arXiv:2303.07455v1 [cs.NE] 13 Mar 2023

(1+1) 使用函数完备指令集的遗传编程能够以任意小的误差演化布尔合取与析取

Benjamin Doerr[1], Andrei Lissovoi[2], 与Pietro S. Oliveto[3]

[1]doerr@lix.polytechnique.fr，巴黎理工学院计算机科学实验室（LIX），´巴黎综合理工学院，巴黎理工学院，法国帕莱索[2]a.lissovoi@sheffield.ac.uk，谢菲尔德大学计算机科学系，英国[3]p.oliveto@sheffield.ac.uk，谢菲尔德大学计算机科学系，英国

摘要

近期研究已证明，只要配备所需的最少组件，简单的遗传编程系统就能高效演化由 n 个变量构成的合取。本文向前迈出重要一步，分析遗传编程系统在使用完整函数集时，演化由 n 个变量构成的布尔合取或析取的行为与性能；该函数集能够表示至多含有 n 个变量的任意布尔函数。首先，我们严格证明：遗传编程系统若使用完整真值表评估程序质量，并配备与运算符、或运算符及正文字，则能够以 $O(\ell n \log^2 n)$ 次迭代的期望运行时间演化出精确的目标函数，其中 $\ell \geq n$ 是任何被接受的树的大小上限。此外，我们证明，当使用可能输入的多项式规模样本来评估解的质量时，以概率 $1-O(\log^2(n)/n)$，能够演化出泛化误差为任意多项式级小量的合取或析取。若遗传编程使用与运算符、或运算符以及正文字和否定文字，该结果同样成立；此时系统能够表示任意由 n 个不同变量构成的布尔函数。为证明这些结果，我们提出了一种超乘性漂移定理：当期望进展仅略微超过与距最优解距离成线性关系时，该定理可给出显著更强的运行时间上界。

1 引言

遗传编程（GP）运用达尔文进化原理，进化出具备特定所需功能的计算机程序。由 Koza（1992年）开创的遗传编程方法最为流行、广为人知，它使用语法树表示程序，并利用遗传变异算子在由可用组件构成的程序空间中进行搜索，倾向于选择在各种可能输入下表现更好的程序。在这种设定下，通过比较程序在不同输入下的输出与目标函数的输出，来评估程序的质量。

尽管遗传编程已有许多成功应用的实例（例如，Koza，2010年；Liu 和 Shao，2013年；Bartoli 等人，2014年；Moore 等人，2018年；Miranda 等人，2019年；Lynch 等人，2019年；Vu 等人，2019年），但我们对其行为和性能的理解仍然有限。

现有少数关于遗传编程的理论分析，沿用了传统进化算法（EA）在函数优化分析中极为成功的路径。此类分析最初考察的是简化的进化算法，例如（1+1）进化算法（Droste 等人，2002年），随后逐步扩展到对使用种群和交叉的现实进化算法进行分析（例如，Jansen 等人，2005年；Witt，2006年；Doerr 等人，2012a；Dang 等人，2018年；Huang 等人，2019年；Corus 和 Oliveto，2020年；Dang 等人，2021年；Sutton，2021年；Corus 等人，2021年；Zheng 等人，2022年）。类似地，研究者也考察了简化的遗传编程系统：这类系统使用一种名为 HVL-Prime 的基于树的变异算子（由 O'Reilly 和 Oppacher 于 1996年提出的分层变长变异算子的改编版）来进化单个程序。本文研究 RLS-GP（Durrett 等人，2011年）。与随机局部搜索算法（例如，Neumann 和 Wegener，2007年）类似，它在评估适应度前执行一次局部变异；不同于（$1+1$）GP，它可以在单次迭代中执行更多次局部变更，这类似于（$1+1$）进化算法。

早期研究分析了这些算法在进化不可执行树结构方面的行为和性能，而非进化计算机程序；在后者中，适应度函数用于估计候选解在输入训练集上的行为与目标程序行为的匹配程度（Durrett 等人，

2011年；K¨otzing 等人，2014；Doerr 等人，2020年；Lissovoi 和 Oliveto，2020年）。直到最近，人们才开始分析简单遗传编程系统在进化可执行函数方面的性能。研究已证明，RLS-GP 和 (1+1) GP 算法能够在期望多项式次数的迭代内进化出由 n 个变量构成的布尔合取（Mambrini 和 Oliveto，2016年；Lissovoi 和 Oliveto，2019年）。当使用完整真值表（即所有 2^n 个可能输入的集合）评估程序质量时，可以进化出与目标合取等价的程序。如果每次迭代都从完整真值表中均匀随机抽取多项式数量的输入来评估解的质量（即采用 Gathercole 和 Ross（1994）提出的动态子集选择来限制总计算开销），那么进化出的程序已被发现具有良好的泛化能力；也就是说，它们仅在任意小的、占所有 2^n 个可能输入的多项式比例的输入上出错。

虽然这些结果令人鼓舞，但所研究的遗传编程系统与实际使用的系统存在很大差异。具体而言，它们必须仅使用基本组件（即二元布尔与运算符和问题所需的变量）来演化一个简单的 n 元布尔合取。在实际应用中，所需的程序组件集合不一定能预先确定，因此遗传编程系统通常可以使用比严格必要范围更广的组件。因此，本文探讨的一个重要问题是：遗传编程系统能否应对这样的函数集，其中包含一些在简洁表达目标函数时应当避免使用，或应完全避免使用的元素。理想情况下，系统应至少配备一套完整的运算符，从中可以构造任意布尔函数。

本文通过分析 RLS-GP 在演化未知布尔函数时的行为和性能，在这一方向上迈出了重要一步。具体而言，我们考虑的目标函数为 AND$_n$（n 个变量的合取）或 OR$_n$（n 个变量的析取）。遗传编程系统可以使用二元合取运算符（即 AND）和析取运算符（即 OR）。将 AND$_n$ 或 OR$_n$ 作为目标函数，有助于我们理解混合使用合取运算符和析取运算符的候选解的质量。此外，我们还考虑一种情形：终端集包含所有 n 个变量及其取反形式。因此，这样的遗传编程系统在布尔域上是完整的，因为凭借其函数集和终端集，它可以表达关于 n 个变量的任意布尔函数。

这些更复杂的问题设定促使我们在 RLS-GP 系统中引入比仅使用 AND 算子演化合取所需的功能更复杂的特性，从而使遗传编程系统更接近实际应用中使用的系统。由于当前解中存在析取可能会降低变异算子生成行为更优程序的效果，我们对语法树的大小设定了上限。这样可以避免膨胀带来的问题（这是遗传编程系统中的常见问题：解的大小往往会增加，但解的质量却没有相应提升（Koza，1992年；Poli 等人，2008年））。

在设定树大小上限后，我们的理论分析表明，先前工作中使用的 HVL-Prime 变异算子（Durrett 等人，2011年；Lissovoi 和 Oliveto，2019年）每次对树执行插入、替换或删除一个节点的操作，可能会陷入局部最优。因此，采用传统 HVL-Prime 算子的 RLS-GP 的期望运行时间为无穷大。为避免这一问题，我们引入了一种更接近最常用子树变异的变异机制（Koza，1992年；Poli 等人，2008年）。具体而言，该机制允许删除算子一次删除整棵子树，而不再将其限制为只能删除单个叶节点及其直接父节点。

我们首先证明，采用上述修改的 RLS-GP 若使用完整真值表评估程序质量，并拒绝任何叶节点数超过 $\ell = (1+c)n$（其中 $c > 0$ 是常数）的树，就能够高效地处理扩展函数集以及终端集中的正文字。具体而言，我们证明，它在期望 $O(\ell n \log^2 n)$ 次迭代内即可进化出精确目标函数。尽管使用完整真值表评估程序质量需要关于变量数的指数时间，但我们出于两个主要原因考虑这种设定。首先，这种设定代表了遗传编程系统行为的最佳情形（也就是说，如果一个系统在能够使用可靠适应度函数的情况下仍无法找到最优解，那么它很可能也无法在适应度函数带有噪声时表现良好）。其次，确定性的适应度值在一定程度上简化了算法的行为，因而也简化了我们的分析。请注意，如果终端集中也包含否定文字，那么已有研究证明，即使只使用 AND 算子，标准 RLS-GP 也无法以压倒性概率在多项式时间内进化出 AND$_n$（Mambrini 和 Oliveto，2016年）。我们推测，对于采用扩展函数集和终端集的修改版 RLS-GP，情况也相同，并提供了实验证据支持这一推测。

另一方面，对于一种更贴近现实的情形，我们给出了一个积极的一般性结果：在每次迭代中，均从完整真值表中均匀随机抽取多项式规模的训练集。在实践中，可以利用关于待演化函数类的一些信息，来决定训练集使用哪些输入。例如，如果已知目标函数是由 n 个变量构成的合取，那么线性规模的紧凑训练集就足以高效地演化出精确解（Lissovoi 和 Oliveto，2019年）。然而，我们假设目标函数是一个未知的、由 n个变量的合取与析取构成的任意函数。我们的目标是估计在这种设定下 RLS-GP 所产生解的质量。

我们展 我们证明，以概率 $1 - O(\log^2(n)/n)$，RLS-GP 能够构造并返回一个合取（或 a

算法 1 树大小限制为 ℓ 的 RLS-GP 算法。

1: 初始化一棵空树 X
2: 对于 $t \leftarrow 1, 2, \cdots$ 执行
3: $X' \leftarrow$HVL-Prime(X)
4: 如果叶子计数(X') $\leq \ell$ 且 $f(X') \leq f(X)$ 则
5: $X \leftarrow X'$

析取），其多项式泛化误差可以任意小，并且只需对数级的迭代次数，即使终端集中包含否定文字也是如此。

为得出这些结果，我们提出了一个超乘性漂移定理。该定理利用的漂移比传统乘性漂移定理（Doerr 等人，2012b）所要求的线性漂移更强。作为随机搜索启发式算法分析方法体系的一项新贡献（Lehre 和 Oliveto，2018；Lengler，2020），该定理使得在存在强乘性漂移时，能够得出小得多的期望运行时间上界。

我们还通过实证研究补充了理论结果，证实了我们的理论直觉：出于膨胀控制的考虑，如果对树大小设定了限制，仅删除叶节点可能会陷入局部最优。此外，实验表明，尽管不限制树大小时算法能更快地演化出解，但大小限制会减少返回程序中预期存在的不需要的二元析取运算符的数量。

本研究的初步版本此前已发表于遗传与进化计算会议（Doerr 等人，2019）。在此版本中，我们还表明，所研究的遗传编程系统无需进一步修改，也能高效地演化出析取；并给出了更一般的结果：使用理想的函数集和终端集，使任意由不超过 n 个不同变量构成的布尔函数都能表示时，该算法同样高效。我们还扩展了实验分析，以补充这一更一般设定下的理论结果。

2 预备知识

本文将分析简单 RLS-GP 算法在 AND$_n$ 和 OR$_n$ 问题上的表现：前者直接分析，后者则通过观察这两个问题的搜索空间之间的对称性，指出可以推导出等价的结果。AND$_n$ 问题涉及在使用 $F = \{$AND、OR$\}$ 二元函数和 $L = \{x_1, \ldots, x_n\}$ 个输入变量的情况下，演化出所有 n输入变量的合取。当使用完整真值表评估程序质量时，适应度函数 $f(X)$统计候选解 X 的输出与ˆ目标函数 h(x)（即 AND$_n$(x) $= x_1 \wedge \ldots \wedge x_n$和 OR$_n$(x) $= x_1 \vee \ldots \vee x_n$，分别对应 AND$_n$ 和 OR$_n$问题）输出不同的真值赋值（或输入）的数量。正如 Lissovoi 和 Oliveto（2019）所观察到的，由 a 个不同变量构成的合取与 AND$_n$ 在 $2^{n-a} - 1$ 个真值赋值上不同。

我们将分析 RLS-GP 算法的性能。该算法会反复在当前解与通过应用树变异算子生成的子代之间选择较优者。除了考虑经典的 HVL-Prime 变异算子（它以相同的概率在当前解中插入、删除或替换一个叶节点）（Durrett 等人，2011年），我们还提出了一个略作修改的版本（带子树删除的 HVL-Prime），该版本能够在一次变异中删除任意子树。

我们观察到，当前解中存在析取可能会导致膨胀问题。每个“或”都会增加精确表示该合取所需的最少叶节点数，并削弱在其下方插入节点对整体程序语义的影响。此外，由于经典的 HVL-Prime 算子通过删除一个叶节点及其直接父节点来执行删除，因此可能难以移除树中位置较高的析取。为解决这一问题，我们为 RLS-GP 添加了一种简单的膨胀控制机制，使其拒绝包含超过 ℓ 个叶节点的树，如算法 1 所述。

在树大小限制 ℓ 下，应用 Durrett等人（2011年）提出的原始 HVL-Prime 变异算子可能会使 RLS-GP 陷入局部最优。

定理 1. 对于采用 HVL-Prime 的仅叶节点删除与替换子操作的 RLS-GP，以及任意 $\ell > 0$ 上的与$_n$ 或$_n$，其中 $F =$ { 为与、或} ，其期望优化时间为无穷大。

证明。RLS-GP 可能构造出包含 ℓ 个叶节点、且无法通过局部变异进一步改进的树。对于与$_n$，考虑这样一棵树：先创建一个由 $\ell/2$ x_1 个叶节点组成的析取，然后将每个 x_1 叶节点转换为一个 $x_1 \wedge x_2$ 子树。最终树中的任何叶节点都无法删除或替换，否则适应度会下降；由于树大小限制，也不会接受任何插入，因此 RLS-GP 无法

算法 2 对树 X 执行子树删除的 HVL-Prime。

1：输入：一棵树 X、可用字面量集合 L、可用二元函数集合 F。
2：从 $op \in$ {INS、DEL、SUB}、 $l \in L$、 $f \in F$ 中均匀随机选择
3：如果 X 是空树，则
4: 将 l 设为 X 的根节点。
5: 否则，如果 $op =$ 是 INS，则
6: 均匀随机选择一个节点 $x \in X$
7: 将 x 替换为 f，把 f 的子节点设为 x 和 l，顺序均匀随机选择。
8: 否则，如果 $op =$ 为删除，则 ▷ 修改后的（子树）删除
9: 均匀随机选择一个节点 $x \in X$
10: 将 X 中 x的父节点替换为 X中 x的兄弟节点
11: 否则如果 $op =$ 替换，则
12: 均匀随机选择一个叶节点 $x \in X$
13: 将 x 替换为 l。
14: 返回修改后的树 X

达到 th 达到最优解。由于这棵树可以以非零概率构造出来，构造它的期望时间 ct
the o P最优 根据全期望定律，解是无限的。
对于或$_n$，当初始合取中的每个 $\ell/2$ x_1 叶节点都转换
成一个 x_2 子树。 □

为避免这一问题，我们按照算法 2 所述修改了 HVL-Prime 的删除操作，使其能够删除子树。与原始 HVL-Prime 算子的唯一区别是：原先第 9 行只允许选择叶节点进行删除，现在则可以选择任意节点进行删除。

我们用“采样误差”指某个解在某次迭代中的适应度值，用“泛化误差”指某个解在从所有 2^n 个可能输入中均匀随机选取的输入上出错的概率。当使用完整真值表评估程序质量时，解的采样误差始终恰好是其泛化误差的 2^n 倍。使用完整真值表时，遗传编程系统的目标是构造一个在语义上等价于目标函数的解，即使采样误差（以及泛化误差）达到 0。

由于当 n 较大时，评估所有 2^n 个可能输入在计算上不可行，我们也分析了 RLS-GP 在根据从所有可能输入中均匀随机选取的 $s \in$ poly(n) 个输入评估解的质量时的行为。每次迭代都会重新选取一组 s 个输入；此时， $f(X)$（即采样误差）指在所选的 s 个输入中， X 与目标函数输出不同的输入数量。因此，采样误差是一个随机变量，其期望恰好是解的泛化误差的 s 倍。我们将在下文的引理 1 中界定采样误差偏离其期望的概率。当使用规模为多项式的训练集评估程序质量时，遗传编程系统的目标是构造一个泛化误差较低的解。对于与$_n$以及大多数其他非平凡问题，只要 s 相对于问题规模仍为多项式，我们就不指望遗传编程系统能达到泛化误差为 0，除非问题的适应度景观已得到充分理解，并且使用了针对该问题的训练集（Lissovoi 和 Oliveto，2019年）。我们假设情况并非如此，目标是找到一个具有任意小多项式泛化误差的解。请注意，本文始终使用以下记号：$\mathbb{N} := \{0, 1, 2, \dots\}$、lg($n$) 和 ln($n$) 分别表示 n 的以 2 为底的对数和自然对数，而 log n 用于渐近界。

引理 1。设 $s \in$ poly(n) 为遗传编程系统采样的输入数量， G 为某个解的泛化误差， X 为表示该解采样误差的随机变量。那么，对于任何至少为正数常数的 c，

$$|Gs - X| \leq \max\{c \lg n, Gs\}$$

其概率至少为 $1 - n^{-\Omega(c)}$。

证明。 X 是 s 个伯努利变量之和，每个变量取值为 1（否则为 0）的概率为 G，因此 $E[X] = Gs$。由于 X 和 Gs 均非负， $Gs - X \leq Gs$；我们只关注 X 显著大于其期望的情形，其概率可通过应用切尔诺夫界加以界定。

假设 $E[X] \geq (c/2)$ lg n；则，$\Pr[X \geq (1+1)E[X]] \leq e^{-E[X]/3} \leq n^{-\Omega(c)}$；因此，以至少 $1 - n^{-\Omega(c)}$ 的概率， $|Gs - X| < Gs$。否则，我们给出 $E[X] \leq \mu^+ = (c/2)$ lg n 的上界，并应用

切尔诺夫界，使用 μ^+ Doerr（2020年，定理 1.10.21），得到 $\Pr[X \geq (1+1)\mu^+] \leq e^{-\mu^+/3} = n^{-\Omega(c)}$; 因此，以至少 $1 - n^{-\Omega(c)}$ 的概率， $|Gs - X| \leq X \leq c\lg\ n$。 □

3 完整真值表

在本节中，我们将分析 RLS-GP 算法在 AND$_n$ 问题上的运行时间。该算法采用子树删除（即使用算法 2 作为变异算子的算法 1），并通过完整真值表评估解的质量，也就是说，在所有 2^n 种可能的输入上执行每个构造的程序。

定理 2. 在 AND$_n$或 OR$_n$ 问题上，采用 $F = \{AND, OR\}$、 $L = \{x_1, \ldots, x_n\}$ 和 $\ell \geq n$ 的 RLS-GP 的期望运行时间为 $E[T] = \Omega(n \log\ n)$。

证明. 不包含全部 n 个不同变量的树，不可能等价于 AND$_n$ 或 OR$_n$ 函数。根据标准的优惠券收集论证，需要 $\Omega(n\log\ n)$ 次插入或替换操作，才能将全部n 个不同变量插入树中。 □

下面的漂移定理要求：当与目标的距离为 d 时，期望进展的量级为 $\Omega(d\log\ d)$。这一假设比乘法漂移定理中所假设的线性（即 $\Omega(d)$）进展略强。尽管差异看似很小，但由此得到的到达目标所需期望时间界却大相径庭。对于初始距离 d_0，粗略来说，乘法漂移情形下为 $O(\log d_0)$，而在我们的超乘性漂移情形下为 $O(\log\log d_0)$。

定理 3（超乘性漂移定理）。设 $\gamma > 1$ 和 $\delta > 0$。设 $X_0, X_1, \ldots$ 为取值于 $\Omega = \{0\} \cup [1, \infty)$ 的随机变量。假设对于所有 $t \in \mathbb{N}$ 以及所有满足 $\Pr[X_t = x] > 0$ 的 $x \in \Omega \setminus \{0\}$，有

$$E[X_t - X_{t+1} \mid X_t = x] \geq (\log_\gamma(x) + 1)\delta x. \tag{1}$$

则零点的首次命中时间 $T = \min\{t \in \mathbb{N} \mid X_t = 0\}$ 满足

$$E[T \mid X_0] \leq \frac{3}{\delta} + \frac{2(2 + \log_2 \log_\gamma \max\{\gamma, X_0\}) \ln \gamma}{\delta}.$$

证明。对于所有 $k \in \mathbb{N}_{\geq 1}$，令 $T_k := \min\{t \in \mathbb{N} \mid X_t < \gamma^{2^{k-1}}\}$。我们首先证明

$$E[T_k - T_{k+1}] \leq \frac{1 + 2^k \ln \gamma}{(2^{k-1} + 1)\delta}$$

对所有 $k \geq 1$ 均成立。为此，我们将过程 Y_t 视为定义在所有 $t \in \mathbb{N}$ 上的过程，其定义为：若 $t \leq T_k - 1$，则 $Y_t = X_t$；否则为 $Y_t = 0$。根据定义， $T_k^Y := \min\{t \in \mathbb{N} \mid Y_t < \gamma^{2^{k-1}}\}$ 满足 $T_k^Y = T_k$。我们证明过程 (Y_t) 满足乘法漂移条件，

$$E[Y_t - Y_{t+1} \mid Y_t] \geq (2^{k-1} + 1)\delta Y_t.$$

若 $t \geq T_k$，则 $Y_t = 0$ 和 $Y_{t+1} = 0$ 均成立。因此，乘法漂移条件显然成立。在更有意思的情形下，即 $t < T_k$，我们有 $Y_t \geq \gamma^{2^{k-1}}$ 和 $Y_t = X_t$。由此，根据 $Y_{t+1} \leq X_{t+1}$ 和 (1)，我们得出

$$E[Y_t - Y_{t+1} \mid Y_t] \geq E[X_t - X_{t+1} \mid X_t] \geq (\log_\gamma(X_t) + 1)\delta X_t \geq (2^{k-1} + 1)Y_t,$$

再次表明乘法漂移条件成立。

令 $T^Y := \min\{t \in \mathbb{N} \mid Y_t = 0\}$。由于 $T^Y = T_k^Y = T_k$，且对所有 $t \geq T_{k+1}$ 都有 $Y_t \leq \gamma^{2^k}$，乘法漂移定理（Doerr 等人，2012b）给出 $E[T_k - T_{k+1}] = E[T^Y - T_{k+1}^Y] \leq \frac{1 + \ln \gamma^{2^k}(2^{k-1}+1)\delta}{\ln \gamma(2^{k-1}+1)\delta} = \frac{1+2^k}{\ln \gamma(2^{k-1}+1)\delta}$。通过简单应用乘法漂移定理，我们还可观察到 $E[T - T_1] \leq \frac{1 + \ln_\delta \gamma}{}$。

接下来，我们以初始值 X_0 为条件。假设对于某个 $k \in \mathbb{N}_{\geq 1}$，有 $X_0 \in [\gamma^{2^{k-1}}, \gamma^{2^k})$。那么 $T_{k+1} = 0$，因此 $T = \sum_{i=1}^{k}(T_i - T_{i+1}) + (T - T_1)$。我们计算

$$\begin{aligned} E[T] &= \sum_{i=1}^{k} E[T_i - T_{i+1}] + E[T - T_1] \leq \sum_{i=1}^{k} \frac{1 + 2^i \ln \gamma}{(2^{i-1} + 1)\delta} + \frac{1 + \ln \gamma}{\delta} \\ &\leq \frac{3}{\delta} + \frac{2(k+1)\ln \gamma}{\delta} \leq \frac{3}{\delta} + \frac{2(2 + \log_2 \log_\gamma X_0)\ln \gamma}{\delta}. \end{aligned}$$

对于 $X_0 < \gamma$，类似地，我们有 $E[T] \leq \frac{1 + \ln_\delta \gamma}{\ln(X_0 \leq 1+1)\ln_\delta \gamma}$。这就证明了该结论。 □

定理 3 的证明通过分段乘性漂移来估计超乘性漂移。我们更倾向于采用这种证明方法，因为它简单，而且通过使用带尾界的乘性漂移定理（Doerr 和 Goldberg，2013年），也可以得到超乘性漂移的尾界[1]。另一种方法可以将时间上界改善一个常数因子（同样，这并非我们在此关注的特点），即使用可变漂移（Mitavskiy 等人，2009年；Johannsen，2010年）。

我们使用超乘性漂移定理，证明在使用完整真值表作为训练集时，RLS-GP 的期望运行时间上界。首先，我们关注与n 问题，之后将说明，由于或n 问题与与n 问题具有对称性，或n也有相同的结果。我们先对树未满时所经历的迭代次数进行界定，也就是说，此时树尚未达到包含 ℓ 个叶节点的大小限制。

引理 2. 考虑 RLS-GP 在与n问题上的一次运行，其中 $F = \{AND, OR\}$、 $L = \{x_1, \ldots, x_n\}$，且树大小限制为 $\ell \geq n$。设 T为找到最优解之前的迭代次数， $T_0 \leq T$为其中父个体不是满树的迭代次数。那么， $E[T_0] = O(\ell n \log^2 n)$。

证明。为了界定 $E[T_0]$，我们将使用解的适应度作为势函数，并且仅考虑树未满时的迭代，从而应用定理 3。当树为满树时，我们则依靠 RLS-GP 算法的精英保留，不接受会增大势函数值的变异（即适应度值更差的子代）。因此，树未满时的 T_0 次迭代不必是连续的。

在一次迭代中，若初始树包含少于 ℓ 个叶节点，则可以插入一个新的叶节点 x_i，并将其父节点 AND 锚定在树的根节点上。我们将这种操作称为根与操作。一次迭代中执行带有固定变量 x_i 的根与操作的概率至少为 $\frac{1}{3} \cdot \frac{1}{2} \cdot \frac{1}{2\ell} \cdot \frac{1}{n} = \frac{1}{12\ell n}$。

我们计算此类修改带来的预期适应度增益。由于适应度不会变差，因此只需考察能够提升适应度的操作。还要回顾一下，适应度就是变量 x_1、……、 x_n 的赋值数量：对于这些赋值，树的计算结果与与运算 n 不同。

设 x_1、……、 x_n 是这样一种赋值。这意味着并非所有 x_i 都为真，因为 RLS-GP 生成的任何树，对全真赋值的计算结果都是正确的真值。假设变量 x_1、……、 x_n 中恰有 $k \geq 1$ 个为假，但我们的树解计算结果为真。那么，恰好有 k 个变量，只要对其中任意一个执行根与操作，程序在此赋值下的计算结果就会变为假（因此适应度会提升，因为正确输出是假）。执行这种变异的概率至少为 $\frac{k}{12\ell n}$。

对于任意 $1 \leq i \leq n$，恰有 $\binom{n}{i}$ 个输入满足恰有 i 个变量被设为 false，恰有 $\sum_{i=1}^{k-1} \binom{n}{i}$ 个输入满足少于 k 个变量被设为 false。因此，如果当前解的适应度至少为 $M_k = 2\sum_{i=1}^{k-1} \binom{n}{i}$，那么对适应度有贡献的输入中，至少有一半满足至少 k 个变量被设为 false。

只考虑这些因素带来的进展，对于 $x \geq M_k$，我们有：

$$E\left[f(X^t) - f(X^{t+1}) \mid f(X^t) = x\right] \geq \frac{1}{12\ell} \frac{k}{n} x. \tag{2}$$

由于当 n 足够大时，对于所有 $k \in [1..n]$，我们都有 $M_k \leq 2n^{k-1}$。这意味着对于所有 $x \in [1..2^n]$ 和所有 $t \in \mathbb{N}$，我们有

$$\begin{aligned} E\left[f(X^t) - f(X^{t+1}) \mid f(X^t) = x\right] &\geq \tfrac{1}{12\ell n} \left(\lfloor \log_n(x/2) \rfloor + 1\right) x \\ &\geq \tfrac{1}{36\ell n} \left(\log_n(x) + 1\right) x, \end{aligned}$$

其中，最后一个估计用到了 $n \geq 2$。因此，结合 $\gamma = n$ 和 $\delta = 1/36\ell n$，定理 3 给出

$$E[T] \leq 36\ell n(3 + 2(2 + \log_2 \log_n 2^n)) \ln n) = O(\ell n \log^2 n).$$

□

为了证明以下定理，我们将证明，以高概率，在任意 $t \in \Omega(\ell n \log^2 n)$ 次迭代中，至少有一个常数比例的迭代满足：父代解包含少于ℓ 个叶节点。直观地说，这意味着引理 2 的条件出现得足够频繁，不会影响渐近期望运行时间（即 $E[T] = O(E[T_0])$）。

定理 4. 考虑在 AND$_n$上运行 RLS-GP，使用 $F = \{AND, OR\}$、 $L = \{x_1, \ldots, x_n\}$，并将树大小限制设为 $\ell = (1 + c)n$。令 T为找到最优解之前的迭代次数。如果 $c = \Theta(1)$，那么$E[T] = O(\ell n \log^2 n)$。

[1]为此，我们注意到，带尾界的乘法漂移定理（Doerr 和 Goldberg，2013年）表明，阶段长度 $T_k - T_{k-1}$和 $T - T_1$ 在随机意义上受其期望加上一个几何分布随机变量所支配（例如，参见 Doerr（2019年））。因此， T超过其期望的偏差在随机意义上受若干相互独立的几何分布随机变量之和所支配，因而可以用此类随机变量之和的尾界来界定（Doerr，2020年，第 1.10.4 节）。由于本文不需要此类尾界，这里省略更多细节。

证明。根据引理 2，对于某个常数 $c^* > 0$，令 $T' = c^* \ell n \log^2 n$作为找到最优解之前树尚未满时的迭代次数期望上界$E[T_0]$ 。应用马尔可夫不等式可知，在至多 $2T'$ 次这样的迭代中找到最优解的概率至少为 $1/2$。我们将证明，如果 $\ell = (1+c)n$，那么对于任意常数 $c > 0$，以高概率在$(2+c')T'$次迭代内会发生 $2T'$ 次这样的迭代，其中 $c' > 0$ 相对于 n为常数。于是，定理陈述可由一个简单的等待时间论证得出：在每个长度为$(2+c')T'$次迭代的时段内，找到最优解的概率至少为$1/2 \cdot (1 - o(1)) = \Omega(1)$，因此找到最优解之前这类时段的期望数量至多为 $O(1)$，故期望运行时间至多为 $O(T') = O(\ell n \log^2 n)$次迭代。

现在我们将证明，在任意 $N \in \Omega(\ell n \log^2 n)$ 次迭代中，以高概率，对于某个常数$c'' > 0$，总计至少 $c''N$ 个叶节点的删除会被接受。由于每次迭代至多使树中的叶节点数量增加 1，因此在任意 $(1+c'')N$ 次迭代中，以高概率至少会有 $c''N$ 次迭代使树不满。由于 $T' \in \Omega(\ell n \log^2 n)$，在 $(2+c')T'$ 次迭代中，以高概率会出现 $2T'$ 次树不满的迭代，其中 $c' = 2/c'' = \Omega(1)$。

考虑一棵恰好有 ℓ 个叶节点的树 X。设 $L_A(X)$ 是一组叶节点，它们与 X 的根之间仅由 AND 节点相连；将这组叶节点中包含某个变量且该变量仅在该集合的节点上出现一次的所有叶节点称为必要的。若 X 非最优，则 X 中至多有 $n-1$ 个叶节点是必要的，且至少有 $\ell - (n-1)$ 个叶节点是非必要的。所有非必要节点都可以直接删除（对于 $L_A(X)$ 中变量的冗余副本），或者间接删除（删除其任一 OR 祖先节点上的分支）。

因此，可以通过对树中的至少一个节点执行 HVL-Prime 删除子操作，删除每个非必要的叶节点。对于某些非必要的叶节点，可能需要删除更大的子树，才能在不对适应度造成负面影响的情况下移除该叶节点。此类子树删除需要等待更长时间（必须选择子树的根进行删除，而不是选择子树中的众多叶节点之一），但变异过程中删除的叶节点数量也会相应增加，从而抵消这一等待时间。我们注意到，树包含 $2\ell - 1$ 个节点，因此对于 $\ell \geq (1+c)n$ 和任意 $c > 0$，一次 HVL-Prime 变异在期望上至少会使树中的叶节点数量减少

$$\frac{1}{3}\frac{\ell - (n-1)}{2\ell - 1} \geq \frac{\ell - n}{6\ell} \geq \frac{c}{6+6c} \geq \delta \in \Omega(1),$$

其中 $\delta > 0$ 是一个正的常数，因为 $c \in \Omega(1)$。

令 $X_1, \ldots, X_N$ 表示树处于满状态时，每次迭代中一次被接受的变异所删除的叶节点数量，并令 $X = \sum_{i=1}^{N} X_i$。此外，定义序列 $Z_0, \ldots, Z_N$，其中 $Z_0 := 0$ 且 $Z_i := Z_{i-1} + X_i - \delta$；显然，$Z_N - Z_0 = Z_N = X - \delta N$。我们将证明 $Z_N > -\delta N/2$ （因此 $X > \delta N/2 \in \Omega(N)$）以高概率成立。

由于 $E[Z_i \mid Z_1, \ldots Z_{i-1}] = Z_{i-1} + E[X_i \mid Z_1, \ldots Z_{i-1}] - \delta \geq Z_{i-1}$，序列 $Z_0, \ldots, Z_N$ 是一个次鞅，并且 $c_i := |Z_i - Z_{i-1}| \leq \ell$。因此，对 $N \in \Omega(\ell n \log^2 n)$ 和 $t = \delta N/2$ 应用 Azuma-Hoeffding 不等式，

$$\Pr[Z_N - Z_0 \leq -t] \leq \exp\left(\frac{-t^2}{2\sum_{i=1}^{N} c_i^2}\right) \leq \exp\left(\frac{-\delta^2 N}{8\ell^2}\right) \leq n^{-\Omega(\log n)}$$

因为对于 $\ell = (1+c)n$，有 $N/\ell^2 = \Omega(n\ell \log^2 n/\ell^2) = \Omega(\log^2 n)$，其中 c 是一个常数。

因此，存在一个常数 $c'' > 0$，使得在树为满树的 $N \in \Omega(n\ell \log^2 n)$ 次迭代期间，至少删除 $\delta N/2 = c''N$ 个叶节点的操作以高概率被接受；因此，在 $(2/c'')N$ 次迭代期间，以高概率至少有 $2N$ 次迭代发生在树未满时。根据引理 2，令每 $N = T' = c^* n \ell \log^2 n$ 次迭代为一组，即完成证明：在 $\Theta(T')$ 次迭代中，至少有 $\Omega(T')$ 次会发生在树未满时，从而可以应用马尔可夫不等式和等待时间论证，得到期望运行时间的上界。 □

此外，我们证明，使用相同函数集和终端集的 RLS-GP 算法，在使用完整真值表评估解的适应度时，能够高效地进化出由 n 个变量构成的析取。定理 5 对这一结果进行了形式化，并采用与定理 4 相同的方法证明，利用了与问题的搜索空间之间的对称性。

定理 5。考虑在 OR_n 上运行 RLS-GP，使用 $F = \{AND, OR\}$、$L = \{x_1, \ldots, x_n\}$，且树大小限制为 $\ell = (1+c)n$。令 T 为找到最优解前的迭代次数。若 $c = \Theta(1)$，则$E[T] = O(\ell n \log^2 n)$。

证明。我们注意到，与$_n$ 和或$_n$ 目标函数的输出向量之间存在对称性：与$_n$ 仅在所有 n 输入变量都为真时返回真，而或$_n$ 仅在所有 n 输入变量都为假时返回假。同样，在或$_n$目标函数上，由 a 个不同变量构成的析取会在 $2^{n-a} - 1$ 个输入上出错。

通过考察在树顶端插入析取以找到最优解所需的时间，引理 2 同样适用于或n 问题。注意，对于任意 $1 \leq i \leq n$，恰有$\binom{n}{i}$ 个输入中恰好有 i 个变量被设为真，恰有$\sum_{i=1}^{k-1}\binom{n}{i}$ 个输入中少于 k 个变量被设为真。因此，如果当前解的适应度至少为$M_k = 2\sum_{i=1}^{k-1}\binom{n}{i}$，那么对适应度有贡献的输入中至少有一半包含至少 k 个被设为真的变量，从而通过插入其中一个 k变量，误差至少可以减半。

定理 4 的证明经过调整后也适用于或n 问题，只需修改关键叶节点的定义：令 L_A（X）表示一组叶节点，它们与 X 的根之间仅连接着或节点；若该集合中的叶节点包含一个仅在该集合的节点中出现一次的变量，则称其为必要的。随后，可以通过移除非必要节点来缩减完整树：对于L_A（X）中变量的冗余副本，可直接移除；对于其他情况，则可删除它们任一与节点祖先相连的分支。

因此，定理 4 的证明中推导出的运行时间上界同样适用于或n 问题。 □

如果通过添加所有正输入变量的否定来扩展终端集，那么带有 $F = \{AND, OR\}$的遗传编程便能够表示任意布尔函数（只需构造一棵表示其析取范式或合取范式的树，并将所有否定一直下推到单个变量上）。对于带有n 的与问题，且使用$F = \{AND\}$时，Lissovoi 和 Oliveto（2019）已证明，这种终端集扩展会使遗传编程在使用完整训练集时效率低下。在这种情况下，任何包含矛盾的解（例如$x_1 \wedge \overline{x}_1$）都会获得次优适应度（仅在全真输入上出错）。这会使遗传编程无法获得适应度信号，除非通过随机机会构造出最优解：将所有正终端插入树中，并从树中移除所有负终端的副本。由于所有未移除最后一个剩余矛盾的变异都会被接受，因此其期望时间至少为指数级。

我们猜想，对于使用 $F = \{AND, OR\}$ 和子树删除的 RLS-GP，在与问题和或问题上也会有类似结果，因为可以构造出在与问题中包含矛盾、或在或问题中包含重言式的树，而这些树只会在单个输入上出错。然而，Lissovoi 和 Oliveto（2019）使用的负漂移分析证明技巧并不容易应用于子树变异算子，因为搜索空间中可能以高概率发生大幅跳跃。因此，我们将这一猜想留待未来工作，并在下一节重点讨论一个积极结果：在使用多项式规模训练集的现实情形下.....

4 多项式规模的训练集

虽然上一节给出了演化出包含所有 n 个变量的合取或析取所需迭代次数的多项式界限，但每次迭代都要通过将程序输出与目标函数在全部 2^n 种可能输入上的输出进行比较来计算解的质量，这需要指数级的计算量。因此，这种方法只有在演化规模适中的布尔函数时才具备计算可行性。

本节中，我们考察 RLS-GP 算法在每次迭代中仅使用多项式计算量时的行为。为此，每次迭代都从所有可能输入的集合中均匀随机采样多项式数量的输入（即“训练集”），并通过评估父节点解、子代解和目标函数在这些输入上的输出，比较解的质量。Lissovoi 和 Oliveto（2019）此前研究过这种设定，并证明 RLS-GP 和使用 $F = \{AND\}$ 的 (1+1) GP 能够构造一个包含 $O(\log n)$ 个不同变量的解；该解能够拟合一个随机生成的、多项式规模的训练集，且期望用时为 $O(\log n)$。

对于下面的主要理论结果，我们选择让 RLS-GP 在训练集上的采样误差低于对数级接受阈值时终止并返回一个解。这样可以有效防止 RLS-GP 进入搜索空间中某个区域，在该区域中，它用于评估程序质量的机制会受到过多噪声的影响。这会略微降低解的期望质量，但仍能保证泛化误差可以任意小。

定理 6。对于任意常数 $c > 0$，考虑一个 RLS-GP 算法实例，其中 $F = \{AND, OR\}$、$L = \{x_1, \ldots, x_n\}$、 $\ell \geq n$；每次迭代均使用从完整真值表中均匀随机抽取的 $s = n^c \lg^2 n$ 行组成训练集来评估解的质量，并在解的采样误差不超过 $c' \lg n$ 时终止，其中 c' 是一个足够大的常数。对于与n 和或n 问题，该算法以至少 $1 - O(\log^2(n)/n)$ 的概率，在 $O(\log n)$ 次迭代内终止，并返回泛化误差至多为 n^{-c} 的解。

为证明该定理，我们将说明，RLS-GP 能够创建一棵树，其中每个变量最多出现一次，不包含不期望的函数（即，在 AND$_n$ 和 OR$_n$ 问题中分别不包含 OR 和 AND），并且包含足够多的不同变量，从而在 $O(\log n)$ 次迭代内，以

至少 $1 - O(\log^2(n)/n)$ 的概率采样到低于接受阈值的误差。此外，我们将证明，以高概率，遗传编程系统不会提前终止（即，不会返回泛化误差大于 n^{-c} 的解，其中 c 为任意常数）。

引理 3. 在定理 6 的设定下，如果 RLS-GP 从不接受包含不期望函数节点的解（即在 AND_n 和 OR_n 问题中分别包含 OR 和 AND），也从不接受包含任一变量多个副本的解，并且从不接受泛化误差比其祖先更差的解，那么它将在 $O(\log n)$ 次迭代内达到采样误差低于 $c' \lg n$ 的解，其中 $c' > 0$ 是一个适当的常数，且其概率至少为 $1 - O(1/n)$。

证明。为确保能够采样到低于 $c' \lg n$ 的误差，我们考虑构造一个期望采样误差至多为 $(c'/4) \lg n$ 的解所需的时间。对于足够大的 n，通过至多为 $((c'/4) \lg n)/(n^c \lg^2 n) = (c'/4) n^{-c}/\lg n \geq n^{-(c+1)}$ 的泛化误差即可达到这样的采样误差；也就是说，需要一个包含 $(c+1) \lg n$ 个或更多变量的合取（对于 AND_n 问题）或析取（对于 OR_n 问题）。

根据引理的条件，构造此类解所需的时间可以通过以下方式界定：先给出插入一个新变量并使用适当函数节点将其连接到树上的概率下界（即，在与$_n$ 上使用与，在或$_n$上使用或），再利用切尔诺夫界证明在特定迭代次数内会发生足够多次这样的插入（因为根据引理的条件，当前解中的不同变量数量不会减少）。具体而言，假设当前解包含$i < n/2$ 个不同变量，且不含不需要的函数节点，并令 X_i 表示一次变异插入一个新变量、使用适当函数节点将其连接到树上且被接受这一事件。我们界定 $\Pr[X_i] \geq (1/3)(1/2)(n-i)/n \geq \delta$，即对于 $i < n/2$的 $\delta \geq 1/12$。因此，通过应用乘法切尔诺夫界，在$(c''/\delta)(c+1) \lg n = O(\log n)$次迭代内至少接受$(c+1) \lg n$次此类变异的概率至少为 $1 - e^{-\Omega(c'' \log n)} = 1 - n^{-\Omega(c'')}$。因此，当 c'' 是一个足够大的常数时，该概率至少为 $1 - O(1/n)$。

我们应用引理 1，界定期望采样误差足够低的解未能达到接受阈值的概率。一旦构造出期望采样误差至多为 $(c'/4) \lg n$ 的解，其采样误差超过接受阈值的概率至多为 $n^{-\Omega(c')}$。因此，若恰当选择 c'，则该解以至少 $1 - O(1/n)$ 的概率立即被接受。

通过并集界合并失败概率，我们得出结论：在满足引理条件并恰当选择常数 c' 的情况下，RLS-GP 能以至少 $1 - O(1/n)$ 的概率，在 $O(\log n)$ 次迭代内构造出采样误差可接受的解。 □

现在，我们将利用 RLS-GP 运行时间的这一上界，说明它很可能避免所有可能阻碍引理 3 应用的陷阱。

引理 4. 在定理 6 的设定下，以至少 $1 - O(\log^2(n)/n)$ 的概率，在前 $O(\log n)$ 次迭代中，且在当前解的期望采样误差仍高于 $(c'/4) \lg n$ 时，RLS-GP 能够避免接受以下变异：(1) 插入当前解中已有变量的副本；(2) 插入不需要的函数节点（即，在与问题中插入或和与节点，在或问题中插入与和或节点）；或 (3) 增大当前解的泛化误差。

证明。对于主张（1），我们注意到，在前 $O(\log n)$ 次迭代中，树中至多包含 $O(\log n)$ 个不同变量（因为 RLS-GP 的每次迭代至多只能插入一个额外变量）。因此，变异操作添加一个解中已存在变量的概率（使用 HVL-Prime 的插入或替换子操作）至多为 $O(\log n/n)$。根据并集界，在前 $O(\log n)$ 次迭代中这种情况不会发生的概率至少为 $1 - O(\log^2(n)/n)$。

对于主张（2），我们注意到，通过插入操作将不需要的函数节点引入解中，主要有两种方式。首先，该函数可能是语义中性的。例如，对于与$_n$ 问题，如果祖先只包含与运算和互不相同的变量，那么一个叶节点 x_i 可以被替换为 $x_i \vee x_i$，而不影响解的任何输出。其次，用于评估解的适应度的采样过程可能没有采样到这样的输入：在这些输入上，子代是错误的，而祖先是正确的。我们将分别考虑这两种可能性。

由于语义中性的非期望函数节点插入需要插入一个变量的重复副本，因此，论断 (1) 已经给出了在 $O(\log n)$ 次迭代内不发生这类插入（因而不会被接受）的所需概率界。所有其他非期望函数节点插入都会增大解的泛化误差。误差增大的幅度取决于插入所替换子树中不同变量的数量，其中只替换单个叶节点的插入最容易被接受。

如果祖先解中的某个叶节点被一次插入操作移位，而该操作添加了一个不需要的函数节点和一个新变量，我们将见证一词用于指称这样一些输入：在这些输入上，构造出的子代解会产生错误输出，而父代解会产生正确输出

。对于与$_n$ 问题，见证输入是指将被移位的变量设为假，同时将子代解中的其余变量设为真时的输入；对于或$_n$ 问题，则将被移位的变量设为真，同时将子代解中的其余变量设为假。见证输入表明，子代解劣于其父代；对于其他所有输入，父代和子代产生的输出都相同。因此，只要采样过程找到至少一个见证，RLS-GP 就会拒绝这个变异后的解。

假设祖先解包含 U 个不同变量（并且只使用所需的函数节点，即 AND 上的与$_n$ 和 OR 上的或$_n$）；那么它在 $2^{n-U}-1$ 个可能的输入上出错，而至少有$2^{n-(U+1)}$ 个见证；也就是说，随机选中一个见证的概率，至少是随机选中一个祖先解出错的输入的概率的一半。因此，如果祖先解的期望采样误差至少为X，那么样本中见证的期望数量至少为 $X/2$。根据切尔诺夫界，样本中见证数量少于 $(c'/16)\lg n$ 的概率至多为 $e^{-(c'/128)\lg n} = n^{-\Omega(c')}$。通过适当设置常数 c'，可将此概率降至 $O(1/n)$。因此，根据并集界，只要解的期望采样误差仍高于 $(c'/4)\lg n$，在 $O(\log n)$ 次迭代内没有增加泛化误差的非期望函数节点插入被接受的概率至少为 $1-O(\log(n)/n)$。

对于论断 (3)，我们注意到，减少解中不同变量的数量，会使其泛化误差增加一倍以上。应用与上述拒绝有害的非预期函数节点插入类似的论证（在采样训练集中至少预期有 X 个见证），在 $O(\log n)$ 次迭代期间，不接受任何会增加泛化误差的变异的概率至少为 $1-O(\log(n)/n)$。

使用并集界合并这三个论断的误差概率，即可得到定理陈述。 □

最后，我们证明，以高概率，RLS-GP 不会过早终止（即，为泛化误差劣于定理 6 所要求的解采样到低于接受阈值的误差）。

引理 5。在定理 6 的设定下，以高概率，在任意多项式次数的迭代中，完整真值表中随机采样的 $s \geq n^c \lg^2 n$ 行组成的集合上，泛化误差大于 n^{-c} 的解，其采样误差都不会小于或等于 $c' \lg n$。

证明。回想一下，从完整真值表中均匀随机抽取 s 行来评估解的适应度时，如果解在抽取的行中至多有 $c' \lg n$ 行被判定为错误，RLS-GP 就会终止并返回当前解。由于解的泛化误差也等于该解在从完整真值表中均匀抽取的一行上出错的概率，因此，泛化误差至少为 n^{-c} 的解 X（X）的期望采样误差为 E（X）$\geq \lg^2 n$，其中从 $s = n^c \lg^2 n$ 行中均匀随机抽样。应用切尔诺夫界可知，采样误差 Y 小于其期望值一半（当 n 足够大时，该值高于 $c' \lg n$ 阈值）的概率超多项式地小：

$$\Pr\left[Y \leq 1/2\, E[Y]\right] \leq e^{-E[Y]/8} \leq n^{-\Omega(\log n)}.$$

根据并集界，在每次迭代中从完整真值表均匀随机抽取 $s = \Omega(n^c \lg^2 n)$ 行来评估解的质量时，RLS-GP 以高概率不会在任何多项式数量的迭代中返回泛化误差至少为 n^{-c} 的解。

□

我们的主要结果通过结合这些引理得以证明。

定理 6 的证明。根据引理 4，至少以 $1-O(\log^2(n)/n)$ 的概率可以应用引理 3，因此至少以 $1-O(\log^2(n)/n) - O(1/n)$ 的概率，能够在 $O(\log n)$ 次迭代内找到并返回一个采样误差达到接受阈值的解。根据引理 5，RLS-GP 在多项式次数的迭代内返回的任意解，其泛化误差以高概率优于期望的 n^{-c}。

□

像实践中通常所做的那样，执行 λ 次 RLS-GP 运行，并在任一实例确定其当前解达到接受阈值时终止，将保证以 $1-n^{-\Omega(\lambda)}$ 的概率，使用 $O(\lambda \log n)$ 次适应度评估，得到具有期望泛化误差的解。

现在我们考虑这样一种遗传编程系统：它配备了完整的布尔函数集和终端集，也就是说，该算法能够表示至多包含 n 个不同变量的任意布尔函数。我们证明，在使用多项式规模的训练集评估解的质量时，将终端集扩展为包含输入变量的否定（从而使 RLS-GP 能够表示任意布尔函数），并不会显著影响该算法在与问题和或问题上生成解的能力，这些解具有所需的多项式级小泛化误差。定理 7 将这一结果形式化；其证明基于以下观察：在这些问题中，插入取反变量几乎与插入正变量同样有用。

实验删除		F	L	训练集 运行	次数	最大迭代次数
RQ 1	leaf	$\{AND, OR\}$	$\{x_1, \ldots, x_n\}$	完整	500	10,000
RQ 2	子树	$\{AND, OR\}$	$\{x_1, \ldots, x_n\}$	完整	500	10,000
RQ 3	子树	$\{AND, OR\}$	$\{x_1, \ldots, x_n, \overline{x}_1, \ldots, \overline{x}_n\}$	完整	500	10,000
RQ 4	子树	$\{AND, OR\}$	$\{x_1, \ldots, x_n\}$	不完整	500	10,000
RQ 5	子树	$\{AND, OR\}$	$\{x_1, \ldots, x_n, \overline{x}_1, \ldots, \overline{x}_n\}$	不完整	500	10,000

表 1：五次实验中每次实验所采用的设置。

定理 7。对于任意常数 $c > 0$，考虑一个 RLS-GP 算法实例，其参数为 $F = \{AND, OR\}$、$L = \{x_1, \ldots, x_n, \overline{x}_1, \ldots, \overline{x}_n\}$、$\ell \geq n$；每次迭代中，从完整真值表均匀随机抽取 $s = n^c \lg^2 n$ 行作为训练集，以评估解的质量；当解的采样误差至多为 $c' \lg n$ 时终止，其中 c' 是一个足够大的常数。对于与问题和或问题，以至少 $1 - O(\log^2(n)/n)$ 的概率，该算法将在 $O(\log\ n)$ 次迭代内终止，并返回一个泛化误差至多为 n^{-c} 的解。

证明。该结果可通过沿用定理 6 的证明，并补充若干关于将取反变量加入当前解的效用的观察来证明。

如果一个解包含某个变量的正形式和负形式，且只包含适用于该问题的函数节点（即，对于与$_n$ 问题使用与节点，对于或$_n$问题使用或节点；分别产生矛盾或重言式），那么它显然能达到所需的多项式级小泛化误差，因为在2^n 个可能的输入中，它只会在一个输入上出错。不过，通常 RLS-GP 会在插入某个变量的正、负两个副本之前，就以所需质量的解终止。

任何由适用于该问题的函数节点以及正、负输入变量混合构成、但不同时包含任何一个变量正负形式的解，其出错的输入恰好比只包含相同变量正形式的解多两个。对于与$_n$ 问题，这两个输入分别是全真输入（包含取反变量的解在该输入上返回假），以及这样一种输入：将正形式出现的变量赋值为真、以取反形式出现的变量赋值为假，从而使所有已用叶节点的值都为真，并将其余变量全部赋值为真。因此，首次插入一个此前未以正形式出现在树中的取反变量，只比插入相应的正变量稍差；此后，只要没有产生矛盾或重言式，插入取反变量和正变量的效果便相同。对于任意多项式次的迭代，在抽取多项式规模的训练集时，这种差异几乎不可能被察觉：根据并集界，在多项式次迭代中，这两个输入被选入任何多项式规模训练集的概率都极低；在此条件下，RLS-GP 接受或拒绝涉及取反文字的变异，与接受或拒绝涉及同一变量正文字的变异，概率相同。

因此，对于引理3和引理4，可以将取反变量视为与其正向对应变量等价；任何包含对数数量的不同变量（无论为正向还是取反形式），且仅包含与问题相适应的函数节点的解，都能达到所需的泛化误差。因此，以高概率，可以在 $O(\log n)$ 次迭代内构造出一棵树，其中每个变量至多以正向或取反形式出现一次，不包含不需要的函数节点，并且达到所需的泛化误差。

按照定理 6 的证明将这些结果与引理 5 结合起来，并在最终的并集界中纳入未能抽取到可区分否定的训练集的概率，即可得到所需结果。 □

5 实验

我们进行了实验，以补充理论结果。对于算法和问题参数的每种组合，我们独立运行遗传编程系统 500 次，并报告找到解所需的平均迭代次数、解的平均大小以及标准差。

我们将探究五个独立的研究问题，以拓展从用于演化合取的 RLS-GP 理论中获得的知识。表 1 报告了每个研究问题的实验设置。

我们首先考察算法使用完整真值表评估解的质量时的行为。定理 1 表明，使用标准 HVL-Prime 算子（该算子执行仅叶节点删除与替换）时，在施加树大小限制的情况下，使用完整真值表的 RLS-GP 可能会陷入局部最优，从而导致期望运行时间无限。不过，该定理并未给出发生这一情况的概率界。

	$\ell = n$			$\ell = n+1$		
n	B	$\overline{T}$	$\overline{S}$	B	$\overline{T}$	$\overline{S}$
4	0.008	46.3 (28.0)	4.0 (0.0)	0.002	40.9 (21.8)	4.4 (0.5)
8	0.002	151.8 (91.9)	8.0 (0.0)	0.004	113.8 (51.5)	8.6 (0.5)
12	0.016	284.1(148.2)	12.0 (0.0)	0002	214.3 (99.5)	12.7 (0.5)
16	0.008	469.9(258.0)	16.0 (0.0)	0010	345.8 (161.0)	16.8 (0.4)

	$\ell = 2n$			$\ell = \infty$		
n	B	$\overline{T}$	$\overline{S}$	B	$\overline{T}$	$\overline{S}$
4	0	42.5 (25.8)	5.1 (1.2)	0	38.9 (24.3)	5.4 (2.0)
8	0	98.8 (49.0)	11.0 (2.3)	0	95.3 (43.8)	11.2 (3.0)
12	0	170.7 (99.7)	17.1 (3.3)	0	160.1 (57.1)	17.9 (4.5)
16	0	232.5 (80.9)	23.8 (4.1)	0	235.3 (92.7)	24.6 (6.0)

表 2：使用完整真值表评估解的质量时，RLS-GP 采用仅叶节点替换和删除操作，在不同的 n 和 ℓ 下，陷入局部最优的运行次数比例（B），以及成功运行的平均运行时间（$\overline{T}$）和解大小（$\overline{S}$）。括号中为标准差。

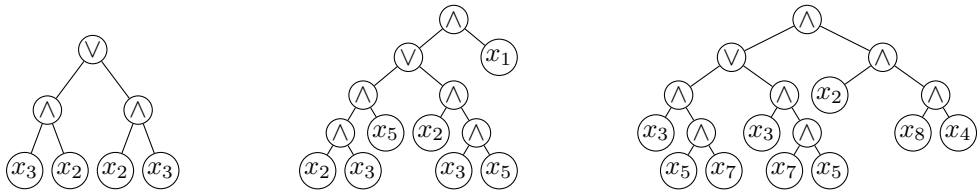

图 1：RLS-GP 使用仅叶节点替换和删除操作构造的局部最优树示例。这些树无法通过替换得到改进，且删除任何单个叶节点都会影响适应度。

研究问题 1：使用仅叶节点删除和完整真值表的 RLS-GP 陷入局部最优的可能性有多大？这一概率如何取决于树大小限制？

表 2 总结了 RLS-GP 的实验表现。实验结果证实，在树大小限制较小时，即使问题规模较小，RLS-GP 确实也会陷入局部最优，因此有必要使用子树删除。运行过程中构造的局部最优树示例见图 1。不过，随着 ℓ（即树大小限制）的增大，陷入局部最优的概率会降低。就解的质量而言，树大小限制较小时，最终程序中的冗余变量数量会减少，但代价是运行时间更长。对于 $\ell = n$，如果算法没有陷入局部最优，则会返回“精确”解。另一方面，较大的树大小限制（包括不设限制）会缩短期望运行时间，但代价是生成的程序中包含冗余变量。

现在我们来关注经过修改、允许进行子树删除的 HVL-Prime 算子，如定理 4 所述。该定理指出，如果设置了适当的树大小限制，算法将利用完整真值表找到最优解。然而，从理论上尚不清楚树大小限制是否必不可少。

研究问题 2：树大小限制如何影响使用子树删除和完整真值表来演化合取的 RLS-GP 的运行时间和解的质量？

正如理论所预测的，在我们的实验中，使用完整真值表和树大小限制时，RLS-GP 从未陷入停滞。表 3 展示了在不同问题规模和树大小限制下，找到全局最优解所需的平均迭代次数。实验再次表明，较小的树大小限制会减少冗余变量的数量，但代价是运行时间更长。较大的限制（包括完全不设限制）则能缩短运行时间，但会引入更多冗余变量。鉴于实际应用中通常需要设置树大小限制，我们将证明算法在不限制树大小的情况下能够演化出精确合取的工作留待未来工作。

现在，我们将否定文字加入终端集，把注意力转向配备完整函数集和终端集的 RLS-GP 的性能；这样的函数集和终端集能够表示任意布尔函数。由于已有证明表明，当 RLS-GP 仅能使用与函数并采用完整真值表时，否定操作无法高效地演化合取（Lissovoi 和 Oliveto，2019年），我们推测，当算法还可以使用或函数时，情况也同样如此。

研究问题 3：使用函数和字面量的完全表达集的 RLS-GP 能否进化出 con-

n	$\ell = n$		$\ell = n+1$		$\ell = 2n$		$\ell = \infty$	
	$\overline{T}$	$\overline{S}$	$\overline{T}$	$\overline{S}$	$\overline{T}$	$\overline{S}$	$\overline{T}$	$\overline{S}$
4	51.2 (31.1)	4.0 (0.0)	42.5 (23.5)	4.4 (0.5)	38.8 (20.8)	5.1 (1.2)	39.1 (22.3)	5.3 (1.8)
8	147.5 (83.3)	8.0 (0.0)	129.9 (69.1)	8.7 (0.5)	93.5 (39.1)	11.3 (2.4)	92.3 (38.1)	11.6 (3.0)
12	325.9 (184.4)	12.0 (0.0)	233.4 (123.9)	12.8 (0.4)	153.6 (56.6)	17.7 (3.1)	151.2 (50.3)	18.3 (3.8)
16	544.6 (333.8)	16.0 (0.0)	377.0 (176.0)	16.9 (0.4)	228.3 (74.6)	24.5 (3.7)	221.0 (72.0)	25.2 (4.9)

表 3：使用子树删除子操作的 RLS-GP 的平均运行时间（$\overline{T}$）和解大小（$\overline{S}$），以及使用完整真值表评估解的适应度时，在不同 n 和 ℓ 下的结果。标准差显示在括号中。

n	$\ell = n$			$\ell = n+1$		
	B	$\overline{T}$	$\overline{S}$	B	$\overline{T}$	$\overline{S}$
4	0.000	655.3 (638.0)	4.0 (0.0)	0.000	513.7 (480.8)	4.5 (0.5)
8	0.998	1750.0 (-)	8.0 (-)	0.994	5535.3 (3066.0)	8.3 (0.6)
12	1.000	-	-	1.000	-	-
16	1.000	-	-	1.000	-	-

n	$\ell = 2n$			$\ell = \infty$		
	B	$\overline{T}$	$\overline{S}$	B	$\overline{T}$	$\overline{S}$
4	0.000	431.0 (392.7)	5.7 (1.3)	0.000	422.7 (425.7)	6.4 (2.5)
8	0.990	6443.8(3789.4)	11.0 (1.6)	0.982	4478.7(2371.6)	12.4 (3.3)
12	1.000	-	-	1.000	-	-
16	1.000	-	-	1.000	-	-

表4：使用子树删除子操作、否定文字，并通过完整真值表评估解的适应度的 RLS-GP，在不同 n 和 ℓ 条件下，未能在 10^4 次迭代（B）内找到全局最优解的运行次数占比，以及成功运行的平均运行时间（$\overline{T}$）和解大小（$\overline{S}$）。可用的标准差列于括号中。

使用完整真值表的合取?

表4展示了允许使用否定文字的影响：随着问题规模增大，RLS-GP 几乎总会构造出一个否定文字，并且无法在分配的 10,000 次迭代运行时间内找到全局最优解，正如我们所推测的那样。

最后，我们考察 RLS-GP 使用不完整训练集时的行为，这使我们能够进化出更大的合取，即更大的问题规模。定理6和定理7的结果依赖于算法在达到对数级采样误差后停止，使用 $F = \{AND, OR\}$ 分别对应不含和含否定文字的情况。然而，从理论上看，并不清楚这一误差阈值是否必要。

研究问题4：RLS-GP 使用训练集进化合取时，是否需要对数级误差阈值？移除该阈值会如何影响算法仅使用正文字时的性能?

我们开展实验，比较 RLS-GP 在误差达到 0 时停止与更早停止时的性能$n = 50$。图2绘出了这两种变体的平均运行时间。该图证实了我们的理论结果：这些算法通常以对数时间运行，并生成相对于训练集大小而言包含对数数量叶节点的解。以误差为 0 时停止可以得到更好的解，但代价是运行时间更长。

现在我们将注意力转向否定文字。

研究问题5：RLS-GP 是否需要对数误差阈值，才能利用训练集演化出合取？移除该阈值后，算法在同时使用正文字和否定文字时的性能会受到怎样的影响?

图3探讨了将取反变量加入文字集合的影响。对于 $n = 50$，当解平均包含约 8 个叶节点时，否定会导致构造出矛盾，使算法即使面对较大的训练集，也能在平均 50 次迭代内终止。图4和图5分别展示了文字集合中包含和不包含否定时，最终程序中的或的平均数量。虽然数量不多，但可接受的采样误差阈值越接近

0.

6 结论

我们分析了 RLS-GP 算法的一个变体的行为，并证明了在使用完整真值表评估解的质量时，以及在使用均匀随机选取的可能输入的多项式规模样本时的期望运行时间上界。在设置树大小限制并采用能够删除整个子树的变异算子后，RLS-GP 能够高效地进化出布尔函数——与 n（n 个变量的合取）（或或 n（n 个变量的析取）），前提是它能够使用二元合取和析取运算符。

使用完整真值表评估解的质量时，我们证明，如果终端集恰好包含目标函数中的不同字面量，那么在期望意义上，经过 $O(\ell n \log^2 n)$ 次迭代即可找到最优解。从实验结果来看，随着 ℓ（树大小限制）的增大，遗传编程系统能够更快地找到解，这表明理论界限对该过程的建模过于悲观。相反，树大小限制较大的解往往包含更多冗余变量，这表明优化时间与解的复杂度之间存在权衡。然而，如果终端集中还包含不必要的否定文字，我们通过实验表明，该算法极有可能陷入局部最优（即矛盾）。形式化证明留待未来工作。

通过采样多项式数量的输入来评估程序质量时，进化得到的解与目标函数并不完全等价，但具有良好的泛化能力：每次迭代中均匀随机采样多项式数量的输入，即可实现任意多项式级小的泛化误差。即使遗传编程系统具有完全表达能力（也就是说，它可以表示至多包含 n 个不同变量的任意布尔函数），这一结果仍然成立。我们的理论结果预测，在这种情况下，RLS-GP 通常能够避免插入析取，这与实验结果一致。然而，当存在不必要的否定文字时，树中很可能会插入一个矛盾，从而得到平均规模较小的解；也就是说，可能会进化出始终返回 false 的 OFF函数。尽管它仅在一个输入上返回错误输出，但仍具有良好的泛化能力。

尽管这些结果代表着遗传编程行为理论分析的一大进步，但仍有许多工作有待完成。除了要解决取消树大小限制这一开放问题外，还应将分析扩展到任意规模的布尔合取与析取的进化，即不必使用所有 n 可用变量。此外，还应考虑更复杂的目标函数，在这些函数中，使用种群和交叉可能是必要的。

致谢 衷心感谢工程与物理科学研究委员会（EPSRC Grant No. EP/M004252/1）提供的经费支持。本研究还获得了未来投资计划项目的公共资助，项目编号为 ANR-11-LABX-0056-LMH，LabEx LMH。

参考文献

Bartoli, A., Cumar, S., Lorenzo, A. D., and Medvet, E. (2014). 深度包检测中的正则表达式集合压缩。载于《自然启发的并行问题求解》（PPSN XIII）论文集，第394–403页。施普林格。Corus, D., Lissovoi, A., Oliveto, P. S., and Witt, C. (2021). 关于稳态进化算法与选择压力：为何基于逆排名的繁殖试验分配最优。ACM进化学习与优化汇刊，1:1–38。Corus, D. and Oliveto, P. S. (2020). 关于种群对标准稳态遗传算法开发速度的益处。Algorithmica，82:3676–3706。Dang, D., Eremeev, A. V., and Lehre, P. K. (2021). 使用非精英进化算法逃离局部最优。载于第三十五届AAAI人工智能会议（AAAI 2021）论文集，第12275–12283页。AAAI出版社。Dang, D.-C., Friedrich, T., K¨otzing, T., Krejca, M. S., Lehre, P. K., Oliveto, P. S., Sudholt, D., and Sutton, A. M. (2018). 利用交叉和涌现多样性逃离局部最优。IEEE进化计算汇刊，22(3):484–497。Doerr, B. (2019年). 通过随机占优分析随机化搜索启发式算法。理论计算机科学，773:115–137。

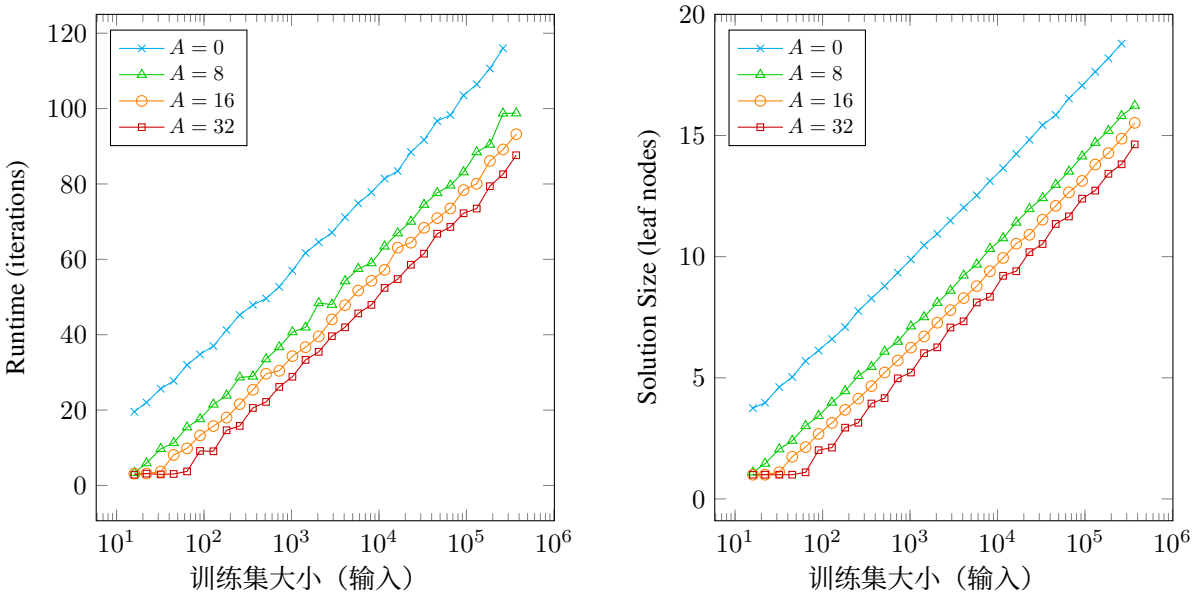

图2：使用不完整训练集时，采用子树删除的 RLS-GP 得出的平均运行时间和树大小；当采样误差不超过 A, $n = 50, \ell = \infty$ 时停止，结果取 500 次独立运行的平均值。

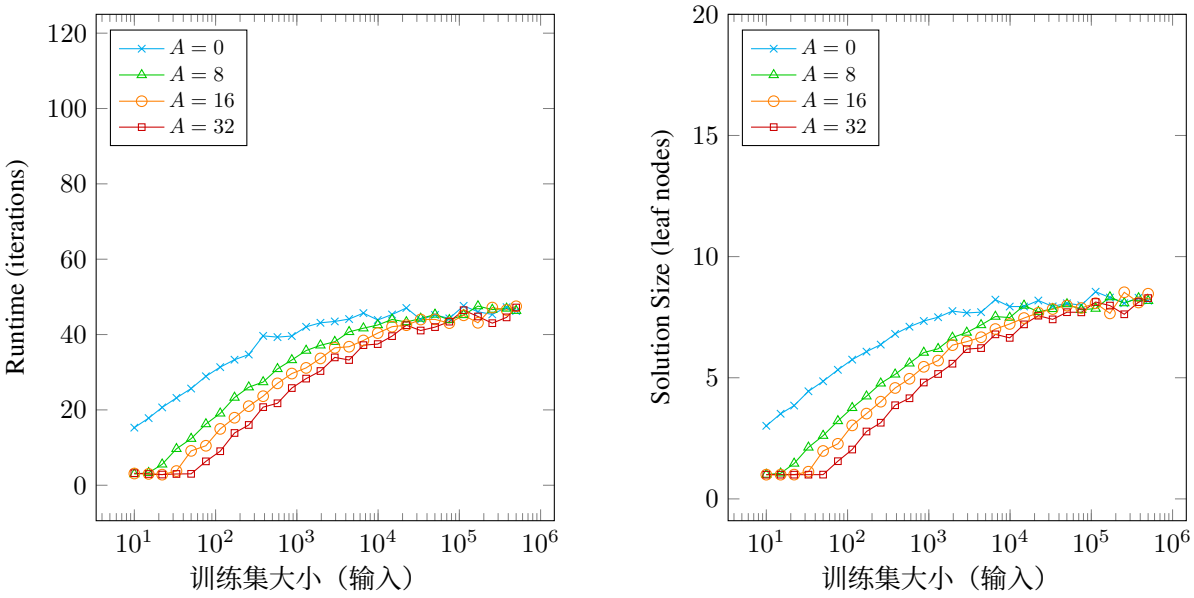

图3：使用不完整训练集时，采用子树删除和否定文字的 RLS-GP 得出的平均运行时间和树大小；当采样误差不超过 $A, n = 50, \ell = \infty$ 时停止，结果取 500 次独立运行的平均值。

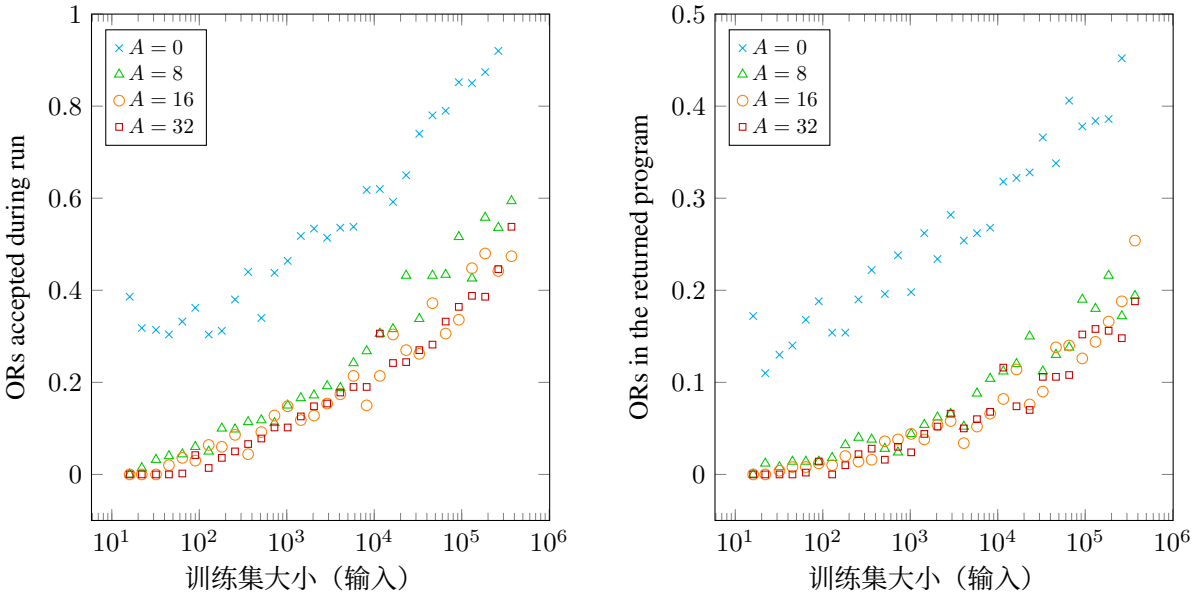

图4：使用不完整训练集时，RLS-GP 通过子树删除插入并保留在最终解中的 OR 节点数量；当采样误差不超过 A, $n = 50, \ell = \infty$ 时停止，数值为 500 次独立运行的平均值。

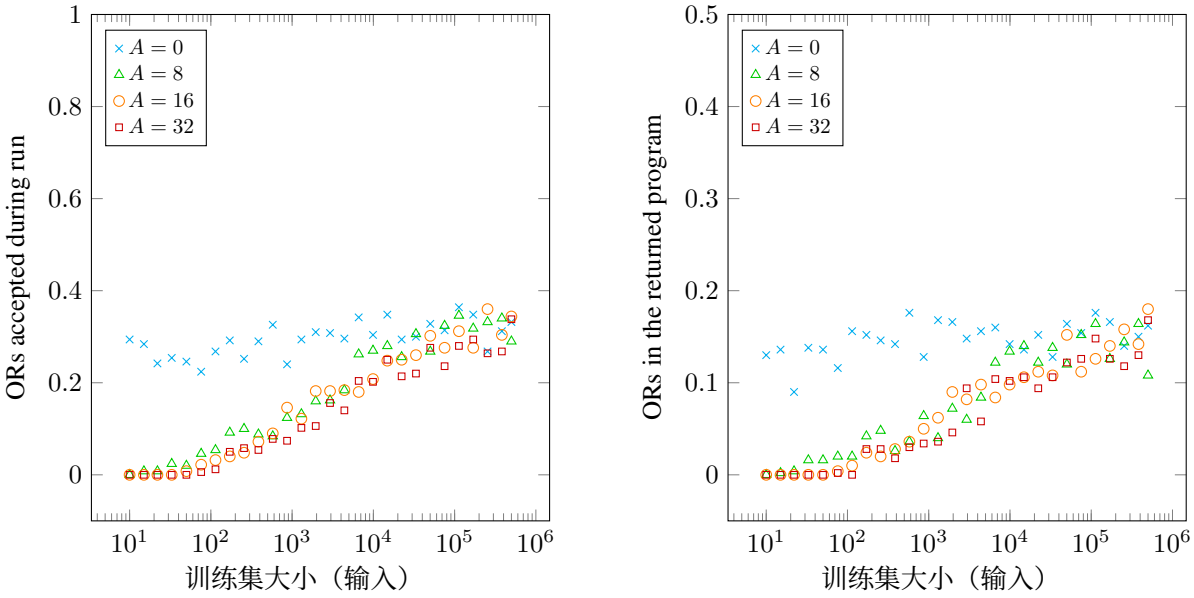

图5：使用不完整训练集时，RLS-GP 通过子树删除和否定文字插入并保留在最终解中的 OR 节点数量；当采样误差不超过 A, $n = 50, \ell = \infty$ 时停止，数值为 500 次独立运行的平均值。

Doerr, B.（2020）。用于分析随机化优化启发式算法的概率工具。载于 Doerr, B. 和 Neumann, F. 编，《进化计算理论：离散优化最新进展》，第 1–87 页。施普林格。Doerr, B. 和 Goldberg, L. A.（2013）。自适应漂移分析。《Algorithmica》，65：224–250。Doerr, B., Happ, E. 和 Klein, C.（2012a）。交叉在进化计算中确实可能发挥作用。《理论计算机科学》，425：17–33。Doerr, B., Johannsen, D. 和 Winzen, C.（2012b）。乘法漂移分析。《Algorithmica》，64(4)：673–697。Doerr, B., K¨otzing, T., Lagodzinski, J. G. 和 Lengler, J.（2020）。字典序简约压力对排序/多数问题运行时间的影响。《理论计算机科学》，816：144–168。Doerr, B., Lissovoi, A. 和 Oliveto, P. S.（2019）。通过遗传编程与合取和析取演化布尔函数。载于《遗传与进化计算会议论文集》（GECCO 2019），第 1003–1011 页。Droste, S., Jansen, T. 和 Wegener, I.（2002）。关于（1+1）进化算法的分析。《理论计算机科学》，第 51–81 页。Durrett, G., Neumann, F. 和 O'Reilly, U.（2011）。对两个模拟孤立程序语义问题的简单遗传编程进行计算复杂性分析。载于《遗传算法基础研讨会论文集》（FOGA 2011），第 69–80 页。美国计算机协会。Gathercole, C. 和 Ross, P.（1994）。遗传编程监督学习中的动态训练子集选择。载于《自然启发的并行问题求解论文集》（PPSN III），第 312–321 页。施普林格。Huang, Z., Zhou, Y., Chen, Z. 和 He, X.（2019）。带交叉的 MOEA/D 在离散优化问题上的运行时间分析。载于《第三十三届AAAI人工智能会议论文集》（AAAI 2019），第 2296–2303 页。AAAI出版社。Jansen, T., J ong, K. A. D. 和 Wegener, I.（2005）。关于进化算法中子代种群规模的选择。《进化计算》，13(4)：413–440。Johannsen, D.（2010）。随机组合结构与随机化搜索启发式算法。博士论文，萨尔大学，萨尔布吕肯。K¨otzing, T., Sutton, A. M., Neumann, F. 和 O'Reilly, U.（2014）。重新探讨 MAX问题：变异在遗传编程中的重要性。《理论计算机科学》，545：94–107。Koza, J. R.（1992）。《遗传编程——通过自然选择实现计算机编程》。复杂自适应系统。麻省理工学院出版社。Koza, J. R.（2010）。遗传编程取得的媲美人类的成果。《遗传编程与可进化机器》，11(3-4)：251–284。Lehre, P. K. 和 Oliveto, P. S.（2018）。随机搜索算法的理论分析。载于 Mart´ı, R., Pardalos, P. M. 和 Resende, M. G. C. 编，《启发式算法手册》，第 849–884 页。施普林格。Lengler, J.（2020）。漂移分析。载于 Doerr, B. 和 Neumann, F. 编，《进化计算理论：离散优化最新进展》，第 89–131 页。施普林格国际出版，查姆。Lissovoi, A. 和 Oliveto, P. S.（2019）。遗传编程演化布尔合取的时间与空间复杂度。《人工智能研究杂志》，66：655–689。Lissovoi, A. 和 Oliveto, P. S.（2020）。遗传编程的计算复杂性分析。载于 Doerr, B. 和 Neumann, F. 编，《进化计算理论：离散优化最新进展》，第 475–518 页。施普林格国际出版，查姆。Liu, L. 和 Shao, L.（2013）。从 RGB-D视频数据中学习判别性表征。载于《第二十三届国际人工智能联合会议论文集》（IJCAI 2013），第 1493–1500 页。IJCAI/AAAI。Lynch, D., Fenton, M., Fagan, D., Kucera, S., Claussen, H. 和 O'Neill, M.（2019）。异构无线通信网络中的自动自优化。《IEEE/ACM网络汇刊》，27(1)：419–432。

Mambrini, A. 和 Oliveto, P. S.（2016）。关于用于演化布尔函数的简单遗传编程分析。载于《遗传编程第十九届欧洲会议论文集》（EuroGP 2016），第 99–114 页。施普林格。Miranda, I. M., Aranha, C. 和 Ladeira, M.（2019）。使用遗传编程进行特征构造的脑电图信号分类。载于《遗传与进化计算会议论文集》（GECCO 2019），第 1275–1283 页。Mitavskiy, B., Rowe, J. E. 和 Cannings, C.（2009）。在保持链路总数不变的条件下优化网络通信的局部搜索策略理论分析。《国际智能计算与控制论杂志》，2(2)：243–284。Moore, J. H., Olson, R. S., Chen, Y. 和 Sipper, M.（2018）。使用遗传编程自动发现检验统计量。《遗传编程与可进化机器》，20(1)：127–137。Neumann, F. 和 Wegener, I.（2007）。随机局部搜索、进化算法与最小生成树问题。《理论计算机科学》，378(1)：32–40。O'Reilly, U. M. 和 Oppacher, F.（1996）。遗传编程的比较分析。载于《遗传编程进展 2》，第 23–44 页。麻省理工学院出版社。Poli, R., Langdon, W. B. 和 McPhee, N. F.（2008）。《遗传编程野外指南》。Lulu Press。Sutton, A. M.（2021）。交叉的固定参数可处理性：稳态遗传算法求解最近字符串问题。《Algorithmica》，83：1138–1163。Vu, T. M., Probst, C., Epstein, J. M., Brennan, A., Strong, M. 和 Purshouse, R. C.（2019）。利用多目标遗传编程迈向逆向生成式社会科学。载于《遗传与进化计算会议论文集》（GECCO 2019），第 1356–1363 页。Witt, C.（2006）。$(\mu + 1)$ 进化算法在简单伪布尔函数上的运行时间分析。《进化计算》，14(1)：65–86。Zheng, W., Liu, Y. 和 Doerr, B.（2022）。非支配排序遗传算法 II（NSGA-II）的首个数学运行时间分析。载于《第三十六届 AAAI 人工智能会议论文集》（ AAAI 2022 ），第 10408–10416 页。AAAI 出版社。