arXiv:2603.07845v1 [cs.SI] 8 Mar 2026

公交因子计算的理论与实践

Sebastiano A. Piccolo[a,*], Pasquale De Meo[b], Giorgio Terracina[a], Gianluigi Greco[a]

[a]*DeMaCS*，卡拉布里亚大学，*Viale P. Bucci*，伦代，科森扎，*87036*，意大利，[b]*DICAM*，墨西拿大学，*98122*，墨西拿，意大利，

摘要

巴士因子是一种衡量人员可用性相关项目风险的指标，非正式定义为：如果某些人员突然无法工作，导致项目停滞或严重延误，这些人员的数量。尽管这一概念直观易懂，现有的巴士因子度量却依赖异质的建模假设和含糊的失败定义，以及领域特定产物，因此其通用性、可比性以及捕捉项目碎片化的能力都受到限制。本文将项目建模为由人员和任务构成的二分图，并将巴士因子的计算转化为一系列组合优化问题，以此构建一个统一的领域无关巴士因子估计框架。在此框架下，我们对巴士因子的两种互补解释——冗余和关键性——进行形式化并加以统一，分别对应最大冗余集和最小关键集，并证明这两种表述均为NP难。在这一理论基础上，我们提出一种受网络鲁棒性启发的新型巴士因子度量。与仅依据任务覆盖的既有方法不同，该度量通过跟踪逐步移除贡献者时最大的连通任务集，同时捕捉覆盖损失和项目碎片化加剧。该度量经过归一化，不依赖阈值，且适用于不同领域；我们还证明，其精确计算同样是NP难的。此外，我们针对所有考察的度量提出了高效的线性时间近似算法。最后，我们依据项目管理理论得出的预期，通过对项目结构施加受控扰动开展敏感性分析，评估这些度量的表现。结果表明，基于鲁棒性的度量与这些预期一致，相较于现有替代方案，能够更具信息量且更稳定地评估项目风险。

关键词：巴士因子、二分图、网络鲁棒性、最大冗余集、最小关键集

1. 引言

项目的巴士因子（也称为卡车因子）通常被定义为：必须有多少人员从项目中消失——仿佛被巴士撞到了一样——才会导致项目停滞或出现重大延误；这可能是由于缺乏具备相关知识的人员 [1]，也可能是由于项目碎片化程度过高 [2]。

研究发现，许多项目的劳动分工极不均衡：少数人员承担了大部分总工作负荷 [3]。因此，即使只有少数专业人员无法参与，也可能对项目完成造成严重后果。由此可见，巴士因子反映了组织和项目面临的真实风险，其实际重要性已得到广泛认可 [4]。

尽管计算巴士因子直观上颇具吸引力，但其中涉及若干不容忽视的设计选择，包括如何对项目建模、如何表示贡献者知识，以及如何定义项目停滞 [5]。现有方法对巴士因子的概念理解各不相同。具体而言，巴士因子有两种理解方式：一种从冗余的角度出发，量化有多少人员可以安全离开项目而不造成损害 [1], ；另一种从关键性的角度出发，量化其离开会危及项目的最小集合 [6]。

*通讯作者电子邮箱：sebastiano.piccolo@unical.it（Sebastiano A. Piccolo）、pdemeo@unime.it（Pasquale De Meo）、giorgio.terracina@unical.it（Giorgio Terracina）、gianluigi.greco@unical.it（Gianluigi Greco）

已获准在 *Information Sciences* 上发表 *2026*年*3*月*10*日

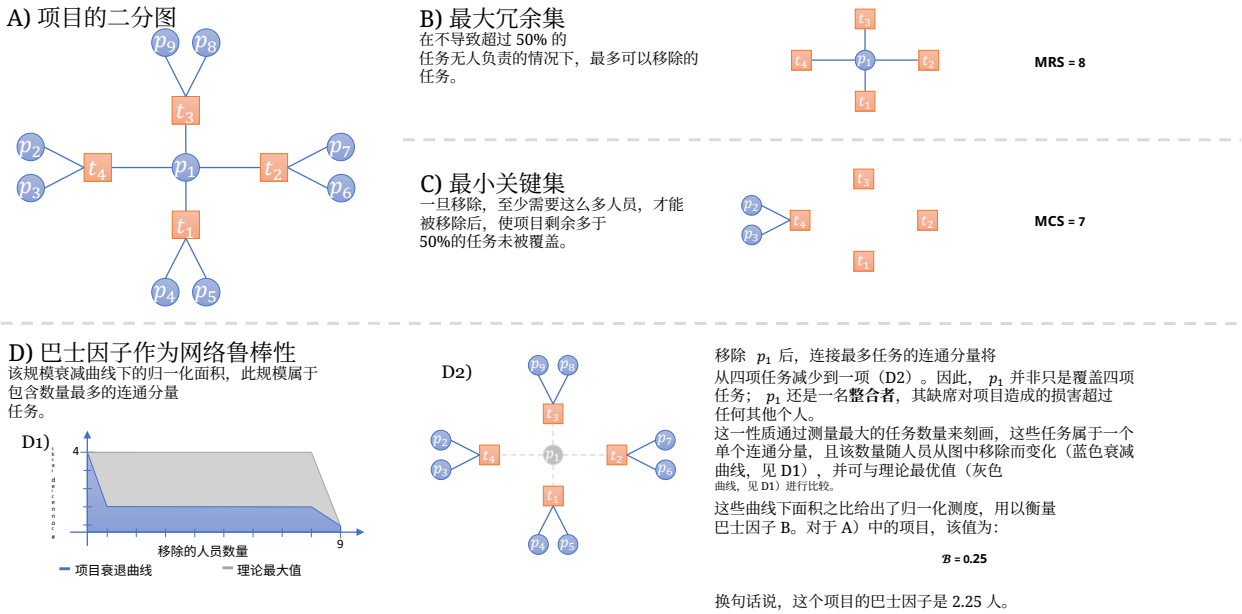


图1：用于估计项目巴士因子的各度量的可视化比较。考虑**A)**中的玩具项目。当p_1 离开时，项目碎片化为四个不连通分量。在进一步移除后，任务很快变得关键性地依赖于单个贡献者。**B)** 最大冗余集（MRS）通过移除冗余贡献者，估计巴士因子为8。**C)** 最小关键集（MCS）通过应用覆盖阈值估计巴士因子为7，而忽略项目拓扑。**D)** 我们基于鲁棒性的度量通过评估人员离开后仍保持连接在单个连通分量中的最大任务数量来刻画碎片化，从而得到一个归一化的、无阈值的巴士因子，并与项目直觉一致。

这种概念上的差异可能导致对同一项目得出截然不同的巴士因子估计值 [7]。此外，现有方法取决于其对领域特定产物（尤其是 GitHub 元数据）的使用方式：有些方法只考虑提交 [6], ，有些考虑文件所有权 [1], ，有些使用完整历史记录 [8] ，还有些则对近期提交或贡献赋予更高权重 [4]。这种差异性使得评估和解读不同的巴士因子估计方法变得困难。

因此，在本文中，我们建立了一个统一的理论框架，使我们能够推理贡献者离开项目时所产生的级联效应，并将巴士因子的计算与领域特定产物解耦。我们采用一种极简而富有表现力的表示，将项目建模为连接人员与任务的二分图，并将先前的巴士因子定义凝练为二分图上的两个组合问题：最大冗余集（MRS）和最小关键集（MCS），它们分别刻画巴士因子的冗余解释与关键性解释。

这一框架表明，先前依赖任务覆盖和固定阈值来界定项目失败的方法，无法刻画项目碎片化以及整合者 [2] 的作用——这些贡献者一旦缺席，便会使项目中原本相互独立的连通分量彼此断开。图1说明了这一弱点。

考虑这个以人员与任务的二分图表示的示例项目（图**1A**）。该项目的结构依赖于单一整合者，即人员p_1,，其将由各项任务$t_1,\ldots,t_4$所定义的四个项目模块维系在一起。直观而言，一旦p_1 离开，项目便会分裂为四个孤立模块，每个模块包含一项任务和两名人员。在这种碎片化状态下，哪怕再有一名人员无法参与，项目也会面临风险，因为某项任务届时可能仅依赖单一贡献者。因此，该项目的巴士因子应介于两到三名人员之间。

然而，若采用上述任一界定，我们会发现MRS将巴士因子估计为8名人员（图**1B**），而MCS得出的巴士因子为7名人员（图**1C**）。这种高估直接源于经由阈值定义的停滞条件，以及基于覆盖率的度量未能考虑整合者这一事实。这凸显了基于覆盖率的度量和阈值的局限性，并说明有必要提出一种能够在贡献者离开时刻画项目碎片化的巴士因子表述。

在本文中，我们通过借鉴网络的理念，构建了这样一种度量（以下简称为 鲁棒性）

鲁棒性的概念，提出了一种这样的度量（下文称为**鲁棒性**），并将任务覆盖替换为与单个连通分量相连的最大任务数（图 **1D**）。此外，我们通过计算项目图中人员逐步移除时最大连通任务数的衰减曲线下方面积，避免了使用固定阈值（图 **1D1**）。我们将该面积除以理论最大值进行归一化；理论最大值是在人员和任务数量相同的完全二部图中得到的。我们的度量得出的归一化分数为 0.25，等价于巴士因子为 2.25 人，这与我们的直觉预期一致。

除提出用于计算巴士因子及一种新度量的统一理论框架外，本文还对巴士因子的定义进行了理论处理，确定了其计算复杂度，提出了可扩展的近似算法，并通过项目管理理论指导下的系统实验和敏感性分析，对其实际表现进行了评估。我们希望本文能成为有意研究巴士因子度量的研究人员和实践者的起点。为促进这一过程以及该领域的未来研究，我们公开了代码和生成的数据 [1]。

贡献总结。 我们将项目建模为二分图 $G = (P, T, E)$，其中P 是由n 名人员组成的集合，T 是由m 个任务组成的集合，并将巴士因子的计算转化为此类图上的一系列组合问题，从而将其定义与特定领域的工件解耦。

在第 4.1 节中，我们通过形式化既有方法，建立了一个统一的、领域无关的理论框架，并证明了基于冗余和基于关键性的巴士因子解释之间的等价性（命题 1）。这一洞见使我们能够将现有度量重新表述为两种互补形式：最大冗余集（MRS），即能够在不妨碍项目进展的情况下被移除的最大贡献者集合；以及最小关键集（MCS），即其被移除后会使项目停滞的最小集合。我们证明了这两种形式都是 NP难的（定理 1 和 2）。

在这一框架的基础上，我们在第 4.3 节引入了一种受网络鲁棒性启发的新型巴士因子度量（鲁棒性），用于评估从项目图中逐步移除贡献者时最大连通任务集的规模。通过将巴士因子定义为所得衰减曲线下的归一化面积，该度量同时捕捉任务覆盖的损失和项目碎片化程度的增加。我们证明，这一表述同样是 NP难问题（定理 3），从而解决了 Zhao 等人 [9]提出的一个开放问题。

随后，我们为这三种度量开发了高效的线性时间近似算法，并分析了它们在基于度的移除策略下最坏情况下的近似保证（第 5 节）。此外，我们通过将基于度的移除与鲁棒性分析中常用的标准基于中心性的策略进行比较，实证评估了这些算法的性能（第 6 节）。

最后，在第 7 节中，我们进行了敏感性分析，以评估 MRS、MCS 和鲁棒性作为巴士因子指标的适用性。我们引入了一组测试，用于模拟针对项目结构的干预策略，其依据是从项目管理理论中推导出的预期。结果表明，鲁棒性始终与这些预期相符，并且具有更优的敏感性和实践相关性；相比之下，MRS 过于乐观，而 MCS 则由于无法捕捉项目碎片化而表现出违反直觉的行为。

2. 相关工作

巴士因子已在多个研究领域得到研究，其中最突出的是软件工程。在该领域，巴士因子用于评估项目因关键贡献者流失而面临的脆弱性。尽管其定义直观，先前研究在项目建模方式、贡献者知识推断方式以及停滞定义方面存在显著差异。本节围绕概念假设梳理相关文献，旨在阐明现有方法的优缺点，并说明为何需要一种不依赖特定领域的表述。

基于制品且针对 *GitHub* 的巴士因子度量。大多数关于巴士因子的实证研究都依赖从版本控制系统，尤其是 GitHub 仓库中提取的数据。在这些方法中，任务通常被等同于文件，而贡献者知识则根据提交、编辑或代码所有权等历史活动推断得出。

Zazworka 等人 [1] 提出了一系列基于覆盖率的度量，将项目定义为：在移除贡献者后，若仍有一定比例的文件处于已覆盖状态，则该项目是安全的。尽管这一方法在概念上颇具吸引力，但它并未明确

[1] https://github.com/sapiccolo/BusFactorX

对开发者知识进行建模 [5]，而且对于贡献者超过几十人的项目而言，计算上不可行 [10]。

后续研究进一步完善了推断贡献者知识的方法。Cosentino 等人 [8] 基于贡献阈值提出了主要开发者与次要开发者的概念，而 Avelino 等人 [6]则提出了一种启发式方法：通过作者度（Degree of Authorship，DoA）衡量开发者的知识水平，并迭代移除知识最丰富的开发者，直到超过 50% 的文件变得无人维护。这种启发式方法此后已被多项后续研究采用 [4]。

这些方法有着共同的假设：任务对应软件制品，贡献者知识根据仓库活动推断，而项目停滞则通过固定阈值来判定。

迈向领域无关的公交因子度量。基于制品的方法存在局限性，因此研究者开始探索更抽象的公交因子定义，重点关注人员与任务之间的分配结构。Piccolo等人 [11] 提出将项目建模为连接贡献者与任务的二分图，从而有效地将贡献度估算与公交因子的计算分离开来。这种抽象使公交因子分析能够应用于软件工程以外的各种协作场景。在同一项研究中，他们借鉴网络鲁棒性的概念定义了公交因子，并使用该度量提高了一个现实世界生物质发电厂项目的公交因子。

然而，他们的研究既没有对以往定义进行统一的理论阐述，也没有与以往的度量进行系统比较。此外，他们没有分析基于任务覆盖的形式化在刻画项目碎片化以及整合项目多个连通分量的贡献者所发挥的作用方面的局限性。

网络鲁棒性与项目韧性。 评估贡献者流失情况下项目脆弱性的问题，与网络鲁棒性研究密切相关；网络鲁棒性研究考察节点或边被移除时网络功能如何退化。Albert等人 [12] 的开创性研究表明，具有异构度分布的网络能够抵御随机失效，但极易受到针对高度节点的攻击。这种现象已在迥异的系统中得到观察，包括互联网 [12], 犯罪网络 [13],以及工程系统 [2]。这种普遍性使网络鲁棒性成为构建巴士因子估计理论的理想起点。

理论研究通过渗流理论解释了这种普遍现象的根源 [14] ，后续研究则提出了鲁棒性度量，其依据要么是在逐步移除节点时最大连通分量的大小 [15]，要么是图谱 [16]。近期研究探索了二分网络中的鲁棒性 [16], ，以及超图中的渗流现象，包括所谓的锚节点在维持全局连通性方面所发挥的作用 [17]。

尽管有这些进展，经典鲁棒性度量并不能直接应用于巴士因子估计。它们通常将所有节点视为等价，并关注节点可达性，而项目网络在贡献者与任务之间存在固有的不对称性。此外，鲁棒性度量很少以能够在不同规模项目之间进行有意义比较的方式进行归一化。

弥合网络鲁棒性与巴士因子估计之间的差距，需要一种尊重项目二分性质、从人员损失角度解释鲁棒性，并同时刻画任务覆盖与项目碎片化的表述。

综合。 总之，现有的巴士因子度量要么依赖领域特定产物以及基于阈值的项目停滞概念，要么采用缺乏统一理论刻画以及度量之间可比性的抽象表述。尽管网络鲁棒性为推理复杂系统的韧性提供了强有力的概念框架，经典鲁棒性度量并不能直接应用于项目场景。

我们的工作通过将项目建模为由人员和任务构成的二分图，并从网络鲁棒性的角度解读巴士因子，弥补了这些局限。我们的框架统一了以冗余或关键性为基础的既有定义，确定了它们的计算复杂度，并提出了一种归一化的基于鲁棒性的度量，能够以不依赖具体领域的方式同时衡量任务覆盖和项目碎片化。

3. 预备知识

网络是一种便捷的数学形式化工具，可用于表示和分析复杂系统，重点关注这类系统中各连通分量的连接方式。连通分量以顶点表示，它们之间的连接则以顶点之间的边表示。图方法足够灵活，能够表示和分析异质系统及其多层连接 [18], ，这些连接会随时间演变 [19]。

图是一个二元组 $G = (V, E)$，其中 V 是顶点集合，$E \subseteq V \times V$ 是边集合。图通过其邻接矩阵 A 表示：若顶点 i 与 j 相连，则 $A_{ij} = 1$，否则 $A_{ij} = 0$。顶点 i 的度 k_i 是与其相连的顶点数目，即 $k_i = \sum_j A_{ij}$。

图 G 的子图 是由 G 的顶点子集 $S \subseteq V$ 所诱导的图，记为 $G[S]$。图的连通分量 是一个极大连通子图。完全连通子图称为团。包含G 每条边至少一个端点的节点集合称为顶点覆盖。

若顶点集合 V 可以划分为两个集合 V_1 与 V_2，且满足 $V_1 \cap V_2 = \emptyset$ 以及 $V_1 \cup V_2 = V$，使得$\forall (i, j) \in E \; i \in V_1 \wedge j \in V_2$, ，则该图称为二分图。在本文其余部分，我们将二分图记为元组 $\mathcal{G} = (V_1, V_2, E)$。

为确立在各种表述下计算巴士因子的NP难性，我们使用来自已知NP困难问题的归约。我们采用以下问题，其形式定义如下。它们的NP难性证明可在Garey和Johnson [20]中找到。

集合覆盖 给定一个全集U，含n 个元素，一个族 $\mathcal{F} = \{X_1, X_2, \ldots, X_m\}$，由$m$ 个U 的子集组成，使得$\bigcup_i X_i = U$，以及一个正整数$k \leq m$，是否存在一个子集$S \subseteq \mathcal{F}$，满足 $|S| \leq k$，使得$\bigcup_{s \in S} s = U$?

团 给定一个 n无向图$G = (V,E)$以及一个正整数$k \leq |V|$，G 是否包含一个大小至少为k的团?

顶点覆盖 给定一个无向图$G = (V,E)$以及一个正整数$k \leq |V|$，G 是否包含一个大小至多为k的顶点覆盖?

4. 巴士因子的计算理论

在本节中，我们介绍一个理论框架，用于理解和建模项目巴士因子的计算。我们的理论贡献有两方面。首先，我们将现有方法系统化为二分图上的组合问题，并以两种互补的形式进行表述，即最大冗余集和最小关键集。其次，考虑到现有方法的缺陷，我们提出一种受网络鲁棒性启发的新型巴士因子度量，以解决这些不足。我们证明了所有提出的巴士因子表述都是 NP难的。

我们的出发点是将项目抽象为二分图 $G = (P,T, E)$，其中 P 是由 n 名人员组成的集合，T是由 m 项任务组成的集合，而 E 是表示人员与任务分配关系的边集。在某些项目中，例如拥有众多志愿贡献者的开源项目，并非每个人对自己编辑的每个文件都有实质性贡献。例如，某人可能只是更新文件的许可证详情。为解决这一问题，可以应用依托领域知识或针对特定领域的图过滤策略；如果某人不具备关于某项任务的有意义的知识，就移除该人员与任务之间的边。例如，GitHub 特有的启发式方法——作者度（DoA） [4], 旨在识别每个源代码文件的有意义的贡献者，在我们的框架中自然就成为图过滤策略。

现在，我们来考察计算巴士因子的现有方法，并将其形式化为二分图上的两个组合问题。

4.1. 将现有方法系统化为二分图上的组合问题

Zazworka 等人 [1]最先考察了计算巴士因子这一问题，并定义了以下一系列度量:

$$Z_{\alpha,t}(G) = \max\{k \in \mathbb{N} : \mathrm{cov}_\alpha(G, k) \geq tm\} \tag{1}$$

使用聚合函数 $\alpha \in \{\min、\max、\mathrm{avg}\}$ 以及 $0 < t \leq 1$。因此，$\mathrm{cov}_\alpha(G, k)$表示$G$ 的最小、最大或平均覆盖——也就是在移除之后仍然属于已覆盖集合的任务的最小/最大/平均数量

任意k 名人员的子集。由于$Z_{avg,t}(G)$被发现并不充分 [10], ，我们在此不予考虑，并将分析限定于$Z_{min,t}(G)$和$Z_{max,t}(G)$。直观而言，公式1从结构上冗余的人员角度定义巴士因子，这些人员可以离开项目而不对其造成损害。结构上冗余的人员数量越多，巴士因子就越高。

Avelino 等人 [6], 则非正式地将巴士因子定义为：导致超过比例 t 的任务被孤立的最少人员数，其中 t 是一个阈值，用于界定项目何时被视为停滞。我们将 Avelino 等人的定义形式化为寻找一个最小关键集的问题，定义如下：

$$MCS_t(G) = \min_{X \subseteq P} |X|, \quad \phi(G[P \setminus X]) > tm \tag{2}$$

其中 $0 < t < 1$，而 $\phi(G)$ 返回 G 中孤立任务的数量。注意，公式 2 刻画了巴士因子的非正式概念：必须有最少数量的人员无法参与，项目才会陷入停滞。最小关键集越大，巴士因子就越高。

现在我们证明 $Z_{min,t}(G)$ 与 $MCS_t(G)$ 之间的以下关系。

命题 1。 对于所有 $0 < t < 1$，$Z_{min,t}(G) = MCS_{1-t}(G) - 1$。

证明。 回顾一下，

$$Z_{min,t}(G) = \max\{k \in \mathbb{N} : \mathrm{cov}_{min}(G, k) \geq tm\}.$$

函数 $\mathrm{cov}_{min}(G,k)$ 返回移除 k 名人员后，仍至少被一名人员覆盖的任务的最小数量。也就是说，

$$\mathrm{cov}_{min}(G, k) = \min_{C \subseteq P,\ |C|=k} m - \phi(G[P \setminus C]).$$

因此，我们可以将$Z_{min,t}(G)$改写如下：

$$\begin{aligned} Z_{min,t}(G) &= \max_{k \in \mathbb{N}} \min_{C \subseteq P,\ |C|=k} m - \phi(G[P \setminus C]) \geq tm \\ &= \max_{k \in \mathbb{N}} \min_{C \subseteq P,\ |C|=k} \phi(G[P \setminus C]) \leq m - tm \\ &= \max_{k \in \mathbb{N}} \min_{C \subseteq P,\ |C|=k} \phi(G[P \setminus C]) \leq m(1-t). \end{aligned}$$

回想一下，$MCS_{1-t}(G)$是能够隔离超过$m(1-t)$个任务所需的最少人员数量。也就是说，$MCS_{1-t}(G)$是使 $\min_{C \subseteq P,\ |C|=k} \phi(G[P \setminus C]) > m(1-t)$成立的最小$k \in \mathbb{N}$。因此，$Z_{min,t}(G) = MCS_{1-t}(G) - 1$。 □

注意，$Z_{min,t}(G)$对于$t = 1$也有定义，即移除最少数量的人员，使剩余的覆盖小于m。这种情况并没有特别的研究意义，而且很容易求解，因为它要求用最少的人员隔离单个任务，也就是度最低的任务。

现在我们可以证明，计算最小关键集（MCS）是NP难的。

定理 1. 最小关键集是*NP*难的。

证明。 最小关键集是一个优化问题。为了证明它是 NP难，我们证明它的判定版本（即可以用是或否作答的问题）——我们称之为关键集——是 NP 完全的。在关键集问题中，给定一个二分图 $G = (P, T, E)$、一个正整数 k 和一个实数 $0 < t < 1$，我们要判断是否存在一个集合 $X \subseteq P$，满足 $|X| \leq k$，且 $\phi(G[P \setminus X]) > tm$。关键集问题属于 NP，因为可以在多项式时间内验证候选解 X：计算导出子图 $G[P \setminus X]$, 中孤立任务的数量，并检查该数量是否超过tm。

我们通过将团归约到关键集，证明关键集是 NP难的。在团问题中，给定一个简单无向图 $G' = (V', E')$ 和一个正整数 $k' < |V'|$，我们要判断是否存在一个大小至少为 k' 的完全连通子图。我们按如下方式将团归约到关键集：构造一个二分图 G，其中 $P = V'$，$T = E'$，$E = \{(a, b)$，当且仅当 $a \in P$、$b \in T$，且 a 是 b$\}$的一个端点；令 $k = k'$，并令 $t = k(k-1)/2 - 2)/|T|$（图 2A）。

在这个关键集实例中，我们要问是否存在一个大小至多为$C \subseteq P$ 的子集k，使得 $\phi(G[P \setminus C]) > t|T| = k(k-1)/2 - 2)/|T| = k(k-1)/2 - 2$。这个关键集实例~~的答案~~为“是”, ~~当且仅~~当原始团实例中存在一个大小至少为k的团。如果G' 中存在一个大小至少为k的团，那么同一组节点构成

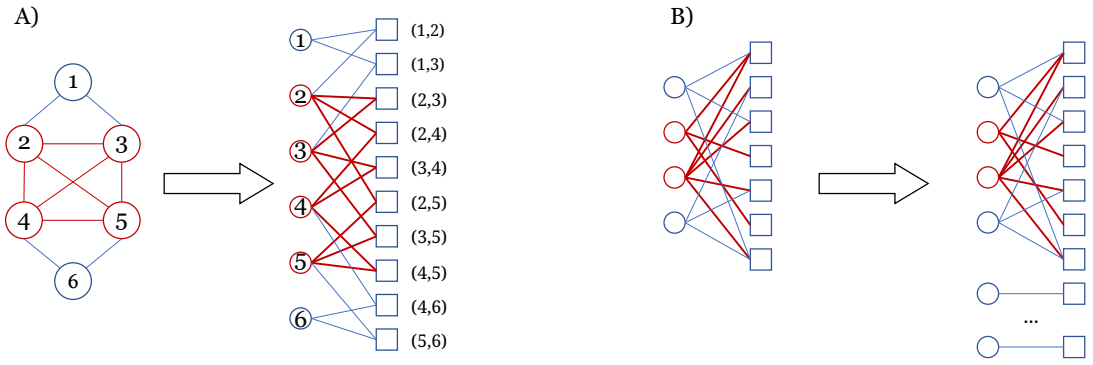


图 2：NP难性归约的可视化。**A)** 从团到最小关键集的归约（突出显示解）：原图中的节点变为二分构造中的左顶点集（P），边变为右顶点集（T）。二分图中的边表示原图中的关联。原图中的一个团对应二分图中的一个最小关键集。**B)** 为证明最大冗余集而从集合覆盖到部分集合覆盖的归约：一个集合覆盖实例（此处以突出显示解的二分图表示）通过添加适当数量的二分顶点对，转换为一个部分集合覆盖实例。

相应关键集实例的一个解。事实上，由k 个节点构成的团，是唯一一个包含恰好$k(k-1)/2$条边的k 节点子图。由于在我们的二分构造中，$P = V'$ 且$T = E'$，因此图G' 中团里的同一组节点，对应于一个子集$C \subseteq P$，使得 $\phi(G[P \setminus C]) > k(k-1)/2$。

反过来，如果二分图 G 存在一个大小至多为 k 的关键集 C，使得 $\phi(G[P \setminus C]) > k(k-1)/2$，那么 P 中这 k 个节点组成的集合会隔离 T 中至少 $k(k-1)/2$ 个节点。然而，根据构造，由于 G′ 是简单无向图，G′ 中由 k 个节点构成的子图至多包含 $k(k-1)/2$ 条边。这意味着集合 C 必须恰好由 k 个节点组成，并且恰好隔离 T 中的 $k(k-1)/2$ 个节点。因此，C 对应于 G′中的一个由 k 个节点组成的团。

因此，团 $\leq_p$ 关键集，进而关键集是 NP 完全的。 □

根据命题 1，除了最小关键集之外，我们还只需考虑$Z_{\max,\,t}(G)$。该问题通过找出最多有多少人员可以安全地变为不可用，同时不导致超过t% 的任务无法完成（即被孤立），来描述一种最佳情形。我们将这一数量形式化为二分图上的以下最大冗余集问题：

$$MRS_t(G) = \max_{X \subseteq P} |X| \quad \text{s.t.} \quad \phi(G[P \setminus X]) < tm, \quad 0 < t \leq 1. \tag{3}$$

注意，当$t = 1$时，式 3 中定义的 MRS 要求找到一个集合，使其为最小集合覆盖的补集。集合覆盖可以自然地表示为二分图$G = (P, T, E)$，其中集合P 对应$\mathcal{F}$，集合T 对应全集U，而P 与T 之间的边表示 $\mathcal{F}$中各子集的覆盖关系：$E = \{(i, j) \mid i \in P,\ j \in T,\ \text{且} j \in s_i \text{ 对于某个} s_i \in \mathcal{F}\}$成立。

类似地，对于阈值 $0 < t < 1$，MRS 要求找到一个集合，它是部分集合覆盖的补集。给定一个由 m 个元素组成的全集 $U = \{u_1, u_2, \ldots, u_m\}$，一个包含 n 个子集 $S_i \subseteq U$ 的集合族 $\mathcal{F} = \{S_1, S_2, \ldots, S_n\}$，使得$\cup_i S_i = U$，一个正整数 k，以及一个实数 $0 < t < 1$，部分集合覆盖问题要判断是否存在一个子集 $X \subseteq \mathcal{F}$，满足 $|X| \leq k$，使得 $\left|\cup_{X_i \in X} X_i\right| \geq tm$。现在我们证明以下结果。

定理2. 部分集合覆盖是 *NP* 完全。

证明。 我们通过将集合覆盖问题归约到部分集合覆盖问题，证明部分集合覆盖是 NP 完全的。部分集合覆盖问题属于 NP，因为可以在多项式时间内验证候选解：只需统计它覆盖的不同元素数量，并检查该数量是否至少为 tm。

假设给定一个集合覆盖实例，其全集为U'，子集族为 $\mathcal{F}'$，且给定一个正整数k'。令 $0 < t \leq 1$。我们按如下方式将该实例转换为一个部分集合覆盖实例：通过向U中添加 $\lceil mt \rceil - m$个新元素e_i，来定义全集U'；并通过添加 $\lceil \overline{m}_t \rceil - m$个子集（每个子集都等于 $\{e_i\}$），来定义集合族 $\mathcal{F}\,\mathcal{F}'$。我们还令$k = k'$（图 **2B**）。

在这个部分集合覆盖实例中，我们要判断是否存在一个子集$X \subseteq \mathcal{F}$，满足 $|X| \leq k$ 且$\left|\bigcup_{X_i \in X} X_i\right| \geq tm$。该实例存在大小至多为$k$ 的解，当且仅当原集合覆盖实例存在大小至多为k'的集合覆盖。如果原集合覆盖实例存在大小至多为k'的解，那么该解也是对应部分集合覆盖实例的解。

反过来，假设构造出的部分集合覆盖实例存在一个大小至多为k 的解，且该解覆盖至少tm 个元素。由于新增的 $\lceil \frac{m}{t} \rceil - m$ 个子集各自恰好覆盖一个元素，它们不可能对一个大小至多为k 且覆盖至少tm 个元素的解有所贡献。因此，任何此类解都只能由 $\mathcal{F}'$ 中的子集组成，故对应于原集合覆盖实例中一个大小至多为k' 的集合覆盖。 □

推论1。 最大冗余集是*NP*难的。

证明。由于冗余集是部分集合覆盖的补集，结论由定理2得出。 . □

*4.2. MCS*和*MRS*的局限性

MCS和MRS都忽略了底层项目结构，仅根据任务覆盖情况来对贡献者进行考量。这使它们无法捕捉整合者的作用 [2], ；从拓扑学角度看，整合者对应于二分图（及超图）中的锚节点，即移除后会使网络分裂的节点 [17]。

类似地，MCS 和 MRS 假设只要至少有一名人员负责某项任务，该任务就能完成。从风险和项目管理的角度来看，这种不切实际的简化意味着，只需为每项任务增加单任务人员 （即只关联一项任务的人员），就能提高项目的巴士因子，而在真实项目中这种情况鲜少发生 [21]。

事实上，可以验证，对于由不相连二元组构成的图（即一名人员只关联一项任务），这两种度量都会赋予项目较高的巴士因子。我们将在第 7 节通过实验展示这一缺陷。此外，这两种度量都依赖一个任意阈值t （通常设为 0.5）来定义项目的停滞条件；因此，项目的巴士因子取决于t。最后，MCS 和 MRS 未经过归一化，因此会受到项目规模影响，妨碍在不同项目之间进行有意义的比较。

因此，在下一节中，我们将基于先前研究 [11], 提出一种全新的巴士因子定义，以克服上述不足。

4.3. 更好的衡量方式：作为网络鲁棒性的巴士因子

我们的衡量方式基于最大连通任务数，即单个连通分量中包含的最大任务数。如图 1 所示，使用最大连通任务数能够反映整合者的作用，因为它同时体现了任务覆盖和连通性。此外，通过计算随着人员逐步移除时最大连通任务数的衰减曲线下面积，我们无需定义明确的停滞条件，也就不再依赖任意阈值（第 4.2 节）。最后，将该面积除以理论最大值进行归一化——理论最大值是在人员数和任务数相同的完全二部图中得到的——由此获得的衡量指标可以有两种解释：*(1)* 项目相对于可能达到的最大值的鲁棒性；以及 *(2)* 预计有多少人员离开会使项目面临风险，这一数值可通过归一化巴士因子乘以项目的人员总数得到。

在正式定义我们的度量之前，我们先引入「骨干集合」这一概念。

定义 1 （骨干集合）。给定一个表示为二分图 $G = (P, T, E)$ 的项目，若集合 $S \subseteq P$ 是骨干集合，则诱导二分子图 $G[P \setminus S]$ 由若干互不连通的星形图组成，每个星形图都以单个人员为中心（即 P 中的一个顶点）。

直观而言，骨干集合包含一个项目的整合者。骨干集合越小，项目的巴士因子就越小，因而项目也就越脆弱。然而，我们不仅关注项目骨干集合的基数；相反，我们关注的是骨干集合中的人员被依次从项目中移除时的最大连通任务数。请注意，*(1)* 一组不连通星形图正是 MCS 与 MRS 失效的配置，并且*(2)* 一旦 G 被约简为一组不连通星形图，按度的递减顺序移除人员会对项目造成最大损害。

设 $\pi = (p_1, p_2, \ldots, p_k)$ 为一个移除顺序，使得集合 $S = \{p \mid p \in \pi\}$ 是大小为 k 的骨干集合。设 $G_i^\pi = G[P \setminus \{p_1, p_2, \ldots, p_i\}]$，其中 $G_0^\pi = G$，并设 $\tau(G)$ 表示 G 中的最大连通任务数。我们度量 $\tau(G_i^\pi)$ 在 $i = 1, \ldots, k$ 上的衰减曲线下面积为

$$R(G, \pi) = \sum_{i=1}^{k} \tau\left(G_i^\pi\right). \tag{4}$$

这个定义让我们能够将广泛使用的网络鲁棒性概念推广到二分图（以及超图）：将鲁棒性定义为G中最大连通任务数的衰减曲线下可能的最小面积：

$$\mathcal{R}(G) = \min_{\pi} R(G, \pi). \tag{5}$$

现在我们证明，计算式 5 中定义的二分网络鲁棒性是 NP难的。

定理 3. 二分网络鲁棒性是 *NP*难的。

证明。 我们证明二分网络鲁棒性的判定版本是 NP 完全的。二分网络鲁棒性的一个实例由一个二分图$G = (P,T, E)$和两个正整数k 和Q组成，并询问是否存在一个移除序列 π，它同时也是一个长度至多为k 的骨干集合，使得$R(G,\pi) \le Q$。

首先，注意二分网络鲁棒性属于 NP，因为验证一个候选解 π 需要计算$R(G,\pi)$，而这可以在多项式时间内完成。事实上，我们将在第 5.3 节中给出一种线性时间算法来计算$R(G,\pi)$。

我们通过将顶点覆盖问题归约到二分网络鲁棒性问题，证明二分网络鲁棒性是 NP难的。给定图 G′ = (V′, E′) 上一个参数为 k′ 的顶点覆盖实例，我们按如下方式构造一个二分网络鲁棒性实例。我们将 G′ 变换为二分图 G = (P, T, E)，其中 P = V′，T = E′，且 E = {(a, b) | a ∈ P、b ∈ T，并且 a 是 b 的一个端点}（与 MCS 使用的变换相同，见图 2A）。我们令 k = k′，并令 Q = k|T|。

如果G' 有一个大小至多为k'的顶点覆盖S，那么从G' 中移除S 中的顶点后，得到的图仅由孤立顶点组成。由于我们的构造保留了关联结构，从G 中的P 移除对应顶点后，得到的图由不连通星形图组成；每个星形图以$P \setminus$ 中不属于S 的一个顶点为中心，并连接到T 中的若干顶点，其数量等于该顶点在G'中的度。因此，G 存在一个大小至多为k的骨干集合。

反过来，假设G 存在一个大小至多为k 的骨干集合 π，使得$R(G,\pi) \le Q$。大小为k 的骨干集合的最坏情况，发生在移除人员的方式使得最大连通任务数在整个移除过程中始终不减少时。在这种情况下，对于所有$i = 1, \ldots, k$，都有 $\tau(G_i^\pi) = |T|$，因此$R(G, \pi) = k|T| = Q$。这意味着，移除 π 中的顶点必须删除G'中的所有边，因此相应的顶点在G'中构成一个大小至多为k' 的顶点覆盖。

因此，G 存在一个大小至多为k 的骨干集合 π，使得$R(G,\pi) \le Q$，当且仅当G' 存在一个大小至多为k'的顶点覆盖，因此顶点覆盖问题 $\le_p$ 二分网络鲁棒性。 □

我们的鲁棒性度量 $\mathcal{R}(G)$推广了针对无向图的一种广泛使用的网络鲁棒性度量，该度量由Schneider *et al.* [15]提出。Zhao *et al.* [9] 指出，证明Schneider*etal.* 提出的鲁棒性度量在计算上的困难性仍是一个开放问题。定理 3 解决了这一开放问题。

公式 4 中的度量在概念上与二分图的自然连通性 有关 [16], 这是一种谱测度，用于刻画网络中替代路径的冗余。当需要考虑流量和路径长度时，自然连通性是更合适的鲁棒性度量。然而，在巴士因子的情形下，我们不涉及网络中的流量；因此，测量最大连通任务数更为合适。此外，计算自然连通性比计算最大连通任务数的开销更大，因为它取决于G的邻接矩阵的所有特征值。

最后，我们将巴士因子的归一化测度定义为二分图G在移除序列 $\pi \in \Pi(P)$（其中 $\Pi(P)$ 表示 P 中所有人员可能排列的集合）下的鲁棒性，与同等规模的完全二部图的鲁棒性之比：

$$\mathcal{B}(G, \pi) = \frac{2}{m(2n-1)} \sum_{i=1}^{n} \left[\tau\left(G_{i-1}^\pi\right) + \tau\left(G_i^\pi\right)\right], \quad \text{with } G_0^\pi = G. \tag{6}$$

请注意，在公式 6 中，我们采用了与公式 4 中网络鲁棒性略有不同的表述。这两种表述只是估算曲线下面积的不同方法：公式 4 对应高斯求和，而公式 6 使用梯形法则，后者能更准确地估算曲线下面积。归一化巴士因子 $\mathcal{B}(G,\pi)$ 的取值范围为 $[0,1]$，可解释为项目鲁棒性相对于理论最大值的比例，因此能够用于比较不同项目。例如，巴士因子为 $\mathcal{B}(G,\pi) = 0.5$ 表示G 的鲁棒性只有同等规模完全连接图的一半。

由于计算一个项目的巴士因子是NP难的，下一节我们将给出高效且可扩展的算法，用以近似本框架所考虑的全部定义下的巴士因子。

5. 近似巴士因子的实践

在本节中，我们为上一节所分析的全部巴士因子表述给出近似算法，并同时兼顾近似质量与可扩展性。对于MCS和鲁棒性，我们以移除顺序 π对算法进行参数化，从而使近似质量仅取决于 π。这一设计使得，若其他移除顺序能够带来更好的近似解，则可将同一算法应用于这些替代移除顺序。

5.1. 近似最大冗余集

在第4.1节中，我们证明冗余集是部分集合覆盖的补集；因此，任何针对最小部分集合覆盖的近似算法都能给出最大冗余集的近似算法。除非P=NP，否则集合覆盖在多项式时间内最好的近似算法保证其解至多比最优解大$O(\log m)$倍 [22]。同一算法及其相同的性能保证可用于近似部分集合覆盖，因为这两个问题都可视为次模集合覆盖的特例 [23]。

回想一下，MRS 的输入是一个图 $G = (P,T,E)$ 和一个实数 $0 < t \leq 1$，并返回能够覆盖少于 tm 个任务的最大人员集合。从概念上讲，我们的近似算法按如下方式运行：在每次迭代中，将覆盖当前未覆盖任务数最多的人员加入集合 S，直到已覆盖任务超过 tm 个。由于 S 是最小部分集合覆盖的一个近似解，因此集合 $X = P \setminus S$构成了 MRS 的一个近似解。借鉴 Lim等人 [24], 的一项技术，我们提出了一种近乎线性时间的 MRS 近似算法，可扩展至超大型二分图（算法1）。

在我们的算法中，我们使用优先队列 Q 来安排任务覆盖最高的人员，并将已覆盖任务集合和集合 S 表示为位数组（算法1，第 2–4 行）。对于每个人员，我们用连续数组的区段 $\mathcal{I}$ 记录未覆盖任务，并根据人员的度为其设置优先级（第 5–8 行）。在每次迭代中，我们从 Q 中取出一名人员 p，并检查其优先级是否等于当前未覆盖任务数，即有效_覆盖数（第 9–13 行）。如果是，则将 p 加入部分集合覆盖 S，并更新已覆盖任务集合（第 13–16 行）；否则，将 p以更新后的优先级和任务覆盖数重新放回队列（第 16–19 行）。当已覆盖任务数达到至少 tm时，循环终止，算法返回集合 $P \setminus S$及其基数，作为 MRS 的解（第 20–22 行）。

算法1是 MRS 的$O(n)$近似算法。下一个定理表明，除非 P = NP，否则多项式时间算法所能达到的近似比已是最优。

定理4. 除非P = NP，否则对于任意常数 $\epsilon > 0$，将Maximum Redundant Set 近似到 $O(n^{1-\epsilon})$因子以内是 NP难的，其中 n 是人员数量。

证明（概要）。最大独立集问题是最小顶点覆盖问题的补问题 [20]。利用先前用于证明 MCS 与网络鲁棒性难度的同一变换，我们可以将无向非二分图 G' 变换为等价的二分图G。在 G 上计算最小集合覆盖等价于在 G' [20]上计算最小顶点覆盖；因此，在G上（在 $t = 1$下）计算 MRS 等价于在G'上计算最大独立集。然而，除非 P = NP，否则对每个常数 $\epsilon > 0$ [25]，将最大独立集问题近似到因子 $O(n^{1-\epsilon})$ 以内是 NP难的。由于 MRS 将最大独立集问题作为一个特例包含在内，该难度结果可转移到 MRS。

□

算法1 最大冗余集（MRS）的近似算法

```
1: 函数最大冗余集合(G, t)
2:     Q ← 最大优先队列()                          ▷ 按任务覆盖对人员排序的优先队列
3:     已覆盖集合 ←位数组(m)                       ▷用于指示已覆盖任务的位数组；初始时所有位均未设置
4:     S = ∅                                       ▷ 存储部分集合覆盖
5:     对所有 p ∈ G。人员()执行
6:         I[p] ←G.邻居(p)                         ▷用于跟踪每个人当前已覆盖任务的数组
7:         Q[p] ←G.度(p)
8:     结束 for
9:     当 已覆盖集合.count() < tm执行
10:        p，优先级 ← Q.pop()
11:        s ← I[p]
12:        有效_覆盖 ← s \ 已覆盖集合
13:        如果 |有效_覆盖| = 优先级则               ▷ 如果 p 的覆盖是最新的，则将 p 添加到S
14:            S ← S ∪ {p}
15:            已覆盖集合.union(有效_覆盖)
16:        else                                    ▷ 否则，更新 p 的覆盖，并将 p 重新插入 Q
17:            I[p] ←有效_覆盖
18:            Q.push(p, |有效_覆盖|)
19:        结束 if
20:    结束 while
21:    返回 n − |S|，P \ S                         ▷ MRS 是部分集合覆盖的补集
22: 结束函数
```

5.2. 近似求解最小关键集合

为了近似最小关键集，我们提出一种节点渗流方法，其中人员按照移除顺序 π从该图中逐步移除。我们初始化一个数组，其中包含覆盖每个任务的人员数量，即它们的度（算法 2，第 2–4 行）。当人员从G中移除时，我们降低与其相连的任务的度（第 6–9 行），在任务 t 变为孤立时更新孤立任务的计数（第 10–12 行），并跟踪已移除的人员数量（第 13–14 行）。当孤立任务的数量超过 tm（其中 t 通常设为 0.5 [6]）时，算法终止，并返回 MCS 的基数（第 15–17 行）。

算法 2 的运行时间随边的数量线性增长，因为它至多只需处理每个人员及其关联的任务，计算复杂度为 $O(|E|)$。由于算法 2 需要以移除顺序 π 作为输入，一个自然的问题是如何选择 π。一种合理的做法是按度递减的顺序移除人员（Avelino等 [6]采用了这一做法），这在实践中通常效果很好。事实上，现实世界网络往往较为稀疏，并呈现长尾度分布：只有相对少数的节点具有较高的度，而绝大多数节点的度较低 [26]。这一特征意味着，高阶节点对网络的连通性至关重要。例如，Shang 已证明，在二分图中，移除相对少量的高阶节点会大幅降低网络的自然连通性 [16]。

尽管 按度移除人员基于上述原因在实践中效果良好，但我们现在将证明，这种移除 1
移除顺序提供，在最坏情况下，最小关键集的近似比为$O(n)$。

定理5。 算法*2*中，当移除顺序 π 按度递减定义时，是最小关键集合问题的（On）-近似算法。

证明。我们通过构造一个实例来对该定理给出构造性证明，在这个实例中，算法 2 的近似比为 O(n)。令 $0 < t < 1, n, m \in \mathbb{N}^+$，使得 $n > tm + 1$。我们按如下方式构造图 $G = (P, T, E)$：集合 P 包含 n 个节点，集合 T 包含 m 个节点。我们将集合 T 划分为两个子集 T_1 和 T_2，使得$|T_1| = tm + 1$ 且 $|T_2| = (1 - t)m - 1$，其中 $T_1 \cup T_2 = T$ 且 $T_1 \cap T_2 = \emptyset$。类似地，我们将 P 划分为两个子集 P_1 和 P_2，使得 $|P_1| = tm + 1$ 且 $|P_2| = n - (tm + 1)$，其中 $P_1 \cup P_2 = P$ 且 $P_1 \cap P_2 = \emptyset$。现在定义边集 E。P_1 中的每个人员恰好与 T_1, 中的一个任务相连，从而形成 $tm + 1$ 个不相连二元组。此外，

算法 2 作为最小关键集的巴士因子

```
1: 函数最小关键集合(G, t, π)
2:     对于所有 t ∈ G。任务() 执行
3:         任务_度数[t] ←G.度(t)                    ▷ 随着人员从G中移除，跟踪任务度数
4:     结束循环
5:     mcs_size, num_孤立的 ← 0                          ▷ MCS 的大小, 以及孤立任务的数量
6:     而 num _孤立的 ≤ tm do
7:         p ← π.next()
8:         对于所有 t ∈ G.邻居(p)执行
9:             任务_度数[t] ← 任务_度数[t] − 1          ▷ 降低了p所覆盖任务的覆盖
10:            如果 任务_度数[t] = 0则
11:                num_孤立 ← 数量 _孤立 + 1                ▷ 增加孤立任务的数量
12:            结束 if
13:        结束 for
14:        mcs 大小 ← mcs 大小 + 1                ▷ 增加迄今已移除的人员数量（当前 MCS 的大小）
15:    结束 while
16:    返回 MCS _size
17: 结束函数
```

我们在P_2 中的人员与T_2中的任务之间构造一个完全二部图。令O 表示一个最优临界集，并令X 表示算法 2 在 π 按度递减顺序移除人员时返回的关键集。最优解为$O = P_1$。然而，近似算法会先移除P_2 中的人员，因为他们的度高于（或等于）P_1中的人员。因此， $|X| \geq |P_2|$。当t 趋近于零时， $|P_1|$趋近于一，而 $|P_2|$ 趋近于$n - 1$。因此， $|X| =$ 为 $\Omega(n)\,|O|$，这意味着近似比为$O(n)$。 □

换句话说，在最坏情况下，按人员度排序的移除顺序可能返回一个规模最大达到$n - 1$ 名人员的最小关键集，而最优值则可能小至 1。

5.3. 将巴士因子近似为二分图网络鲁棒性

计算鲁棒性时，随着人员从G中移除，需要遍历G中的所有连通分量，统计每个分量包含的任务数，并取其中的最大值。基于n 次重复执行深度优先搜索（DFS）的朴素实现，会导致算法的计算复杂度随人员数量呈二次增长，计算复杂度为$O(n^2m)$。

这里，我们提出一种高效实现方法：使用并查集数据结构跟踪最大连通任务数。并查集是一种不相交集合结构，其中每个集合都表示为一棵树，树的根节点是该集合的代表元素。合并两个集合时，将对应树的根节点连接起来 [22]。如果不按移除人员，而是按移除顺序的逆序添加人员，这种数据结构就能在连通分量增长时跟踪最大连通任务数。由此可得以下线性时间算法[2], , 其计算复杂度为$O(|E|)$。

算法 3 的输入为图G 和移除序列 π。我们使用包含m 个互不相交任务的并查集数据结构 $\mathcal{U}$进行初始化（第 2 行），并按相反顺序处理移除序列 π，即从最后被移除的人员处理到最先被移除的人员（第 4 行）。每次处理一名人员p时，我们合并与p 相连的任务所对应的连通分量（第 5–6 行），并在向量 tau 中存储 $\mathcal{U}$ 的最大连通任务数（第 7 行）。最后，将公式 6 应用于向量 tau，计算出巴士因子 $\mathcal{B}$。

再次按度移除人员，在实践中仍能很好地近似二分网络的鲁棒性。Piccolo等人通过实证研究证明了基于度的移除对于研究一个涉及生物质发电厂建设的现实项目的鲁棒性是有效的 [2]。相比之下，Shang [16], 通过模拟验证了渗流理论对二分网络中移除高阶节点的预测，

[2]同一算法也可用于计算任意网络的鲁棒性。

算法 3 巴士因子作为网络鲁棒性

1: 函数巴士因子(G, π)
2: $\mathcal{U} \leftarrow$并查集(G.num_任务()) ▷ 用于m 任务的并查集结构
3: tau $\leftarrow$ [0] ▷ 一个存储值 $\tau\left(G_i^{\pi}\right)$的向量
4: **对于所有** $p \in \pi$.reverse() **执行**
5: 邻居 $\leftarrow G$.neighbors(p) $\cup \{p\}$ ▷获取与p相连的任务
6: $\mathcal{U}$.union(neigs) ▷在 $\mathcal{U}$中将这些任务连接成单个连通分量
7: tau.push($\mathcal{U}$.max_num_connected_tasks()) ▷ 计算G中的最大连通任务数 $\tau\binom{\pi}{i}$
8: **结束循环**
9: $\mathcal{B} \leftarrow \frac{}{2\,m(2n-1)} \sum_{i=1}^{n}(\text{tau}[i-1] + \text{tau}[i])$ ▷ 根据公式 6 计算鲁棒性
10: **返回** $\mathcal{B}$
11: **结束函数**

并随后 y 通过引入锚节点 [17], 这一概念扩展了该理论，这些节点与
我们的积分器概念 界。

尽管按度移除在实践中效果良好，但我们将证明，在最坏情况下，它得到的网络鲁棒性近似比为$O(n)$。

定理 6。该 按度递减顺序定义的移除序列 π 是（n）-近似，可用于二部
网络鲁棒 棒性 问题。

证明。 与我们在定理5中的做法类似，我们构造一个图，对其按度移除可达到$O(n)$的近似比。设$k \in \mathbb{N}$，且$k \geq 2$。

我们按如下方式构建该图。我们实例化k 个星形图，每个由一名人员与$k+1$ 个任务组成。然后，我们再增加一名人员，并将该人员连接到先前引入的k 个星形图中每一个的一个任务。现在我们得到一棵以一名中心人员为根节点的树，该中心人员连接到k 个任务，并有k 名外围人员，每名外围人员连接到$k+1$ 个任务。这棵树由$n = k+1$ 名人员以及$m = k(k+1)$个任务组成。

该树的最优移除序列首先移除中心人员（度为k），随后移除外围人员（度为$k+1$）。移除中心人员后，网络退化为k 个星形图，每个星形图有$k+1$ 个任务。此后每次移除都会分离出$k+1$ 个任务，直到最后一名人员被移除，此时图由m个互不连通的任务组成。根据式4，该序列的总代价为$R(G,\pi^*) = k(k+1) = k^2 + k$。

算法 3（其中 π 按度递减顺序定义）则首先移除度为 k + 1 的人员，每移除一人就会使 k 个任务孤立。经过 k 步后，网络缩减为一个中心人员与 k 个任务相连。移除最后一名人员后，网络变为由孤立任务组成的集合，不再包含人员，因此最大连通分量中的任务数为零。该过程的代价为 $R(G,\pi) = (m-k) + (m-2k) + \cdots + (m-k^2) = km - kS(k)$，其中 $S(k) = k(k+1)/2 = (k^2+k)/2$。因此，$R(G,\pi) = k(k(k+1)) - k((k^2+k)/2) = k^3+k^2-(k^3+k^2)/2 = (k^3+k^2)/2$。

因此，近似比为：

$$\frac{R(G,\pi)}{R(G,\pi^*)} = \frac{k^3+k^2}{2(k^2+k)} = \frac{k^2(k+1)}{2k(k+1)} = \frac{k}{2} = \frac{n-1}{2} = O(n)$$

□

我们指出，定理 6 不仅为巴士因子，也为一般的网络鲁棒性（包括非二分图）确立了一个 $O(n)$ 最坏情况近似界。

虽然最坏情况近似界很有参考价值，但它们很难说明近似算法的典型 表现，也无法反映其在现实图上的平均性能。因此，在下一节中，我们将通过实证评估算法的运行时间以及基于度的移除策略的有效性。

6. 算法性能评估

我们使用 Python 实现了所有算法和巴士因子度量，并在一台配备 2.9 GHz 中央处理器和 32 GB 随机存取存储器的笔记本电脑上运行所有实验。我们在随机二部网络上评估算法的运行时间

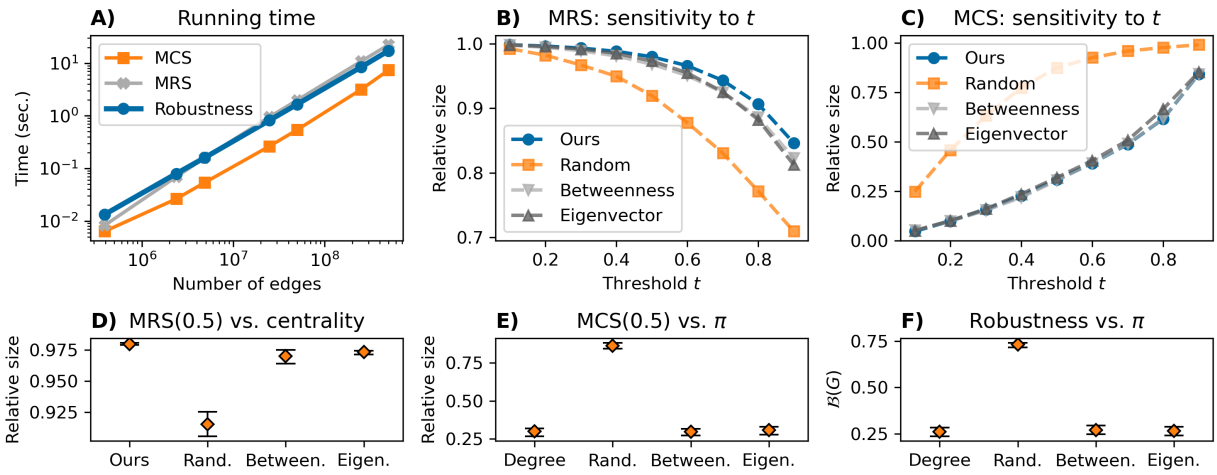


图 3：算法性能分析。**A)** 运行时间比较。**B)** MRS 对阈值 t 的敏感性。**C)** MCS 对阈值 t 的敏感性。**D)** 不同处理策略下 MRS 的平均相对规模。**E)** 不同移除顺序（π）下 MCS 的平均相对规模。**F)** 不同移除顺序（π）下的平均鲁棒性 $\mathcal{B}(G,\pi)$。**MRS**：最大冗余集；**MCS**：最小关键集；**鲁棒性**：作为二分网络鲁棒性的巴士因子。

其中两个节点集的大小均从 1 000 到 1 000 000 个节点不等，边的总数介于 400 000 到约 500 000 000 之间。尽管随机网络并非现实世界项目网络的良好模型 [2], 但它们非常适合用于测量运行时间，因为它们不具备可利用的结构。对于最大冗余集（下文简称 MRS）和最小关键集（下文简称 MCS），我们均设置阈值 $t = 0.9$，以迫使算法处理整个网络。我们发现，算法的运行时间随边数线性增长，即使在超大规模网络上也依然高效，处理 5 亿条边所需时间不到 20 秒（图 3**A**）。

接下来，为了在最坏情况分析之外评估近似解的质量，我们将自己的实现与其他处理策略和移除顺序进行比较。我们采用随机顺序作为基线，因为合理的近似算法应优于随机解。除随机基线外，我们还将结果与介数中心性和特征向量中心性所确定的移除顺序进行比较；这两种强基线广泛应用于网络鲁棒性研究 [27]。我们评估 MRS 和 MCS 解的相对规模：对于 MRS，解越大越好；对于 MCS，则是越小越好。对于鲁棒性，我们则评估根据各移除策略定义 π 后得到的曲线下面积 $\mathcal{B}(G,\pi)$：面积越小，解越好。

在这项比较中，我们生成二部幂律网络，因为这类网络能够再现真实项目网络的许多特性 [2]。具体而言，我们从幂律概率密度函数中抽样得到两个节点集的度分布，并使用二部配置模型生成图。幂律概率密度函数定义如下：

$$f(x,\alpha,a,c) = \frac{\alpha}{c}\left(\frac{x-a}{c}\right)^{\alpha-1}, \quad \text{with support } [a, a+c]. \tag{7}$$

我们抽样生成包含 7500 名人員和 10000 项任务的网络。人员的度根据式 7 和参数$\alpha = 0.2$、$a = 1$、$c = 100$进行抽样，而任务度数则根据参数 $\alpha = 0.2$、$a = 1$、$c = 70$进行抽样。

这组参数使我们能够再现真实项目网络的关键特征，即：*1)*任务通常多于人员；*2)* 度分布呈右偏；并且*3)*只有一小部分人员（整合者）与大量任务相连，而大多数人员是只负责少数任务的专业人员。这些特征已在不同类型的项目中得到观察，项目范围涵盖软件开发 [3]到工业设计 [2]。我们强调，这些具体参数并无特殊之处；我们使用了不同的参数值进行实验，得到的结果在定性上相似。

由于 MRS 和 MCS 取决于阈值 t，我们首先考察了它们的输出如何随 t 变化。我们观察到，随着 t 增大，MRS 的规模单调递减（图 3**B**），而 MCS 的规模则递增

单调递增t （图 3**C**）。对于t的所有取值，我们的实现始终优于所有基线方法。这种单调行为表明，从定性角度看，t 的选择不会显著影响观察到的趋势。然而，从定量角度来看，如果将 MRS 或 MCS 解释为巴士因子度量，那么t 的选择至关重要，必须仔细论证。在先前研究中，t 通常设为 0.5 [7]。因此，为与现有文献保持一致，我们在后续对 MRS 和 MCS 的分析中将$t = 0.5$。

最后，我们在 1 000 个二部幂律网络上，将我们的实现与基线方法进行比较，以评估其性能。对于 MRS（图 3**D**），我们的实现得出的解比随机基线得出的解大约大 6%。对于 MCS（图 3**E**）和鲁棒性（图 3**F**），我们的实现得出的解约为随机基线所得解的三分之一。此外，我们的实现至少与基于中介中心性和特征向量中心性的策略表现相当，同时计算速度更快。

现在，我们将评估 MRS、MCS 和鲁棒性作为巴士因子度量的适用性。

7. 公交因子度量的实验评估

如何在保持领域无关视角的同时评估巴士因子度量？Avelino等人 [6] 和 Ferreira等人 [7] 开展的实证评估在这方面存在问题：它们针对 GitHub 仓库量身定制，而且只考察了相对较少的项目，其中大多数项目的巴士因子都显而易见地为 1。

在这里，我们采用了不同的方法。我们提出对合成幂律二部图（与前文一样，根据公式 7 采样）进行敏感性分析，并施加受控扰动，以相对变化的形式评估巴士因子度量的响应。我们施加的扰动对应于特定的管理行动，这些行动会在贡献者之间重新分配工作负荷。

具体而言，我们探讨以下研究问题：

Q1 网络密度变化时，巴士因子会如何变化？

为回答这个问题，我们在网络中添加或移除边，以模拟人员承担的任务数量比原始网络中更多或更少的情形。

Q2 向网络中添加人员时，巴士因子会如何变化？

我们添加单任务人员 （即只负责一项任务的人员）和重复人员 （即现有贡献者的克隆），以模拟向项目中增加新人员，从而提高冗余度并进而增强鲁棒性的情形。

Q3 巴士因子如何 因人员重新分配而发生变化 到任务吗？

在此，我们利用度相关性与网络鲁棒性之间已知的关系 [28]。具体而言，我们在不添加或移除节点或边的情况下，增加或降低网络中的度相关性，从而引发鲁棒性的相应变化。

我们将在以下各节中分别探讨这些问题。

7.1. 不同工作负载分布下巴士因子度量的敏感性 *(Q1)*

向人员分配更多任务是组织内部增加人员知识的常见策略，从而使项目不易受到贡献者不可用性的影响 [2]。在我们的图论框架中，这对应于向初始二分网络添加边，使图的连通性更强，因而更加稳健。相反，移除边则模拟了人员平均承担较少任务的情况， resulting in a less robust project structure.

我们预期，设计良好的巴士因子度量与网络密度呈正相关：网络越稠密，项目的鲁棒性越强；网络越稀疏，鲁棒性就越弱。我们通过每次向初始网络添加或删除 100 条边来进行这组模拟，总共添加或删除 50 000 条边。

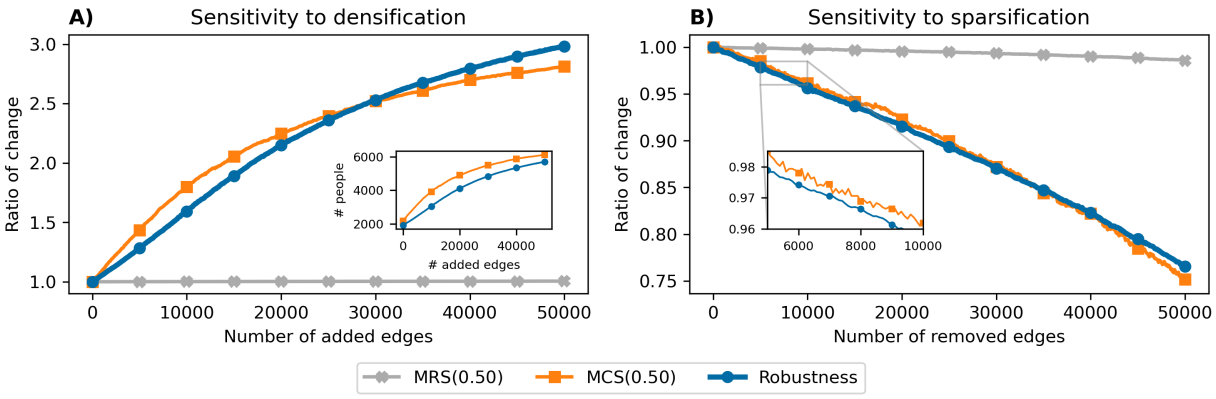


图 4：巴士因子度量对网络密度变化的敏感性（**Q1**）。**A)** 稠密化：与预期一致，MCS 和鲁棒性随网络密度增加而提高。插图：以人员数量表示的 MCS 和鲁棒性数值。**B)** 稀疏化：与预期一致，MCS 和鲁棒性随网络密度降低而下降。插图：放大显示移除 $[5\,000, 10\,000]$ 条边的范围，表明鲁棒性比 MCS 更稳定。**MRS**：最大冗余集；**MCS**：最小关键集；**Robustness**：作为二分网络鲁棒性的巴士因子。

我们发现，MRS 对网络密度的变化基本不敏感，而 MCS 和鲁棒性在定性上表现出预期的行为（图 **4A**）。然而，这项实验已凸显出 MCS 对固定阈值的依赖所带来的影响。随着网络变得更加稠密，MCS 比鲁棒性更早达到饱和（图 **4A**）：在最初添加的 25000 条边期间，MCS 的增长速度高于鲁棒性，随后便趋于平稳（见图 **4A**的插图）。这种行为是阈值参数 t 直接造成的。

类似地，随着网络变得更加稀疏，MCS 和鲁棒性以相近的速率下降（图 **4B**），但 MCS 会出现明显波动（见图 **4B**的插图）。这些波动既源于使用固定阈值，也源于网络稀疏化会改变节点度，进而改变移除顺序。相比之下，鲁棒性的表现更稳定，因为它基于连通分量和最大连通任务数，而非仅仅基于任务覆盖。

总的来说，这项实验已经证明，与基于覆盖度的替代方法相比，使用鲁棒性来估算项目的巴士因子具有优势。

7.2. 针对提高人员冗余策略的巴士因子度量敏感性（*Q* *2)*

现在我们转而讨论另一种常用于增强项目并提高其鲁棒性的策略：招聘。向项目中增加人员，可以有效提高人员冗余和项目鲁棒性，因为新成员不仅能覆盖任务，还可以充当整合者或锚点，改善项目的整体连通性，从而使底层网络中较稀疏的区域更加密集。

然而，招聘并非万灵药；如果实施不当，这一策略可能适得其反。首先，招聘流程本身复杂、受预算限制，而且往往缓慢、低效或无效 [29]。其次，向项目中增加人员可能会提高人员之间的协调成本 [30]。最后，增加整合者与增加专业人员（即只承担少数任务的人员）对项目鲁棒性的影响有显著差异：如前所述，整合者才是主要提升项目整体鲁棒性的人员 [2]。

在这里，我们研究巴士因子度量对两种旨在提高冗余度的策略的响应：

1. 添加单任务人员：我们在初始图中为每项任务添加一名单任务人员。单任务人员是只负责一项任务的专业人员。添加单任务人员虽然会增加任务层面的冗余，但不会提升项目层面的冗余，因为这无法整合网络中的远距离模块。2. 添加重复人员：我们从度最高的人员开始，按度从高到低，将现有人员的副本（重复节点）添加到初始图中。高连接度人员是整合者，低连接度人员则是专业人员。这种策略会在一定程度上提升冗余——具体来说，直到整合者被复制为止；此后则会呈现收益递减。

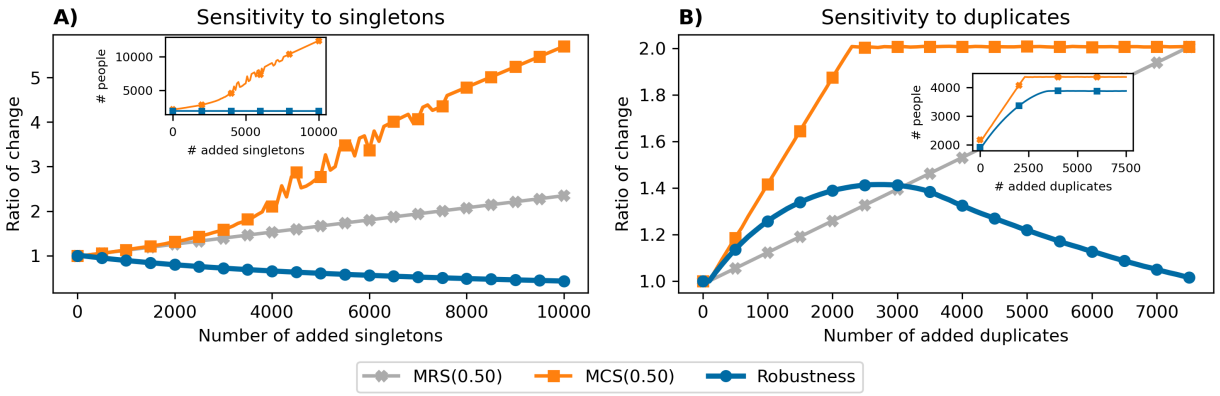


图 5：针对旨在提高人员冗余的策略，巴士因子度量的敏感性（**Q2**）。**A)** 对单任务人员（即只参与一项任务的人员）的敏感性。随着单任务人员加入网络，MCS 和 MRS 均会增加，且没有上限。这种行为不适合作为巴士因子度量。相比之下，鲁棒性能够正确反映增加单任务人员所带来的收益递减，并且会随着单任务人员加入网络而降低。插图中：MCS 和鲁棒性以人员数量表示。尽管归一化后的鲁棒性有所降低，反映了收益递减，但瓦解网络所需的人员数量仍保持不变，这符合预期。**B)** 对重复人员的敏感性。MRS 会无限增长，没有上限。随着重复人员加入，MCS 会不断增长，直至因其阈值而趋于饱和。符合预期的是，鲁棒性能够反映增加度逐渐降低的人员所带来的收益递减，其表现与关于整合者重要性的实证发现一致。**MRS**：最大冗余集；**MCS**：最小关键集；**鲁棒性**：作为二分网络鲁棒性的巴士因子。

这与项目管理 [21] 和软件工程 [31], 领域的实证发现一致：良好的巴士因子度量应对添加单任务人员相对不敏感，并应准确体现添加重复人员所带来的收益递减。

我们通过每批增加 100 名单任务人员或重复人员来模拟这两种策略，并在图 5 中报告结果。我们发现，MRS 和 MCS 对单任务人员的增加都过于敏感（图 5**A**）。两种度量都无限增长，这意味着仅通过增加单任务人员，就可以无限提高项目的巴士因子。这一结果与关于组织分化 [21], 协调忽视在组织中的表现 [32], 以及迭代工作动态 [33], 的实证证据严重矛盾，凸显了基于 MRS 和 MCS 的巴士因子度量存在重大局限性。

相反，随着单任务人员加入项目，鲁棒性会下降（图 5**A**）。这一下降是其对最大连通任务数及其归一化因子（公式 6）的依赖所直接导致的。由于鲁棒性反映了人员离开时项目的碎片化程度，单任务人员无法提高其数值。此外，由于鲁棒性经过归一化处理，与整合者可能作出的贡献相比，新增的单任务人员实际上被视为被浪费的资源。与此一致的是，在整个模拟过程中，破坏项目所需的人员数量保持稳定（图 5**A**的插图），进一步凸显了鲁棒性的灵活性及其双重可解释性。

MRS 对重复人员也相对不敏感，而 MCS 会趋于饱和，无法捕捉增加重复人员所带来的收益递减（图 5**B**）。再次，鲁棒性能够有效捕捉度逐渐降低的人员所带来的收益递减，其增长速度比 MCS 更缓慢，并最终在新增人员不再提供有意义的冗余时下降——这一细微差别是 MCS 无法体现的。

总的来说，这项实验表明，MRS 和 MCS 不适合作为巴士因子度量，因为它们高估了单任务人员的影响，并低估了整合者的重要性。相比之下，鲁棒性能够准确捕捉单任务人员和整合者的影响，其表现与理论预期和实证发现一致。

7.3. 巴士因子度量对不同人员任务分配的敏感性（*Q3*）

在不招聘新人员、也不改变既有工作量分布的情况下，将人员重新分配到任务上，是提高项目鲁棒性的另一种有效策略。Piccolo 等人 [11], 例如，设计了两种启发式方法，使一个生物质发电厂项目的巴士因子提高了40%。

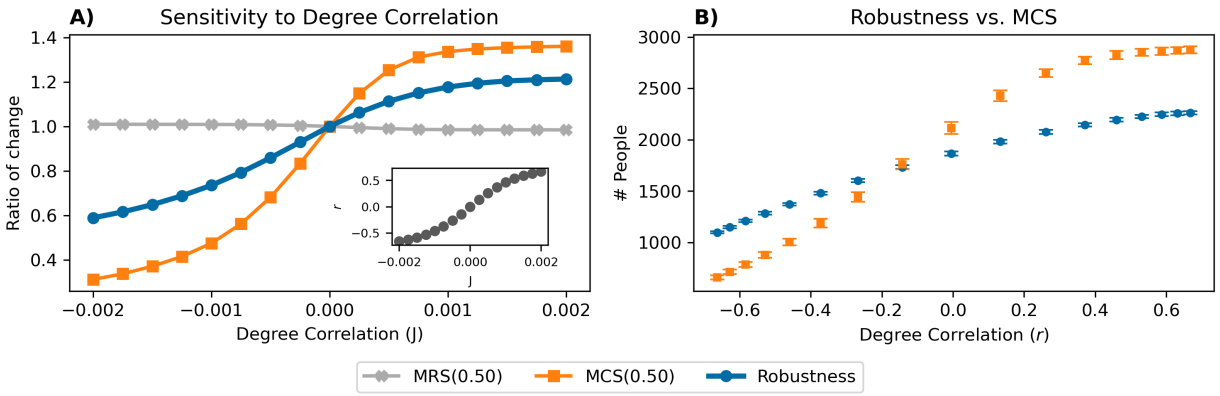


图6：巴士因子度量对度相关性变化的敏感性（**Q3**）。每个点表示对100个独立网络实现取平均值，总计1 700个网络。**A)** MRS 对度相关性不敏感，而 MCS 和鲁棒性则呈现预期的上升趋势。**B)** 以人员数量表示的平均巴士因子估计值及其置信区间：与 MCS 相比，鲁棒性与r 呈现更紧密的线性关系，且方差更低。**MRS**：最大冗余集；**MCS**：最小关键集；**Robustness**：作为二分网络鲁棒性的巴士因子。

在此，我们利用网络科学中一项公认结论，该结论将网络鲁棒性与度相关性（亦称度同配性）联系起来 [28]。度相关性衡量高度（低度）节点倾向于连接其他高度（低度）节点的程度。它被定义为一条边两端的度的皮尔逊相关系数：

$$r = \frac{\langle k_i k_j \rangle - \langle \frac{1}{2}(k_i + k_j) \rangle^2}{\langle \frac{1}{2}(k_i^2 + k_j^2) \rangle - \langle \frac{1}{2}(k_i + k_j) \rangle^2} \tag{8}$$

其中，若节点i 和j 由一条边相连，则k_i 和k_j 分别表示这两个节点的度，而 $\langle \cdot \rangle$ 表示对所有边取平均值。一般而言，较高的度相关性与更强的网络鲁棒性相关 [28], ，因为高阶节点往往彼此相连，形成致密且具有韧性的子网络。

在这组实验中，我们评估了度相关性水平各异的合成网络中巴士因子度量的响应，同时保持人员数量、任务数量及其度分布不变。

实现这一点需要一个非平凡的抽样过程。我们对统计系综 $\{G\}$ 按照概率测度$\mu(G) \propto e^{-H(G)}$进行抽样，该测度由哈密顿量$H(G) = -J\sum_{(i,j)\in E} k_i k_j$所诱导，其中$J$是控制所抽样网络的度相关性的参数。

该抽样过程包括对原始图G 应用梅特罗波利斯动力学，即反复重连两条随机选取的边(u,z)和(w, v)，从而生成一对新的边（网络中尚未存在）(u, v)和(w,z)。对于所得网络G'，我们计算能量差 $\Delta H = H(G') - H(G)$，并接受G'，其概率为$e^{-\Delta H}$。我们考察J 在区间 $[-0.002, 0.002]$, 内的17个等间距取值，每个取值生成100个独立网络，共计1 700个网络。

如图 6**A**插图所示，采样网络的平均度相关性随J单调增加。从图 6**A**可以看出，MRS 对人员向任务的分配变化仍基本不敏感，甚至表现出与预期相反的行为：随着度相关性增加而略微下降。相比之下，MCS 和鲁棒性的表现符合理论预期以及网络科学的先前研究结果 [28]。

不过，我们注意到，与鲁棒性相比，MCS 对度相关性r 造成了更大的扭曲（图 6**B**）。鲁棒性产生的曲线更接近线性，且总体方差低于 MCS 产生的曲线。

8. 讨论

至此，我们对巴士因子及其近似方法背后的理论已有了相当全面的理解，也详细比较了 MRS、MCS 和鲁棒性作为巴士因子度量的表现。我们的理论对网络科学和项目管理有何启示？我们的研究结果如何转化为可操作的建议？未来研究有哪些值得探索的方向？接下来，我们将回答这些问题。

8.1. 对网络科学和项目管理的启示

本文提出了一个与领域无关的图论框架，使我们能够将巴士因子的测量问题转化为二分图上的组合问题。我们定义了三个编码巴士因子不同定义的组合问题：最大冗余集（MRS）、最小关键集（MCS）以及作为网络鲁棒性的巴士因子（鲁棒性）。通过证明这三种表述都是 NP难问题，我们表明，计算项目的巴士因子不仅是与项目管理相关的风险度量，也涉及计算理论中的重要问题：如果我们能够在多项式时间内解决这三个问题中的任意一个，那么 P = NP。此外，我们对鲁棒性为 NP难问题的证明（定理 3）意味着，计算由 Schneider 等人 [15], 提出的网络鲁棒性度量 $\mathcal{R}$ 也是 NP难的，而这此前仍是一个悬而未决的问题 [9]。

我们的框架将网络科学、计算机科学与项目管理联系起来。正如本文所示，项目管理问题可以直接受益于网络科学和理论计算机科学所发展的算法与理论工具。例如，渗流理论 [26] 为按度递减顺序移除人员提供了理论依据，而近似算法工具则使我们能够正式分析启发式策略的质量与局限。就此而言，据我们所知，我们对最坏情况近似界的证明（定理4、5、6）首次为 MRS、MCS、鲁棒性和基于度的移除策略提供了近似结果。最后，项目管理领域的理论和实证发现为我们提供了依据，用于实验比较各种巴士因子度量。

我们的研究通过大量实验和理论证据，支持整合者对于提升项目鲁棒性的重要作用 [2], ，并强化了有关软件项目 [31], 和迭代开发动态[33]的先前研究结果。这些结果揭示了该研究与锚节点概念以及二分图和超图鲁棒性理论 [17]之间的紧密联系。此外，我们还发现项目鲁棒性与度相关性之间存在关联 [28], ，这意味着可以通过重新分配人员到任务来提升项目鲁棒性。最后，大量测试表明，鲁棒性是唯一始终表现符合预期的度量，而基于覆盖率的度量（MRS 和 MCS）存在致命缺陷，尤其是无法捕捉项目碎片化，因此不适合作为巴士因子度量。

简而言之，计算巴士因子的理论与实践可以自然地表述为计算二分网络鲁棒性的理论与实践（反之亦然）。

8.2. 将我们的研究发现付诸实践

巴士因子度量已经在实践中得到实施，并被证明有助于理解项目鲁棒性和指导纠正措施 [4]。因此，我们希望在此强调分析中的关键要点；我们认为，这些要点为估算和提升巴士因子提供了可付诸行动的见解。

1. 鲁棒性目前是唯一符合理论预期与实证发现的公交因子度量。此外，它经过归一化，不依赖阈值，并且能够刻画项目碎片化，这些特性使其易于在实践中使用。2. 若招聘旨在提升项目鲁棒性，最佳策略是招聘能够增加冗余并连接彼此相距较远的项目模块的整合者。3. 项目巴士因子可以通过将人员重新分配到各项任务来改善，而无需增加个人工作量或招聘新人员。如我们所示，这可以通过专用算法 [11], 实现，或通过提高项目二分网络的度相关性来实现。4. MCS不宜作为公交因子度量。不过，它可用于估计工作量分布的不均衡程度。例如，可以估计至少产出了代码库80%的较小人员集合，以检验该项目是否遵循80–20帕累托分布。

5. MRS作为公交因子度量不具信息量。然而，其补集——部分集合覆盖——可用于识别一小部分能够推动知识共享实践并指导新员工的主题专家和整合者，从而提高项目可持续性。这将巴士因子与在网络中寻找最具影响力节点的问题联系起来。

我们指出，在实践中使用巴士因子度量以及上述策略时，应始终结合具体案例加以考量，例如人员技能和任务相对重要性。

8.3. 未来研究方向

我们的研究为巴士因子的未来扩展、其理论基础及其近似铺平了道路。以下简要勾勒未来的发展方向:

1. 我们证明了 MRS、MCS 和鲁棒性的 NP难性，并证明 MRS 在$O(n^{1-\epsilon})$ 因子内的近似属于 NP难问题。MCS 和鲁棒性的可近似性仍是一个开放问题，其答案将使我们深入了解任何启发式方法所能达到的近似质量。2. 我们的巴士因子度量仅考虑结构因素，未考虑任务相对重要性或人员技能。将鲁棒性所提供的结构性见解与其他社会和技术因素 [33] 相结合，可能有助于更全面地理解项目鲁棒性。3. 巴士因子理论可以通过发展其渗流理论，以及发掘除度相关性之外与之相关的网络度量来扩展。4. 我们证明了基于度的移除在实践中效果良好，但很可能存在能够提供更准确估算的算法。这是一个很有前景的研究方向，有望带来更好的算法，并加深我们对巴士因子的理解。

我们希望，我们的工作能够激发网络科学、计算机科学与项目管理交叉领域的更多研究。

9. 结论

巴士因子是衡量项目抵御人员损失、人员更替和知识碎片化能力的一项指标。从本质上说，公交因子计算意味着要回答一个看似简单的问题：必须有多少人员离开项目，项目才会停滞?

这个问题表面上的简单性掩盖了巨大的概念和计算困难。定义项目何时停滞，建模人员离开后知识和职责如何碎片化，以及制定一种独立于特定领域或平台的度量方式，都是并非易事的挑战。因此，现有的公交因子计算方法大多是临时性的：它们针对特定项目，或专门面向 GitHub 仓库；依赖简单的基于覆盖的模型；并依靠任意阈值来定义失败。

在本文中，我们提出了一个领域无关的理论框架，将公交因子计算视为二分图上的一系列组合问题。该框架能够在最大冗余集（MRS）和最小关键集（MCS）这两种互补形式化方法下，对现有方法进行形式化和统一，并明确揭示它们的局限性。我们的分析表明，尽管基于覆盖的形式化方法具有直观吸引力，但它们未能捕捉项目鲁棒性的关键方面，因此不适合作为巴士因子度量。

在此框架的基础上，我们提出了一种基于网络鲁棒性的新型公交因子度量。与先前的定义不同，该度量经过归一化、无阈值，并将项目退化刻画为由碎片化驱动的连续过程，而非二元的失败概念。理论分析与广泛的实证评估均表明，这种基于鲁棒性的形式化符合预期行为，并持续优于现有度量。

从计算视角来看，我们证明，在本研究考察的所有表述下，计算巴士因子都是 NP难问题。这为精确计算确立了根本性限制，并促使我们采用近似算法。在此框架下，我们提出了近似策略，并证明了基于度的移除的最坏情况保证，为公交因子计算提供了首批正式的近似结果。

总而言之，本研究将巴士因子从非正式的启发式指标提升为一种有理有据、具有严谨理论基础的项目鲁棒性度量。通过将巴士因子重新表述为二分网络鲁棒性问题，我们提供了一个统一的理论基础，将项目管理、网络科学与计算复杂性联系起来，并使协作系统中的鲁棒性分析、比较和近似成为可能。

致谢

Sebastiano A. Piccolo 感谢 Marco Manna、Simona Perri 和 Aldo Ricioppo 提供的宝贵意见，以及围绕 NP难优化问题的 NP难性与可近似性展开的富有成效的讨论。本研究部分由 MUR 通过国家复苏与韧性计划项目 PE0000013-FAIR（专题9——绿色感知人工智能——WP9.2）以及 PN RIC 项目 ASVIN“智能商店虚拟助手”（CUP B29J24000200005）资助。

参考文献

[1] N. Zazworka、K. Stapel、E. Knauss、F. Shull、V. R. Basili、K. Schneider，开发人员是否遵循流程：一项极限编程研究，载于：2010年ACM-IEEE实证软件工程与度量国际研讨会论文集，ESEM ’10，计算机协会，纽约，NY，美国，2010年。doi:10.1145/1852786.1852805。URL https://doi.org/10.1145/1852786.1852805[2] S. A. Piccolo、S. Lehmann、A. Maier，设计过程鲁棒性：二分网络分析揭示了人员的核心重要性，《设计科学》4（2018年）e1。[3] K. Yamashita、S. McIntosh、Y. Kamei、A. E. Hassan、N. Ubayashi，重新审视帕累托原则对开源软件项目核心开发团队的适用性，载于：第14届软件演化原理国际研讨会论文集，IWPSE 2015，计算机协会，纽约，NY，美国，2015年，第46–55页。doi:10.1145/2804360.2804366。URL https://doi.org/10.1145/2804360.2804366 [4] E. Jabrayilzade、M. Evtikhiev、E. Tüzün、V. Kovalenko，实践中的巴士因子，载于：第44届国际软件工程会议论文集：软件工程实践，ICSE-SEIP ’22，计算机协会，纽约，NY，美国，2022年，第97–106页。doi:10.1145/3510457.3513082。URL https://doi.org/10.1145/3510457.3513082[5] F. Ricca、A. Marchetto、M. Torchiano，论计算卡车因子的困难，载于：第12届以产品为中心的软件过程改进国际会议论文集，PROFES’11，施普林格出版社，柏林，海德堡，2011年，第337–351页。[6] G. Avelino、L. Passos、A. Hora、M. T. Valente，一种估算卡车因子的新方法，载于：2016年IEEE第24届程序理解国际会议（ICPC），2016年，第1–10页。doi:10.1109/ICPC.2016. 7503718。[7] M. Ferreira、M. T. Valente、K. Ferreira，三种计算卡车因子算法的比较，载于：2017年IEEE/ACM第25届程序理解国际会议（ICPC），2017年，第207–217页。doi: 10.1109/ICPC.2017.35。[8] V. Cosentino、J. L. C. Izquierdo、J. Cabot，评估Git代码库的巴士因子，载于：2015年IEEE第22届软件分析、演化与再工程国际会议（SANER），2015年，第499–503页。doi:10.1109/SANER.2015.7081864。[9] T. Zhao、W. Si、W. Li、A. Y. Zomaya，面向最小化用于度量网络鲁棒性的r 指标，《IEEE网络科学与工程汇刊》8（4）（2021年）3290–3302。doi:10.1109/TNSE.2021. 3110195。[10] C. Hannebauer、V. Gruhn，卡车因子计算的算法复杂度，载于：以产品为中心的软件过程改进：第15届国际会议，PROFES 2014，芬兰赫尔辛基，2014年12月10日至12日。论文集15，Springer，2014年，第119–133页。[11] S. A. Piccolo、P. De Meo、G. Terracina，评估并改进项目的巴士因子：一种网络分析框架，载于：第16届国际会议，ASONAM 2024，意大利伦代，2024年9月2日至5日，论文集，第一部分，《计算机科学讲义》，Springer，2024年。

[12] R. Albert、H. Jeong、A.-L. Barabási，复杂网络的错误与攻击容忍性，《自然》406（6794）（2000年）378–382。[13] S. Agreste、S. Catanese、P. De Meo、E. Ferrara、G. Fiumara，黑手党犯罪集团的网络结构与韧性，Information Sciences 351（2016年）30–47。doi:https://doi.org/10.1016/j.ins.2016.02.027。URL https://www.sciencedirect.com/science/article/pii/S0020025516300925[14] D. S. Callaway、S. H. Strogatz、D. J. Watts，网络鲁棒性与脆弱性：随机图上的渗流，《物理评论快报》85（25）（2000年）5468。[15] C. M. Schneider、A. A. Moreira、J. S. Andrade Jr、S. Havlin、H. J. Herrmann，缓解对网络的恶意攻击，《美国国家科学院院刊》108（10）（2011年）3838–3841。[16] Y. Shang，度量具有二部结构的无标度网络中的度依赖失效，《国际仿真与过程建模杂志》8（1）（2013年）。[17] Y. Shang，带锚节点的随机超图中的渗流，《物理评论E》111（5）（2025年）054304。[18] M. Kivelä、A. Arenas、M. Barthelemy、J. P. Gleeson、Y. Moreno、M. A. Porter，多层网络，《复杂网络杂志》2（3）（2014年）203–271。[19] P. Holme、J. Saramäki，时序网络，《物理学报告》519（3）（2012年）97–125。[20] M. R. Garey、D. S. Johnson，《计算机与难解问题》，第174卷，freeman，旧金山，1979年。[21] P. R. Lawrence、J. W. Lorsch，组织与环境：管理差异化与整合，哈佛商业评论出版社，1986年。[22] T. H. Cormen、C. E. Leiserson、R. L. Rivest、C. Stein，算法导论，2022年。[23] L. A. Wolsey，次模集合覆盖问题贪心算法分析，《组合数学》2（4）（1982年）385–393。[24] C. L. Lim、A. Moffat、A. Wirth，集合覆盖问题的惰性与急切方法，载于：第37届澳大利亚计算机科学会议论文集，第147卷，2014年，第19–27页。[25] D. Zuckerman，线性度提取器以及最大团与色数的不可近似性，载于：第38届ACM计算理论研讨会年会论文集，2006年，第681–690页。[26] R. Albert、A.-L. Barabási，复杂网络的统计力学，《现代物理评论》74（1）（2002年）47。[27] O. Artime、M. Grassia、M. De Domenico、J. P. Gleeson、H. A. Makse、G. Mangioni、M. Perc、F. Radicchi，复杂网络的鲁棒性与韧性，《自然评论：物理学》6（2）（2024年）114–131。[28] M. E. Newman，网络中的混合模式，《物理评论E》67（2）（2003年）026126。[29] M. Behroozi、S. Shirolkar、T. Barik、C. Parnin，招聘调试：技术面试流程中的成功与失误，载于：ACM/IEEE第42届国际软件工程会议论文集：社会中的软件工程，ICSE-SEIS '20，计算机协会，纽约，NY，美国，2020年，第71–80页。doi:10.1145/3377815.3381372。URL https://doi.org/10.1145/3377815.3381372[30] F. P. Brooks Jr，《人月神话》（周年纪念版），Addison-Wesley Longman Publishing Co., Inc.，1995年。[31] S. Majumder、J. Chakraborty、A. Agrawal、T. Menzies，为何软件项目需要英雄（经验教训来自1100+项目），arXiv预印本arXiv:1904.09954（2019年）。[32] C. Heath、N. Staudenmayer，协调忽视：关于组织的民间理论如何使组织中的协调复杂化，《组织行为研究》22（2000年）153–191。[33] S. A. Piccolo、A. M. Maier、S. Lehmann、C. A. McMahon，作为社会和技术因素结果的迭代：来自大型设计项目的实证证据，《工程设计研究》30（2）（2019年）251–270。