|  |  |  |
| --- | --- | --- |
|  |  | 学号 1906010519 |
|  |  | 年级 2019级 |
| 说明: 河海大学  本科毕业论文 | | |
| 基于Rust的高并发场景技术探索与应用 | | |

|  |  |
| --- | --- |
| **专 业** | 计算机科学与技术 |
| **姓 名** | 李云汉 |
| **指导老师** | 陆佳民 |
| **评 阅 人** |  |

|  |
| --- |
| **2023年4月** |
| **中国 南京** |

|  |
| --- |
| BACHELOR'S DEGREE THESIS  OF HOHAI UNIVERSITY |
| Exploration and Implementation of High Concurrency Technique Based on Rust Programming Language |

|  |  |
| --- | --- |
| College : | College of Computer and Information |
| Major : | Computer Science |
| Name : | Li Yunhan |
| Directed by : | Lu Jiamin |

**NANJING CHINA**

郑 重 声 明

本人呈交的毕业论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本设计（论文）的研究成果不包含他人享有著作权的内容。对本设计（论文）所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本设计（论文）的知识产权归属于培养单位。

本人签名： 日期：

摘要

ABSTRACT

Fluvial river processes evolve over time in response to the constant interaction between sediment and the water column. If vegetation is present within the water column, the change in turbulence characteristics will impact the movement of sediment, in particular the settling velocity. In this paper, the influence of vegetation on the settling velocities of sediment particles is studied experimentally. The non-submerged vegetation friction factor in steady uniform flow is considered by under different flume discharge quantities. The main outcomes can be summarized as follows:…

**Key words:** sediment；rigid vegetation；settling velocity；turbulence characterize

目录

[摘要 I](#_Toc131254612)

[ABSTRACT II](#_Toc131254613)

[目录 III](#_Toc131254614)

[第1章 绪论 1](#_Toc131254615)

[第2章 Rust及其基本特性概述 1](#_Toc131254616)

[第3章 蓄水池算法简述及其基于Java的实现 1](#_Toc131254617)

[3.1 蓄水池算法简述 1](#_Toc131254618)

[3.2 基于Java的蓄水池算法实现 1](#_Toc131254619)

[第4章 基于Rust的蓄水池算法实现 1](#_Toc131254620)

1. 绪论
   1. 研究背景和意义

Rust作为一门新兴的高级语言，以其接近C/C++运行效率、基于强大的编译器静态检查能力的内存管理等特点获得了业内广泛的关注，并同时取得了来自学院派和工业界的认可：Linux kernel在6.1版本引入了对于Rust编程语言的支持；同时包括Microsoft在内的各大厂商也在着手对Rust进行技术储备。

作为一门系统级语言，高效的并发处理是Rust的主要目标之一。Rust原生支持OS线程级的并发编程，并且支持async/await并发模型。

* 1. 研究现状

Rust是一门新兴语言，学习Rust时的主要知识来源是GitHub在线开源官方文档。如Rust语言圣经（中文） (RustCourse, 2023)、The Rust Programming Language（英文） (RustBook, 2023)、Asynchronous Programming in Rust（英文） (async-book, 2023)等。

对于Rust的研究，能查到的文献非常有限。但是主要可以分为三类：

1. 关于Rust语言正确性的形式化验证。
2. 经验主义/实证研究。
3. 应用Rust进行实际开发

Boqin Qin等 (Boqin Qin, 2020)、秦伯钦等 (秦伯钦, 2021)以及Zemin Yu等 (Zemin Yu, 2019)则对于Rust在并行编程、unsafe代码段等实践中的安全问题进行了基于代码分析的经验主义/实证研究。其中秦伯钦等还基于研究成果；牛聚川等提出并实现了一种融合模糊测试和形式化验证的Rust测试工具 (牛聚川, 2022)；王烁程等提出了基于Rust Async/Await的基于协程的飞地内部调度方案，并设计了调度机制 (王烁程, 2021)、

* 1. 研究成果

本文为对Rust并发开发的实证研究。在第2章中笔者将对Rust的基本特性进行介绍。

为了从语言特性、抽象层次、并发性能等层面对比新兴的Rust语言和传统的Java语言，本论文中选择了“蓄水池取样算法”这样一个高并发的算法，并分别用Java语言的线程库（第3章）和Rust语言OS线程级的并发模型（第4章）对其进行了实现，并据此比较二者在各层面上的差异。

1. Rust及其语言特性概述
   1. Rust的历史

Rust是一种由Graydon Hoare设计，Mozilla支持的的高级编程语言。由Mozilla于2010年正式发布，于2015年发布了第一个稳定版（Rust 1.0 stable release），并在2016年及以后趋于稳定。Rust的发展经历了多次重要迭代，包括引入了特性（trait）、类（class）、析构函数（destructor）等语言特性，以及Non-Lexical Lifetime（即NLL优化，这种优化专门用于找到某个引用在作用域结束前就不再被使用的代码位置）等编译器优化。

疫情期间，Mozilla出于成本控制等考虑，解散了Servo（一个基于Rust的浏览器引擎）的开发小组。小组成员中很多都是Rust社区的核心成员。这对于Rust的发展产生了很大冲击。但令人欣慰的是，2021年Rust基金会由AWS、Huawei、Google、Microsoft、和 Mozilla等企业宣布注资成立。@todo

本文中使用的Rust版本是rustc 1.67.1。

* 1. Rust的语言特性

尽管Rust与C/C++、Java具有很强的相似性，但其优秀的安全性和抽象能力来自于Rust的一些设计良好的语言特性。

* + 1. 内存安全：所有权与生命周期

Rust非常擅长保证内存安全：这种语言依赖强大的编译器静态检查来保证对于所有的引用都指向内存中的合法资源。这种由编译器负责的静态检查不需要在程序运行时消耗资源。无需Java、Go等语言的垃圾回收（GC，Garbage Collection），也不需要像C/C++一样手动管理内存。

Rust使用所有权来管理内存。简单来说，内存（堆空间）中的每一个资源（值，value）都为一个变量（variable）所拥有，这个变量称为该内存资源的所有者（owner of the value）。一个内存资源只能有一个所有者。当所有者离开作用域时，其所拥有的内存资源被释放（drop）。

可以说，Rust通过编译器静态检查的方式，强制程序员使用了RAII （Resource Acquisition Is Initialization，资源获取就是初始化）的概念。

除了所有权，Rust还有生命周期（Lifetime）的概念。想要使用某个内存资源，除了获得其所有权，还可以对其进行借用，获得指向这个资源的一个不可变或可变的引用。Rust编译器会在编译期间检查对内存资源的引用是否会超出拥有这个资源的变量的生命周期，以避免在使用引用时，其对应的资源被错误地提前释放了。

* + 1. 抽象特性：Rust的面向对象特性与函数式编程

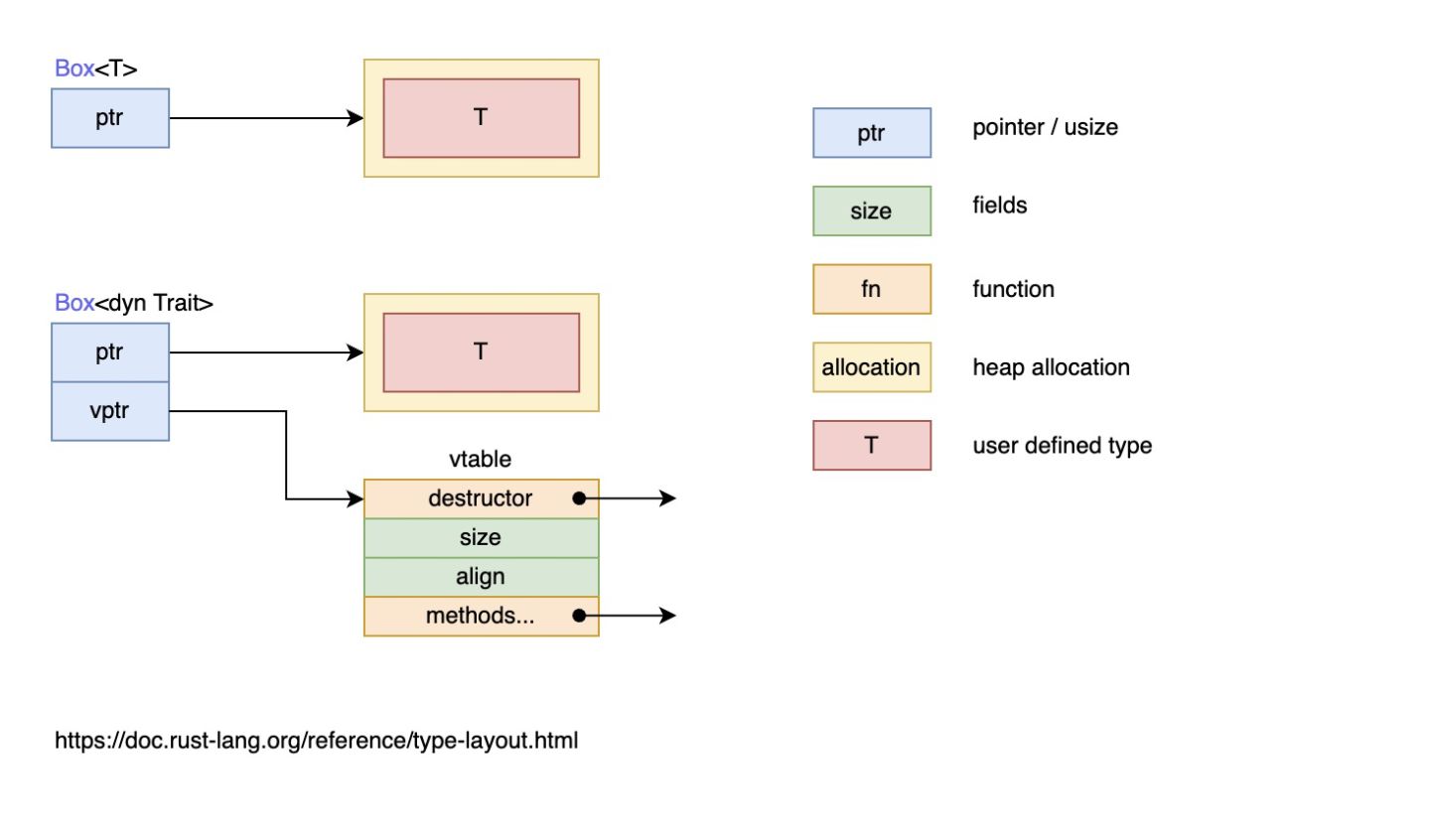
尽管Rust是一个面向对象语言，Rust支持在结构体（struct）中封装一些外部不可见的成员变量和成员函数，但Rust并没有像C/C++、Java那样支持类（class）之间的继承。Rust通过为类添加特征（trait）的方式支持面向对象编程中的继承和多态。

特征类似于Java中的接口（interface），特征定义了一组可以被共享的行为，只要实现了特征，你就能使用这组行为。换句话说，拥有某个特征的类需要实现这个特征所定义的方法。特征中也可以定义具有默认实现的方法。通过特征（trait），Rust实现了代码的复用，间接地支持了继承。尽管这种方式是比较啰嗦的，因为特征虽然支持多态（默认实现的方法中可以调用没有默认实现的方法），但接口默认实现的方法使用起来没有继承方便，需要手动为每个拥有特征的类实现特征所定义的方法。

需要指出的是，一些程序员使用Rust的Deref特征来间接地模拟了继承功能。但一些人认为其为一种反模式（背离Rust设计初衷的做法），是一种误用编译器特性的错误做法 (simonsan, 2023)。其问题包括但不限于：只支持单继承、不能继承父类的特征等。

此外，特征有标记的功能（marker trait）。举例来说，Rust多线程编程中，编译器仅允许在线程之间传递拥有Sync特征的类型的引用；仅允许在线程之间传递拥有Send特征的类型的所有权。

此外，Rust类型系统还支持特征对象，这是一种动态分发（dynamic dispatch）方式的多态。举例来说，函数可以返回一个指向了实现了某种特征的类型（的指针，Box<dyn Trait>）。



图片 2‑1 由智能指针包装的特征对象Box<dyn Trait>

除此以外，Rust还支持函数式编程。在写Rust代码的时候，程序员可以使用闭包（closure，与Java中的lambda表达式类似）、模式匹配、迭代器等语言特性来简化自己的代码。

* + 1. 并发特性：基于OS线程的并发编程、异步编程与可扩展并发编程

事实上，Rust的并发特性大部分都不是来源于Rust语言本身，而是来自Rust的标准库和第三方库。Rust标准库支持基于OS线程的并发编程，程序员可以创建操作系统级的线程来完成他们的任务。同时，Rust通过语言特性（Trait等）、标准库和第三方库（tokio等）支持async/await并发模型。

本文需要实现的蓄水池算法是一种长时间运行的CPU密集型任务，需要CPU在某个线程持续运行；相比之下，async并发模型可以在一个任务处于IO或者阻塞状态时及时切换到另一个任务，任务切换的开销远小于线程切换的开销，async并发模型更适合服务器场景下的高并发。

事实上, async 底层也是基于线程实现，但是它基于线程封装了一个运行时，可以将多个任务映射到少量线程上，然后将线程切换变成了任务切换，后者仅仅是内存中的访问，因此要高效的多。 (RustCourse, 2023)

同时，Rust的异步编程暂时存在着一些兼容性问题，并且因为async和社区开发的运行时依旧在不断革新而有着更高的维护成本。当然，Rust也有成熟的异步编程的第三方库：tokio。

基于以上事实，本文选用OS多线程的并发模型实现蓄水池算法。@todo

此外，Rust语言本身支持可扩展的并发编程。在std::marker中有两个标记特征（marker trait）：Send和Sync。Rust编译器的类型检查和特征系统仅允许在线程之间传递拥有Send特征的类型的所有权；仅允许在多个线程中通过引用访问拥有Sync特征的类型。在这两个标记特征的基础上，我们可以扩展标准库和第三方库所提供的并发功能。

1. 蓄水池算法简述及其基于Java的实现
   1. 串行蓄水池算法简述

蓄水池算法是一种抽样算法，它可以用于在数据流（只能以实现定好的顺序依次读取的元素序列，假设共有N个，N是一个很大的数字）中抽取k个元素，并确保数据流中的每一个元素可以以相同的概率被抽取到。其时间复杂度为，空间复杂度为。

简单说，串行蓄水池算法维护一个长度为k的向量（vector）。对于每一个数据流中的元素，以一定的概率决定其是否被采样。倘若决定采样，则用这个元素随即替换向量中的一个元素。

* + 1. 形式化的算法过程

假设元素序列的规模为N，需要采样的数量为k。

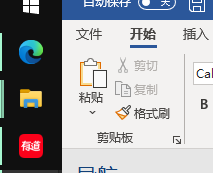
1. 首先创建一个可以容纳k个元素的向量
2. 对于序列中的第个元素，全部放入向量中
3. 对于序列中的第个元素开始，以的概率来决定该元素是否被替换到数组中（数组中的元素被替换的概率是相同的，即每个元素被替换的概率均为）。
4. 遍历完所有元素后，数组中剩下的元素即为算法结果，即采样结果。

形式化的算法证明

* 1. 基于Java的蓄水池算法实现

1. 基于Rust的蓄水池算法实现

课程论文-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。图片 4‑3 你好图片

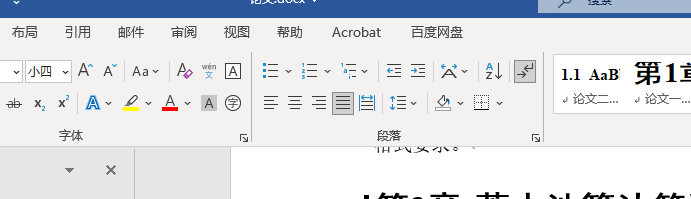


图片 4‑2

学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。

利用Rust

课程论文-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。



图片 4‑3 你好图片

课程论文-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。-学位论文-Thesis-格式要求。 (RustCourse, 2023)

1. 参考文献

[1]黄煜东. 基于rust编程语言的i2p匿名通信设计[D].华中师范大学,2018.