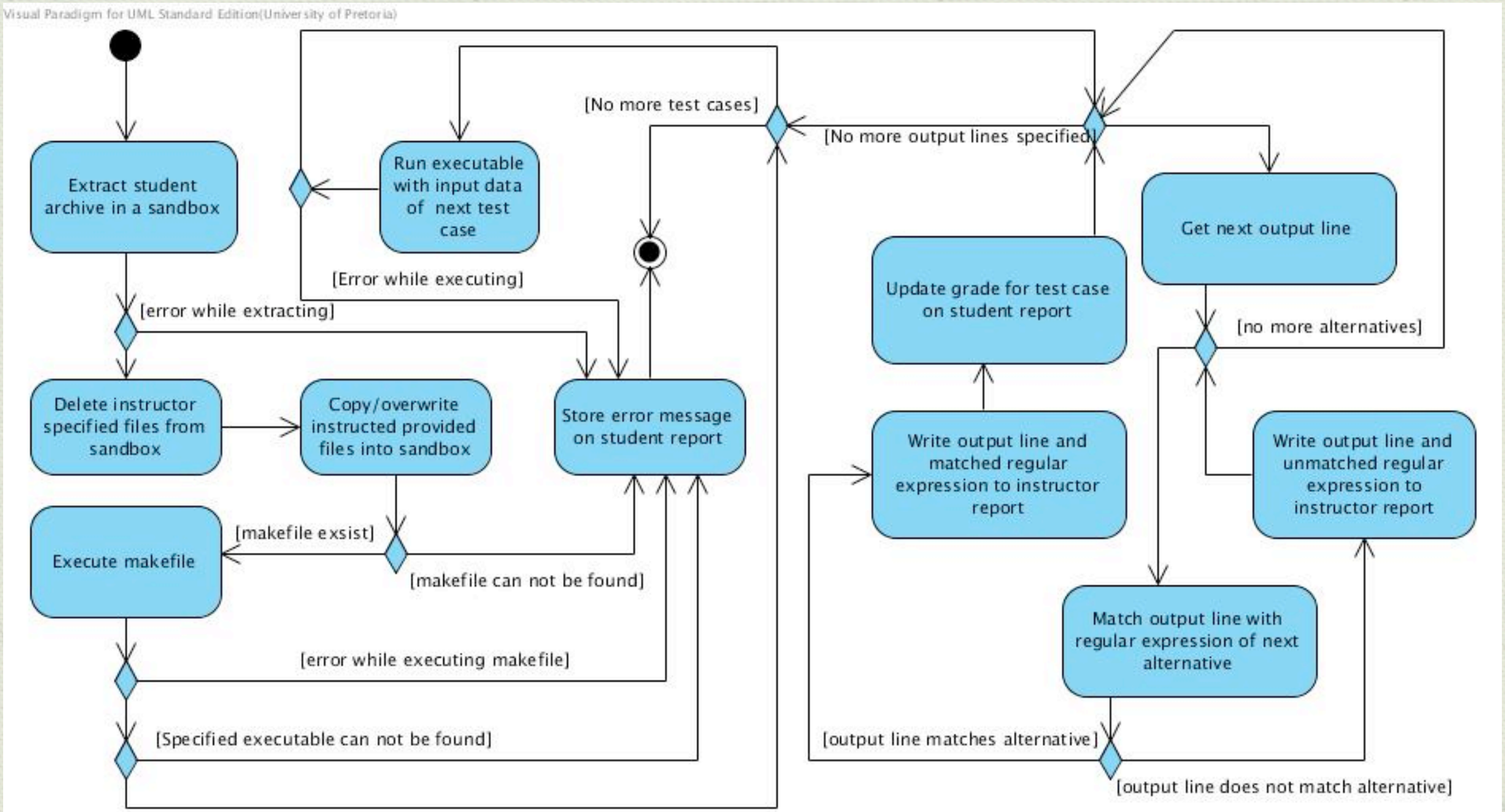
Outline of this presentation

- The context in which we applied automatic assessment.
- How our system works.
- Some issues that should be considered when automatic assessment of programming assignments is applied.
- Factors that can contribute to more successful application of automatic assessment of programming assignments.

Context

- Country: South Africa
- Institution: University of Pretoria
- Course: Introduction to programming
- Programming language: C++
- Level: First year graduate course (first semester)
- Enrollment: between 450 - 500

Our system


Visual Paradigm for UML Standard Edition(University of Pretoria)
Extract student archive in a sandbox
[error while extracting]
Delete instructor specified files from sandbox
Copy/overwrite instructed provided files into sandbox
Store error message on student report
[makefile exsist]
Execute makefile
[makefile can not be found]
[error while executing makefile]
[Specified executable can not be found]
[Error while executing]
Run executable with input data of next test case
[No more test cases]
[No more output lines specified]
Update grade for test case on student report
Write output line and matched regular expression to instructor report
Get next output line
[no more alternatives]
Write output line and unmatched regular expression to instructor report
Match output line with regular expression of next alternative
[output line matches alternative]
[output line does not match alternative]

A portion of an example of an assessment specification

```
<line>
  <alt><exact mark='5'>The ASCII value of m is 109</exact></alt>
  <alt><regexp mark='4'>.*The\s+ASCII\s+value\s+of\s+m\s+is\s+109</regexp></alt>
  <alt><regexp mark='3'>.*(ASCII|ascii)*.m*.109</regexp></alt>
  <alt><regexp mark='2'>.*The\s+ASCII\s+value\s+of\s+m\s+is\s+110</regexp></alt>
  <alt><regexp mark='2'>.*The\s+ASCII\s+value\s+of\s+m\s+is\s+108</regexp></alt>
</line>
```

Technical issues

- Security

  Student programs need to executed in a sandbox with extreme limited privileges to avoid accidental or malicious damage of the system.

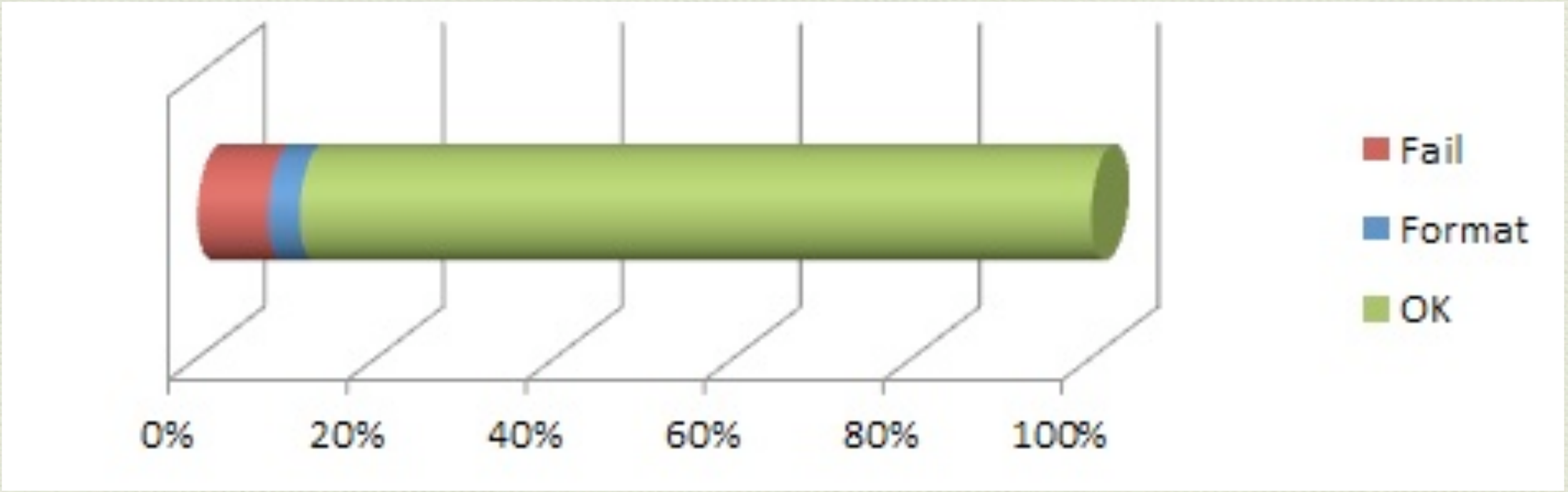
- Limited capability

  There are various ways is which the system can fail to assess fairly

  - Failing before execution
  - Failing as a result of formatting or misalignment

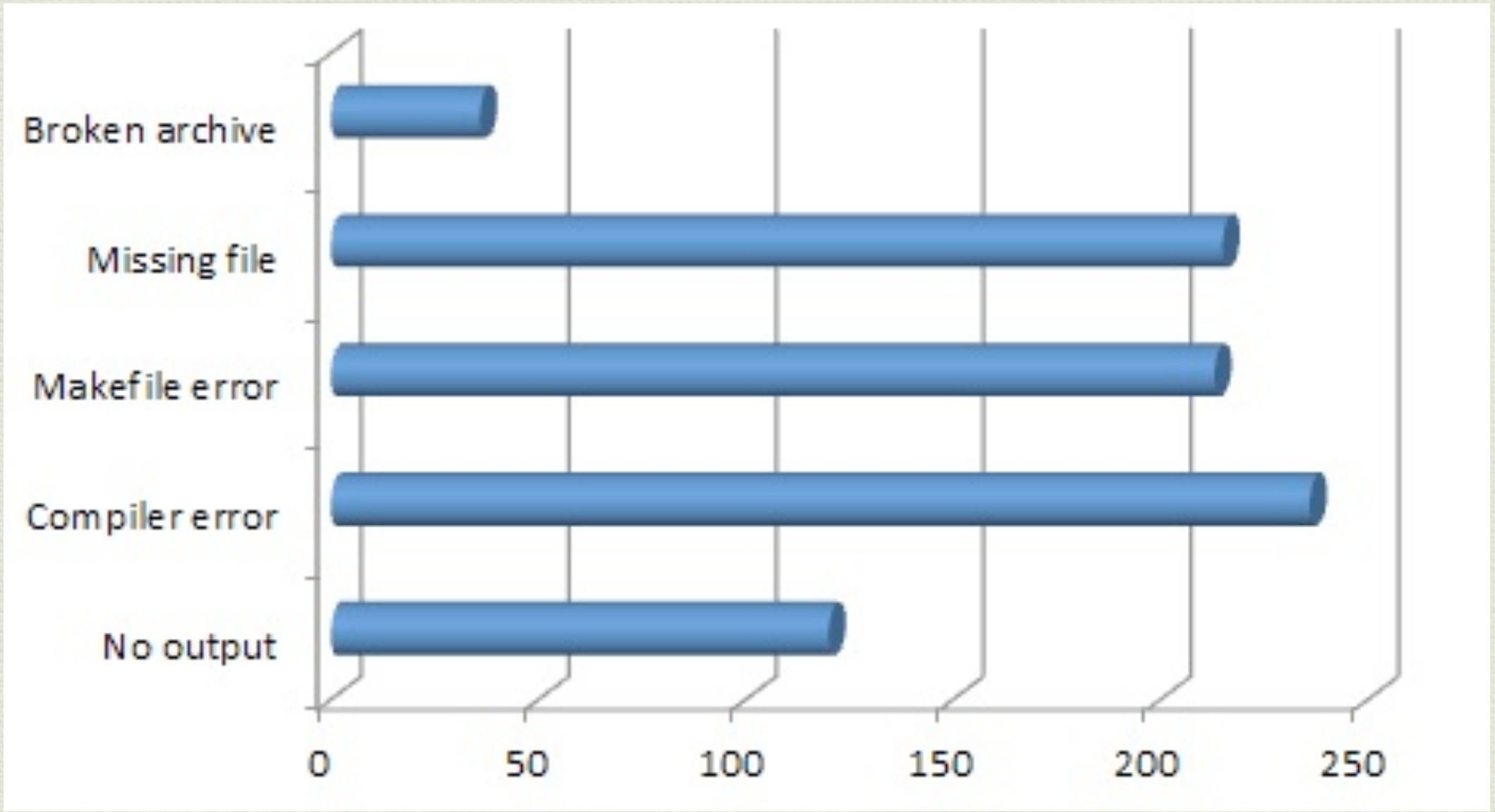
Unfair marking owing to automation

Out of 11233 uploads over the semester, 820 failed before they could be assessed, 365 were awarded zero marks as a consequence of ill-formatting or misalignment.



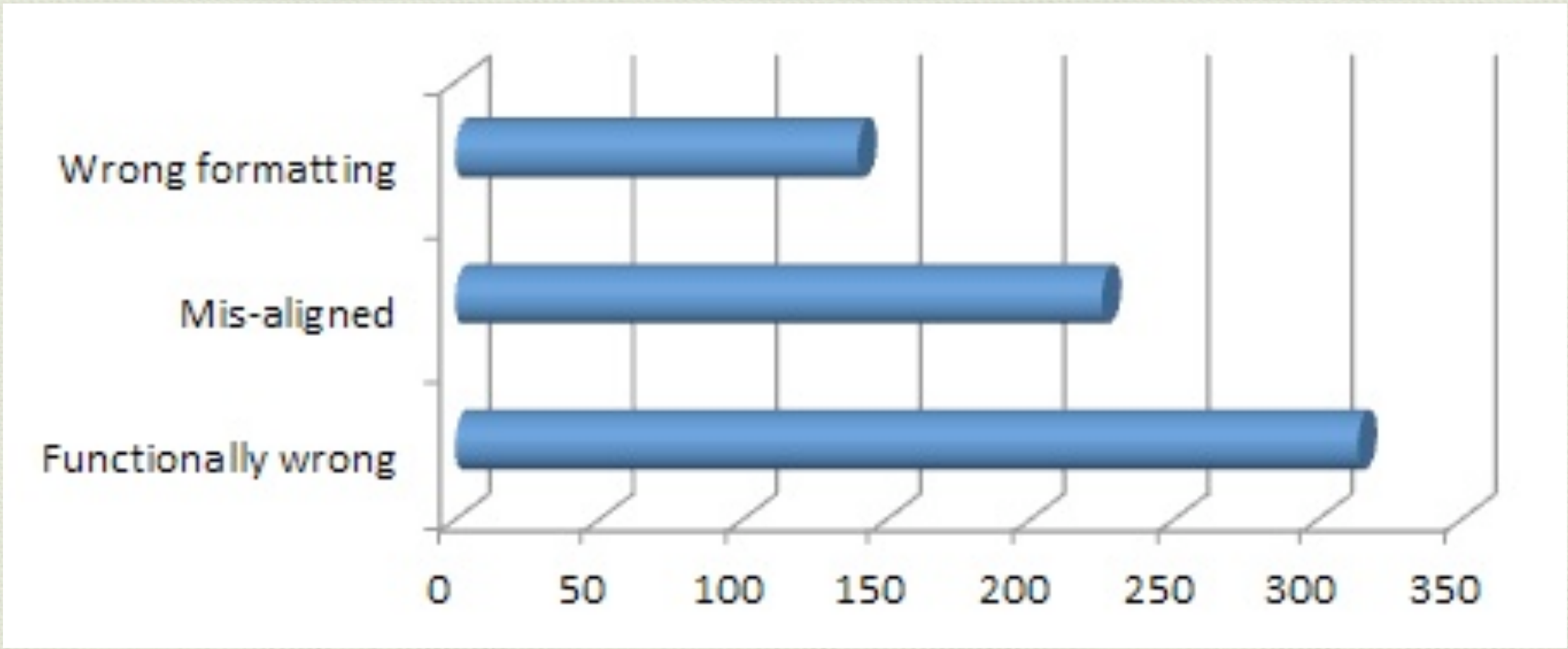
Failing before assessment

n = 820



Failing when assessed

n = 679



Example of misalignment

Expected	Actual

`Enter a value:  5`
`Enter a value:  23`
`5 + 23 = 28`
	`Enter a value:`
	`5`
	`Enter a value:`
	`23`
	`5 + 23 = 28`

Educational Issues

- More challenging and increased effort for the lecturers.

  The formulation of assignments, the design of test cases and the formulation of assessment instructions is time consuming and challenging.

- Higher skills expected from students.

  Students have to follow instructions with greater care and have to acquire additional skills to be able to use gain maximum benefit from the system.

- Suppressing creativity.

  Students are more inclined to follow instructions without deepening their understanding of the work they are doing.

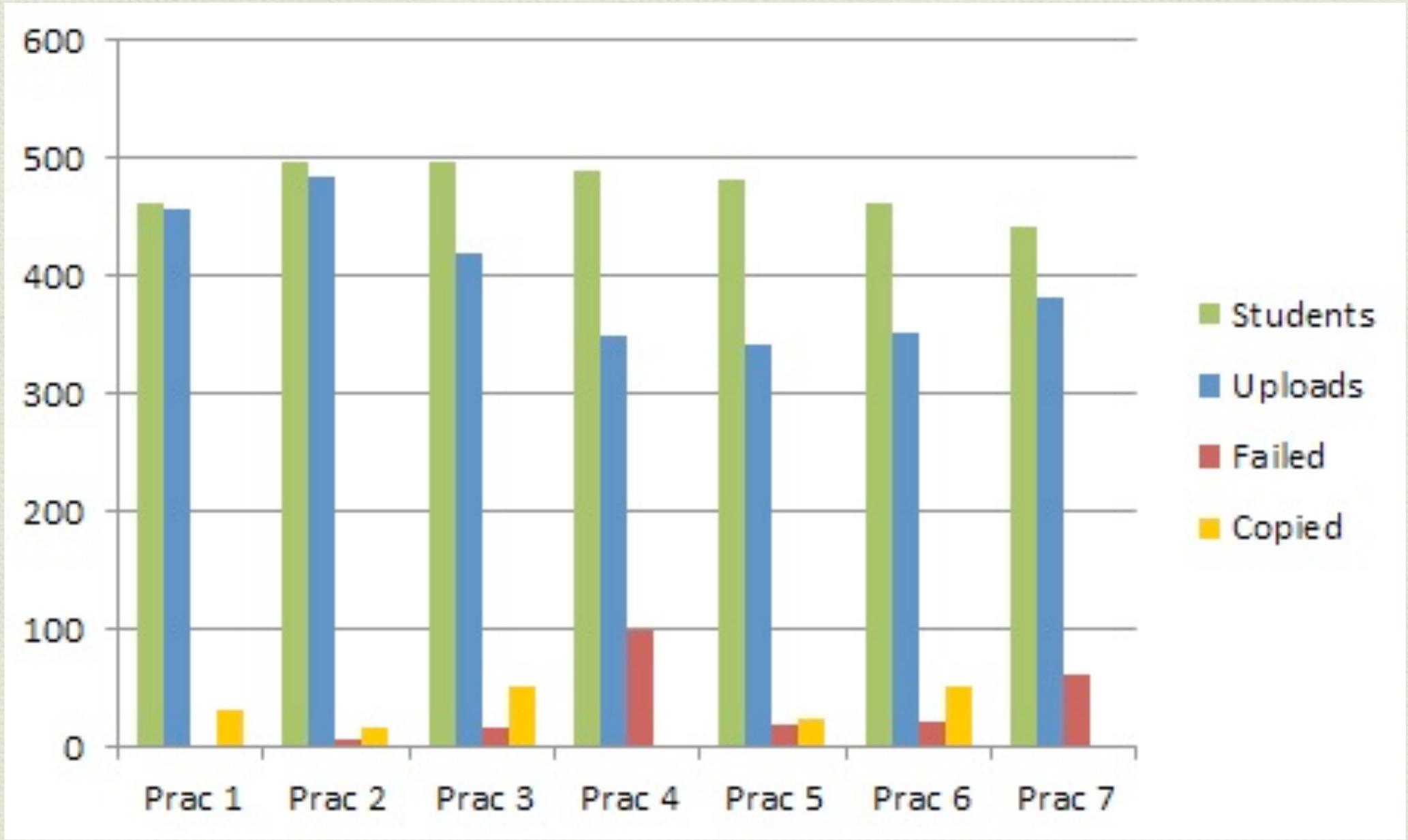
- Inflaming copying.

  Owing to code not being scrutinized by people, coupled with more precise instructions, it is easier to get away with copying.

Copying

My program was exactly the same as his. I compared it and could not see the difference. Yet my program failed and his program got full marks. I think I deserve these marks. Fitchfork is wrong for not giving me my marks.

Relation between number of students, uploads, failures and copied work.



Success factors

The successful application of automatic assessment of assignments is dependent on the following:

- The quality of the assignments and clear formulation of the assignments.
- Well designed test cases coupled with proper training of the students to understand testing.
- The quality and accuracy of automated feedback.
- Continued human involvement.

Thank you