在多目标局部搜索中，随机方法比系统方法更快

Zimin Liang 和 Miqing Li*

arXiv:2601.06318v1 [cs.NE] 9 Jan 2026

摘要——**局部搜索是运筹学和组合优化中的一种基础方法。它已广泛应用于各种具有挑战性的问题，包括多目标优化问题；在这类问题中，需要同时考虑多个（且往往相互冲突的）目标。在多目标局部搜索算法中，一种常见做法是维护一个档案集，记录迄今找到的所有非支配解；算法会从中迭代采样一个解，以探索其邻域。在这一过程中，一个核心问题是如何探索所选解的邻域。一般而言，主要有两种方法：1）系统探索；2）随机采样。前者会系统地探索该解的邻居，直到满足停止条件——例如，邻域中的解均已检查完毕（即最佳改进策略），或找到更优解后立即停止（即首次改进）。与之相反，后者会随机选取并评估该解的一个邻居。人们可能会认为，系统探索效率更高，因为它能避免多次访问相同的邻居。然而，本文表明事实未必如此。我们首先通过实证证明，在一系列多目标问题中，随机采样方法始终比系统探索方法更快。随后，我们通过玩具示例直观地解释这一现象，指出随机采样方法的优越性能取决于“优质邻居”的分布。接着，我们证明搜索过程中此类邻居的数量遵循某种概率分布。最后，基于这一分布，我们从理论上解释了为何无论采用最佳改进策略还是首次改进策略，随机采样都比系统探索更高效。**

索引词—**多目标组合优化、多目标优化、局部搜索**。

I. 引言

局部搜索（LS）是一类搜索启发式方法，它通过应用局部变化，在候选解空间中从一个解移动到另一个解。局部搜索已广泛用于解决各种具有挑战性的搜索与优化问题，包括布尔可满足性问题 [45, 67, 70], 最大割 c最大准团问题 [26], 最大k-plex [61], 图边划分 [30], 拉丁方补全[66], 拟阵优化 [6], 整数线性规划[48], 以及其他应用问题 [57, 69, 60, 29]。

多目标优化是指需要同时考虑多个目标的优化场景。LS 因其简单的算法结构以及适用于离散搜索空间的邻域驱动探索，成为解决多目标组合优化问题（MOCOPs）的一种常用工具 [9], 由于其简单的算法结构和适合离散搜索空间的邻域驱动探索。

早期的多目标局部搜索方法通常通过将一个多目标组合优化问题转化为若干单目标问题，并使用标量适应度（由多个目标聚合而成）对解进行排序来设计。代表性算法包括 [33, 34, 52]。后来，基于帕累托的局部搜索（PLS）启发式方法日益受到关注，并逐渐成为主流方法 [51, 27, 49, 21, 20, 35, 58, 54, 37]。PLS采用帕累托支配来比较解 [51]。它维护一个包含迄今为止发现的非支配解的档案集，并迭代地从档案集中采样解，以探索其邻域。这类PLS算法在多目标旅行商问题 [49],背包问题 [5],NK景观 [2] 以及调度 [47] 等多目标组合优化问题上表现出有竞争力的性能。最近，若干研究 [32, 53, 43]表明，与成熟的多目标进化算法（例如NSGA-II [16], SMS-EMOA [7] 和MOEA/D [68]）相比，PLS具有性能优势。

PLS算法中的一个关键问题是如何探索从档案集中采样得到的解的邻域。一般而言，有两种方法：1）系统探索；2）随机采样。前者称为系统化PLS或s-PLS，会系统地探索该解的邻居，直至满足停止条件——例如邻域已被穷尽（即最佳改进策略），或找到更优解后停止（即首次改进策略）。后者称为随机化局部搜索或r-PLS，会随机选择并评估该解的一个邻居。系统化PLS在运筹学和元启发式算法领域广受欢迎 [21, 17, 59, 56, 12, 25]。另一方面，只有少数研究考虑随机化PLS[14, 35], ， 尤其是在进化计算理论界 [40, 19, 18, 64]。

由此自然产生的一个问题是，这两种方法中哪一种更高效。系统搜索似乎更为可取，因为它会有条理地探索一个解的邻域，并确保每个邻居至多被访问一次。相比之下，随机化PLS可能会多次访问同一邻居（因为在每一步中，它都会从档案集中随机选择一个解，然后随机选取其邻居之一），从而导致重复评估。

然而，本文表明，通常随机化PLS比系统化PLS更快。我们首先通过实验证明，在广泛的问题中，无论邻域探索采用最佳改进策略还是首次改进策略，随机化PLS的表现都始终优于系统化PLS。接着，我们给出一种

Zimin Liang和Miqing Li（通讯作者）任职于英国伯明翰大学计算机科学学院，地址为英国伯明翰埃奇巴斯顿，邮编B15 2TT（电子邮件：zxl525@student.bham.ac.uk；m.li.8@bham.ac.uk）。

算法 1 系统化帕累托局部搜索（s-PLS）

要求： $max\ \underline{e}val$ （算法终止条件）， $\mathcal{N}$ （邻域函数，例如 1 位邻域）
$s \leftarrow random_solution()$
$explored(s) \leftarrow$ FALSE ▷ 将 s 标记为未探索
$A \leftarrow \{s\}$ ▷ 将 s 放入档案集
$eval \leftarrow 1$
当 $eval \leq max\ \underline{e}val$ **和** $\exists a \in A:\ explored\ (a)\ =$ FALSE **时**
  $s \leftarrow selection\ (\{a \in A | explored\ (a) = 假\})$ ▷选择一个尚未探索的解，并根据某种策略从 A 中选择（例如，随机选择或使用指标函数/标量化函数）
  重复
    $s' \leftarrow next_neighbour(\mathcal{N}(s))$ ▷ 从以下位置获取下一个邻居 s的邻域，通常按字典序或随机顺序
    如果 $\nexists a \in A:\ a \preceq s'$ **则**
      $A \leftarrow A \cup \{s'\} \setminus \{a' \in A \mid s' \prec a'\}$ ▷ 确保档案集仅包含唯一非支配解
      $explored(s') \leftarrow$假
    结束 if
    $eval \leftarrow eval + 1$
  直到 $stop_condition()$ ▷ 当满足某个条件时，停止探索此邻域，例如邻域已穷尽，或找到满意解（例如该解未被档案集支配）
  $explored(s) \leftarrow$真 ▷ 标记 s为已探索
结束循环
返回 A

算法 2 随机化帕累托局部搜索（r-PLS）

要求： $max\ \underline{e}val$ （算法终止条件）， $\mathcal{N}$ （邻域函数，例如 1 位邻域）
$s \leftarrow random\ \underline{s}olution()$
$A \leftarrow \{s\}$ ▷将 s 放入档案集
$eval \leftarrow 1$
当 $eval \leq max\ \underline{e}val$ **时**
  $s \leftarrow selection(A)$ ▷ 根据某种策略从 A中选择一个解（例如，随机选择或使用指标函数/标量化函数）
  $s' \overset{U}{\sim} \mathcal{N}(s)$ ▷从.....中均匀随机采样一个邻居邻域 s
  如果 $\nexists a \in A:\ a \preceq s'$ **则**
    $A \leftarrow A \cup \{s'\} \setminus \{a' \in A \mid s' \prec a'\}$ ▷确保档案集仅包含唯一非支配解
  结束如果
  $eval \leftarrow eval + 1$
结束循环
返回 A

利用玩具示例对这一现象的直观解释。最后，我们通过分析两种PLS方法在所研究问题的邻居解分布下找到优良解的运行时间，提供理论上的理解（即找到一个不被档案集支配、从而能够提升档案集质量的解）。本工作的主要贡献可概括如下：

- 在多目标优化中两大主要局部搜索算法类别中，我们证明其中一类比另一类更快，从而驳斥了一个普遍看法：这两类局部搜索算法各有所长，适用于不同的问题。

- 我们通过实验研究了 PLS 算法搜索过程中的邻域结构（优质邻居的数量），并证明它们遵循某种概率分布。

- 我们从理论上探究了实验观察结果背后的原因，并证明在该概率分布下，一类算法找到优良解所需的时间更短。

值得注意的是，随机化帕累托局部搜索启发式算法（如 SEMO） [40], 在进化计算理论领域的运行时间分析文献中得到了广泛研究 [8, 19, 18, 71, 64]。这些研究以及理论领域的其他工作，关注的是找到帕累托最优解所需的运行时间，并探讨诸如算法是否难以找到帕累托最优解，以及找到一个解后，在帕累托前沿中找到其他最优解有多容易等问题。我们的工作不考虑找到最优解所需的时间。本文的结果并不意味着 PLS 找到的解具有任何最优性保证，而是考察不同 PLS算法在搜索过程中找到更优解的速度。这具有实际意义，因为对于大多数现实世界中的多目标组合优化问题而言，通常无法在合理时间内获得最优解 [24], ，因此了解哪些类型的算法能够更快地取得进展是很有价值的了解哪些类型的算法能够更快地取得进展。

II. 预备知识

令 $\boldsymbol{f}(\mathbf{x}) = (f_1(\mathbf{x}) \ldots, f_m(\mathbf{x}))$ 表示决策空间 X上的一个 m 目标（最小化）问题，其中 $\mathbf{x} \in X$,$f_i : X \to \mathbb{R}$。若且唯若 $\forall i \in \{1, \ldots, m\}:\ f_i(\mathbf{x}) \leq f_i(\mathbf{y})$，我们称 x 弱帕累托支配 y（记作：$\mathbf{x} \preceq \mathbf{y}$）；若至少一个不等式为严格不等式，则称 x 帕累托支配 y（记作：$\mathbf{x} \prec \mathbf{y}$）。帕累托集由所有非支配解组成，记作 $PS = \{\mathbf{x} \in X \mid \nexists \mathbf{y} \in X : \mathbf{y} \prec \mathbf{x}\}$; 帕累托前沿由这些解在目标空间中对应的点组成，记作 $PF = \{\boldsymbol{f}(\mathbf{x}) \mid \mathbf{x} \in PS\}$.

局部搜索算法中的一个核心概念是邻域，它定义了搜索景观的结构。解$\mathbf{x}$ 的邻域记作 $\mathcal{N}(\mathbf{x}) \subseteq X \setminus \mathbf{x}$，通常由通过一次基本操作即可从$\mathbf{x}$ 到达的解构成（例如，在二进制编码问题中翻转一位）。

III. 两类基于帕累托的局部搜索

在本文中，我们重点研究基于帕累托的局部搜索（PLS）算法 [51],，这类算法涵盖了大多数多目标局部搜索启发式方法 [9]。PLS 是一种迭代搜索过程，通过对解进行局部变异，并利用帕累托支配将其与新解进行比较。它维护一个档案集，用于存储迄今为止找到的所有非支配解。在每次迭代中，PLS 都会从档案集中选取一个解，通过探索其邻域来生成新解。根据探索解的邻域的方式，PLS算法可分为两类：系统化PLS（s-PLS）和随机化PLS（r-PLS）。

s-帕累托局部搜索算法会逐一访问解的邻居（并测试是否可以将其放入档案集），直至满足某个条件。该条件可以是邻域穷尽（即最佳改进），也可以是找到满意解（即首次改进）。在多目标优化中，满意解可以有不同的定义，但最常见的两种是：i）不受档案集中任何解支配的解；ii）支配档案集中至少一个解的解 [47, 21]。本研究同时考虑这两种情况。一旦解的邻域已探索完毕，就会将其标记为

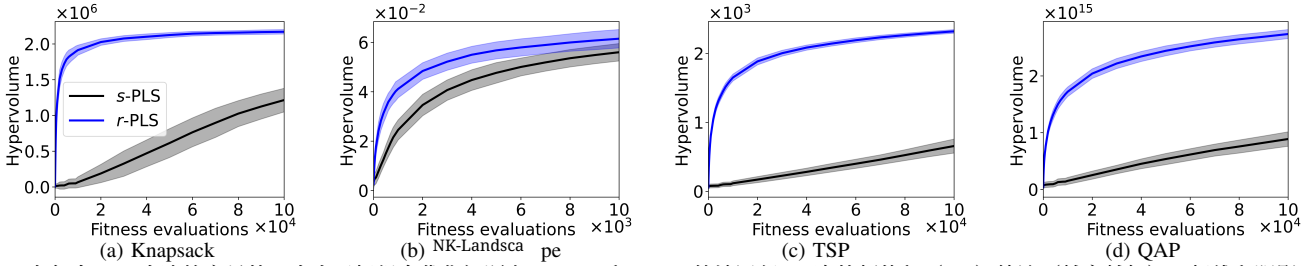


图 1：在包含 100 个决策变量的四个多目标组合优化问题上，*s*-PLS 和 *r*-PLS 算法运行 30 次的超体积（HV）轨迹（越高越好）。粗线和阴影区域分别表示 HV 的均值和标准差。

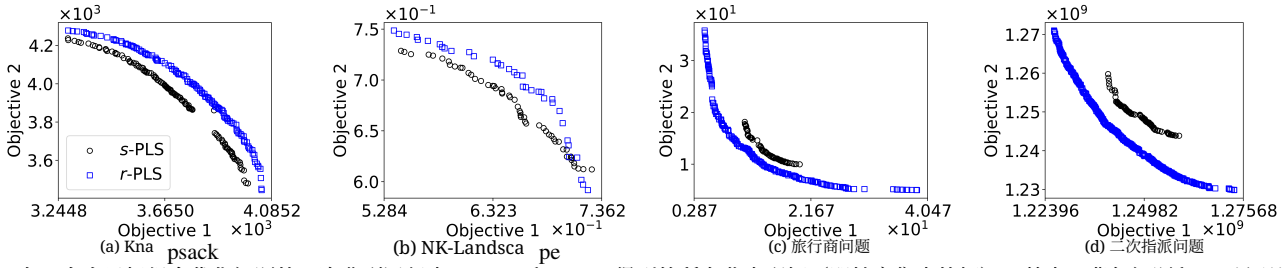


图 2：在四个多目标组合优化问题的一次典型运行中，*s*-PLS 和 *r*-PLS 得到的所有非支配解（即档案集中的解）。其中，背包问题和 NK景观是最大化问题，而旅行商问题和二次指派问题是最小化问题。

“已探索”，且不再重新访问。*s*-帕累托局部搜索的框架见算法 1。*s*-帕累托局部搜索在该领域广受欢迎。许多现有的帕累托局部搜索启发式算法都属于这一类，例如原始的帕累托局部搜索 [51],MORBC [2]及其变体 [31],GPLS [5]，2PPLS [50]，任意时刻 PLS [21]，PPLS/D [58]，LOMONAS [53]，以及一种新型增强帕累托局部搜索 [37]。

与*s*-PLS 不同，*r*-PLS 在每次迭代中随机从档案集中选择一个解，然后随机访问其一个邻居，而不记录哪些解已被访问或探索。这种纯随机采样方法可能会多次访问某个解的同一邻居，导致冗余开销。算法 2 给出了*r*-PLS 的框架。开发*r*-PLS 算法的现有研究相对较少。值得注意的例子包括 SEMO [27, 40]，、SPLS [20]和 MPLS [35]。尽管如此，由于*r*-PLS 具有引人注目的特性（简单、精简且易于推广），进化计算理论领域经常对其进行研究 [8, 19, 18,71, 64]，。

一个值得关注的问题是，PLS 的两类算法中哪一类速度更快。人们可能会认为*s*-PLS 更高效，因为它采用系统搜索，避免反复探索相同的解，尤其是在使用首次改进策略时。然而，在接下来的章节中，我们将说明事实未必如此。

IV. *s*-PLS 与*r*-PLS 的实证比较

我们考察帕累托局部搜索算法类别中最基础的版本：它们会随机选择档案集中的一个解进行邻域探索，因为我们的重点是通用的邻域探索策略，而不是其他可调节的算法组件（例如选择准则）。具体而言，对于*r*-PLS，我们考察 SEMO [40]。对于*s*-PLS，我们考察规范版本 [51]，该版本会从档案集中随机选择一个尚未探索的解来探索其邻域，并在邻域耗尽时停止探索，即采用最佳改进策略。此外，我们之后还将考察首次改进策略，该策略被认为比原始的*s*-PLS [47, 21]更快。

我们考察了四种常用的多目标组合优化问题：多目标 0/1 背包问题 [62]，旅行商问题（TSP） [55]，二次指派问题（QAP） [38] 和 NK 景观 [3]。这些问题具有多样的特征，包括伪布尔（背包问题和 NK 景观）与排列（TSP 和 QAP）形式；适应度景观从平滑（背包问题和 TSP）到崎岖（QAP 和具有较大 K的 NK 景观）；基于顺序（TSP）与基于位置（QAP）的排列；以及受约束（背包问题）与无约束（TSP、QAP 和 NK 景观）设置。每个问题均设置了三种规模（100、200 和 500 个决策变量）。附录 A 对这些多目标组合优化问题作了详细介绍。我们使用超体积 [72]，这一强帕累托相容指标来评估搜索算法所得解的质量，该指标能够衡量收敛性、分布范围、均匀性和基数 [44]，。超体积的参考点以及详细的实验和算法设置见附录 B。

图1展示了在四个包含100个决策变量的问题上，*s*-PLS（黑色）和*r*- PLS（蓝色）在30次独立运行中的平均超体积（粗线）和标准差（阴影区域）。可以看出，在所有问题的整个搜索过程中，*r*-PLS获得的HV值始终优于*s*-PLS。在NK景观问题上，*r*-PLS的性能优势相对较小；而在其他三个问题上，*r*-PLS从搜索过程一开始就展现出明显优势。问题规模较大时，这种差异甚至更加明显（参见附录C中四个200D和500D问题的结果）。图7展示了图1中搜索结束时，两种算法在四个问题的一次典型运行中获得的非支配解。显然，在背包问题和NK景观问题上，*r*-PLS获得的解具有更好的收敛性，

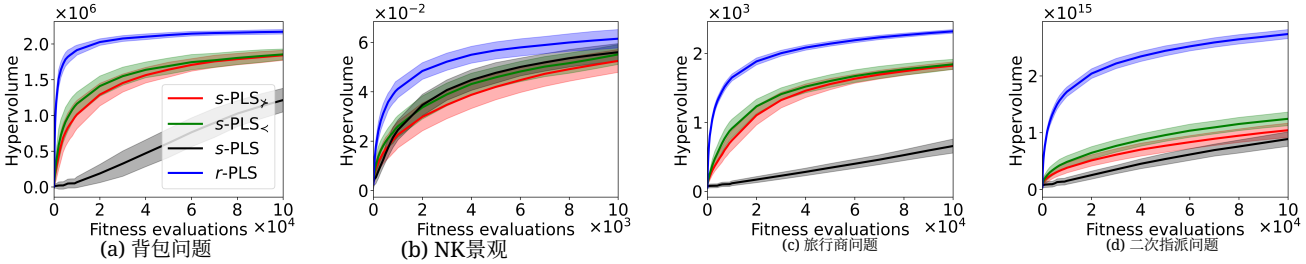


图 3：在四个具有 100 个决策变量的多目标组合优化问题上，s-帕累托局部搜索、s-帕累托局部搜索$\nsucc$（找到一个不受档案集支配的解时停止）、s-帕累托局部搜索$\prec$（找到一个支配档案集中至少一个解的解时停止），以及r-帕累托局部搜索在 30 次运行中的超体积轨迹（越高越好）。粗线和阴影区域分别表示超体积的均值和标准差。

而在旅行商问题和二次指派问题上，其收敛性和多样性都更好。

s-帕累托局部搜索进展缓慢的一种解释是其穷举邻域搜索（最佳改进策略）——它总是在继续之前评估一个解的整个邻域。相比之下，首次改进策略——找到满意解后即停止搜索——可能更快，如 [47, 21]所示。因此，我们现在考虑两种具有代表性的s-帕累托局部搜索替代方案，它们会更早停止邻域探索 [21]。第一种在找到非支配解（即不受档案集中任何解支配的解）时停止邻域探索，记为s-帕累托局部搜索$\nsucc$。另一种在找到支配解（即支配档案集中至少一个解的解）时停止，记为s-帕累托局部搜索$\prec$。

图6展示了两种s-PLS算法的平均超体积（粗线）和标准差（阴影区域），以及基础s-PLS和r-PLS在100D四个问题上的30次独立运行结果。更高维问题（200D和500D）的结果见附录C。如图所示，两种首次改进s-PLS算法在大多数问题上的超体积表现明显优于基础s-PLS。在这两种算法中，s- PLS$\prec$ 比s-PLS$\nsucc$更快。然而，它们的表现仍远逊于r-PLS。

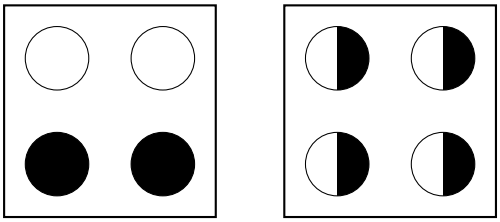
上述结果颇为有趣——作为一种简单的随机局部搜索启发式算法，r-PLS可能会多次重新访问解，但研究表明，它的效果显著优于s-PLS算法；后者采用系统搜索，避免重复探索和重新访问解。接下来，我们将探究这一现象的原因。

V. 玩具示例中的直观认识

本节将介绍两个档案集中解分布的玩具示例，以帮助说明为什么以及何时 r-帕累托局部搜索比 s-帕累托局部搜索更快。我们关注的是算法能以多快的速度（按评估次数计）找到一个“优良”解。优良解是指未被档案集支配的解，它会促使档案集更新，并提升档案集的质量。

A. 例*1*

设想这样一种情形：档案集中有四个解，其中一半有希望，另一半没有希望，如图4(a)所示。这里，有希望的解（图中的实心圆）是指其每个



(a) 例1　　(b) 例2

图4：档案集中解分布的两个玩具示例。这里，优良解（实心圆）表示其所有邻居均不被档案集中的解支配；不优良解（空心圆）表示其所有邻居均被档案集中的至少一个解支配；半优良解（半实心圆）表示其一半邻居不被档案集支配。在例1中，档案集包含两个优良解和两个不优良解。在例2中，档案集中的四个解都是半优良解。

邻居都是优良解（即相对于档案集而言非支配）；没有希望的解（空心圆）则是指其每个邻居都至少被档案集中的一个解支配。首先，s-帕累托局部搜索和r-帕累托局部搜索都会从档案集中均匀随机地选择一个解进行探索。

对于 r-帕累托局部搜索，如果选中了一个有希望的解，那么随机采样得到的邻居将是非支配解，搜索只需一次评估即可成功。如果选中了一个没有希望的解，那么随机采样得到的邻居将被支配；这会消耗一次评估，然后算法继续执行。由于选中有希望的解的概率为 $1/2$，找到一个优质邻居所需的预期评估次数为 2。

对于 s-PLS，如果选中一个有希望的解，选取的第一个邻居必须是非支配解，搜索只需一次评估便能成功。然而，如果选中一个没有希望的解，s-PLS 会穷举评估其整个邻域，在将其标记为已探索并选择另一个解进行探索之前，浪费 $|\mathcal{N}|$ 次评估，其中 $|\mathcal{N}|$ 表示邻域大小。因此，尽管两种算法选中有希望的解的概率均为 50%，但选中一个没有希望的解后，s-PLS 可能付出的代价要高得多。事实上，在这种情况下，s-PLS 找到一个优良解所需的期望评估次数为 $\frac{n|\mathcal{N}|}{n+2}+1$，其中 n 是档案集中的解数。证明见附录E。可以计算得出，对于邻域大小为 10 的问题，在我们的示例中，s-PLS 找到一个优良解所需的期望评估次数为 $\frac{4\times10}{4+2}+1=7.67$，远

高于*r*-PLS 所需的期望评估次数（即 2）。

B. 例2

上述示例展示了邻居分布极不均匀的情况：一半的解只有优质邻居，而另一半的解只有较差的邻居。在例2中，我们展示相反的情况：每个解的优质邻居和较差邻居数量相同（图4(b)）。

对于*r*-帕累托局部搜索，找到一个新的优良解所需的期望评估次数仍为 2，因为从档案集中的每个解中选中优质邻居的概率为 $1/2$。对于 *s*-帕累托局部搜索，由于所有解的邻居分布都相同，其中一半是优质邻居，因此从一个解中选中优质邻居的概率为 $1/2$。若*s*-帕累托局部搜索第一次选择了较差的邻居，那么第二次选择该解的优质邻居的概率为 $\frac{|\mathcal{N}|}{2(|\mathcal{N}|-1)}$，因为算法不会重复访问同一个邻居。这个过程可以持续进行，直到该解只剩下优质邻居。事实上，*s*-帕累托局部搜索找到优质邻居所需的期望评估次数为 $\frac{|\mathcal{N}|+1}{(|\mathcal{N}|/2)+1}$，始终小于*r*-帕累托局部搜索所需的次数（即 2）。证明见附录E。

从上述两个例子可以看出，这两种帕累托局部搜索算法的表现取决于档案集中各解的邻居分布。如果解的邻居（按质量衡量）分布均匀，那么*s*-PLS速度更快；如果分布不均匀，那么*r*-PLS更具优势。但是，在常见多目标组合优化问题的帕累托局部搜索算法搜索过程中，邻居的分布是怎样的？下一节将尝试回答这个问题。

第六节. *s*-PLS与*r*-PLS中解的邻居分布

本节将研究*s*-PLS和*r*-PLS在多目标组合优化问题上的搜索过程中，解的邻居分布。具体而言，我们统计档案集中各解的优质邻居数量，并检验该数量是否符合已知的离散概率模型（例如均匀分布、二项分布和泊松分布）。需要注意的是，这是一种检验经验数据集是否符合已知离散分布的常见方法，例如拟合网络请求数量（已发现其符合齐夫分布） [10], 以及科学论文的引文计数 [65]（其符合几何分布）。本研究考察了常见的离散分布 [36],，这些分布以不同的概率尾部类型为特征：均匀分布（均衡分布）、齐夫分布 （具有多项式衰减的重尾分布）、几何分布 （具有指数衰减的轻尾分布）、泊松分布 （具有超指数衰减的轻尾分布）和二项分布（具有有界支撑和硬截断的轻尾分布）。按照惯例，我们通过显著性水平为 $\alpha = 5\%$ 的 χ^2 检验来考察拟合优度。有关分布拟合、不同分布的比较以及参数设置的详细信息，见附录D。

图5绘制了搜索过程中，档案集中解的优质邻居数量所对应的五种离散分布的拟合优度

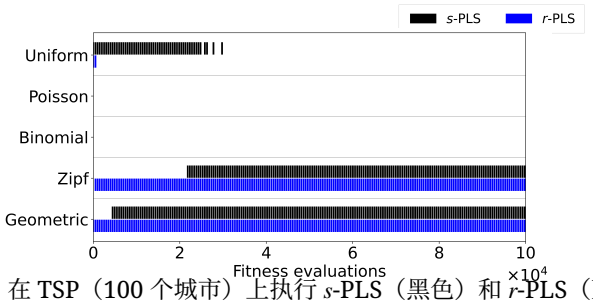


图5：在 TSP（100 个城市）上执行 *s*-PLS（黑色）和 *r*-PLS（蓝色）的搜索过程中，档案集中的解所拥有的优质邻居数量分布的拟合优度。某一行中的彩色刻度表示，在该点上，对应算法的数据未被模型拒绝。例如，对于几何分布模型，蓝色刻度几乎覆盖了整条色带。这表明，在整个搜索过程中，*r*-PLS 档案集中的解所拥有的优质邻居数量都服从几何分布。请注意，一个算法可能符合多个分布模型。

图5绘制了在旅行商问题上运行 *s*-帕累托局部搜索和*r*-帕累托局部搜索的搜索过程中，档案集中解的优质邻居数量所对应的五种离散分布的拟合优度。其他三个问题的结果见附录D。在图中，每条水平带对应一种候选分布；在每个抽样评估点，两个彩色刻度分别以黑色和蓝色表示*s*-帕累托局部搜索和*r*-帕累托局部搜索在 $\alpha = 0.05$ 处拟合优度良好，即 χ^2 检验未拒绝该分布。如图5所示，*s*-帕累托局部搜索起初遵循均匀分布，但在其大部分搜索过程中遵循几何分布。相比之下，*r*-帕累托局部搜索的表现同时符合齐夫分布和几何分布。请注意，多个分布都可能获得统计上可接受的拟合。为进一步比较齐夫分布和几何分布，我们使用赤池信息准则（AIC） [4], ，该准则同时考虑拟合的似然和模型复杂度（即对自由参数较多的分布施加惩罚）。在两者之中，几何分布的 AIC 始终较低，表明其拟合更佳（详情见附录D）。总之，在所有考察的问题上，几何分布都能最好地描述*s*-帕累托局部搜索和*r*-帕累托局部搜索的行为。

几何分布由概率质量函数 $\mathbb{P}(G = k) = p(1-p)^k$, 定义，其中 $k \in \mathbb{N}$ 和 $p \in (0, 1]$。它是轻尾分布，这意味着其概率质量会随着 k 增大而呈指数衰减。与均匀分布不同（例如上一节中的例2），在均匀分布中，解所拥有的优质邻居数量分布均匀；轻尾分布则会产生高度不均匀的分布——大多数解只有少量优质邻居，而拥有许多优质邻居的解虽然存在，却很少见。

七、理论分析

上述结果表明，在搜索过程中，几何分布模型最适合这两种PLS算法。本节在前述发现的基础上，旨在从理论上解释为什么*r*-PLS比*s*-PLS更高效。也就是说，在相同的几何分布下（即，相同的参数

值 p，该分布描述了档案中各个解的优质邻居数量），我们将从分析角度比较这两种算法找到新的优质邻居所需的期望时间。需要注意的是，尽管两种算法都符合相同的分布模型，但它们在特定时间步上很可能遵循不同的几何分布（即，不同的 p 值）。也就是说，考虑相同的分布能够让我们在相同的搜索状态下分析两者的效率，因此，性能差异完全可以归因于算法的邻域探索方式，即随机采样与系统性探索。

我们先介绍一个引理，它将在证明*s*-PLS 主定理时用到。

引理1. 假设一个包含 N 个（$N \in \mathbb{Z}^+$ 个）解的集合中有 k 个优良解（$1 \le k \le N$）。假设集合中的解按照随机顺序逐一访问，直到找到一个优良解为止。令 J 表示访问过的解的数量。那么，J 的期望值为 $\mathbb{E}[J] = \frac{N+1}{k+1}$。

证明. 这一过程等价于从 N 项中进行无放回均匀随机抽样，其中 k 项带有标记。直到遇到第一个带标记的项为止所检查的项数，其期望值已知为 $\mathbb{E}[J] = \frac{N+1}{k+1}$ [39, （第3.4节1]）。 □

该引理表明，要首先找到某个解的一个优质邻居，*s*-PLS 需要访问的邻居数的期望值为 $\frac{|\mathcal{N}|+1}{k+1}$，其中 $\mathcal{N}$ 表示问题的邻域大小，k 表示该解拥有的优质邻居数。

定理1 (*r*-帕累托局部搜索比*s*-帕累托局部搜索更快找到下一个优质解)。 设s-帕累托局部搜索和r-帕累托局部搜索的档案集中，解所包含的优质邻居数量服从参数为 $p \in (0,1]$，即 $\mathbb{P}(G = k) = p(1-p)^k$的几何分布，$k \in \mathbb{N}$. 那么，对于任意邻域大小$|\mathcal{N}| \in \mathbb{Z}^+$，r-帕累托局部搜索比s-帕累托局部搜索以更短的期望时间（更少的评估次数）找到下一个优质解。

证明。 对于*r*-帕累托局部搜索，在每次迭代中，都会从档案集中随机选取一个解，并对其随机抽取一个邻居。此时，找到优质邻居的概率为$p_{\text{success}} = \frac{\mathbb{E}[G]}{|\mathcal{N}|}$，其中 $\mathbb{E}[G]$ 表示优质邻居的期望数量。由于 G 服从几何分布，因此$\mathbb{E}[G] = \frac{1-p}{p}$ [36]。所以，*r*-帕累托局部搜索找到下一个优质邻居所需的期望评估次数为：

$$\mathbb{E}[T_{r\text{-PLS}}] = \frac{1}{p_{\text{success}}} = \frac{|\mathcal{N}|}{\mathbb{E}[G]} = \frac{p|\mathcal{N}|}{1-p}. \tag{1}$$

现在考虑 *s*-帕累托局部搜索。首先计算从档案集中探索单个解所需的期望评估次数。有两种可能的情况。第一种情况是该解没有优质邻居；这种情况发生的概率为 $\mathbb{P}(G = 0) = p(1-p)^0 = p$，此时算法需要扫描整个邻域，因此期望评估次数为 $p|\mathcal{N}|$。第二种情况是该解包含 $k \ge 1$ 个优质邻居；这种情况发生的概率为 $\mathbb{P}(G = k) = p(1-p)^k$，此时算法平均需要访问 $\frac{|\mathcal{N}|+1}{k+1}$ 个邻居（引理1），因此期望评估次数为 $p(1-p)^k \cdot \frac{|\mathcal{N}|+1}{k+1}$。因此，探索一个解所需的总体期望评估次数如下。

$$\begin{aligned}
\mathbb{E}[T^*] &= p|\mathcal{N}| + \sum_{k=1}^{|\mathcal{N}|} p(1-p)^k \cdot \frac{|\mathcal{N}|+1}{k+1} \\
&= p|\mathcal{N}| + p(|\mathcal{N}|+1) \sum_{j=2}^{|\mathcal{N}|+1} \frac{(1-p)^{j-1}}{j} \quad (\text{let } j = k+1) \\
&= p|\mathcal{N}| + p(|\mathcal{N}|+1) \cdot \frac{1}{1-p} \Big(\sum_{j=1}^{|\mathcal{N}|+1} \frac{(1-p)^j}{j} - (1-p) \Big) \\
&= p|\mathcal{N}| + p(|\mathcal{N}|+1) \cdot \frac{1}{r} \sum_{j=1}^{|\mathcal{N}|+1} \frac{r^j}{j} - \frac{r}{r} p(|\mathcal{N}|+1) \\
&= \frac{p|\mathcal{N}|+1}{1-p} \sum_{j=1}^{|\mathcal{N}|+1} \frac{(1-p)^j}{j} - p
\end{aligned}$$

现在我们已知探索单个解所需的期望评估次数，将该值乘以从档案集中已探索解的期望数量，即可得到期望评估总次数。由于随机选取的解至少有一个优质邻居的概率为 $1 - P(G = 0) = 1 - p$，因此已探索解的期望数量为 $\frac{1}{1-p}$。因此，*s*-帕累托局部搜索找到下一个优质邻居所需的期望评估次数为：

$$\begin{aligned}
\mathbb{E}[T_{s\text{-PLS}}] &= \mathbb{E}[T^*] \cdot \frac{1}{1-p} \\
&= \frac{p(|\mathcal{N}|+1)}{(1-p)^2} \sum_{j=1}^{|\mathcal{N}|+1} \frac{(1-p)^j}{j} - \frac{p}{1-p}.
\end{aligned} \tag{2}$$

现在，我们通过考察*s*-帕累托局部搜索与*r*-帕累托局部搜索所需评估次数之比，对二者进行比较。

$$\begin{aligned}
\frac{\mathbb{E}[T_{\text{s-PLS}}]}{\mathbb{E}[T_{\text{r-PLS}}]} &= \frac{\frac{p(|\mathcal{N}|+1)}{(1-p)^2} \sum_{j=1}^{|\mathcal{N}|+1} \frac{(1-p)^j}{j} - \frac{p}{1-p}}{\frac{p|\mathcal{N}|}{1-p}} \\
&= \frac{(|\mathcal{N}|+1) \sum_{j=1}^{|\mathcal{N}|+1} \frac{(1-p)^j}{j} - (1-p)}{|\mathcal{N}|(1-p)} \\
&= \frac{|\mathcal{N}|(1-p) + (|\mathcal{N}|+1) \sum_{j=2}^{|\mathcal{N}|+1} \frac{(1-p)^j}{j}}{|\mathcal{N}|(1-p)} \\
&= 1 + \frac{|\mathcal{N}|+1}{|\mathcal{N}|} \cdot \frac{\sum_{j=2}^{|\mathcal{N}|+1} \frac{(1-p)^j}{j}}{1-p} > 1
\end{aligned} \tag{3}$$

□

上述证明表明，与*s*-帕累托局部搜索相比，*r*-帕累托局部搜索找到优良解所需的期望评估次数更少。该结果并未区分*s*-帕累托局部搜索中的最佳改进策略与首次改进策略，因为它只关注识别算法最先遇到的优良解。

进一步考察这两种算法的期望评估次数之比 $\frac{\mathbb{E}[T_{s\text{-PLS}}]}{\mathbb{E}[T_{r\text{-PLS}}]}$，第二项的最后一部分 $\frac{}{\sum_{j=2}^{|\mathcal{N}|+1} \left(\frac{(1-p)^j}{j} \right)}$ 可改写为 $\sum_{j=2}^{|\mathcal{N}|+1} \left(\frac{(1-p)^{(j-1)}}{j} \right)$。该值随 $(1-p)$ 增大（p 减小）而增大，说明当优质邻居较多时，*s*-帕累托局部搜索与*r*-帕累托局部搜索之间的差距会进一步扩大。这一结果证实了先前的观察：在搜索的早期阶段（见图 1），*r*-帕累托局部搜索与*s*-帕累托局部搜索之间的性能差距更为显著；此时，在当前解附近更容易找到优质邻居。

注意到，在单目标优化中，随机化局部搜索与系统化局部搜索之间存在一种权衡，这种方法称为多重选择中的最佳项（BMS） [11]。BMS 扩展了

随机邻居采样的思路：从一个解中抽取 k 个邻居，并保留其中最好的一个。在此处采用的概率模型下，BMS 是*r*-帕累托局部搜索的一种自然推广（对应于 $k = 1$）。BMS 的期望评估次数为 $\mathbb{E}[T_{r-\mathrm{PLS}}] \cdot (1 + O(k/|\mathcal{N}|))$（见附录 F），略高于*r*-帕累托局部搜索。

值得指出的是，上述定理给出的是相邻时间间隔比较。在优质邻居数量的分布相同的情况下，*r*-帕累托局部搜索总是比*s*-帕累托局部搜索更快找到新的优良解。这并非运行结束时的分析，因此并不意味着搜索终止（且运行时间充足）后，*r*- 帕累托局部搜索得到的最终解必然优于*s*-帕累托局部搜索得到的解。事实上，随着搜索推进，解的邻域结构会持续变化，优质邻居也会变得越来越稀少。由于*r*-帕累托局部搜索找到优良解的速度更快，相关分布可能变化得更快（即几何分布中的 p 不断增大），从而可能使发现新的优良解变得更加困难。尽管如此，虽然我们无法判断哪种算法最终更优，但在搜索的早期阶段，当分布相近时，*r*-帕累托局部搜索远快于*s*- 帕累托局部搜索；这一优势在大规模问题上尤为明显，因为更大的邻域大小 $|\mathcal{N}|$ 会带来更高的比率（公式 3）。200D 和 500D 问题上的结果也印证了这一点（见附录C）。

VIII. 结论

本研究通过实证和理论分析表明，在两种标准的多目标局部搜索方法中，随机化方法比系统化方法更高效。因此，当搜索效率优先时，尤其是在大规模问题中，我们建议在多目标优化中采用随机局部搜索。

论文通过实验发现，PLS 搜索过程中遇到的优质邻居数量服从重尾分布（具体而言，是几何分布），而非均匀分布或类高斯分布。这一观察与许多优化问题中的普遍规律一致：高质量解较为罕见，因此随着搜索推进，优质邻居的数量会逐渐减少。不过，对于某些伪布尔基准测试 [46]，例如 OneMinMax [28], LOTZ [41], OJZJ [19], 这一规律可能并不成立。在这种情况下，系统化 PLS 可能更快。

参考文献

[1] Hern´an Aguirre 和 Kiyoshi Tanaka。MNK 景观上多目标进化算法的工作原理、行为与性能。《欧洲运筹学杂志》，181(3)：1670–1690，2007年。[2]Hernan Aguirre 和 Kiyoshi Tanaka。多目标 MNK 景观上的随机位爬山算法：记忆与种群爬山的影响。*IEICE* 基础电子、通信与计算机科学汇刊，88(1)：334–345，2005。[3]Hern´an E Aguirre 和 Kiyoshi Tanaka。精英保留和种群爬山对多目标 MNK 景观的影响。载于*CEC*，第 449–456 页，2004。

[4] Hirotugu Akaike。《统计模型识别的新视角》。*IEEETrans. Autom. Control.*，19(6):716–723，1974。[5] Abdullah Alsheddy 和 Edward E. P. K Tsang。用于双目标优化的引导式帕累托局部搜索框架。载于 *CEC*，第 1–8 页，2010。[6] Nawal Benabbou、Cassandre Leroy、Thibaut Lust 和 Patrice Perny。将偏好 elicitation 与局部搜索和贪心搜索相结合，用于拟阵优化。载于 *AAAI*，第 12233–12240 页，2021。[7] Nicola Beume、Boris Naujoks 和 Michael Emmerich。SMS-EMOA：基于支配超体积的多目标选择。《欧洲运筹学杂志》，(3):1653–1669，2007。[8] Chao Bian、Chao Qian 和 Ke Tang。多目标进化算法运行时间分析的一般方法。载于 *IJCAI*，第 1405–1411 页，2018。[9] Aymeric Blot、Marie-´El´eonore Kessaci 和 Laetitia Jour- dan。多目标组合优化元启发式算法中局部搜索技术的综述与统一。*JournalofHeuristics*，24(6):853–877，2018。[10] L. Breslau、Pei Cao、Li Fan、G. Phillips 和 S. Shenker。网络缓存与齐夫型分布：证据与启示。载于 *IEEE*计算机与通信学会年度联合会议，第 126–134 页，1999年。[11] Shaowei Cai。复杂性与质量之间的平衡：大规模图最小顶点覆盖的局部搜索。载于 *AAAI*，第 747–753 页，2015。[12] Sara Ceschia、Luca Di Gaspero 和 Andrea Schaerf。教育排课：问题、基准测试与最新成果。《欧洲运筹学杂志》，308(1):1–18，2023。[13] Jiejiang Chen、Shaowei Cai、Shiwei Pa n、Yiyuan Wang、Qingwei Lin、Mengyu Zhao 和 Minghao Yin。Nuqclq：最大拟团问题的有效局部搜索算法。载于 *AAAI*，第 12258–12266 页，2021。[14] Francisco Chicano、Darrell Whitley 和 Renato Tinos。约束伪布尔优化问题的高效爬山算法。载于 *GECCO*，第 309–316 页，2016。[15]Fabio Daolio、Arnaud Liefooghe、S´ebastien Verel、Hern´an Aguirre 和 Kiyoshi Tanaka。多目标 NK 景观上的全局搜索与局部搜索：问题特征影响的对比。载于 *GECCO*，第 369–376 页，2015。[16]Kalyanmoy Deb、Amrit Pratap、Sameer Agarwal 和 T Meyarivan。一种快速且具有精英保留机制的多目标遗传算法：NSGA-II。*IEEE Trans. Evol.Comp.*，6(2):182–197，2002。[17] Bilel Derbel、Arnaud Liefooghe、Qingfu Zhang、Hernan Aguirre 和 Kiyoshi Tanaka。基于分解的多目标局部搜索。载于 *PPSN*，第 431–441 页，2016。[18] M. Dinot、B. Doerr、U. Hennebelle 和 Will S.。存在噪声时多目标进化算法的运行时间分析。载于 *IJCAI*，第 5549–5557 页，2023。[19] Benjamin Doerr 和 Weijie Zheng。多目标进化算法在多目标问题上的理论分析。

目标问题。载于 AAAI，第 12293–12301 页，2021。[20] M˘ad˘alina M Drugan 和 Dirk Thierens。随机帕累托局部搜索：帕累托邻域探索与扰动策略。Journal of Heuristics，18:727–766，2012。[21]J´er´emie Dubois-Lacoste、Manuel L´opez-Ib´a˜nez 和 Thomas St¨utzle。随时帕累托局部搜索。《欧洲运筹学杂志》，243(2):369–385，2015。[22] Juan J. Durillo 和 Antonio J. Nebro。jMetal：用于多目标优化的 Java框架。《工程软件进展》，42(10):760–771，2011。[23] Agoston E Eiben 和 James E Smith。《进化计算导论》。施普林格，2015。[24] Jos´e Rui Figueira、Carlos M Fonseca、Pascal Halffmann、Kathrin Klamroth、Lu´ıs Paquete、Stefan Ruzika、Britta Schulze、Michael Stiglmayr 和 David Willems。说它们难很容易，但看出它们易却很难——迈向易解多目标组合优化问题的分类。Journal of Multi-Criteria Decision Analysis，24(1-2):82–98，2017。[25] Mengxing Gao、ChenGuang Liu 和 Xi Chen。具有额外资源和软优先约束的双目标非相关并行机调度问题。《欧洲运筹学杂志》，325(1):53–66，2025年。[26]Jaroslav Garvardt、Niels Gr¨uttemeier、Christian Komusiewicz 和 Nils Morawietz。针对 max c-cut 的参数化局部搜索。载于 IJCAI，第 5586–5594 页，2024。[27] O. Giel。一个简单多目标进化算法的期望运行时间。载于 CEC，第 1918–1925 页，2003。[28] Oliver Giel 和 Per Kristian Lehre。种群对进化多目标优化的影响。载于 GECCO，第 651–658 页，2006。[29]Niels Gr¨uttemeier、Christian Komusiewicz 和 Nils Morawietz。通过在拓扑排序之上进行参数化局部搜索，高效学习贝叶斯网络结构。载于 AAAI，第 12328–12335 页，2021。[30] Zhenyu Guo、Mingyu Xiao、Yi Zhou、Dongxiang Zhang 和 Kian-Lee Tan。利用有效的局部搜索增强平衡图边划分。载于 AAAI，第 12336–12343 页，2021。[31] Felipe Honjo Ide、Hernan Aguirre 和 Kiyoshi Tanaka。在大规模二元 MNK 景观上采用加权逐变异的多目标随机位爬山算法。载于 PPSN，第 169–185 页，2024。[32]Maarten Inja、Chiel Kooijman、Maarten de Waard、Diederik M Roijers 和 Shimon Whiteson。用于多目标优化的队列式帕累托局部搜索。载于 PPSN，第 589–599 页，2014。[33] Hisao Ishibuchi 和 Tadahiko Murata。一种多目标遗传局部搜索算法及其在流水车间调度中的应用。IEEE Trans. Syst., Man, Cybern.，28(3):392–403，1998。[34] Andrzej Jaszkiewicz。多目标遗传局部搜索在 0/1 背包问题上的性能——一项比较实验。IEEE Trans. Evol. Comp.，6(4):402–412，2002。[35] Andrzej Jaszkiewicz。多目标帕累托局部搜索。《欧洲运筹学杂志》，271(3):1001–1013，2018。[36]Norman L. Johnson、Samuel Kotz 和 Adrienne W. Kemp。《单变量离散分布》。John Wiley & Sons，第 3 版，2005 年。[37] Yuhao Kang、Jialong Shi、Jianyong Sun、Qingfu Zhang 和 Ye Fan。改进帕累托局部搜索邻域探索的新技术。《*Expert Syst.Appl.*》，254：124296，2024 年。[38] Joshua Knowles 和 David Corne。多目标二次分配问题的实例生成器与测试套件。载于《*EMO*》，第 295–310 页，2003 年。[39] Donald E. Knuth。《计算机程序设计艺术，第 *1* 卷：基本算法》。Addison-Wesley，1997 年。[40]M. Laumanns、L. Thiele 和 E. Zitzler。伪布尔函数上多目标进化算法的运行时间分析。《*IEEE Trans.Comp.*》，8(2)：170–182，2004 年。[41] Marco Laumanns、Lothar Thiele、Kalyanmoy Deb 和 Eckart Zitzler。在进化多目标优化中结合收敛性与多样性。《*Evol. Comput.*》，10(3)：263–282，2002 年。[42] Miqing Li、Tao Chen 和 Xin Yao。如何评估基于帕累托的搜索型软件工程中的解？批判性综述与方法论指导。*IEEE Trans. Evol. Comput. on SoftwareEngineering*，48(5)：1771–1799，2020 年。[43]Miqing Li、Xiaofeng Han、Xiaochen Chu 和 Zimin Liang。多目标组合优化问题中多目标进化算法与局部搜索的实证比较。载于《*GECCO*》，第 547–556 页，2024 年。[44] Miqing Li 和 Xin Yao。多目标优化中解集的质量评估：综述。《*ACMComputing Surveys*》，(2)，2019 年。[45] Jiaxin Liang、Junping Zhou 和 Minghao Yin。Diversat：一种用于多样化 SAT 问题的新颖且高效的局部搜索算法。载于《*AAAI*》，第 11290–11298 页，2025 年。[46] Zimin Liang 和 Miqing Li。运行时间分析中多目标伪布尔函数的问题特征。载于《遗传算法基础》，第 166–177 页，2025 年。[47] Arnaud Liefooghe、J´er´emie Humeau、Salma Mesmoudi、Laetitia Jourdan 和 El-Ghazali Talbi。基于支配的多目标局部搜索：在调度问题和旅行商问题上的设计、实现与实验分析。《*Journalof Heuristics*》，18：317–352，2012 年。[48] Peng Lin、Mengchuan Zou、Zhihan Chen 和 Shaowei Cai。Parailp：一种用于整数线性规划的并行局部搜索框架，采用协同进化机制。载于《*IJCAI*》，第 6949–6957 页，2024 年。[49] Thibaut Lust 和 Jacques Teghem。用于双目标旅行商问题的两阶段帕累托局部搜索。《*Journal of Heuristics*》，16(3)：475–510，2010 年。[50] Thibaut Lust 和 J acques Teghem。多目标多维背包问题：综述与一种新方法。《*Int. Trans. Oper. Res.*》，19(4)：495–520，2012 年。[51] Luis Paquete、Marco Chiarandini 和 Thomas St¨utzle。双目标旅行商问题中的帕累托局部最优集：一项实验研究。

旅行商问题：一项实验研究。载于《多目标优化元启发式算法》，第 177–199 页，2004 年。[52] Luis Paquete 和 Thomas St¨utzle。双目标旅行商问题的两阶段局部搜索。载于《*EMO*》，第 479–493 页，2003 年。[53] Quan Minh Phan 和 Ngoc Hoang Luong。帕累托局部搜索在多目标神经架构搜索中具有与进化算法相竞争的性能。载于《*GECCO*》，第 348–356 页，2023 年。[54] Zhongqiang Ren、Akshaya Kesarimangalam Srinivasan、Bhaskar Vundurthy、Ian Abraham 和 Howie Choset。用于多目标遍历搜索的帕累托最优局部优化框架。《*IEEE*机器人学汇刊》，39(5)：3452–3463，2023 年。[55] Celso C Ribeiro、Pierre Hansen、Pedro Castro Borges 和 Michael Pilegaard Hansen。多目标旅行商问题的全局凸性研究。《元启发式论文与综述》，第 129–150 页，2002 年。[56] V´ıvian Ludimila Aguiar Santos、Thales Francisco Mota Carvalho、Luciana Pereira de Assis、Miri Weiss-Cohen 和 Frederico Gadelha Guimar˜aes。基于分解的多目标迭代局部搜索：用于处理机器劣化效应的作业调度问题。《人工智能工程应用》，112：104826，2022 年。[57]Swakkhar Shatabda、MA Newton 和 Abdul Sattar。用于蛋白质结构预测的混合启发式局部搜索。载于《*AAAI*》，第 876–882 页，2013 年。[58] Jialong Shi、Jianyong Sun、Qingfu Zhang、Haotian Zhang 和 Ye Fan。利用协作并行策略改进多目标组合优化中的帕累托局部搜索。《*IEEE*控制论汇刊》，第 2369–2382 页，2022 年。[59] Jialong Shi、Qingfu Zhang 和 Jianyong Sun。PPLS/D：基于分解的并行帕累托局部搜索。*IEEE*控制论汇刊，50(3)：1060–1071，2020 年。[60]Zhouxing Su、Qingyun Zhang、Zhipeng L¨u、Chu-Min Li、Weibo Lin 和 Fuda Ma。用于最优摄像机布置的基于加权的变邻域搜索。载于《*AAAI*》，第 12400–12408 页，2021 年。[61]Rui Sun、Yiyuan Wang、HL Shimao Wang、Hui Li、Ximing Li 和 Minghao Yin。Nukplex：用于最大 k-plex 问题的高效局部搜索算法。载于《*IJCAI*》，第 7029–7037 页，2024 年。[62]Jacques Teghem。多目标组合优化问题：综述。《《多准则决策分析杂志》》，3：83–104，1994 年。[63] S´ebastien Verel、Arnaud Liefooghe、Laetitia Jourdan 和 Clarisse Dhaenens。多目标组合搜索空间的结构：具有相关目标的 MNK 景观。《《欧洲运筹学杂志》》，227(2)：331–342，2013 年。[64] S. Wietheger 和 B. Doerr。多目标进化算法近乎紧致的运行时间保证。载于《*PPSN*》，第 153–168 页，2024 年。[65] Zheng Xie、Zhenzheng Ouyang、Qi Liu 和 Jianping Li. 指数增长科学论文引文网络的几何图模型。*PhysicaA:StatisticalMechanics andits Applications*，456:167–175，2016年。[66] Zhenxuan Xie、Zhipeng L¨u、Zhouxing Su、Chu-Min Li、Junwen Ding 和 Yuxuan Wang。用于拉丁方补全问题的交换松弛型局部搜索。在*IJCAI*，第7047–7055页，2024。[67] Furong Ye、Chuan Luo 和 Shaowei Cai。通过随时性能分析改进 MaxSAT 随机局部搜索求解器的理解与配置。在*AAAI*，第27153–27160页，2025年。[68] Qingfu Zhang 和 Hui Li。MOEA/D：一种基于分解的多目标进化算法。*IEEETrans.Evol. Comp.*，11(6):712–731，2007年。[69]Zhen Zhang、Junfeng Yang、Limei Liu、Xuesong Xu、Guozhen Rong 和 Qilong Feng。探究局部搜索为何适用于公平中心表示聚类的理论基础。在*AAAI*，第16953–16960页，2024。[70] Jiongzhi Zheng、Kun He、Jianrong Zhou、Yan Jin、Chu-Min Li 和 Felip Many`a。将多臂老虎机与局部搜索相结合以求解 MaxSAT。*Artificial Intelligence*，338:104242，2025年。[71] W. Zheng、M. Li、R. Deng 和 B. Doerr。如何将 Metropolis算法用于多目标优化？在*AAAI*，第20883–20891页，2024。[72] Eckart Zitzler 和 Lothar Thiele。多目标进化算法：比较案例研究与强度Pareto方法。*IEEETrans. Evol. Comp.*，3(4):257–271，1999年。

附录

A. 多目标组合问题

我们考虑四种多目标组合优化问题：多目标0/1背包问题 [62], 旅行商问题（TSP） [55], 二次指派问题（QAP） [38]以及 NK 景观 [3]。每种问题都设置了三种规模（100、200 和 500 个变量）。

多目标0-1背包问题（背包问题）。多目标0-1背包问题 [62]是一种得到广泛研究的多目标组合优化问题。给定一组 D 项 $\mathbf{x} = (x_1, x_2, \ldots, x_D) \in \{0,1\}^D$，该 m目标问题定义如下。

$$\max \ f_j(x) = \sum_{i=1}^{D} v_{ji} x_i, \ j = 1, \ldots, m \quad \text{s.t.} \ \sum_{i=1}^{D} w_i x_i \leq c \tag{4}$$

这里， $v_{ji} \geq 0$ 表示物品在目标 j中的价值， i 表示该物品的重量， $c = \frac{1}{2}\sum w_i$ 表示容量。按照 [43]，， v_{ji} 和 w_i 均从 $\{10, 11, \ldots, 100\}$中均匀抽样。

多目标旅行商问题（TSP）。多目标旅行商问题扩展了经典旅行商问题，在每对城市之间设置多个成本 [55]，，旨在寻找一条路线，使访问所有城市各一次并返回起点的多个旅行成本最小化。形式上，给定一个

网络 $L=(V,\ C)$，其中 $V=\{v_1,\ldots,v_D\}$ 是一个包含 D 节点的集合，而 $C=\{C_j\colon\ j\in\{1,\ldots,m\}\}$ 是节点间（$C_j\colon\ V\times V$）的 m 成本矩阵集合，问题是找出能使每个 m 成本目标最小化的哈密顿回路的帕累托最优集。

多目标二次指派问题 (QAP)。多目标 QAP [38]对具有多种流类型的设施位置进行指派。给定 m 成本矩阵$[C_{1,i,j}]$、.....、$[C_{m,i,j}]$和一个距离矩阵 $[L_{u,v}]$，一个解是一个排列 $x=(x_1,\ .....,\ x_D)$，其中 x_i 表示设施 i的位置。该问题定义如下。

$$\min\ f_k(x)=\sum_{i=1}^{D}\sum_{j=1}^{D}C_{k,i,j}L_{x_i,x_j},\quad k=1,\ldots,m \tag{5}$$

多目标 NK 景观。多目标 NK 景观 [3]因其可调节的崎岖度 [63]而被广泛使用。此处，N 表示位串的长度，K表示上位性阶数（即每个变量受 K 个其他变量影响，这些变量共同称为其基因位点）。为保持一致，我们将位串的长度记为 D。那么，在一个 m目标 NK 景观问题中，每个目标 f_j 定义为:

$$\max f_j(x)=\frac{1}{D}\sum_{i=1}^{D}c_{ij}(x_i,x_{k_{ij1}},...,x_{k_{ijK}}),\ j=1,\ldots,m. \tag{6}$$

其中，c_{ij} 表示第 i 个变量的适应度贡献，该变量受其基因位点中的 K 个其他变量影响，这些变量共同决定其对第 j 个目标的贡献。每个 c_{ij} 取决于 i变量以及其基因位点中变量的取值，因此有 2^{K+1} 种可能的输入组合及相应的输出值。每个输出均从 $(0,1]$ 中随机抽取。根据 [1, 15], ，对于每个 i （变量）和 j （目标），变量基因位点中的 K 个其他变量均独立且均匀随机抽取，这形成了一种随机上位性模式。

B. 实验设置

在实验中，对于每个多目标组合优化问题，我们考虑三种问题规模（100、200 和 500）。需要注意的是，*r*-帕累托局部搜索和*s*-帕累托局部搜索通常采用不同的停止条件：*r*-帕累托局部搜索会一直运行，直到评估预算耗尽；而*s*- 帕累托局部搜索可能会在档案集中没有更多未探索的解时提前终止。为确保公平，所有算法都从相同的初始解开始。在某些情况下，*s*-帕累托局部搜索可能会很早终止，例如在 10^5 次评估后。在这种情况下，我们会在*s*-帕累托局部搜索终止时停止*r*-帕累托局部搜索。所有算法均使用 Java 在 jMetal 平台上实现 [22]。对于每个问题实例，每种算法均独立运行 30 次，每次运行使用一颗独立的核心，运行于 AMD Ryzen R9-7945HX CPU 上，并分配 16 GB 内存。

针对每个问题，邻域算子都与其变量的编码方式相匹配：NK景观使用 1 位翻转（二进制编码），TSP 使用 2-opt（基于顺序的排列编码），QAP 使用 2-swap（基于位置的排列编码） [23]。根据本研究的做法，背包问题需要采用 2 位翻转邻域，因为在约束边界附近，1 位翻转往往无法得到可行或更优的解 [43]。

我们使用超体积（HV）指标 [72], 评估性能，参考点通过随机采样估算。具体而言，我们从决策空间中生成 100,000 个随机解，并仅保留非支配解。根据 [42], ，对于最小化问题，我们将参考点定义为 $r_i=max_i+(max_i-min_i)/10$；对于最大化问题，则定义为 $r_i=min_i-(max_i-min_i)/10$，其中 max_i 和 min_i 分别表示非支配解集在第 i 个目标上的最大值和最小值。与一种常见做法相比，这种方式更为合适：该做法使用所有生成解的合并非支配解集来确定参考点，因为在多目标组合优化中，不同算法的收敛程度（即与帕累托前沿的接近程度）可能存在显著差异 [43]。

C. 补充结果

本节给出了四种多目标组合优化问题在问题规模为 200 和 500 时的补充实验结果。

图 6 展示了基本 *s*-帕累托局部搜索（粗线）和 *r*-帕累托局部搜索，以及两种*s*-帕累托局部搜索变体在四种问题上进行 30 次独立运行时的平均超体积（粗线）和标准差（阴影区域）。上排对应 200 个变量的问题，下排对应 500 个变量的问题。在所有问题类型中，*r*-帕累托局部搜索在整个搜索过程中始终比所有*s*-帕累托局部搜索变体取得更高的超体积值。它的最终性能也更好——根据显著性水平为 0.05 的 Wilcoxon秩和检验，*r*-帕累托局部搜索的最终 HV值显著更高。在 NK景观问题上，*r*-帕累托局部搜索的性能优势相对较小；而在其他三种问题上，*r*-帕累托局部搜索从搜索过程一开始就展现出明显优势，尤其是在问题规模增大时。还值得注意的是，在大多数问题上（NK景观问题除外），*s*-帕累托局部搜索$\prec$和*s*-帕累托局部搜索$\not\prec$（首次改进变体）的速度快于原始的*s*-帕累托局部搜索（最佳改进），这与先前的研究结果一致 [47, 21]。

图7展示了在图6所示搜索结束时，四种算法在四个问题的一次典型运行中得到的非支配解。版式与图6相同：较小规模的实例位于上排，较大规模的实例位于下排。对于背包问题和NK景观问题（最大化），较优的解位于右上角附近；对于旅行商问题和二次指派问题（最小化），理想区域则位于左下角。如图所示，与所有*s*-PLS变体相比，*r*-PLS得到的解在收敛性和多样性方面更优。尤其是原始*s*-PLS，其解在目标空间中的分布范围往往明显小于其他算法。

值得指出的是，对于所有问题规模，*s*-PLS 在 TSP 和 QAP 上表现出的多样性都低于在 NK景观 上的表现（此处见图7，正文中见图2）。*s*-PLS 采用系统化的邻域探索，评估

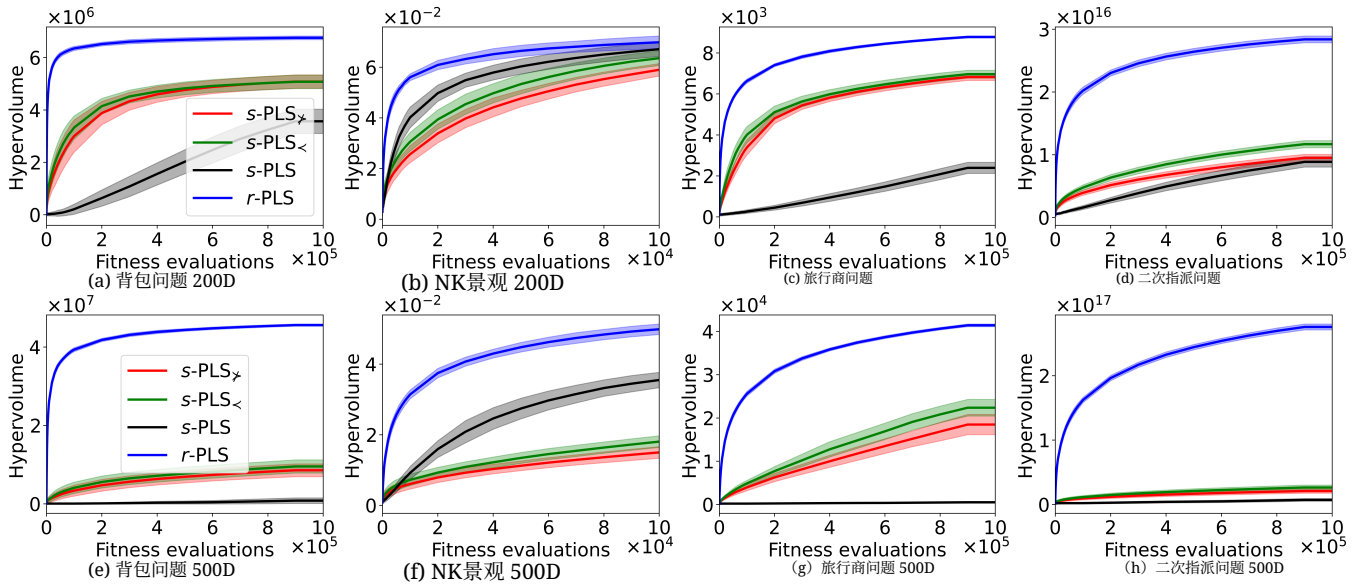


图 6：在四个具有 200 个变量（上图）和 500 个变量（下图）的多目标组合优化问题上，所考察的*s*-PLS、*s*-PLS≯、*s*-PLS≺ 和*r*-PLS在 30 次运行中的超体积轨迹（越高越好）。粗线和阴影区域分别表示超体积的均值和标准差。

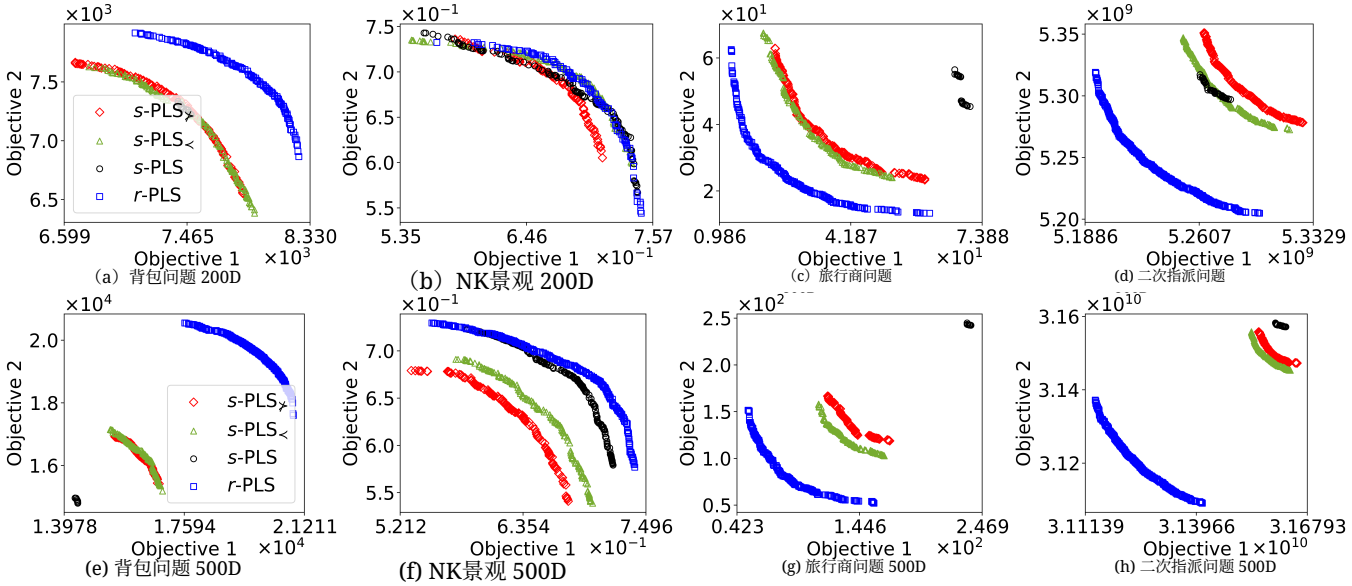


图 7：在四个具有 200 个变量（上图）和 500 个变量（下图）的多目标组合优化问题上，所考察的*s*-帕累托局部搜索、*s*-帕累托局部搜索≯、*s*-帕累托局部搜索≺ 和*r*-帕累托局部搜索在一次典型运行中得到的所有非支配解（即档案集中的解）。其中，背包问题和NK景观是最大化问题，而旅行商问题和二次指派问题是最小化问题。

一个解的所有邻居——其有效性取决于这些邻居的多样性。在 NK景观 上，随机的上位相互作用 [3] 使每次比特翻转带来的目标变化难以预测，因此即使从同一个解出发，也会在目标空间中产生不同的方向。然而，在 TSP 和 QAP 上，邻域内并不存在这样的多样性。

上述结果表明，*r*-帕累托局部搜索在收敛性和多样性方面均表现更佳，且随着问题规模增大，这一优势愈发明显。

D. s-帕累托局部搜索与r-帕累托局部搜索中解的邻居分布

本节详细介绍如何研究*s*-PLS 和*r*-PLS 的优质邻居数量分布。具体而言，我们统计在四个多目标组合优化问题上运行*s*-PLS 和*r*-PLS 的搜索过程中，档案集中各解的优质邻居数量，并检验该数量是否符合某种已知的离散概率模型。本文考虑的分布模型 [36] 是最常见的几种，其特征在于概率尾部类型各不相同：均匀分布 （分布均衡）、齐夫分布 （具有多项式衰减的重尾）、几何分布（具有指数衰减的轻尾）、泊松分布（具有超指数衰减的轻尾），

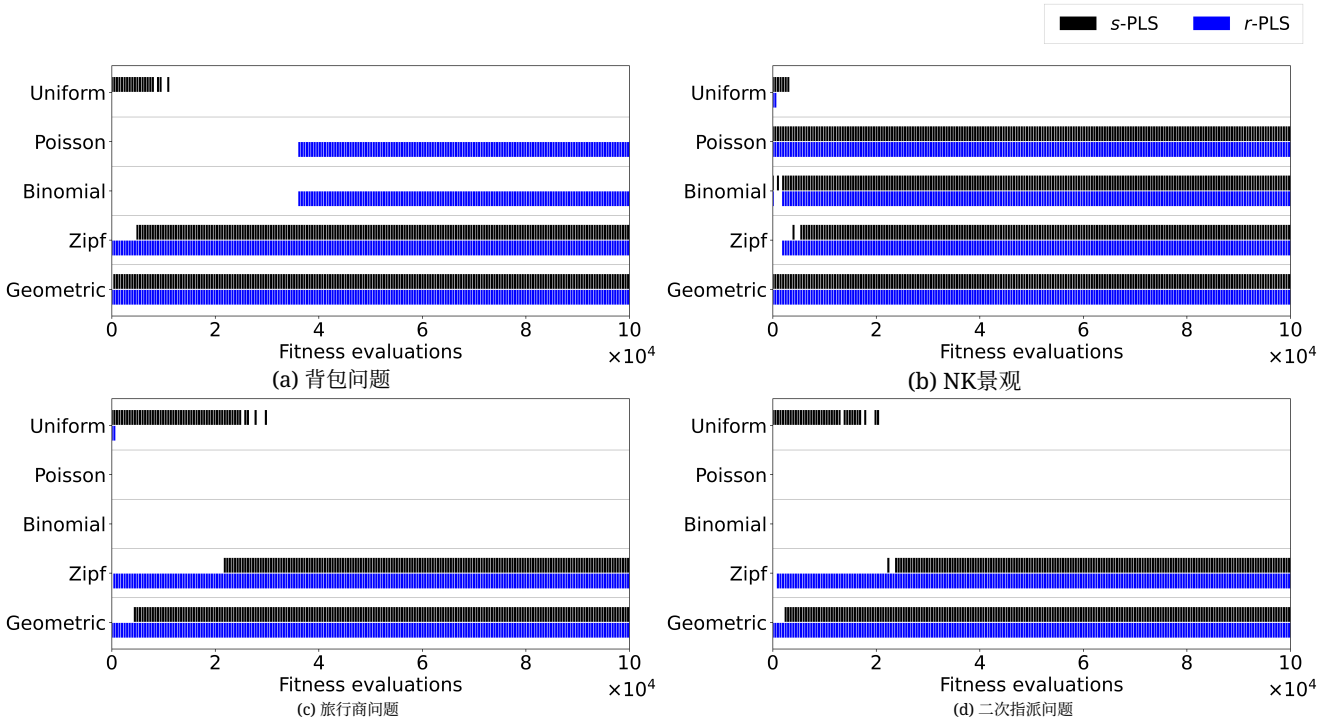


图 8：在 (a) 背包问题（100D）、(b) NK景观（N=100，K=10）、(c) 旅行商问题（100D）和 (d) 二次指派问题（100D）上，*s*-PLS（黑色）和*r*-PLS（蓝色）的搜索过程中，档案集中解的优质邻居数量分布的拟合优度。某一行中的彩色刻度表示，在该点对应算法的数据未被模型拒绝。

图 9：在 (a) 背包问题（100D）、(b) NK景观（N=100，K=10）、(c) 旅行商问题（100）和 (d) 二次指派问题（100个工厂）上，*s*-PLS（黑色）和*r*-PLS（蓝色）的搜索过程中，档案集中解的优质邻居数量分布的拟合优度。某一行中的彩色刻度表示，在该点对应算法的数据未被模型拒绝。

以及二项分布 （具有有界支撑和硬截断的轻尾）分布。

按照常见做法，我们在显著性水平为 $\alpha = 5\%$ 时，使用 χ^2 拟合优度检验评估绝对拟合质量。如果检验未能拒绝“观测数据来自所检验分布”这一原假设，则认为拟合可以接受。各分布的参数均通过最大似然法估计。对于泊松分布，速率参数 λ设为样本均值。同样，对于几何分布和二项分布，成功概率 p 也根据样本均值进行估计。对于齐夫分布，指数参数 s通过在区间 $[1.01, 10]$内对负对数似然进行数值最小化来估计。

图 9 展示了在四个多目标组合优化问题中，*s*-PLS 和 *r*-PLS 的搜索过程中，档案集中各解的优质邻居数量与五种离散分布的拟合优度。每条水平带对应一种候选分布；在每个采样评估点，黑色和蓝色标记分别表示在 $\alpha = 0.05$ 处，*s*-PLS 和 *r*-PLS 的拟合效果良好，也就是说， χ^2 检验未拒绝该分布。从图中可以看出，在几乎所有问题实例中，均匀分布仅在搜索初期适用于*s*-PLS。相比之下，几何分布几乎在所有问题的所有水平带上都通过了 χ^2 检验适用于*s*-PLS 和*r*-PLS，这表明优质邻居的数量符合几何分布。

请注意，在图9所示的所有问题中，齐夫分布与几何分布的拟合次数大致相当。为了比较这两种分布的相对拟合质量，我们使用赤池信息准则（AIC） [4]，该准则通过对具有更多自由参数的模型施加惩罚，在拟合优度与模型复杂度之间进行平衡。对于每种分布，每个模型的自由度计算如下： $n_{\text{bin}} - 1 - k$ ，其中 n_{bin} 表示至少拥有一个优质邻居的解的数量， k 表示该分布的自由参数数量。AIC越低，表示模型对数据的拟合越好。图10绘制了四个问题中*s*-PLS和*r*-PLS搜索过程期间几何分布与齐夫分布的AIC轨迹。实线和虚线分别对应几何分布与齐夫分布。如图所示，在所有情况下，实线都位于虚线下方，表明在*s*-PLS和*r*-PLS的整个搜索过程中，几何分布始终比齐夫分布提供更好的拟合。

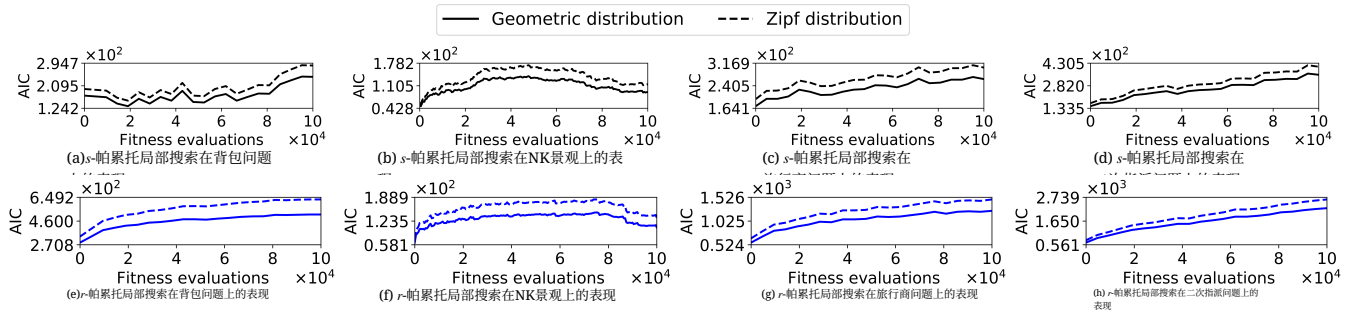


图10：在具有100个变量的四个多目标组合优化问题上，s-帕累托局部搜索（上图）和r-帕累托局部搜索（下图）的搜索过程中，几何分布（实线）与齐夫分布（虚线）的赤池信息准则（AIC） [4]变化轨迹（越低越好）。所有子图中，实线均位于虚线下方，表明几何分布始终比齐夫分布拟合得更好。

E. 两个示例的证明

在主论文中，我们通过两个玩具示例展示档案集中解的分布，以帮助说明为什么r-PLS可能比s-PLS更快。在例1中，档案集包含两个有潜力的解（每个解都只有优质邻居）和两个没有潜力的解（每个解都没有优质邻居）。在例2中，档案集中的四个解都处于中等潜力状态（每个解的优质邻居和劣质邻居各占一半）。这里，我们关注算法能够多快找到一个优质邻居（以评估次数计）。在本节中，我们给出证明，说明r-PLS在例1中更快，而s-PLS在例2中更快。

命题1 (r-帕累托局部搜索在示例1中比 s-帕累托局部搜索更快)**.** 设一个档案集包含 n 个解，其中恰有 $n/2$ 个解没有优质邻居（$G = 0$），其余 $n/2$ 个解的所有邻居都是优质邻居（$G = |\mathcal{N}|$）。那么，对于任意邻域大小 $|\mathcal{N}| \geq 2$，在找到下一个新的优良解时，r-帕累托局部搜索所需时间（评估次数）少于s-帕累托局部搜索。

证明。 对于r-帕累托局部搜索，算法从档案集中均匀随机地选择一个解。以$_1$ $_2$的概率，它会选中一个其整个邻域都由优质解组成的解。由于邻居也是均匀随机抽样的，因此选中的任何邻居都是优质解。因此，成功概率为：

$$p_{\text{succ}} = \tfrac{1}{2}, \quad \text{and} \quad \mathbb{E}[T_{r\text{-PLS}}] = \frac{1}{p_{\text{succ}}} = \frac{1}{1/2} = 2. \tag{7}$$

相比之下，s-帕累托局部搜索会按顺序扫描档案集，并将每个已访问的解标记为“已探索”。对于每个没有希望的解（即没有优质邻居的解），它会评估其所有 $|\mathcal{N}|$个邻居，然后再继续。这个过程会一直持续，直到从档案集中选中一个有希望的解。该过程等同于主论文引理1中所述的过程（即按顺序扫描邻域，找到第一个优质邻居）。因此，找到第一个有希望的解所需的期望评估次数为：

$$\mathbb{E}[J] = \frac{n+1}{(n/2)+1} = \frac{2(n+1)}{n+2}.$$

评估总次数包括此前每个未显示出潜力的解所需的 $|\mathcal{N}|$ 次评估（共 $\mathbb{E}[J]-1$ 个），以及第一个有潜力的解只需 1 次评估（因为它的所有邻居都很好）。因此：

$$\begin{aligned}\mathbb{E}[T_{s\text{-PLS}}] &= \big(\mathbb{E}[J]-1\big)|\mathcal{N}| + 1 \\ &= \left(\frac{2(n+1)}{n+2} - 1\right)|\mathcal{N}| + 1 = \frac{n|\mathcal{N}|}{n+2} + 1.\end{aligned} \tag{8}$$

由于 $|\mathcal{N}| \geq 2$，因此可得 $\mathbb{E}[T_{s\text{-PLS}}] > \mathbb{E}[T_{r\text{-PLS}}]$。 □

上述命题表明，在例1中，r-帕累托局部搜索比s-帕累托局部搜索更快地找到第一个优质解。关键原因在于，s-帕累托局部搜索选中一个没有希望的解时，可能会浪费大量评估次数，因为它会评估该解的所有邻居。相比之下，r-帕累托局部搜索每次只抽样一个邻居，从而避免了这种情况。然而，在例2中，档案集不含没有希望的解；相反，所有解都只有一半希望——每个解的邻居中，优质邻居和较差邻居各占一半。接下来我们将证明，在例2中，s-帕累托局部搜索比r-帕累托局部搜索更快。

命题2 （s-帕累托局部搜索比 r-帕累托局部搜索在例2中更快）。设档案集包含 n个解，并假设对于每个解，其邻居中恰有一半是优质邻居；即 $G = |\mathcal{N}|/2$。那么，对于任意邻域大小 $|\mathcal{N}| \geq 2$，s-帕累托局部搜索找到下一个新的优质解所需的时间（评估次数）少于r-帕累托局部搜索。

证明。 在r-帕累托局部搜索中，从档案集中均匀随机地选取一个解，并从其邻域中均匀随机地抽取一个邻居。由于每个解的邻居中恰有一半是优质邻居，成功的概率（即选中一个优质邻居的概率）为：

$$p_{\text{succ}} = \tfrac{1}{2}, \quad \text{and} \quad \mathbb{E}[T_{r\text{-PLS}}] = \frac{1}{p_{\text{succ}}} = \frac{1}{1/2} = 2. \tag{9}$$

对于s-PLS，该算法一次探索一个解的邻域，依次扫描其中的邻居，直到找到一个优质邻居。由于档案集中的所有解具有相同的邻域结构，因此只需考虑在单个邻域中进行的期望评估次数。在这种情况下，s-PLS 只需扫描一个邻域，因为所有邻域都有 $|\mathcal{N}|/2$ 个优质邻居。根据主论文中的引理1，当一半邻居是优质邻居时，在一个邻域中找到优质邻居所需的期望评估次数为：

在一个邻域中找到优质邻居所需的期望评估次数为:

$$\mathbb{E}[T_{s\text{-PLS}}] = \mathbb{E}[J] = \frac{|\mathcal{N}|+1}{(|\mathcal{N}|/2)+1} < 2 = \mathbb{E}[T_{r\text{-PLS}}].$$

□

F. 使用多重选择中的最优项的r-*PLS* 的证明

在主论文中，我们指出，在*r*-PLS 中使用多重选择中的最优项（BMS） [11] 需要 $\mathbb{E}[T_{r-\text{PLS}}]\cdot(1+O(k/|\mathcal{N}|))$ 次评估才能找到下一个优质邻居。本节给出这一结论的证明。

考虑一种配备多重选择中的最优项（BMS）的*r*-PLS 变体：每次迭代时，从档案集中均匀随机抽取一个解，并从该解的邻居中均匀地无放回抽取至多 k个（$1 \le k \le |\mathcal{N}|$）进行逐一评估，一旦找到优质邻居便停止；若经过 k次评估仍未遇到优质邻居，则停止。令 T_{BMS}表示找到下一个优质邻居所需的评估次数。

命题 3 （*r*-PLS 使用多重选择中的最优项时的期望时间）。设 $|\mathcal{N}| \in \mathbb{Z}^+$ 为邻域大小， G 为档案集中各解的优质邻居数量，其中 G服从参数为 $p \in (0,1]$的几何分布。设 T_{BMS} 为r-*PLS* 的 *BMS* 变体找到下一个优质邻居前所需的评估次数。那么

$$\mathbb{E}[T_{\text{BMS}}] > \mathbb{E}[T_{r\text{-}PLS}]$$

where $\mathbb{E}[T_{r\text{-}PLS}] = p|\mathcal{N}|/(1-p)$.

证明。 回顾主定理中的*r*-帕累托局部搜索，在给定 G 的条件下，成功概率为

$$q := \mathbb{E}\left[\frac{G}{|\mathcal{N}|}\right] = \frac{\mathbb{E}[G]}{|\mathcal{N}|} = \frac{1-p}{p|\mathcal{N}|},$$

且 $\mathbb{E}[T_{r\text{-PLS}}] = 1/q = p|\mathcal{N}|/(1-p)$。

接下来分析 BMS变体，它每次抽取一个解的 $k>1$ 个邻居。关于被选中的解，通常有两种情况： $G=0$ （没有优质邻居）和 $G>0$ （至少有一个优质邻居）。

我们先考虑 $G>0$ 的情况。每次 BMS 迭代开始时，都会选取一个具有 $G=g$ 个优质邻居的解。在该迭代中，假设已经检查了 h 个邻居，并发现它们都较差（$0 \le h \le k-1$）。下一次评估会从剩余的 $|\mathcal{N}|-h$ 个邻居中均匀分布地选取一个；这些邻居中仍包含相同的 g 个优质邻居（否则该迭代早已停止）。因此，在给定 $G=g$ 的条件下，

$$\Pr(\text{success at this evaluation} \mid G=g,\ h) = \frac{g}{|\mathcal{N}|-h}.$$

由于 $0 \le h \le k-1 \le |\mathcal{N}|-1$，我们有

$$|\mathcal{N}|-(k-1) \le |\mathcal{N}|-h \le |\mathcal{N}|,$$

因此

$$\frac{g}{|\mathcal{N}|} \le \frac{g}{|\mathcal{N}|-h} \le \frac{g}{|\mathcal{N}|-(k-1)}.$$

根据 $0 \le h \le k-1$，我们可以得到 BMS 过程中第 t 次评估的成功概率 q_t 的界（假设此前尚未成功）：

$$\mathbb{E}[G]/|\mathcal{N}| \le q_t \le \frac{\mathbb{E}[G]}{|\mathcal{N}|-(k-1)} \quad \forall t \ge 1.$$

代入*r*-帕累托局部搜索的成功概率，我们得到

$$q \le q_t \le \frac{q}{1-\frac{k-1}{|\mathcal{N}|}} \quad \forall t \ge 1.$$

现在根据 q_t的界来界定停止时间 T_{BMS}。令 $q_{\min} = q$ 和 $q_{\max} = q/(1-\frac{k-1}{|\mathcal{N}|})$。对于 BMS 过程,记 q_t 为第 t 次评估的成功概率（以此前均未成功为条件），我们有

$$\frac{1}{q_{\max}} \le \frac{1}{q_t} \le \frac{1}{q_{\min}} \quad \forall t \ge 1.$$

将方程代入，得到找到优质邻居之前的期望评估次数 $\mathbb{E}[T_{\text{bms}}]$,

$$\frac{1}{q_{\max}} \le \mathbb{E}[T_{\text{bms}}] \le \frac{1}{q_{\min}}.$$

将 $q_{\min}$ 和 $q_{\max}$ 代回，得到

$$\left(1-\tfrac{k-1}{|\mathcal{N}|}\right)\mathbb{E}[T_{r\text{-PLS}}] \le \mathbb{E}[T_{\text{bms}}] \le \mathbb{E}[T_{r\text{-PLS}}].$$

两边除以 $\mathbb{E}[T_{r\text{-PLS}}]$，得到

$$1-\tfrac{k-1}{|\mathcal{N}|} \le \frac{\mathbb{E}[T_{\text{bms}}]}{\mathbb{E}[T_{r\text{-PLS}}]} \le 1,$$

接着，我们加入不存在优质邻居的情形（$G=0$）：在已探索的 $\frac{1}{1-p}$ 个解中，以 p 的比例浪费 k 次评估,由此得到

$$\mathbb{E}[T_{\text{BMS}}] = \frac{kp}{1-p} + \mathbb{E}[T_{\text{bms}}].$$

代入上述不等式，可得

$$\frac{kp}{1-p} + (1-\tfrac{k-1}{|\mathcal{N}|})\mathbb{E}[T_{r\text{-PLS}}] \le \mathbb{E}[T_{\text{BMS}}] \le \frac{kp}{1-p} + \mathbb{E}[T_{r\text{-PLS}}]$$

$$\mathbb{E}[T_{r\text{-PLS}}] + \frac{\mathbb{E}[T_{r\text{-PLS}}]}{|\mathcal{N}|} \le \mathbb{E}[T_{\text{BMS}}] \le \frac{k}{|\mathcal{N}|}\mathbb{E}[T_{r\text{-PLS}}] + \mathbb{E}[T_{r\text{-PLS}}]$$

$$1+\frac{1}{|\mathcal{N}|} \le \frac{\mathbb{E}[T_{\text{BMS}}]}{\mathbb{E}[T_{r\text{-PLS}}]} \le 1+\frac{k}{|\mathcal{N}|}$$

由于 $1+\frac{1}{|\mathcal{N}|} \le \frac{\mathbb{E}[T_{\text{BMS}}]}{\mathbb{E}[T_{r\text{-PLS}}]}$，BMS变体比*r*-帕累托局部搜索慢。

□

值得注意的是，由于 $\frac{\mathbb{E}[T_{\text{BMS}}]}{\mathbb{E}[T_{r\text{-PLS}}]} \le 1+\frac{k}{|\mathcal{N}|}$，当 k较小时，BMS变体仅比*r*-帕累托局部搜索稍慢。