问题的出色介绍;现有方法的全面覆盖/比较。该解决方案非常详细，并对基本原理进行了深入解释。所有的实验设置都经过精心设计，细节极佳。定量结果有很好的演示，有深入的讨论，而建议的解决方案至少在一个指标上超过了基线。彻底和建设性的思考和结论。结果是完全可重复和可验证的。

请使用 Time New Roman 或 Arial 字体类型，10pt，单列，单行距来编写报告。预期长度应在 2 到 6 页之间（包括参考文献），A4 大小。更长的报告是可能的，但您需要有充分的理由。

1.3 成果提交要求

最终成果主要为一份系统性的报告，用于记录你的解决方案及其有效性的证明。你不需要提交代码，但应将代码存储在一个公开的代码仓库中，并在报告中附上链接。

报告应至少包含以下几个部分（你可以自由更改标题名称或添加子章节）：

引言（Introduction）：本节应提供你所解决问题的背景信息，并说明你选择该问题的原因。

相关工作（Related Work）：本节应讨论已有的解决该问题的方法，并对它们的优缺点进行分析。引用请采用计算机科学领域常见的学术引用格式（如 Chicago 风格），详情可参考：https://pitt.libguides.com/citationhelp/ieee。

解决方案（Solution）：在这里你可以详细阐述你提出的解决方案。虽然鼓励你提出原创思路，但如果你选择复现已有工作的某个方法，以测试它是否适用于你所选的系统或项目，也完全可以。但你需要用自己的语言详细描述该方法的设计思路，包括背后的设计动机。你也可以设计一个图形用户界面（GUI），但最重要的还是能解决智能软件工程问题的核心算法。

实验设置（Setup）：本节需要描述你的实验设置和流程，比如所选的系统/项目简介、参数设置、评估指标、对比基线方法，以及使用了哪些统计分析方法等。

实验结果（Experiments）：你应在这里对你的方案进行定量评估，并理想情况下与基线方法进行对比。需要使用图表来展示结果，并采用课程中介绍的统计检验方法。你还应考虑在多样化场景下进行评估，例如在不同的系统/项目、指标和/或目标下。作为最低要求，你需要涵盖至少一个系统/项目、一个评估指标和一个优化目标（如适用）。结果讨论部分需要描述你从实验观察中得到的发现。

反思（Reflection）：本节讨论你提出方法的局限性，并思考有哪些改进空间。

结论（Conclusion）：根据实验结果，总结你的研究结论。

实验材料链接（Artifact）：在此处提供你代码仓库的链接（如 GitHub 或 Zenodo），其中应包括源码和原始实验数据。

参考文献（References）：任何引用的论文/工作都应在此处按照学术格式正确引用。

Abstract

本文基于ISE lab 3的Configuration Performance Tuning 问题提供的baseline 和数据集设计并部署了6个intelligent turning methods , 并对这些方法进行了测试。最终得到了两个优于baseline 的一个有模型方法（Linear FLASH）和一个无模型方法（Linear BestConfig ），并且提出了一种改进的。本文还探讨了不同前期采样方法的优势和劣势，以及不同模型在一些系统下的理论表现，同时验证了两种假设。

Introduction：

在系统复杂度越来越高的今天，无论是人类还是算法都已经越来越难以理解日益增多的选项对他们的软件所造成的影响，例如其中数据库的设置调优是一个np问题（1），尝试不同的配置的代价也十分高昂。但是，与此同时，软件的设置对于软件的性能有着至关重要的影响。因此我们需要一个能在尽可能少的尝试下尽可能自动算出最优配置的方法，同时在完成lab3时作者基于自身修改软件配置的经验想到了一个可能比随机抽样更好的初始采样。出于智能软件调优是非常重要的领域和作者想验证自己的想法的原因，作者选择了这个主题。

Related Work：

在配置性能调优领域，调优的方法主要被分为Model free tuning，Model base tuning，以及一些其他的新的调优方法，例如Cross environment tuning，Cost aware tuning，Code sensitive tuning。受限于模拟数据集的限制，我们将不讨论除了有模型调优和无模型调优之外的方法，因为我们的数据集只有一个目标维度，也没有使用来自不同环境的信息，代码等，这导致那些新的调优方法将会失效。下面我将会讨论现有的方法和它们在这项任务中的局限性。

1. Model free tuning

无模型优化不依赖于预测模型，而是直接评估目标系统或数据集上的每个采样配置。相比于有模型调优这些方法非常准确，因为性能测量是真实的。但是，缺点是评估通常需要大量的计算，导致高昂的计算成本。下面是几种经典的无模型调优

**1.1Random Search**: 它盲目地从整个空间对配置进行采样，尽管结果往往很差，但是只要你够幸运，它就是最快最好的方法。而且这个方法完全不受维度灾难，复杂度指数上升，局部最优，过拟合等问题的影响。本文使用**Random Search作为实验的baseline。**

**1.2BestConfig:** 由 Zhu 等人 [2] 提出，通过引入两阶段本地搜索框架（DDS- Divide & Diverge Sampling ，RBS: Recursive Bound & Sea ）进行搜索。DDS 将配置空间划分为子空间以确保覆盖范围和多样性，而 RBS 则根据先前的性能递归地将搜索范围缩小到高潜力区域。在现实生活中BestConfig 往往被用于单目标优化任务。作者认为**BestConfig十分适合这个任务，因此尝试了这个模型。**

1.3 **Meta Multi-Objectivization（MMO）:**由Chen 和 Li [3] 提出。MMO 在搜索过程中引入了一个辅助目标，以重塑配置空间，有助于避免较差的局部最优值并鼓励更好的探索。但是作者没有在数据集中找到合适的辅助目标，而且加入太多的先验经验并不是一个好主意，因此本文没有使用MMO

1.4 NSGA-II : 这个方法被Singh et al [4] 用于 ORM 配置优化。这个方法将配置编码为二进制字符串，并使用遗传算法将它们代代发展。由于其基于遗传算法的性质，它经常收敛缓慢且计算成本高，因此本文没有使用这种方法。

1.5 Bayesian Optimization：这是一种非常经典的方法，用于优化具有未知解析形式的昂贵黑盒函数。在软件配置调优中，BO 维护一个高斯过程 （GP）来近似性能函数。然后，它使用采集功能来平衡采样。这个算法有很多问题，但是作为上面算法的基础，Bayesian Optimization被作者所实现。

1. Model base tuning

基于模型的优化使用预测模型来估计配置性能，从而减少所需的昂贵测量次数。这些方法在测量实际性能和更新学习模型以指导未来采样之间交替。下面是几种经典的有模型调优

2.1 FLASH，FLASH ，由 Nair 等人 [5] 提出，。它由“精确的搜索是不必要的作为“中心指导思想，使用 CART 决策树代替传统的GP作为代理模型（避免了维度更高和假定“smoothness”的问题），并使用贝叶斯优化根据采集函数选择下一个配置，同时应用BAZZ算法用于多目标的调优（本文的FLASH不使用BAZZ算法，因为目标只有一个）。作者认为FLASH**十分适合这个任务，因此尝试了这个模型。**

2.2 BOCA ,由Chen 等人 [6]将贝叶斯优化应用于这个领域，这个模型使用了Random Forest替换了传统的GP，同时具有降低维度的特点，非常适合于二进制的系统设置tuning。不过本文的系统大多都不符合二进制的特点，因此本文没有测试这个方法。

解决方案（Solution）：

在本文中作者设计了一个方法（lab3\_forier.py）部署了3个已有的方法（lab3\_bayesian.py，lab3\_bestconfig.py，lab3\_flash.py）改进了两个方法（lab3\_bestconfig\_linear.py , lab3\_flash.py）提出了一种基于选项独立的改进的具有线性复杂度的DDS，验证了两种假设。

1.两种假设：

假设一：

假设二：选项是

在这里你可以详细阐述你提出的解决方案。虽然鼓励你提出原创思路，但如果你选择复现已有工作的某个方法，以测试它是否适用于你所选的系统或项目，也完全可以。但你需要用自己的语言详细描述该方法的设计思路，包括背后的设计动机。

实验设置（Setup）：本节需要描述你的实验设置和流程，比如所选的系统/项目简介、参数设置、评估指标、对比基线方法，以及使用了哪些统计分析方法等。

实验结果（Experiments）：你应在这里对你的方案进行定量评估，并理想情况下与基线方法进行对比。需要使用图表来展示结果，并采用课程中介绍的统计检验方法。你还应考虑在多样化场景下进行评估，例如在不同的系统/项目、指标和/或目标下。作为最低要求，你需要涵盖至少一个系统/项目、一个评估指标和一个优化目标（如适用）。结果讨论部分需要描述你从实验观察中得到的发现。

反思（Reflection）：本节讨论你提出方法的局限性，并思考有哪些改进空间。

结论（Conclusion）：根据实验结果，总结你的研究结论。

实验材料链接（Artifact）：

参考文献（References）：任何引用的论文/工作都应在此处按照学术格式正确引用。

[1]： SULLIVAN D G, SELTZER M I, PFEFFER A. Using probabilistic reasoning to automate software

tuning[C]//Proceedings of the joint international conference on Measurement and modeling of

computer systems. ACM, 2004: 404-

[2]

[3]