基于 Frama-C 的 Select 算法 C 源码的形式化验证

姜雨函
辽宁师范大学 数学学院，辽宁 大连 116081
DOI:10.61369/RSTD.2026020002

摘　　要： 本文针对 Select 算法 C 源码实现的逻辑正确性和运行可靠性提出了一种形式化验证方法，该方法通过 Frama-C 形式化验证工具及其配套插件的相互协作，并采用 ACSL 规范语言对 Select 算法的 C 源码需满足的核心属性进行标注，旨在对该算法形成清晰的形式化表达，最后基于抽象解释与定理证明技术完成严谨的验证推理，以确保算法实现与预设规范的一致性 . 通过此方法可以证明 Select 算法的 C 源码的逻辑正确性以及运行可靠性，从而解决该算法在常规手工测试中因穷举情况覆盖不完整而造成的隐性缺陷逃逸和效率低下问题。

关 键 词： Frama-C；Select 算法；形式化验证

Formal Verification of the C Source Code of the Select Algorithm Based on Frama-C

Jiang Yuhan
School of Mathematics, Liaoning Normal University, Dalian, Liaoning 116081

Abstract: This paper proposes a formal verification method for the logical correctness and operational reliability of the C source code implementation of the Select algorithm. This method utilizes the collaboration of the Frama-C formal verification tool and its accompanying plugins, and employs the ACSL specification language to annotate the core attributes that the C source code of the Select algorithm must satisfy. The aim is to form a clear formal expression for the algorithm. Finally, rigorous verification reasoning is completed based on abstract interpretation and theorem proving techniques to ensure the consistency between the algorithm implementation and the preset specifications. Through this method, the logical correctness and operational reliability of the C source code of the Select algorithm can be proved, thereby addressing the hidden defect escape and low efficiency problems caused by incomplete exhaustive coverage in conventional manual testing of the algorithm.

Keywords: Frama-C; Select Algorithm; formal verification

引言

由于 Select 算法具有能在$O(n)$时间复杂度内定位无序集合中第 k 小数据的根本特性，使之能够有效地贴近工程领域对高效数据处理的要求，在机器学习与数据分析等场景中，Select 算法可以快速获取关键统计数据并节省预处理时间 [1]。正是由于 Select 算法的在核心工程领域的重要应用，所以保证其实现逻辑、运行的可靠性是必要的。为验证 Select 算法在所有合法输入情况下的一致行为特性，可以利用 Frama-C 工具及其相关插件，通过向程序中添加 Frama-C 支持的 ACSL 规范语言进行注释，利用抽象解释、演绎验证、定理证明等技术，对 Select 算法的各个属性加以形式化验证，得到 Select 算法预设的形式化契约与实现完全相符的结论，并对 Select 算法实现进行了严谨的正确性证明，使得 Select 算法在工程领域的正确性和可靠性，避免了因常规手工测试输入空间覆盖不全面造成的隐含缺陷漏检和测试效率低下问题 [2]。

一、预备知识

（一）Select 算法

Select 算法是一般性的选第 k 小问题的以算法，需要用到分治策略 [3]，主要思想为以 S 中的某个元素 m^* 作为标准将 S 划分成两个子数组 S_1 和 S_2，其中 S_1 中的元素比 m^* 小，S_2 中的元素比 m^* 大。设 S_1 中的元素数是 $|S_1|$，如果 $k \le |S_1|$，那么原问题就归约为在 S_1 中找第 k 小的子问题 . 如果 $k = |S_1|+1$，那么 m^* 就是所要找的第 k 小元素 . 如果 $k > |S_1|+1$，那么原问题就归约为在 S_2 中找第 k' 小的子问题，其中 $k' = k - |S_1| - 1$ 该算法的核心在于通过递归调用确定基准元素 m^*，首先将集合 S 分组划分为每组 5 个元素的子集合，共得到 $[n/5]$ 个分组，接着求解每个子集合的中位数，并将这 $[n/5]$ 个中位数纳入集合 M 中，最终通过在 M 中递归调用选择算法得到的中位数，即为目标基准元素 m^*. 算法的伪码描述如下：

Select算法

输入：n个数的数组S，正整数k

输出：S中的第k小元素

1.将S划分成5个一组，共$[n/5]$个组

2.每组找一个中位数，把这些中位数放到集合M中

3. $m^* \leftarrow Select(M,[|M|/2])$

//选M的中位数m^*，将S中的数划分成A,B,C,D四个集合

4.把A和D中的每个元素与m^*比较，小的构成S_1,大的构成S_2；

5. $S_1 \leftarrow S_1 \cup C$； $S_2 \leftarrow S_2 \cup B$；

6.if $k=|S_1|+1$ then 输出m^*

7.else if $k \le |S_1|$

then Select(S_1,k)

else Select$(S_2,k-|S_1|-1)$

（二）Frama-C工具

Frama-C是一款聚焦于C语言源代码的开源静态程序分析工具，通过Frama-C支持的ACSL规范语言编写注释，可将待验证C程序的属性规约进行清晰的形式化表达。Frama-C采用模块化结构，核心为内核组件，该内核负责承担信息管理职责，提供基础工具集，并且统筹协调各项功能的执行和实现。Frama-C的具体功能依靠不同的插件实现，所有插件信息暂存于内核，再由内核同步至所有插件以防漏查，各插件通过协作共享数据，最终为验证的程序生成全面的分析报告[4]。

1.基于抽象解释的插件Value和EVA

Value和EVA是基于抽象解释技术的两个关键插件。其中，Value可快速扫描程序代码中的运行时错误，包括数组越界、除零错误、指针错误及变量未初始化使用等错误。该插件分析效率较高，但因采用抽象解释技术，分析结果存在误报现象。相较于Value插件，EVA插件还可支撑自定义抽象域、增量分析、多线程代码分析。虽核心仍为抽象解释技术，但其功能更强大。

2.基于定理证明的插件WP、Qed、RTE

WP、Qed、RTE是基于定理证明技术的三个关键插件。其中WP作为Frama-C的核心定理证明插件，其原理是调用自动定理证明器，通过数学逻辑推演验证代码与规范的一致性。由于基于数学逻辑推演，其结论具有绝对可靠性，适用于核心代码验证，如操作系统内核及医疗设备控制代码，但复杂代码在自动证明器受阻时需人工干预。Qed插件辅助WP处理简单命题，以提升整体分析效率。RTE插件可针对运行时错误进行更精细的检查，能够排除程序分析结果的误报。

（三）ACSL规范语言

ACSL是“ANSI/ISOC规范语言”的缩写，作为Frama-C静态程序分析框架所实现的行为接口规范语言，其核心目标是通过特定格式的注释嵌入C源码，以形式化方式规定C源码的行为特性[5]，该类注释不干扰程序的正常编译流程，仅在静态程序分析阶段发挥作用，为分析工具提供精确的行为约束依据。ACSL标注采用/*@…*/或//@…作为界定符，依据作用域可分为全局标注、函数标注和语句标注三类，分别对应不同层级的行为规范需求。

在ACSL规范体系中，requires、ensures、invariant、assert与assigns为核心关键字，各关键字遵循特定语法格式并承担明确语义功能：其一，/*@ requires 逻辑表达式；*/为关键字requires的标准语法，其语义是定义函数前置条件，即函数被调用前必须满足的逻辑约束，若该条件不成立，则函数行为不具备确定性；其二，/*@ ensures 逻辑表达式；*/为关键字ensures的标准语法，用于定义函数的后置条件，指函数执行完毕返回后必须满足的逻辑断言，用以表达函数执行的预期结果；其三，/*@ invariant 逻辑表达式；*/为关键字invariant的标准语法，功能是定义循环不变式，该表达式需在循环每次迭代的起始与终止时刻均保持成立，是验证循环正确性与终止性的关键依据。

（四）形式化验证

形式化方法思想于20世纪50年代首次出现，其以严谨的数学和逻辑理论为基石，得以对软件和硬件系统的验证有着重要作用。1967年，Floyd[6]首次提出使用形式化验证的方法证明程序的可靠性，历经多年，最终取得了众多理论与实践得重大进展，在采用此方法验证分析程序时，通过Floyd-Hoare逻辑，可将待验证的程序替换为抽象模型，再进行逻辑归纳演绎，证明程序是否满足程序的规范契约。形式化验证是通过数学方法保证C源码逻辑的正确性和运行的可靠性，在形式化验证流程中，开发者需要给出待验证函数的规范语言，规范语言作为关键环节，承担着对系统功能需求及属性约束条件的精确化描述任务[7]。可以将自然语言描述中存在的模糊性需求转化为具有可验证性的数学表达式，为后续验证过程提供明确的判定依据。

二、Select算法的形式化验证设计

本文针对Select算法的形式化验证，聚焦于其四大核心特性：其一为该算法C源码实现功能的正确性，也就是指输入满足预设的前置条件，则其输出结果必定满足后置条件所描述的对应逻辑约束[8]；其二为该算法C源码实现的内存安全性，即需要确保该算法在运行过程中无数组越界访问、无空指针引用、无内存泄漏等可能导致运行异常的内存访问问题[9]；其三为该算法C源码实现的参数有效性，即指算法在运行过程的内部函数调用中，不存在程序内部函数参数传入不合规的问题，排除因函数参数非法而引发的逻辑错误；其四为该算法C源码实现的执行终止性，即该算法存在对于所有符合算法前置条件的输入，其运行执行步骤必定有限终止的执行限制，否则将存在执行无限循环等非终止性运行问题[10]。

（一）基本谓词与逻辑定义

设计核心谓词与逻辑函数，用于描述数组有效性、元素计数、第k小关系等基础逻辑，为后续规约设计奠定基础：

1.数组区间有效性谓词in_range：刻画数组指针的可访问区间[0,n)的有效性，约束所有下标访问的合法性，作为所有函数的前置条件，配合RTE插件避免数组越界等未定义行为；

2.计数逻辑函数count_le与count_lt：以递归方式定义数组中“小于等于v”和“严格小于v”的元素个数，避免采用“排

序后第 k 位”的定义，更适配 MoM 算法的分区逻辑，且利于 Frama-C WP 插件的展开与归纳推理；

3. 第 k 小关系谓词 is_kth：通过“存在性 + 计数上下界”精确定义“第 k 小”，即存在数组元素等于目标值 v，且数组中小于等于 v 的元素个数≥ k、严格小于 v 的元素个数≤ k-1，作为 select_kth 函数的核心后置条件。

（二）公理化引理族设计

围绕 MoM 算法的分区 - 提升核心逻辑，设计 4 类公理化引理族，将算法的结构性事实转化为定理证明器可消费的代数等式与不等式，提升验证自动化率：

1.CountProps 引理族：涵盖计数的传递性、前缀稳定性、左右分解等性质，用于合并阈值大小关系、保证计数在数组前缀 / 后缀扩展下的稳定性，为分区后的计数推理提供支撑；

2.KthLift 引理族：形式化“子段第 k' 小提升为全局第 k 小”的三种场景（左分支、右分支、枢轴落点），是 select_kth 函数递归验证的核心推理依据；

3.PartitionGlue 与 SplitClosure 引 理 族：PartitionGlue 提供分区后计数的“粘合”事实，减少调用点的重复推导；SplitClosure 用拓展式重述分区结论，便于自动化匹配量词域；

4.CountFixes 引理族：针对分割后计数易出现的建模错误，补充两条狭义引理，分别刻画“右切片无小于阈值的元素”和“左前缀全小于阈值”的计数值，避免计数误用。

（三）函数的合同和循环不变量设计

针对 4 个核心函数，结合其 C 程序实现逻辑，设计精准的 ACSL 函数合同与循环不变量，约束函数行为、保证循环终止性：

1. 函数合同：每个函数均定义前置条件（requires）、后置条件（ensures）与赋值域（assigns），明确输入约束、输出性质及函数对内存的修改范围。例如，select_kth 函数的前置条件约束数组有效、k 值在合法范围、数组长度≤ 50000（与临时数组上界匹配），后置条件用 is_kth 谓词声明输出为第 k 小元素，decreases 语句保证递归终止；

2. 循环不变量：针对函数中的循环结构（如 group_median 的插入排序循环、partition3 的两段分区循环），设计包含索引边界、元素关系、赋值域的不变量，并通过 loop variant 语句定义终止度量，确保循环终止，同时支撑 RTE 检查与功能正确性验证。例如，partition3 的第一段循环不变量维持“[0,store) 全小于枢轴、[store,i) 全不小于枢轴”的分区结构，第二段循环不变量确保等于枢轴的元素被正确搬移至指定位置。

三、结论

本文以选择算法为研究对象，基于 Frama-C 平台与 ACSL 规约语言，完成了从程序设计、规约设计到验证实施、优化的全流程形式化验证。通过分层设计 ACSL 规约（基本谓词、公理化引理、函数合同、循环不变量），结合 Frama-C WP 插件与 RTE 插件的分阶段验证，有效证明了选择算法的功能正确性与运行安全性，规避了常见的运行时错误。实验结果表明，该验证方法具有可复现性与实用性，验证优化策略能够显著提升验证效率，兼顾工程可维护性。本文的研究成果，为复杂选择算法及同类 C 程序的形式化验证提供了实践参考，对提升安全关键领域 C 程序的可靠性具有重要意义。

参考文献

[1] 侯潇逸，田杰斌，赵硕，等 . 基于多头注意力机制的相似源码漏洞检测仿真 [J]. 计算机仿真，2025,42(11):242-247.

[2] 马佳鑫，张铮，刘鹏，等 . 基于反向数据流传播的 SQL 语句随机化 [J]. 计算机应用研究，2025,42(10):3106-3113.

[3] 王帅，黄晨，江云松，等 .AFL-VTest: 航天嵌入式软件模糊测试框架 [J]. 计算机科学，2025,52(12):9-17.

[4] 崔少轩，喻垚慎 . 静态程序分析过程中形式化验证工具 Frama-C 的应用 [J]. 计算技术与自动化，2019,38(01):114-117.

[5] 包晟宏，姚有健，李小丫，等 . 集成式 PU 学习方法 PUEVD 及其在软件源码漏洞检测中的应用 [J]. 计算机科学，2025,52(S1):865-873.

[6] 张元，熊风光，杨晓文，等 . 处理机调度实验系统开发及实验案例设计 [J]. 工业和信息化教育，2025,(03):84-89.

[7] 付俊仪，张倩颖，王国辉，等 . TrustZone 中断隔离机制的形式化验证 [J]. 小型微型计算机系统，2023, 44(09):2105-2112.

[8] 郭肖旺，赵德政 . 基于 NuSMV 的 LD 和 ST 语言形式化验证研究与实现 [J]. 电子技术应用，2022, 48(12):94-99.

[9] 钱汉伟，王承毅 . 操作系统形式化验证综述 [J]. 河海大学学报 (自然科学版), 2021, 49(05):473-481.

[10] 罗娟，王燕芩 . 形式化方法应用于计算机联锁软件的安全验证研究 [J]. 铁路计算机应用，2016, 25(11):53-57.

