Automatisches Feedback zu Programmieraufgaben

Niels Pinkwart[1]

Eines der wiederkehrenden Probleme bei der Gestaltung von Online-Lernumgebungen für die Programmierausbildung ist es, Nutzern sinnvolles Feedback zu ihren Lösungsversuchen bei Programmieraufgaben zu geben. Dies ist notwendig: Eigene Aktivität ist wichtig für den Lernerfolg [Ko15], und gerade in selbstgesteuerten Lernszenarien sind Rückmeldungen für die Nutzer von immenser Bedeutung, um ihren Lernfortschritt einschätzen und ihre weiteren Schritte planen zu können [BW95]. Problematisch ist die automatisierte Berechnung von Feedback auf Lernerlösungen u.a. deshalb, weil Programmieraufgaben oft viele verschiedene Lösungsmöglichkeiten (unterschiedlicher Qualität) haben. Es besteht somit die Herausforderung, ein ggf. fehlerhaftes vom Nutzer geschriebenes Programm nicht nur auf Korrektheit zu prüfen, sondern auch darüber hinaus lernförderliche Rückmeldungen zu generieren, welche die vom Nutzer vermutlich favorisierte Lösungsstrategie ebenso berücksichtigen wie mögliche Fehler in der aktuellen Lernerlösung. Nachfolgend werden vier Möglichkeiten zur Generierung von Feedback vorgestellt, die auf unterschiedlichen Prinzipien beruhen. Zu jedem Ansatz werden dabei empirische Erkenntnisse diskutiert.

Der erste Ansatz zielt dabei nicht primär auf Online-Lernszenarien, sondern auf „klassische“ universitäre Programmierkurse. Ziel des GATE-Systems [SOP11] ist es, Studierende, Tutoren und Dozenten beim Übungsbetrieb zu unterstützen. Das GATE-System kombiniert aufgabenunabhängige Syntaxtests und aufgabenspezifische Funktionstests (als Feedback für Studierende) mit Plagiatserkennungsverfahren (als Feedback für Tutoren). Ergebnisse zeigen, dass sowohl Tutoren als auch Studierende Vorteile durch den Einsatz des Systems hatten und dass mit dem Konzept studentischer Tests vor Abgabeschluss durchaus Qualitätssteigerungen erreicht werden konnten.

Der zweite vorgestellte Ansatz ist in der Online-Lehre verwendbar, wurde aber auch im Rahmen von Programmierkursen an Universitäten eingesetzt und evaluiert [LP12]. Hier ist dem System (INCOM) für jede Aufgabe eine feste Menge von Lösungsstrategien bekannt (z.B. unterschiedliche rekursive und iterative Verfahren). Für jede dieser abstrakten Strategien lassen sich mittels bekannter allgemeiner Regeln (z.B. arithmetische Gesetze, Freiheiten bei Reihenfolgen von Kommandos) eine Vielzahl von korrekten Lösungsmöglichkeiten generieren. Eine Lernerlösung wird im INCOM-System mittels einer KI-gestützten Heuristik über eine gewichtete Constraint-Logik mit allen bekannten Lösungsstrategien und deren Varianten verglichen, um so die Strategie zu bestimmen, die der Nutzer vermutlich verfolgte. Auf Basis der Unterschiede zwischen der ähnlichsten Musterlösung und der (fehlerhaften) Lernerlösung kann dann Feedback gegeben werden.

[1] Institut für Informatik, Humboldt-Universität zu Berlin, Unter den Linden 6, 10099 Berlin, niels.pinkwart@hu-berlin.de

Der dritte vorgestellte Ansatz setzt kollaborative Filteralgorithmen und Peer Reviews im E-Learning ein und wurde zur Klausurvorbereitung in Informatikkursen erfolgreich eingesetzt [LP09]. Ergebnisse von kontrollierten Laborstudien mit dem CiTUC-System belegen die grundsätzliche Eignung des Verfahrens im Sinne einer hohen Korrelation von Peer-Review-Ergebnissen mit Expertenbewertungen von Lernerlösungen. Probleme bei der praktischen Nutzung des CiTUC-Systems ergaben sich in der Motivation der Studierenden zur kontinuierlichen und sinnvollen Nutzung des Systems bei freiwilliger Teilnahme – vor der Klausur wurde das System jedoch intensiv verwendet und sowohl durch den Übungsleiter als auch durch die Studierenden als hilfreich angesehen.

Im vierten vorgestellten Ansatz stehen Visualisierungen von Lösungsräumen im Mittelpunkt. Hier wird auf die formale Analyse der Korrektheit von Lernerlösungen (insbesondere der semantischen Korrektheit) komplett verzichtet. Stattdessen werden Lernerlösungen und deren Zwischenschritte sowie verschiedene Musterlösungen über Ähnlichkeitsmetriken auf Basis von Codestruktur und Stichworten miteinander verglichen. Die resultierenden Ähnlichkeiten werden verwendet, um den Raum aller Lösungen visuell zu strukturieren und Nutzern so Orientierungshilfen zu geben – z.B. dahingehend, wie nah sie an Musterlösungen sind oder ob ihre eigene Lösung ein „Einzelgänger“ ist. Geeignete Feedbackmechanismen in diesem Szenario sind z.B. in [Gr14] beschrieben.

Literaturverzeichnis

[BW95] Butler, D.; Winne, P.: Feedback and Self-Regulated Learning: A Theoretical Synthesis. Review of Educational Research, 65 (3), 1995; S. 245-281.

[Gr14] Gross, S. et. al.: Example-based feedback provision using structured solution spaces. International Journal of Learning Technology, 9(3), 2014; S. 248-280.

[Ko15] Koedinger, K. et. al.: Learning is not a Spectator Sport. In Proceedings of the Second ACM Conference on Learning @ Scale. ACM, New York, 2015; S. 111-120.

[LP09] Loll, F.; Pinkwart, N.: CITUC: Automatisierte Lösungsbewertung im E-Learning durch kollaboratives Filtern. In Tagungsband 2 der 9. Internationalen Tagung Wirtschaftsinformatik. Wien, Österreichische Computer Gesellschaft, 2009; S. 411-420.

[LP12] Le, N. T.; Pinkwart, N.: Can Soft Computing Techniques Enhance the Error Diagnosis Accuracy for Intelligent Tutors? In Proceedings of the 11th International Conference on Intelligent Tutoring Systems. Berlin, Springer, 2012; S. 320-329.

[SOP11] Strickroth, S.; Olivier, H.; Pinkwart, N.: Das GATE-System: Qualitätssteigerung durch Selbsttests für Studenten bei der Onlineabgabe von Übungsaufgaben? In GI Lecture Notes in Informatics (P-188) – Tagungsband der 9. e-Learning Fachtagung Informatik (DeLFI). Bonn, GI, 2011; S. 115-126.