Programação com contractos no *Learn-OCaml* ⋆

Rui Barata, Carlos Pinto, and Simão Melo de Sousa

NOVA-LINCS, Universidade da Beira Interior, Portugal
rui.barata@ubi.pt
caramelo.pinto@ubi.pt
desousa@di.ubi.pt

O projeto *Learn-OCaml* [1] é uma plataforma online que oferece tutoriais interativos e exercícios para aprender a linguagem de programação *OCaml*. Esta é uma linguagem funcional substancialmente usada tanto na indústria como na academia para diversas tarefas, desde processamento de dados e *blockchains* a desenvolvimento *web*. O *Learn-OCaml* foi criado para fornecer uma maneira mais abrangente e acessível para pessoas aprenderem *OCaml*, quer sejam iniciantes ou programadores experiente.

Neste momento, existe uma séria ameaça ao ensino online, esta é a dependência, que os alunos têm, de ferramentas de inteligência artificial (IA) que os ajudam a completar as tarefas das aulas. Embora estas ferramentas possam ser extremamente úteis para programadores experientes, estas podem ser utilizadas indevidamente por alunos que procuram concluir as tarefas com o menor esforço possível. Isto pode levar a que, os alunos não entendam completamente o código que escreveram, o que pode prejudicar seu desenvolvimento como programadores.

Com este documento, apresentamos a nossa proposta, ainda em desenvolvimento, para melhorar a plataforma *Learn-OCaml*, abordando a desvantagem mencionada anteriormente. Planeamos introduzir uma ferramenta de verificação de contratos de código que não requer provas formais, apenas utiliza a geração de testes. Com essa ferramenta, os alunos poderão especificar pré-condições, pós-condições, variantes e invariantes para um subconjunto de programas OCaml, o que lhes dará um primeiro contacto com a especificação comportamental e lhes fará dar uma explicação semântica sobre o comportamento esperado de suas soluções.

Pretendemos desenvolver uma ferramenta que seja capaz de ler contratos de código semelhantes aos usados no *Why3* a partir de comentários de código em OCaml escritos na linguagem de especificação *Gospel* [2]. Com base nesses contratos, planeamos implementar uma ferramenta equivalente ao *jmlc* para testar as asserções criadas a partir dos contratos de código presentes nos ficheiros *.ml* e comparar as respostas desta ferramenta aplicada ao código dos alunos com as respostas da ferramenta aplicada ao código do professor usando apenas testes.

⋆ Este trabalho foi parcialmente financiado por OCaml Software Foundation, no projeto LEAF e NOVA-LINCS (Ref. UIDP/04516/2020). Como também HORUS (Plataforma Inteligente para Mapeamento e Avaliação de Competências, PT2020 - SI I&DT - I&D em Co-Promoção) e GREENSTAMP (GreenStamp: Mobile Energy Efficiency Services, CENTRO-01-0247-FEDER-047256).

Esta ferramenta terá como objetivo auxiliar os alunos a garantir a correção e a fiabilidade do código *OCaml*, oferecendo uma abordagem baseada em contratos para verificar propriedades e comportamentos do programa. Acreditamos que essa ferramenta será uma adição valiosa ao ecossistema de desenvolvimento *OCaml* e *Learn-OCaml*, promovendo a escrita de código mais robusto e seguro.

O desenvolvimento da ferramenta está a seguir um plano de trabalho composto por etapas fundamentais para alcançar os objetivos propostos. Este plano inclui as seguintes fases:

Projetar a arquitetura da ferramenta Nesta fase, serão identificados os componentes e a interação entre eles, garantindo uma estrutura adequada para o funcionamento da mesma.

Modificação do parser Gospel Será adaptado um módulo de análise sintática baseado na ferramenta *Gospel* responsável por extrair as informações dos contratos de código presentes nos comentários. O *parser* será capaz de identificar as especificações e transformá-las asserções adequadas para processamento posterior.

Desenvolvimento do mecanismo de verificação no Learn-OCaml baseado em testes Nesta fase, será implementado um mecanismo capaz de verificar as asserções geradas a partir dos contratos de código diretamente na plataforma *Learn-OCaml* apenas usando testes. Serão utilizadas técnicas de geração de testes baseados nas asserções do professor e nos mecanismos de geração de testes da plataforma (ou introduzindo novos mecanismos como foi feito em [4] ou até baseados em ferramentas de *property-based testing* como o *qcheck [3]* e incorporar estes no *Learn-OCaml*), comparando as respostas das especificações dos alunos com as respostas das especificações do professor a estes testes. Isto permitirá que os alunos submetam as suas especificações do código que escrevem e recebam *feedback* em tempo real sobre a correção das mesmas. A integração garantirá uma experiência contínua e integrada aos alunos, facilitando o uso da ferramenta no contexto do ambiente de aprendizagem.

Testes e validação Serão realizados testes para verificar a qualidade e robustez da ferramenta. Serão criadas e adaptadas fichas de exercícios que abrangem uma variedade de cenários e situações para garantir que a ferramenta seja eficaz na deteção de erros de especificação e na melhoria da qualidade do código produzido pelos alunos. Os resultados dos testes serão analisados e utilizados para melhorar a ferramenta.

Ao seguir este plano de trabalho, espera-se desenvolver uma ferramenta eficiente e confiável, capaz de auxiliar os alunos na escrita de especificações corretas e melhorar a qualidade do código produzido no contexto do ensino de programação.

No futuro, pretendemos acabar de implementar e avaliar esta ferramenta, bem como expandir a variedade de exercícios e cenários de uso. Acreditamos que a nossa abordagem pode contribuir significativamente para o ensino e a

prática de programação, permitindo ao alunos aprimorar as suas habilidades e capacita-los a escreverem código mais robusto e confiável.

References

1. Canou, B., Henry, G., Bozman, Ç., Le Fessant, F.: Learn ocaml, an online learning center for ocaml. In: OCaml Users and Developers Workshop 2016. p. 4. Nara, Japan (September 2016)
2. Charguéraud, A., Filliâtre, J.C., Lourenço, C., Pereira, M.: GOSPEL—Providing OCaml with a Formal Specification Language. In: ter Beek, M.H., McIver, A., Oliveira, J.N. (eds.) Formal Methods – The Next 30 Years, vol. 11800, pp. 484–501. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-30942-8_29, series Title: Lecture Notes in Computer Science
3. Cruanes, S.: QCheck (Feb 2023), https://github.com/c-cube/qcheck, original-date: 2013-10-06T21:06:54Z
4. Sylvestre, L., Chailloux, E.: Expérimentations pédagogiques en Learn-OCaml. In: JFLA 2020 - 31ème Journées Francophones des Langages Applicatifs. Gruissan, France (Jan 2020), https://hal.science/hal-03154266