# 项目概述

* 1. 项目背景及研究（/开发）内容

近年来，以自然语言生成为核心的大语言模型技术正在人工智能领域掀起热潮，并持续向更多的领域穿透其影响力。以ChatGPT为代表的大模型，已经在软件工程的多项活动中展示出其通过自然交互方式给人提供一定程度帮助的能力和潜力，正在发展成为一种基于自然交互的人机协同软件开发与演化工具。大模型能够根据自然语言需求直接生成预测性程序代码，尽管在一定程度上可以提高软件开发效率，但是其所生成的预测性代码缺失可信性判断。如不针对其生成的预测性代码进行可信性判断和确认，即加以使用，将会带来不可估量的巨大风险与损失。在软件工程领域中，逐步精化的程序设计方法是由图灵奖获得者Niklaus Wirth教授于上世纪70年代提出的经典程序设计方法，其基本思想为采用自顶向下的设计策略，通过连续精化层次结构的程序细节来实现编程。通过逐步分解、精化自然语言描述的功能直至形成程序设计语言的语句，大模型打开了从自然语言需求到预测性设计与实现的通道。在此基础上，如何基于主动逐步精化的方式驾驭大模型工具，从自然语言需求出发，构建程序详细设计原型，并进一步生成程序具体实现是本项目关注的重要研究目标。在这一目标下，本项目将伪代码视为自然语言与程序代码的中间桥梁，借助伪代码实现主动逐步精化的编程新范式，因此需要解决以下核心问题：

1. **如何利用伪代码设计高效的人机交互形式，降低开发者分析与理解预测性代码的难度与复杂性，支撑其修改和确认代码？**
2. **如何设计合适的伪代码粒度，以便在保持伪代码可读性的同时，提供足够的细节以支持高效的代码生成？**
3. **如何选择伪代码精化基本块，实现在伪代码层面的精化内容的自动选择与对应代码生成？**
4. **如何检测代码与相应伪代码之间的不一致性，从而降低程序员修正预测性代码生成错误的难度？**

* 1. 项目技术方法、路线及目标

**探索主动逐步精化式IDE的设计与实现技术。主要包括：**

1. **以人为主逐步精化的代码生成框架设计**

**探索一种以人为主逐步精化的代码生成框架，该框架能够支持开发者从高层次的自然语言需求描述逐步细化到具体的实现代码，帮助开发者理解逻辑，并在代码生成过程中提供反馈，确保生成的代码符合预期的功能需求。**

1. **问题描述：**

**如何充分利用大语言模型对于自然语言的理解能力，有效引导模型生成预测性程序代码同时保障代码的正确性？**

1. **基本思路：**

**自然语言**

**描述**

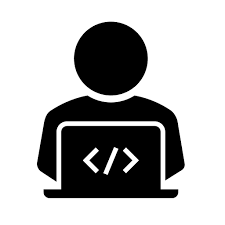
**伪代码**



**程序代码**

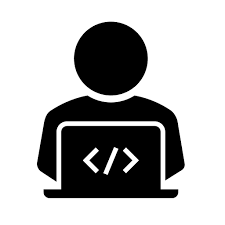
**审核与**

**纠错**



**审核与**

**纠错**



**图1代码生成框架设计**

**我们的研究思路如图1所示，主要的研究内容是构建以人为主逐步精化的代码生成框架。将以自然语言描述为主的软件开发需求输入给大语言模型，经由伪代码阶段逐层精化，生成以伪代码形式表征的代码实现的详细设计。在此基础上，通过合适的伪代码粒度与表征方式，实现伪代码模块划分，逐步引导大语言模型完成特定功能模块的代码生成。上述过程中，伪代码与代码的生成均由用户审核，以便逐层对生成的代码进行逻辑和功能验证。**

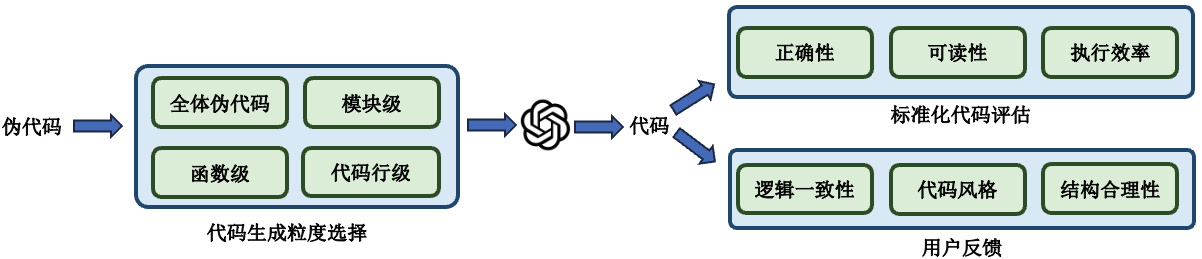
1. **技术关键点：**
   * + 1. **基于分层伪代码的逐步精化生成机制。通过明确的伪代码语法规范与粒度划分，将整体功能拆解为可单独处理的逻辑模块，分阶段生成并验证，从而确保代码逻辑的正确性与可控性。**
       2. **人机协作的审核与反馈环节嵌入。提供直观的交互接口和标注工具，让开发者对生成的中间产物（伪代码或初步实现代码）进行评估和修改，并将反馈信息用于后续生成优化，有效提升生成结果的精确性与可用性。**
2. **预期目标：**

**旨在通过逐步精化框架，提升代码生成准确性，于此同时，提升开发者的控制权，从而提高生成代码的实用性与可靠性。**

1. **伪代码粒度与机制探索**

**软件开发人员手动选择伪代码内容，生成对应代码，因此需探索用户选择伪代码的不同粒度（行、功能体、全部伪代码等不同方式），是否会影响模型生成代码质量，该研究需要设计定量、定性的实证研究方案进行系统化分析。**

1. **问题描述：探究选择不同粒度的伪代码输入（如行级、功能模块级、完整伪代码）对于生成代码质量的影响？**
2. **基本思路：**

****

**图2伪代码粒度与机制探索**

**我们的研究思路如图2所示，将伪代码划分为不同的粒度层次，包括行级、功能体级（例如单个函数或模块）、全体伪代码等，邀请软件开发人员对同一任务选择不同粒度的伪代码，使用大型语言模型（如GPT-4）进行代码生成，并对结果进行对比与分析。具体而言，在评估方面我们主要使用定量评估和定性评估的评估方法，在定量评估中引入标准化的代码质量评估指标，其中包括代码的正确性、可读性、执行效率等，从而帮助我们进行量化对比。于此同时我们进一步设计人工评估的方式，收集软件开发人员的用户反馈信息，分析不同伪代码粒度生成的代码在逻辑一致性、代码风格、结构合理性等主观感受。**

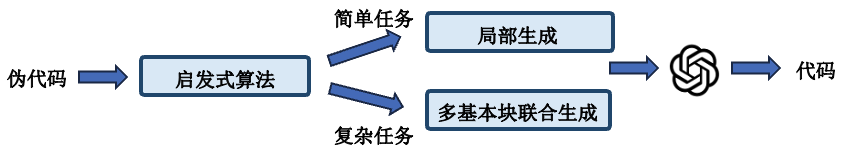
1. **技术关键点：**
   * + 1. **伪代码多粒度划分与选择机制。将伪代码划分为行级、功能体级、全体伪代码等不同粒度层次，允许开发者在同一任务中灵活选择伪代码输入给大语言模型。**
       2. **定量与定性结合的多维评估方法。综合采用标准化的代码质量指标（正确性、可读性、执行效率等）与开发者主观评价（逻辑一致性、结构合理性、代码风格等）来对比不同粒度伪代码生成效果。**
2. **预期目标：**

**我们旨在验证不同粒度选择的伪代码内容在代码生成中的适用性和质量表现，从而为软件开发提供针对不同代码生成任务的不同粒度选择指导，为未来的模型设计提供伪代码粒度优化的依据，以提升生成代码的准确性和效率。**

1. **基于伪代码基本块的代码精化与生成机制**

**在上述伪代码粒度与机制探索中，我们旨在研究合适的伪代码粒度选择，以此形成适合于程序员理解与反馈的逐层精化机制。进一步而言，在合适的伪代码粒度基础上，我们将进一步探索如何以伪代码基本块为单位进行代码精化与生成，从而提升代码质量。**

1. **问题描述：探索如何以伪代码基本块为单位进行代码精化与生成？**
2. **基本思路：**

****

**图3代码精化与生成机制**

**我们的研究思路如图3所示，打算结合任务复杂度，设计启发式算法从而实现以下目标：根据任务难度，自动划分基本块范围，利用模型进行代码生成。对于简单任务，算法能够推荐正确的基本块内容用于局部生成，而对于复杂任务，算法能够自动推荐多个基本块内容，用以联合生成，从而保障模型生成代码的逻辑完整性。通过启发式算法，根据任务难度，向软件开发人员推荐合适的基本块范围，从而帮助模型更好地生成代码。**

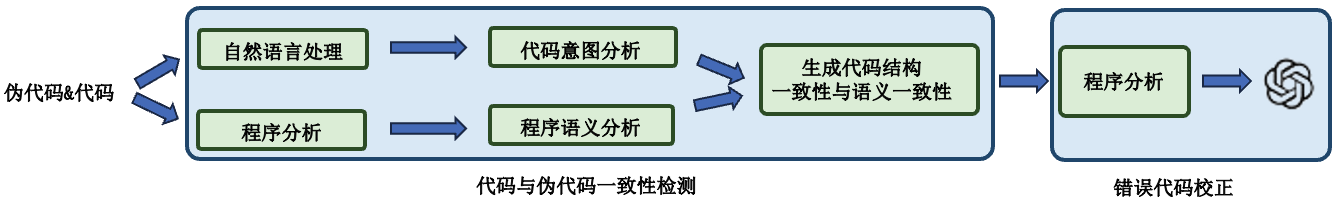
1. **技术关键点：**
   * + 1. **基于任务复杂度的启发式基本块划分与推荐。设计启发式算法，根据任务难度和复杂度特征动态划分基本块范围，为模型生成提供明确的输入内容。**
       2. **联合生成机制与逻辑完整性保障。针对复杂任务的多基本块划分，通过联合生成机制协调各基本块内容，确保代码生成的逻辑一致性和语义完整性。**
2. **预期目标：**

**我们旨在研究自动化选择伪代码精化基本块进行代码生成，从而在一定程度上减少软件开发人员的参与，提高软件开发效率。**

1. **代码与伪代码一致性检测与校正方法探索**

**探索用于检测和修正伪代码与生成代码不一致方法。该方法应能够分析并修正伪代码与生成代码之前的差异，确保两者在功能实现上保持一致，从而提高代码的准确性和可靠性。**

1. **问题描述：如何检测与修正模型生成代码与伪代码的不一致性？**
2. **基本思路：**

****

**图4代码与伪代码一致性检测与校正**

**我们的研究思路如图4所示，拟打算利用自然语言处理技术和程序分析技术，分析伪代码的意图和生成代码的程序语义，从而自动化地识别两者在功能上的差异，以实现一致性检测的目的。检测内容主要包括结构一致性与语义一致性。在此基础上，进一步提出自动化校正技术，利用代码生成模型和程序分析技术进行修正，进一步引入迭代生成与校正机制，修复模型生成生代码与伪代码的不一致性。**

1. **技术关键点：**
   * + 1. **基于自然语言处理与程序分析的伪代码-实现代码一致性检测。利用自然语言处理技术解析伪代码意图，并结合程序分析手段识别生成代码的语义，自动检测二者在功能和逻辑上的不一致性。**
       2. **生成错误代码自动化校正机制。在检测到不一致后，结合代码生成模型和程序分析技术进行自动化修正，并通过迭代生成与校正机制减少偏差，从而保证模型生成代码与定义的伪代码的一致性。**
2. **预期目标：**

**我们旨在辅助代码生成能遵循伪代码描述的意图和逻辑，通过一致性检测与修正方法，增强用户对生成代码的信任度，减少开发人员的二次修正工作量。**