识别函数式编程作业中的不同学生群组：从快速学习者到困难型学生

Chuqin Geng 麦吉尔大学 加拿大魁北克省蒙特利尔 chuqin.geng@mail.mcgill.ca

Wenwen Xu 麦吉尔大学 加拿大魁北克省蒙特利尔 wenwen.xu2@mail.mcgill.ca

Yingjie Xu 麦吉尔大学 加拿大魁北克省蒙特利尔 yj.xu@mail.mcgill.ca

Brigitte Pientka 麦吉尔大学 加拿大魁北克省蒙特利尔 bpientka@cs.mcgill.ca

Xujie Si 麦吉尔大学 加拿大魁北克省蒙特利尔 xsi@cs.mcgill.ca

arXiv:2301.02611v1 [cs.CY] 6 Jan 2023

摘要

授课教师和学生往往都关注编程作业的成绩，并将其视为衡量学生掌握材料程度以及是否在学习中挣扎的关键指标。然而，这可能会造成误导。尤其是在学生能够使用自动评分器的情况下，他们的成绩可能会严重失真。

本文分析了麦吉尔大学一门函数式编程课程中收集的学生作业提交数据，并纳入了广泛的特征。除了成绩，我们还考察了活动时间数据、投入时间以及静态错误的数量。通过聚类算法，我们据此识别出四类学生群组：“快速学习型”“勤奋型”“满足型”和“挣扎型”。随后，我们分析学习习惯、工作时长、错误范围以及修正错误的能力如何影响不同的学生群组。这种结构化分析为授课教师提供了宝贵的见解，使他们能够主动帮助不同类型的学生，并在整体课程设计中着重关注不同方面。此外，这也有助于学生了解自己仍在哪些方面存在困难，从而寻求讲解并调整学习习惯。

CCS 概念

• **社会与专业议题** →**学生评估**。

关键词

在线编程平台；计算机科学教育；聚类分析

1 引言

在线编程环境，例如用于 C 的 RoboProf [8] 、用于 Scheme 的 DrScheme ++，以及较新的 Mumuki [13, 14] ，在提升大规模编程课程中学生的学习体验方面潜力巨大。它们不仅降低了初学者的入门门槛，往往还提供自动评分功能，让学生在开发程序时运行代码并获得反馈，从而有机会立即修复程序错误并纠正理解上的错误。虽然人们已经认识到，及时获取代码反馈能显著改善学生学习成果并提高参与度（例如参见 [4] , ），但授课教师和学生往往都过于关注成绩，将其视为衡量能力的关键指标。尤其是当学生能够使用自动评分器时，他们的成绩可能会严重失真，产生误导。

本文提出一种数据驱动方法，以更好地理解学生在函数式编程课程中完成编程作业时的行为。除了成绩之外，我们还提出考量静态错误数量、完成编程作业总耗时等其他因素，以识别学生群组。我们运用此方法，分析了麦吉尔大学开设的一门函数式编程课程所收集的作业提交数据；该课程使用 Learn-OCaml 在线编程平台 [5, 6, 17]。据此，我们识别出四类学生群组：“快速学习型”、“勤奋型”、“满足型”和“挣扎型”。前两类群组可归为最大化者，即学生力求取得尽可能高的成绩，并不断改进自己的作业；但他们在完成一项家庭作业所花费的时间和精力上仍有差异。相比之下，满足型[1] 学生会将可能并非最优的结果视为“足够好”，从而以节省时间和精力的方式充分实现目标。我们还从学习习惯以及常见错误的数量和类型等方面进一步分析了这些群组。由此得出四项关键发现：

- 借助昼夜节律类型这一人类和动物的昼夜节律类型学概念，我们证实，学生倾向于在早晨学习的模式与学业成功有关。具体而言，快速学习者倾向于在早晨投入更多时间，而其他学生群组则更多依赖下午和傍晚。
- 一般而言，较早开始做家庭作业与较高的成绩有关。不过，我们也注意到，满意化型学生开始得相对较晚，却最早完成。这进一步说明，满意化型学生追求的是令人满意的结果，而非最优结果。同时，满意化型学生的编程错误数量处于最低之列，这表明，与勤奋型学生等相比，他们在静态错误方面遇到的困难要小得多。
- 我们对静态错误的分析表明，所有学生中语法错误和类型错误都很普遍。此外，学生

[1]“满意化”一词由H. Simon [27] 提出，用来描述一种决策过程：个体做出的选择是令人满意的，而非最优的。

在整个学期中都持续受到这些错误的困扰。此外，我们的分析还指出了其他常见错误，例如非穷尽式案例分析和使用未绑定变量。

- 考虑到学生修复静态错误的能力，即学生修复某个特定错误所需的尝试次数，我们注意到，勤奋型学生的失败/成功比率尤其高。这既凸显了他们争取最佳成绩的愿望和动力，也表明他们的学习道路上充满了许多小的障碍。

我们相信，我们提出的特征集合和数据驱动分析能够为授课教师提供一幅更加清晰、详细的学生行为和表现图景。反过来，这些信息可以用于调整某些概念的教学方式，例如如何避免静态错误。它还可以用于针对不同学生制定不同策略，以提升学生的学习体验。此外，这些数据对学生自身也可能很有意义，帮助他们更好地了解自己在课堂上的表现，并找出可以主动改进和尽早寻求帮助的领域。

2 相关工作

分析编程课程中的学生数据是学习分析的核心主题，而随着近年来数据存储和处理技术的发展，这一主题正受到越来越多的关注。分析学生数据的主要目标之一是了解学生的行为，进而改善学生的学习体验 [21]。

过去十年间，已有多项研究专注于通过聚类分析识别学生群组。例如，Emerson 等人 [12] 根据程序结构，在面向非计算机科学专业学生的基于积木块的编程环境中使用聚类算法识别学生的概念误解。Wiggins 等人 [29]在配备智能导师的基于积木块的编程系统中发现了五种主要的提示请求聚类。Hossein 等人 [20] 通过在每次发生操作时收集编程快照，利用聚类分析进一步探究学生编程速度与编程行为之间的相关性。随后，他们将学生的编程速度与中位速度进行比较，并据此将学生分为两个聚类。Lahtinen 等人 [23] 使用布鲁姆分类法的不同层级作为特征，识别出六个不同的学生群组，教师在教授入门编程课程时应对此有所了解。

与这些已有研究不同，我们的工作考虑了多类别特征，包括成绩、作业总耗时以及遇到的静态错误数量，以识别学生群组。

基于识别出的聚类，我们沿用现有研究，了解学生的工作与休息模式。具体而言，Claes 等人[7] 通过对 86 个大型开源软件项目中已提交活动的时间戳进行聚类分析，研究程序员的工作模式。Zavgorodniaia 等人 [31] 利用击键数据和聚类算法研究学生的昼夜节律类型。在我们的研究中，我们使用活动数据（例如学生是否编译或批改了家庭作业）来研究学生的工作与休息模式。这是首项在带类型的函数式编程领域开展的此类研究。

我们进一步分析类型化函数式编程作业中的静态错误及其对不同学生群组的影响。这是首次在这一情境下开展此类研究。以往研究主要关注用 Java 编写的面向对象程序中的编译事件。例如，Ahmadzadeh 等人 [1] 考察了学生程序中的编译器错误频率以及 Java 程序中的调试活动模式。他们认为，在编程教学中应强调调试技能。Altadmri 等人 [2] 收集了一个大型数据集，其中包含 25 万名学生的编译事件，为分析错误模式以及修复不同错误所需的时间提供了可靠依据。Denny 等人 [9] 也研究了各种语法错误的频率，以及学生修复常见语法错误所花费的时间。他们还发现，即使是能力较强的学生，也会觉得某些类型的错误颇具挑战性。Edwardset 等人 [11]分析了学生在五个学期内提交的程序中发现的 1000 万个静态分析错误，这些程序提交超过 50 万份。实验结果表明，计算机科学专业学生和非专业学生所犯错误的频率一致。

我们的分析是首批深入调查带类型的函数式编程作业中各种静态错误发生频率的研究之一。这里的静态错误不仅包括语法错误和简单的类型错误，还包括例如检测程序中缺失分支的情况。

3 研究设计

本研究旨在深入了解学生如何编写类型化函数式程序（TFP）。我们认为，单凭成绩并不能很好地反映学生对基本概念的掌握程度和能力水平。相反，我们提出，综合考虑学生花费的时间以及遇到的错误数量，能够更细致地呈现情况。因此，本文要探讨的核心问题是：如何综合考虑成绩、耗时和错误数量，最佳地识别不同的学生聚类？随后，我们将围绕五个假设分析这些聚类：

H1: 即使编程作业成绩很高，学生仍可能在处理各种静态错误时遇到很大困难。**H2:** 尽管成绩较低，但花费时间较少且静态错误数量较少的学生，整体表现确实不错。**H3:** 学生的工作与休息模式以及花在家庭作业上的时间，都会影响学生能否取得高分。这体现了学生的投入程度。**H4:** 类型化函数式编程中的静态错误涵盖语法错误、类型错误，以及程序中未绑定变量和缺失分支的检测。这些类型多样的静态错误能够细致地呈现学生觉得有挑战的概念。
H5: 错误修复比率，即学生修复一个静态错误所需的尝试次数，可反映学生对类型化函数式编程基本概念的理解程度，并与他们的理解和表现相关。

3.1 课程背景

我们的研究关注麦吉尔大学一门面向本科二年级学生的计算机科学课程。该课程介绍函数式编程和编程范式的相关概念。每学期都有这门课程，注册的本科生超过300名。本研究的所有数据均收集于2021年秋季

	编程主题	#任务
HW1	基本表达式、递归	7
HW2	数据类型和模式匹配	6
HW3	高阶函数	11
HW4	引用，状态，记忆	5
HW5	异常，续延	5
HW6	惰性编程，玩具语言	5

表 1：六项编程作业概览。

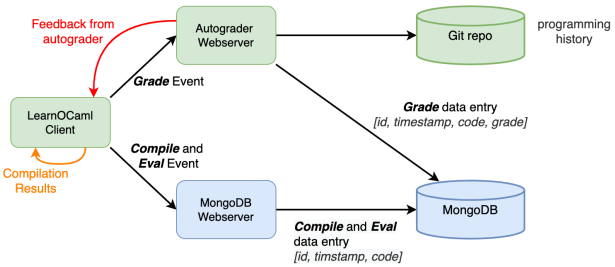


图 1：数据收集流程。 成绩、编译和评估事件由不同的服务器处理，所有提交数据都存储在 *MongoDB* 数据库中。浅绿色高亮的组件是 Learn-OCaml 平台的原有组件，浅蓝色高亮的组件则是我们新增的。

学期，当时学生可以在线通过 Zoom 上课，也可以参加线下课程。

课程共设有六次每两周一次的编程作业，每次占期末成绩的 5%。每份家庭作业都包含若干编程任务，要求实现函数和测试用例。家庭作业信息汇总见表 1。所有家庭作业均通过 Learn-OCaml [6], 这一 OCaml 在线编程平台发布，学生可以在同一平台上编辑、编译、测试和调试代码。我们使用了 Hameer 和 Pientka [18] 开发的 Learn-OCaml 修改版，并添加了风格检查以及对学生编写的测试用例进行评估等功能。

3.2 数据收集

我们的数据收集流程基于 Learn-OCaml 平台构建，可以自动记录学生的操作。具体来说，我们通过异步日志请求，将编译和评估（用于测试和调试）等本地编程事件发送到后端服务器。图 1 展示了从在线环境 Learn-OCaml 收集数据的流程。

约 52.81% 的学生（即 320 人中的 169 人）同意我们访问其数据。我们收集了超过 270,000 条编程事件，每条事件都存储了代码快照以及反馈信息（例如时间戳、静态错误、成绩等）。

3.3 特征

对于每份家庭作业，我们会收集一系列编程活动事件。这些活动事件包括成绩、编译和评估事件。这样，我们就能为每位学生创建一个**活动密度向量** 。这是一个包含四个元素的向量，表示学生在不同时间段 [0-6、6-12、12-18、18-0], 内发生的活动事件所占的百分比；这些时间段与 [31] 中建议的划分相同。

此外，我们基于活动事件序列设计了以下特征：

• **开始时间**。根据收集到的活动事件，学生开始积极完成作业的日期。• **结束时间**。学生完成作业的日期，即最后一个评分事件发生的日期。

• **工作时段**。定义为活动事件发生的时间窗口。如果 30 分钟内没有活动事件，则认为该工作时段已结束。• **总耗时**。所有工作时段时长之和。

• **错误数量**。学生完成作业时产生的静态错误数量。• **成绩**。学生在作业中获得的最终成绩。

3.4 特征工程

在本研究中应用聚类算法和统计检验面临两项挑战。第一项是数据偏斜。例如，成绩严重偏斜，因为学生可以通过与自动评分器互动不断提高成绩。第二项是特征尺度之间的差异，这会导致聚类结果不一致。我们采用两种方法应对这些挑战。首先，我们使用包括斯皮尔曼相关和克鲁斯卡尔-沃利斯 H 检验在内的非参数检验。其次，我们对特征进行秩变换，以便应用聚类算法。

4 学生群组识别

为识别学生群组，我们对六次家庭作业中三个最重要特征（即成绩、错误数量 和 耗时）的聚合值（均值）运行 K- 均值 [19] 聚类算法。我们使用肘部法 确定最优 k（聚类数量）为 4。确定最优 k 后，我们重新运行 K-均值算法，并在表 2 中报告结果。时间以小时为单位，并且所有聚类的学生人数相近（$\#Std$）。

为确定得到的四个聚类是否存在差异，我们针对每个聚类的三个特征（耗时、错误数和成绩）进行了克鲁斯卡尔-沃利斯 H 检验，这是一种非参数检验，相当于方差分析。结果具有统计学显著性，统计量分别为 113.26、100.87 和 123.02，且所有 p 值均为 < 0.0001。这表明这四个聚类在统计学上存在差异。

聚类 A 中的学生平均成绩最高（95.24），而完成家庭作业所花的时间不到预期的 6 小时。这表明他们能够相对轻松地达成目标。事实上，

聚类	#Std	时间（小时）	# 错误	成绩
A - 快速学习	46	5.30 (± 0.94)	66.11 (± 26.95)	95.24 (± 3.25)
B - 勤奋型	46	8.24 (± 1.52)	148.67 (± 63.26)	94.25 (± 3.90)
C - 满足型	31	4.47 (± 1.01)	52.26 (± 21.89)	74.43 (± 11.31)
D - 挣扎型	46	6.49 (± 0.94)	118.14 (± 35.32)	72.81 (± 11.03)

表 2：学生群组

该聚类中的学生表现远胜其他聚类中的学生。我们将这一聚类描述为**快速学习**。

聚类B中的学生平均成绩排名第二（94.25）。然而，他们的平均错误数量也是最高的（148.67），且花在家庭作业上的时间显著多于其他任何群体，达到8.24小时。具体而言，他们花费的时间显著超出预期。这表明他们面临许多困难，但通过投入大量时间成功克服了这些困难。这些学生积极改进自己的作业，以取得尽可能高的成绩。因此，我们将他们归类为**勤奋型**。这些数据支持我们的假设**H1**。

聚类C的平均错误数量最低（52.26），花在家庭作业上的时间也最少（4.47小时）。他们的平均成绩为74.43，仍然取得了“足够好”的成绩。这些学生通过节省时间和精力来实现目标。同时，较低的平均错误数量也表明，他们达到了令人满意的能力水平。我们将这些学生称为**满足型**学生。这支持我们的假设**H2**。

聚类D中的学生实际上与聚类B中的学生密切相关，后者同样表现出较高的平均错误数量（118.14）和较长的用时（6.49小时）。然而，与聚类B中的学生相比，他们未能克服学习过程中的困难。这些学生属于**挣扎型**。

5 理解学生群组

5.1 不同学生群组的学习习惯有何差异?

为检验假设**H3**，我们根据活动数据考察学生何时活跃。先前研究指出，chronotype类型（即一个人偏好在一天中的特定时段进行活动）受昼夜节律周期支配，而昼夜节律周期由时钟基因控制 [10, 25]。本节旨在探究学生的昼夜节律类型，换言之，即学生的学习习惯。特别是，有研究发现，“早晨型”与学业成就呈正相关 [24, 31]。

为识别潜在的昼夜节律类型，我们在由活动密度向量张成的特征空间中运行 K-均值聚类算法。肘部法得到 $k = 3$，表明可能存在三种昼夜节律类型，这不同于[31]中报告的四种昼夜节律类型。我们在表 3 中报告每个昼夜节律类型聚类的质心。

时间聚类	0 - 6	6 - 12	12 - 18	18 - 0	昼夜节律类型
簇 1	8%	14%	26%	**52%**	夜型（Eve）
簇 2	4%	**26%**	20%	50%	晨型（Mor）
簇 3	2%	19%	**37%**	42%	下午型（Aft）

表 3：各昼夜节律类型的质心。

如我们所见，三个聚类中的大多数活动都发生在 18:00 - 00:00。这并不奇怪，因为大多数学生白天可能有课。根据这一观察，我们还会考虑次要活动峰值来定义昼夜节律类型。我们注意到，簇 2 的次要活动峰值（26%）出现在 6:00 - 12:00，而簇 3 的次要活动峰值（37%）出现在 12:00 - 18:00。因此，我们将簇 2 和簇 3 定义为**早晨**（Mor）和**下午**（Aft）类型。簇 1 在 18:00 - 00:00 只有一个活动峰值，因此我们将其定义为**傍晚**（Eve）类型。

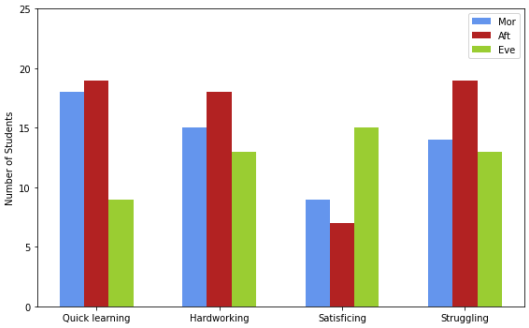


图 2：各学生簇中的昼夜节律类型分布。

如图 2 所示，快速学习型学生通常倾向于在早晨和下午学习，而满足型学生则在傍晚完成家庭作业。这表明快速学习型学生更有进取心、更具学习动力，并且可能拥有更好的时间管理能力。总体而言，满足型学生是唯一明显倾向于在傍晚学习的群体。这可能表明学生还承担着其他事务，或课程负担较重。下午型最常出现在困难型和勤奋型簇中。这可能是因为他们在白天提供的办公时间寻求帮助，或者他们通常只是需要更多时间。总体来看，我们的结果证实了先前的研究发现，即某些昼夜节律类型与学业成就相关[24, 31]。

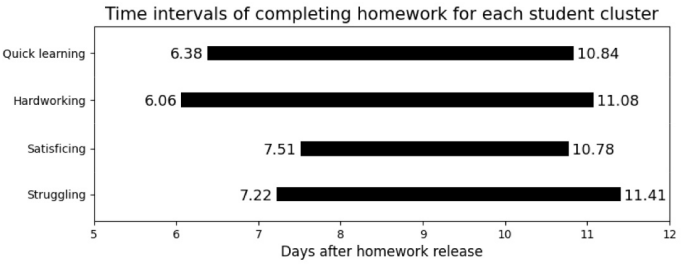


图 3：不同类型学生的聚类结果时间间隔的起点表示平均开始时间，而终点表示平均结束时间。

5.2 不同学生聚类在家庭作业上花费多长时间?

为进一步探究假设 **H3**，我们考察特定聚类中的学生何时开始和完成家庭作业。我们在图 3 中报告了各聚类的平均开始时间和结束时间。此外，克鲁斯卡尔-沃利斯 H 检验表明，开始日期存在统计学上的显著差异（统计量 $= 22.59$，p 值 < 0.0001），而结束日期则无显著差异（统计量 $= 3.12$，p 值 $= 0.37$）。尽管如此，我们仍能观察到一些有趣的模式。

错误组	错误类别	HW1	HW2	HW2	HW4	HW5	HW6
A. 一般静态错误	1. 类型错误	38.12%	30.94%	40.93%	32.65%	36.90%	34.83%
	2. 语法错误	42.33%	21.54%	21.79%	32.68%	17.80%	25.66%
	3. 未绑定值	10.42%	7.19%	9.06%	13.42%	7.02%	7.27%
	4. 缺少 else 分支	1.92%	0.75%	0.43%	0.08%	1.03%	1.07%
B. 命令式思维错误	5. 未使用的变量	0.74%	0.65%	0.63%	6.37%	21.34%	7.23%
C. 模式匹配错误	6. 模式匹配类型错误	0.84%	5.24%	2.13%	0.62%	1.37%	1.40%
	7. 非穷尽模式匹配	1.02%	16.78%	15.74 %	2.47%	4.62%	11.92%
D. 函数应用错误	8. 参数数量错误	1.67%	2.19%	3.38%	1.17%	2.09%	1.89%
	9. 误用非函数值	2.50%	2.10%	2.07%	1.72%	1.50%	1.77%
	10. 其他	0.88%	12.6%	5.89%	8.83%	6.33%	6.96%
	错误总数	7,850	27,519	14,331	19,859	22,467	26,681

表 4：错误组和错误类别及其在家庭作业中的分布

值得注意的是，满意化型学生和困难型学生开始做家庭作业的时间都相对较晚，平均分别为 7.51 天和 7.22 天。然而，满意化型学生最早完成作业（10.78 天）。这凸显了他们满足于“足够好”的结果，而非追求更好的成果。此外，满意化型学生的工作时长最短。这进一步印证了我们的观点：这些学生通过节省时间和精力来实现目标。

挣扎型学生遇到了许多困难，静态错误数量众多便是明证。这些学生确实最后才完成（完成时间为 11.41）。这表明他们处境艰难，尽管他们确实努力坚持到最后。然而，他们缺乏克服困难所需的技能或支持。

勤奋型学生的时间间隔最长。虽然他们开始得最早（6.06），但完成时间排在倒数第二（11.08）。这体现了他们对学习任务的投入与坚持。

快速学习型学生往往开始得比较早（6.38），但没有勤奋型学生那么早。这表明他们对自己顺利完成家庭作业的能力充满信心。

我们进行了斯皮尔曼相关分析，以考察开始时间与作业成绩之间的相关性。统计结果具有显著性（相关性 = -0.42，p 值 < 0.0001），这表明拖延会对学生学习成果产生负面影响，这一点已有大量研究报告 [3, 16, 22]。

5.3 静态错误如何影响不同聚类中的学生?

OCaml 等带类型的函数式编程语言的编译器会向程序员提供大量错误信息和反馈。它们不仅会报告语法错误和类型错误，还会报告未使用的变量，以及 case 语句和 if 表达式中的缺失分支。这为我们更好地了解学生最难掌握哪些基本概念奠定了基础。

5.3.1 静态错误概述。为验证我们的假设 **H4**，我们分析每次编译失败事件中的错误类型，并将错误归为四个主要类别：一般静态错误（例如 A 组）、由命令式思维导致的错误（B 组），以及与模式匹配和函数相关的错误（例如 C 组和 D 组）。我们还统计了各类错误在作业提交中出现的频率（见表 4）。

第一次家庭作业中遇到的语法错误显著增多（42.33%）。这并不令人意外，因为这是学生首次尝试用一种新语言编写程序。然而，整个学期中遇到的错误有 20% 到 30% 与语法和类型错误（A 组）有关，这或许令人意外。事实上，表 4 中每份家庭作业的错误中，这类错误都约占 60%。这或许说明，在类型化函数式编程中，类型错误能较早地捕捉到程序员思维中的概念性错误，而不是等到测试阶段才发现。这也可能意味着授课教师应投入更多时间，帮助学生理解类型错误分析。

对于模式匹配等带类型的函数式编程中的关键概念，我们的错误分析表明，学生确实有所进步，对这些概念也有了更深入的理解。在 HW2 中首次引入模式匹配时，模式匹配错误和非穷尽模式匹配错误（C 组）占静态错误总数的 22%。在完成 HW2 和 HW3 的练习后，错误组 C 的占比大幅下降，这表明随着编程练习的增加，学生对相关概念有了更深入的理解。

本课程的一项先修要求是修读入门计算机科学课程，该课程在我校以 Java 或 Python 授课。这意味着所有参与者在此之前都有编程经验，并且需要将命令式编程／面向对象编程（Python 或 Java）的概念迁移到函数式编程（OCaml）。对于条件语句 以及函数和作用域[28]等概念，学生通常表示，在过程式语言和面向对象语言之间转换很顺利。根据我们的观察，学生在转向函数式编程时遇到的困难更多。具体而言，他们难以理解已绑定或未绑定变量、if 表达式中的缺失分支，以及函数应用错误。尽管这些错误出现的频率低于语法错误和类型错误，但我们认为，这凸显了学生在递归思考以及考虑递归程序中所有情况方面存在困难（错误编号 4、7）。因此，在 OCaml 这类语言中，缺少 else 分支的 if-else 表达式也常会导致类型错误。

此外，命令式编程支持在局部或全局状态中声明变量，而在 OCaml 等函数式语言中，我们区分可更新的有状态变量和已绑定变量。尽管本课程经常讨论自由变量与已绑定变量的概念，以及有状态变量之间的区别

，学生仍会继续遇到与变量有关的错误。尤其是，整个学期都不断出现未绑定值错误。这似乎表明，学生思考特定问题时，仍受命令式编程中有状态变量声明这一概念的影响。函数式编程最核心的概念是函数是一等公民。因此，高阶函数——即将函数作为参数或返回函数的函数——被频繁使用，尤其是在 HW3 及后续作业中。如果函数使用不当，最常见的情况是被标记为类型错误。不过，OCaml 也会报告其他错误。具体而言，它可能会报告参数数量不正确（错误编号 8），或在非函数值上使用函数值而不是传入参数进行调用（错误编号 9）。这些错误构成了一个不可忽视的类别，反映出学生容易在哪些地方遇到困难。

5.3.2 各聚类中的学生修复错误的效率如何？最后，我们考察假设 **H5**，旨在了解不同聚类中的学生在快速修复错误的能力上有何差异。表 5 展示了整个学期中不同学生群组经历的成功编译事件和失败编译事件的平均数量。失败/成功比率 x 可大致解释为调试效率或错误修复率，即学生平均需要经历x次编译失败，才能成功编译一次。

	快速学习型	勤奋型	满足型	挣扎型
成功	37.9	60.4	28.1	40.7
失败	85.7	162.3	66.9	118
F/S	2.26	2.67	2.38	**2.90**

表 5：各学生簇编译事件的平均成功次数、失败次数及失败/成功比率（F/S）

困难型学生修复静态错误最为吃力，平均需要 2.9 次编译失败才能修复错误。相比之下，快速学习型学生的调试能力最强，平均只需经历 2.26 次编译失败便能成功。此外，如果比较他们平均的失败和成功次数，两者的调试效率差距就更加明显。困难型学生（40.7 次）与快速学习型学生（37.9 次）的平均成功次数相近，但平均失败次数差距显著：困难型学生平均比快速学习型学生多经历约 30 次编译失败。

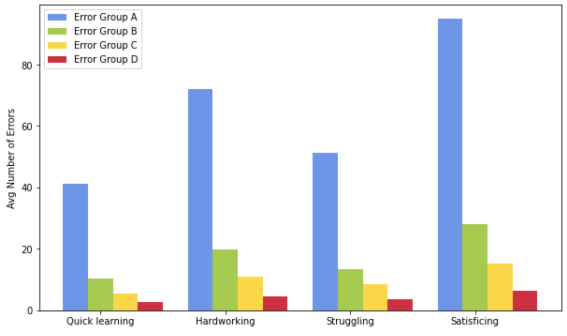


图 4：各学生簇中静态错误的分布。

表 5 中“失败”一行的数据，还可以通过图 4 中四个学生聚类各类静态错误的平均数量进一步呈现。对于所有聚类而言，类型错误和语法错误（A 组）都占主导地位，但也存在值得注意的差异。快速学习型学生在所有错误组中的错误都较少，不仅一般静态错误较少，函数式编程特有的错误也较少。满足型学生在 B、C 和 D 组中的错误最少，这或许表明他们确实具备了相应能力。最后，勤奋型学生和挣扎型学生在所有错误组中的错误都显著更多。尤其是，他们在边界变量或未使用变量、缺失分支以及函数的正确使用等基本概念方面遇到更多困难。

6 结论

本研究旨在基于 Learn-OCaml 编程平台收集的数据，了解学生如何完成函数式编程作业。我们分析了成绩、总耗时和静态错误总数，以识别出四个学生群组：“快速学习型”、“勤奋型”、“满足现状型”和“挣扎型”。我们使用统计检验验证了聚类结果及其他分析结果。这不仅细致呈现了学生的行为，也揭示了他们在课程中取得学业成功的不同路径。对昼夜节律类型的分析证实，早晨学习的学生能以最快且最顺利的方式取得最高成绩。学生完成家庭作业所花费的总时间，也凸显了不同学生群组之间的差异与相似之处。尽管这部分分析是在函数式编程课程背景下开展的，我们预计该方法也适用于其他编程课程，并有助于识别需要额外支持的学生群组。

我们对类型化函数式编程中静态错误的详细分析，也指出了授课教师可以调整课程内容并可能重新讲解的主题。我们认为，我们的分析也能为学生，尤其是勤奋型学生，提供启示，帮助他们了解自己仍在哪些方面遇到困难，并寻求澄清。这或许能让他们成为更高效的调试人员，减少完成家庭作业所花的时间，并加深概念理解。

引用

[1] Marzieh Ahmadzadeh、Dave Elliman 和 Colin Higgins。2005。计算机科学新手学生调试模式分析。第十届 *SIGCSE* 计算机科学教育创新与技术年会论文集——*ITiCSE ’ 05* 10 1145 1067445 1067472 。[2] 2015 37 https://doi.org/ 。/ 。Amjad Altadmri 和 Neil C.C. Brown。 。百万次编译。第 *46*届 *ACM*计算机科学教育技术研讨会论文集。https: //doi.org/ 10 1145 2676723 2677258 。/ 。Rahim Badri Gargari、Hossein [3] Sabouri 和 Fatemeh Norzad。2011。学业拖延：因果归因风格与行为拖延之间的关系。《伊朗精神病学与行为科学杂志》5, 2 (2011), 76–2。[4] Luciana Benotti、Federico Aloi、Franco Bulgarelli 和 Marcos J. Gomez。2018。基于网络的编码工具及自动反馈对学生表现和认知的影响。载于第*49*届 *ACM*计算机科学教育技术研讨会论文集（美国马里兰州巴尔的摩）*(SIGCSE ’18)*。计算机协会，纽约，纽约州，美国，2–7。https://doi.org/ 10 1145 3159450 3159579 。/ 。Benjamin Canou、Roberto Di [5] Cosmo 和 Grégoire Henry。2017。拓展函数式编程教育：OCaml 大规模开放在线课程幕后探秘。*ACM*《编程语言》论文集1, ICFP（2017年8月），1–25。https://doi.org/10.1145/3110248

[6] Benjamin Canou、Grégoire Henry、Çagdas Bozman 和 Fabrice Le Fessant。2016。Learn OCaml：面向 OCaml 的在线学习中心。[7] Maëlick Claes、Mika V. Mäntylä、Miikka Kuutila 和 Bram Adams。2018。程序员会在夜间或周末工作吗？第*40*届国际软件工程会议论文集。https://doi.org/10.1145/3180155.3180193[8] Charlie Daly。1999。RoboProf 与计算机编程入门课程。载于第四届 *SIGCSE/SIGCUE ITiCSE* 计算机科学教育创新与技术年会论文集（克拉科夫，波兰）*(ITiCSE '99)*。计算机协会，纽约，纽约州，美国，155–158。https://doi.org/ 10 1145 305786 305904 。/ 。Paul Denny、Andrew Luxton-Reilly 和 Ewan Tempero。2012。语法错误并非都一样。第*17*届 *ACM* 计算机科学教育创新与技术年会论文集——*ITiCSE '12*。https://doi.org/10.1145/2325296.2325318[10] Charna Dibner、Ueli Schibler 和 Urs Albrecht。2010。哺乳动物昼夜节律计时系统：中央时钟与外周时钟的组织和协调。《生理学年度评论》72, 1, 517–549。https://doi.org/10.1146/annurev-physiol- 021909 135821 - Stephen H. Edwards、Nischel Kandru 和 Mukund B.M. Rajagopal。2017。调查学生 Java 程序中的静态分析错误。*2017*年 *ACM*国际计算教育研究会议论文集。https: //doi.org/ 10 1145 3105726 3106182 。/ 。Andrew E merson、Andy Smith、Fernando J. Rodriguez、Eric N. Wiebe、Bradford W. Mott、Kristy Elizabeth Boyer 和 James C. Lester。2020。基于聚类的基于积木块的编程中新手编程错误认知分析。第*51*届 *ACM*计算机科学教育技术研讨会论文集。https: //doi.org/ 10 1145 3328778 3366924 。/ 。Mattias Felleisen、R. B. Findler、M. Flatt 和 S. Krishnamurthi。1998。DrScheme项目：概述。*SIGPLAN* 通讯33, 6（1998年6月），17–23。https://doi.org/ 10 1145 284563 284566 。/ 。Robert Bruce Findler、John Clements、Cormac Flanagan、Matthew Flatt、Shriram Krishnamurthi、Paul Steckler 和 Matthias Felleisen。2002。DrScheme：Scheme 编程环境。《函数式编程杂志》12, 2（2002年3月），159–182。https://doi.org/10.1017/S0956796801004208[15] Vincent Gramoli、Michael Charleston、Bryn Jeffries、Irena Koprinska、Martin McGrane、Alex Radu、Anastasios Viglas 和 Kalina Yacef。2016。挖掘计算机科学教育中的自动评分数据。载于澳大拉西亚计算机科学周多会议论文集（堪培拉，澳大利亚）*(ACSW '16)*。ACM，纽约，纽约州，美国，文章 1, 10 10 1145 2843043 2843070 页。https://doi.org/ 。/ 。T. Hailikari、N. Katajavuori 和 H. Asikainen。。理解拖延：一个学习技能课程的案例。教育社会心理学24, 2 (2021), 589 606 10 1007 11218 021 09621 2 – 。https://doi.org/ 。/s - - - Aliya Hameer 和 Brigitte Pientka。。通过自动评分教授函数式编程艺术（经验报告）。*ACM*《编程语言》论文集3, ICFP (2019), 115:1–115:15。[18] Aliya Hameer 和 Brigitte Pientka。2019。使用自动评分教授函数式编程艺术（经验报告）。*Proc. ACM Program. Lang.* 3, 115 2019 15 10 1145 3341719 ICFP, 文章（7月），页码。https://doi.org/ 。/ John A Hartigan 和 Manchek A Wong。。算法 AS：K均值聚类算法。英国皇家统计学会期刊。系列 *C*（应用统计）28，1（1979），100–108。[20] Roya Hosseini、Arto Vihavainen 和 Peter Brusilovsky。2014。在 Java 编程课程中探索问题求解路径。载于*PPIG*。编程心理学兴趣小组，9。[21] C. D. Hundhausen、D. M. Olivares 和 A. S. Carter。2017。面向计算教育的基于集成开发环境的学习分析。*ACM*《计算教育汇刊》 17, 3, 1 26 10 1145 3105759 – 。https://doi.org/ 。/ Irshad Hussain 和 Sarwat Sultan。。大学生拖延行为分析。《社会与行为科学进展》5（2010），1897–1904。https://doi.org/ 10 1016 2010 07 385 。/j.sbspro。。Essi Lahtinen。。新手程序员的分类：一项聚类分析研究。载于第*19*届编程心理学兴趣小组年度研讨会论文集，约恩苏，芬兰。32–41。[24] Franzis Preckel、Anastasiya A Lipnevich、Sandra Schneider 和 Richard D Roberts。2011。昼夜节律类型、认知能力与学业成就：一项荟萃分析研究。《学习与个体差异》21，5（2011），483–492。[25] Steven M. Reppert 和 David R. Weaver。2002。哺乳动物昼夜节律时序的协调。《自然》 418, 6901, 935 941 10 1038 00965 – 。https://doi.org/ 。/nature Mark Sherman、Sarita Bassil、Derrell Lipman、Nat Tuck 和 Fred Martin。2013。自动评分对计算机入门课程的影响。*J. Comput. Sci. Coll.* 28, 6 69 75 =2460156 2460171 （6月），– 。http://dl.acm.org/citation.cfm?id 。Herbert Simon。。理性选择与环境结构。《心理学评论》。63，2（1956），129–138。[28] Ethel Tshukudu 和 Quintin Cutts。2020。理解学习新编程语言的学生的概念迁移。载于*2020*年*ACM*国际计算教育研究会议论文集（线上会议，新西兰）（*ICER '20*）。计算机协会，纽约，纽约州，美国， 227 237 10 1145 3372782 3406270 – 。https://doi.org/ 。/ 。Joseph B. Wiggins、Fahmid M. Fahid、Andrew Emerson、Madeline Hinckle、Andy Smith、Kristy Elizabeth Boyer、Bradford Mott、Eric Wiebe 和 James Lester。2021。在智能积木式编程环境中探索新手程序员的提示请求。第*52*届*ACM*计算机科学教育技术研讨会论文集。https://doi.org/ 10 1145 3408877 3432538 。/ 。Chris Wilcox。。自动化在本科计算机科学教育中的作用。载于第*46*届*ACM*计算机科学教育技术研讨会论文集。ACM，90–95。[31] Albina Zavgorodniaia、Raj Shrestha、Juho Leinonen、Arto Hellas 和 John Edwards。2021。早晨还是傍晚？对计算机科学导论课程学生昼夜节律的考察。*2021 IEEE/ACM*第*43*届国际软件工程会议：软件工程教育与培训（*ICSE-SEET*）（2021）。https://doi.org/10.1109/icse-seet52601.2021.00036