LEARNING LETTERS

2024
Volume 4
Article 28
https://doi.org/10.59453/ll.v4.28

If cheating is optimisation then assessment must not be pure: Effect tracking and assessment

William Billingsley

University of New England, Australia

The emergence of generative AI has broadened the question of how to ensure academic integrity. Where, in the past, many tools sought to counter specific threats to integrity (for example, detecting plagiarism or outsourcing), we now need to consider integrity more fundamentally: What work can we be assured the student did? One way of viewing cheating is from the perspective of optimisation: producing a result without performing the work. This perspective creates an interesting parallel with computer programming. A programmer often wants their compiler to optimise their program, altering its instructions to make it run more efficiently, without affecting its observable behaviour. Some characteristics of programs make this easier or harder to do. For example, a function is said to be "pure" if it always produces the same outputs for the same inputs and has no other effects. If the result is known, then performing the function can be invisibly replaced by inserting its result. If, however, a function has an observable effect, then the work of performing the function cannot be optimised away. This paper explores academic integrity assurance through the lens of "effect types". In functional languages, functions are often written in such a manner that the types of effects they have are explicit and tracked by the compiler. The paper likens this to tracking the types of academic integrity measures that are present in each assessment, how they compose across a course, and therefore what observable effects are present to assure each learning outcome.

Keywords: academic integrity, constructive alignment, effect tracking, generative AI

Corresponding author: William Billingsley, wbilling@une.edu.au

Recommended citation: Billingsley, W. (2024). If cheating is optimisation then assessment must not be pure: Effect tracking and assessment. *Learning Letters*, 4, Article 28. https://doi.org/10.59453/ll.v4.28

Introduction

In the last two years, generative artificial intelligence (GenAI) has prompted widespread reflection on higher education assessment. GenAI has been both a surprise and a long time coming. Historically, artificial intelligence (AI) has been interlinked with education and assessment for fifty years, as many of the early AI pioneers were also interested in human cognition (Doroudi, 2022). Nonetheless, the capabilities and ease of access of tools such as ChatGPT were a signal moment for academics and regulators to consider its implications, especially regarding academic integrity (Cotton et al., 2023).

The impact of GenAI on assessment goes beyond the simple question of cheating. As workplaces adopt AI, the skills that students need to be taught are changing (Markauskaite et al., 2022). As students and educators interact more frequently with AI, their expectations on how assessment is delivered also change (Smolensky et al., 2023). Universities are adapting how they produce their material (Diwan et al., 2023), how they deliver feedback (Dai et al., 2023), and how they offer monitoring and support services throughout students' studies. Academics are also predicting how social GenAI in

education may be improved – for example, GenAI conversations are currently isolated from your social context and each conversation has only a limited sense of your past conversations (Sharples, 2023). So, the questions of what universities assess, how, why, and when, are in flux. The need to assure that a student's qualification accurately represents skills they have learnt, however, remains, even as the landscape of skills and assessment being assured becomes more complex.

The question of ensuring authenticity is itself multi-faceted. Academics have explored questions of detection (Perkins, 2023), university policies and the guidance that is given to students and academics (Moorhouse et al., 2023; Luo, 2024), and the experiences of students who have been accused of misconduct using generative AI (Gorichanaz, 2023). Best practice advice on assessment design has typically been generated in workshops, consultations, and reviews (e.g., Lodge et al., 2023; Hsiao et al., 2023), so is usually descriptive rather than prescriptive.

Most reports recommend approaches around assessment redesign and gathering evidence of learning, rather than relying on detection or current gaps in GenAI's abilities. This makes sense, given the nature of GenAI evolution. In 2022, academics were publishing findings that GenAI could perform well on higher education assessments (e.g., Finnie-Ainsley et al., 2022); by March 2023, the developers of GPT-4 were reporting using its performance in human examinations as a quality measure in the AI's own technical report (OpenAI: Achiam et al., 2023). We should expect this to continue – where AI is designed to assist with complex human tasks, the assessments that educators write to assess human skills will be an attractive training ground. Reports also recommend focusing authenticity efforts on the parts of assessment that most relate to degree learning outcomes (Lodge et al., 2023), and on means of ensuring meaningful student learning in assessment that may include AI, rather than a simplistic notion that students' work without AI input is more original (Luo, 2024); in other words, a focus on what the student did, rather than what the AI might have done.

Academic integrity assurance can carry a cost or time burden, for example, traditionally, the costs of proctoring or the time needed to conduct oral examinations. Institutions, therefore, should consider which assessments to assure as well as how to assure them. Australia has long used constructive alignment (Biggs, 1996) in curriculum design – essentially, aligning assessment tasks to subject learning outcomes, and subject learning outcomes to degree learning outcomes. So, Australian advice often focuses on whether there is sufficient assurance that a degree's learning outcomes have been demonstrated. The regulator's advice in this area (Lodge et al., 2023) takes a case study approach, using human review of curricula and assessment strategies to determine the level of risk.

There are some examples of augmenting this with mapping: collecting the strategies used for evidencing learning in subjects or assessments and collating these across each degree to demonstrate to reviewers the methods that are being used (e.g., Billingsley, 2022). However, this is currently only at a coarse level of detail, and it may be possible to produce finer-grained automated analysis. In the past, proposals (e.g., Veltri et al., 2015) have suggested using assessment weighting across curriculum maps to understand the relative weighting of different learning outcomes within a degree. This paper raises the question of how assessments and their evidence of authenticity could be modelled across a degree to understand where integrity risks exist and what evidence of learning may be collected.

A modern degree is already, to an extent, a complex evaluable expression – albeit

spread across different systems. Rules and pre-requisite pathways are evaluated within the enrolment system. Assessment scores are recorded and combined within subjects' gradebooks. A student's journey can be considered as the composition of these rules across their subject choices. From a computer science perspective, then, the question of what evidence of learning will be accrued across a student's subject choices can be related to the concept of effect tracking: analysing the observable effects a task has and the resources it requires, and how these combine as individual tasks to comprise a larger program. For example, in both an invigilated exam and in a student teacher's practicum, the student is observed, and this would be considered an effect on the observer.

Effect tracking

Abstractly, academic misconduct could be considered as an attempt to produce a submission that is indistinguishable from an authentic submission, without performing the same work. Students' motivations for engaging in misconduct vary but framing it as a deviation in process allows us to consider it from a computing perspective. To a compiler, minimising the work involved in a procedure to produce a result, without the changes being outwardly observable, is optimisation. Consequently, computing has long been interested in how and when programs and processes can be invisibly altered, and when they cannot be.

In computing, an expression is "pure" or "referentially transparent" if performing the expression can be replaced by inserting its result without changing the observable behaviour of the program. For example, if we were to ask a program to sum an immutable list of numbers and then later to re-sum the same list of numbers, then the second time we might not need to redo the calculation as we could reuse the value we calculated previously. Or, if we had an oracle available (something external to our program that we can just ask for the result), we could replace performing the sum with its answer. However, if the summation task were to have a "side-effect", such as printing out the numbers to the console as it went, then these optimisations could not be performed as they would change the observable behaviour of the program by printing the numbers out fewer times. This concept of referential transparency has been understood for a long time and functional programming languages have sought to maximise the use of pure functions since at least LISP in the 1960s (McCarthy, 1978).

Gifford and Lucassen (1986) introduced the concept that the compiler could model the effects that are present in a program. Their scheme ascribed an "effect class" to each function, and tracked how they combine as expressions are composed into a program. This effectively meant that each expression would have two types: the type of data it evaluated to, and the class of effects that it involved.

Later, Moggi (1991) proposed combining both sets of information into the return type of the expression, by representing the effects that are present in an expression using "monads" (a mathematical concept from category theory). For example, if we were to take our list summation, then without side-effects this function would receive a list of integers and return a single integer and therefore have a type such as:

$$List[Int] \Rightarrow Int$$

This would be deemed equivalent to another expression that takes a list of integers and produces the same result, including looking up a previously generated answer or asking an oracle. (This has a clear analogy with students submitting copied work or asking GenAI to produce an answer.) However, if the side-effect in our example –

outputting the numbers as it works – were expressed as part of the type system, its type would become

$$List[Int] \Rightarrow IO[Int]$$

and the expression would only be deemed equivalent to a process that printed the same output as well as returning the same answer. There are a wide range of effects programmers commonly seek to track; for example, access to outside systems which is important because those calls might fail or might never complete. The mathematical properties of a monad ensure that the effects are tracked as a program is composed from the functions it calls. For example, if a function calls a function that may fail, then the outer function may fail unless it handles the failure. A convention has evolved among programmers of calling an effect a “side-effect” if it is not represented in the type of the function, but just an “effect” if it is.

Modelling the type of an expression has relevance to academic integrity. Many traditional attempts to combat plagiarism have relied on strategies such as randomising questions or the values within them (e.g., Brusilovsky & Partak, 2002) or ensuring that different students’ work takes different paths (Sakzad et al., 2024). However, randomisation alone cannot combat outsourcing to generative AI as it does not change the type signature the assessment represents. It changes the input to each student, or the procedure each student is asked to perform, but it remains a simple function from input to answer with no outwardly observable effects. To a compiler, asking an oracle (e.g., an AI) would be deemed equivalent. If we introduce an observable effect, for example by proctoring the task or by logging students’ actions as they work, the type signature changes and asking an AI would no longer be equivalent to performing the task. The observable effects of students’ work render it less amenable to optimisation. Those side-effects can be considered the evidence of authenticity of work in the task. Some examples of observable effects are shown in Table 1.

Table 1: An example typology of assessments and their side-effects

Side effect	**Examples**
Observed work	Proctored exams, assessed practicals
External records	Version histories, system interaction logs, logs of automated test runs on past revisions
Human interaction	Client meetings, oral examinations, facilitated group work

If we are to track these observable effects (evidence of authenticity of student work) as tasks combine across the curriculum, then any scheme we use to model this must not become unwieldy. In computing, combining separate monads can become complex as programs grow. Consider, for example that to a compiler, *Task[Maybe[Int]]* (a task that will run and might produce an integer or nothing) is distinct from *Maybe[Task[Int]]* (a task that might or might not run but will produce an integer). The order of the types becomes significant, and this can make the types hard to work with. For simplicity, it may be better to use a single type to record effect information and annotate it with the kinds of effects that are present. This is conceptually similar to an effect-tracking scheme for computer programs proposed by Wadler and Thiemann (2003). In assessment, we can therefore propose annotating any assessment items that have observable side-effects. These side-

effects may include observation by invigilators, but can also include the impact that students' work has on the world: from children being taught in student-teacher practicums, to the continuous integration test run history of student work on software projects. Then, across any arbitrary course of study it is feasible to compose a description of what forms of evidence of learning may be produced.

Co-effects

As well as having effects, a program can also require resources, such as access to a data stream on which it will perform its computation, or memory requirements. Uustalu and Vene (2008) proposed tracking these using the mathematical dual of monads: comonads. Consequently, these context requirements are sometimes referred to as "co-effects". More recently, Gaboardi et al. (2016) proposed an approach that combined effects and co-effects, so that the effects a program has and the resources it requires can be tracked and analysed together. Their proposal also allowed for finer-grained consideration of effects, as it was able to consider whether an effect might, will, or will not occur.

Co-effects have their own natural parallel in assessment. Just as a project conducted by a student may produce certain effects as students work, they may also require certain resources to perform their task. For example, a project with an industry partner requires that partner.

Comonads are also relevant because assessment is the dual of work. Thus far, we have considered the problem from the perspective of students' work and the effects it may have. A system for understanding the risk present in a curriculum, however, would need to model the university's assessment system rather than modelling the student. To a student, the impacts of their work on observers and the world are effects. To an assessment system, however, they should be considered on the input rather than the output side. Where are these effects observed and what is the impact on credentialling if they are not? Rather than model them as effects of the student's work, we would model them as co-effects of the assessment: the required observations and evidence of learning that must be seen.

Conclusion

From a computational perspective, it seems realistic to propose a model of how assessment strategies compose across the assessments within a subject and across the subject options available in a curriculum. To construct such a model, assessments would need to describe any observable effect that they have and how that evidence of learning is considered within its grading. An assessment scheme within a subject can be modelled as an expression that composes its individual assessment items. A degree can be modelled as an expression of its course rules and the pre-requisite pathways by which subjects compose into a learning pathway. In a similar manner to effect tracking in software, the observable evidence of students' work can be composed across these expressions.

An initial attempt to create such a model is underway, being worked on in a public GitHub repository.[1] This provides a JavaScript API for users to describe degrees, subjects, assessments, and their academic integrity measures. It is based on earlier work that mapped academic integrity at subject-level (Billingsley, 2022), adapted to use a

[1] https://github.com/tweakedinfo/course-and-effect

comonad to model academic integrity at assessment level. From this model, we can derive interactive HTML views of how academic integrity measures, and the evidence of learning they produce, play out across curriculum pathways and learning outcomes.

The software is an experiment in modelling. Different universities have different structures and terminology for how they organise their courses, which may be a barrier to adopting the tool. However, by modelling courses with code, we can explore which analytical views of academic integrity are helpful in reviewing risks to academic integrity in qualifications.

Funding

No funding was received for the conduct of this research.

Disclosure statement

The author reports no potential conflict of interest.

Disclosure of the use of AI-assisted technologies during writing

No AI-assisted technologies were used during the writing process.

About the author

William Billingsley is an associate professor in computer science at the University of New England, the coordinator of UNE's undergraduate computing courses, and currently the Chair of the School Education Committee for UNE's School of Science & Technology. He has a background as a software engineer and as a researcher in human-computer interaction. His research focuses on technology education and education technology, and particularly how to make online and distance education smarter, more social, and more authentic.

ORCID: https://orcid.org/0000-0002-1720-9076

References

Biggs, J. (1996). Enhancing teaching through constructive alignment. *Higher Education*, *32*(3), 347–367. https://doi.org/10.1007/BF00138871

Billingsley, W. (2022). Lightweight mapping of identify [sic] verification methods and secondary course aspects: "Swiss cheese" modelling. In S. Wilson, N. Arthars, D. Wardak, P. Yeoman, E. Kalman, & D. Y. T. Liu (Eds.), *Reconnecting relationships through technology. Proceedings of the 39th international conference on innovation, practice and research in the use of educational technologies in tertiary education (ASCILITE 2022)*: (e22199). https://doi.org/10.14742/apubs.2022.199

Brusilovsky, P., & Pathak, S. (2002). Assessing student programming knowledge with web-based dynamic parameterized quizzes. In P. Barker & S. Rebelsky (Eds.), *ED-MEDIA 2002: World conference on educational multimedia, hypermedia & telecommunications* (pp. 1548–1553). Association for the Advancement of Computing in Education. https://www.learntechlib.org/p/9952/

Cotton, D., Cotton, P., & Shipway, J. (2023). Chatting and cheating: Ensuring academic integrity in the era of ChatGPT. *Innovations in Education and Teaching International*, *61*(2), 228–239, https://doi.org/10.1080/14703297.2023.2190148

Dai, W., Lin, J., Li, T., Tsai, Y., Gasevic, D., & Chen, G. (2023). Can large language models provide feedback to students? A case study on ChatGPT. In M. Chang, N.-S. Chen, R. Kuo, G. Rudolph, D. G. Sampson, & A. Tlili (Eds.), *2023 IEEE international conference on advanced learning technologies (ICALT)*, (pp. 323–325). https://doig.org/10.1109/ICALT58122.2023.00100

Diwan, C., Srinivasa, S., Suri, G., Agarwal, S., & Ram, P. (2023). AI-based learning content generation and learning pathway augmentation to increase learner engagement. *Computers and Education: Artificial Intelligence*, *4*, 100110. https://doi.org/10.1016/j.caeai.2022.100110

Doroudi, S. (2022). The intertwined histories of artificial intelligence and education. *International Journal of Artificial Intelligence in Education*, *33*, 885–928. https://doi.org/10.1007/s40593-022-00313-2

Finnie-Ansley, J., Denny, P., Becker, B. A., Luxton-Reilly, A., & Prather, J. (2022). The robots are coming: Exploring the implications of OpenAI codex on introductory programming. In J. Sheard & P. Denny (Eds.) *ACE '22: Proceedings of the Australasian computing education conference* (pp. 10–19). Association for Computing Machinery. https://doi.org/10.1145/3511861.3511863

Gaboardi, M., Katsumata, S., Orchard, D., Breuvart, F., & Uustalu, T. (2016). Combining effects and coeffects via grading. In *Proceedings of the 21st ACM SIGPLAN international conference on functional programming (ICFP 2016)* (pp. 476–489). Association for Computing Machinery. https://doi.org/10.1145/2951913.2951939

Gifford, D.K. & Lucassen, J.M. (1986). Integrating functional and imperative programming. In W. L. Scherlis, J. H. Williams, & R. P. Gabriel (Eds.), *Proceedings of the 1986 ACM conference on LISP and functional programming (LFP '86)* (pp. 28–38). Association for Computing Machinery. https://doi.org/10.1145/319838.319848

Gorichanaz, T. (2023). Accused: How students respond to allegations of using ChatGPT on assessments. *Learning: Research and Practice*, *9*(2), 183–196. https://doi.org/10.1080/23735082.2023.2254787

Hsiao, Y., Klijn, N., & Chiu, M. (2023). Developing a framework to re-design writing assignment assessment for the era of Large Language Models, *Learning: Research and Practice*, 9(2), 148–158. https://doi.org/10.1080/23735082.2023.2257234

Lodge, J.M., Howard, S., & Bearman, M. (2023). *Assessment reform for the age of artificial intelligence*. Tertiary Education Quality and Standards Agency.

Luo, J. (2024). A critical review of GenAI policies in higher education assessment: A call to reconsider the "originality" of students' work. *Assessment & Evaluation in Higher Education, 49*(5), 651–664. https://doi.org/10.1080/02602938.2024.2309963

Markauskaite, L., Marrone, R., Poquet, O., Knight, S., Martinez-Maldonado, R., Howard, S., Tondeur, J., De Laat, M., Shum, S. B., Gašević, D., & Siemens, G. (2022). Rethinking the entwinement between artificial intelligence and human learning: What capabilities do learners need for a world with AI? *Computers and Education: Artificial Intelligence*, *3*, 100056. https://doi.org/10.1016/j.caeai.2022.100056

McCarthy, J. (1978). History of LISP. In R. L. Wexelblat (Ed.), *History of programming languages* (pp. 173–185). Association for Computing Machinery. https://doi.org/10.1145/800025.1198360

Moggi, E. (1991). Notions of computation and monads. *Information and Computation*, *93*(1), 55–92. https://doi.org/10.1016/0890-5401(91)90052-4

Moorhouse, B., Yeo, M., & Wan, Y. (2023). Generative AI tools and assessment: Guidelines of the world's top-ranking universities. *Computers and Education Open*, *5*, 100151. https://doi.org/10.1016/j.caeo.2023.100151

OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D. Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., … Zoph, B. (2023). GPT-4 technical report. *arXiv preprint arXiv:2303.08774*. https://doi.org/10.48550/arXiv.2303.08774

Perkins, M. (2023). Academic Integrity considerations of AI Large Language Models in the pandemic era: ChatGPT and beyond. *Journal of University Teaching & Learning Practice*, *20*(2). https://doi.org/10.53761/1.20.02.07

Sakzad, A., Paul, D., Sheard, J., Brankovic, L., Skerritt, M. P., Li, N., Minagar, S., Simon, & Billingsley, W. (2024). Diverging assessments: What, why, and experiences. In *Proceedings of the 55th ACM technical symposium on computer science education (SIGCSE 2024)* (pp. 1161–1167). Association for Computing Machinery. https://doi.org/10.1145/3626252.3630832

Sharples, M. (2023). Towards social generative AI for education: theory, practices and ethics.

Learning: Research and Practice, *9*(2), 159–167. https://doi.org/10.1080/23735082.2023.2261131

Smolansky, A., Cram, A., Raduescu, C., Zeivots, S., Huber, E. & Kizilcec, R. (2023). Educator and student perspectives on the impact of generative AI on assessments in higher education. In D. Spikol (Ed.), *Proceedings of the tenth ACM conference on learning @ scale (L@S '23)* (pp. 378–382). Association for Computing Machinery. https://doi.org/10.1145/3573051.3596191

Uustalu, T. & Vene, V. (2008). Comonadic notions of computation. *Electronic Notes in Theoretical Computer Science*, *203*(5), 263–284. https://doi.org/10.1016/j.entcs.2008.05.029

Veltri, N.F., Webb, H.W., Matveev, A.G., & Zapatero, E.G. (2015). Curriculum mapping as a tool for continuous improvement of IS curriculum. *Journal of Information Systems Education*, *22*(1), 31–42. https://aisel.aisnet.org/jise/vol22/iss1/4

Wadler, P. & Thiemann, P. (2003). The marriage of effects and monads. *ACM Transactions on Computational Logic*, *4*(1), 1–32. https://doi.org/10.1145/601775.601776