arXiv:2603.11478v1 [stat.ME] 12 Mar 2026

部分信息下的图生成方法

Tong Sun管理学院，哈尔滨工业大学，中国黑龙江省哈尔滨市 150001，
24b910002@stu.hit.edu.cn

Jianshu Hao中国建设银行公司银行部，中国北京 100033，
haojianshu1.zh@ccb.com

Michael C.Fu罗伯特·H·史密斯商学院、系统研究所，马里兰大学，美国马里兰州大学公园市 20742，mfu@umd.edu

Guangxin Jiang*管理学院，哈尔滨工业大学，中国黑龙江省哈尔滨市 150001，
gxjiang@hit.edu.cn

我们研究具有给定度序列的二分图、有向图和无向图的生成问题。首先，我们提出一种二分图生成的序贯方法，并建立一个刻画每一步允许的连接数的充要区间条件，从而保证全局可行性。在此基础上，我们开发了适用于不同问题规模的二分图枚举与抽样算法。随后，我们通过引入额外的连接约束、可行性验证和对称连接步骤，将这些二分图算法扩展到有向图和无向图情形，同时保留相同的算法原理。最后，数值实验表明，所提出算法具有良好的性能，尤其是在现有方法计算上难以处理的大规模实例中表现出良好的可扩展性。

Key words: 图生成、度序列、枚举与抽样、系统性风险

1. 引言

在许多风险管理问题中，相互连接实体的网络结构是决定冲击如何传播，以及局部扰动是否会演变为系统性事件的关键因素。某些网络结构可能会放大传染，使小规模冲击在整个系统中广泛蔓延；而另一些结构则可能抑制传播，增强系统韧性（Acemoglu 等人 2012，Fu 等人 2025）。这些机制不仅在金融系统中至关重要——金融机构之间的关联可能会将财务困境传导至各个市场（Eisenberg 和 Noe 2001，Adrian 和 Brunnermeier 2016，Huang 等人 2024）——在供应链中也同样如此，生产瓶颈可能会使中断沿供应链向下游扩散（Barrot 和 Sauvagnat 2016，Carvalho

* 通讯作者

等人 2021，Birge 等人 2023）。在这两种情形下，网络结构都深刻影响着系统性风险，因此，准确刻画网络结构对于风险评估和制定有效的缓解策略至关重要。

然而，在实践中，网络的完整结构鲜少能够被观测到。通常只能获得部分信息，并据此推断可能的网络结构。本文聚焦于此类信息中最常见且在实践中最易处理的一种，即与每个节点相关联的连接数量，而具体的对应节点则无法观测到。这类度信息在许多情境中自然存在。在金融系统中，监管机构可能能够观测到每家投资公司持有的资产数量，以及持有每项资产的公司数量，但无法得知具体的所有权关系（Glasserman 和 de Larrea 2018）。在供应链中，可能会有关于企业供应商和客户数量等统计信息，但无法得知各个交易伙伴的身份（Mungo 等人 2024）。在社交网络中，基于调查的数据可能会提供个体拥有的社交联系数量，但无法得知具体联系对象的身份（Marsden2014）。这促使我们研究如何重建与此类部分信息相符的可信网络结构，从而在无法获得完整数据的情境下开展有意义的风险分析。

当可用的部分信息仅限于与每个节点相关联的连接数量时，网络重构问题可以抽象为具有给定度序列的图生成。度是与一个节点相关联的边的数量，度序列则是网络中所有节点的度的集合。目标是生成恰好满足给定度序列的图，并受限于这些图是简单图，即不含自环或重边。这一抽象直接刻画了上文讨论的例子：金融系统中的资产—公司关系对应于二分图，其中资产与公司构成两个不相交的节点集；供应链可以表示为有向图，边从供应商指向客户，用以描述投入产出流；社交网络通常被建模为无向图，其中边表示个体之间相互的社会联系。由此，生成与给定度序列一致的图，为分析不同应用领域中的网络重构提供了一个统一框架。

以往研究提出了多种方法来应对部分信息下的网络重构难题。根据网络类型，这些方法大致可分为二分图、有向图和无向图。对于二分图，Verhelst (2008) 开发了一种用于抽取具有固定边际的二元矩阵的马尔可夫链蒙特卡罗（MCMC）算法；而 Barvinok(2010) 建立了均匀抽样的二元矩阵与相应最大熵解之间的渐近联系。 Miller 和 Harrison(2013) 进一步提出了

一种用于二元列联表的精确采样算法，实现了可证明的均匀采样。对于有向图，Gandy 和 Veraart (2017) 开发了一个贝叶斯马尔可夫链蒙特卡罗框架，用于根据汇总资产负债表数据推断和抽样加权有向金融网络；而 Squar- tini 等人 (2017) 提出了一种基于最大熵的方法，根据部分信息重构加权有向网络，推断出与已知节点强度和抽样边密度相一致的最可能拓扑结构及边权重。对于无向图，Wormald(1999年) 提出了用于生成随机 d-正则无向图的切换方法，该方法既能保持度约束，又能在可行配置空间中进行均匀探索；此外，Fosdick等人 (2018年) 研究了用于生成具有给定度序列的简单图的马尔可夫链蒙特卡罗方法。最近，Glasserman 和 de Larrea (2023) 提出了一种使用最大熵的顺序算法，并将该方法用于重构二分图、有向图和无向图。

现有方法通常基于抽样，并且一般依赖 MCMC 程序或反复优化；随着问题规模增大，这些方法会面临计算可扩展性方面的挑战。在小规模场景中，高效的精确枚举方法可能更受青睐，因为它能够生成所有可行图，并支持对图统计量和系统性风险度量进行精确评估。随着问题规模扩大，可行图的数量可能呈指数级甚至超指数级增长，使得穷举在计算上不可行，因此需要可扩展的抽样算法。

本文针对不同问题规模下具有给定度序列的二分图、有向图和无向图，开发了枚举与抽样算法。我们的框架利用有向图和无向图均存在等价二分图表示这一事实，统一处理不同网络类型。因此，我们首先聚焦于二分图。具体而言，我们首先提出一种序贯方法，并根据 Gale–Ryser 定理（Gale1957），推导出每一步允许的连接数所需的充分必要区间条件，以确保图能够可行地完成。基于该方法，我们设计了一种精确枚举算法，能够系统地生成所有可行图且不重复，适用于小规模网络。

对于规模较大的问题，可行图的数量可能超过 10^{20}, ，此时穷举枚举不可行，因此需要采用抽样方法。在这些情况下，效率不仅意味着快速生成，还意味着近似均匀性，以确保样本不会过度集中在一小部分图上。在除度序列之外没有其他先验信息的情况下，对可行集合进行均匀采样是一项自然的准则。基于序贯方法，我们设计了两种抽样算法。第一种是均匀二部图抽样（UBGS）算法，它通过计算与每个可行值相关的不同补全方式数量，在每一步确定精确权重。尽管该方法实现了均匀

采样，但由于需要获取精确权重，它会带来额外的计算开销和较高的内存使用。为了解决这一问题，我们进一步提出了高效二分图采样（EBGS）算法，该算法以近似权重取代精确权重，从而在保持近似均匀性的同时提高可扩展性。

在为二分图开发了枚举与抽样算法后，我们将这些方法扩展到有向图和无向图。核心思路是构造二分图表示，并纳入额外的自环和单度连接约束，以及可行性验证和对称连接步骤，从而捕捉方向性或对称性，并使相同的算法原理能够应用于不同类型的图。对于有向图，我们将每个节点拆分为一个“入”副本和一个“出”副本，将问题转化为源节点与目标节点之间的二分匹配，并满足指定的入度和出度。对于无向图，我们采用对称二分图表示，其中可行性要求边在划分两侧互为镜像。在这些表示中，我们的方法同时提供枚举和抽样流程。对于小规模有向或无向网络，枚举算法会穷尽生成所有可行图且不重复，从而确保完整覆盖解空间。对于规模较大的此类网络实例，我们调整抽样算法，利用可行补全的精确或近似计数来指导节点连接。由此，我们的框架为二分图、有向图和无向图提供了并行的解决方案：小规模实例采用精确枚举，大规模实例则采用均匀采样或高效但略有偏差的采样。

与我们最为相关的工作是 Glasserman 和 de Larrea (2023)，该研究基于最大熵分布，提出了一种用于给定度序列的图重构的顺序算法。他们提出的近似均匀采样方法适用于二部图、有向图和无向图，但需要反复求解复杂的优化问题，因此计算成本会随问题规模迅速增长。另一项密切相关的工作是 Miller 和 Harrison (2013)，该研究针对元素为二元值或非负整数的二维列联表，开发了一种精确采样算法。他们的均匀采样方法既未涉及有向图或无向图，也无法扩展到大型实例。此外，这两项工作都没有考虑枚举算法。总之，我们的工作有以下贡献：

(i) 我们提出了一种用于二分图生成的序贯方法，并建立了一个充要区间条件，用于刻画每一步允许的连接数，从而保证全局可行性。(ii) 我们开发了一种无重复二分图枚举算法，可穷举生成所有可行图，适用于小规模问题；此外，我们还开发了二分图

采样算法，可用于大规模实例的均匀采样，或高效但略有偏差的采样。据我们所知，本研究首次开发了能够处理大规模问题的采样算法。

(iii) 通过纳入额外的自环和单度连接约束，以及可行性验证和对称连接步骤，我们将二分图枚举与抽样算法扩展到有向和无向情形，同时保留相同的算法原理，并确保其适用于不同规模的问题。

(iv) 我们通过数值实验展示了所提算法的性能，尤其是其在大规模实例上的可扩展性；对于这些实例，现有方法的计算成本已高得难以承受。

本文其余部分安排如下。第2 节定义三类图以及图空间生成问题。第3节介绍生成具有指定度序列的二分图的序贯方法，并确立该方法所需的充要区间条件；基于此，我们进一步提出二分图枚举与抽样算法。第4节将这些算法扩展到有向和无向情形。第5节通过数值实验评估所提算法。最后，第6节总结全文。

2. 问题表述

令 $\mathcal{G}(\mathbf{a},\ \mathbf{b})$ 表示图空间，其定义为所有具有给定度序列$\mathbf{a}$ 和$\mathbf{b}$的二分图、有向图或无向图的集合，其中$\mathbf{a}$和$\mathbf{b}$的含义取决于图的类型：

二分图。在此情况下，$\mathbf{a} \in \mathbb{N}_+^m$ 和$\mathbf{b} \in \mathbb{N}_+^n$ 分别是两个不相交顶点集中的节点的度序列。例如，考虑一个由 m金融资产和 n 投资公司组成的金融网络，其中每家公司持有可用资产的一个子集。此类网络可以建模为二分图，其中一个顶点对应资产，另一个顶点对应公司，每条边表示某家公司持有某项特定资产。在此图中，资产节点的度等于持有该资产的公司数量，而公司节点的度则对应于该公司拥有的不同资产数量。

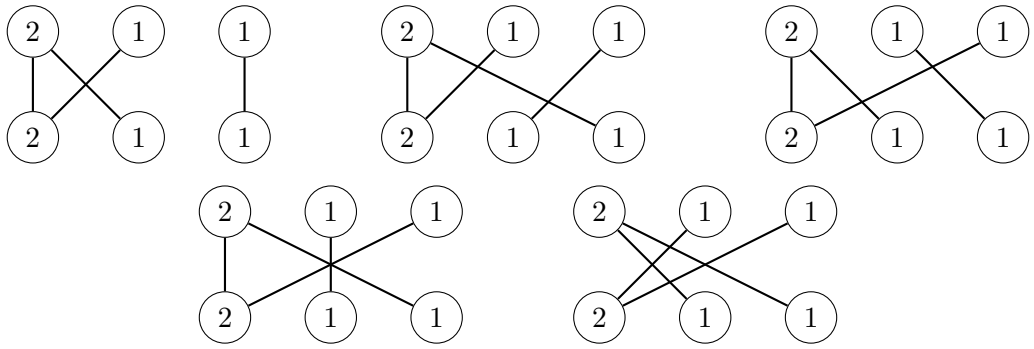
有向图。在这种情况下，$\mathbf{a} \in \mathbb{N}_0^m$和$\mathbf{b} \in \mathbb{N}_0^n$ 分别对应于节点的出度序列和入度序列，因此 $m = n$。例如，考虑一个由 m家企业组成的供应链网络，其中每家企业都可能作为供应商或客户，与其他企业的某个子集发生关系。这样的网络可以表示为一个有向图，其中每个节点对应一家企业，每条有向边从供应商指向客户。在这种表示中，企业节点的出度等于其供应的不同企业数量，而企业节点的入度对应于其采购来源的不同供应商数量。

无向图。在这种情况下，$\mathbf{a} = \mathbf{b} \in \mathbb{N}_+^m$ 是节点的度序列。例如，考虑一个由 m 个个体组成的社交网络，其中每个个体与其他个体的某个子集进行互动。这样的网络可以表示为一个无向图，其中每个节点对应一个个体，每条无向边表示两个个体之间的相互社交连接。在此，个体节点的度等于该个体在网络中维持的不同连接数量。

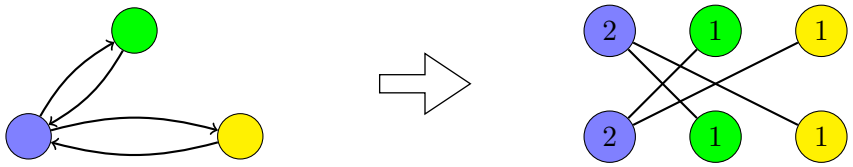
以下示例说明了一对度序列如何映射到相应的二分图、有向图和无向图空间。

EXAMPLE 1. 对于度序列 $\mathbf{a} = \mathbf{b} = (2,\ 1,\ 1)$，图 1 展示了与这些度序列一致的五个二分图、一个有向图和一个无向图。也就是说，二分图空间包含五个元素，而有向图和无向图空间各自仅包含一个元素。

(1) 五个二分图：



(2) 一个有向图：



（3）一个无向图：

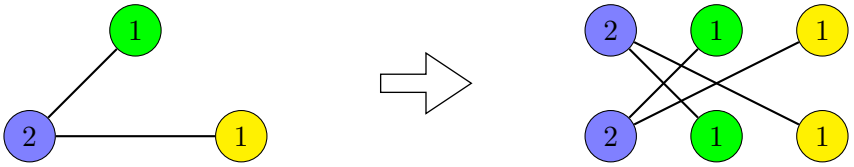


图1　五个二部 图、一个有向图和一个无向图，与 $\mathbf{a} = \mathbf{b} = (2,\ 1,\ 1)$一致。

图空间生成问题旨在生成 $\mathcal{G}(\mathbf{a},\ \mathbf{b})$ 的所有元素或一个相当大的子集，其中 $\mathcal{G}(\mathbf{a},\ \mathbf{b})$ 可表示二分图、有向图或无向图空间。本文提出了高效的类型特定算法：用于穷举生成整个图空间 $\mathcal{G}(\mathbf{a},\ \mathbf{b})$ 的枚举算法，适用于小规模问题；以及用于随机生成图实例的采样算法，可应用于更大规模的实例。

3. 二部图算法

我们首先考虑生成二分图的问题：这些图需满足给定的一对度序列 $\mathbf{a}$ 和 $\mathbf{b}$，其中 $\in \mathbb{N}_{+}^{m}$ and $\mathbf{b} \in \mathbb{N}_{+}^{n}$。不失一般性，我们假设度序列 $\mathbf{b}$ 按非递增顺序排列 $[\mathbf{b}]_1 \geq [\mathbf{b}]_2 \geq \cdots \geq [\mathbf{b}]_n$，其中 $[\mathbf{x}]_i$ 表示向量 $\mathbf{x}$ 的第 i 个元素。稍微滥用一下记号，$[\mathbf{b}]_i$ 也指节点本身。生成机制的关键步骤如下。首先，根据不同度值，将 $\mathbf{a}$ 中的节点划分为 p 组：第 k 组包含 m_k 个度为 α_k 的节点，其中 $\alpha_1 < \alpha_2 < \cdots < \alpha_p$and $\sum_{k=1}^{p} m_k = m$。接着，$\mathbf{b}$ 中的第一个节点（记为 $[\mathbf{b}]_{1,}$）依次从第 1 组到第 p 组，选择要与之连接的 $\mathbf{a}$ 中节点数。建立这些连接后，所有度降为零的节点都会被移除。更新后的度序列记为 $\mathbf{a}^{[2]} \in \mathbb{N}_{+}^{m^{[2]}}$ 和 $\mathbf{b}^{[2]} \in \mathbb{N}_{+}^{n-1}$，其中 $m^{[2]}$ 是 $\mathbf{a}$ 中剩余节点的数量（取决于被移除的节点数），而 $\mathbf{b}$ 的长度则减至 $n-1$。更一般地，我们用 $\mathbf{a}^{[j+1]}$ 和 $\mathbf{b}^{[j+1]}$ 表示处理完节点 $[\mathbf{b}]_j$后得到的更新度序列，其中 $\mathbf{a}^{[1]}$ =$\mathbf{a}$ 且 $\mathbf{b}^{[1]}$ =$\mathbf{b}$。随后反复执行这些分组和连接步骤，直至处理完 $\mathbf{b}$ 中的所有节点。由于每次迭代都可以视为使用更新后的给定度序列求解同一问题的新实例，因此只需关注与第一个节点 $[\mathbf{b}]_1$的连接过程。

回到例1，我们用图 2 来说明生成机制。首先，将 $\mathbf{a}$ 中的节点划分为两组（$p=2$）：第 1 组由两个度为 1 的节点（$\alpha_1 = 1$,$m_1 = 2$）组成，第 2 组由一个度为 2 的节点（$\alpha_2 = 2$, $m_2 = 1$）组成。接下来，度为 2 的节点 $[\mathbf{b}]_1$ 依次与 $\mathbf{a}$ 中的节点连接，从第 1 组开始，再到第 2 组。例如，节点 $[\mathbf{b}]_1$ 与第 1 组中的两个节点连接，使它们的度从 1 降至 0。完成这些连接后，移除 $[\mathbf{b}]_1$ 以及第 1 组中两个度为 0 的节点，得到更新后的度序列 $\mathbf{a}^{[2]} = (2)$ 和 $\mathbf{b}^{[2]} = (1,1)$。随后，使用更新后的度序列，为 $\mathbf{b}$ 中的下一个节点重复这一连接过程，并持续进行，直至所有节点都处理完毕。

这一生成机制的关键挑战在于确定每一步可以连接多少个节点，从而确保生成的二分图符合给定度序列。

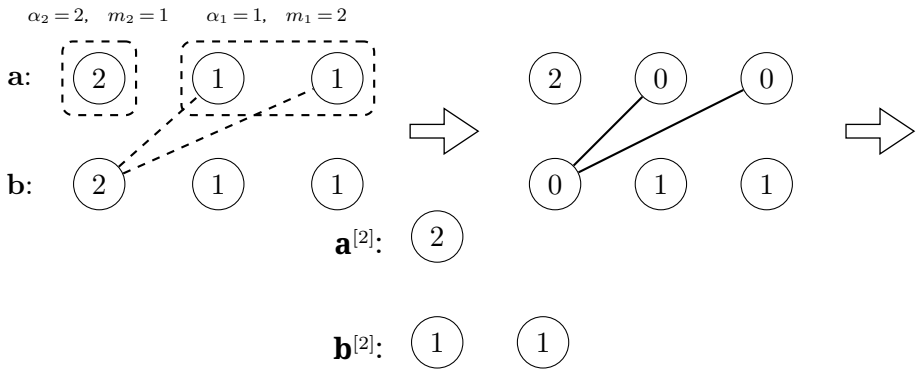


图 2 **示例 $[\mathbf{b}]_1$ 的连接过程，其中度序列为 1 为 $\mathbf{a}=\mathbf{b}=(2,\ 1,\ 1)$。**

3.1. 节点连接区间

本小节将给出一个定理，为每一步允许的连接数确立必要且充分的区间条件，在该条件下可保证全局可行性。首先，我们介绍表述该定理所需的几个基本数学定义。

DEFINITION 1. $\mathscr{Z}\colon(\mathbb{N}^p,\ \mathbb{N}_+)\to\{0,1\}^{p\times c}$ 是一种矩阵运算，它将向量 $\mathbf{x}\in\mathbb{N}^p$ 和正整数 $c\in\mathbb{N}_+$ 映射为一个大小为 $p\times c$ 的二元矩阵。该运算按元素定义如下：

$$[\mathscr{Z}(\mathbf{x},c)]_{ij}=\begin{cases}1, & \text{if } 1\le j\le[\mathbf{x}]_i,\\ 0, & \text{otherwise.}\end{cases}\tag{1}$$

请注意，所得二元矩阵 $\mathscr{Z}(\mathbf{x}c)$ 满足：对于每个 $i\in\{1,2,\ldots,p\}$，将行 i 中的前 $\mathbf{x}_i$ 列设为 1，其余元素设为 0。

Let $\mathbf{z}\in\mathbb{N}^{|\mathbf{b}|}$ 表示由矩阵 $\mathscr{Z}(\mathbf{a}\ |\mathbf{b}|)$ 导出的向量，定义如下：

$$[\mathbf{z}]_j=\sum_{i=1}^{m}[\mathscr{Z}(\mathbf{a},|\mathbf{b}|)]_{ij},\tag{2}$$

表示 $\mathscr{Z}(\mathbf{a}\ |\mathbf{b}|)$ 的第 j 列之和。

为了说明上述定义，我们重新考察例 1，其中 $\mathbf{a}=\mathbf{b}=(2,\ 1,\ 1)$。根据（1）和（2），我们得到：

$$\mathscr{Z}(\mathbf{a},|\mathbf{b}|)=\begin{bmatrix}1&1&0\\1&0&0\\1&0&0\end{bmatrix},$$

以及 $\mathbf{z}=(3,\ 1,\ 0)$。

利用上述定义，我们提出以下引理，该引理给出了盖尔-赖瑟定理（Gale 1957）的一种等价表述，用于刻画具有给定度序列的二分图存在的充要条件。证明见电子附录第 EC.1 节。

LEMMA 1. 对于给定度序列$\mathbf{a}$ 和$\mathbf{b}$，当且仅当$\sum_{i=1}^{m}[\mathbf{a}]_i=\sum_{j=1}^{n}[\mathbf{b}]_j$，且对所有 $r\in\{1,2,\ldots,n\}$ 均满足以下条件时，二分图存在：

$$\sum_{j=1}^{r}[\mathbf{z}]_j\geq\sum_{j=1}^{r}[\mathbf{b}]_j\,. \tag{3}$$

接下来，我们介绍与 $[\mathbf{b}]_1$ 相连的各组节点数的符号。令$\mathscr{F}(k)\in[0,m_k]\cap\mathbb{N}$(对于 $k=1$、2、.....、p) 表示组 k中与 $[\mathbf{b}]_1$ 相连的 m_k 个节点的数量。将组 $k-1$ 中的节点连接到 $[\mathbf{b}]_1$, 后，引入以下定义：

(i) 更新后的 a，记为 $\mathbf{a}^{(k)}$，其中 $\mathbf{a}^{(1)}$ =a；(ii) 根据 $\mathscr{L}\left(\mathbf{a}^{(k)}\right.$、$\left.|\mathbf{b}|\right)$ 得到的更新后的 z，记为 z $^{(k)}$，其中 $\mathbf{z}^{(1)}=\mathbf{z}$；(iii) $[\mathbf{b}]_1$, 的剩余度，记为 $[\mathbf{b}]_1^{(k)}$，其中 $[\mathbf{b}]_1^{(k)}=[\mathbf{b}]_1-\sum_{k'=1}^{k-1}\mathscr{F}(k')$ 且 $[\mathbf{b}]_1^{(1)}=[$ $\mathbf{b}]_1$；(iv) $\mathbf{a}^{(k)}$中组 $k+1$ 至 p的节点总数，记为 $M(k)$，其中 $M^{(k)}=\sum_{k'=k+1}^{p}m_{k'}$ 且 $M^{(p)}=0$。

为说明上述定义，我们重新考察例 1 中的 $\mathbf{a}=\mathbf{b}=(2,1,1)$。考虑这种情况$k=2$，并假设 $\mathscr{F}$ $(1)=1$，即，节点 $[\mathbf{b}]_1$ 与 a 的第 1 组中的一个节点相连。在这种情况下，更新后的 a 为 $\mathbf{a}^{(2)}=$ $(2,0,1)$；对应的矩阵为

$$\mathscr{L}\left(\mathbf{a}^{(2)},|\mathbf{b}|\right)=\begin{bmatrix}1&1&0\\0&0&0\\1&0&0\end{bmatrix},$$

由此得到 $\mathbf{z}^{(2)}=(2,1,0)$； $[\mathbf{b}]_1$的剩余度为 $[\mathbf{b}]_1^{(2)}=[\mathbf{b}]_1-\mathscr{F}(1)=2-1=1$；组 $k+1$至 p中的节点总数为 $M_{(2)}=0$，因为 $M^{(k)}=0$当 $k=p=2$。

REMARK 1. $\mathscr{F}(k)$ 表示 α_k 矩阵 $\mathscr{L}(\mathbf{a}$ $|\mathbf{b}|$ 中第 α_k 列中从 1 变为 0 的元素数，即 z 的第 $k=2$ 种情况 $\mathscr{F}(1)=1$，其中 1 例中的 $\mathbf{a}=\mathbf{b}=(2,1,1)$。这里，$\mathscr{F}(1)=1$ 表示对应矩阵第 $\alpha_1=1$ 列中的一个元素从 1 变为 0，即：

$$\mathscr{L}(\mathbf{a},|\mathbf{b}|)=\begin{bmatrix}1&1&0\\1&0&0\\1&0&0\end{bmatrix}\quad\longrightarrow\quad\mathscr{L}\left(\mathbf{a}^{(2)},|\mathbf{b}|\right)=\begin{bmatrix}1&1&0\\0&0&0\\1&0&0\end{bmatrix}.$$

因此，　　第 $\alpha_1=1$ 个分量的a从$\mathbf{z}=(3,\ 1,\ 0)$ 降至$\mathbf{z}^{(2)}=(2,\ 1,\ 0$　　).

我们接下来引入一种广义向量减法运算，用于定义定理中出现的若干量。

DEFINITION 2. $\mathscr{D}$: $(\mathbb{N}^p,\mathbb{N}^q)\to\mathbb{Z}^{\max\{p,q\}}$ 是一种向量运算，它将两个向量 $\mathbf{x}\in\mathbb{N}^p$ 和$\mathbf{y}\in\mathbb{N}^q$映射为另一个长度为 max $\{p,q\}$ 的向量。该运算按元素定义如下：

$$[\mathscr{D}(\mathbf{x},\mathbf{y})]_h=\begin{cases}[\mathbf{x}]_h-[\mathbf{y}]_h\,, & \text{if } h\leq\min\{p,q\}\,,\\ [\mathbf{x}]_h\,, & \text{if } h>\min\{p,q\}\text{ and }p>q,\\ -[\mathbf{y}]_h\,, & \text{otherwise.}\end{cases} \tag{4}$$

根据此定义，所得向量 $\mathscr{D}(\mathbf{x},\mathbf{y})$ 满足：对于每个 $h\in\{1,2,\ldots,\max\{p,q\}\}$，其值由 $[\mathbf{x}]_h-[\mathbf{y}]_h$ 给出；如果$\mathbf{x}$ 或$\mathbf{y}$ 的长度小于 $\max\{p,q\}$，则在其后补零。

通过结合$\mathbf{z}^{(k)}$、$\mathbf{b}^{[2]}$, 和（4），我们定义定理中出现的以下量：

$$D_{\text{left}}^{(k)}=\sum_{h=1}^{\alpha_k-1}\left[\mathscr{D}\left(\mathbf{z}^{(k)},\mathbf{b}^{[2]}\right)\right]_h,\quad D_{\text{now}}^{(k)}=\left[\mathscr{D}\left(\mathbf{z}^{(k)},\mathbf{b}^{[2]}\right)\right]_{\alpha_k},\tag{5}$$

and

$$D_{\text{right}}^{(k)}=\min\left\{\min_{r\in\{\alpha_k+1,\,\alpha_k+2,\,\ldots,\,n\}}\sum_{h=\alpha_k+1}^{r}\left[\mathscr{D}\left(\mathbf{z}^{(k)},\mathbf{b}^{[2]}\right)\right]_h,0\right\},\tag{6}$$

其中 $\mathscr{D}\left(\mathbf{z}^{(k)}、\mathbf{b}^{[2]}\right)\in\mathbb{Z}^n$，并回顾一下，$\alpha_k$是组 k在$\mathbf{a}^{(k)}$中的度。请注意，由矩阵 $\mathscr{Z}\left(\mathbf{a}^{(k)}、|\mathbf{b}|\right)$ 导出的向量$\mathbf{z}^{(k)}$与向量$\mathbf{b}^{[2]}$ 不处于相同状态。尽管如此，为了证明该定理，我们对这两个向量都应用 $\mathscr{D}$运算。

本文采用以下求和算子的记号约定。当其上限小于下限时，求和算子 $\sum$ 定义为零。此外，索引超出求和范围的任何项均视为零。

基于前述定义和引理1，我们给出以下定理，证明见电子附录 EC.2 节。

THEOREM 1. 假设存在一个二分图，其度序列为 $\mathbf{a}$ 和 $\mathbf{b}$。当且仅当对于 $\mathbf{a}$ 中的每个组 k，连接节点的数量 $\mathscr{F}(k)$ 满足以下条件时，才存在一个具有更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 的二分图：

$$\mathscr{F}(k)\in\left[\max\left\{[\mathbf{b}]_1^{(k)}-M^{(k)},0\right\},\min\left\{m_k,D_{\text{left}}^{(k)}+D_{\text{now}}^{(k)}+D_{\text{right}}^{(k)}\right\}\right].\tag{7}$$

定理 1 刻画了从 a 的组中与 b 相连的 m_k 个节点的可行节点数所满足的充要区间条件。具体而言，对于给定度序列，为了生成二分图，连接节点数 $\mathscr{F}(k)$ 必须满足式 (7) 给出的区间条件。下界中的第一项 $[\mathbf{b}]_1^{(k)}-M^{(k)}$ 描述了这样一种情况：b 的剩余度超过 a 的第 $k+1$ 至 p 组中的节点总数，此时，b 必须至少与第 k 组中的 $[\mathbf{b}]_1^{(k)}-M^{(k)}$ 个节点相连，才能满足其剩余度。加入零可确保 $\mathscr{F}(k)$ 非负。上界中的第一项 m_k 反映了一个显而易见的限制，即与第 k 组相连的节点数不能超过 m_k。加入 $D^{(k)}$ left $+D_{\text{now}}^{(k)}+D^{(k)}$ right 可确保更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 满足引理 1 中的式 (3)，从而在确定 b [的连接后，仍然适用于二分图生成。

3.2. 二分图枚举算法

基于定理 1，我们设计了一种双层广度优先搜索算法，用于枚举所有满足给定度序列的二分图。其基本思想是：针对区间 $(\mathscr{F}(k)$ 中的每个可行值，遍历与之对应的所有连接配置，从而系统地探索整个状态空间。连接配置是指从 $\mathscr{F}(k)$ 个节点中选取一种不同方式，并从 m_k 组的节点中选择 k 个节点与 $[\mathbf{b}]_j$ 相连。该算法通过两个嵌套的广度优先搜索过程遍历状态空间：外层搜索遍历 $\mathbf{b}$ 的节点，内层搜索则探索与每个节点关联的度数组。此构造确保所有可行二分图均被枚举，且不会重复。

在介绍伪代码之前，我们先简要回顾队列数据结构及其相关操作 入队 和 出队，这些操作是广度优先搜索过程的基础。队列遵循先进先出原则： 入队 操作将元素插入队尾，而 出队 操作则移除并返回队首元素。这些操作使算法能够按照中间图状态生成的确切顺序，对所有中间图状态进行系统探索。基于队列数据结构，二分图枚举（BGE）算法见算法 1。

算法 1 二分图枚举（BGE）算法

输入： 度序列 $\mathbf{a} \in \mathbb{N}_+^m$ 和 $\mathbf{b} \in \mathbb{N}_+^n$
输出： 所有与度序列 $\mathbf{a}$ 和 $\mathbf{b}$一致的二分图集合 $\mathcal{G}(\mathbf{a},\ \mathbf{b})$

初始化 $\mathcal{G}(\mathbf{a},\ \mathbf{b}) \leftarrow \emptyset$，一个空二分图 $\mathbf{G}(\mathbf{a},\ \mathbf{b})$，两个空队列 $\mathcal{Q}_{\text{outer}}$ 和 $\mathcal{Q}_{\text{inner}}$
将对（$\mathbf{G}(\mathbf{a},\mathbf{b}),1$）入队到 $\mathcal{Q}_{\text{outer}}$ ▷ 节点索引 $j=1$
当 $\mathcal{Q}_{\text{outer}} \neq \emptyset$ **时** ▷ 对 n 节点进行外层广度优先搜索
  从 $\mathcal{Q}_{\text{outer}}$ 中将第一对（$\mathbf{G}(\mathbf{a},\mathbf{b})$ j）出队，并记录节点索引 j
  若 $j > n$ **则**
    将 $\mathbf{G}(\mathbf{a},\ \mathbf{b})$ 添加到 $\mathcal{G}(\mathbf{a},\ \mathbf{b})$
    继续
  将 $\mathbf{a}^{[j]}$ 分成 p 组（$\alpha_k,\ m_k$），使用 $\alpha_1 < \cdots < \alpha_p$
  将对 $(\mathbf{G}(\mathbf{a},\ \mathbf{b}),\ 1)$ 入队到 $\mathcal{Q}_{\text{inner}}$ ▷ 组索引 $k=1$
  当 $\mathcal{Q}_{\text{inner}} \neq \emptyset$ **时** ▷对每个节点在 p 组上进行内层广度优先搜索
    从 $\mathcal{Q}_{\text{inner}}$ 出队第一个对（$\mathbf{G}(\mathbf{a},\mathbf{b})k$），并记录组索引 k
    如果 $k > p$ **则**
      将对（$\mathbf{G}(\mathbf{a},\mathbf{b})$ $j+1$）入队到 $\mathcal{Q}_{\text{outer}}$
      继续
    计算区间条件 $[\ell_k, u_k]$ 通过（7）
    对于 $\mathscr{F}(k) = \ell_k, \ldots, u_k$ **执行**
      生成所有 $\binom{m_k}{\mathscr{F}(k)}$ 连接配置
      对于 每个连接配置**执行**
        复制 $\mathbf{G}(\mathbf{a},\mathbf{b})$，并将其记为 $\mathbf{G}'(\mathbf{a},\mathbf{b})$
        将连接配置追加到 $\mathbf{G}'(\mathbf{a},\mathbf{b})$
        将对 $(\mathbf{G}'(\mathbf{a},\mathbf{b})$ $k+1)$ 入队到 $\mathcal{Q}_{\text{inner}}$
返回 $\mathcal{G}(\mathbf{a},\mathbf{b})$

为说明 BGE 算法的运行机制，我们重新考察例1，其中 b = (2, 1, 1)。图 3 展示了第一个节点 $[\mathbf{b}]_1$，的枚举过程，并重点呈现外层队列与内层队列之间的交互。算法首先将带有节点索引 $j = 1$ 的空二分图 G(a, b) 入队至空的外层队列 $\mathcal{Q}_{\text{outer}}$，如图 3 中面板 1 至面板 2 所示。当状态 $\big(\text{G(a, b)}\ j = 1\big)$ 从外层队列出队时，算法开始处理第一个节点 $[\mathbf{b}]_1$。此时，度序列 $\mathbf{a}^{[1]}$ 被划分为两组 ($p = 2$)：第 1 组由两个度为 1 的节点组成 ($\alpha_1 = 1,\ m_1 = 2$)，第 2 组由一个度为 2 的节点组成 ($\alpha_2 = 2,\ m_2 = 1$)。随后，算法将带有组索引 $k = 1$ 的空二分图 G(a, b) 入队至空的内层队列 $\mathcal{Q}_{\text{inner}}$，表示接下来应处理第 1 组；见图 3 中面板 2 至面板 3。当状态 $\big(\text{G(a, b)}\ k = 1\big)$ 从内层队列出队时，算法开始处理第 1 组。根据定理 1，第 1 组的区间条件为 $\mathscr{F}(1) \in [1, 2]$。对于每个可行整数 $\mathscr{F}(1)$，算法通过从第 1 组的两个节点中选取 $\mathscr{F}(1)$ 个节点，生成所有 $\binom{2}{\mathscr{F}(1)}$ 个连接配置。例如，当 $\mathscr{F}(1) = 1$ 时，会生成两个不同的连接配置。对于每个连接配置，将其附加到当前部分图 G(a, b) 的副本上，从而得到一个新的部分图 G′(a, b)。随后，将得到的状态 $\big(\text{G}'\text{(a, b)}\ k = 2\big)$ 入队至内层队列 $\mathcal{Q}_{\text{inner}}$，表示接下来应处理节点 $[\mathbf{b}]_1$ 的第 2 组；见图 3 中面板 3 至面板 4。内层广度优先搜索通过反复从内层队列 $\mathcal{Q}_{\text{inner}}$ 出队状态并处理第 2 组来继续进行，如图 3 中面板 4 至面板 5 所示。由于本例中 $p = 2$，处理完第 2 组后，节点 $[\mathbf{b}]_1$ 的连接步骤即告完成。因此，每个生成的部分图都与下一个节点索引关联，并作为状态 $\big(\text{G}'\text{(a, b)}\ j = 2\big)$ 入队至外层队列 $\mathcal{Q}_{\text{outer}}$；见图 3 中面板 5 至面板 6。随后，从外层队列 $\mathcal{Q}_{\text{outer}}$ 中将每个状态 $\big(\text{G}'\text{(a, b)}\ j = 2\big)$ 出队，并将其入队至内层队列 $\mathcal{Q}_{\text{inner}}$，表示算法继续处理第二个节点 $[\mathbf{b}]_2$；见图 3 中面板 6 至面板 7。

同样的枚举过程随后对 $j = 2$ 和 $j = 3$ 重复进行。对于每个节点 $[\mathbf{b}]_j$，更新后的度序列 $\mathbf{a}^{[j]}$ 首先被划分为度数组，每个组的区间条件利用（7）依据定理 1 重新计算，并通过内层广度优先搜索探索所有可行的组级连接配置。一旦$\mathbf{b}$ 中的所有节点均已处理完毕，所得图 $\mathbf{G}$（$\mathbf{a}$，$\mathbf{b}$）即被加入集合 $\mathcal{G}$（$\mathbf{a}$，$\mathbf{b}$）。通过这种方式，BGE 算法系统地枚举所有与 $\mathbf{a}$ 和$\mathbf{b}$ 一致且无重复的二分图。

3.3. 二分图抽样算法

枚举算法适用于小规模情形，但对于规模更大的度序列，二分图的数量很容易达到数亿，使得穷举

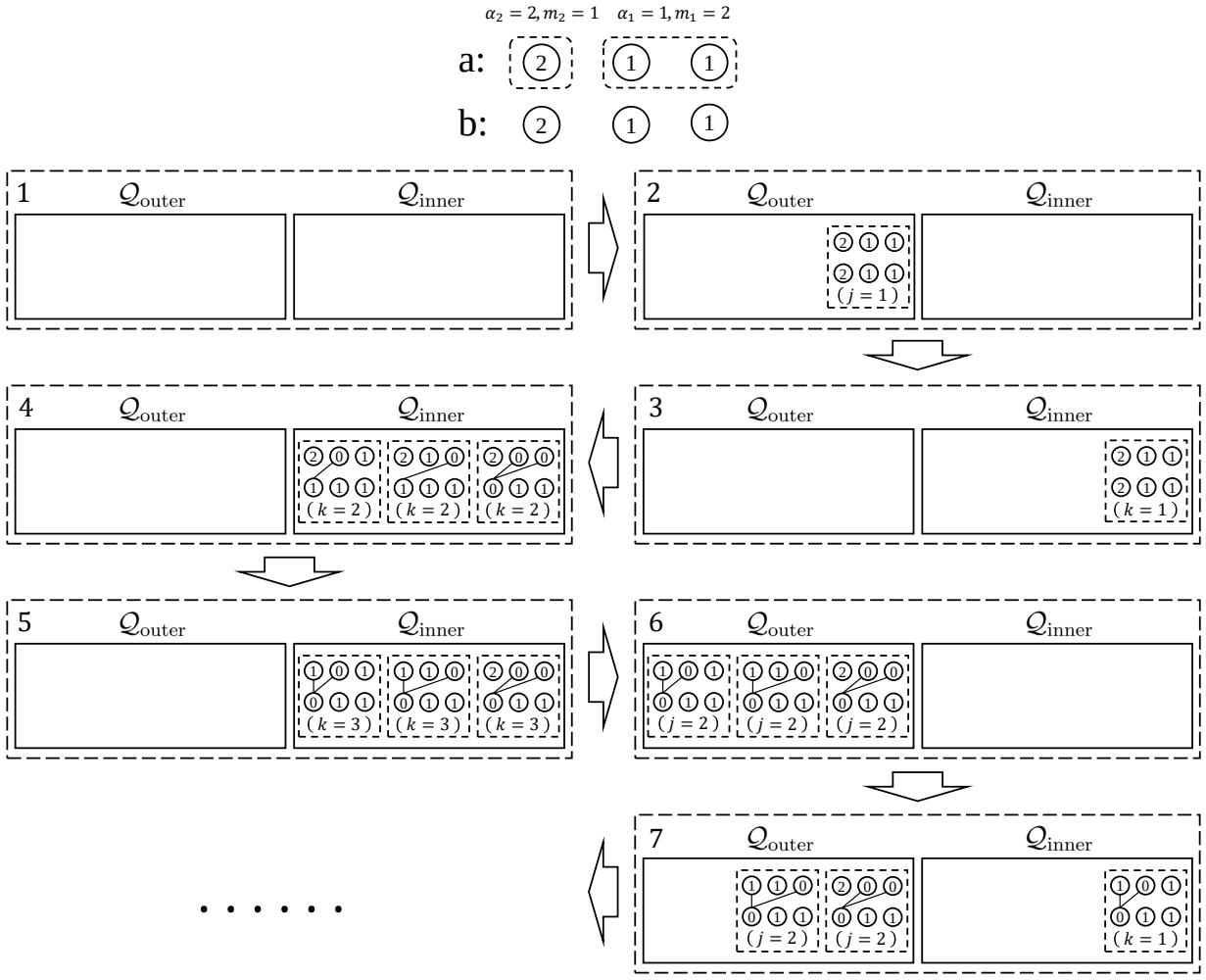


图 3 **节点 $[\mathbf{b}]_1$ 在示例 1 中，度序列为 $\mathbf{a} = \mathbf{b} = (2, 1, 1)$的枚举过程** .

枚举变得不切实际。例如，满足度序列 $\mathbf{a} = \mathbf{b} = (6, 6, 6, 5, 5, 5, 5, 4, 4, 3, 2)$ 的二分图超过 10^{20} 个。因此，采样算法提供了一种更易处理的替代方案。

如第1节所述，在缺乏额外先验信息的情况下，采样算法的目标是确保抽样图在可行二分图集合上均匀分布。这是一项自然的评价标准，因为均匀性使算法能够无偏地探索整个解空间，确保每个图被生成的可能性相同。Glasserman and de LarreaGlasserman 和 de Larrea（2023）也采用了均匀性标准，他们假设使用均匀采样，以确保抽样过程不会无意中偏向任何特定的图结构。因此，在本文中，我们同样将采样均匀性作为评估采样算法性能的标准之一。

与必须无重复地识别所有可行二分图的 BGE 算法相比，采样算法在逻辑上更为简单。给定由

(7) 计算出的区间条件，在该区间内随机选取 $\mathscr{F}(k)$ 的一个值，然后随机选择一种可行的连接配置。例如，考虑例1，其中 $\mathbf{a}=\mathbf{b}=(2,1,1)$。当将 $[\mathbf{b}]_1$ 连接到第 1 组（即 $k=1$）时，该组包含两个度为一的节点，由 (7) 计算得到的区间条件为 $\mathscr{F}(1)\in[1,2]$。因此，$[\mathbf{b}]_1$ 可以连接到该组中的一个或两个节点。在采样算法中，我们随机选择 $\mathscr{F}(1)=1$ 或 $\mathscr{F}(1)=2$。此外，当$\mathscr{F}(1)=1$时，还会产生一个额外的选择，即具体连接到哪个节点。这通过随机选择第 1 组中的两个节点之一来处理。

然而，一个自然的问题是：连接节点的数量和具体的节点是否都应当均匀选择？回到示例 1，图 4 展示了所有五个符合给定度序列的二分图。当节点 $\mathscr{F}(1)=1$ 的 $[\mathbf{b}]_1$,时，存在四个可行图；而当 $\mathscr{F}(1)=2$时，则只有一个可行图。因此，在 $\mathscr{F}(1)=1$和 $\mathscr{F}(1)=2$之间均匀选择，会导致可行二分图集合上的分布不均匀。为确保对这五个可行图进行均匀抽样，应以 0.8 的概率将 $\mathscr{F}(1)$ 设为 1，以 0.2 的概率设为 2。也就是说，选择概率应与此步骤中每种连接节点数量的选择所能生成的可行图数量成正比。因此，抽样算法的关键挑战在于，如何为区间内的可行值分配适当权重，从而实现对图的均匀抽样。

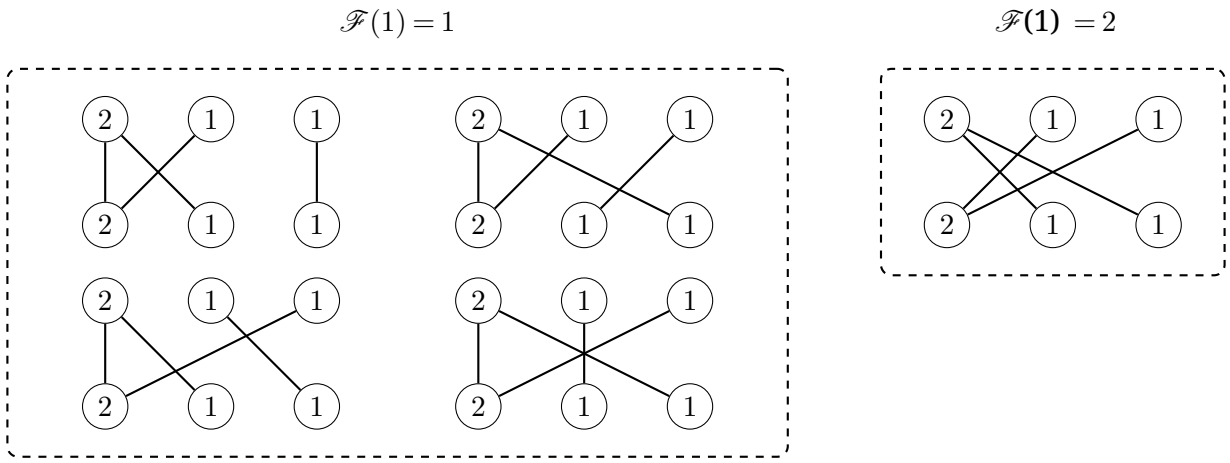


图4 例1 中度序列为$\mathbf{a}=\mathbf{b}=(2,1,1)$的所有二分图。

基于定理1，我们提出两种基于权重的抽样算法，从图空间 $\mathcal{G}(\mathbf{a},\mathbf{b})$ 中生成可行二分图。第一种算法——均匀二分图采样（UBGS）——通过为每种部分构造分配与其可完成的二分图数量成正比的权重，实现均匀采样。不过，该算法需要额外计算，并且在每一步评估这些数量时会占用相对较多的内存。

为解决这一问题，我们提出第二种算法——高效二分图采样（EBGS）。该算法以近似权重替代精确计数，从而降低计算和内存成本，同时保持近似均匀采样。

3.3.1. 均匀二分图采样算法。在此算法中，对于每一对度序列 $(\mathbf{a}^{[j]}, \mathbf{b}^{[j]})$（其中 $j = 1, \ldots, n$，且对于每个 $k = 1, \ldots, p$），我们需要确定与每个可行的 $\mathscr{F}_i(k)$ 相关联的不同二分图的数量，其中 $\mathscr{F}_i(k)$ 表示定理 1 给出的区间 $[\ell_k, u_k]$ 内，$\mathscr{F}(k)$ 的第 i 个可行值。具体而言，令 $N_B\big(\mathbf{a}^{[j]}, \mathbf{b}^{[j]}, \mathscr{F}(k)\big)$ 表示可由 $\mathscr{F}(k)$ 生成的不同二分图数量，已知 $[\mathbf{b}]_j$ 已完成第 1 组至第 $k-1$ 组的连接；也就是说，$\mathscr{F}(1), \ldots, \mathscr{F}(k-1)$ 已确定。类似地，定义 $N_B\big(\mathbf{a}^{[j]}, \mathbf{b}^{[j]}, \mathscr{F}_i(k)\big)$ 为可由 $\mathscr{F}(k)$ 的第 i 个可行值 $\mathscr{F}_i(k)$ 生成的不同二分图数量。随后，我们给出以下命题，用以刻画 $N_B\big(\mathbf{a}^{[j]}, \mathbf{b}^{[j]}, \mathscr{F}(k)\big)$ 与 $N_B\big(\mathbf{a}^{[j]}, \mathbf{b}^{[j]}, \mathscr{F}_i(k)\big)$ 之间的递归关系。证明见电子附录的 EC.3 节。

PROPOSITION 1. 对于每一对度序列（$\mathbf{a}^{[j]}$、$\mathbf{b}^{[j]}$）（其中 $j = 1$、……、n），以及每个 $k = 1$、……、p，从 $\mathscr{F}(k)$ 及其第 i 个可行值 $\mathscr{F}_i(k)$ 生成的不同二分图数量分别为

$$N_B\big(\mathbf{a}^{[j]}, \mathbf{b}^{[j]}, \mathscr{F}(k)\big) = \sum_{i=1}^{u_k - \ell_k + 1} N_B\big(\mathbf{a}^{[j]}, \mathbf{b}^{[j]}, \mathscr{F}_i(k)\big) \tag{8}$$

and

$$N_B\big(\mathbf{a}^{[j]}, \mathbf{b}^{[j]}, \mathscr{F}_i(k)\big) = \begin{cases} \binom{m_k}{\mathscr{F}_i(k)} N_B\big(\mathbf{a}^{[j]}, \mathbf{b}^{[j]}, \mathscr{F}(k+1)\big), & \text{if } k < p, \\ \binom{m_k}{\mathscr{F}_i(k)} N_B\big(\mathbf{a}^{[j+1]}, \mathbf{b}^{[j+1]}, \mathscr{F}(1)\big), & \text{otherwise.} \end{cases} \tag{9}$$

通过递归应用（8）和（9），并结合初始条件 N_B（$\mathbf{0}$、$\mathbf{0}$、0）$= 1$，我们可以计算所有的量 $N_B\big(\mathbf{a}^{[j]}$、$\mathbf{b}^{[j]}$、$\mathscr{F}_i(k)\big)$，即与每个可行 $\mathscr{F}_i(k)$ 相关联的不同二分图数量。因此，可以得到用于抽样 $\mathscr{F}(k)$ 的精确权重向量 $\mathbf{w}^{\mathrm{UB}}$，其中每个条目为

$$\big[\mathbf{w}^{\mathrm{UB}}\big]_i = N_B\big(\mathbf{a}^{[j]}, \mathbf{b}^{[j]}, \mathscr{F}_i(k)\big). \tag{10}$$

接下来，我们介绍 UBGS 算法，该算法在算法 2 中进行了描述，构建于序贯方法、定理 1 和均匀权重公式（10）之上。具体而言，该算法以度序列 $\mathbf{a}$ 和 $\mathbf{b}$ 以及一个初始为空的二分图 $\mathbf{G}(\mathbf{a},\mathbf{b})$ 开始。对于每个节点 $[\mathbf{b}]_j$，将 $\mathbf{a}^{[j]}$ 中的节点按照其不同度值划分为组，其中组 k 包含 m_k 个度为 α_k 的节点。对于每个组，依据定理 $\mathscr{F}(k)$ 获得

定理 1，并使用（10）计算精确权重向量 $\mathbf{w}^{\mathrm{UB}}$ 。根据 $\mathbf{w}^{\mathrm{UB}}$ 对 $\mathscr{F}(k)$ 进行采样后，从组 k 中均匀选择 $\mathscr{F}(k)$ 个节点，与 $[\mathbf{b}]_j$ 连接。处理完 $[\mathbf{b}]_j$ 后，通过将连接节点的度减一，并移除度变为零的节点，更新序列 $\mathbf{a}^{[j]}$ 和 $\mathbf{b}^{[j]}$ 。对 $\mathbf{b}$ 中的所有节点重复此过程，算法生成一个满足给定度序列的二分图 $\mathbf{G}(\mathbf{a},\mathbf{b})$。

算法 2 均匀二部图抽样（UBGS）算法

输入： 度序列 $\mathbf{a} \in \mathbb{N}_+^m$ 和 $\mathbf{b} \in \mathbb{N}_+^n$，一个初始为空的二分图 $\mathbf{G}(\mathbf{a},\mathbf{b})$
输出： 一个二分图 $\mathbf{G}(\mathbf{a},\mathbf{b})$

```
对于 j = 1 到 n 执行
    将 a^[j] 分为 p 组 (α_k, m_k)，其中 α_1 < α_2 < ··· < α_p
    对于 k = 1 到 p 执行
        通过（[ℓ_k, u_k] 7）计算区间条件
        通过（10）计算精确权重向量 w^UB
        根据 w^UB 对 F(k) ∈ [ℓ_k, u_k] 进行采样
        从组 k 中均匀选取 F(k) 个不同的节点，并将它们连接到 b]_j，位于 G(a, b) 中
返回 G(a, b)
```

请注意，在 UBGS 算法的每个采样步骤中计算 $N_B\big(\mathbf{a}^{[j]}、\mathbf{b}^{[j]}、\mathscr{F}_i(k)\big)$ 的计算开销很大，因为每次考虑可行取值 $\mathscr{F}_i(k)$ 时，都必须重新计算这些值。避免这种重复开销的一种自然方法是预先计算 $N_B\big(\mathbf{a}^{[j]}、\mathbf{b}^{[j]}、\mathscr{F}_i(k)\big)$ 的所有可行取值，并将其存储在可供多次采样运行重复使用的查找表中。查找表构建完成后，便无需进行即时计算，从而显著加快 UBGS 算法。

3.3.2. 高效二分图采样算法。 构建查找表本身需要额外的计算，而存储查找表会带来较高的内存使用，这两方面的开销都会随着问题规模的增大而快速增长，尽管查找表构建完成后可以重复使用。为减轻这些计算和内存负担，我们引入近似权重替代精确权重，从而完全避免构建查找表，并同时降低时间和内存开销，同时仍保持近似均匀采样。

如上所述，分配给节点 $[\mathbf{b}]_j$ 的可行值 $\mathscr{F}_i(k)$ 的权重应与其可生成的不同二部图数量成正比；该数量由以下三个过程决定：

1. 从组 k 中的 m_k 个节点里选取 $\mathscr{F}_i(k)$ 个节点构成$\mathbf{a}$。此步骤生成的不同二分图数量记为 N_1，其值为

$$N_1 = \binom{m_k}{\mathscr{F}_i(k)}.$$

2. 分别从$\mathbf{a}$的组 $k+1,\ldots,p$ 中选取 $\mathscr{F}$（$k+1$）$\ldots,\mathscr{F}$（p）个节点。此处生成的不同二分图数量记为 N_2，我们将其近似为

$$\widehat{N}_2 = \binom{M^{(k)} + m_k - \mathscr{F}_i(k)}{[\mathbf{b}]_j^{(k)} - \mathscr{F}_i(k)}.$$

3. 使用更新后的度序列 $\mathbf{a}^{[j+1]}$ 和 $\mathbf{b}^{[j+1]}$，在处理 $[\mathbf{b}]_j$后生成二分图。为提高计算效率，我们假设对于 $\mathscr{F}(k)$ 的不同可行取值，此步骤生成的不同二分图数量保持不变。

因此，与 $\mathscr{F}_i(k)$ 相关联的不同二分图近似数量，记为 $\widehat{N}_B\big(\mathbf{a}^{[j]},\ \mathbf{b}^{[j]},\ \mathscr{F}_i(k)\big)$，为 $N_1 \times \widehat{N}_2$。由此得到用于高效抽样的近似权重向量 $\mathbf{w}^{\mathrm{EB}}$，其各项定义如下

$$\left[\mathbf{w}^{\mathrm{EB}}\right]_i = \widehat{N}_B\big(\mathbf{a}^{[j]}, \mathbf{b}^{[j]}, \mathscr{F}_i(k)\big) = \binom{m_k}{\mathscr{F}_i(k)} \cdot \binom{M^{(k)} + m_k - \mathscr{F}_i(k)}{[\mathbf{b}]_j^{(k)} - \mathscr{F}_i(k)}. \tag{11}$$

将算法$\mathbf{w}^{\mathrm{UB}}$ 替换为$\mathbf{w}^{\mathrm{EB}}$，即可得到 EBGS 算法，具体做法是在算法2的步骤 5 中进行替换。EBGS 算法的伪代码见电子附录 EC.4 节中的算法A.3。请注意，EBGS 算法生成的样本在二分图空间 $\mathcal{G}(\mathbf{a},\ \mathbf{b})$ 上并非完全均匀。按照 Glasserman 和 de Larrea (2023) 的思路，当目标是计算关于 $\mathcal{G}(\mathbf{a},\ \mathbf{b})$ 上均匀分布的期望时，可以采用重要性抽样来校正这种非均匀性，详情见电子附录EC.5节。

4. 扩展算法

在本节中，我们通过引入额外的自环和单度连接约束，以及可行性验证和对称连接步骤，将二部图算法扩展到有向图和无向图。对于每种图类型，我们开发了一种枚举算法，在小规模设置下穷举生成相应图空间 $\mathcal{G}\mathbf{G}(\mathbf{a},\mathbf{b})$ 内的所有可行图实例 $(\mathbf{a},\mathbf{b})$；对于大规模问题，则开发了两种采样算法，分别在同一图空间上执行均匀采样，以及高效但略有偏差的采样。有向图和无向图的算法分别在第 4.1 节和 4.2节中介绍。

4.1. 有向图算法

在本小节中，我们首先介绍自环和单度连接约束，以及使二分图算法能够扩展到有向情形的可行性验证步骤。随后，我们以二分图算法为基础，并纳入这些额外要素，为有向图开发相应的枚举与抽样算法。

有向图中，向量 $\mathbf{a} \in \mathbb{N}_0^m$和 $\mathbf{b} \in \mathbb{N}_0^n$ 分别表示节点的出度序列和入度序列。忽略 $\mathbf{a}$ 和 $\mathbf{b}$ 中度为零的节点后，该有向图可视为一个不含自环的二分图。因此，为了将二分图算法扩展到有向情形，我们引入额外的自环和单度连接约束，以及可行性验证步骤，以共同排除自环并确保生成有向图。

为在生成有向图时排除自环，必须确保每个 $[\mathbf{b}]_j$都不与其在 $\mathbf{a}^{[j]}$中的对应节点相连。若直接施加此约束，则需要通过回溯过程丢弃违反自环排除约束的配置，这在计算上效率低下；详情请参见电子附录第 EC.6.1 节。为避免这种低效做法，我们采用一种间接但在结构上更便于处理的变换。具体而言，我们将 $\mathbf{a}$ 和 $\mathbf{b}$ 中每个节点的度都增加 $\mathbf{1}$，包括度为零的节点，从而得到增广度序列。然后，我们要求每个节点 $[\mathbf{b}]_j$ 与其对应节点构成一个自环，为方便起见，将该对应节点记为 $[\mathbf{a}]_j$，并令 k_j 表示包含该节点的组的索引。生成一个与增广度序列一致的二分图后，移除所有自环，即可得到一个满足原始度序列 $\mathbf{a}$ 和 $\mathbf{b}$且不含自环的有向图。在生成增广二分图的过程中，要求每个节点与其对应节点构成自环，这一要求称为自环连接约束。

除了自环连接约束外，我们还施加单度连接约束，以处理一种特殊情况。当每个节点 $[\mathbf{b}]_j$ 连接到$\mathbf{a}^{[j]}$中第 $\mathbf{1}$ 组度为一的节点时，它无法连接到那些唯一剩余度必须保留用于形成自环的节点。

基于自环和单度连接约束，我们建立以下定理，用以更新定理 1 中针对二分图的区间条件，从而适用于有向图情形中使用的增广度序列。证明见电子附录的第 EC.6.2节。

THEOREM 2. 假设存在一个具有给定度序列 $\mathbf{a}$ 和 $\mathbf{b}$的有向图。为在有向图情形下满足自环和单度连接约束，针对增广度序列，将二分图定理 *1* 中的区间更新如下：

(i) 对于 $k < k_j$:

$$\mathscr{F}(k) \in \begin{cases} \left[\max\left\{[\mathbf{b}]_j^{(k)} - M^{(k)}, 0\right\},\ \min\left\{D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}, m_k - n_{\text{one}}, u_\Delta^{(k)}\right\}\right], & \text{if } k = 1, \\ \left[\max\left\{[\mathbf{b}]_j^{(k)} - M^{(k)}, 0\right\},\ \min\left\{D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}, m_k, u_\Delta^{(k)}\right\}\right], & \text{if } k > 1 \end{cases} \tag{12}$$

(ii) 对于 $k = k_j$:

$$\mathscr{F}(k) \in \begin{cases} \left[\max\left\{[\mathbf{b}]_j^{(k)} - M^{(k)}, 1\right\},\ \min\left\{D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}, m_k - n_{\text{one}} + 1\right\}\right], & \text{if } k = 1, \\ \left[\max\left\{[\mathbf{b}]_j^{(k)} - M^{(k)}, 1\right\},\ \min\left\{D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}, m_k\right\}\right], & \text{if } k > 1 \end{cases} \tag{13}$$

(iii) 对于 $k > k_j$:

$$\mathscr{F}(k) \in \Big[\max\big\{[\mathbf{b}]_j^{(k)} - M^{(k)}, 0\big\},\ \min\big\{D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}, m_k\big\}\Big] \tag{14}$$

其中， n_{one} 表示组1 中尚未与其对应节点形成自环的度为一的节点数量； $u_\Delta^{(k)} = D^{(1,k_j)_{\text{左}}} + D^{(1,k_j)_{\text{现在}}} + D^{(1,k_j)_{\text{右}}} - \sum_{k'=1}^{k-1} \mathscr{F}(k') - 1$，其中

$$D_{\text{left}}^{(1,k_j)} = \sum_{h=1}^{\alpha_{k_j}-1} \big[\mathscr{D}\big(\mathbf{z}^{(1)}, \mathbf{b}^{[j+1]}\big)\big]_h, \quad D_{\text{now}}^{(1,k_j)} = \big[\mathscr{D}\big(\mathbf{z}^{(1)}, \mathbf{b}^{[j+1]}\big)\big]_{\alpha_{k_j}},$$

and

$$D_{\text{right}}^{(1,k_j)} = \min\left\{\min_{r\in\{\alpha_{k_j}+1,\,\alpha_{k_j}+2,\,n-j+1\}} \sum_{h=\alpha_{k_j}+1}^{r} \big[\mathscr{D}\big(\mathbf{z}^{(1)}, \mathbf{b}^{[j+1]}\big)\big]_h, 0\right\}.$$

然而，尽管自环和单度连接约束大幅限制了$\mathscr{F}(k)$的可选项，但它们不足以保证生成有向图。具体而言，某些$\mathscr{F}(k)$与 $[\mathbf{b}]_j$ 的组合可能满足这些局部约束，却会导致更新后的度序列$\mathbf{a}^{[j+1]}$ 和$\mathbf{b}^{[j+1]}$ 无法用于生成有向图。为排除这类情况，我们进一步采用基于有向图存在充要条件的可行性验证步骤（例如，参见Berger (2014)中的定理 2）。具体而言，在此验证中，我们暂时将度恢复到原始尺度：将增广度序列中尚未形成自环的节点的度减一，然后检查所得度序列是否能够构成有向图。这一额外的可行性验证步骤确保 $[\mathbf{b}]_j$ 的连接配置在有向图生成方面全局可行。

总之，为了将二分图算法扩展到有向情形，我们引入以下两项连接约束和一个可行性验证步骤：

- 自环连接约束。每个节点 $[\mathbf{b}]_j$ 都必须在$\mathbf{a}^{[j]}$ 中，其度数加一后与其对应节点 $[\mathbf{a}]_j$ 形成自环；这等价于 $[\mathbf{b}]_j$ 在原始度序列下不形成自环。

- 单度连接约束。每个节点 $[\mathbf{b}]_j$ 都不能连接到尚未形成自环的度为一的节点，因为这些节点唯一的剩余度必须留给它们自身形成自环。

- 可行性验证步骤。每个 $[\mathbf{b}]_j$ 完成其连接后，使用有向图存在性的必要且充分条件验证更新后的度序列$\mathbf{a}^{[j+1]}$ 和$\mathbf{b}^{[j+1]}$，以确保 $[\mathbf{b}]_j$ 的连接配置可行。

4.1.1. 有向图枚举算法 回顾一下，BGE 算法的基本思想是遍历定理 1中（7）所指定区间内每个可行值对应的所有连接配置。具体而言，BGE 算法采用两层广度优先搜索：外层搜索遍历$\mathbf{b}$中的节点，内层搜索则探索与每个节点关联的度数组。为将 BGE 算法扩展到有向情形，我们将自环、单度连接约束以及可行性验证步骤纳入内层。具体来说，有向图枚举（DGE）算法由算法 1 经以下修改得到：(i) 在算法 **1**的初始化阶段，将每个节点的度增加一，以扩充$\mathbf{a}$ 和 b；(ii) 根据定理 2中的（12）–（14），更新步骤 15 中的区间条件；(iii) 在步骤 17 中，将每个节点 $[\mathbf{b}]_j$ 连接到 k_j 组中的对应节点，以形成相应的自环；(iv) 对每个节点 $[\mathbf{b}]_j$完成连接过程后，插入可行性验证步骤；(v) 将每个有向图添加到输出集合前，移除所有自环。DGE 算法的伪代码见电子附录 EC.6.3 节中的算法A.4。

4.1.2. 有向图采样算法为了将二分图采样算法扩展到有向情形，我们从 UBGS 算法入手。回顾一下，UBGS 算法的关键在于为区间内每个可行值分配精确权重，其中每个权重都应与选择该值后能够生成的不同二分图的数量成正比。为了将 UBGS 算法扩展到有向情形，我们首先将自环和单度连接约束以及可行性验证步骤纳入有向图精确权重的计算中。由于单度连接约束和可行性验证步骤取决于具体连接的是哪些节点，而不只是连接的数量，因此必须针对每种连接配置分配权重，而不是针对可行值分配权重。因此，分配给每种配置的精确权重应与根据该配置能够生成的不同有向图的数量成正比。

具体而言，令 $N_D\big(\mathrm{a}^{[j]}、\mathrm{b}^{[j]}、\mathscr{F}(k)\big)$ 表示由 $\mathscr{F}(k)$ 生成的不同有向图的数量。定义 $N_D\big(\mathrm{a}^{[j]}、\mathrm{b}^{[j]}、\mathscr{F}_{il}(k)\big)$ 为由与 i 个可行值$\mathscr{F}_i(k)$（对应于 $\mathscr{F}(k)$ 的第 l 个连接配置）生成的不同有向图的数量。随后，我们建立以下命题，用于刻画 $N_D\big(\mathrm{a}^{[j]}、\mathrm{b}^{[j]}、\mathscr{F}(k)\big)$ 与 $N_D\big(\mathrm{a}^{[j]}、\mathrm{b}^{[j]}、\mathscr{F}_{il}(k)\big)$之间的递归关系。证明见电子附录的 EC.6.4 节。

PROPOSITION 2. 对于每一对度序列 $(\mathbf{a}^{[j]}、\mathbf{b}^{[j]})$，其中 $j=1、..、n$，以及每个 $k=1、...、p$，由 $\mathscr{F}(k)$ 生成的不同有向图数量，以及由与第 i 个可行值 $\mathscr{F}_i(k)$ 相关联的第 l 个连接配置生成的不同有向图数量，分别为

$$N_D\left(\mathbf{a}^{[j]},\mathbf{b}^{[j]},\mathscr{F}(k)\right)=\sum_{i=1}^{u_k-\ell_k+1}\sum_{l=1}^{\binom{m_k}{\mathscr{F}_i(k)}}N_D\left(\mathbf{a}^{[j]},\mathbf{b}^{[j]},\mathscr{F}_{il}(k)\right) \tag{15}$$

and

$$N_D\left(\mathbf{a}^{[j]},\mathbf{b}^{[j]},\mathscr{F}_{il}(k)\right)=\begin{cases}N_D\left(\mathbf{a}^{[j]},\mathbf{b}^{[j]},\mathscr{F}(k+1)\right), & \text{if } k<p,\\ N_D\left(\mathbf{a}^{[j+1]},\mathbf{b}^{[j+1]},\mathscr{F}(1)\right), & \text{otherwise.}\end{cases} \tag{16}$$

为纳入自环和单度连接约束，以及可行性验证步骤，根据 (15)–(14)，更新了 (ℓ_k 下界和上界 u_k 中的15)：12)–(14)，并适用以下情形：

$$\begin{cases}N_D\left(\mathbf{a}^{[j]},\mathbf{b}^{[j]},\mathscr{F}_{il}(k_j)\right)=0, & \text{if } [\mathbf{b}]_j \text{ does not connect its counterpart node,}\\ N_D\left(\mathbf{a}^{[j+1]},\mathbf{b}^{[j+1]},\mathscr{F}(1)\right)=0, & \text{if the feasibility verification for } (\mathbf{a}^{[j+1]},\mathbf{b}^{[j+1]}) \text{ fails.}\end{cases} \tag{17}$$

通过递归应用带更新后界限的 (15)，结合 (16) 和 (17)，并使用初始条件 $N_D(\mathbf{0},\ \mathbf{0},\ 0)=1$，我们可以计算出所有量 $N_D\left(\mathbf{a}^{[j]}、\mathbf{b}^{[j]}、\mathscr{F}_{il}(k)\right)$。因此，可以得到用于对每种连接配置进行抽样的精确权重向量 $\mathbf{w}^{\mathrm{UD}}$，其中每个分量定义如下：

$$\left[\mathbf{w}^{\mathrm{UD}}\right]_{il}=N_D\left(\mathbf{a}^{[j]},\mathbf{b}^{[j]},\mathscr{F}_{il}(k)\right). \tag{18}$$

基于精确权重向量 $\mathbf{w}^{\mathrm{UD}}$ 和 UBGS 算法，我们提出均匀有向图采样（UDGS）算法。具体而言，UDGS 算法通过对算法 2进行以下修改得到：(i) 在算法2的初始化阶段，将每个节点的度增加一，以扩充 $\mathbf{a}$ 和 $\mathbf{b}$；(ii) 根据 (12)–(14) 更新步骤 4 中的区间条件；(iii) 将步骤 5–7 中对可行取值的抽样替换为对连接配置的抽样；(iv) 将步骤 5 中使用的权重替换为 (18) 给出的权重；(v) 在返回最终有向图之前移除所有自环。UDGS 算法的伪代码见电子附录 EC.6.5 节中的算法 A.5 。

接下来，我们通过加入自环和单度连接约束以及可行性验证步骤，将 EBGS 算法扩展到有向图情形。具体而言，高效有向图采样算法（EDGS）由算法 A.3经以下修改得到：(i) 在算法 **A.3**的初始化阶段，将每个节点的度加一，从而扩增 **a**和 b；(ii) 根据（12）–（14）更新第 4 步中的区间条件；(iii) 在第 7 步中，将每个节点 $[\mathbf{b}]_j$ 与组 k_j 中的对应节点相连，以形成相应的自环；(iv) 完成每个节点的

连接过程后，$[\mathbf{b}]_j$，并且 (v) 在返回最终有向图之前移除所有自环。EDGS 算法的伪代码见电子附录第 EC.6.6节中的算法A.6。请注意，与 EBGS 算法类似，在根据 $\mathcal{G}$（$\mathbf{a}$，$\mathbf{b}$）上的均匀分布计算期望时，也可以对 EDGS 算法应用重要性抽样，以校正其非均匀性。

4.2. 无向图算法

在本小节中，我们首先介绍对称连接步骤，以便将有向图算法扩展到无向情形。随后，我们将该步骤纳入有向图算法，进而为无向图构建相应的枚举与抽样算法。

无向图中，向量 $\mathbf{a}=\mathbf{b}\in\mathbb{N}_+^m$ 表示节点的度序列；该无向图可以视为一个出度与入度度序列相同的有向图。为了将有向图算法调整到无向情形，我们引入对称连接步骤，对所有 $j=1,\ldots\ldots,n$强制执行 $\mathbf{a}^{[j]}=\mathbf{b}^{[j]}$。具体而言，在每个节点 $[\mathbf{b}]_j$完成其连接后，其对应节点 $[\mathbf{a}]_j$ 会同时建立一组对称连接，从而镜像复制 $[\mathbf{b}]_j$ 的每条边，并确保 $\mathbf{a}^{[j+1]}=\mathbf{b}^{[j+1]}$。此外，在可行性验证步骤中，有向图的存在性所需的充要条件将替换为无向图的相应条件（例如，参见 Erd˝os–Gallai 定理；Erd˝os 和 Gallai（1960年））。由于上述对称连接步骤，度序列 $\mathbf{b}^{[j+1]}$ 会被更新，因此需要重新确定下一个待处理节点，即剩余度最大的节点。

4.2.1. 无向图枚举算法为了将 DGE 算法扩展到无向情形，我们将对称连接步骤纳入算法的内层。具体而言，无向图枚举（UGE）算法是在算法A.4的基础上，通过以下修改得到的：(i) 每个节点 $[\mathbf{a}]_j$ 完成其连接后，为节点 $[\mathbf{b}]_j$ 建立相应的对称连接；(ii) 在可行性验证步骤中，将有向图存在性的充要条件替换为无向图相应的条件；(iii) 重新确定$\mathbf{b}^{[j+1]}$ 中剩余度最大的节点，作为下一个待处理节点。UGE 算法的伪代码见电子附录EC.7.1节中的算法A.7。

4.2.2. 无向图采样算法为了将有向图采样算法扩展到无向情形，我们从 UDGS 算法入手。回顾一下，UDGS 算法的关键在于：为区间内每个可行值所对应的每种连接配置赋予精确权重，其中每个权重应与该配置能够生成的

不同有向图的数量成正比。我们首先将对称连接步骤纳入无向图精确权重的计算中。具体而言，在计算精确权重的递归过程中，每当节点 $[\mathbf{b}]_j$ 完成其连接后，其对应节点 $[\mathbf{a}]_j$ 便建立相应的对称连接。完成对称连接步骤后，我们依据无向图存在的充要条件执行可行性验证步骤。随后，对于每个更新后的度序列 $\mathbf{b}^{[j+1]}$，重新确定剩余度最大的节点，将其作为下一个待处理节点。因此，可以得到无向情形下的精确权重向量 $\mathbf{w}^{\mathrm{UU}}$ 。

基于精确权重向量 $\mathbf{w}^{\mathrm{UU}}$ 和 UDGS 算法，我们提出了均匀无向图抽样（UUGS）算法。具体而言，UUGS算法通过对算法 A.5 作如下修改得到：（i）将步骤 7 中使用的权重替换为 $\mathbf{w}^{\mathrm{UU}}$；（ii）在每个节点 $[\mathbf{a}]_j$ 完成其连接后，为节点 $[\mathbf{b}]_j$ 建立相应的对称连接；（iii）重新确定 $\mathbf{b}^{[j+1]}$ 中剩余度最大的节点，将其作为下一个待处理节点。UUGS算法的伪代码见电子附录 EC.7.2 节中的算法 A.8。EC.7.2电子附录。

接下来，我们通过在算法中加入对称连接步骤，将 EDGS 算法扩展到无向情形。具体而言，高效无向图采样算法（EUGS）通过对算法 A.6 作如下修改得到：（i）在每个节点 $[\mathbf{a}]_j$ 完成其连接后，为节点 $[\mathbf{b}]_j$ 建立相应的对称连接；（ii）在可行性验证步骤中，将有向图存在性的充要条件替换为无向图对应的条件；（iii）重新确定 $\mathbf{b}^{[j+1]}$ 中剩余度最大的节点，将其作为下一个待处理节点。EUGS 算法的伪代码见电子附录 EC.7.3 节中的算法 A.9 。EC.7.3 电子附录。注意，与 EBGS 算法类似，在针对 $\mathcal{G}(\mathbf{a},\mathbf{b})$上的均匀分布计算期望时，也可以对 EUGS 算法应用重要性抽样，以校正其非均匀性。

5. 数值实验

在本节中，我们展示所提出的枚举与抽样算法的性能。对于枚举算法，我们将其运行时间表现与专为参照而设计的朴素"蛮力算法进行比较。对于抽样算法，我们将其与*Glasserman*和*de Larrea*的顺序算法(2023)以及*Miller*和*Harrison*的精确采样算法(2013)进行比较，重点考察采样速度、图分布均匀性和图空间覆盖率。二分图、有向图和无向图算法的结果分别在第5.1、5.2和5.3节中展示。所有数值实验均以

Python 编写，并在配备 3.6 GHz 英特尔酷睿 i7-12700K 处理器和 16 GB RAM 的台式计算机上运行。

5.1. 二分图

我们首先从枚举运行时间方面评估 BGE 算法的性能，然后从采样速度、均匀性和图空间覆盖率方面评估 UBGS 和 EBGS 算法的性能。

枚举运行时间。我们将 BGE 算法与以下朴素的¨蛮力算法进行比较：给定度序列 $\mathbf{a}$ 和 $\mathbf{b}$，生成仅满足两个度序列之一的所有图，然后从中筛选出满足两个度序列的所有可行二分图。蛮力算法的伪代码见电子附录第A.10 节EC.8中的算法。

我们在六个度序列上实现了蛮力算法和 BGE 算法，并记录了每个序列的枚举运行时间。结果列于表1，其中包括不同二分图的数量（N_B），以及蛮力算法（T_{BF}）和 BGE 算法（T_{BGE}）的枚举运行时间。结果表明，BGE 的速度比蛮力算法快几个数量级。例如，BGE 约用 0.02 秒枚举出 1,372 个二分图，用 0.3 秒枚举出 15,732 个二分图。相比之下，蛮力算法分别需要约 14 秒和 10^6 秒。

表1 二分图确定性算法的运行时间。

度序列		N_B	蛮力算法	BGE 算法
a	**b**		T_{BF} (秒)	T_{BGE} (秒)
(3, 3, 3, 2, 1)	(5,3,2,1,1)	27	0.03	0.0003
(3,2,2,2,2)	(4, 3, 1, 1, 1, 1)	688	0.4	0.007
(5,5,4,4,3,2)	(5, 5, 5, 3, 3, 2)	1, 372	14	0.02
(7, 7,6, 5,4, 3,2, 2)	(7, 7, 6, 5,4, 3, 3, 1)	15, 732	$\approx 10^6$	0.3
(8, 7,7, 6,5, 4,2, 2, 2)	(8,7, 7,6, 5,4, 3, 2,1)	194, 010	$\gg 10^6$	2.9
(8, 7,6, 6,5, 4,2, 2, 2)	(8,7, 6,6, 5,4, 3, 2,1)	1,991,820	$\gg 10^6$	31

采样速度。我们通过与顺序算法和精确抽样算法进行比较，评估 UBGS 和 EBGS 算法的采样速度。具体而言，我们在银行间-1度序列 a = b = (6, 6, 6, 5, 5, 5, 5, 4, 4, 3, 2) 和银行间-2度序列 a = b =(9, 9, 9, 9, 9, 9, 8, 8, 8, 7, 6) 上实现了所有算法，这两个度序列均用于 Glasserman 和 de Larrea（2023）的数值实验。对于每个度序列，每种算法都运行 10,000 次，与他们论文中的设置一致。表 2 报告了顺序算法原始实现（据其论文所述）、我们复现的版本、精确抽样算法、UBGS 和 EBGS 的平均运行时间。结果表明，

表2 Glasserman 和 de Larrea提出的度序列算法运行时间（均值、标准差）（2023）。

度序列	顺序算法		精确采样算法	UBGS 算法	EBGS 算法
	原始	复现			
银行间-1	0.17	0.15（0.013）	0.0028（0.00065）	0.0017（0.00055）	0.0013（0.00052）
银行间-2	0.15	0.14 (0.017)	0.0017 (0.00053)	0.0016 (0.00048)	0.0012 (0.00043)

在两个度序列上，EBGS 和 UBGS 的速度略快于精确抽样算法，而这三种算法的速度都明显快于顺序算法。

为进一步评估随着问题规模增大这些算法的可扩展性，我们考虑以下度序列族$\mathbf{a} = \mathbf{b} = (n-1, n-1, n-2, n-3, \ldots, 2, 1)$，其中 n表示节点数，而 $n \geq 3$。我们对 $n = 3$ 至 200 的情形评估 UBGS 和 EBGS，对$n = 3$ 至 95 的情形评估顺序算法，因为当 $n \geq 95$时，其平均运行时间超过 100 秒；由于相同的运行时间限制，对 $n = 3$ 至 26 的情形评估精确采样算法。对于每个 n取值，所有算法的每项实验均重复 20 次。图5 报告了随着节点数增加，顺序算法、精确采样算法、UBGS 和 EBGS 的平均运行时间。结果表明，EBGS 和 UBGS 的可扩展性显著优于顺序算法，并且对于这一度序列族，三种算法的可扩展性均优于精确采样算法。

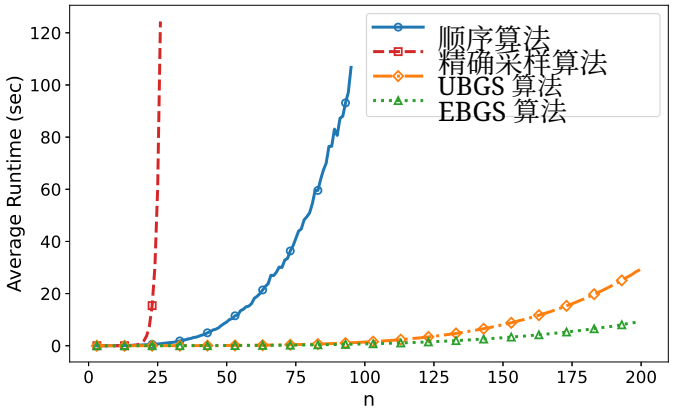


图5 度序列为 $\mathbf{a} = \mathbf{b} = (n-1,\ n-1,\ n-2,\ n-3,\ \ldots\ldots,\ 2,\ 1)$的二分图的运行时间与节点数的关系，取20次重复试验的平均值。

采样均匀性。我们使用两个指标评估所有算法的采样均匀性：变异系数（CV），用于衡量每个采样分布内的离散程度；以及采样分布与相应均匀分布之间的 Kullback–Leibler（KL）散度，用于量化其偏离完全均匀性的程度。对于这两个指标，数值越小表示均匀性越好。比较基于枚举运行时间实验中使用的四个度序列进行，并且对于每个序列，所有算法均生成 5,000

个独立样本。表 3 报告了所得的 CV 和 KL 值。结果表明，在采样均匀性方面，UBGS 和精确采样算法的均匀性最高，其次是顺序算法，而 EBGS 的均匀性最低，尽管其表现仍相当不错。

表 3　二分图算法的采样均匀性：变异系数（变异系数）和 Kullback–Leibler（KL散度）；越小越好。

度序列		顺序算法		精确算法		UBGS 算法		EBGS 算法	
a	**b**	CV	KL	CV	KL	CV	KL	CV	KL
(3, 3, 3, 2, 1)	(5, 3, 2, 1, 1)	9.99E-02	5.04E-03	6.29E-02	1.98E-03	7.31E-02	2.72E-03	2.81E-01	3.95E-02
(3, 2, 2, 2, 2)	(4, 3, 1, 1, 1, 1)	4.02E-01	8.36E-02	3.71E-01	7.01E-02	3.72E-01	6.98E-02	4.92E-01	1.24E-01
(5, 5, 4, 4, 3, 2)	(5, 5, 5, 3, 3, 2)	5.41E-01	1.45E-01	4.96E-01	1.23E-01	4.91E-01	1.20E-01	5.99E-01	1.69E-01
(7, 7, 6, 5, 4, 3, 2, 2)	(7, 7, 6, 5, 4, 3, 3, 1)	3.68E-01	5.46E-02	3.59E-01	5.22E-02	3.54E-01	5.12E-02	4.10E-01	6.37E-02

图空间覆盖率。我们评估 UBGS 和 EBGS 算法的图空间覆盖率，并将其性能与顺序抽样算法和精确抽样算法进行比较。图空间覆盖率定义为算法生成的不同可行二分图数量占整个图空间的比例。我们采用枚举运行时间实验中的一个度序列，即 a = (7, 7, 6, 5, 4, 3, 2, 2) 和 b =(7, 7, 6, 5, 4, 3, 3, 1)，该序列对应 15,732 个不同的二分图，如表 1 第四行所示。对于每种采样算法，我们将运行时间预算设为 10 至 100 秒（每次递增 10 秒），并记录：(i) 生成的样本总数（N_T），(ii) N_T 中不同二分图的数量，即 N_B，以及 (iii) 覆盖率 $R_B = N_B/15{,}732$，即生成的不同可行二分图占整个图空间的比例。对于每种运行时间设置，所有实验均重复 20 次，以计算相应的标准差。

图6 展示了四种算法的覆盖率 R_B 随运行时间预算从 10 秒到 100 秒变化的情况。表4 汇总了四个具有代表性的运行时间预算（10、20、40 和 80 秒）下的详细数值结果，包括 N_T、 N_B和 R_B。图6和表4 中的结果表明，在覆盖率方面，UBGS、EBGS 和精确采样算法的表现始终优于顺序算法。例如，在 10 秒内，UBGS、EBGS 和精确采样算法各自生成了超过 50% 的所有不同二分图，而顺序算法生成的比例不到 3%。此外，UBGS、EBGS 和精确采样算法能够在 80 秒内生成几乎所有不同的二分图，而在相同的运行时间预算下，顺序算法生成的比例不到 19%。此外，在抽样的早期阶段，EBGS 的表现略强于精确采样算法，但在后期最终被其超越，因为精确采样算法根据均匀分布生成样本，而 EBGS 仅近似均匀。

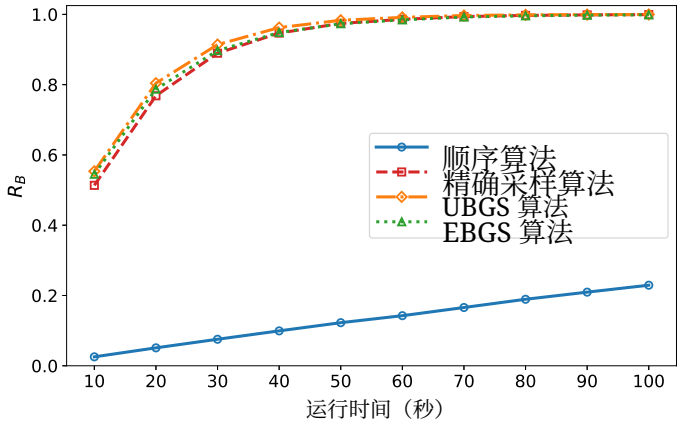


图 6 度序列为 a =(7, 7, 6, 5, 4, 3, 2, 2) **和 b** =(7, 7, 6, 5, 4, 3, 3, 1) **的二分图的覆盖率与运行时间。**

表 4 度序列为 a =(7, 7, 6, 5, 4, 3, 2, 2) **和 b** =(7, 6, 6, 5, 4, 3, 3, 1) **的二分图的覆盖率（标准差）。**

时间（秒）	顺序算法			精确采样算法			UBGS 算法			EBGS 算法		
	N_T	N_B	R_B	N_T	N_B	R_B	N_T	N_B	R_B	N_T	N_B	R_B
10	405.90 (6.17)	400.55 (6.72)	**2.55%** (0.04%)	11,325.25 (133.84)	8,078.30 (63.22)	**51.35%** (0.40%)	12,662.10 (191.59)	8,707.70 (96.45)	**55.35%** (0.61%)	13,152.55 (173.79)	8,575.30 (84.69)	**54.51%** (0.54%)
20	825.75 (6.66)	802.80 (6.30)	**5.10%** (0.04%)	23,033.90 (135.10)	12,088.40 (40.15)	**76.84%** (0.26%)	25,695.35 (255.34)	12,651.50 (64.39)	**80.42%** (0.41%)	26,723.80 (256.41)	12,384.20 (62.48)	**78.72%** (0.40%)
40	1,654.40 (4.99)	1,562.50 (8.64)	**9.93%** (0.05%)	46,084.45 (137.64)	14,894.45 (24.85)	**94.68%** (0.16%)	51,393.90 (456.89)	15,140.35 (23.80)	**96.24%** (0.15%)	53,574.75 (439.99)	14,920.10 (32.99)	**94.84%** (0.21%)
80	3,313.30 (10.20)	2,977.45 (18.71)	**18.93%** (0.12%)	92,190.10 (406.69)	15,686.65 (7.22)	**99.71%** (0.05%)	103,307.30 (884.87)	15,709.55 (6.72)	**99.86%** (0.04%)	107,535.45 (947.81)	15,666.55 (5.80)	**99.58%** (0.04%)

5.2. 有向图

我们首先评估 DGE 算法在枚举运行时间方面的性能，然后评估 UDGS 和 EDGS 算法在采样速度、均匀性和图空间覆盖率方面的性能。

枚举运行时间。我们通过将 DGE算法与蛮力算法进行比较，评估 DGE算法的枚举运行时间。具体而言，我们对六个度序列应用了这两种算法，并记录每个度序列的枚举运行时间。结果见表 5，其中包括不同有向图的数量（N_D），以及蛮力算法（T_{BF}）和 DGE算法（T_{DGE}）的枚举运行时间。结果表明，DGE 比蛮力算法快数个数量级。例如，DGE 枚举 4,628 个有向图仅需约 0.2 秒，枚举 48,700 个有向图仅需 1.8 秒。相比之下，蛮力算法分别需要约 380 秒和 10^4 秒。

采样速度。我们将 UDGS 和 EDGS 算法与顺序算法进行比较，以评估其采样速度。具体而言，我们考虑度序列族$\mathbf{a} = (n-2,\ 1,\ 1,\ 2,\ 3,\ \ldots\ldots,\ n-3,\ n-2)$ 和$\mathbf{b} = (n-1,\ n-3,\ n-3,\ n-4,\ n-5,\ \ldots\ldots,\ 2,\ 1,\ 1)$，其中$n$ 表示节点数，且 $n \geq 4$。我们评估节点数从 $n = 4$ 到 200 时 UDGS 和 EDGS 的表现，并且

表5 有向图确定性算法的运行时间。

度序列 一个= b	N_D	蛮力算法	DGE算法
		T_{BF} (秒)	T_{DGE} (秒)
(2, 2, 1, 1, 1)	42	0.1	0.002
(2, 2, 2, 1, 1, 1)	686	7.6	0.03
(3, 2, 2, 1, 1, 1, 1)	4,628	380	0.2
(3, 3,2, 1,1, 1,1, 1)	48, 700	$\approx 10^4$	1.8
(3, 2,2, 2,1, 1,1, 1)	191, 334	$\approx 10^4$	6.8
(3, 3, 2, 2, 2, 2, 1, 1)	1, 977, 666	$\gg 10^4$	67

将顺序算法的 $n = 4$ 范围设为 78，因为当 $n \geq 78$时，其平均运行时间超过 100 秒。对于每个 n值，每种算法的实验均重复 20 次。图7 展示了随着节点数增加，顺序算法、UDGS 和 EDGS 的平均运行时间。结果表明，对于这一度序列族，EDGS 和 UDGS 的运行速度更快，且扩展性显著优于顺序算法。

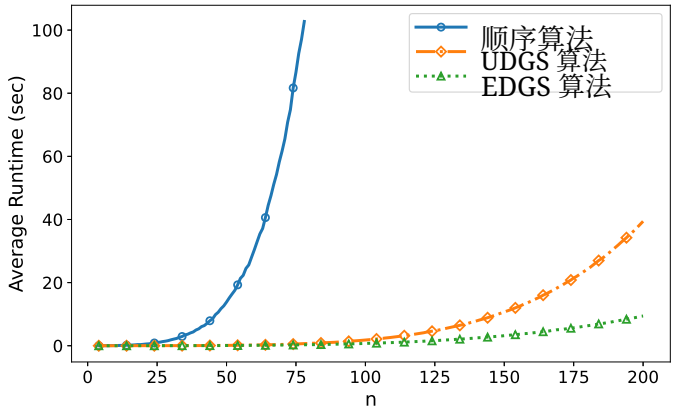


图7 运行时间作为节点数的函数，针对具有度序列 a $= (n-2, 1, 1, 2, 3, \ldots, n-3, n-2)$ **和 b** $= (n-1, n-3, n-3, n-4, n-5, \ldots, 2, 1, 1)$ **的有向图，结果为20次重复试验的平均值。**

采样均匀性。我们使用与二分图实验相同的两个指标，即变异系数和KL散度，评估三种算法的采样均匀性。比较在枚举-运行时间实验所用的四个度序列上进行，对于每个序列，所有算法均生成5,000个独立样本。表6 报告了所得的变异系数和KL值。结果表明，就采样均匀性而言，UDGS达到了最高的均匀性水平，其次是顺序算法，而EDGS的均匀性最低，但仍相当良好。

图空间覆盖率。我们评估 UDGS 和 EDGS 算法的图空间覆盖率，并将其性能与顺序算法进行比较。我们使用枚举运行时间实验中的一个度序列，即**a** =**b** =(3，3，2，1，1，1，1，1)，该序列对应 48,700 个不同有向图，如第四

表6 有向图算法的抽样均匀性。

度序列	顺序算法		UDGS 算法		EDGS 算法	
a =**b**	CV	KL	CV	KL	CV	KL
(2, 2, 1, 1, 1)	1.25E-01	7.45E-03	8.85E-02	3.88E-03	4.00E-01	8.34E-02
(2, 2, 2, 1, 1, 1)	4.04E-01	8.25E-02	3.57E-01	6.53E-02	7.34E-01	2.33E-01
(3, 2, 2, 1, 1, 1, 1)	5.29E-01	1.21E-01	5.22E-01	1.19E-01	6.25E-01	1.64E-01
(3, 3, 2, 1, 1, 1, 1)	2.22E-01	1.95E-02	2.13E-01	1.78E-02	2.59E-01	2.58E-02

行表5所示。对于每种采样算法，我们将运行时间预算从 10 秒调整到 100 秒（每次递增 10 秒），并记录：(i) 生成的样本总数（N_T）、(ii) N_T中的不同有向图数量，即 N_D，以及 (iii) 覆盖率 $R_D = N_D/48,700$，即在整个图空间中生成的不同可行有向图所占的比例。对于每种运行时间设置，所有实验均重复 20 次，以计算相应的标准差。

图8 报告了覆盖率 R_D 随运行时间预算从10到100秒的变化情况，针对三种算法。表7 汇总了四个代表性运行时间预算（10、20、40和80秒）下的详细数值结果，包括 N_T、 N_D和 R_D。图8 和表7 中的结果表明，就覆盖率而言，UDGS和EDGS始终优于顺序算法。例如，在10秒内，UDGS生成了全部不同有向图的37%以上，EDGS超过29%，而顺序算法不到1%。此外，在80秒内，UDGS生成了全部不同有向图的97%以上，EDGS超过86%，而顺序算法不到6%。

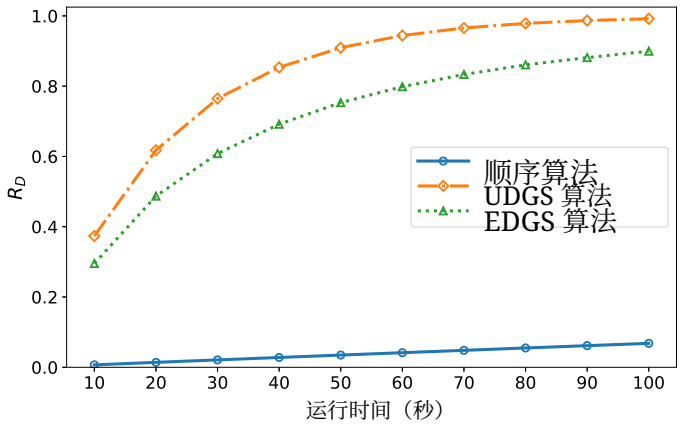


图8 覆盖率 有向图度序列为 $\mathbf{a} = \mathbf{b} = (3,3,2,1,1,1,1,1)$时的比率与运行时间 .

5.3. 无向图

我们首先从枚举运行时间的角度评估 UGE算法的性能，然后从采样速度、均匀性和图空间覆盖率的角度评估 UUGS 和 EUGS算法的性能。

表 7 度序列为 $\mathbf{a}=\mathbf{b}=(3,3,2,1,1,1,1,1)$的有向图覆盖率。

时间（秒）	顺序算法			UDGS 算法			EDGS 算法		
	T_D	N_D	R_D	T_D	N_D	R_D	T_D	N_D	R_D
10	335.15 (3.86)	333.55 (3.56)	**0.68%** (0.01%)	22,767.40 (162.00)	18,192.80 (114.18)	**37.36%** (0.23%)	18,753.40 (131.41)	14,384.00 (81.65)	**29.54%** (0.17%)
20	690.15 (7.62)	685.40 (6.82)	**1.41%** (0.01%)	46,783.25 (281.79)	30,075.55 (114.31)	**61.76%** (0.23%)	38,745.10 (182.53)	23,721.45 (77.69)	**48.71%** (0.16%)
40	1,380.00 (8.34)	1,360.90 (9.73)	**2.79%** (0.02%)	93,568.65 (778.12)	41,567.50 (152.09)	**85.35%** (0.31%)	77,256.95 (651.99)	33,687.10 (136.34)	**69.17%** (0.28%)
80	2,757.40 (15.82)	2,678.45 (13.17)	**5.50%** (0.03%)	187,559.90 (1,082.68)	47,660.65 (38.54)	**97.87%** (0.08%)	155,001.65 (1,022.14)	41,915.80 (81.17)	**86.07%** (0.17%)

枚举运行时间。我们通过将 UGE算法与蛮力算法进行比较，评估 UGE算法的枚举运行时间。具体而言，我们将两种算法分别应用于六个度序列，并记录每个度序列的枚举运行时间。结果如表 8 所示，其中包括不同的无向图数量（N_U）、蛮力算法的枚举运行时间（T_{BF}）以及 UGE算法的枚举运行时间（T_{UGE}）。结果表明，UGE 的速度比蛮力算法快几个数量级。例如，UGE 约在 0.01 秒内枚举出 130 个无向图，并在 0.1 秒内枚举出 1,074 个无向图。相比之下，蛮力算法分别需要约 10^3 秒和 10^6 秒。

表8 无向图确定性算法的运行时间。

度序列 $\mathbf{a}=\mathbf{b}$	N_U	蛮力算法	UGE算法
		T_{BF} (秒)	T_{UGE} (秒)
(2,2,2,2,1,1)	31	19	0.003
(3, 3, 2, 2, 2, 1, 1)	130	$\approx 10^3$	0.01
(3, 3,2, 2,2, 2,1, 1)	1,074	$\approx 10^6$	0.1
(3, 3,3, 3,2, 2,2, 1, 1)	13,801	$\gg 10^6$	1.8
(4, 3,3, 3,3, 2,2, 2, 1, 1)	187, 426	$\gg 10^6$	27
(4, 3,3, 3,3, 3,2, 2, 1, 1, 1)	1,542,445	$\gg 10^6$	263

采样速度。我们通过与顺序算法进行比较，评估 UUGS 和 EUGS 算法的采样速度。具体而言，我们考虑以下度序列族，其可行无向图的数量会随着节点数的增加而增加：

$$\mathbf{a}=\mathbf{b}=\big(c+a,\, c+b,\, \underbrace{c,\ldots,c}_{c-1 \text{ times}},\, \underbrace{1,\ldots,1}_{l \text{ times}}\big), \tag{19}$$

其中 $c=n-l-1$，$a=\lceil l/2\rceil$，$b=l-a$，$l=\lfloor \text{对数}_2\, n\rfloor$，$n$ 表示节点数，而 $n\geq 8$。我们评估了顺序算法、UUGS 和 EUGS 算法在 $n=8$ 至 200 的情况下的表现。对于 n的每个取值，所有算法的每项实验均重复 20 次。图 9 展示了三种算法的平均运行时间随节点数增加的变化结果。结果表明，对于这一类度序列，EUGS 和 UUGS

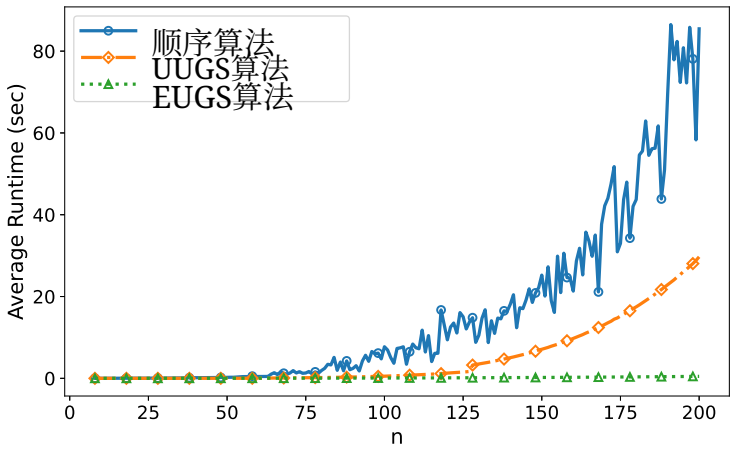


图9 运行时间随节点数的变化：度序列由（19）给出的无向图，对20次重复试验取平均。

比顺序算法运行更快，并且具有显著更好的扩展性。

采样均匀性。我们采用与二分图实验相同的两项指标，即变异系数和KL散度，评估三种算法的采样均匀性。比较基于枚举运行时间实验中使用的四个度序列进行；对于每个序列，所有算法均生成5,000个独立样本。表9 报告了所得的变异系数与KL散度值。结果表明，就采样均匀性而言，UUGS达到最高的均匀性水平，其次是顺序算法，而EUGS的均匀性最低，但仍相当良好。

表9 无向图算法的采样均匀性。

度序列	顺序算法		UUGS算法		EUGS算法	
a =**b**	CV	KL	CV	KL	CV	KL
(2, 2,2, 2,1, 1)	1.22E-01	7.46E-03	8.16E-02	3.38E-03	3.70E-01	4.68E-02
(3, 3, 2, 2, 2, 1, 1)	1.67E-01	1.38E-02	1.58E-01	1.24E-02	4.06E-01	6.25E-02
(3, 3,2, 2,2, 2,1, 1)	4.55E-01	1.07E-01	4.54E-01	1.05E-01	6.33E-01	1.75E-01
(3, 3,3, 3,2, 2,2, 1, 1)	3.80E-01	5.88E-02	3.70E-01	5.57E-02	4.77E-01	8.73E-02

图空间覆盖率。我们评估 UUGS 和 EUGS 算法的图空间覆盖率，并将其性能与顺序算法进行比较。我们采用枚举运行时间实验中的一个度序列，即 $\mathbf{a} = \mathbf{b}$ =(3，3，3，3，2，2，2，1，1)，对应 13,801 个不同的无向图，如表 8第四行所示。对于每种采样算法，我们将运行时间预算设为 10 到 100 秒（每次递增 10 秒），并记录：(i) 生成的样本总数（N_T）；(ii) N_T内不同无向图的数量，即 N_U；以及 (iii) 覆盖率 $R_U = N_U/13,801$，即所生成的不同可行无向图占整个

图空间的比例。对于每种运行时间设置，所有实验均重复 20 次，以计算相应的标准差。

图 10 报告了三种算法在运行时间预算为 10 到 100 秒时的覆盖率 R_U 。表 10 汇总了三个具有代表性的运行时间预算（10、20 和 40 秒）下的详细数值结果，包括 N_T、 N_U和 R_U。图 10 和表 10 显示，UUGS 和 EUGS 在覆盖率方面始终优于顺序算法。例如，在 10 秒内，UUGS 生成了所有不同无向图的 78% 以上，EUGS 生成了 64% 以上，而顺序算法生成的不到 4%。此外，在 40 秒内，UUGS 生成了所有不同无向图的 99% 以上，EUGS 生成了 96% 以上，而顺序算法生成的不到 13%。

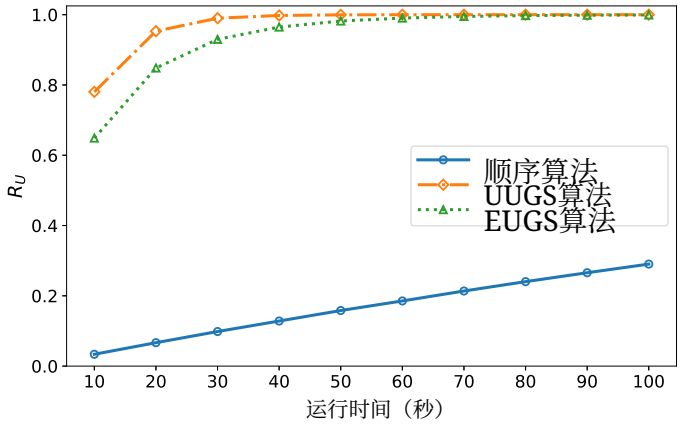


图10 度序列为 a ＝b ＝(3, 3, 3, 3, 2, 2, 2, 1, 1) 的有向图覆盖率与运行时间的关系。

表10　　度序列为 a ＝b ＝(3, 3, 3, 3, 2, 2, 2, 1, 1 的无向图覆盖率).

时间（秒）	顺序算法			UUGS算法			EUGS算法		
	T_U	N_U	R_U	T_U	N_U	R_U	T_U	N_U	R_U
10	473.15 (6.08)	464.35 (7.96)	**3.36% (0.06%)**	20,969.65 (125.21)	10,770.30 (45.27)	**78.04% (0.33%)**	17,820.15 (78.74)	8,955.40 (40.52)	**64.89% (0.29%)**
20	952.55 (7.00)	919.50 (6.76)	**6.66% (0.05%)**	42,322.85 (349.68)	13,154.65 (36.61)	**95.32% (0.27%)**	35,707.10 (206.52)	11,706.40 (40.39)	**84.82% (0.29%)**
40	1,898.50 (11.82)	1,770.30 (13.14)	**12.93% (0.10%)**	84,394.40 (610.73)	13,770.60 (7.56)	**99.78% (0.05%)**	71,369.40 (645.63)	13,315.15 (24.98)	**96.48%** (0.18%)

6. 结论

本文提出了一种统一的序贯方法，用于为二分图、有向网络和无向网络生成具有给定度序列的图。在此方法中，我们建立了二分图生成的充要区间条件，用以刻画

每一步允许的连接数，从而保证全局可行性。在此方法基础上，我们设计了二分图的枚举与抽样算法：前者穷举生成所有可行配置，后者采用均匀采样，或高效但略有偏差的采样。通过加入额外的自环和单度连接约束，并结合可行性验证与对称连接步骤，这些二分图算法得以扩展至有向和无向情形，同时保留相同的算法原理。数值实验表明，所提出算法在枚举运行时间以及抽样速度、均匀性和图空间覆盖率方面均表现有效。

致谢

本研究的初步版本发表于 2025 年冬季仿真会议论文集（Sun 等人，2025）。

参考文献

Acemoglu D, Carvalho VM, Ozdaglar A, Tahbaz-Salehi A (2012) 总体波动的网络根源。《计量经济学》80(5):1977–2016. Adrian T, Brunnermeier MK (2016) CoVaR.《美国经济评论》106(7):1705–41. Barrot JN, Sauvagnat J (2016) 生产网络中的投入特异性与特异性冲击传播。《经济学季刊》131(3):1543–1592. Barvinok A (2010) 关于具有给定行和、列和及 0–1 元素的矩阵数量和随机矩阵。《数学进展》224(1):316–339. Berger A (2014) 关于有向图度序列刻画的一则短论。《离散数学》314:38–41. Birge JR, Capponi A, Chen PC (2023) 供应链网络中的中断与改道。《运筹学》71(2):750–767. Blitzstein J, Diaconis P (2011) 一种用于生成具有给定度的随机图的序贯重要性抽样算法。《互联网数学》6(4):489–522. Carvalho VM,Nirei M,Saito YU, Tahbaz-Salehi A (2021) 供应链中断：来自东日本大地震的证据。《经济学季刊》136(2):1255–1321. Eisenberg L, Noe TH (2001) 金融系统中的系统性风险。《管理科学》47(2):236–249. Erd˝os P, Gallai T (1960) 具有给定顶点度的图。《数学笔记》11:264–274. Fosdick BK, Larremore DB, Nishimura J, Ugander J (2018) 通过固定度序列配置随机图模型。《*SIAM* 评论》60(2):315–355. Fu MC, Li B, Li F, Wu R (2025) 传染网络、投资组合信用风险与金融危机。《欧洲运筹学杂志》321(3):942–957.

Gale D (1957) 网络流定理。《太平洋数学杂志》7(2):1073–1082. Gandy A, Veraart LAM (2017) 一种用于金融网络系统性风险评估的贝叶斯方法。《管理科学》63(12):4428–4446. Glasserman P, de Larrea EL (2018) 使用最大熵概率对具有给定度序列的二分图或有向图进行仿真。Rabe M, Juan AA, Mustafee N, Skoogh A, Jain S, Johansson B, 编，*2018*年冬季仿真会议论文集，1658–1669（美国新泽西州皮斯卡塔韦：IEEE）。Glasserman P, de Larrea EL (2023) 最大熵分布及其在图仿真中的应用。《运筹学》71(5):1908–1924. Glasserman P, Young HP (2016) 金融网络中的传染。《经济文献杂志》54(3):779–831. Huang W, Lin N, Hong LJ (2024) CoVaR的蒙特卡洛估计。《运筹学》72(6):2337–2357. Marsden PV (2014) 网络数据的调查方法。Scott J, Carrington PJ, 编，《*SAGE*社会网络分析手册》，370–388（伦敦：SAGE Publications Ltd）。Miller JW, Harrison MT (2013) 固定边际矩阵的精确抽样与计数。《统计年鉴》41(3):1569–1592. Mungo L, Brintrup A, Garlaschelli D, Lafond F (2024) 供应网络重构。《物理学杂志：复杂性》5(1):012001. Squartini T, Cimini G, Gabrielli A, Garlaschelli D (2017) 通过密度抽样进行网络重构。《应用网络科学》2(1):3. Sun T, Hao J, Zhang Z, Jiang G (2025) 一种基于给定度序列的高效二分图采样算法。Azar E, Djanatliev A, Harper A, Kogler C, Ramamohan V, Anagnostou A, Taylor SJE, 编，*2025*年冬季仿真会议论文集，259–270（美国新泽西州皮斯卡塔韦：IEEE）。Verhelst ND (2008) 一种用于抽取具有固定边际和的二元矩阵的高效马尔可夫链蒙特卡洛算法。《心理计量学》73(4):705–728. Wormald NC (1999) 随机正则图模型。Lamb JD, Preece DA, 编，《组合学综述，*1999*》，239–298，伦敦数学学会讲义系列（剑桥：剑桥大学出版社

).

《部分信息下的图仿真方法》电子附录

EC.1. 引理1的证明

在证明引理1之前，我们需要以下引理。

LEMMA A.1**(Gale–Ryser 定理（Gale 1957年）)**。 对于给定度序列 $\mathbf{a}$和 $\mathbf{b}$，存在一个二分图，当且仅当 $\sum_{i=1}^{m}[\mathbf{a}]_i=\sum_{j=1}^{n}[\mathbf{b}]_j$，且对于所有 $r\in\{1,2,\ldots,n\}$，以下条件均成立：

$$\sum_{i=1}^{m}\min\{[\mathbf{a}]_i,r\}\geq\sum_{j=1}^{r}[\mathbf{b}]_j.$$

接下来，我们证明引理 1。

证明。 根据（1）和（2），可得 $\min\{[\mathbf{a}]_i,r\}=\sum_{j=1}^{r}[\mathscr{L}(\mathbf{a},|\mathbf{b}|)]_{ij}$ 以及 $[\mathbf{z}]_j=\sum_{i=1}^{m}[\mathscr{L}(\mathbf{a},|\mathbf{b}|)]_{ij}$。于是，我们有

$$\sum_{i=1}^{m}\min\{[\mathbf{a}]_i,r\}=\sum_{i=1}^{m}\sum_{j=1}^{r}[\mathscr{L}(\mathbf{a},|\mathbf{b}|)]_{ij}=\sum_{j=1}^{r}\sum_{i=1}^{m}[\mathscr{L}(\mathbf{a},|\mathbf{b}|)]_{ij}=\sum_{j=1}^{r}[\mathbf{z}]_j.$$

因此，根据引理A.1，我们得出该引理。 □

EC.2. 定理1的证明

在证明该定理之前，我们介绍两种贪心算法，它们将第一个节点 $[\mathbf{b}]_1$与 $\mathbf{a}$ 中的节点相连，并生成更新后的度序列 $\mathbf{a}^{[2]}$ 和$\mathbf{b}^{[2]}$。这些贪心算法不同于第3节中介绍的二分图生成机制，并将在证明中作为辅助构造。

EC.2.1. 贪心算法

我们首先从标准贪心算法出发，然后构造其 k步变体。标准贪心算法总结于算法A.1。给定度序列$\mathbf{a}$和$\mathbf{b}$，$\mathbf{a}$中的节点按其不同度值被划分为 p 组：组 k 由度为 α_k的 m_k个节点组成，其中 $\alpha_1<\alpha_2<\cdots<\alpha_p$ 且$\sum_{k=1}^{p}m_k=m$。标准贪心算法以贪心反向方式将节点 $[\mathbf{b}]_1$ 连接到$\mathbf{a}$中的节点。具体而言，它从度最大的组（组 p）开始，将 $[\mathbf{b}]_1$ 连接到该组中的所有节点，然后依次处理组 $p-1$、组 $p-2$，依此类推，直到 $[\mathbf{b}]_1$的度被耗尽。建立这些连接后，任何度降至零的节点都会被移除，从而得到更新后的度序列$\mathbf{a}^{[2]}$ 和$\mathbf{b}^{[2]}$。与第3节中介绍的二分图生成机制不同，后者通过优先处理低度数组来连接 $[\mathbf{b}]_1$，而贪心算法采用

算法 A.1 标准贪心算法

输入： 度序列 $\mathbf{a} \in \mathbb{N}_+^m$ 和 $\mathbf{b} \in \mathbb{N}_+^n$
输出： 更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$
将 $\mathbf{a}$ 分成 p 组 $(\alpha_k,\ m_k)$，其中包含 $\alpha_1 < \alpha_2 < \cdots < \alpha_p$
对于 $k = p$ **到** 1，**执行**
    令 $\mathscr{F}(k) \leftarrow \min\left\{m_k, [\mathbf{b}]_1^{(k)}\right\}$，其中 $[\mathbf{b}]_1^{(k)}$ 表示 $[\mathbf{b}]_1$ 的剩余度
    均匀选择组 k 中 $\mathscr{F}(k)$ 个互不相同的节点，并将它们连接到节点 $[\mathbf{b}]_1$
从度序列中移除零度节点
返回 更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$

相反策略，即优先处理高度数组，并在继续之前将每一组完全耗尽。

基于这一思想，我们引入 k步贪心算法，其概述见算法A.2。与之前一样，$\mathbf{a}$ 中的节点按其度被划分为 p 组。它与标准贪心算法的关键区别在于前 $k-1$ 步。在前 $k-1$ 步中，节点 $[\mathbf{b}]_1$ 按照第 3 节所述二分图生成机制中的相同顺序连接到第 1 组至第 $k-1$ 组。此外，从这些组中各自连接到 $[\mathbf{b}]_1$ 的节点数规定如下。由于度序列 $\mathbf{a}$ 和 $\mathbf{b}$ 存在二分图，因此 $[\mathbf{b}]_1$ 至少存在一种连接到 $\mathbf{a}$ 中节点的可行方式，该方式可由各组的连接节点数来刻画。相应地，存在一个可行集合 $\sim\{\mathscr{F}(k')\}_{k'=1}^{p}$，其中 $\mathscr{F}(k')$表示组 k'中$m_{k'}$个节点的连接节点数，使得所得更新后的度序列 $\mathbf{a}^{[2]}$ 与 $\mathbf{b}^{[2]}$ 对于二分图 $\sim$ 生成仍然可行。对于 k步贪心算法，在前 $k-1$ 步中，将每个组 $k' \in \{1, 2, \ldots, k-1\}$中的$\mathscr{F}(k')$个节点连接到 $[\mathbf{b}]_1$。从第 k 步起，算法切换为贪心反向策略：节点 $[\mathbf{b}]_1$ 按降序依次连接到其余各组，从组 p开始，然后是组 $p-1$、组 $p-2$，依此类推，直至其度耗尽。建立这些连接后，移除所有度降至零的节点，从而得到更新后的度序列 $\mathbf{a}^{[2]}$ 与 $\mathbf{b}^{[2]}$。对于由 k步贪心算法生成的度序列 $\mathbf{a}^{[2]}$ 与 $\mathbf{b}^{[2]}$，我们建立如下引理，以保证二分图的存在性。

LEMMA A.2. 假设对于给定度序列$\mathbf{a}$ 和$\mathbf{b}$存在一个二分图。那么，对于由k步贪心算法生成的更新后的度序列$\mathbf{a}^{[2]}$ 和$\mathbf{b}^{[2]}$，也存在一个二分图。

证明。 由于度序列$\mathbf{a}$ 和$\mathbf{b}$存在二分图，因此存在一个可行性集合 $\sim\{\mathscr{F}(k')\}_{k'=1}^{p}$，使得 $[\mathbf{b}]_1$能够连接到 $\mathscr{F}(k')$ 个来自组 k'的已分组$\mathbf{a}$中的节点，从而恰好耗尽 $[\mathbf{b}]_1$的度。在完成这些连接之后，所得的更新后的度序列$\mathbf{a}^{[2]}$ 和$\mathbf{b}^{[2]}$ 对于二分图生成仍然具有可行性。

算法 A。2 k步贪心算法

输入：度序列 a $\in \mathbb{N}_+^m$和 b $\in \mathbb{N}_+^n$ 输出：更新后的度序列 a $^{[2]}$ $^{[2]}$和 b：将 a 分成 p $\alpha_{k'}$ $m_{k'}$ $\alpha_1 < \alpha_2 < \cdots < \alpha_p$ 组 (,)，其中：对于 to 执行 $k'=1$ $k=1$ $\mathscr{F}$ k' k' $[\ ]_1$：均匀选择组中 ( ) 个不同节点，并将它们连接到节点 b：对于向下 to 执行 $k'=p$ k $\mathscr{F}$ k' ← $\{m_{k'}, [\ ]_1^{(k'}\}$ $[\ ]_1^{(k'}$ $[\ ]_1$：令 ( ) min b)，其中 b) 表示 b 的剩余度：均匀选择组中 ( ) 个不同节点，并将它们连接到节点 $\mathscr{F}$ k' k' $[\ ]_1$ b：从度序列中移除零度节点 ：返回更新后的度序列 a$^{[2]}$和 b$^{[2]}$

根据引理 1，由于由 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 诱导的度序列 $\mathscr{F}(k')$ 对二分图生成仍具可行性，因此对所有 $r \in \{1, 2, \ldots, n-1\}$，

$$\sum_{j=1}^{r} \left[\widetilde{\mathbf{z}}^{[2]}\right]_j \geq \sum_{j=1}^{r} \left[\mathbf{b}^{[2]}\right]_j, \tag{A.1}$$

其中 $\widetilde{\mathbf{z}}^{[2]}$ 表示由 $\mathbf{z}^{[2]}$ 根据（$\mathscr{Z}\left(\mathbf{a}^{[2]}, \left|\mathbf{b}^{[2]}\right|\right)$）中的 2得到的向量，更新后的度序列 $\widetilde{\mathbf{a}}^{[2]}$ 和 $\widetilde{\mathbf{b}}^{[2]}$ 由 $\mathscr{F}(k')$ 诱导。

令 $\widehat{\mathscr{F}}(k')$ 表示通过 k 步贪心算法，从组 k' 中的 $m_{k'}$ 个节点连接到的节点数。不失一般性，

$$\widehat{\mathscr{F}}(k') = \widetilde{\mathscr{F}}(k') \quad \text{for all } k' \in \{1, 2, \ldots, k-1\}.$$

从第 k 步开始，k 步贪心算法遵循贪心反向策略，将节点 $[\mathbf{b}]_1$ 的剩余度依次分配给组 p，$p-1$ ..... k，按度数降序排列，并在转向度较低的组之前，先将每个度较高的组分配完。因此，对于任何不小于 k 的组索引，在贪心反向策略下，从组 k 到该组的连接节点累计数，不大于任何可行分配下的累计数，即，

$$\sum_{k'=k}^{l} \widehat{\mathscr{F}}(k') \leq \sum_{k'=k}^{l} \widetilde{\mathscr{F}}(k'), \quad \text{for all } l \in \{k, k+1, \ldots, p\}$$

将此不等式与前 $k-1$ 组中的等式结合可得

$$\sum_{k'=1}^{l} \widehat{\mathscr{F}}(k') \leq \sum_{k'=1}^{l} \widetilde{\mathscr{F}}(k'), \quad \text{for all } l \in \{1, 2, \ldots, p\} \tag{A.2}$$

根据注记 1，量 $\mathscr{F}$（k'）也表示第 $\alpha_{k'}$ 个分量（即 $\mathbf{z}$ 的该分量）的减少，从而决定所得向量 $\mathbf{z}^{[2]}$。因此，$\mathscr{F}$（k'）的值越小，对应第 $\alpha_{k'}$ 个分量（即 $\mathbf{z}^{[2]}$ 的该分量）越大。由（A.2）可知，对所有 $r \in \{1, 2, \ldots, n-1\}$，

$$\sum_{j=1}^{r} \left[\widehat{\mathbf{z}}^{[2]}\right]_j \geq \sum_{j=1}^{r} \left[\widetilde{\mathbf{z}}^{[2]}\right]_j, \tag{A.3}$$

其中 $\widehat{\mathbf{z}}^{[2]}$ 表示向量 $\mathbf{z}^{[2]}$，该向量由 $\mathscr{L}\left(\mathbf{a}^{[2]}, |\mathbf{b}^{[2]}|\right)$ 经（2）得到，其中更新后的度序列 $\mathbf{a}^{[2]}$ 与 $\mathbf{b}^{[2]}$ 由 k 步贪心算法产生。

将 (A.3) 与 (A.1) 结合，可得对所有 $r \in \{1, 2, \ldots, n-1\}$，

$$\sum_{j=1}^{r} \left[\widehat{\mathbf{z}}^{[2]}\right]_j \geq \sum_{j=1}^{r} \left[\mathbf{b}^{[2]}\right]_j . \tag{A.4}$$

由引理 1可知，对于由 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 所生成的更新后度序列，存在一个二分图。该序列由 k 步贪心算法生成。因此，我们得出引理的证明。□

为了说明 k 步贪心算法和引理 A.2，我们考虑一个示例，其中 $\mathbf{a}$ =(1, 1, 2, 2, 3)，$\mathbf{b}$ =(4, 3, 2)。对于 2 步贪心算法，步骤如下。首先，将 $\mathbf{a}$ 中的节点分成三组（$p = 3$）：第 1 组由两个度为 1 的节点组成（$\alpha_1 = 1, m_1 = 2$），第 2 组由两个度为 2 的节点组成（$\alpha_2 = 2, m_2 = 2$），第 3 组由一个度为 3 的节点组成（$\alpha_3 = 3, m_3 = 1$）。接下来，度为 4 的节点 $[\mathbf{b}]_1$ 取一个可行值$\mathscr{F}$(1)—例如 $\mathscr{F}(1) = 1$—并依次与 $\mathbf{a}$ 中的所有节点连接，从第 3 组到第 2 组。因此，$[\mathbf{b}]_1$ 与第 1 组中的一个节点连接（使其度从 1 降至 0）、第 3 组中的一个节点连接（使其度从 3 降至 2），并与第 2 组中的两个节点连接（使它们的度从 2 降至 1）。这些连接完成后，$[\mathbf{b}]_1$ 和第 1 组中度为 0 的那个节点被移除，得到更新后的度序列 $\mathbf{a}^{[2]}$ =(1, 1, 1, 2) 和 $\mathbf{b}^{[2]}$ =(3, 2)，它们存在一个二分图。

EC.2.2. 定理1的证明

我们通过对组索引 k 使用数学归纳法来证明该定理。

归纳命题。对于每个 $k \in \{1, 2, \ldots, p\}$，将命题 $\mathcal{P}(k)$ 定义如下。

假设给定度序列 $\mathbf{a}$ 和 $\mathbf{b}$对应的二分图存在，并且在节点 $[\mathbf{b}]_1$ 与 $\mathbf{a}$的第 1 组至第 $k-1$ 组中的节点连接后，得到的更新后的度序列对于二分图生成仍然可行。[1] 那么，在节点 $[\mathbf{b}]_1$与组 k中的 $\mathscr{F}(k)$ 个节点连接后，当连接节点的数量 $\mathscr{F}(k)$ 满足以下条件时，得到的更新后的度序列仍然可行，当且仅当

$$\mathscr{F}(k) \in \left[\max\left\{[\mathbf{b}]_1^{(k)} - M^{(k)}, 0\right\}, \min\left\{m_k, D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}\right\}\right].$$

证明思路。我们首先建立基例 $\mathcal{P}$(1)，其证明分为两步。首先，我们推导出 $\mathscr{F}$(1) 的一个区间，使得存在一个整数集合 $\{\mathscr{F}(k)\}_{k=2}^{p}$，可将节点 $[\mathbf{b}]_1$ 的度分配到 $\mathbf{a}$的 p 个组中。然而，该区间内的 $\mathscr{F}$(1) 并不能保证得到的更新后的度序列对于二分图

[1] 后一种可行性假设等价于存在一个集合 $\{\mathscr{F}(k')\}_{k'=k}^{p}$，使得所得更新后的度序列$\mathbf{a}^{[2]}$ 与$\mathbf{b}^{[2]}$ 对于二分图生成仍然可行。

图生成。其次，利用引理 1 和标准贪心算法，我们刻画$\widetilde{\ }$该区间的子集，其中存在可行集合 $\{\mathscr{F}(k)\}_{k=2}^{p}$ 使得所得更新后的度序列 $\mathbf{a}^{[2]}$ 和$\mathbf{b}^{[2]}$ 仍然可行。

接下来考虑归纳步骤。任取 $k \in \{2,3,\ldots,p\}$，并假设对于所有 $k' \in \{1,2,\ldots,k-1\}$， $\mathcal{P}(k')$ 均成立。特别地，存在一个可行集合 $\{\widetilde{\mathscr{F}}(k')\}_{k'=1}^{k-1}$，满足 $\mathcal{P}(k')$ 为所有 $k' \in \{1,2,\ldots,k-1\}$规定的区间条件，使得在节点 $[\mathbf{b}]_1$ 连接到 $\mathbf{a}$ 的第 1 组至第 $k-1$ 组中的节点后，得到的更新后的度序列仍然可行。在这一归纳假设下， $\mathcal{P}(k)$ 的证明沿用基例中的两步结构。我们首先推导出一个 $\mathscr{F}(k)$ 区间，使得存在一个整数集合 $\{\mathscr{F}(k')\}_{k'=k+1}^{p}$ ，可以将节点 $[\mathbf{b}]_1$ 的剩余度分配到 $\mathbf{a}$ 的剩余组中；然后利用引理 1 和 k 步贪心算法，刻画该区间的一个子集，在该子集中存在一个可行集合 $\{\mathscr{F}(k')\}_{k'=k+1}^{p}$ ，使得得到的更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 仍然可行。

证明。基例： $k=1$。我们首先在以下引理中证明 $\mathcal{P}(1)$。

Lemma A.3. 假设给定度序列 $\mathbf{a}$ 和 $\mathbf{b}$对应的二分图存在。那么，在节点 $[\mathbf{b}]_1$ 与 (1) 个第 *1* 组中的节点相连后，所得的更新后的度序列对于二分图生成仍然可行，当且仅当相连节点数 $\mathscr{F}(1)$ 满足

$$\mathscr{F}(1) \in \left[\max\left\{[\mathbf{b}]_1^{(1)} - M^{(1)}, 0\right\}, \min\left\{m_1, D_{\text{left}}^{(1)} + D_{\text{now}}^{(1)} + D_{\text{right}}^{(1)}\right\}\right].$$

引理 *A.3* 的证明。我们首先推导出一个 $\mathscr{F}(1)$ 的区间，使得存在一个整数集合 $\{\mathscr{F}(k)\}_{k=2}^{p}$，满足 $0 \le \mathscr{F}(k) \le m_k$，从而节点 $[\mathbf{b}]_1$的度可以分配到 p $\mathbf{a}$ 的各个组中，其中 $\mathbf{a}$， m_k 表示 k 个 $\mathbf{a}$组中的节点数。

回忆一下， $[\mathbf{b}]_1^{(1)}$ 表示节点 $[\mathbf{b}]_1$的初始度。由于第 1 组中相连节点的数量 $\mathscr{F}(1)$ 不能超过该组的节点数，也不能超过节点$[\mathbf{b}]_1$, 的度，因此我们得到

$$\mathscr{F}(1) \in \left[0, \min\left\{m_1, [\mathbf{b}]_1^{(1)}\right\}\right]. \tag{A.5}$$

回顾一下， $M^{(1)}$ 表示从第 2 组到第 p 组中**的**节点总数。如果度$[\mathbf{b}]_1^{(1)}$大于 $M_{(1)}$，那么节点 $[\mathbf{b}]_1$ 必须至少与第 1 组中的 $[\mathbf{b}]_1^{(1)} - M^{(1)}$个节点相连，这样其剩余度才能由其余各组容纳，这意味着 $\mathscr{F}(1) \ge$最大值$\left\{[\mathbf{b}]_1^{(1)} - M^{(1)},\ 0\right\}$。将此不等式与（A.5）结合，可得

$$\mathscr{F}(1) \in \left[\max\left\{[\mathbf{b}]_1^{(1)} - M^{(1)}, 0\right\}, \min\left\{m_1, [\mathbf{b}]_1^{(1)}\right\}\right]. \tag{A.6}$$

注意，对于任意满足（ $\mathscr{F}(1)$）和（A.6）的整数集合 $\{\mathscr{F}(k)\}_{k=2}^{p}$，存在 $0 \le \mathscr{F}(k) \le m_k$ 使得

$$\mathscr{F}(1) + \sum_{k=2}^{p} \mathscr{F}(k) = [\mathbf{b}]_1^{(1)}, \tag{A.7}$$

也就是说，节点 $[\mathbf{b}]_1$ 的度可以分配到 $\mathbf{a}$ 的所有 p 组中。然而，仅凭这一点并不能保证由满足(A.6)该条件的 $\mathscr{F}(1)$ 与满足 (A.7) 的集合 $\{\mathscr{F}(k)\}_{k=2}^{p}$ 所诱导的更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$，仍可用于二分图生成。事实上，对于满足 (A.6) 的某些 $\mathscr{F}(1)$ 取值，存在一个可行集合$\backsim\{\mathscr{F}(k)\}_{k=2}^{p}$，使得所得的更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 仍然可行；而对于其他取值，则不存在这样的集合。

接下来，我们利用引理 A.6，以及标准贪心算法，刻画区间（$\backsim\{\mathscr{F}(k)\}_{k=2}^{p}$ 的子集，其中存在一个可行集合，使得所得的更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 仍然可行。1

根据引理1，对于由给定的$\mathbf{a}^{[2]}$ 和$\mathbf{b}^{[2]}$ 诱导出的更新后的度序列 $\mathscr{F}$（1）满足（A.6），并结合满足（A.7）的整数集合，若要使其仍满足二分图生成的可行性条件，则 $\sum_{i=1}^{m^{[2]}} \left[\mathbf{a}^{[2]}\right]_i = \sum_{j=1}^{n-1} \left[\mathbf{b}^{[2]}\right]_j$ 和

$$\sum_{j=1}^{r} [\mathbf{z}^{[2]}]_j \ge \sum_{j=1}^{r} [\mathbf{b}^{[2]}]_j \quad \text{for all } r \in \{1, 2, \ldots, n-1\}, \tag{A.8}$$

其中，$m^{[2]}$ 表示从$\mathbf{a}^{[2]}$ 中移除零度节点后剩余节点的数量，而向量$\mathbf{z}^{[2]}$ 由 $\mathscr{Z}\left(\mathbf{a}^{[2]}、|\mathbf{b}^{[2]}|\right)$ 根据（2）得到；更新后的度序列$\mathbf{a}^{[2]}$ 和$\mathbf{b}^{[2]}$ 由给定的 $\mathscr{F}$（1）以及集合 $\{\mathscr{F}(k)\}_{k=2}^{p}$诱导得到。总度数相等的条件 $\sum_{i=1}^{m^{[2]}} \left[\mathbf{a}^{[2]}\right]_i = \sum_{j=1}^{n-1} \left[\mathbf{b}^{[2]}\right]_j$自动成立，因为初始度序列满足$\sum_{i=1}^{m} [\mathbf{a}]_i = \sum_{j=1}^{n} [\mathbf{b}]_j$，且每次连接节点 $[\mathbf{b}]_1$ 都会同时使$\mathbf{a}$和$\mathbf{b}$的总度数减少1。因此，只需关注累积不等式条件（A.8）。

根据引理 A.2 证明中的 (A.3)，可知对于任意满足 (A.6) 的给定 $\mathscr{F}(1)$，在所有满足 (A.7) 的集合 $\{\mathscr{F}(k)\}_{k=2}^{p}$ 中，按照贪心反向策略生成的集合——即按度数降序，将节点 $[\mathbf{b}]_1$ 的剩余度依次分配给组p、$p-1$、……、2，并在转向度数较低的组之前先将每个度数较高的组分配完——使所有 $r \in \{1, 2, \ldots, n-1\}$的和 $\sum_{j=1}^{r} \left[\mathbf{z}^{[2]}\right]_j$最大化。由于对于所有此类集合，条件 (A.8) 的右侧相同，且该条件关于这些$\widetilde{\ }$和单调递增，因此，对于给定的 $\mathscr{F}(1)$，存在一个可行集合 $\{\mathscr{F}(k)\}_{k=2}^{p}$，使所得 $\mathbf{z}^{[2]}$ 满足 (A.8)，当且仅当贪心反向策略所得向量 $\mathbf{z}^{[2]}$ 满足 (A.8)。因此，在满足 (A.6) 的 $\mathscr{F}$(1) 的可行取值中，存在此类可行集合的取值，恰好就是相应的贪心反向策略向量 $\mathbf{z}^{[2]}$ 满足 (A.8) 的那些取值。

因此，我们可以通过分析向量 $\mathscr{F}(1)$ 满足条件 (A.6) 时，条件 (A.8) 的情况，推导出其可行取值。该向量为 $\mathbf{z}^{[2]},$，其中 $\mathbf{z}^{[2]}$ 由 $\mathscr{L}\left(\mathbf{a}^{[2]}、|\mathbf{b}^{[2]}|\right)$ 根据 (2) 得出；更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 由 $\mathscr{F}(1)$ 与通过贪心反向策略生成的集合 $\{\mathscr{F}(k)\}_{k=2}^{p}$ 共同确定。

根据 $\mathscr{D}$-运算（定义 2），条件（A.8）等价于

$$\sum_{h=1}^{r}\left[\mathscr{D}\left(\mathbf{z}^{[2]},\mathbf{b}^{[2]}\right)\right]_h \geq 0 \quad \text{for all } r\in\{1,2,\ldots,n-1\}. \tag{A.9}$$

为便于推导 $\mathscr{F}(1)$ 的可行取值，我们定义以下三个 $\mathscr{D}$-向量：

- $D_1=\mathscr{Q}\left(\mathbf{z}^{[2]},\mathbf{b}^{[2]}\right)\in\mathbb{Z}^{n-1}$。这里，向量 $\mathbf{z}^{[2]}$ 由 $\mathscr{L}\left(\mathbf{a}^{[2]}、|\mathbf{b}^{[2]}|\right)$ 根据（2）得到，其更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 由满足（ $\mathscr{F}(1)$ 和条件（A.6）的 $\{\mathscr{F}(k)\}_{k=2}^{p}$ 通过贪心反向策略生成。

—

- $D_2=\mathscr{D}\left(\mathbf{z}(1),\mathbf{b}^{[2]}\right)\in\mathbb{Z}^{n}$。这里，向量 $\mathbf{z}(1)=\mathbf{z}$ 由 $\mathscr{L}(\mathbf{a}\ |\mathbf{b}|)$ 根据（2）得到，其度序列 $\mathbf{a}$ 和 $\mathbf{b}$ 为初始度序列。向量 $\mathbf{z}(1)$ 对应于初始状态，此时节点$[\mathbf{b}]_1$ 尚未与 $\mathbf{a}$ 的第 1 组中的任何节点相连。此 $\mathscr{D}$-向量 D_2 用于根据（5）和（6）定义以下量 $D(1)$ left、 $D(1)$now 和 $D(1)$ right；这些量均可在初始状态下确定，并用于刻画 $\mathscr{F}(1)$ 的可行取值：

$$D_{\text{left}}^{(1)}=\sum_{h=1}^{\alpha_1-1}[D_2]_h\,,\quad D_{\text{now}}^{(1)}=[D_2]_{\alpha_1}\,, \tag{A.10}$$

and

$$D_{\text{right}}^{(1)}=\min\left\{\min_{r\in\{\alpha_1+1,\alpha_1+2,\ldots,n\}}\sum_{h=\alpha_1+1}^{r}[D_2]_h\,,0\right\}, \tag{A.11}$$

其中 α_1 表示$\mathbf{a}$的第 1 组节点的度。

- $D_3=\mathscr{D}\left(\widehat{\ }\mathbf{z}^{[2]},\mathbf{b}^{[2]}\right)\in\mathbb{Z}^{n-1}$。这里，向量 $\widehat{\ }\mathbf{z}^{[2]}$ 由 $\mathscr{L}\left(\widehat{\ }\mathbf{a}^{[2]}、|\mathbf{b}^{[2]}|\right)$ 根据式 (2) 得到，其中更新后的度序列 $\widehat{\ }\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 由在标准贪心算法下生成的集合 $\{\widehat{\mathscr{F}}(k)\}_{k=1}^{p}$ 导出；具体而言，按照度数降序，将节点 $[\mathbf{b}]_1$ 的度依次分配给组p、$p-1$、.....、1，并在转向度数较低的组之前，先将每个度数较高的组分配完毕。根据引理 A.2，标准贪心算法（即 1 步贪心算法）得到的更新后的度序列 $\widehat{\ }\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 仍满足二分图生成的可行性。由于条件 (A.9) 是更新后的度序列保持二分图生成可行性的必要条件，因此我们有

$$\sum_{h=1}^{r}[D_3]_h=\sum_{h=1}^{r}\left[\mathscr{D}\left(\widehat{\mathbf{z}}^{[2]},\mathbf{b}^{[2]}\right)\right]_h\geq 0\quad \text{for all } r\in\{1,2,\ldots,n-1\}. \tag{A.12}$$

三个 $\mathscr{D}$-向量之间逐分量的关系如下所示。根据备注 1，量 $\mathscr{F}(k)$ 也表示 α_k 的第 $\mathbf{z}$ 个分量的减少。这种

减少确定了更新向量的第 α_k 个分量$\mathbf{z}^{[2]}$, ，而后者又确定了相应 $\mathscr{D}$-向量 $\mathscr{D}\left(\mathbf{z}^{[2]}, \mathbf{b}^{[2]}\right)$ 的第 α_k 个分量。由于由$\{\mathscr{F}(k)\}_{k=1}^{p}$引起的减少从第 α_1 个分量开始，因此前 α_1-1 个分量不受影响。因此，对于三个 $\mathscr{D}$-向量 D_1、 D_2 和 D_3，它们的前 α_1-1 个分量相同，由此得到

$$\sum_{h=1}^{r}[D_1]_h=\sum_{h=1}^{r}[D_2]_h=\sum_{h=1}^{r}[D_3]_h \quad \text{for all } r\in\{1,2,\ldots,\alpha_1-1\}. \tag{A.13}$$

考虑第 α_1个分量，即 $\mathscr{D}$向量 D_1 与 D_2。对于 D_1，我们注意到量$\mathscr{F}(1)=\mathscr{F}(1)$，这会引起 $\mathscr{F}(1)$在第 α_1个分量上的减少，而 $\mathscr{F}(1)=0$对于D_2。因此，相对于第 α_1个分量（ D_2的）， D_1 的相应分量减少了 $\mathscr{F}(1)$。将其与（A.10）结合，我们得到

$$[D_1]_{\alpha_1}=[D_2]_{\alpha_1}-\overline{\mathscr{F}}(1)=D^{(1)}_{\text{now}}-\overline{\mathscr{F}}(1). \tag{A.14}$$

考虑 $\mathscr{D}$向量 D_1 与D_2的其余分量（不包括前 α_1 个分量）。对于 D_1，在贪心反向策略下针对 $k\in\{2,3,\ldots,p\}$生成的量 $\overline{\mathscr{F}}(k)$会在相应的第 α_k个分量中引起 $\mathscr{F}(k)$的减少，而 $\mathscr{F}(k)=0$对于 D_2。因此，相对于 D_2的第 α_k个分量，对每一个 $k\in\{2,3,\ldots,p\}$， D_1 的相应分量减少了 $\mathscr{F}(k)$。由此，对所有 $h\in\{\alpha_1+1,\alpha_1+2,\ldots,n-1\}$， D_1的第 h个分量小于或等于 D_2的相应分量。于是，我们得到

$$\sum_{h=\alpha_1+1}^{r}[D_1]_h\le\sum_{h=\alpha_1+1}^{r}[D_2]_h \quad \text{for all } r\in\{\alpha_1+1,\alpha_1+2,\ldots,n-1\}. \tag{A.15}$$

基于上文定义的三个 $\mathscr{D}$向量及其在(A.13)–(A.15)中建立的逐分量关系，我们现在推导 $\mathscr{F}(1)$满足(A.6)的可行取值，方法是分析条件(A.9)，该条件针对$\mathscr{D}$向量 D_1:

$$\sum_{h=1}^{r}[D_1]_h\ge 0 \quad \text{for all } r\in\{1,2,\ldots,n-1\}. \tag{A.16}$$

将(A.12)与(A.13)结合，可得

$$\sum_{h=1}^{r}[D_1]_h=\sum_{h=1}^{r}[D_3]_h\ge 0 \quad \text{for all } r\in\{1,2,\ldots,\alpha_1-1\}.$$

因此， 只需关注条件(A.16)，其对所有 $r\in\{\alpha_1,\alpha_1+1,\ldots,n-1\}$成立，即，

$$\sum_{h=1}^{r}[D_1]_h\ge 0 \quad \text{for all } r\in\{\alpha_1,\alpha_1+1,\ldots,n-1\}.$$

我们将上述条件分为两种情形： $r=\alpha_1$和 $r\in\{\alpha_1+1,\alpha_1+2,\ldots,n-1\}$，即

$$\begin{cases}\sum_{h=1}^{\alpha_1-1}[D_1]_h+[D_1]_{\alpha_1}\ge 0, & \text{for } r=\alpha_1,\\ \sum_{h=1}^{\alpha_1-1}[D_1]_h+[D_1]_{\alpha_1}+\sum_{h=\alpha_1+1}^{r}[D_1]_h\ge 0, & \text{for all } r\in\{\alpha_1+1,\alpha_1+2,\ldots,n-1\}.\end{cases} \tag{A.17}$$

由(A.10)和(A.13)，我们得到$\sum_{h=1}^{\alpha_1-1}[D_1]_h=\sum_{h=1}^{\alpha_1-1}[D_2]_h=D^{(1)}$ 左边。结合此式与(A.14)，条件(A.17)等价于

$$\begin{cases} D_{\text{left}}^{(1)}+D_{\text{now}}^{(1)}-\overline{\mathscr{F}}(1)\geq 0, & \text{for } r=\alpha_1, \\ D_{\text{left}}^{(1)}+D_{\text{now}}^{(1)}-\overline{\mathscr{F}}(1)+\sum_{h=\alpha_1+1}^{r}[D_1]_h\geq 0, & \text{for all } r\in\{\alpha_1+1,\alpha_1+2,\ldots,n-1\}. \end{cases} \tag{A.18}$$

为进一步分析条件（A.18），我们定义 μ 如下。令 μ 为使$\sum_{h=\alpha_1+1}^{r}[D_1]_h$在 $r\in\{\alpha_1+1,\alpha_1+2,\ldots,n-1\}$上取得最小值的最小索引，即

$$\mu=\min\left\{r\,\middle|\,r\in\arg\min_{r}\sum_{h=\alpha_1+1}^{r}[D_1]_h\right\},\quad \text{for all } r\in\{\alpha_1+1,\alpha_1+2,\ldots,n-1\}.$$

如果 $\sum_{h=\alpha_1+1}^{\mu}[D_1]_h>0$，那么对于所有 $r\in\{\alpha_1+1,\alpha_1+2,\ldots,n-1\}$，都有 $\sum_{h=\alpha_1+1}^{r}[D_1]_h>0$。在这种情况下，条件（A.18）化简为

$$D_{\text{left}}^{(1)}+D_{\text{now}}^{(1)}-\overline{\mathscr{F}}(1)\geq 0.$$

另一方面，如果 $\sum_{h=\alpha_1+1}^{\mu}[D_1]_h\leq 0$，那么（A.18）中限制最严格的不等式在 $r=\mu$处取到。在这种情况下，条件（A.18）化简为

$$D_{\text{left}}^{(1)}+D_{\text{now}}^{(1)}-\overline{\mathscr{F}}(1)+\sum_{h=\alpha_1+1}^{\mu}[D_1]_h\geq 0.$$

因此，条件（A.18）等价于

$$\begin{cases} D_{\text{left}}^{(1)}+D_{\text{now}}^{(1)}-\overline{\mathscr{F}}(1)\geq 0, & \text{if } \sum_{h=\alpha_1+1}^{\mu}[D_1]_h>0, \\ D_{\text{left}}^{(1)}+D_{\text{now}}^{(1)}-\overline{\mathscr{F}}(1)+\sum_{h=\alpha_1+1}^{\mu}[D_1]_h\geq 0, & \text{if } \sum_{h=\alpha_1+1}^{\mu}[D_1]_h\leq 0. \end{cases} \tag{A.19}$$

我们现在推导 $\mathscr{F}$（1）的可行取值，¯方法是分析（A.19）中的两种情形。为此，我们将 ν定义为$\sum_{h=\alpha_1+1}^{r}[D_2]_h$在 $r\in\{\alpha_1+1,\alpha_1+2,\ldots\ldots,n-1\}$上取得其最小值的最小索引，即

$$\nu=\min\left\{r\,\middle|\,r\in\arg\min_{r}\sum_{h=\alpha_1+1}^{r}[D_2]_h\right\},\quad \text{for all } r\in\{\alpha_1+1,\alpha_1+2,\ldots,n-1\}.$$

- $\sum_{h=\alpha_1+1}^{\mu}[D_1]_h>0$。

由（A.15），我们有 $\sum_{h=\alpha_1+1}^{\nu}[D_2]_h\geq\sum_{h=\alpha_1+1}^{\nu}[D_1]_h$。将其与$\sum_{h=\alpha_1+1}^{\mu}[D_1]_h>0$这一事实相结合，我们得到

$$\sum_{h=\alpha_1+1}^{\nu}[D_2]_h\geq\sum_{h=\alpha_1+1}^{\nu}[D_1]_h\geq\sum_{h=\alpha_1+1}^{\mu}[D_1]_h>0.$$

根据 ν 的定义以及 $[D_2]_n=\left[\mathscr{D}\left(\mathbf{z}^{(1)},\mathbf{b}^{[2]}\right)\right]_n=\left[\mathbf{z}^{(1)}\right]_n\geq 0$，我们有 $\min_r\sum_{h=\alpha_1+1}^{r}[D_2]_h=\sum_{h=\alpha_1+1}^{\nu}[D_2]_h$。结合（A.11）以及$\sum_{h=\alpha_1+1}^{\nu}[D_2]_h>0$，可得

$$D_{\text{right}}^{(1)}=\min\left\{\min_{r}\sum_{h=\alpha_1+1}^{r}[D_2]_h,0\right\}=\min\left\{\sum_{h=\alpha_1+1}^{\nu}[D_2]_h,0\right\}=0.$$

因此，此处的条件（A.19）等价于

$$\overline{\mathscr{F}}(1) \le D^{(1)}_{\text{left}} + D^{(1)}_{\text{now}} = D^{(1)}_{\text{left}} + D^{(1)}_{\text{now}} + D^{(1)}_{\text{right}}.$$

- $\sum_{h=\alpha_1+1}^{\mu} [D_1]_h \le 0$。

根据（A.15）以及如下事实：$\sum_{h=\alpha_1+1}^{\mu} [D_1]_h \le \sum_{h=\alpha_1+1}^{\nu} [D_1]_h$，可得

$$\sum_{h=\alpha_1+1}^{\mu} [D_1]_h \le \sum_{h=\alpha_1+1}^{\nu} [D_1]_h \le \sum_{h=\alpha_1+1}^{\nu} [D_2]_h . \tag{A.20}$$

根据（A.11）以及$\min_r \sum_{h=\alpha_1+1}^{r} [D_2]_h = \sum_{h=\alpha_1+1}^{\nu} [D_2]_h$这一事实，我们得到

$$D^{(1)}_{\text{right}} = \min\left\{ \min_r \sum_{h=\alpha_1+1}^{r} [D_2]_h , 0 \right\} = \min\left\{ \sum_{h=\alpha_1+1}^{\nu} [D_2]_h , 0 \right\}. \tag{A.21}$$

基于（A.20）、（A.21）以及$\sum_{h=\alpha_1+1}^{\mu} [D_1]_h \le 0$这一事实，我们有$\sum_{h=\alpha_1+1}^{\mu} [D_1]_h \le D^{(1)}$ 成立。因此，条件（A.19）在此情形下等价于

$$\overline{\mathscr{F}}(1) \le D^{(1)}_{\text{left}} + D^{(1)}_{\text{now}} + \sum_{h=\alpha_1+1}^{\mu} [D_1]_h \le D^{(1)}_{\text{left}} + D^{(1)}_{\text{now}} + D^{(1)}_{\text{right}}.$$

基于对 上述两种情况的分析，我们有条件（A.19）等价于

$$\overline{\mathscr{F}}(1) \le D^{(1)}_{\text{left}} + D^{(1)}_{\text{now}} + D^{(1)}_{\text{right}}.$$

将此不等式与（A.6）结合，得到

$$\overline{\mathscr{F}}(1) \in \left[\max\left\{ [\mathbf{b}]^{(1)}_1 - M^{(1)}, 0 \right\}, \min\left\{ m_1, [\mathbf{b}]^{(1)}_1, D^{(1)}_{\text{left}} + D^{(1)}_{\text{now}} + D^{(1)}_{\text{right}} \right\} \right]. \tag{A.22}$$

以下分析表明 $D^{(1)}_{\text{左}} + D^{(1)}_{\text{now}} + D^{(1)}_{\text{右}} \le [\mathbf{b}]^{(1)}_1$，这简化了（A.22）中的上界。根据（$D^{(1)}_{\text{右}}$在（A.11）中的定义，我们有

$$D^{(1)}_{\text{right}} = \min\left\{ \min_r \sum_{h=\alpha_1+1}^{r} [D_2]_h , 0 \right\} \le \sum_{h=\alpha_1+1}^{n} [D_2]_h .$$

将此结果与（$D^{(1)}_{\text{左}}$和 $D^{(1)}_{\text{now}}$ 在（A.10）中的定义结合，得到

$$D^{(1)}_{\text{left}} + D^{(1)}_{\text{now}} + D^{(1)}_{\text{right}} \le \sum_{h=1}^{n} [D_2]_h . \tag{A.23}$$

回顾一下，$D_2 = \mathscr{D}\left(\mathbf{z}^{(1)}, \mathbf{b}^{[2]}\right)$。将其与 $\mathscr{D}$-运算（定义2）结合起来，可得

$$\sum_{h=1}^{n} [D_2]_h = \sum_{h=1}^{n} \mathscr{D}\left(\mathbf{z}^{(1)}, \mathbf{b}^{[2]}\right) = \sum_{h=1}^{n} \left[\mathbf{z}^{(1)}\right]_h - \sum_{h=1}^{n-1} \left[\mathbf{b}^{[2]}\right]_h = \sum_{h=1}^{m} [\mathbf{a}]_h - \sum_{h=1}^{n-1} \left[\mathbf{b}^{[2]}\right]_h = [\mathbf{b}]^{(1)}_1 . \tag{A.24}$$

根据（A.23）和（A.24），可得

$$D_{\text{left}}^{(1)} + D_{\text{now}}^{(1)} + D_{\text{right}}^{(1)} \le \sum_{h=1}^{n} [D_2]_h = [\mathbf{b}]_1^{(1)}.$$

将此不等式与（A.22）结合，得到

$$\overline{\mathscr{F}}(1) \in \left[\max\left\{[\mathbf{b}]_1^{(1)} - M^{(1)}, 0\right\}, \min\left\{m_1, D_{\text{left}}^{(1)} + D_{\text{now}}^{(1)} + D_{\text{right}}^{(1)}\right\}\right].$$

因此，假设给定度序列**a** 和**b**存在二分图，则在节点 $[\mathbf{b}]_1$ 与第 **1** 组**a**中的 $\mathscr{F}$**(1)** 个节点连接后，若且唯若满足所述区间条件，所得更新后的度序列仍可用于二分图生成。由此，我们完成了该引理的证明。□

归纳步骤： $k \in \{2,3,\ldots,p\}$。固定任意 $k \in \{2,3,\ldots,p\}$，并假设对所有$k' \in \{1,2,\ldots,k-1\}$，$\mathcal{P}(k')$ 均成立。我们将在以下引理中证明 $\mathcal{P}(k)$。

Lemma A.4. 假设给定度序列**a** 和**b**存在二分图，并且在节点 $[\mathbf{b}]_1$ 与1 至 $k-1$ 组的**a**中各节点连接后，所得更新后的度序列仍可用于二分图生成。那么，在节点 $[\mathbf{b}]_1$与第 k 组中的 $\mathscr{F}(k)$ 个节点连接后，若且唯若连接节点的数量 $\mathscr{F}(k)$ 满足

$$\mathscr{F}(k) \in \left[\max\left\{[\mathbf{b}]_1^{(k)} - M^{(k)}, 0\right\}, \min\left\{m_k, D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}\right\}\right].$$

引理的证明 *A.4*. 根据归纳假设，存在一个可行集合$\sim\{\mathscr{F}(k')\}_{k'=1}^{k-1}$，满足 $\mathcal{P}(k')$ 为所有 $k' \in \{1,2,\ldots,k-1\}$规定的区间条件，使得节点 $[\mathbf{b}]_1$ 连接到组 **1** 至 $k-1$ 中的节点后，所得更新后的度序列仍然可行 **a**。证明 $\mathcal{P}(k)$ 遵循与基例相同的两步结构。

我们首先推导 $\mathscr{F}$**(**k**)** 的一个区间，在该区间内存在一个整数集合 $\{\mathscr{F}(k')\}_{k'=k+1}^{p}$，其满足 $0 \le \mathscr{F}(k') \le m_{k'}$，使得节点 $[\mathbf{b}]_1$ 的剩余度可以分配到 **a** 的其余各组中。根据基例中的相同分析，我们得到

$$\mathscr{F}(k) \in \left[\max\left\{[\mathbf{b}]_1^{(k)} - M^{(k)}, 0\right\}, \min\left\{m_k, [\mathbf{b}]_1^{(k)}\right\}\right]. \tag{A.25}$$

接下来，我们刻画 **(A.25)** 中满足以下条件的区间子集：存在一个可行的集合，$\widetilde{\ }\{\mathscr{F}(k')\}_{k'=k+1}^{p}$ 使得所得的更新后的度序列 $\mathbf{a}^{[2]}$ 和 $\mathbf{b}^{[2]}$ 仍然可行。应用引理 1 以及 k步贪心算法，并对处理完第 **1** 组到 $k-1$组后的更新度序列进行分析，采用与基例相同的方法，可得

$$\mathscr{F}(k) \le D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}.$$

将此不等式与 (A.25) 结合，可得

$$\mathscr{F}(k) \in \left[\max\left\{[\mathbf{b}]_1^{(k)} - M^{(k)}, 0\right\}, \min\left\{m_k, [\mathbf{b}]_1^{(k)}, D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}\right\}\right]. \tag{A.26}$$

根据基例中的相同分析，我们进一步得到 $D^{(k)}{}_{\text{左}} + D_{\text{now}}^{(k)} + D^{(k)}{}_{\text{右}} \le [\mathbf{b}]_1^{(k)}$。将此不等式与（A.26）结合，可得

$$\mathscr{F}(k) \in \left[\max\left\{[\mathbf{b}]_1^{(k)} - M^{(k)}, 0\right\}, \min\left\{m_k, D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}\right\}\right].$$

因此，在归纳假设下，当 $[\mathbf{b}]_1$ 连接到组 k 中的 $\mathscr{F}(k)$个节点后，当且仅当所述区间条件成立时，所得的更新后的度序列仍然可行。因此，引理的证明得证。□

结论。 根据数学归纳法， $\mathcal{P}(k)$对所有 $k \in \{1, 2, \ldots, p\}$均成立。因此，在节点 $[\mathbf{b}]_1$ 与<a>的第 1 组至第 p 组中的节点连接后，所得的更新后的度序列$\mathbf{a}^{[2]}$ 和$\mathbf{b}^{[2]}$ 在且仅在对于<a>的每个组k，连接节点的数量 $\mathscr{F}(k)$满足以下条件时，才适用于二分图生成

$$\mathscr{F}(k) \in \left[\max\left\{[\mathbf{b}]_1^{(k)} - M^{(k)}, 0\right\}, \min\left\{m_k, D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}\right\}\right].$$

因此，我们得出定理的证明。□

EC.3. 命题的证明1

证明。 (8) 可由以下事实得出： $\mathscr{F}(k) \in [\ell_k, u_k]$ 以及 $\mathscr{F}_i(k)$ 的定义。(9) 可由以下事实得出：当节点 $[\mathbf{b}]_j$ 完成了与组 k 中节点的连接后，流程会在 $\mathbf{a}^{[j]}$ 中移至第 $k+1$ 组；而当 $\mathbf{b}]_j$ 完成了与组 p中节点的连接后，流程会在 $\mathbf{a}^{[j+1]}$ 中移至第 1 组。因此，我们得出命题的证明。□

EC.4. EBGS 算法伪代码

EBGS 算法的伪代码见算法A.3。

算法 A.3 高效二分图采样（EBGS）算法

输入： 度序列 $\mathbf{a} \in \mathbb{N}_+^m$和 $\mathbf{b} \in \mathbb{N}_+^n$，一个初始为空的二分图 $\mathbf{G(a,b)}$
输出： 二分图 $\mathbf{G(a,\ b)}$

对于 $j=1$ **到** n **执行**
  将组 $\mathbf{a}^{[j]}$划分为 p 组（α_k，m_k），每组包含 $\alpha_1 < \alpha_2 < \cdots < \alpha_p$
  对于 $k=1$ **至** p，**执行以下操作**
    通过 (via (7) 计算可行条件 $[\ell_k, u_k]$
    获取近似权重向量$\mathbf{w}^{\text{EB}}$ via (11)
    按照$\mathbf{w}^{\text{EB}}$抽取样本 $\mathscr{F}(k) \in [\ell_k, u_k]$
    在组 k中均匀选取 $\mathscr{F}(k)$ 个不同节点，并将它们连接到 $[\mathbf{b}]_j$，位于 $\mathbf{G(a,b)}$
返回 G(**a**,**b**)

EC.5. EBGS 算法的重要性抽样

当目标是计算关于 $\mathcal{G}(\mathbf{a},\mathbf{b})$ 上均匀分布的期望时，我们采用重要性抽样来校正 EBGS 算法所引起的非均匀性，即，

$$\theta \triangleq \mathbb{E}_{\mathbf{w}^{\mathrm{UB}}}\Big[f\big(\mathbf{G}(\mathbf{a},\mathbf{b})\big)\Big],$$

其中，$f(\cdot)$ 是我们所关注的函数。关于均匀分布计算期望在实践中有许多应用。例如，正如 Glasserman 和 Young（2016）所讨论的，在银行间网络的背景下，可以在每个抽样网络上评估各种系统性风险度量 $f(\cdot)$，然后对多次抽样结果取平均，从而获得$\mathcal{G}$（$\mathbf{a}$、$\mathbf{b}$）上的预期风险度量。

具体而言，与$\mathbf{G}$（$\mathbf{a}$，$\mathbf{b}$）$\in\mathcal{G}$（$\mathbf{a}$，$\mathbf{b}$）相关联的重要性权重为

$$w_{\mathbf{G}} = \frac{1}{|\mathcal{G}(\mathbf{a},\mathbf{b})|\, p_{\mathbf{G}}},$$

其中，$p_{\mathbf{G}}$ 是 EBGS 算法下$\mathbf{G}$（$\mathbf{a}$，$\mathbf{b}$）的抽样概率，表示为

$$p_{\mathbf{G}} = \prod_{j=1}^{n}\prod_{k=1}^{p} \frac{[\mathbf{w}^{\mathrm{EB}}]_i}{\left(\sum_{s=1}^{\mu_k-\ell_k+1}[\mathbf{w}^{\mathrm{EB}}]_s\right)\cdot\binom{m_k}{\mathscr{F}_i(k)}}.$$

该 $p_{\mathbf{G}}$ 表达式源于以下事实：归一化权重表示在区间内选择 $\mathscr{F}_i(k)$ 的概率，而因子 $1/\binom{m_k}{\mathscr{F}_i(k)}$ 则表示从 $\binom{m_k}{\mathscr{F}_i(k)}$ 个可能配置中均匀选择一个配置。因此，为使用 EBGS 算法估计 θ，我们生成独立样本 $\mathbf{G}_1$（$\mathbf{a}$，$\mathbf{b}$）.....$\mathbf{G}_N$（$\mathbf{a}$，$\mathbf{b}$），并构造估计量

$$\widetilde{\theta} = \frac{1}{N}\sum_{t=1}^{N} w_{\mathbf{G}} f\big(\mathbf{G}_t(\mathbf{a},\mathbf{b})\big).$$

然而，在大规模问题中，由于资源限制，计算 $|\mathcal{G}(\mathbf{a}，\mathbf{b})|$ 命题 1 中的量可能不可行。在这种情况下，可以采用一种修正后的估计量（如 Blitzstein 和 Diaconis（2011）所述）：

$$\widehat{\theta} = \frac{\sum_{t=1}^{N} w_{\mathbf{G}} f\big(\mathbf{G}_t(\mathbf{a},\mathbf{b})\big)}{\sum_{t=1}^{N} w_{\mathbf{G}}},$$

其中，未知因子 $|\mathcal{G}(\mathbf{a}，\mathbf{b})|$ 会相互抵消。作为比率估计量，θ 存在偏差，但随着$N\to\infty$（Glasserman 和 de Larrea（2023）所示）而一致。

EC.6. 有向图情形的技术细节

本节汇总了有向图情形下的技术细节，包括自环排除约束、主要理论结果的证明，以及所提出的有向图算法的伪代码。

EC.6.1. 有向图的自环排除约束

为排除自环，每个 $[\mathbf{b}]_j$ 都不得与 $\mathbf{a}^{[j]}$中的对应节点相连。具体而言，为方便记号，令 $[\mathbf{a}]_j$ 表示 $\mathbf{a}^{[j]}$中 $[\mathbf{b}]_j$ 的对应节点，并令 k_j 表示包含 $[\mathbf{a}]_j$的组的索引。由于 $[\mathbf{b}]_j$ 不能与 $[\mathbf{a}]_j$相连，因此 $\mathscr{F}(k_j)$ 的下界（记为 ℓ_{k_j}）必须满足 $\ell_{k_j} \le m_{k_j}-1$，其中 m_{k_j}表示组 k_j中的节点数。否则，若 $\ell_{k_j}=m_{k_j}$，则 $[\mathbf{b}]_j$将被迫与组 k_j中的所有节点相连，包括 $[\mathbf{a}]_j$，从而产生自环。根据定理 1，

$$\ell_{k_j}=\max\left\{[\mathbf{b}]_j^{(k_j)}-M^{(k_j)},0\right\}.$$

因此，自环排除约束可表示为

$$\begin{aligned}[\mathbf{b}]_j^{(k_j)}-M^{(k_j)}&\le m_{k_j}-1,\\ [\mathbf{b}]_j-\Delta-M^{(k_j)}&\le m_{k_j}-1,\\ [\mathbf{b}]_j-M^{(k_j)}-m_{k_j}+1&\le \Delta,\end{aligned}$$

其中，$\Delta=\sum_{k=1}^{k_j-1}\mathscr{F}(k)$ 表示前 k_j-1 组中已与 $[\mathbf{b}]_j$相连的$\mathbf{a}^{[j]}$ 的节点总数。原则上，可以尝试在不等式不成立时，通过追溯调整 Δ 来强制满足上述不等式。然而，这样做需要回溯先前构造的部分连接，从而导致严重的计算低效。

EC.6.2. 定理2的证明

证明。对于增广度序列，由于 $[\mathbf{b}]_j$ 必须与 $[\mathbf{a}]_j$相连，因此 $\mathscr{F}(k_j)$ 的上界（记为 u_{k_j}）必须满足 $u_{k_j}\ge 1$。否则，若 $u_{k_j}=0$，则会迫使 $[\mathbf{b}]_j$ 无法与组 k_j 中的任何节点相连，包括 $[\mathbf{a}]_j$，从而避免形成自环。回顾定理 1 可知，$u_{k_j}=\min\left\{D^{(k_j)\,\text{left}}+D^{(k_j)\,\text{now}}+D^{(k_j)\,\text{right}}\ m_{k_j}\right\}$，因此自环连接约束可表示为

$$\begin{aligned}D_{\text{left}}^{(k_j)}+D_{\text{now}}^{(k_j)}+D_{\text{right}}^{(k_j)}&\ge 1,\\ D_{\text{left}}^{(1,k_j)}-\Delta+D_{\text{now}}^{(1,k_j)}+D_{\text{right}}^{(1,k_j)}&\ge 1,\\ D_{\text{left}}^{(1,k_j)}+D_{\text{now}}^{(1,k_j)}+D_{\text{right}}^{(1,k_j)}-1&\ge \Delta,\end{aligned}\tag{A.27}$$

其中，$\Delta=\sum_{k=1}^{k_j-1}\mathscr{F}(k)$表示前 k_j-1 组中已与$\mathbf{a}^{[j]}$ 相连的节点总数，以及与 $[\mathbf{b}]_j$相连的节点总数，

$$D_{\text{left}}^{(1,k_j)}=\sum_{h=1}^{\alpha_{k_j}-1}\left[\mathscr{D}\left(\mathbf{z}^{(1)},\mathbf{b}^{[j+1]}\right)\right]_h,\quad D_{\text{now}}^{(1,k_j)}=\left[\mathscr{D}\left(\mathbf{z}^{(1)},\mathbf{b}^{[j+1]}\right)\right]_{\alpha_{k_j}},$$

and

$$D_{\text{right}}^{(1,k_j)}=\min\left\{\min_r\sum_{h=\alpha_{k_j}+1}^{r}\left[\mathscr{D}\left(\mathbf{z}^{(1)},\mathbf{b}^{[j+1]}\right)\right]_h,0\right\}.$$

第二个不等式由以下事实得出： $D^{(k_j)\text{左}} = D^{(1,k_j)\text{左}} - \Delta$、 $D^{(k_j)\text{现在}} = D^{(1,k_j)\text{现在}}$ 以及 $D^{(k_j)\text{右}} = D^{(1,k_j)\text{右}}$。 $D^{(1,k_j)\text{左}}$ 和 $D^{(k_j)\text{左}}$ 分别考虑 $\mathbf{z}^{(1)}$ 和 $\mathbf{z}^{(k)}$ 的前 $\alpha_{k_j} - 1$ 个分量之和；因此，两者相差 Δ，这表示这些分量从 $\mathbf{z}^{(1)}$ 到 $\mathbf{z}^{(k)}$ 的总减少量。相比之下， $D^{(1,k_j)\text{现在}}$ 和 $D^{(k_j)\text{现在}}$ 考虑第 α_{k_j} 个分量，因此不受前面分量变化的影响，而 $D^{(1,k_j)\text{右}}$ 和 $D^{(k_j)\text{右}}$ 则类似。

为满足增广度序列的自环连接约束，我们首先计算 $u_\Delta = D^{(1,k_j)\text{左侧}} + D^{(1,k_j)\text{当前}} + D^{(1,k_j)\text{右侧}} - 1$ 在 $[\mathbf{b}]_j$ 开始连接 $\mathbf{a}^{[j]}$ 中的节点之前的值。该值表示在前 $k_j - 1$ 个组中， $[\mathbf{b}]_j$ 最多可以连接的节点总数。随后，我们将动态更新的上限 $u_\Delta^{(k)}$ 纳入 $k = 1, \ldots, k_j - 1$ 中每个 $\mathscr{F}(k)$ 的上界，其中 $u_\Delta^{(k)} = u_\Delta - \sum_{k'=1}^{k-1} \mathscr{F}(k')$ 且 $u_\Delta^{(1)} = u_\Delta$。相应地， $k = 1, \ldots, k_j - 1$ 中 $\mathscr{F}(k)$ 的上界修改为

$$u_k = \min\left\{D_{\text{left}}^{(k)} + D_{\text{now}}^{(k)} + D_{\text{right}}^{(k)}, m_k, u_\Delta^{(k)}\right\}. \tag{A.28}$$

注意， $u_\Delta^{(k)}$ 大于或等于 $k = 1, \ldots, k_j - 1$ 的下界 $\ell_k(\mathscr{F})$，即 $u_\Delta^{(k)} \geq \max\left\{[\mathbf{b}]_j^{(k)} - M^{(k)}, 0\right\}$，因为如果 $\ell_k = [\mathbf{b}]_j^{(k)} - M^{(k)}$，那么 $[\mathbf{b}]_j$ 就必须与组 k_j 中的所有节点相连，包括 $[\mathbf{a}]_j$；此外， $u_\Delta^{(k)} \geq 0$ 显然成立。因此，所提出的动态调整保证 $\mathscr{F}(k_j)$ 的上界满足 $u_{k_j} \geq 1$。与此同时， $\mathscr{F}(k_j)$ 的下界更新为

$$\ell_{k_j} = \max\left\{[\mathbf{b}]_j^{(k_j)} - M^{(k_j)}, 1\right\}. \tag{A.29}$$

在增广度序列下，式（A.28）和（A.29）均保证 $[\mathbf{b}]_j$ 与 $[\mathbf{a}]_j$ 形成自环；等价地，在原始度序列下， $[\mathbf{b}]_j$ 不会与 $[\mathbf{a}]_j$ 相连。

为强制执行单度连接约束，我们首先统计第 1 组中尚未形成自环的度为一的节点数量，记为 n_{one}，并将该数量纳入 $\mathscr{F}(1)$ 的上界中，更新后如下所示：

$$u_1 = \begin{cases} \min\left\{D_{\text{left}}^{(1)} + D_{\text{now}}^{(1)} + D_{\text{right}}^{(1)}, m_1 - n_{\text{one}} + 1\right\}, & \text{if } k_j = 1, \\ \min\left\{D_{\text{left}}^{(1)} + D_{\text{now}}^{(1)} + D_{\text{right}}^{(1)}, m_1 - n_{\text{one}}, u_\Delta^{(1)}\right\}, & \text{if } k_j > 1. \end{cases} \tag{A.30}$$

因此，根据（A.28）、（A.29）和（A.30），我们得出该定理。 □

EC.6.3. DGE 算法的伪代码

DGE算法的伪代码见算法A.4。

算法 A.4 有向图枚举（DGE）算法

输入： 度序列 $\mathbf{a} \in \mathbb{N}_+^m$ 和 $\mathbf{b} \in \mathbb{N}_+^n$

输出： 与度序列 $\mathbf{a}$ 和 $\mathbf{b}$ 一致的所有有向图的集合 $\mathcal{G}(\mathbf{a},\mathbf{b})$

初始化 $\mathcal{G}(\mathbf{a},\mathbf{b}) \leftarrow \emptyset$, $\tilde{\mathbf{a}} \leftarrow \mathbf{a}+\mathbf{1}$, $\mathbf{b} \leftarrow \mathbf{b}+\mathbf{1}$, 一个空二分图 $\mathbf{G}(\tilde{\mathbf{a}}, \tilde{\mathbf{b}})$, 两个空的队列 $\mathcal{Q}_{\text{outer}}$ 和 $\mathcal{Q}_{\text{inner}}$
将对 $(\mathbf{G}(\tilde{\mathbf{a}}, \tilde{\mathbf{b}}), 1)$ 转换为 $\mathcal{Q}_{\text{outer}}$ ▷ 节点索引 $j=1$
当 $\mathcal{Q}_{\text{outer}} \neq \emptyset$ **时** ▷ 对 n 节点进行外层广度优先搜索
  将第一个对 $(\mathbf{G}(\tilde{\mathbf{a}}, \tilde{\mathbf{b}}), j)$ 从 $\mathcal{Q}_{\text{outer}}$ 出队，并记录节点索引 j
  如果 $j > n$，**则**
    从 $\mathbf{G}(\tilde{\mathbf{a}}, \tilde{\mathbf{b}})$ 中移除所有自环，以获得与 $\mathbf{a}$ 和 $\mathbf{b}$ 一致的有向图
    将生成的有向图添加到 $\mathcal{G}(\mathbf{a}, \mathbf{b})$
    继续
  将 $\tilde{\mathbf{a}}$ 分成 p 组 (α_k, m_k)，其中包含 $\alpha_1 < \cdots < \alpha_p$
  将对 $(\mathbf{G}(\tilde{\mathbf{a}}, \tilde{\mathbf{b}}), 1)$ 入队到 $\mathcal{Q}_{\text{inner}}$ ▷ 组索引 $k=1$
  当 $\mathcal{Q}_{\text{inner}} \neq \emptyset$ **执行** ▷ 对每个节点在 p 组上进行内层广度优先搜索
    出队第一个对 $(\mathbf{G}(\tilde{\mathbf{a}}, \tilde{\mathbf{b}}), k)$ 从 $\mathcal{Q}_{\text{inner}}$ 并记录组索引 k
    如果 $k > p$ **则**
      如果 $\textsc{Feasible}(\mathbf{G}(\tilde{\mathbf{a}}, \tilde{\mathbf{b}}), j)$ **然后** ▷ 可行性验证步骤
        将对 $(\mathbf{G}(\tilde{\mathbf{a}}, \tilde{\mathbf{b}}), j+1)$ 入队到 $\mathcal{Q}_{\text{outer}}$
      继续
    计算区间 $[\ell_k, u_k]$ 通过 (12)–(14)
    对于 $\mathscr{F}(k) = \ell_k, \ldots, u_k$ **执行**
      如果 $k = k_1$ **则**
        生成所有带自环的 $\binom{m_k}{\mathscr{F}(k)}$ 连接配置
      else 生成所有 $\binom{m_k}{\mathscr{F}(k)}$ 连接配置
      对于 每个连接配置 **执行**
        复制 $\mathbf{G}(\tilde{\mathbf{a}}, \mathbf{b})$，将其记为 $\mathbf{G}'(\tilde{\mathbf{a}}, \tilde{\mathbf{b}})$
        将连接配置追加到 $\mathbf{G}'(\tilde{\mathbf{a}}, \tilde{\mathbf{b}})$
        将对 $(\mathbf{G}'(\tilde{\mathbf{a}}, \tilde{\mathbf{b}}), k+1)$ 入队 放入 $\mathcal{Q}_{\text{inner}}$
返回 $\mathcal{G}(\mathbf{a},\mathbf{b})$

EC.6.4. 命题2的证明

证明。（15）源于以下事实：$\mathscr{F}(k) \in [\ell_k, u_k]$，并且对于 $\mathscr{F}_i(k)$，存在 $\binom{m_k}{\mathscr{F}_i(k)}$ 种连接配置。（16）源于以下事实：当节点 $[\mathbf{b}]_j$ 完成与 k 组中节点的连接后，该过程在 $k < p$ 中移动到 $k+1$ 组；当 $[\mathbf{b}]_j$ 完成与 p 组中节点的连接后，该过程在 $[j+1]$ 中移动到第 1 组。因此，我们得出命题的证明。□

EC.6.5. UDGS 算法伪代码

UDGS 算法的伪代码见算法A.5。

EC.6.6. EDGS 算法伪代码

EDGS 算法的伪代码见算法 A.6。

EC.7. 无向图算法伪代码

Th本节给出了所提出的无向图算法的伪代码 s.

算法 A.5 均匀有向图采样算法（UDGS）

输入：度序列 $\mathbf{a} \in \mathbb{N}_+^m$ 和 $\mathbf{b} \in \mathbb{N}_+^n$ 输出：与 a 和 b 一致的有向图

~1：初始化 $\tilde{\mathbf{a}} \leftarrow \mathbf{a}+\mathbf{1}$、$\mathbf{b} \leftarrow \mathbf{b}+\mathbf{1}$，以及空二分图 $G(\tilde{\mathbf{a}}, \mathbf{b})$ 2：对于 $j=1$ 到 n 执行

3 p α_k m_k $\alpha_1<\alpha_2<\cdots<\alpha_p$：将 $\tilde{\mathbf{a}}^{[j]}$ 分组为若干组 (,),

4 $k=1$ 其中：对于到 p 执行 5：通过 (12)–(14) 计算区间 $[\ell_k, u_k]$

6 $\mathscr{F}$ k $\in[\ell_k, u_k]$：为每个 ( ) 生成

7 UD 18 所有连接配置：通过 ( ) 获取精确权重向量 $\mathbf{w}^{\mathrm{UD}}$ 8：根据 $\mathbf{w}^{\mathrm{UD}}$ ~ 采样一个连接配置 9：在 $G(\tilde{\mathbf{a}}, \mathbf{b})$ ~中为节点 $[\mathbf{b}]_j$ 实现采样得到的连接配置 10：从 $G(\tilde{\mathbf{a}},\mathbf{b})$ 中移除所有自环，以获得与 a 和 b 一致的有向图 11：返回所得有向图

算法 A.6 高效有向图采样算法（EDGS）

输入：度序列 $\mathbf{a} \in \mathbb{N}_+^m$ 和 $\mathbf{b} \in \mathbb{N}_+^n$ 输出：与 a 和 b ~ ~一致的有向图1：初始化 $\tilde{\mathbf{a}} \leftarrow \mathbf{a}+\mathbf{1}$、$\mathbf{b} \leftarrow \mathbf{b}+\mathbf{1}$，以及空二分图 $G(\tilde{\mathbf{a}}, \mathbf{b})$ 2：对于 $j=1$ 到 n，执行

3 p α_k m_k $\alpha_1<\alpha_2<\cdots<\alpha_p$：将 $\tilde{\mathbf{a}}^{[j]}$ 分成若干组 (,)，其中：

4 $k=1$ 对于 到 p，执行 5：通过 (12)–(14) 计算区间 $[\ell_k, u_k]$ 6：通过 (11) 得到近似权重向量 $\mathbf{w}^{\mathrm{ED}}$

7 $\mathscr{F}$ k $\in[\ell_k, u_k]$ ^{ED}：根据 w 对 ( ) 进行抽样：如果 $k=k_j$，则 ~

8

9：将节点 $[\mathbf{b}]_j$ 与其对应节点相连，形成自环 10：在组 k 中均匀选择 k 个互不相同的剩余节点 $\mathscr{F}$ $(k)-1$，并将它们与 $G(\tilde{\mathbf{a}},\mathbf{b})$ 中的节点 $[\mathbf{b}]_j$ 相连 11：否则 ~12：在组中均匀选择 ( ) 个互不相同的节点，并将它们与 $G(\tilde{\mathbf{a}}, \mathbf{b})$ 中的节点 b 相连 ~13：如果不满足 FEASIBLE($G(\tilde{\mathbf{a}}, \mathbf{b})$ j)，则 ▷ 执行可行性验证步骤 14：从 $j=1$ ~ 重新抽样 15：从 $G(\tilde{\mathbf{a}},\mathbf{b})$ 中移除所有自环，得到与 a 和 b 一致的有向图 16：返回所得有向图

EC.7.1. UGE算法的伪代码

UGE算法的伪代码见算法A.7。

EC.7.2. UUGS算法的伪代码

UUGS算法的伪代码见算法A.8。

EC.7.3. EUGS算法的伪代码EUGS算法的伪代码见算法 A.9。

EC.8. 暴力算法的伪代码

蛮力算法的伪代码见算法A.10。

算法 A.7 无向图枚举（UGE）算法

输入： 度序列 $\mathbf{a}=\mathbf{b}\in\mathbb{N}_+^m$
输出： 所有与度序列 $\mathbf{a}$ 和$\mathbf{b}$相一致的无向图集合 $\mathcal{G}(\mathbf{a},\mathbf{b})$

初始化 $\mathcal{G}(\mathbf{a},\mathbf{b})\leftarrow\emptyset$, $\tilde{\mathbf{a}}\leftarrow\mathbf{a}+\mathbf{1}$, $\mathbf{b}\leftarrow\mathbf{b}+\mathbf{1}$, 一个空二分图 $\mathbf{G}(\tilde{\mathbf{a}},\ \tilde{\mathbf{b}})$, 两个空队列 $\mathcal{Q}_{\text{outer}}$ 和 $\mathcal{Q}_{\text{inner}}$
将对 $(\mathbf{G}(\tilde{\mathbf{a}},\ \tilde{\mathbf{b}}),1)$ 至 $\mathcal{Q}_{\text{outer}}$ ▷ 节点索引 $j=1$
当 $\mathcal{Q}_{\text{outer}}\neq\emptyset$ **时** ▷ 对 n 节点进行外层广度优先搜索
    将第一个对 $(\mathbf{G}(\tilde{\mathbf{a}},\ \tilde{\mathbf{b}}),\ j)$ 从 $\mathcal{Q}_{\text{outer}}$ 出队，并记录节点索引 j
    如果 $j>m$，**则**
        从 $\mathbf{G}(\tilde{\mathbf{a}},$ 中移除所有自环 $\tilde{\mathbf{b}})$，以获得满足 $\mathbf{a}$ 和 $\mathbf{b}$ 的无向图
        将生成的无向图添加到 $\mathcal{G}$（$\mathbf{a}$，$\mathbf{b}$）
        继续
    查找一个索引 $j^\star$，其对应的节点具有最大的剩余度 $\tilde{\mathbf{b}}^{[j]}$
    交换 j 和 $j^\star$ 中的索引 $\tilde{\mathbf{b}}^{[j]}$
    将 $\tilde{\mathbf{a}}^{[j]}$ 分成 p 组 $(\alpha_k,\ m_k)$，其中有 $\alpha_1<\cdots<\alpha_p$
    将这对 $(\mathbf{G}(\tilde{\mathbf{a}},\ ),\ \tilde{\mathbf{b}}),\ 1)$ 放入 $\mathcal{Q}_{\text{inner}}$ ▷ 组索引 $k=1$
    当 $\mathcal{Q}_{\text{inner}}\neq\emptyset$ **时** ▷对每个节点，在 p 组上执行内层广度优先搜索
        将第一个对 $(\mathbf{G}(\tilde{\mathbf{a}},\ \tilde{\mathbf{b}}),\ k)$ 出队，并记录组索引 k（来自 $\mathcal{Q}_{\text{inner}}$）
        如果 $k>p$ **则**
            为节点 $[\tilde{\mathbf{a}}]_j$ 建立与 [相对应的对称连接 $\tilde{\mathbf{b}}]_j$
            若 FEASIBLE($\mathbf{G}(\tilde{\mathbf{a}},\ \tilde{\mathbf{b}}),\ j$) **则** ▷ 可行性验证步骤
                将该对 $(\mathbf{G}(\tilde{\mathbf{a}},$入队 $\tilde{\mathbf{b}}),\ j+1)$ 转换为 $\mathcal{Q}_{\text{outer}}$
            继续
        计算区间 $[\ell_k,u_k]$ 通过 (12)–(14)
        对于 $\mathscr{F}(k)=\ell_k,\ldots,u_k$ **执行**
            如果 $k=k_j$ **then**
                生成所有带自环的 $\binom{m_k}{\mathscr{F}_{k()}}$ 连接配置
            else
                生成所有 $\binom{m_k}{\mathscr{F}_{k()}}$ 连接配置
            对每个连接配置**执行**
                复制 $\mathbf{G}(\tilde{\mathbf{a}},\ \mathbf{b})$ 并将其记为 $\mathbf{G}'(\tilde{\mathbf{a}},\ \tilde{\mathbf{b}})$
                将连接配置追加到 $\mathbf{G}'(\tilde{\mathbf{a}},\ \tilde{\mathbf{b}})$
                将这一对入队 $(\mathbf{G}'(\tilde{\mathbf{a}},\ \tilde{\mathbf{b}}),\ k+1)$ 放入 $\mathcal{Q}_{\text{inner}}$
返回 $\mathcal{G}(\mathbf{a},\mathbf{b})$

算法 A.8 均匀无向图采样（UUGS）算法

输入： 度序列 $\mathbf{a}=\mathbf{b}\in\mathbb{N}_+^m$
输出： 一个与度序列 $\mathbf{a}$ 和 $\mathbf{b}$ 一致的无向图

初始化 $\tilde{\mathbf{a}}\leftarrow\mathbf{a}+\mathbf{1}$, $\mathbf{b}\leftarrow\mathbf{b}+\mathbf{1}$, 一个空二分图 $\mathbf{G}(\tilde{\mathbf{a}},\ \tilde{\mathbf{b}})$
对于 $j=1$ **到** m，**执行**
    找到一个索引 $j^\star$，它对应于剩余度最大的节点 $\tilde{\mathbf{b}}^{[j]}$
    交换其中的索引 j 和 $j^\star$ $\tilde{\mathbf{b}}^{[j]}$
    将 $\tilde{\mathbf{a}}^{[j]}$ 分成 p 组 (α_k,m_k)，每组包含 $\alpha_1<\alpha_2<\cdots<\alpha_p$
    对于 $k=1$ **到** p，**执行**
        计算区间 $[\ell_k,u_k]$，通过 (12)–(14)
        为每个 $\mathscr{F}(k)\in[\ell_k,u_k]$ 生成所有连接配置
        获取精确权重向量 $\mathbf{w}^{\text{UU}}$，通过 (18) 和对称连接步骤
        根据 $\mathbf{w}^{\text{UU}}$ 抽取一个连接配置
        为节点 [实现抽取到的连接配置 $\tilde{\mathbf{b}}]_j$ 在$\mathbf{G}(\tilde{\mathbf{a}},\ \tilde{\mathbf{b}})$
    为节点 $[\tilde{\mathbf{a}}]_j$ 建立与 [对应的对称连接 $\tilde{\mathbf{b}}]_j$
从$\mathbf{G}(\tilde{\mathbf{a}},$中移除所有自环 $\tilde{\mathbf{b}})$，以获得与$\mathbf{a}$和$\mathbf{b}$一致的有向图
返回生成的有向图

算法 A.9 高效无向图采样算法（EUGS）

输入： 度序列 $\mathbf{a}=\mathbf{b}\in\mathbb{N}_+^m$

输出： 一个与度序列$\mathbf{a}$和$\mathbf{b}$一致的无向图

初始化 $\tilde{\mathbf{a}}\leftarrow\mathbf{a}+\mathbf{1}$, $\mathbf{b}\leftarrow\mathbf{b}+\mathbf{1}$，一个空的二分图 $\mathbf{G}(\tilde{\mathbf{a}},\ \widetilde{\mathbf{b}})$

对于 $j=1$ **到** m **执行**

  找到剩余度最大的节点的索引 $j^\star$，在 $\widetilde{\mathbf{b}}^{[j]}$

  交换索引 j 和 $j^\star$，在 $\widetilde{\mathbf{b}}^{[j]}$

  将 $\tilde{\mathbf{a}}^{[j]}$ 分成 p 组 (α_k, m_k)，其中 $\alpha_1<\alpha_2<\cdots<\alpha_p$

  对于 $k=1$ **到** p **执行**

    计算区间 $[\ell_k,u_k]$ 通过 (12) – (14)

    通过 (11)获取近似权重向量$\mathbf{w}^{\mathrm{EU}}$

    根据$\mathbf{w}^{\mathrm{EU}}$采样 $\mathscr{F}(k)\in[\ell_k,u_k]$

    如果 $k=k_j$，**则**

      连接节点 $[\widetilde{\mathbf{b}}]_j$ 与其对应节点相连，形成一个自环

      在组 k 中均匀选择 $\mathscr{F}(k)-1$ 个不同的剩余节点，并将它们连接到节点 $[\mathbf{b}]_j$ 中$\mathbf{G}(\tilde{\mathbf{a}},\ \mathbf{b})$

    else

      从组 k 中均匀选取 $\mathscr{F}(k)$ 个不同的节点，并将它们连接到节点 $[\widetilde{\mathbf{b}}]_j$ 位于 $\mathbf{G}(\tilde{\mathbf{a}},\ \widetilde{\mathbf{b}})$

  为节点 $[\tilde{\mathbf{a}}]_j$ 建立与 $[\widetilde{\mathbf{b}}]_j$ 对应的对称连接

  如果不是 FEASIBLE($\mathbf{G}(\tilde{\mathbf{a}},\ \widetilde{\mathbf{b}})$ j) **那么** ▷ 可行性验证步骤

    从 $j=1$ 中重新采样

从 $\mathbf{G}(\tilde{\mathbf{a}},$ 中移除所有自环 $\widetilde{\mathbf{b}})$，以获得与 $\mathbf{a}$ 和 $\mathbf{b}$一致的无向图

返回生成的无向图

算法 A.10 蛮力算法

输入： 度序列 $\mathbf{a}\in\mathbb{N}_+^m$和 $\mathbf{b}\in\mathbb{N}_+^n$

输出： 与度序列 $\mathbf{a}$ 和 $\mathbf{b}$相符的所有二分图的集合 $\mathcal{G}(\mathbf{a},\ \mathbf{b})$

初始化 $\mathcal{G}(\mathbf{a},\ \mathbf{b})\leftarrow\emptyset$

如果 $\prod_{i=1}^m([\mathbf{a}]_i)\neq\prod_{j=1}^n([\mathbf{b}]_j)$ **则**

  设 $\mathbf{d}^{(1)}\leftarrow\mathbf{a}$，$\mathbf{d}^{(2)}\leftarrow\mathbf{b}$ ▷ 先满足 $\mathbf{a}$，再按 $\mathbf{b}$ 筛选

否则

  设置 $\mathbf{d}^{(1)}\leftarrow\mathbf{b}$、$\mathbf{d}^{(2)}\leftarrow\mathbf{a}$ ▷ 先满足 $\mathbf{b}$，再按 $\mathbf{a}$ 筛选

对 $t=1$ **至** $|\mathbf{d}^{(1)}|$ **执行** ▷ 构造所有满足 $\mathbf{d}^{(1)}$ 的连接配置

  初始化 $\mathcal{C}_t\leftarrow\emptyset$

  生成 $\binom{|\mathbf{d}^{(2)}|}{[\mathbf{d}^{(1)}]_t}$ 连接配置 ▷ 从 $|\mathbf{d}^{(2)}|$ 个$\mathbf{d}^{(2)}$节点中选择 $[\mathbf{d}^{(1)}]_t$

  将这些连接配置附加到 $\mathcal{C}_t$

令 $\mathcal{C}_T\leftarrow\mathcal{C}_1\times\mathcal{C}_2\times\cdots\times\mathcal{C}_{|\mathbf{d}^{(1)}|}$ ▷ $\mathcal{C}_1,\mathcal{C}_2,\ldots,\mathcal{C}$ 的笛卡尔积

对于 $\mathbf{C}_{\mathrm{one}}\in\mathcal{C}_T$，**执行**

  初始化一个空的二分图 $\mathbf{G}(\mathbf{a},\mathbf{b})$

  将连接配置 $\mathbf{C}_{\mathrm{one}}$ 追加到 $\mathbf{G}(\mathbf{a},\mathbf{b})$

  如果 $\mathbf{G}(\mathbf{a},\mathbf{b})$ **满足** $\mathbf{d}^{(2)}$，**则**

    将 $\mathbf{G}(\mathbf{a},\mathbf{b})$ 追加到 $\mathcal{G}(\mathbf{a},\mathbf{b})$

返回 $\mathcal{G}(\mathbf{a},\ \mathbf{b})$