Feedback generation in programming tutors
The paper

- A systematic literature review of automated feedback generation for programming exercises
- Discussed about 100 different tutors
- Excludes just automated assessment tools

Feedback

**Nature**
What kind of feedback

**Generation techniques**
How can we generate

**Adaption**
How can teachers adapt the tutor
Nature of the feedback

Knowledge about task constraints
Knowledge about concepts
Knowledge about mistakes
Knowledge about how to proceed
Knowledge about meta-cognition
Knowledge about task constraints

- Hints on task requirements
  - Hints about something task specific a student just did wrong
- Hints on task-processing rules
  - General information on how to proceed with the specific exercise
Knowledge about concepts

- Explanations on subject matter
  - being able to explain subject matter
- Examples on illustrating concepts
  - being able to give examples of subject matter
Knowledge about mistakes

- Compiler errors
- Test failures
- Solution errors
- Style issues
- Performance issues
Knowledge about how to proceed

- Bug related hints for error corrections
  - improvements on simple errors
- Hints for task processing steps
  - can help a student solve exercise step by step (guided programming)
- Hints for improvements
  - How to make the program better
Knowledge about meta-cognitions

- Knowledge how the student learns.
Data discussion

- Mostly focused only identifying mistakes
Generation techniques

General ITS techniques
Domain-specific techniques for programming
Combining techniques
General ITS techniques

- Model tracing
  - Trace the steps a student takes and compare these to model solutions and buggy solutions
- Constraint based modeling
  - Having constraints you test the student solution against
- Data analysis
  - Using large sets of historical student data to generate hints
Domain-specific techniques for programming

- Dynamic code analysis using automated testing
  - Tests
- Basic static analysis
  - Tutor specific static analysis
- Program transformations
  - Transform code into a Semantic preserving variation or abstract language for comparison
- Intention-based diagnosis
- External tools
  - External style and type checkers
Other program domain specific techniques

- ProPL tutor uses natural language processing to engage in a dialogue
- The COALA system uses fuzzy logic to analyse similarity to a model solution
How often are they used?

Fig. 15. Cumulative number of tools with technique over the years, based on the earliest paper on a tool from our data set. The legend has the same order as the endings of the lines in the chart.
Adaption of the exercises
Adaption of the exercises

- Using solution templates
  - Fill in the gaps type of tutor
- Using model solutions
  - Solutions are compared to a model solution
- Using test data
  - Solutions are tested to have the expected outcome with certain data
- Using error data
  - Solutions are compared to a buggy solution
How often are they used?

- Model solutions and test data used most
- Quite a lot don’t use any
Conclusion

- ITS systems have been around for quite some time
- Most Types of tutors focus
  - identifying mistakes
  - or where they went wrong
- There was experimentation with different types of tutors
Questions?