基于 EASE 的分区器选择：优化分布式图处理

Nikolai Merkel*、Ruben Mayer*、Tawkir Ahmed Fakir* 和 Hans-Arno Jacobsen†
*慕尼黑工业大学，†多伦多大学{nikolai.merkel、
ruben.mayer、tawkir.fakir}@tum.de
jacobsen@eecg.toronto.edu

arXiv:2304.04976v1 [cs.DC] 11 Apr 2023

摘要——针对海量图的分布式图处理，需要将图划分为多个大小相等的部分，并将这些部分分配到计算集群中的多台机器上。在过去十年中，人们开发了许多分区算法，它们在划分质量、分区运行时间以及最适用的图类型方面各不相同。图分区算法种类繁多，因此，为给定场景选择分区器是一项颇具挑战性的任务。已有多项研究对图分区算法的特性进行了定性分析，为选择算法提供参考。然而，要实现自动选择，就需要定量预测划分质量、分区运行时间以及后续图处理作业的运行时间。本文提出一种基于机器学习的方法，可对不同类型的边划分算法和图处理工作负载进行此类定量预测。我们表明，基于生成图进行训练能够达到较高的准确率，而使用真实世界数据还能进一步提高准确率。根据预测结果，与随机选择相比，自动选择平均可将端到端运行时间缩短 11.1%；与选择割大小最低的分区器相比，可缩短 17.4%；与最差策略相比，则可缩短 29.1%。此外，在 35.7% 的情况下，系统选择了最佳策略。

索引词—**图划分、自动分区器选择、分布式图处理、机器学习**

I. 引言

过去十年间，涌现出许多分布式图处理系统和数据库，用于处理包含数十亿条边的大型图，例如 Pregel [1]，Giraph [2], PowerGraph[3], PowerLyra [4]，GraphX [5]和 Neo4j [6]。为了实现分布式图处理，必须将图划分为多个大小相等的部分，并将其分配到计算集群中的多台机器上 [7]–[9]。每台机器对其分区执行计算，而机器之间的通信则通过网络进行。机器之间的通信量受划分质量影响，并会影响分布式图处理的性能 [9]–[12]。

许多图划分算法已经被开发出来，并可分类如下：流式算法 [4],[5], [13]–[23] 以边列表等形式对图进行流式处理，并根据割模型，依次动态地将边或顶点分配到各个分区。内存算法 [8], [9],[24]–[27]将整个图加载到内存中进行分区。混合算法 [11] 以内存方式对图的一部分进行分区，并以流式方式处理其余部分。

分区算法彼此的差异体现在划分质量、分区的运行时间以及最适合处理的图类型方面 [11], [12], [28]–[30]。

图分区算法种类繁多，因此要针对特定场景选择合适的分区器是一项颇具挑战性的任务。很难判断哪种分区器能在给定图上实现最佳划分质量，也难以估计两种分区器之间的质量差距。要选出能使端到端运行时间最低的分区算法则更为困难，这里的端到端运行时间包括分区运行时间和后续图处理运行时间。然而，分区算法的选择至关重要。正如我们在论文中所展示的，与最差选择相比，最佳选择可将端到端运行时间最多缩短 71%。

现有关于图划分算法的研究 [12], [28]–[30] 对这些算法的特性提供了有趣的定性见解。例如，研究表明，不存在适用于所有图处理工作负载的最佳分区器；分区运行时间并非总是可以忽略；图的度分布会影响划分结果。然而，这些研究不足以自动选择能够针对给定图和图处理工作负载最大限度缩短端到端时间的图划分器。要在多个分区器中自动进行选择，就需要对分区运行时间和图处理运行时间进行定量预测。例如，如果目标是最小化端到端时间，那么分区运行时间的预测值与图处理运行时间的预测值之和就至关重要。

我们提出了一种基于机器学习的方法，以提供此类预测，从而实现自动分区器选择，以最小化图处理运行时间或端到端处理运行时间。为此，我们在广泛的图上对不同类型的分区器进行分析，并测量分区运行时间、分区质量指标以及图的属性。然后，我们在经过划分的图上执行不同的图处理算法，以测量图处理运行时间。最后，我们利用分析所得的数据构建机器学习模型，以预测任意图在使用不同分区器时的分区运行时间和分区质量指标。此外，我们还训练了一个模型，根据预测的分区质量指标和图属性来预测不同图处理算法的运行时间。尽管图存在不同的形式化表示

© 2023 IEEE。允许将本材料用于个人用途。对于其他任何用途，无论是在当前还是未来的媒体中，包括为广告或促销目的重印/再版本材料、创作新的汇编作品、转售或重新分发至服务器或列表，或在其他作品中重复使用本作品中任何受版权保护的组成部分，均须获得 IEEE 的许可。最终版本发表于《2023年IEEE第三十九届国际数据工程会议论文集》（ICDE'23）。

分区方面，我们在这项工作中专注于边划分问题，因为许多分布式图处理框架 [3], [5], [31] 都普遍采用这一方法，且相关研究十分活跃 [7], [8], [11], [25]。

将机器学习用于上述预测任务的一项主要挑战是，需要训练数据覆盖各种可能的图（具有不同属性）。由于以下原因，获取足够且具有代表性的真实世界图用于训练可能很困难。首先，可用的真实世界图数量可能有限。其次，训练只能使用已知的图，但未来的工作负载中可能会出现具有不同属性的新图。第三，可能需要从不同来源下载不同格式的图，这是一个耗时且脆弱的过程。另一项挑战是为机器学习模型选择特征，以及如何对模型进行微调。

在我们的工作中，我们作出了以下贡献：

1) 我们提出了*EASE*，这是一种基于机器学习的系统，可预测给定图、图划分器和图处理算法的分区运行时间与处理运行时间。我们证明，这些预测能够实现自动边分区器选择。与随机选择相比，该方法将端到端运行时间平均缩短了 11.1%；与选择割大小最低的分区器相比，缩短了 17.4%；与最差策略相比，缩短了 29.1%。此外，在 35.7% 的情况下，系统选择了最佳策略。我们的方法具有可扩展性，因此无需重新训练整个模型，即可纳入新的分区器和图处理算法。2) 我们展示了如何通过使用图生成器，应对获取种类丰富的训练图这一挑战。我们提供图生成器及其设置，使模型无需依赖外部图数据集即可完成训练，因此我们的方法易于应用和复现。3) 我们说明了在条件允许时，通过使用真实世界图丰富合成图，可以进一步提高机器学习模型的预测准确度。这样，用户便可以纳入真实世界图，从而进一步提升机器学习模型在特定图类型上的性能。通过这种丰富方式，我们将复制因子的预测误差最多降低至原来的 $3.2\times$ 分之一，从而使丰富后的图类型的端到端时间缩短 5%。

本文其余部分安排如下。第 II 节介绍边划分问题、划分质量度量以及常见的图属性。第 III 节说明分区算法的选择对图处理性能的重要性。第 IV 节介绍我们基于机器学习的方法，用于预测划分质量、分区运行时间和处理运行时间。第 V 节评估我们的方法。第 VI 节讨论相关工作。最后，我们在第 VII 节总结全文。

II. 背景

令 $G = (V, E)$ 为一个由一组顶点V 和一组边 $E \subseteq V \times V$构成的图。边划分的目标是将 G 切分为 k 个分区。在边划分中[10]，边被划分为 k 个两两不相交的分区$P = \{p_1, \ldots, p_k\}$，且满足 $\cup_{i=1}^{k} p_i = E$。每个分区 p_i 覆盖一组顶点 $V(p_i) = \{v \in V | \exists u \in V : (u, v) \in p_i \vee (v, u) \in p_i\}$。分区覆盖的顶点还可进一步分为源顶点和目标顶点，定义如下：$V_{src}(p_i) = \{u \in V | \exists v \in V : (u, v) \in p_i\}$ 以及 $V_{dst}(p_i) = \{v \in V | \exists u \in V : (u, v) \in p_i\}$。被多个分区覆盖的顶点称为割点，需要进行复制。边划分的目标 [25] 是在边分区满足 α-平衡约束的前提下，尽可能减少复制顶点的数量：$\forall p_i \in P : |p_i| \leq \alpha \cdot \frac{|E|}{k}$。

A. 划分质量指标

存在不同的分区质量指标。首先，复制因子 定义为 $RF(P) = \frac{1}{|V|} \sum_{i \in [k]} |V(p_i)|$。它表示一个顶点跨越的分区平均数量。复制因子与通信开销 [10], [28]密切相关。其次，还有不同的平衡指标 用于衡量分区的均衡程度。在分布式图处理过程中，保持分区均衡很重要，这样可以避免机器过载并成为拖尾节点 [28]。在顶点割划分中，通常会考虑边平衡 （例如， [7]，[8], [10], [11], [16], [20], [21]）。此外，顶点平衡会影响处理性能 [8], [11] ，我们将在第 III 节中对此进行说明。对于有向图，如果计算或通信负载取决于每个分区中的源顶点或目标顶点数量，还可以进一步区分源顶点均衡 和目标顶点均衡。

设 $M = \{m_1, \ldots, m_k\}$ 为一个包含自然数的集合。我们将 $max(M)$ 定义为满足 $\forall y \in M : y \leq x$ 的最大元素 $x \in M$ s.t.。而 $avg(M) = \frac{1}{|M|} \sum_{i \in [k]} m_i$ 定义为该集合的平均值。现在，我们定义四种平衡指标。

1) *Edge balance:* $B_{edge}(P) = \frac{max(|p_1|,\ldots,|p_k|)}{avg(|p_1|,\ldots,|p_k|)}$
2) *Vertex balance:* $B_v(P) = \frac{max(|V(p_1)|,\ldots,|V(p_k)|)}{avg(|V(p_1)|,\ldots,|V(p_k)|))}$
3) *Source balance:* $B_{src}(P) = \frac{max(|V_{src}(p_1)|,\ldots,|V_{src}(p_k)|)}{avg(|V_{src}(p_1)|,\ldots,|V_{src}(p_k)|))}$
4) *Destination balance:* $B_{dst}(P) = \frac{max(|V_{dst}(p_1)|,\ldots,|V_{dst}(p_k)|)}{avg(|V_{dst}(p_1)|,\ldots,|V_{dst}(p_k)|))}$

在本工作中，我们使用机器学习模型来预测这五个分区质量指标（即复制因子和四个负载均衡指标），作为图处理运行时间预测的基础。

B. 图属性

图具有不同的属性。以下图属性被用作我们的机器学习模型的特征。在第IV-B节中，详细讨论了为什么这些图属性是我们预测任务中很有潜力的特征候选项。

1) 密度：密度定义为 $dens(G) = \frac{|E|}{|V| \cdot (|V|-1)}$，用于描述其中包含多少条边。

在图中，与使用给定顶点 [32] 可以创建的所有可能边相比。

2) 平均度：平均度定义为 $deg(G) = \frac{2 \cdot |E|}{|V|}$。

3) 平均三角形数：三角形是无向图 $G = (V, E)$ 的一个完全子图 $T = \{V_T, E_T\}$，恰好包含三个顶点 [33]。顶点的三角形数定义为 $t(v) = |\{T | v \in V_T\}|$。平均三角形数定义为 $t(G) = \frac{1}{|V|} \sum_{v \in V} t(v)$。4) 平均局部聚类系数 (LCC)：顶点 v 的局部聚类系数 [33]–[35] 定义为 $c(v) = \frac{t(v)}{0.5 \cdot deg(v) \cdot (deg(v)-1)}$，其中 $deg(v)$ 是 v 的度，而 $t(v)$ 是三角形的数量（见上文）。平均局部聚类系数定义为 $C(G) = \frac{1}{|V|} \sum_{v \in V} c(v)$。5) 偏度：对于图中每个顶点 v 都计算的所有指标（例如度），还可以用皮尔逊第一偏度系数描述这些值的偏度：$skew(values) = \frac{mean(values) - mode(values)}{\sigma}$，其中 σ 为这些值的标准差。

III. 图分区器选择

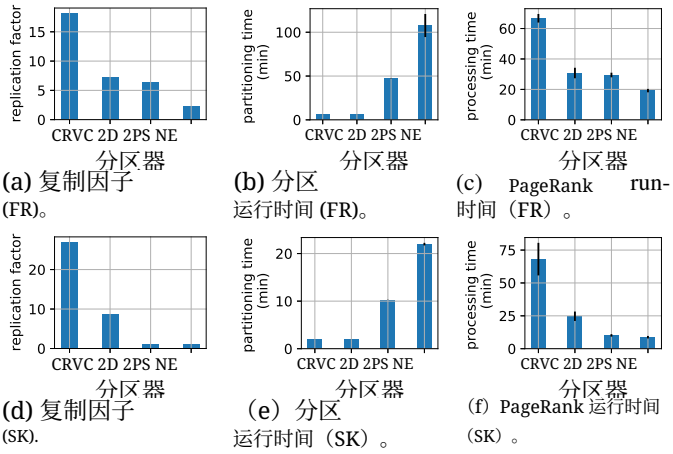
如第 I 节所述，各种分区算法之间存在很大差异。图划分算法的特性使得分区器选择面临复杂的权衡。在许多情况下，较好的分区质量会带来更快的分布式图处理，但实现这一点也可能需要付出更高的分区运行时间。

我们通过以下实验展示了这一点，实验涉及两种图处理算法：PageRank [36]和标签传播 [37]。我们选择 PageRank 作为一种通信受限型算法，因为它对复制因子较为敏感；选择标签传播作为一种计算受限型算法，因为它对负载均衡较为敏感。

A. PageRank

我们在一个包含 64 台机器的 Spark/GraphX 集群上，对图 *Friendster* [38] 和 *sk-2005* [39]–[41]运行了 50 轮 PageRank。*Friendster* 和 *sk-2005* 分别包含 1.8B和 1.9 B 条边，以及 66M 和 51 M 个顶点。这些图被划分为 64 个分区，并分别采用 2D [5],2PS [21] 和 NE [25] 作为无状态流处理、有状态流处理和内存分区算法的代表。此外，我们还使用 CRVC [5] 作为基线。图 1 报告了复制因子、分区运行时间和图处理运行时间。

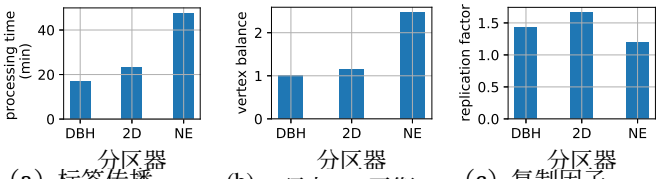
我们观察到，较好的复制因子会带来更短的处理运行时间。然而，产生较低.....的算法
复制因子会导致更长的分区运行时间。在这两个图上，采用 NE 的内存中分区在复制因子和图处理运行时间方面都远优于采用 2D 的无状态流式处理，但分区运行时间要长得多。采用 2PS 的有状态流式处理，其图划分运行时间介于 2D 和 NE 的运行时间之间，



(a) 复制因子 (FR)。 (b) 分区运行时间 (FR)。 (c) PageRank 运行时间（FR）。

(d) 复制因子 (SK)。 (e) 分区运行时间（SK）。 (f) PageRank 运行时间（SK）。

图 1：在 *Friendster*（*FR*） 和 *sk-2005*（*SK*） 图上进行 PageRank 计算时的图划分算法性能比较。



（a）标签传播运行时间（FB）。 (b) 顶点平衡 (FB)。 （c）复制因子 (FB)。

图 2：在*Socfb-A-anon(FB)* 图上进行标签传播计算的图划分算法性能比较。

而 2PS 的复制因子和图处理运行时间则取决于图的类型。在*sk-2005*上，2PS 与 NE 非常接近，因此也远优于 2D。在*Friendster*上，2PS 则远逊于 NE，与 2D 相近。这些示例表明，复制因子与图处理运行时间之间存在相关性。因此，复制因子是预测图处理运行时间的一项重要划分特征。

B. 标签传播

标签传播在图 *Socfb-A-anon*[42] 上运行 10次迭代，使用包含 4 台机器的 Spark/GraphX 集群。*Socfb-A-anon* 由 3.1 M 个顶点和24M条边组成。图使用二维 [5], DBH [16] 和 NE [25]进行分区，划分为 4个分区。在图2 中，我们报告了图处理运行时间、顶点平衡和复制因子。在此场景中，由于图划分运行时间相较于图处理运行时间可以忽略不计，因此未报告图划分运行时间。我们观察到，更好的顶点平衡（即接近1.0）会带来更低的处理运行时间。由于标签传播受计算限制，复制因子的重要性较低。我们得出结论，平衡指标可能是图处理运行时间预测的重要特征。

这些示例表明，对图处理运行时间的预测可能取决于不同的分区指标，并且

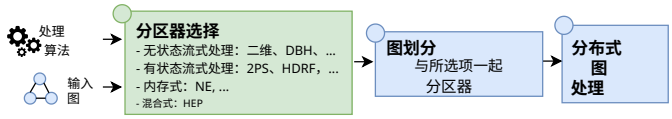


图 3：*EASE* 是图处理流水线的一部分，用于自动分区器选择。

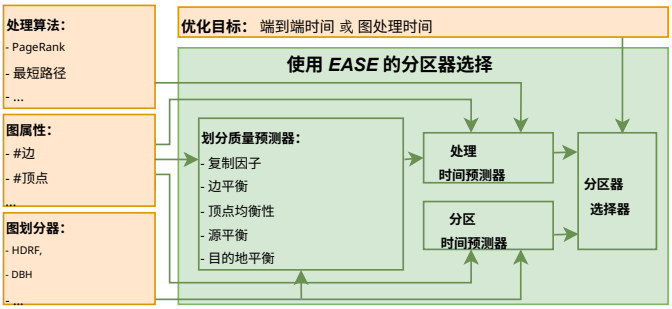


图 4：*EASE* ——系统概述。

分区运行时间在选择图划分器时也可能发挥重要作用。我们利用这些洞见设计了*EASE*，具体内容将在下一节介绍。

IV. 方法

动机：图 3 展示了一个由三个阶段组成的图处理流水线：首先，选择一个图划分器；其次，对图进行划分；最后，执行分布式图处理。分区器选择是该流水线中的一个重要阶段，因为它会影响图划分和图处理阶段的运行时间。然而，迄今为止，图划分器只能通过人工启发式方法和最佳实践来选择，这对缺乏经验的用户来说是一项颇具挑战性的任务。对于给定的图处理工作负载和图，很难判断哪种选择能带来最低的端到端运行时间。此外，选择阶段必须非常快速，以免拖慢流水线。为应对这些挑战，我们提出了一种名为***E**dge p**A**rtitioner **SE**lection* (*EASE*)的基于机器学习的系统，以实现高效的自动分区器选择。

方法概述：*EASE*的设计如图4 所示。该系统由四个组件组成：(1) 分区质量预测器，用于预测给定图上的不同分区器的不同分区质量指标（复制因子、边平衡、顶点平衡、目标平衡和源平衡）。(2) *PartitioningTimePredictor*，用于预测给定图上不同分区器的分区运行时间。(3) *ProcessingTimePredictor*，用于预测给定图处理算法在具有相应分区质量指标的已分区图上的图处理运行时间。(4) *PartitionerSelector*，基于对分区运行时间和处理运行时间的预测，自动选择分区器，以最小化处理运行时间或端到端运行时间。

训练阶段概述：*EASE* 的训练阶段由四个步骤组成（见图 5）。首先，收集或生成具有不同属性的图。其次，对这些图

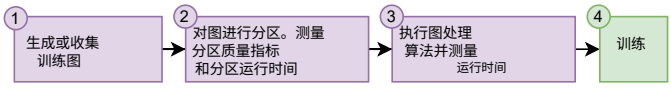


图5：EASE*EASE* 的训练阶段由四个步骤组成。

使用不同的分区算法划分为不同数量的分区，并测量分区质量指标和分区运行时间。第三，对每种图与分区器的组合执行不同的图处理算法，并测量处理运行时间。第四，使用获取的数据训练用于分区质量预测器、分区时间预测器 和处理时间预测器的机器学习模型。最后，可以应用这些模型来实现自动分区器选择。

下面，我们展示如何获取足够的训练数据，以涵盖具有不同图属性的广泛图集合（第 IV-A 节），以及为机器学习模型选择哪些图属性作为特征（第 IV-B 节）。随后，我们描述如何选择和微调机器学习模型（第 IV-C 节），以及如何将模型应用于给定的图问题（第 IV-D 节）。最后，我们讨论*EASE* 的设计决策（第 IV-E 节）。

A. 训练数据获取

数据获取的目标是涵盖预期作为工作负载、需要执行三项预测任务的未知真实世界图的广泛图属性。同时，这些图应易于获取。在第 I 节中，我们说明了获取足够数量的真实世界图可能面临多种挑战。我们通过使用合成图来应对这一挑战。这样一来，我们便能生成大量图，并涵盖广泛的图属性。这为构建机器学习模型奠定了基础。随后，我们在真实世界图上测试这些模型，并识别可能存在的弱点，例如某些图类型与分区算法的组合无法得出准确结果。针对模型的这些弱点，如果能获取真实世界图，就可以将其加入合成数据中，以此改进训练。

在我们的工作中，我们使用 R-MAT [43] 作为图生成器，并采用 Khorasani 等人 [44]的实现。R-MAT 轻量、可良好扩展，能够生成逼真的图 [45]–[47]，并用于著名的 Graph500 基准测试 [48]。根据 R-MAT 原论文 [43]，，分区 a和 d 可以视为由分区 b 和 c中的边连接起来的社区。递归生成过程会形成子社区。为了生成不同的图，我们选择不同的参数组合。首先，我们采用不同数量的顶点（2^{15} 至 2^{27}）和边（1 M 至 200 M）。所有组合均列于表 Ia。这样，我们便能系统地生成具有不同平均度和密度的图（这类图见于 SNAP [38] 和 KONECT [49]等图存储库）。其次，对于每种顶点数与边数的组合，我们为 a、 b、 c 和 d 采用九种数值组合。通过这种方式，我们可以影响度分布的偏斜程度、平均局部ayeит

聚类系数和三角形的平均数量。对于所有图，我们将 d 固定为5%。我们将 c设为 19% 或 34%，这会导致社区 a 与 d之间的边数不同。现在，我们对 a 和 b采用不同的组合。a 的值越大，度分布的偏斜程度越高，社区 a也越大。b 的值越大，a 与 d 之间的边（簇间边）越多。所有九种 R-MAT 参数组合如表 II 所示。

我们将 297 个生成图的图属性与真实世界图进行比较。图 6a 至图 6e 中的小提琴图展示了图属性值的分布，以及真实世界图和生成图的最小值、最大值和中位数。真实世界图的图属性在很大程度上都被生成图所涵盖，这表明生成图对真实世界图属性具有良好的覆盖度。

图 6f 展示了 a、b、c 和 d 的不同取值组合如何影响生成图的聚类系数，以及这些图能否“轻松”地进行分区。图中以如下方式表示多个图。所有图均有 $|E| = 160$M 条边。图中的每条线表示该图包含多少个 $n \in \{2^{22}, 2^{23}, 2^{24}, 2^{25}, 2^{26}, 2^{27}\}$ 顶点。线上的标记对应 a、b、c和 d 的九种不同取值组合，因此每个标记分别代表一个图。纵轴表示图的聚类系数。横轴表示使用流式分区器 *High-Degree Replicated First (HDRF)* [7] 将图分为 64 个分区后得到的复制因子。我们观察到，较高的聚类系数与较低的复制因子相关。对于其他图和分区算法，也观察到了类似的结果。

这 297 个 R-MAT 图用于训练 *PartitioningQual-ityPredictor*。然而，这些图太小，无法测量分布式图处理性能的代表性指标。因此，我们还使用上述 R-MAT 生成器的相同参数组合，额外生成了 180 个更大的图（见表 II）。这些图包含 1.8 M 至 50 M 个顶点，以及 100M 至 500 M 条边（见表 Ib）。这些较大的图用于训练 *ProcessingTimePredictor* 和*PartitioningTimePredictor*。

我们还尝试使用 Barabási–Albert 模型 [50]作为图生成器（Hagberg 等人 [51]的实现）。该生成器需要输入顶点总数 $|V|$，以及为每个新顶点添加的边数 m。我们创建了 70 个图，顶点数为 $|V| = 1$ M，边数为 $m \in \{1, 2, \ldots, 70\}$，以生成平均度介于 2 和 140 之间的图。平均度会影响复制因子）。然而，如果固定 m 并更改 $|V|$，则所有分区算法的复制因子都不会改变。因此，无法创建平均度相同但复制因子不同的图。与 R-MAT 不同，该方法也无法涵盖真实世界图的所有图属性。

总之，使用 R-MAT 进行图生成取得了令人鼓舞的结果，而 Barabási–Albert 模型则不然。

表 I：用于（a）分区质量预测器 以及（b）处理时间预测器和 *PartitioningTimePredictor*训练的生成式 R-MAT 图中边数和顶点数的不同组合。

（a）R-MAT-SMALL

$\|E\|$ (M)	$\|V\|$
1	2^{15}, 2^{16}, 2^{17}, 2^{18}, 2^{19}
40	2^{21}, 2^{22}, 2^{23}, 2^{24}, 2^{25}
80	2^{21} 2^{22} 2^{23} 2^{24} 2^{25} 2^{26}
120	2^{22}, 2^{23}, 2^{24} 2^{25} 2^{26}
160	2^{22}, 2^{23}, 2^{24}, 2^{25}, 2^{26}, 2^{27}
200	2^{22}, 2^{23}, 2^{24}, 2^{25}, 2^{26}, 2^{27}

(b) R-MAT-大型

$\|E\|$ (M)	$\|V\|$ (M)
100	1.8, 2.5, 4, 10
200	3.6, 5, 8, 20
300	5.4, 7.5, 12, 30
400	7.3, 10, 16, 40
500	9.1, 12.5, 20, 50

表 II：用于生成训练图的 R-MAT 参数 a、b、c 和 d 的九种组合 $\mathbf{C_i}$

	$\mathbf{C_1}$	$\mathbf{C_2}$	$\mathbf{C_3}$	$\mathbf{C_4}$	$\mathbf{C_5}$	$\mathbf{C_6}$	$\mathbf{C_7}$	$\mathbf{C_8}$	$\mathbf{C_9}$
a	0.35	0.45	0.55	0.60	0.40	0.50	0.60	0.65	0.70
b	0.26	0.16	0.06	0.01	0.36	0.26	0.16	0.11	0.06
c	0.34	0.34	0.34	0.34	0.19	0.19	0.19	0.19	0.19
d	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05

足够灵活，可以生成种类繁多的图。

B. 机器学习特征

对于*EASE*的全部三个预测组件，我们使用两类特征：(1) 常见的图属性，用于将图映射到我们已处理过的最相似图；(2) 其他特征，这些特征针对特定的预测组件。

1) 图属性：总共，我们使用三组不同的图属性作为特征（见表 III）。下文将说明每项预测任务使用了哪些特征。

分区质量预测器 将基本和高级图属性用作特征。采用两种不同特征集的原因是，与计算开销较大的高级图属性相比，基本图属性即使未知，也可以轻松计算。然而，高级特征能更深入地揭示图的特性，因此可能带来更好的预测。许多图数据集存储库中 [38], [42], [49],都预先计算了高级属性。基本特征包括图的平均度 、入度分布的偏度、出度分布的偏度 以及图的密度 。这些都是图的常见属性，预计它们能够帮助了解图的可分区性。在实验研究中 [12],[28],，度分布被用于分区器的选择。此外，研究发现，不同的分区器可能对度分布的偏度敏感 [7]。除基本特征外，高级特征还包括三角形的平均数量和平均局部聚类系数。研究发现，许多真实世界的网络具有社区结构，因此聚类系数较高 [34]。这可以作为衡量图的可分区性的一个有趣指标。在 [52],中，作者发现他们的分区器在聚类系数较高的图上表现更好（以割边数衡量）。因此，我们使用高级特征来研究如何。

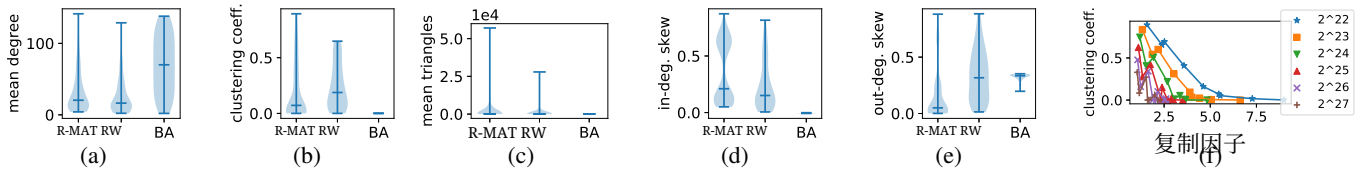


图6：(a)-(e)：生成的 R-MAT 和Barabasi-Albert(BA)图以及真实世界（RW）图的图属性，(f) 高聚类系数与低复制因子之间的相关性。

它们能在多大程度上改进复制因子预测。然而，我们尚未找到任何文献表明高级特征会影响负载均衡。因此，我们仅将高级特征用于复制因子预测，而不用于负载均衡预测。

处理时间预测器 仅考虑简单特征。许多图处理算法会对每条边或每个顶点执行计算。因此，边和顶点的数量是最重要的图属性。

对于*PartitioningTimePredictor*，会考虑所有特征集。对于无状态分区器，简单特征最为重要，因为它们代表了图的大小。然而，对于更高级的分区器，基本属性和高级属性也可能是重要特征。例如，在 HEP [11], 中，图有多少部分以内存式处理、多少部分以流式方式处理，是根据平均度决定的。因此，平均度会影响分区运行时间 [11]。再比如，2PS 会将聚类作为预处理步骤，并对簇排序 [21]。因此，聚类系数可能是一个重要特征。

2）其他特征：除了图属性外，我们还考虑了预测组件专属的特征。

PartitioningQualityPredictor 始终将图划分器和分区数量作为专属特征。某些图划分器使用会影响划分质量的配置参数。我们将划分器特定参数的不同设置视为彼此独立的划分器。在第 V 节中，我们以 HEP 划分器 [11] 为例，使用不同的参数 $\tau \in \{1, 10, 100\}$，这些参数会影响图中有多少条边以内存式方式划分，有多少条边以流式方式划分。这三种设置产生了三个不同的划分器：HEP-1、HEP-10 和 HEP-100。

ProcessingTimePredictor 将划分质量指标（复制因子、边平衡、顶点平衡、源平衡和目标平衡）用作专属特征。针对每种图处理算法，都会单独训练一个模型。因此，可以轻松添加新的图处理算法，并分别对其进行训练。

PartitioningTimePredictor 将分区器视为特定特征。对于质量预测，我们将分区器特定参数的不同设置视为不同的分区器。

C. 训练

对于三个预测组件中的每一个，我们比较了六种监督式机器学习算法。多项式回归在实践中能够有效预测计算成本 [53]。支持向量回归（*SVR*）能够取得非常出色的性能，并被认为是回归任务中的重要技术 [54], [55]。随机森林回归器（*RFR*）对离群值和噪声具有相对较强的鲁棒性，易于并行化，并且可以解释所使用特征的重要性 [56] ——我们在第 V-E 节中利用了 RFR 的可解释性。极端梯度提升（*XGB*）能够取得最先进的结果，并且常用于知名的机器学习竞赛 [57]。诸如 RFR 或 XGB 之类的决策树集成能够在表格数据上与深度神经网络竞争，甚至超越它们 [58]–[61]。*K* 近邻回归器（*KNN*）作为一个简单的基线。除这些传统的机器学习方法之外，我们还使用全连接多层感知机（*MLP*）作为前馈深度神经网络的代表。

数据划分： 所有模型均使用合成生成的 R-MAT 图进行训练。分区质量预测器使用较小的 297 个 R-MAT 图进行训练，而分区时间预测器 和处理时间预测器 使用较大的 180 个 R-MAT 图进行训练。针对每种机器学习算法，在训练集上使用五折交叉验证 [62]进行超参数调优。根据交叉验证得分对模型进行比较，并使用最佳超参数在完整训练集上重新训练。最后，在仅由真实世界图组成的测试集上评估这些模型。

预处理：我们按如下方式对生成的数据进行预处理：使用 z 分数归一化对数据进行标准化 [63]。对于 分区质量预测器 和 分区时间预测器，使用独热编码将分区算法转换为数值特征。

超参数搜索：对于全部六种机器学习算法，都可以对超参数进行调优。我们针对每种机器学习算法执行了网格搜索，以确定最佳超参数。请在我们的GitHub代码仓库中查找所使用的组合 [64]。

测试与扩充：最后一步是使用真实世界图测试我们的模型。为找出分区质量预测器的不足，我们会评估不同分区算法与图类型组合的预测性能。如果某种组合的表现不佳，但与用户相关，则可以使用真实世界 图扩充训练集。例如，如果用户主要处理

表 III：概述哪些图属性用于哪个特征集，以及哪些特征用于哪个预测组件。I是给定图处理算法（例如 PageRank）的迭代次数，k 是分区数量。

	图属性（第 II-B 节）								其他特征（第 IV-B2 节）				
描述	|E|	|V|	Mean 度数	密度	入度。分布式。	出度 Dist.	平均值 #三角形	LCC	质量指标（第 II-A 节）	k	分区器	I	Opt. Goal
特征集													
简单	✓	✓											
基础	✓	✓	✓	✓	✓	✓							
高级	✓	✓	✓	✓	✓	✓	✓	✓					
预测任务与分区器选择器													
划分质量			✓	✓	✓	✓	✓	✓		✓	✓		
划分时间	✓	✓	✓	✓	✓	✓					✓		
处理时间	✓	✓							✓			✓	
选择器	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

维基图谱，但针对这种图类型的预测不够准确，用户可以使用真实世界的维基图谱扩充训练数据，从而提升预测性能。

D. 推理

为了对未见过的图进行预测，对于*Par-titioningQualityPredictor*，需要提供图属性、分区器和分区数量。对于*Process-ingTimePredictor*，需要提供要执行的图处理算法、图属性和分区质量指标。与运行至收敛的图处理算法不同，对于运行固定迭代次数的图处理算法，还需要提供迭代次数。对于*PartitioningTimePredictor*，推理时需要提供图属性和分区器。为了使用*PartitionerSelector*自动选择分区器，需要提供优化目标，即最小化处理时间或端到端时间。表 III 展示了各预测任务所需的特征。

E. 设计选择讨论

用于预测任务的独立机器学习模型：我们讨论两种不同于本方法的替代方案。第一种，不分别预测分区和处理运行时间，而是训练一个直接预测端到端时间的单一模型（替代方案 1）。第二种，在处理运行时间预测中，可以将分区器用作特征，而不是 使用分区质量指标。因此，替代方案 2 无需预测分区质量指标。

我们出于两个原因没有采用替代方案 1。首先，它不如我们的方法灵活，因为它只能预测端到端运行时间。有些用户可能只关注如何缩短图处理运行时间，例如，分区可以在廉价的计算节点上离线完成，而处理则在昂贵的计算集群上运行，因此只需尽量降低处理成本。其次，如果添加了新的分区器或更换了处理框架，就需要重新训练整个模型。采用我们的方法，在这种情况下，只需重新训练用于预测分区运行时间或 处理运行时间的模型。

我们基于以下原因决定不采用替代方案 2。首先，采用我们的方法，可以轻松添加新的分区算法。以较低的开销纳入：如果应包含一个新的分区器，只需在训练图上运行该分区器，以测量质量指标和分区运行时间。然后，分区质量预测器 和 分区时间预测器 便可以重新训练。处理时间预测器 不需要重新训练，因为已经存在一个基于质量指标预测图处理运行时间的模型。相比之下，采用替代方案2时，所有图处理算法都需要在训练图上重新运行，因为使用的是分区器而不是分区质量作为特征。其次，采用我们的方法，可以进一步了解分区质量指标如何影响处理运行时间。这对于研究人员来说很有用，能够帮助他们更好地理解：对于哪种图处理算法，哪些质量指标最为重要。这一洞见可能有助于设计新的分区算法或优化图处理框架。第三，复制因子预测对于内存受限的处理同样有用，因为每个副本都会生成一份顶点状态的副本。例如，在图神经网络（图神经网络）中，顶点状态可能由数千个特征组成，而由于处理是在图形处理器上执行的，内存受到严格限制。在这种情况下，预测的复制因子可能是选择分区器时的重要决策标准。

将图嵌入用作特征：还可以通过使用图神经网络（GNN）计算图嵌入，以低维向量表示图，从而为预测任务使用复杂的图特征 [65]–[68]。在我们的实验中[1]，，我们测量了表 IV 中所有图在中央处理器以及两种不同图形处理器上，使用均值池化的两层 GraphSage 图神经网络 [66] 的推理时间。我们发现，中央处理器上的推理时间是最慢分区器分区时间的 9 到 19 倍。因此，可以运行所有分区器并测量实际指标，而不是预测这些指标，所以我们无法在端到端流程中摊销高昂的推理时间。在这一规模下，图形处理器上的推理受到图形处理器内存的严重限制。即使使用一块拥有 48 GB 图形处理器内存的大型图形处理器，我们也只能为表 IV 中的前四个图计算嵌入，其余图则会耗尽图形处理器内存。它不会

[1]我们在 GitHub 代码仓库中报告了在中央处理器和两种不同图形处理器上的所有推理时间，并将其与 EASE 进行了比较。下面总结这些结果。

表 IV：用于分区-时间预测器 和处理时间预测器。

图	边（M）	顶点 (M)
com-orkut.ungraph	117.2	3.1
enwiki-2021	150.1	6.3
eu-2015-tpd	165.7	6.7
hollywood-2011	229.0	2.0
out.orkut-groupmemberships	327.0	8.7
eu-2015-host	379.7	11.3
gsh-2015-tpd	581.2	30.8

仅仅为了优化分布式图处理的分区步骤，就采用这种重量级处理并不划算。因此，我们的系统不使用图神经网络。

V. 评估

在下文中，我们评估*EASE* 预测分区质量指标、分区运行时间和处理运行时间的能力。我们还将*EASE* 与选择图划分器的人工启发式方法进行比较，例如随机选择，或选择能够产生最低复制因子的分区器。

A. 评估指标

我们使用回归任务中常用的评估指标来评估机器学习模型。在以下公式中，y_i 是第i 个样本的真实值（例如复制因子），$\hat{y}_i$ 是第 i个样本的预测值，y 是所有样本的均值，而 ϵ 是一个较小的值，用于防止除以零。

1) 均方根误差定义为$\text{均方根误差}(y,\hat{y}) = \sqrt[2]{\frac{1}{n}\sum_{i=0}^{n-1}(y_i-\hat{y}_i)^2}$。该值越接近 0 越好。2) 平均绝对百分比误差定义为 $\text{平均绝对百分比误差}(y,\hat{y}) = \frac{1}{n}\sum_{i=0}^{n-1}\frac{|y_i-\hat{y}_i|}{max(\epsilon,|y_i|)}$。该值越接近 0 越好。

B. 测试集

为评估 分区质量预测器，我们从 SNAP [38], 、KONECT [49] 和网络数据存储库 [42]中收集了175个真实世界图。这些图可分为以下图类型：(1) 12个隶属关系图，(2) 3个引用图，(3) 6个合作图，(4) 5个交互图，(5) 5个互联网图，(6) 1个产品网络，(7) 31个社交网络，(8) 12个万维网图，(9) 101个维基图谱。除101个维基图谱中的96个外，所有图均用作所有评估的最终测试集。这96个维基图谱随后用于评估训练数据富集。我们从相同的图数据存储库中收集了7个真实世界图，用于评估 *PartitioningTimePredictor* 和 处理时间预测器（见表 IV）。所有图数据集的链接均已发布在我们的 GitHub 代码仓库 [64] 中。

C. 训练、验证与测试

数据生成：对于三个预测组件，我们测量训练图和测试图中第二节B 部分介绍的图属性。我们使用四类共 11 种不同的分区器。(1) 无状态流式处理与

目标顶点的一维哈希（1DD）和源顶点的一维哈希（1DS） [5], 2；二维哈希（2D） [5],；规范随机顶点切分（CRVC） [5] ；以及基于度数的哈希（DBH） [16]。(2) 有状态流式处理：高入度优先复制（HDRF） [7] 和两阶段流式处理（2PS） [21]。(3) 使用邻域扩展（NE） [25]进行内存分区。(4) 使用混合边划分器（HEP） [11] 进行混合式分区。HEP 使用一个参数 τ 来控制有多少边采用内存方式划分，以及有多少边采用流式方式划分 [11]。按照作者的建议，我们将 HEP 配置为 $\tau \in \{1, 10, 100\}$，简称 HEP-1、HEP-10 和 HEP-100，并将它们视为不同的分区器。因此，我们总共有 11 个分区器。所选的分区算法在各自类别中都是强有力的基线，并声称在划分质量和分区运行时间之间取得了良好的平衡。因此，它们都可能成为最小化端到端运行时间的候选方案，而手动确定最优分区策略并非易事。

对于 分区质量预测器，我们将图划分为 $k \in K = \{4, 8, 16, 32, 64, 128\}$ 个分区，并测量第 II-A 节中讨论的五项分区质量指标。

对于 处理时间预测器，我们将所有图通过所有分区器划分为 4 个分区，并测量分区质量指标。然后，我们在包含 4 台机器的 Spark/GraphX 集群上执行 6 种图处理算法：PageRank (PR)，运行 10 次迭代；连通分量 (CC)；使用随机选取的种子顶点的单源最短路径 (SSSP)；以及 $k = deg(G)$ 的 K 核。这些算法分别体现了不同类型的工作负载。在 PageRank 中，每次迭代的所有顶点都处于活动状态。在连通分量算法中，第一次迭代时所有顶点都处于活动状态，活动顶点的数量会随时间减少。在单源最短路径算法中，第一次迭代时只有一个顶点处于活动状态。后续迭代中，活动顶点的数量先增加，随后减少，直至不再有顶点处于活动状态。在 K 核算法中，第一次迭代时有许多顶点处于活动状态，并会随时间变为非活动状态。除这 4 种真实世界算法外，我们还实现了一种合成算法：图中的每个顶点都包含一个由 s 64 位双精度数构成的特征向量，并在每次迭代中沿出边发送其特征向量。通过 s，可以影响通信量。我们将 s 设为 1 和 10，并分别将这两种配置简称为合成低和合成高，表示低通信负载和高通信负载。我们将迭代次数设为 5。对于合成工作负载和 PageRank，计算负载和通信负载在整个迭代过程中保持不变，迭代次数是算法的一个参数。因此，我们测量平均迭代时间，它也是预测目标。总处理运行时间为平均迭代时间乘以迭代次数。对于其余算法，计算负载和通信负载会随时间变化，且算法会一直运行到收敛，因此我们测量直到收敛为止的总处理时间。

对于 *PartitioningTimePredictor*，我们对所有图进行分区

表 V：处理时间预测器 在测试集上的质量指标（真实世界图的平均值）。

算法	模型	MAPE
连通分量	多项式回归	0.272
K 核	XGB	0.401
PageRank	XGB	0.295
单源最短路径	XGB	0.300
合成高	多项式回归	0.259
合成低	多项式回归	0.271

使用所有分区器将其划分为 4 个分区，并测量分区运行时间。

训练与验证：对于全部三个预测组件，我们使用第 IV-C 节中提到的全部六种机器学习算法训练回归模型，并根据合成图上的五折交叉验证得分选出最佳模型。接下来，我们在由真实世界图组成的测试集上评估所选模型。

测试： 我们在表 IV 中列出的七个真实世界图上评估了分区时间预测器 和 处理时间预测器。对于分区时间预测器 ，我们使用 XGB 在测试集上取得了 0.335 的平均绝对百分比误差。对于处理时间预测器，使用 XGB 后，PageRank、K 核和单源最短路径的平均绝对百分比误差分别为0.295、0.401和0.300（见表 V）。对于连通分量、合成低和合成高，多项式回归得到的平均绝对百分比误差分别为0.272、0.271 和0.259。我们使用第 V-B 节中提到的 80 个真实世界图评估了分区质量预测器。结果如表 VI 所示。对于顶点平衡、源平衡、边平衡和目标平衡，RFR 得到的平均绝对百分比误差分别为 0.152、0.144、0.079 和0.154。在复制因子方面，XGB 使用基本特征时得到的平均绝对百分比误差为0.296，而使用高级特征后可略微提升至 0.288。这意味着，平衡指标能够比复制因子预测得更准确。我们进一步研究了对于图类型和分区算法的哪些组合，模型能够取得良好效果，以及对于哪些组合，进一步训练会有所帮助。复制因子的结果如图 7a 所示。我们观察到，所有分区算法在图类型引用、交互、互联网、产品网络和 社交上都取得了良好的分数。然而，分区器 2PS、HDRF、HEP 和NE 在图类型合作、万维网 和 维基上的预测效果并不理想。对于高级特征也得到了类似的结果。顶点平衡的结果如图 7c所示。由于篇幅限制，此处省略了其他平衡指标的结果，但这些结果可在我们的 GitHub 代码仓库中找到 [64]。

与复制因子预测相比，顶点平衡的预测结果对图类型的依赖较小，但对分区算法的依赖更大。值得注意的是，对于 NE 和 HEP-100，准确率相当低。在对该问题进行更详细的调查时，我们发现这可能与分区器自身的不稳定行为有关：我们观察到，当在同一图上使用相同的分区数量多次执行 NE 时，顶点平衡可能会大幅波动，最高可达

表 VI：测试集上分区质量预测器的质量指标（在真实世界图上取平均值）。

目标	模型	特征	MAPE	RMSE
$RF(P)$	XGB	基础	0.296	1.197
$RF(P)$	XGB	adv.	0.288	1.238
$B_v(P)$	RFR	基础	0.152	0.921
$B_{src}(P)$	RFR	基本的	0.144	0.851
$B_{edge}(P)$	RFR	基础	0.079	1.055
$B_{dst}(P)$	RFR	基础	0.154	1.020

2.1在不同运行之间。这是由于 NE 中随机种子顶点选择所导致的，从而使不同运行中的分区有所不同。HEP 在较小程度上也受到同一问题的影响。与顶点平衡不同，NE 和 HEP 各自不同运行之间的复制因子保持稳定。

D. 富集

当使用合成数据训练的模型在特定图类型与分区器组合上表现不佳时，我们建议使用真实世界图来扩充分区质量预测器 的训练集。

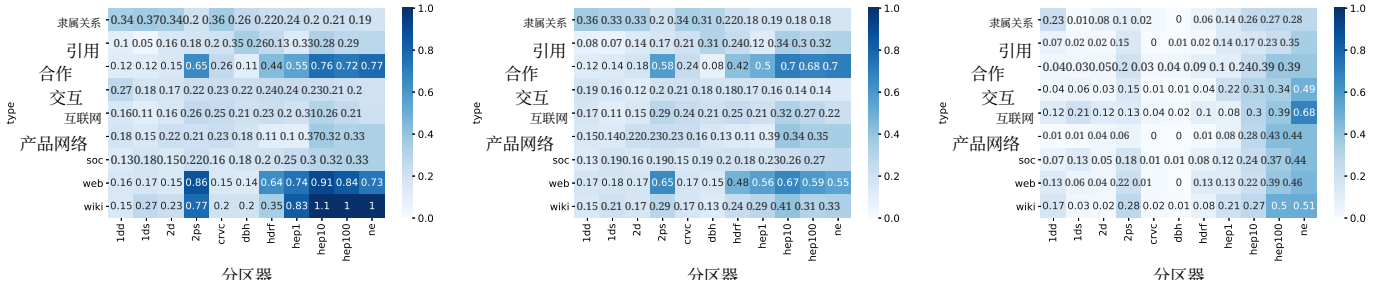
与其他组合相比，分区器 2PS、HDRF、HEP 和 NE 在合作、万维网 和维基 图类型上的复制因子预测表现不佳（见图 7a）。因此，我们在合成训练集中加入最多 96 个额外的真实世界维基图谱（G_{wiki}）。为了研究富集所需的图数量，我们选取 19、38、57、76 和 96 个维基图谱（$G_{selected}$）的子集进行富集。我们使用与第 V-B 节相同的测试集。由于我们从 G_{wiki}中随机选取 $G_{selected}$，因此每项实验重复三次，并报告平均质量指标及其标准差。

对于平衡指标和复制因子，我们采用表 VI 中的 RFR 模型。使用基本特征时，XGB 的平均绝对百分比误差略低，为 0.296，而 RFR 为 0.303；但对于每个富集级别，XGB 的训练时间远长于 RFR（140 分钟对 1 分钟）。

96 个真实世界图的富集结果按分区算法和图类型的各个组合展示在图 7b 中。图 8 展示了富集数据集的大小对质量指标的影响。复制因子预测的主要观察结果如下:

(1)使用全部96个真实世界图进行富集时，维基图谱的平均绝对百分比误差分别从0.555 降至 0.244（基本特征）以及从0.566降至0.178（高级特征）。因此，通过富集，可以减弱针对维基图谱的合成训练模型的弱点。

(2) 高级特征结合富集后带来了更低的平均绝对百分比误差（0.178对比 0.242）。因此，在使用富集时，值得考虑高级特征。对此现象的一种可能解释是，通过富集，训练中使用了高级特征具有更现实取值的维基图谱，因此，未见过的图可以更好地映射到这些图谱。



(a) 复制因子。 (b) 复制因子（含富集）。 (c) 顶点平衡。

图 7：复制因子和顶点平衡预测的平均绝对百分比误差评分。

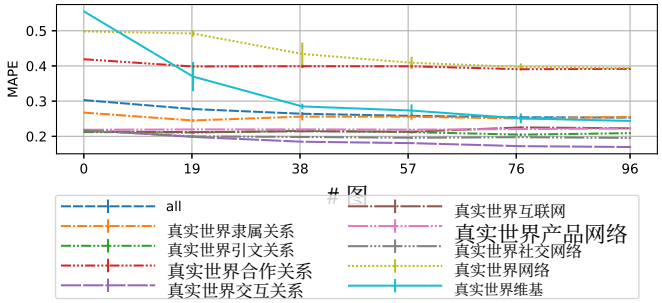


图 8：不同富集程度下，各种图类型的平均绝对百分比误差均值及标准差。

(3) 如图 8 所示，富集程度越高，预测效果越好。然而，即使仅富集 19 个图，也能大幅提升预测质量（平均绝对百分比误差从 0.555 降至 0.370）。换句话说，只需在训练数据集中加入少量真实世界图，就能改善模型对“薄弱点”的预测。

(4) 除维基图谱外，富集也降低了其他图类型（尤其是万维网图）的预测误差，或至少使其保持稳定（见图 8）。预计维基图谱与万维网图相似，因此基于维基图谱获得的见解可以迁移到万维网图。

通过富集，也可以提升维基图谱的平衡指标预测性能。维基图谱的平均绝对百分比误差平均略有下降，从 0.156 降至 0.114。然而，所有图类型的整体平均绝对百分比误差略有上升，从 0.132 增至 0.154。总体而言，即使不进行富集，负载均衡预测也已经相当准确。因此，在这种情况下，富集的效果较弱。

E. 特征重要性

为了了解不同特征对于预测分区质量指标的重要性，我们计算了 RFR 模型中每个特征的特征重要性。随机森林是由多个决策树组成的集成模型。决策树根据特征递归地将所有样本划分为更小的子集。使用纯度指标来决定使用哪个特征划分节点。

表 VII：质量指标的特征重要性。

特征	RF	B_v	B_{src}	B_{dst}	B_{edge}
分区器	0.299	0.268	0.245	0.542	0.244
平均度	0.274	0.065	0.059	0.058	0.036
分区数量	0.256	0.271	0.293	0.177	0.472
度分布	0.165	0.368	0.372	0.214	0.214
密度	0.007	0.028	0.031	0.009	0.034

在我们使用的机器学习框架中，均方误差被用作纯度指标 [69]。特征的重要性通过基于该特征拆分节点所带来的纯度提升来表示，并根据该节点所包含的样本数除以样本总数进行加权。

表 VII 列出了所有分区质量指标对应的基本特征重要性。我们有以下观察：首先，对于所有分区质量指标，分区算法和分区数量都非常重要，其重要性分别介于 0.244 至 0.542 和 0.177 至 0.477 之间。其次，度分布是一项重要特征，对于复制因子的重要性为 0.165；对于平衡指标，则是一项非常重要的特征（0.214 至 0.372）。第三，平均度对于复制因子是一项重要特征（0.274），而对于平衡指标则是一项不太重要的特征（0.036 至 0.065）。在所有质量指标中，密度都是最不重要的特征（0.007 至 0.034）。

这些结果似乎合乎情理。首先，分区算法和分区数量对于机器学习模型都很重要，这是意料之中的。其次，平均度对复制因子很重要，这也合乎情理。我们观察到，对于所有分区算法，平均度越高，复制因子也越高。第三，度分布的重要性也在预期之中。此前已经发现，度分布会影响分区算法 [7], [21]。有趣的是，密度并不重要。或许平均度已经捕捉到了类似的特征，因此无需额外加入密度特征。因此，可以从特征集中移除密度。

自动分区器选择

接下来，我们将评估*PartitionerSelector*能否基于预测自动选择分区器。

表 VIII：不同分区器选择策略的比较： S_{PS}、 S_O、 S_{SRF}、 S_R和 S_W分别表示 PartitionerSelector（我们的方法）、最优分区器、复制因子最小的分区器、随机选择的分区器以及最差分区器所达到的时间。我们区分两种选择目标：最小化端到端时间（端到端）或处理时间（处理）。

(a) 无富集

Goal	算法	S_{PS} 占基线的百分比 S_O	S_{SRF}	S_R	S_W	S_{SRF} 占比（%）S_O的
E2E	SSSP	102	69	84	67	152
E2E	CC	103	58	76	57	184
E2E	PR	110	96	96	79	119
E2E	K 核	111	76	91	73	152
E2E	合成高	113	98	92	73	119
E2E	合成低	117	99	95	76	121
Pro.	SSSP	106	93	94	80	117
Pro.	CC	107	92	91	74	121
Pro.	PR	111	99	96	79	116
Pro.	K 核	113	94	97	80	123
Pro.	合成高	114	99	92	72	117
Pro.	合成低	119	101	96	76	120

(b) 富集

Goal	丰富。	S_{PS} 占基线的百分比 (Enwiki-2021) S_O	S_R	S_W	S_{PS} 占基线的百分比（所有图）S_O	S_R	S_W
E2E	No	113	94	78	110	89	71
E2E	Yes	107	89	74	112	91	72
Pro.	No	111	91	73	114	96	78
Pro.	Yes	106	88	71	117	98	80

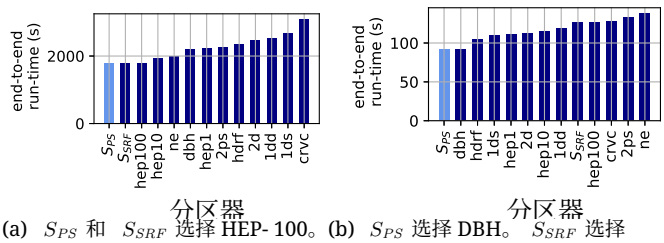
的 分区质量预测器、分区时间预测器和处理时间预测器，以最小化端到端时间和图处理时间。我们将分区器选择器 (S_{PS}) 与四种基线策略进行比较：(1)选择最差的分区器(S_W)，(2)选择随机分区器 (S_R)，选择最优分区器 (S_O)，以及 (4) 选择复制因子最小的分区器 (S_{SRF})。

我们的主要观察结果如下。在 26.2% 和 35.7% 的情况下，S_{PS} 会从 11 个分区器中选出最优分区器，分别使处理时间和端到端时间最短。相比之下，S_{SRF} 和 S_R 等手动选择策略表现较差。 S_{SRF} 在 7.1% 和 14.3% 的情况下，分别选出了处理时间和端到端时间方面的最佳分区器。对于两个优化目标， S_R 仅在 9.1%（$= \frac{1}{11} \cdot 100\%$）的情况下选出最佳分区器。

此外，我们将采用 S_{PS} 进行选择所产生的处理时间和端到端时间，与四种基线进行比较分析。表 VIIIa（见第 3 至 6 列）展示了对于每种图处理， S_{PS}所选分区器对应的处理时间和端到端时间分别占相应基线的百分比算法（越低越好）。

最差选择(S_W)： S_{PS} 的处理时间平均为 S_W 的 72%（合成高）至 80%（单源最短路径），端到端时间平均为 S_W 的 57%（连通分量）至 79%（页面排名）（见表 VIIIa 中的列 S_W）。因此， S_{PS} 在所有情况下都能带来显著更好的性能。

随机选择(S_R)： S_{PS} 的端到端时间平均为 S_R 的 76%（连通分量）至 96%（页面排名），而



(a) S_{PS} 和 S_{SRF} 选择 HEP- 100。(b) S_{PS} 选择 DBH。 S_{SRF} 选择 HEP-100。

图 9：图*Enwiki-2020*上不同分区器以及选择策略 S_{PS} 和 S_{SRF} 的端到端时间比较，其中 (a) 为合成高，(b) 为连通分量。

处理运行时间平均为 S_R 的 91%（连通分量）至 97%（K 核）（参见表 VIIIa 中的 S_R 列）。因此，S_{PS} 也优于随机选择。

最优选择 (S_O)： 与最优选择相比S_O， S_{PS} 带来的端到端时间平均为 S_O 的 102%（单源最短路径）至 117%（合成低），处理时间平均为 S_O 的 106%（单源最短路径）至 119%（合成低）（见表 VIIIa 中的 S_O列）。因此，对于单源最短路径、连通分量和页面排名算法，在这两个优化目标下，该选择都相对接近最优值；而对于 K 核和合成算法，仍有更大的改进空间。

最小复制因子(S_{SRF})： S_{PS} 导致端到端时间平均为 S_{SRF}的58%（CC）至99%（合成低）。处理时间为 S_{SRF} 的92%（CC）至101%（合成低）（见表 VIIIa中的第S_{SRF} 列）。仅在一种情况下（对于算法合成低），S_{PS} 导致图处理时间为 S_{SRF}的101%。

我们进一步将 S_{SRF} 与最优策略 S_O进行比较。在表 VIIIa 的最后一列中，我们报告了对于每种图处理算法，S_{SRF}的选择所需的处理时间和端到端时间占 S_O 的百分比。我们观察到，即使已知所有分区器的复制因子，也不足以完成分区器选择。首先，除了复制因子之外，平衡指标也很重要。其次，分区时间也很重要；有些情况下，为了获得最小复制因子而投入大量时间进行分区并不值得，因为节省的时间无法通过缩短图处理时间来弥补。如图 9 所示：对于通信受限的算法 Synthetic-High，HEP-100 达到了最小复制因子，并使端到端时间最短（参见表 9a）。分区时间所带来的开销可以得到弥补。然而，对于 CC，使用 DBH 快速分区能够实现最低的端到端时间（参见表 9b）。 S_{PS}同时考虑分区时间和处理时间，因此在多种工作负载下都能做出良好选择。我们还想指出， S_{SRF} 是一种相当假设性的策略，因为通常在图完成分区之前，并不知道复制因子。

我们还研究了通过利用维基图谱进行富集，是否能够提升 S_{PS} 在 *Enwiki-2021* 上的性能。我们使用相同的随机森林模型进行预测。

如第 V-D 节实验所示，比较富集与未富集时的划分质量。与未富集相比， S_{PS} 富集后，*enwiki-2021*的处理时间和端到端时间平均分别缩短了 4% 和 5%。在所有 图上进行评估时，选择富集后的 S_{PS} 使处理时间和端到端时间平均分别增加了 2% 和 3%。在表 VIIIb 中，我们以相应基线 的百分比报告了 S_{PS}选择方案（富集和未富集）的平均处理时间和端到端时间：在*enwiki-2021* 上评估的结果（见第 3–5 列），以及在所有 图上评估的结果（见第 6–8 列）。目标类别中的改进幅度大于其他类别中的性能下降幅度，因此，如果大多数工作负载都涉及增强图类型，富集就能带来收益。

总结： 我们的实验表明，*EASE* 能够对不同类型的图划分算法的分区运行时间和划分质量提供定量预测。在此基础上，*EASE* 能够预测具有不同工作负载特征的图处理算法的运行时间。我们仅使用合成图进行训练，就实现了较高的预测准确率；通过使用真实世界图丰富训练数据，还可以进一步提高准确率。基于运行时间预测，我们提供了自动图划分器选择，以最小化图处理运行时间或端到端运行时间。在这两种情况下，自动选择都优于手动选择启发式方法。

VI. 相关工作

过去几年中，已有实验研究 [12], [28]–[30], [70]开展，深入分析不同图分区算法的特性，以帮助选择算法。不过，这些研究不足以针对给定场景实现自动分区器选择，也没有提供自动纳入新分区算法和图处理算法的方法。在我们的工作中，我们对预期的分区质量、分区时间和图处理时间进行定量预测，并且能够轻松扩展以纳入新的分区算法和图处理算法。

图生成器领域已经开展了大量研究（例如，[34], [43], [45], [46], [50], [71]–[73]），以生成具有现实世界属性的合成图。图生成器是我们方法的重要组成部分，我们提出选择一个合适的图生成器，该生成器能够生成预期作为工作负载的代表性图，并且应对这些图的划分质量、分区和图处理运行时间进行预测。在我们的实验中，我们使用了 R-MAT这一领域无关通用图生成器[46] ，它能够创建具有现实世界属性的图 [45]。然而，未来还可以探索其他图生成器，例如 [73]–[75]，以应用于我们的方法。

已经开发出许多图分区算法，可分为（1）流式算法、（2）内存算法和（3）混合分区算法。流式分区算法 [4], [5], [13], [14], [16]–[23], [76]以流的形式读取图（例如边列表），并分配边（顶点切分）或顶点（边切分）即时分配到分区。流式算法可以是无状态的，也可以是有状态的。内存式分区器 [8], [9], [24]–[27], [77]将完整的图加载到内存中以执行分区。因此，与流式分区相比，可用于分区的是图的完整视图。这在许多情况下会带来更低的复制因子，但分区运行时间和内存开销可能更高。混合边分区 [11] 是内存式分区与流式分区的组合。图的一部分在内存中进行分区，其余部分以流式方式进行分区。在评估中，我们展示了系统可以预测每个类别中代表性分区算法的划分质量和运行时间。我们还以HEP为例展示了我们的方法可以处理使用影响划分质量和运行时间的参数的分区器。这同样可以应用于具有类似参数的其他分区器 [15], [18] 。

Fan 等人 [78] 提出了一种将顶点切分和边切分分区器增量化的方法。我们的方法可用于选择一个分区器，然后对其进行增量化。Fan等人 [79] 进一步研究了并行图处理的动态扩缩容。为了扩容或缩容，图通常需要重新进行分区。我们的方法可用于预测分区质量指标、分区和处理运行时间将如何变化。

Fan 等人 [53] 提出了一种应用驱动的分区策略。他们展示了如何针对图处理任务，将给定的顶点划分或边划分细化并转换为混合式划分。我们的方法可用于支持初始划分的选择，随后可将其细化并转换为混合式划分。

已有多项研究使用机器学习来优化数据管理系统，例如学习索引结构[80]、基数估计 [81] 或自动配置调优 [82]。我们的工作是最早尝试使用机器学习优化图处理系统的方法之一。

VII. 结论

为了实现分布式图处理，需要对图进行划分。然而，图划分算法种类繁多，因此很难针对给定场景选择合适的分区器。本文提出了一种基于机器学习的方法，用于预测未见过的图上不同类型的分区算法和图处理算法的划分质量、分区运行时间及图处理运行时间。我们表明，这种定量预测能够自动选择使图处理时间或端到端时间最短的分区器。

未来的工作中，我们计划将该方法扩展到超图分区算法 [83]–[87] 和系统 [88], [89]。

致谢： 本研究部分由德国研究联合会（DFG，德国研究基金会）资助，项目编号：438107855。

参考文献

[1] G. Malewicz, M. H. Austern, A. J. Bik, J. C. Dehnert, I. Horn, N.Leiser, 和 G.Czajkowski，“Pregel：大规模图处理系统”，载于《2010年ACM SIGMOD数据管理国际会议论文集》，系列：SIGMOD’10。纽约，美国：计算机协会，2010年。 [在线]。可访问：https://doi.org/10 1145 1807167 1807184 。 / 。A.Ching, S. Edunov, M. Kabiljo, D. Logothetis, 和 S.Muthukrishnan，“一万亿条边：Facebook规模的图处理”，《VLDB基金会会刊》，第8卷，第12期，2015年8月。 [在线]。可访问：https://doi.org/10 14778 2824032 2824077 。 / 。J. E. Gonzalez, Y. Low, H. Gu, D. Bickson,和C.Guestrin，“Powergraph：自然图上的分布式图并行计算”，载于《第10届USENIX操作系统设计与实现会议论文集》，系列：OSDI’12。美国：USENIX协会，2012年。[4] R. Chen, J. Shi,Y. Chen, 和 H. Chen，“Powerlyra：偏斜图上的差异化图计算与划分”，载于《第十届欧洲计算机系统会议论文集》，系列：EuroSys ’15。纽约，美国：计算机协会，2015 [ ] 10 1145 2741948 2741970 。在线。可访问：https://doi.org/。 / 。J.E.Gonzalez, R. S. Xin, A. Dave, D. Crankshaw, M.J.Franklin, 和 I.Stoica，“Graphx：分布式数据流框架中的图处理”，载于《第11届USENIX操作系统设计与实现会议论文集》，系列：OSDI’14。美国：USENIX协会，2014年。[6] J.Webber，“Neo4j程序化入门”，载于《第三届系统、编程与应用年会：面向人类的软件论文集》，系列：SPLASH ’12。纽约，美国：计算机协会，2012年。 [在线]。可访问：https://doi.org/10 1145 2384716 2384777 。 / 。F. Petroni, L.Querzoni, K. Daudjee,S.Kamali,和 G.Iacoboni，“HDRF：面向幂律图的基于流的划分”，载于《第24届ACM信息与知识管理国际会议论文集》，系列：CIKM ’15。纽约，美国：计算机协会，2015年。 [在线]。可访问：https://doi.org/10 1145 2806416 2806424 。 / 。M. Hanai, T. Suzumura, W. J. Tan, E. Liu,G.Theodoropoulos,和 W. Cai，“万亿边图的分布式边划分”，《VLDB基金会会刊》，第12卷，第13期，2019年9月。 [在线]。可访问：https://doi.org/10 14778 3358701 3358706 。 / 。C. Martella, D. Logothetis, A. Loukas, 和G. Siganos，“Spinner：云中的可扩展图划分”，载于《2017年第33届IEEE数据工程国际会议》（ICDE），2017年4月，第1083–1094页。[10] F. Bourse,M. Lelarge, 和 M. Vojnovic，“平衡图边划分”，载于《第20届ACM SIGKDD知识发现与数据挖掘国际会议论文集》，系列：KDD’14。纽约，美国：计算机协会，2014年，第1456–1465页。 [在线]。可访问：https://doi.org/10.1145/2623330。2623660[11] R.Mayer 和 H.-A. Jacobsen，“混合边划分器：内存限制下的大型幂律图划分”，载于《2021年数据管理国际会议论文集》，系列：SIGMOD/PODS ’21。纽约，美国：计算机协会，2021年。 [在线]。可访问：https://doi.org/10.1145/3448016。3457300[12]A. Pacaci 和M. T. ¨Ozsu，“图划分流式算法的实验分析”，载于《2019年数据管理国际会议论文集》，系列：SIGMOD ’19。纽约，美国：计算机协会，2019年。 [ ] 10 1145 3299869 3300076 在线。可访问：https://doi.org/。 / 。C. Tsourakakis, C. Gkantsidis, B. Radunovic,和 M.Vojnovic，“Fennel：超大规模图的流式图划分”，载于《第7届ACM网络搜索与数据挖掘国际会议论文集》，系列：WSDM ’14。纽约，美国：计算机协会，2014年。 [在线]。可访问：https://doi.org/10 1145 2556195 2556213 。/。I.Stanton 和 G. Kliot，“大型分布式图的流式图划分”，载于《第18届ACM SIGKDD知识发现与数据挖掘国际会议论文集》，系列：KDD’12。纽约，美国：计算机协会，2012年，第1222–1230页。 [在线]。可访问：https://doi.org/10。1145/2339530.2339722

[15] L. Hoang、R. Dathathri、G. Gill和K.Pingali，“Cusp：用于分布式图分析的可定制流式边分区器”，载于 2019年IEEE 国际并行与分布式处理研讨会（IPDPS），2019年5月，第439–450 页。[16] C. Xie、L. Yan、W.-J. Li和 Z. Zhang，“分布式幂律图计算：理论与实证分析”，载于Nips，第 27 卷，2014年，第 1673–1681 页。[17] N. Jain、G. Liao和T. L. Willke，“Graphbuilder：可扩展的图 ETL 框架”，载于第一届图数据管理经验与系统国际研讨会，系列 GRADES’13。美国纽约州纽约市：计算机协会，2013年。 [在线]。可用地址：https://doi.org/10 1145 2484425 2484429 。/。H. P. Sajjad、A. H. Payberah、F. Rahimian、V.Vlassov 和S. Haridi，“增强流式图的顶点割划分”，载于 2016年 IEEE 大数据国际大会（BigData Congress），2016年，第 1–8 页。[19] J.Nishimura和 J.Ugander，“重新流式处理图划分：用于高级负载均衡的简单通用算法”，载于第19 届 ACM SIGKDD知识发现与数据挖掘国际会议论文集，系列KDD’13。美国纽约州纽约市：计算机协会，2013年。 [在线]。可用地址：https://doi.org/10 1145 2487575 2487696 。/。C. Mayer、R. Mayer、M. A. Tariq、H. Geppert、L. Laich、L. Rieger和K. Rothermel，“Adwise：用于高速图处理的基于自适应窗口的流式边划分”，载于2018年 IEEE第 38 届分布式计算系统国际会议（ICDCS），2018年7月，第685–695页。[21] R. Mayer、K. Orujzade 和H. Jacobsen，“2PS：采用两阶段流式处理的高质量边划分”，CoRR，第 abs/2001 07086，卷。。在线。2020 [ ] 2001 07086 可用地址：https://arxiv.org/abs/ 。

[22] R. Mayer、K. Orujzade 和H.-A.Jacobsen，“线性运行时间的核外边划分”，载于 2022年 IEEE 第38 届数据工程国际会议（ICDE），2022年，第 2629–2642 页。[23] M. A.K.Patwary、S.Garg 和 B. Kang，“基于窗口的流式图划分算法”，载于澳大拉西亚计算机科学周多会议论文集，系列ACSW2019年。美国纽约州纽约市：计算机协会，2019年。 [在线]。可用地址：https://doi.org/10 1145 3290688 3290711 。/。G. Karypis和V. Kumar，“非规则图的并行多级 k 路划分方案”，载于 1996年 ACM/IEEE超级计算会议论文集，系列Supercomputing’96。美国：IEEE 计算机学会，1996年。 [在线]。可用地址：https://doi.org/10.1145/ 369028.369103[25]C. Zhang、F. Wei、Q. Liu、Z.G.Tang 和 Z. Li，“通过邻域启发式方法进行图边划分”，载于第 23 届 ACM SIGKDD知识发现与数据挖掘国际会议论文集，系列 KDD ’17。美国纽约州纽约市：计算机协会，2017年。 [在线]。可用地址：https://doi.org/10 1145 3097983 3098033 。/。D. Margo 和M. Seltzer，“可扩展的分布式图划分器”，VLDB基金会会刊，第8卷，第 12 期，2015年8月。 [在线]。可用地址：https://doi.org/10 14778 2824032 2824046 。/。S. Schlag、C. Schulz、D. Seemaier 和D. Strash，《可扩展的边划分》，2019年，第 211–225 页。[28] S. Verma、L.M. Leslie、Y. Shin 和 I.Gupta，“分布式图处理中的分区策略实验比较”，VLDB基金会会刊，第10 卷，第 5 期，2017年1月。 [在线]。可用地址：https://doi.org/10 14778 3055540 3055543 。/。Z.Abbas、V. Kalavri、P. Carbone 和 V.Vlassov，“流式图划分：一项实验研究”，VLDB基金会会刊，第11卷，第 11 期，2018年7月。 [在线]。可用地址：https://doi.org/10.14778/3236187。3236208[30] G. Gill、R. Dathathri、L. Hoang 和 K. Pingali，“大规模分布式平台上图分析的划分策略研究”，VLDB基金会会刊，第12卷，第 4 期，2018年12月。 [在线]。可用地址：https://doi.org/10 14778 3297753 3297754 。/。V. Md、S. Misra、G. Ma、R. Mohanty、E. Georganas、A. Heinecke、D. Kalamkar、N.K.Ahmed和S.Avancha，“Distgnn：大规模图神经网络的可扩展分布式训练”，载于高性能计算、网络、存储与分析国际会议论文集，系列 SC ’21。美国纽约州纽约市：计算机协会，2021年。 [在线]。可用地址：https://doi.org/10 1145 3458817 3480856 。/。S. Fortunato，“图中的社区检测”，《物理学报告》，

可获取 488 https://www.sciencedirect.com/science/article/pii/S 0370157300002841 。[33] U. Brandes 和 T. Erlebach 编，《网络分析》。德国柏林：Springer-Verlag，2005年。[34] D. J. Watts 和 S. H. Strogatz，“小世界网络的集体动力学”，《自然》，第393卷，第6684期，第440–442页，1998年。[35] M. E. J. Newman，“复杂网络的结构与功能”，《SIAM评论》，第45卷，第2期，第167–256页，2003年。 [在线]。可获取：https://doi.org/ 10 1137 003614450342480 。/S L. [36] Page、S. Brin、R. Motwani 和 T. Winograd，“PageRank引文排名：为万维网带来秩序。”斯坦福InfoLab，技术报告，1999年。[37] U. N. Raghavan、R. Albert 和 S. Kumara，“检测大规模网络中社区结构的近线性时间算法”，《物理评论E》，第76卷，第3期，第036106页，2007年。[38] J. Leskovec 和 A. Krevl，“SNAP数据集：斯坦福大型网络数据集集合”，http://snap.stanford.edu/data，2014年6月。[39] P. Boldi、B. Codenotti、M. Santini 和 S. Vigna，“Ubicrawler：可扩展的完全分布式网络爬虫”，《软件：实践与经验》，第34卷，第8期，第711–726页，2004年。[40] P. Boldi 和 S. Vigna，“WebGraph框架 I：压缩技术”，载于《第十三届国际万维网大会论文集》（WWW 2004）。美国曼哈顿：ACM出版社，2004年，第595–601页。[41] P. Boldi、M. Rosa、M. Santini 和 S. Vigna，“分层标签传播：用于压缩社交网络的多分辨率无坐标排序”，载于《第20届国际万维网大会论文集》，S. Srinivasan、K. Ramamritham、A. Kumar、M. P. Ravindra、E. Bertino 和 R. Kumar 编。ACM出版社，2011年，第587–596页。[42] R. A. Rossi 和 N. K. Ahmed，“网络数据存储库：交互式图分析与可视化”，载于 AAAI，2015年。 [在线]。可获取：http://networkrepository.com [43] D. Chakrabarti、Y. Zhan 和 C. Faloutsos，《R-MAT：图挖掘递归模型》，2004年，第442–446页。[44] F. Khorasani、R. Gupta 和 L. N. Bhuyan，“图形处理器上的可扩展SIMD高效图处理”，载于《第24届并行体系结构与编译技术国际会议论文集》，丛书 PACT ’15，2015年，第39–50页。[45] D. Chakrabarti 和 C. Faloutsos，“图挖掘：定律、生成器与算法”，《ACM计算综述》，第38卷，第1期，2006年6月。[在线]。可获取：https://doi.org/ 10 1145 1132952 1132954 。/ . A. Bonifati、I. [46] Holubova、A. Prat-Perez 和 S. Sakr，“图生成器：最新进展与开放挑战”，《ACM计算综述》，第53卷，第 2， 2020 [ ] 10 1145 3379445 4月。在线。可获取：https://doi.org/。/ H. Park 和 M.-S. Kim，“TrillionG：使用递归向量模型的万亿规模合成图生成器”，载于《2017年ACM国际数据管理会议论文集》，丛书 SIGMOD ’17。美国纽约：计算机协会，2017年，第913–928页。 [在线]。可获取：https://doi.org/ 10 1145 3035918 3064014 。/ . [47] R. C. Murphy、K. B. Wheeler、B. W. Barrett 和 J. A. Ang，“图500简介”，Cray用户组（CUG），第19卷，第45–74页，2010年。[49] J. Kunegis，“KONECT：科布伦茨网络集合”，载于《第22届国际万维网大会论文集》，丛书 WWW ’13 Companion。美国纽约：计算机协会，2013年。 [在线]。可获取：https://doi.org/10.1145/2487788。2488173[50] A.-L. Barab´asi 和 R. Albert，“随机网络中标度的涌现”，《科学》，第286卷，第5439期，第509–512页，1999年。 [在线]。可获取：https://science.sciencemag.org/content/ 286 5439 509 / / A. A. [51] Hagberg、D. A. Schult 和 P. J. Swart，“使用NetworkX探索网络结构、动力学和功能”，载于《第七届Python科学会议论文集》，G. Varoquaux、T. Vaught 和 J. Millman 编，美国加利福尼亚州帕萨迪纳，2008年，第11–15页。[52] Z. Zeng、B. Wu 和 H. Wang，“用于加速大规模分布式图挖掘的并行图划分算法”，载于《第一届大数据、流与异构源挖掘国际研讨会：算法、系统、编程模型与应用论文集》，丛书 BigMine ’12。美国纽约：计算机协会，2012年。 [在线]。可获取：https://doi.org/ 10 1145 2351316 2351325 。/ . [53] W. Fan 、 R. Jin 、 M. Liu 、 P. Lu 、 X. Luo 、 R. Xu 、 Q. Yin 、 W. Yu 和 J. Zhou，“应用驱动的图划分”，载于《2020年ACM SIGMOD国际数据管理会议论文集》，丛书 SIGMOD ’20。美国纽约：计算机协会，2020年。 [在线]。地址：https://doi.org/ 10 1145 3318464 3389745 。/。C.-H. Wu、J.-M. Ho 和 D. [54] Lee，“基于支持向量回归的旅行时间预测”，IEEE智能交通系统汇刊，第 5 卷，第 4 期，第 276–281 页，2004 年。[55] S. Alrefaee、S. Al Bakal 和 Z. Algamal，“使用黑洞算法优化支持向量回归的超参数”，《非线性分析与应用国际期刊》，第 13 卷，第 1 期，第 3441–3450 页，2022 年。 [在线]。地址：https://ijnaa.semnan. ac.ir/article 6107.html[56] L. Breiman，“随机森林”，《机器学习》，第 45 卷，第 1 期，第 5–32 页，2001 年。[57] T. Chen 和 C. Guestrin，“Xgboost：一种可扩展的树提升系统”，载于第 22 届 ACM SIGKDD知识发现与数据挖掘国际会议论文集，KDD ’16 系列。美国纽约州纽约市：计算机协会，2016 年，第 785–794 页。[58] V. Borisov、T. Leemann、K. Seßler、J. Haug、M. Pawelczyk 和 G. Kasneci，“深度神经网络与表格数据：综述”，IEEE神经网络与学习系统汇刊，第 1–21 页，2022 年。[59] L. Grinsztajn、E. Oyallon 和 G. Varoquaux，“为什么基于树的模型在表格数据上的表现仍优于深度学习？”[在线]。地址：https://arxiv.org/abs/ 2207 08815 。Y. Gorishniy、I. [60] Rubachev、V. Khrulkov 和 A. Babenko，“重新审视表格数据的深度学习模型”，神经信息处理系统进展，第 34 卷，第 18 932–18 943 页，2021 年。[61] R. Shwartz-Ziv 和 A. Armon，“表格数据：深度学习并非你所需的一切”，信息融合，第 81 卷，第 84–90 页，2022 年。[62] M. Mohri、A. Rostamizadeh 和 A. Talwalkar，《机器学习基础》，自适应计算与机器学习系列。麻省理工学院出版社，2012年。[63] E. Kreyszig，《高等工程数学》。约翰·威立父子公司，2010年。[64] N. Merkel、R. Mayer、T. A. Fakir 和 H.-A. Jacobsen，“GitHub 代码仓库”，2023 年 1 月。 [在线]。地址：https://github.com/EASE23/ EASE23[65] H. Cai、V. W. Zheng 和 K. C.-C. Chang，“图嵌入综述：问题、技术与应用”，IEEE知识与数据工程汇刊，第 30 卷，第 9 期，第 1616–1637 页，2018 年。[66] W. Hamilton、Z. Ying 和 J. Leskovec，“大型图上的归纳式表示学习”，载于神经信息处理系统进展，I. Guyon、U. V. Luxburg、S. Bengio、H. Wallach、R. Fergus、S. Vishwanathan 和 R. Garnett 编，第 30 卷。Curran Associates, Inc.，2017 年。[67] T. N. Kipf 和 M. Welling，“基于图卷积网络的半监督分类”，载于国际学习表征会议（ICLR），2017 年。[68] F. Scarselli、M. Gori、A. C. Tsoi、M. Hagenbuchner 和 G. Monfardini，“图神经网络模型”，IEEE神经网络汇刊，第 20 卷，第 1 期，第 61–80 页，2009 年。[69] F. Pedregosa、G. Varoquaux、A. Gramfort、V. Michel、B. Thirion、O. Grisel、M. Blondel、P. Prettenhofer、R. Weiss、V. Dubourg 等，“Scikit-learn：Python 中的机器学习”，《机器学习研究期刊》，第 12 卷，第 2825–2830 页，2011 年。[70] I. G. Kolokasis 和 P. Pratikakis，“量体裁衣：针对计算定制分区”，载于第 2 届图数据管理经验系统（GRADES）与网络数据分析（NDA）联合国际研讨会论文集，GRADES-NDA’19 系列。美国纽约州纽约市：计算机协会，2019 年。 [在线]。地址：https://doi.org/ 10 1145 3327964 3328498 。/。P. Erdos 和 A. [71] Renyi，“随机图的演化”，载于《匈牙利科学院数学研究所出版物》，1960 年，第 17–61 页。[72] J. Leskovec、D. Chakrabarti、J. Kleinberg 和 C. Faloutsos，“利用克罗内克乘法生成和演化逼真且数学上易于处理的图”，载于《数据库中的知识发现：PKDD 2005》，A. M. Jorge、L. Torgo、P. Brazdil、R. Camacho 和 J. Gama 编。柏林，海德堡：施普林格，2005 年，第 133–145 页。[73] L. Akoglu 和 C. Faloutsos，“Rtg：一种采用随机打字的递归逼真图生成器”，载于机器学习与数据库知识发现联合欧洲会议。施普林格，2009 年，第 13–28 页。[74] T. G. Kolda、A. Pinar、T. Plantenga 和 C. Seshadhri，“一种具有社区结构的可扩展生成式图模型”，《工业与应用数学学会期刊》

《科学计算》，第 36 卷，第 5 期，第 C424–C452 页，2014年。 [在线]。地址：https://doi.org/ 10 1137 130914218 。/ S. Edunov、D. Logothetis、C. Wang、A. Ching 和 M. Kabiljo，“使用 darwini 生成合成社交图”，载于*2018 IEEE 第 38 届分布式计算系统国际会议（ICDCS）*，2018 年 7 月，第 567–577 页。[75] [76] M. Zwolak、Z. Abbas、S. Horchidan、P. Carbone 和 V. Kalavri，“Gcnsplit：界定流式图划分的状态”，载于利用人工智能技术处理数据（*aiDM'22*）。美国纽约州纽约市：计算机协会，2022 年。 [ ] 10 1145 3533702 3534920 在线。地址：https://doi.org/。/。[77] G. M. Slota、S. Rajamanickam、K. Devine 和 K. Madduri，“数分钟内划分万亿边图”，载于*2017 IEEE* 国际并行与分布式处理研讨会（*IPDPS*），2017 年 5 月，第 646–655 页。[78] W. Fan、M. Liu、C. Tian、R. Xu 和 J. Zhou，“图分区算法的增量化”，*VLDB*基金会会刊，第 13 卷，第 10 期，第 1261–1274 页，2020 年。[79] W. Fan、C. Hu、M. Liu、P. Lu、Q. Yin 和 J. Zhou，“并行图计算的动态扩缩容”，*VLDB*基金会会刊，第 12 卷，第 8 期，第 877–890 页，2019 年。[80] T. Kraska、A. Beutel、E. H. Chi、J. Dean 和 N. Polyzotis，“采用学习型索引结构的理由”，载于*2018*年数据管理国际会议论文集，SIGMOD '18 系列。美国纽约州纽约市：计算机协会，2018 年。 [在线]。地址：https://doi.org/ 10 1145 3183713 3196909 。/。X. Wang、C. Qu、W. Wu、J. Wang 和 Q. Zhou，“我们是否已准备好进行学习型基数估计？” *VLDB*基金会会刊，第 14 卷，第 9 期，5 月 2021 [ ] 10 14778 3461535 3461552 。在线。地址：https://doi.org/。/。[82] J. Zhang、Y. Liu、K. Zhou、G. Li、Z. Xiao、B. Cheng、J. Xing、Y. Wang、T. Cheng、L. Liu、M. Ran 和 Z. Li，“一种采用深度强化学习的端到端自动云数据库调优系统”，载于*2019*年数据管理国际会议论文集，SIGMOD '19 系列。美国纽约州纽约市：计算机协会，2019 年。 [在线]。地址：https://doi.org/ 10 1145 3299869 3300085 。/。L. Gottesburen、T. Heuer、P. Sanders 和 S. Schlag，“可扩展的共享内存超图划分”，载于*2021*年算法工程与实验研讨会论文集（*ALENEX*）。SIAM，2021 年，第 16–30 页。[83] [84] I. Kabiljo、B. Karrer、M. Pundir、S. Pupyrev 和 A. Shalita，“社交哈希分区器：一种可扩展的分布式超图分区器”，*VLDB*基金会会刊，第 10 卷，第 11 期，2017 年 8 月。 [在线]。地址：https://doi.org/ 10 14778 3137628 3137650 。/。G. Karypis、R. Aggarwal、V. Kumar 和 S. Shekhar，“多层超图划分：在 VLSI 领域的应用”，*IEEE*超大规模集成（*VLSI*）系统汇刊，第 7 卷，第 1 期，1999 年。[85] [86] U. Catalyurek 和 C. Aykanat，“基于超图划分的并行稀疏矩阵向量乘法分解”，*IEEE*并行与分布式系统汇刊，第 10 卷，第 7 期，第 673–693 页，1999 年 7 月。[87] C. Mayer、R. Mayer、S. Bhowmik、L. Epple 和 K. Rothermel，“Hype：采用邻域扩展的大规模超图划分”，载于*2018 IEEE* 国际大数据会议（*Big Data*），2018 年 12 月，第 458–467 页。[88] W. Jiang、J. Qi、J. X. Yu、J. Huang 和 R. Zhang，“Hyperx：一种可扩展的超图框架”，*IEEE*知识与数据工程汇刊，第 31 卷，第 5 期，第 909–922 页，2019 年。[89] B. Heintz、R. Hong、S. Singh、G. Khandelwal、C. Tesdahl 和 A. Chandra，“Mesh：一种灵活的分布式超图处理系统”，载于*2019 IEEE* 云工程国际会议（*IC2E*），2019 年，第 12–22 页。