线性子分包的多接入编码缓存

Srinivas Reddy Kota 印度理工学院马德拉斯分校
ksreddy@ee.iitm.ac.in

Nikhil Karamchandani 印度理工学院孟买分校 nikhilk@ee.iitb.ac.in

arXiv:2305.05514v1 [cs.IT] 9 May 2023

摘要——**我们研究多接入编码缓存问题，其中包含一个拥有 N 个文件的中央服务器、 K 个缓存（每个缓存的存储容量为 M 个单位），以及 K 个用户；每个用户连接到 $L(\geq 1)$ 个连续缓存，并采用循环回绕方式。缓存中存放与文件相关的内容，每个用户随后请求一个文件，该文件需要借助缓存，通过中央服务器广播的消息提供给用户。我们的目标是为该设置设计放置与传输策略，在满足额外的线性子分包约束的同时，尽可能降低中央服务器的传输速率。我们提出了满足这一约束的策略，并推导出所实现服务器传输速率的上界；通过与现有文献中的结果比较，证明了本文结果带来的改进。为推导这些结果，我们将多接入编码缓存问题映射为若干已知索引编码问题的变体。在此过程中，我们还为一种“结构化”索引编码问题推导出新的最优传输规模界，这一结果本身也可能具有独立的研究价值。**

I. 引言

Maddah-Ali 和 Niesen 的开创性工作 [1]开启了对缓存系统的信息论研究。缓存系统通过将数据流量从高峰时段转移到非高峰时段，帮助缓解网络拥塞。此后，大量研究探讨了这一“编码缓存”（CC）问题的各种变体。特别是，受无线网络的启发， [2] 研究者提出了多接入编码缓存（MACC）问题。该问题包含一个中央服务器，其中存有 N 个文件， K 个缓存，每个缓存的存储容量为 M 单位，以及 K 个用户，每个用户连接到 L 个连续缓存，并采用循环回绕方式；示意图见图1 。缓存中预先存入与这些文件相关的内容，随后每个用户从 N 个文件中请求一个文件，所有用户的请求都必须由中央服务器借助缓存通过广播消息来满足。目标是设计高效的放置与传输策略，以最小化中央服务器的传输速率。

CC问题和MACC问题的最优传输速率已在文献中得到广泛研究，并且已分别推导出可达性结果和信息论反向界；更多详细信息请参见 [1]–[8] 。具有阶最优传输速率的方案通常需要将文件划分为许多更小的子文件（称为子分包），其中子文件的数量通常随系统规模呈指数增长。这反过来会引入巨大的开销，而这些开销可能会抵消带宽编码传输方案所带来的增益，因此人们一直非常关注为 CC问题和 MACC问题设计低子分包方案 [9]–[16]，以使服务器传输速率方面的代价尽可能低。

本文聚焦于具有线性子分包的MACC问题，即子文件的数量随用户数量 K线性增长。近期已有一些关于该问题的研究。具体而言， [15]基于放置传输阵列（PDA）的思想提出了线性子分包方案。然而，这些组合结构仅存在于受限的系统参数集合中。另一方面，尽管 [16] 提出了更一般的构造，但所需的子分包有时是超线性的（$O(K^2)$）。我们为MACC问题提供了替代性的低子分包方案，并能够改进此前的研究结果，具体讨论如下。

具体而言，我们采用与 [16] 相同的放置策略，但提出了一种不同的传输策略，该策略基于与索引编码问题（ICP） [17],[18] 的联系，以获得更优的传输速率。若固定放置策略和用户请求，那么求解编码缓存问题等价于求解索引编码问题（ICP）。这种关系已被证明对相关文献中的研究至关重要，既可用于设计更优的可达性方案 [4], [19]，也可用于证明逆定理结果 [3]。在我们的策略中，我们将MACC问题映射为 [19] 中先前研究过的一类“结构化”索引编码问题，并利用其中的结果以及本文推导出的新上界，为线性分包下MACC问题的最优服务器传输速率建立了新的、更优的界。本文的主要贡献总结如下：

- 我们为文献中已有研究的结构化索引编码问题推导了新的最优传输速率上界。
- 基于映射到这一结构化索引编码问题的方法，我们为 MACC问题提出了一种采用线性分包的策略，并能够为任意给定的系统参数给出该策略传输速率的上界。
- 我们将结果与文献中的结果进行了比较，并展示了在服务器传输速率和子分包级别方面的改进。
- 尽管我们主要关注采用线性分包的多接入编码缓存（MACC），但我们也使用我们的策略改进了采用 $O(K^2)$ 子分包的结果。

Srinivas Reddy Kota 的工作部分得到了印度政府印度科学技术部提供的 INSPIRE 教师奖学金的支持。Nikhil Karamchandani 的工作部分得到了 SERB MATRICS 资助的支持。

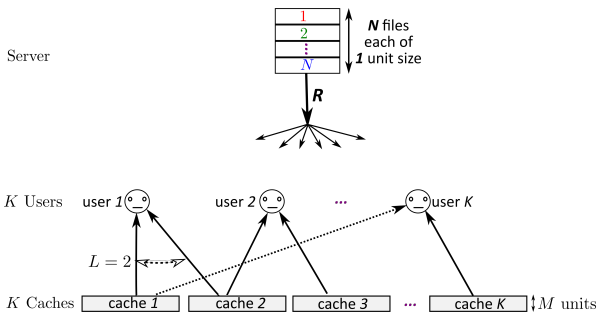


图 1. 多接入编码缓存（*MACC*）示意图，其中包含一个存有 N 个文件的中央服务器、 K 个缓存（每个大小为 M 个单位）以及 K 个用户；每个用户通过循环回绕的方式连接到 $L = 2$ 个缓存。

本文其余部分安排如下。第 II 节和 III 节介绍所需的符号，并描述形式化问题设定。第 IV节讨论索引编码问题的预备知识。第V 节讨论我们的策略，第VI 节简要介绍本文的主要结果。我们在第 VII 节进行简要讨论，并将所有证明和一个示例放在附录中。

II. 符号

- $[m] = \{1, 2, 3, ..., m\}$
- 对于给定的整数 [2]，满足 K 使得 $K \geq m$， n，

$$[n : m] = \begin{cases} \{n, n+1, ..., m\} & \text{if } n \leq m \\ \{n, n+1, ..., K, 1, 2, ..., m\} & \text{if } n > m \end{cases}$$

-

$$< n >_m = \begin{cases} n \pmod m & \text{if } n \pmod m \neq 0 \\ m & \text{if } n \pmod m = 0 \end{cases}$$

- $|S|$–文件 / 子文件 / 集合 S 的大小
- $\mathcal{F}_j$, $\mathcal{S}$ 表示文件 j 中仅对索引属于集合 $\mathcal{S}$ 的用户可用的部分

III. 系统设定

我们首先关注 [2],中最初研究的多接入编码缓存设置，它由一台中央服务器、 K 个缓存和 K 个用户组成，见图1。我们假设

- 中央服务器拥有 N个文件 $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_N$, 每个文件大小为 1 个单位（$= B$ 比特[3]）
- 每个缓存的存储容量为 M 个单位，
- 每个用户可接入 L 个连续缓存，并采用循环回绕
- 每个用户从 N 个文件中请求一个文件，并且
- 中央服务器与用户之间的共享通信链路无误码。

类似于经典的编码缓存（CC）问题 [20], MACC问题包含两个阶段：放置阶段和传输阶段，具体描述如下。

[2]在我们的稿件中， K 是索引编码问题／多接入编码缓存问题中的用户数量。[3]我们假设 B 足够大

1) 放置阶段： 在放置阶段，每个缓存存储与 N 文件相关的内容。本文将分析范围限定为采用线性子分包的无编码放置策略，这意味着我们允许将文件划分为 $O(K)$ 个部分并存储这些文件分块，但不允许跨文件分块进行编码。

放置阶段结束后，用户会公开其请求的文件。设用户 j 请求文件 d_j，而（d_1， d_2，...， d_K）表示请求配置。

2）传输阶段：在传输阶段，中央服务器根据用户请求配置和缓存内容广播一条编码消息。每个用户利用广播的消息以及其可接入缓存中的缓存内容，尝试重建自己请求的文件。

目标是设计放置策略和编码传送策略，使服务器的总传输速率最小化，其中传输速率定义为服务器传输的总比特数除以文件大小 B。请注意，如果某种策略在放置阶段和传输阶段将一个文件划分为 F 个部分，那么我们称其采用了 F子分包。本文关注线性分包策略，即 F 的数量级为 K的策略。在本研究中，我们的主要目标是设计采用线性分包且服务器传输速率尽可能低的缓存与传送策略。

A. 先前结果

关于原始 CC 设置（$L = 1$）的线性子分包策略，已有多项研究 [9]–[13], ; 此外，也有许多研究工作关注不受子分包限制的多接入编码缓存（MACC）问题 [2], [4]–[6], [19]。然而，据我们所知，只有 [15] 和 [16] 同时关注这两个方面，即采用线性分包的多接入编码缓存，其主要研究结果将在下文讨论。

以下引理描述了 [15] 的主要结果。

命题1。 *[15, 定理 1] 考虑一个多接入编码缓存系统，其中有 N 个文件、 K 个缓存，每个缓存的存储容量为 M 单位，以及 K个用户，每个用户可接入 L 个连续缓存，并采用循环回绕。对于 $M \in \{iN/K : i \in [\lfloor K/L \rfloor], i|K, (K - iL + i)|K\}$, 我们可以通过线性分包实现 $R_1(M)$，其中*

$$R_1(M) = \frac{(K - iL)(K - iL + i)}{2K}.$$

以下引理描述了 [16] 的主要结果。

命题 2. *[16, 定理 2]考虑一个多接入编码缓存（MACC）系统，其中有N 个文件、 K 个缓存，每个缓存的存储容量为 M 单位，以及 K个用户，每个用户可接入 L 个连续缓存，并采用循环回绕。对于 $M \in \{iN/K : i \in [\lfloor K/L \rfloor]\}$, 我们可以实现 $R_2(M)$，使用 $F_2(M)$–子包划分，其中*

$$R_2(M) = \begin{cases} \frac{(K-iL)(K-iL+1)}{2K} & \text{如果} (K - iL + 1)|K \text{ 或} \\ \frac{K-iL}{2\lfloor \frac{K}{K-iL+1} \rfloor + 1} & K - iL = 1\text{否则如果} \\ \frac{K-iL}{2\lfloor \frac{K}{K-iL+1} \rfloor} & < K >_{K-iL+1} = K - iL\text{否则} \end{cases}$$

and

$$F_2(M)=\begin{cases}K & \text{if } (K-iL+1)|K \text{ or } K-iL=1\\ \left(2\lfloor\frac{K}{K-iL+1}\rfloor+1\right)K & \text{else if } <K>_{K-iL+1}=K-iL\\ \left(2\lfloor\frac{K}{K-iL+1}\rfloor\right)K & \text{else.}\end{cases}$$

请注意， [15] 中的结果仅适用于一组受限的系统参数。另一方面，虽然[16]提供了更一般的结果，但所需的子分包有时会呈超线性增长（$O(K^2)$）。在本文中，我们重点研究适用于所有系统参数且具有线性分包的多接入编码缓存。此外，在某些参数区域内，我们的结果优于先前的结果。以下示例说明了这一点。

例1. 考虑一个参数为 $N=K=8$、$L=2$ 和 $M=i=3$ 的 *MACC*问题。命题 *1* 不适用于这些参数，而命题 *2* 使用 $R_2(M)$ =0。5 个单位，子分包数为 $F=4K=32$。不过，我们的策略可以用 $F=K=8$ 个子分包达到 *3/8* 个单位的传送率，详见例 *5* 。

IV. 索引编码问题——预备知识

在编码缓存问题中，如果固定放置策略和用户请求配置，那么求解编码缓存问题等价于求解索引编码问题 [17], [18],，后者在文献中已有深入研究。我们提出的策略利用了这一联系，并基于索引编码问题的解设计传送策略。因此，我们首先介绍索引编码问题的一些预备知识和对本文有用的已有结果。特别地，我们简要关注文献 [19] 中研究的一种结构化索引编码问题，该研究旨在为 MACC问题推导出更优的传送率。在本文中，我们也针对这一结构化索引编码问题推导出更优结果，以改进当前最先进的成果；这些结果将在本节末尾讨论。

索引编码问题（ICP）由一个拥有 N 个文件的服务器和 K 个用户组成，这些用户分别需要文件的某些子集。每个用户都拥有一部分文件作为侧信息。服务器通过了解用户的侧信息和需求，广播文件的线性组合，使用户能够解码其所需的文件。其目标是最小化服务器的传输速率。设文件集合为 $\mathcal{F}=\{x_1,x_2,\ldots,x_N\}$， K 个用户的集合为 $\mathcal{U}=\{U_1,U_2,\ldots,U_K\}$。每个用户表示为 $U_k=(\mathcal{K}_k,\mathcal{W}_k)$，其中 $\mathcal{K}_k\subset\mathcal{F}$ 表示第 k 个用户的已知集或侧信息集，而 $\mathcal{W}_k\subseteq\mathcal{F}$ 表示其需求集或需求文件集合。U_k 的干扰集为 $\mathcal{I}_k=(\mathcal{K}_k\cup\mathcal{W}_k)^c$. 。尽管一般索引编码问题（ICP）的最优传输率仍是一个尚未解决的开放问题，但文献中已经提出了若干上界和下界。其中，一些上界基于最小秩 [18], 、局部色数、分数局部色数 [21], 以及局部部分团覆盖 [22]。一些下界则基于最大诱导无环子图（MAIS） [23], 、冲突距离 [24], 和维数计数 [25]。

单播索引编码问题（UICP）是一种索引编码问题，其中同一个文件不会被多个用户请求。单播索引编码问题（SUICP）是一种统一索引编码问题，其中每个用户只请求一个文件。我们将含有 K 个用户的 SUICP 称为某个非负整数 a_1、a_2 和自然数 z 下的 $(a_1,a_2)_z$–索引编码问题，使得

$$a_1+a_2+z=K-1,$$

若 $\forall k\in[K]$，用户 k

- 想要 x_k，即 $\mathcal{W}_k=\{x_k\}$ 和
- 知道 $\mathcal{K}_k=\{x_b:\ b=<k+a_1+r>_K,\ r\in[z]\}$，即侧信息 $\mathcal{K}_k$ 是由 z个连续元素组成的集合。更明确地说，用户 k 的侧信息由以下内容给出。

$$\mathcal{K}_k=\{x_{<k+a_1+1>_K},x_{<k+a_1+2>_K},\ldots,x_{<k+a_1+z>_K}\}.$$

以下结果给出了（a_1， a_2）$_z$–索引编码问题的传输速率上界。

命题 3。 *[26,* 命题 *4]* 设 R^* 为一个（a_1, a_2）$_z$–索引编码问题的最优传输率，则

$$R^*\le R=a_1+a_2+1.$$

设有 K 个用户，使得在第一个实例中，用户 k 想要 $\mathcal{W}_k^1$ 并且知道 $\mathcal{K}_k^1$，而同一用户 k 在第二个实例中想要 $\mathcal{W}_k^2$ 并且知道 $\mathcal{K}_k^2$。假设这两个实例不涉及共同文件，我们将这两个实例的并集定义为一个索引编码问题，其具有

- want set $\mathcal{W}_k=\mathcal{W}_k^1\cup\mathcal{W}_k^2$ and
- known set $\mathcal{K}_k=\mathcal{K}_k^1\cup\mathcal{K}_k^2$.

W我们可以将其推广到多个实例的并集。

对于某些非负整数 $a_1,a_2(\le a_1)$ 和一个自然数 z，$(a_1,a_2)_z$–索引编码问题与 $(a_2,a_1)_z$–索引编码问题的并集称为 $(a_1,a_2)_z$−索引编码问题。令 $x_{k,j}$为用户 k 请求的 j^{th} 文件。在 $(a_1,a_2)_z$−索引编码问题中，$\forall k\in[K]$，用户 k

- want set $\mathcal{W}_k=\{x_{k,1},x_{k,2}\}$
- known set

$$\mathcal{K}_k=\{x_{b,t}:b=<k+a_t+r>_K,t\in[2],r\in[z]\}.$$

更多详情请参见 [19] 。注意，$(a_1,a_2)_z$–索引编码问题是一个$(a_1,a_2,\ldots,a_i)_z$–索引编码问题，而$(a_1,a_2)_z$−索引编码问题是一个具有 $i=2$的$(a_1,a_2,\ldots,a_i)_z$−索引编码问题，这些问题在[19]中进行了研究。以下引理出自 [19] ，给出了$(a_1,a_2)_z$−索引编码问题的最优传输率数量的上界和下界。

命题 4. *[19,*定理 *1*、定理 *2* 和定理*3]* 对于某些非负整数 $a_1,a_2(\le a_1)$ 以及一个自然数 z，考虑一个具有 K 个用户的 $(a_1,a_2)_z$− 索引编码问题。令 R^* 为该索引编码问题的最优传输率，则

$$\overline{R}^*\le\overline{R}_1=\begin{cases}a_1+a_2+2 & \text{if } (a_1+a_2+2)|K\\ \min\{a_1+2a_2+2,K\} & \text{otherwise}\end{cases}$$

and

$$\overline{R}^*\ge\overline{R}_l=a_1+a_2+2.$$

在下述定理中，我们给出了 $(a_1,\ a_2)_z$ 索引编码问题传输速率的一个新上界。

定理5。设 R^* 为 $(a_1, a_2)_z$ 索引编码问题的最优传输率，其中 a_1、a_2 为非负整数，且 $(\leq a_1)$ 为自然数。若存在一个整数 R_2，使得 $R_2 \geq a_1 + a_2 + 2$ 且 $R_2 | K$，则 $R^* \leq R_2$。

下面的示例比较了命题4 与定理5。

例2。对于 $(2,2)_9$ 索引编码问题，我们有 $K = 14$ 个用户，下界为 $R_l = 6$ 个单位，上界为 $R_1 = 8$ 个单位、$R_2 = 7$ 个单位。我们可以清楚地看到 $R_2 < R_1$。

命题 4和定理 5 中的上界基于局部色数 [21, 定理 1]，对应于标量索引编码。我们可以通过分数局部色数 [21, 定理 2]所得到的上界，利用向量索引编码来提高可达速率；改进后的上界如下所示：

定理 6. 对于某些非负整数 $a_1, a_2 (\leq a_1)$ 以及自然数 z，考虑一个 $(a_1, a_2)_z$ 索引编码问题，其中有 K 个用户。设 R^* 为该索引编码问题的最优传输率，则

$$\overline{R}^* \leq \overline{R}_3 = \frac{\min\{\lfloor \frac{K}{a_1+a_2+2} \rfloor (a_1 + a_2 + 2) + a_2, K\}}{\lfloor \frac{K}{a_1+a_2+2} \rfloor}.$$

下面的例子对比了命题 4 和定理 6。

例3。对于 $(10,10)_{1000}$ 索引编码问题，我们有 $K = 1021$ 个用户，下界为 $R_l = 22$ 个单位，上界分别为 $R_1 = 32$ 个单位和 $R_3 \approx 22.2173$ 个单位。显然，$R_3 < R_1$ 和 R_3 近似等于 R_l。

V. 我们的策略

现在，我们讨论线性分包的 MACC问题的策略。我们的策略针对角点 $M = iN/K$，其中 $i \in [\lceil K/L \rceil] \cup \{0\}$。对于中间值，我们适当地采用内存共享。如前所述，我们的策略分为两个阶段。第一阶段是放置阶段，第二阶段是传输阶段。在本文中，我们采用与 [16], 相同的放置策略，但使用不同的传输策略来改进先前结果。我们的传输策略采用了第 $(a_1, a_2)_z$ 节中提到的索引编码问题上界IV。

A. 放置阶段：

我们可以分别在平凡角点 $M = 0$ 和 $\lceil K/L \rceil$ 处实现传输速率 K 和 0。现在，我们简要介绍非平凡内存点 $M = iN/K$ 处的放置策略，其中 $i \in [\lfloor K/L \rfloor]$。

我们将每个文件 $\mathcal{F}_n\ n \in [N]$ 划分为 K 个大小相等且互不相交的子文件 $\mathcal{F}_{n,\ m}\ m \in [K]$ 。子文件 $\mathcal{F}_{n,\ m}$ 存储在缓存 m, $m + L$, ... $m + (i-1)L$中。因此，缓存 $k, k \in [K]$ 存储子文件 $\mathcal{M}_k = \{\mathcal{F}_{n,<k-(r-1)L>_K} :\ n \in [N],\ r \in [i]\}$。

放置策略的一些有用性质是

- 每个缓存存储每个文件的 i 个子文件。因此，满足内存约束 $M = iN/K$ 。
- 没有两个缓存存储相同的子文件集合
- 每个子文件恰好可供 iL 个连续用户使用，并且没有两个子文件可供同一组缓存使用。
- 每个用户可接入 iL 个子文件，且没有两个用户拥有相同的可用子文件集合

由于子文件 $\mathcal{F}_{n,\ m}$ 存储在缓存 m, $m + L$, ..., $m+(i-1)L$中，因此索引属于集合 $[< m-L+1 >_K : < m+(i-1)L >_K]$的用户均可使用该子文件。因此，$\mathcal{F}_{n,\ m}$ 表示为 $\mathcal{F}_{n,\ [<m-L+1>_K :\ <m+(i-1)L>_K]}$，其中下标集合中的索引表示相应用户可使用该子文件。对于集合 $[K]$ 中每个包含 iL 个循环连续元素的子集，都存在一个子文件，反之亦然。注意，共有 K 个子集和 K 个子文件。

B. 传输阶段：

在此阶段，每个用户都会公布其请求。设请求模式为 $\mathbf{d} = (d_1,\ d_2,\ .....,\ d_K) \in [N]^K$，即用户 j 请求文件 $\mathcal{F}_{d_j}$。换言之，用户 j 请求文件 $\mathcal{F}_{d_j}$的 K个子文件。根据我们的放置策略，用户 j 已经可以使用 iL 个子文件，因此只需获取文件 $\mathcal{F}_{d_j}$剩余的 $K - iL$ 个子文件。具体而言，用户 j连接到集合 $[j$ 中的缓存：$j+L-1]$，而对于 $k \in [K]$，缓存 k存储子文件 $\mathcal{M}_k = \{\mathcal{F}_{n,<k-(r-1)L>_K} : n \in [N],\ r \in [i]\}$。因此，用户 j已经可以使用$\mathcal{F}_{d_j,\ <j+l-1-(r-1)L>_K}$，$r \in [i],\ l \in [L]$，还需要剩余的 $K - iL$ 个子文件。

现在，我们构造一个包含 K 个用户的索引编码问题实例，使得每个用户 j 需要 $K - iL$ 个文件 $\mathcal{F}_{d_j}$中不可用的子文件。用户 j 的侧信息是其余 $K - 1$ 个用户所需且存储在用户j可用缓存中的子文件 $[j :\ j + L - 1]$。中央服务器根据索引编码问题的解发送消息。注意，我们证明所构造的索引编码问题是若干个 $(a_1, a_2)_{z=iL}$ 索引编码问题的并集，并利用第IV节中的结果，得到这些索引编码问题最优传输率的上界。最后，我们的多接入编码缓存索引编码问题的最优传输率不超过各个 $(a_1,\ a_2)_{z=iL}$ 索引编码问题上界之和，详见附录 C 。

VI. 主要结果

在本节中，我们刻画该策略的传输速率。首先，我们讨论具有线性子分包的多接入编码缓存传输速率的改进上界，这是命题 1 的推广。

定理7。考虑具有 N 个文件、K 个缓存的多接入编码缓存系统，每个缓存的存储容量为 M 单位，系统中有 K 个用户，每个用户可接入 L 个连续缓存，并采用循环回绕。对于 $M \in \{iN/K :\ i \in [0, K/L]\}$，若存在 $X \in \mathbb{N}$使得 $X \geq K - iL + 1$且 $X | K$，则我们可以使用线性分包实现$R_3(M)$</style}，其中

$$R_3(M) = \frac{(K - iL)X}{2K}.$$

如果 i 且 $K - iL + i$ 整除 K，代入 $X = K - iL + i$后，我们即可从定理 1 中得到命题 7。为了

为实现这些结果，当 $K-iL$ 为偶数时，我们使用 $F=K$；当 $K-iL$ 为奇数时，我们使用 $F=K+1$；两者均关于 K 呈线性关系。

现在，我们讨论一般 MACC 问题在采用线性分包且参数不受限制时，传输速率的另一个上界。令 $R_s(M)$、$R_e(M)$ 和 $R_o(M)$ 在 $M=iN/K, i\in[\lfloor K/L\rfloor]$ 处定义如下：

$$R_s(M):=\frac{(K-iL)(K-iL+1)}{2K} \tag{1}$$

$$R_e(M):=\begin{cases}\frac{K(K-iL)-iL(iL-1)}{2K} & \text{if } K\geq 3iL\\ \frac{(K-iL)(5K-5iL+2)}{8K} & \text{otherwise}\end{cases} \tag{2}$$

$$R_o(M):=\begin{cases}\frac{K(K-iL+1)-iL(iL+1)}{2K} & \text{if } K\geq 3iL+1\\ \frac{(K-iL-1)(5K-5iL+9)}{8K}+1 & \text{otherwise}\end{cases} \tag{3}$$

下述定理讨论了采用线性分包的一般多接入编码缓存的传输速率上界。

定理 *8.* 考虑一个 *MACC*，其中有 N 个文件、K 个缓存，每个缓存的存储容量为 M 单位，以及 K 个用户；每个用户可接入 L 个连续缓存，并采用循环回绕。对于 $M\in\{iN/K:\ i\in[0,\lfloor K/L\rfloor]\}$，令 $R_s(M)$、$R_e(M)$ 和 $R_o(M)$ 分别按式 *(1)*、*(2)* 和 *(3)* 定义。那么，采用线性分包时，我们可以实现 $R_4(M)$，其中

$$R_4(M)=\begin{cases}R_s(M) & \text{if } (K-iL+1)|K \text{ or } K-iL=1\\ R_e(M) & \text{else if } K-iL \text{ is even}\\ R_o(M) & \text{else}\end{cases}$$

如果考虑 $iL<K/3$、$K>K-iL+1\neq K-iL$和 $K-iL$ 为偶数的情况，那么根据命题 2，我们有 $R_2(M)=\frac{K-iL}{2}$；而根据定理 8，我们有 $R_4(M)=R_e(M)=\frac{K-iL}{2}-\frac{iL(iL-1)}{2K}<R_2(M)$。这表明，定理 8 的结果可能优于命题 2 的结果。此外，命题 2 对某些参数采用非线性子分包，而定理 9 对所有参数仅采用线性分包。

如果我们将允许的子包划分从线性增加到 $O(K^2)$，则无需改变放置策略，就可以进一步改进定理 8 的结果。令 $R_e(M)$ 和 $\sim R_o(M)$在 $M=iN/K$，$i\in[\lfloor K/L\rfloor]$ 定义如下：

$$\widetilde{R}_e(M):=\sum_{j=0}^{\frac{K-iL}{2}-1}\frac{\min\{\lfloor\frac{K}{K-iL+1}\rfloor(K-iL+1)+j,K\}}{\lfloor\frac{K}{K-iL+1}\rfloor K} \tag{4}$$

$$\widetilde{R}_o(M):=\frac{\min\{\lfloor\frac{K}{K-iL+1}\rfloor(K-iL+1)+\frac{K-iL-1}{2},K\}}{2\lfloor\frac{K}{K-iL+1}\rfloor K}+\sum_{j=0}^{\frac{K-iL-3}{2}}\frac{\min\{\lfloor\frac{K}{K-iL+1}\rfloor(K-iL+1)+j,K\}}{\lfloor\frac{K}{K-iL+1}\rfloor K} \tag{5}$$

采用 $O(K^2)$ 子分包可获得的改进结果如下：

定理 *9*。考虑一个多接入编码缓存系统（*MACC*），其中有 N 个文件、K 个缓存，每个缓存的存储容量为 M 单位，以及 K 个用户；每个用户可接入 L 个连续缓存，并采用循环回绕。对于 $M\in\{iN/K:\ i\in[0,\lfloor K/L\rfloor]\}$，令 $\widetilde{R}_e(M)$ 和 $\widetilde{R}_o(M)$ 分别按式 *(4)* 和 *(5)* 定义。那么，我们可以通过使用 $F_5(M)$– 子分包实现 $R_5(M)$，其中

$$\widetilde{R}_5(M)=\begin{cases}\widetilde{R}_e(M) & \text{if } K-iL \text{ is even}\\ \widetilde{R}_o(M) & \text{otherwise}\end{cases}$$

and

$$\widetilde{F}_5(M)=\begin{cases}\widetilde{K} & \text{if } (K-iL+1)|K \text{ or } K-iL=1\\ \widetilde{K}\lfloor\frac{K}{K-iL+1}\rfloor & \text{else}\end{cases}$$

其中，

$$\widetilde{K}=\begin{cases}K & \text{if } K-iL \text{ is even or } K-iL=1\\ K+1 & \text{else.}\end{cases}$$

下面，我们通过一个示例将本文结果与文献中已有的结果进行比较。

示例 *4.* 考虑一个具有 $K=100, i=20, L=4$ 的 *MACC*，其传输速率为 $R_2(M)\sim=R_3(M)=2.5$，$R_4(M)=2.55$，以及 $R_5(M)\approx 2.14$，子分包为 $F_2(M)=800(>K)$ 和 $\sim F_5(M)=400=F_2(M)/2$。注意，对于上述参数，$R_1(M)$ 不存在。

该示例表明，无论是在传输速率还是子分包方面，我们的结果都优于早期研究中的结果。

注：10。通过代入 $L=1$，我们也可以将本文结果用于原始 *CC*问题，从而推导出具有线性分包的改进速率。

注：11。如前所述，我们利用局部色数的界来界定映射 *MACC*问题所得结构化索引编码问题的传输速率。其中一个潜在问题是，构造 *MDS* 码需要较大的域大小，详情见 *[21]*。这并非总是可行，因此一个自然的问题是，我们能否构造性能相当的二进制码。*[21,* 推论 *1],* 提供了一种部分解决方案，它表明，以较高概率，我们可以通过额外增加 $2^{\log(K(K-iL))}K$ 个单位的传输速率，为 *MACC*问题构造二进制 *MDS* 码。

VII. 结论

本文聚焦于具有线性分包的 MACC问题中缓存与传输方案的设计和分析。这些方案以及相应的服务器传输速率上界，基于编码缓存问题与先前研究的一类结构化索引编码问题之间的联系；针对后者，我们推导了新的改进界，其本身也具有研究价值。我们的结果在若干参数范围内优于当前最佳结果。推导传输速率的下界，并刻画具有线性分包的 MACC 的精确速率—存储容量权衡，是我们未来的研究工作之一。

参考文献

[1] M. A. Maddah-Ali 和 U. Niesen，“缓存的基本极限”，*IEEE*信息论汇刊，第 60 卷，第 5 期，第 2856–2867 页，2014 年。[2] J. Hachem、N. Karamchandani 和 S. N. Diggavi，“面向多级流行度和接入的编码缓存”，*IEEE*信息论汇刊，第 63 卷，第 5 期，第 3108–3141 页，2017 年。[3] K. Wan、D. Tuninetti 和 P. Piantanida，“论无编码缓存放置的最优性”，载于*IEEE*信息论研讨会 *(ITW)*，2016 年，第 161–165 页。[4] K. S. Reddy 和 N. Karamchandani，“采用无编码放置的多接入编码缓存的速率—存储容量权衡”，*IEEE*通信汇刊，第 68 卷，第 3261–3274 页，2020 年。[5] S. Sasi 和 B. S. Rajan，“一种改进的采用无编码放置的多接入编码缓存方案”，*arXiv* 预印本 *arXiv:2009.05377*，2020 年。[6] B. Serbetci、E. Parrinello 和 P. Elia，“多接入编码缓存：超越缓存冗余的增益”，载于*2019 IEEE*信息论研讨会 *(ITW)*，2019 年，第 1–5 页。[7] G. S. Paschos、G. Iosifidis、M. Tao、D. Towsley 和 G. Caire，“缓存在未来通信系统与网络中的作用”，*IEEE*通信领域精选期刊，第 36 卷，第 6 期，第 1111–1125 页，2018 年。[8] M. Cheng、K. Wan、D. Liang、M. Zhang 和 G. Caire，“面向多接入网络的共享链路编码缓存方案新型变换方法”，*IEEE*通信汇刊，第 69 卷，第 11 期，第 7376–7389 页，2021 年。[9] H. H. S. Chittoor、P. Krishnan、K. S. Sree 和 B. Mamillapalli，“基于射影几何的次指数和线性分包编码缓存”，*IEEE*信息论汇刊，第 67 卷，第 9 期，第 6193–6222 页，2021 年。[10] K. Shanmugam、A. M. Tulino 和 A. G. Dimakis，“利用 Ruzsa–Szemerédi 图实现线性分包编码缓存”，载于*IEEE*国际信息论研讨会 *(ISIT)*，2017 年，第 1237–1241 页。[11] V. Aravind、P. K. Sarvepalli 和 A. Thangaraj，“面向线性分包编码缓存的 PDA 提升构造”，*IEEE*通信汇刊，第 70 卷，第 7817–7829 页，2022 年。[12] L. Tang 和 A. Ramamoorthy，“基于线性分组码的低分包编码缓存方案”，*IEEE*信息论汇刊，第 64 卷，第 4 期，第 3099–3120 页，2018 年。

[13] M. Salehi、A. Tolli、S. P. Shariatpanahi 和 J. Kaleva，《多天线编码缓存中的子分包-传输率权衡》，载于*IEEE*全球通信大会（*GLOBECOM*），2019年，第1–6页。[14] M. Cheng、Q. Yan、X. Tang 和 J. Jiang，《低传输率和低子分包数的编码缓存方案》，*arXiv*预印本 *arXiv:1703.01548*，2017年。[15] S. Sasi 和 B. S. Rajan，《使用PDA实现线性分包的多接入编码缓存方案》，*IEEE* 通信汇刊，第69卷，第12期，第7974–7985页，2021年。[16] J. Wang、M. Cheng 和 Y. Wu，《PDA和连续循环放置下具有最优传输率和线性分包的多接入编码缓存》，*arXiv*预印本 *arXiv:2209.02987*，2022年。[17] Y. Birk 和 T. Kol，《广播信道上的知情源按需编码（iscod）》，载于*IEEE INFOCOM'98*会议论文集，第3卷，1998年，第1257–1264页。[18] Z. Bar-Yossef、Y. Birk、T. Jayram 和 T. Kol，《带旁信息的索引编码》，*IEEE*信息论汇刊，第57卷，第3期，第1479–1494页，2011年。[19] K. S. Reddy 和 N. Karamchandani，《结构化索引编码问题与多接入编码缓存》，*IEEE*信息论精选领域期刊，第2卷，1281，2021 4，1266期，第 – 页。[20] M. A. Maddah-Ali 和 U. Niesen，《去中心化编码缓存实现阶数最优的存储容量-传输率权衡》，*IEEE/ACM*网络汇刊，第23卷，第4期，第1029–1040页，2015年8月。[21] K. Shanmugam、A. G. Dimakis 和 M. Langberg，《局部图着色与索引编码》，载于*IEEE*国际信息论研讨会（*ISIT*），2013年，第1152–1156页。[22] A. Agarwal 和 A. Mazumdar，《索引编码的局部部分团覆盖》，*arXiv*预印本 *arXiv:1603.02366*，2016年。[23] F. Arbabjolfaei、B. Bandemer、Y.-H. Kim、E. Sa̧so ̆glu 和 L. Wang，《索引编码的容量区域》，载于*IEEE*国际信息论研讨会（*ISIT*），2013年，第962–966页。[24] S. A. Jafar，《通过索引编码进行拓扑干扰管理》，*IEEE*信息论汇刊，第60卷，第1期，第529–568页，2014年。[25] H. Maleki、V. R. Cadambe 和 S. A. Jafar，《索引编码——干扰对齐视角》，*IEEE*信息论汇刊，第60卷，第9期，第5402–5432页，2014年。[26] K. S. Reddy、N. S. Karat 和 N. Karamchandani，《对称索引编码问题的最优传输率》，载于*IEEE*国际信息论研讨会（*ISIT*），2021年，第220–225页。

附录 A 定理证明定理 5

我们采用相同的证明技术 [19, 定理 1]。我们使用以下来自 [19,第 IV 节] 的索引编码问题术语定义进行证明。如果用户 j的请求的文件在用户 k处不可用，那么用户 j 是用户k的干扰用户。包含用户 k 及其所有干扰用户的集合称为用户 k的闭反外邻域。着色方案为每个用户分配一种颜色；如果其分配颜色的方式使得没有用户与其任何干扰用户共享颜色，则称为"合法着色方案"。索引编码问题的局部色数 $\mathcal{X}_l$ 定义为：在所有合法着色方案中取任意用户的闭反外邻域内出现的不同颜色的最大数量的最小值。我们的证明还使用以下引理。

引理 1。 *[21, 定理 1]* 设 $\mathcal{X}_l$ 为索引编码问题侧信息图的局部色数，则该索引编码问题的最优传输率有上界$\mathcal{X}_l$。

现在，我们讨论定理 5的证明。

证明。 回顾第 IV 节可知，在 $(a_1, a_2)_z$−索引编码问题中，$\forall k \in [K]$ 用户 k

- want set $\mathcal{W}_k = \{x_{k,1}, x_{k,2}\}$
- known set

$$\mathcal{K}_k = \{x_{b,t} : b =< k + a_t + r >_K, t \in [2], r \in [z]\}.$$

也就是说，在 $(a_1, a_2)_z$−索引编码问题中，每个用户需要 2 个文件，并拥有$2z$ 个文件作为旁信息。我们构造一张 $K \times 2$ 表，使得 p^{th} 行和 q^{th} 列包含 q^{th} 个由用户 p请求的文件，见表 I。我们将 p^{th} 行和 q^{th}列的条目记为节点 (p, q)。

$x_{1,1}$	$x_{1,2}$
$x_{2,1}$	$x_{2,2}$
⋮	⋮
$x_{K,1}$	$x_{K,2}$
$(a_1, a_2)_z$−索引编码问题	$(a_2, a_1)_z$−索引编码问题

表 I$(a_1, a_2)_L$ −索引编码问题的表格表示。$x_{p,\ q}$ 表示用户 p 请求的文件。在该表中，第一列对应于$(a_1, a_2)_z$−索引编码问题，第二列对应于$(a_2, a_1)_z$−索引编码问题。

回顾一下，$(a_1, a_2)_z$−索引编码问题是一个包含 K 个用户的统一索引编码问题，且每个用户请求 2 个不同的文件。我们将此统一索引编码问题转换为一个包含 $2K$ 个虚拟用户的对称统一索引编码问题，每个虚拟用户请求一个不同的文件。具体而言，统一索引编码问题中的每个用户都映射为对称统一索引编码问题中的 2 个虚拟用户，使得每个虚拟用户请求原用户所请求文件中的一个不同文件。虚拟用户的侧信息与对应的原始用户相同。请注意，构造的对称统一索引编码问题中的每个节点都代表一个虚拟用户。在表 I 中，构造的对称统一索引编码问题中节点 $(p,\ q)$ 的侧信息与原统一索引编码问题中用户 p 的侧信息相同。统一索引编码问题和构造的对称统一索引编码问题的最优传输速率相同。因此，我们给出构造的对称统一索引编码问题的一个上界，该上界同样适用于原统一索引编码问题。

一种合法着色方案：我们使用 R_2 种颜色，并为节点 $(k,1)$ 分配颜色 $< k >_{\bar{R}_2}$，为节点 $(k,2)$ 分配颜色 $< k+a_1+1 >_{\bar{R}_2}, \forall k \in [K]$。注意，每种颜色在一列中恰好出现 K/R_2 次。现在，我们验证这是一种合法着色方案。

考虑第 1 列中的一个节点，例如节点 $(k,\ 1)$。注意，其颜色为 $< k >_{\bar{R}_2}$，其在第 1 列中的干扰节点为节点 $(< k-j >_K,\ 1)\ \forall j \in [a_2]$ 和节点 $(< k+j >_K,\ 1)$ $\forall j \in [a_1]$，它们的颜色分别为 $< k-j >_{\bar{R}_2} \forall j \in [a_2]$ 和 $< k+j >_{\bar{R}_2}\ \forall j \in [a_1]$。我们可以轻易验证，这些颜色均与颜色$< k >_{\bar{R}_2}$不重叠。因此，节点 $(k,\ 1)$ 的颜色与其第一列干扰节点的颜色不同。节点 $(k,\ 1)$ 在第 2 列中的干扰节点为节点 $(< k-j >_K,\ 2)\ \forall j \in [a_1]$、节点 $(k,$ 2) 和节点 $(<k+j >_K,\ 2)\ \forall j \in [a_2]$，它们的颜色分别为 $< k-j+a_1+1 >_{\bar{R}_2}\ \forall j \in [a_1]$、$< k+a_1+1 >_{\bar{R}_2}$ 和 $< k+j+a_1+1 >_{\bar{R}_2}\ \forall j \in [a_2]$。同样，我们可以轻易验证，这些颜色均不是颜色 $< k >_{\bar{R}_2}$。因此，节点 $(k,$ 1) 的颜色也与其第二列干扰节点的颜色不同。因此，在所提出的着色方案下，第 1 列中的所有节点的颜色均不同于对应干扰节点的颜色。通过类似的论证，可以证明第 2 列中的所有节点也满足这一点。因此，该着色方案是合理的。

由于我们只使用 R_2 种颜色，每个节点包含至多 R_2 种颜色，其闭反外邻域中的颜色数不超过该值。因此，根据引理 1，我们有 $R^* \leq R_2$。

□

附录 B 定理6的证明

证明。若将（a_1，a_2）$_z$–索引编码问题中的一个文件划分为 m 个大小相等的子文件，则对于这些子文件，我们得到 m 个（a_1，a_2）$_z$–索引编码问题的并集，详情请参见 [19, 定理 4 的证明]。为证明定理 6，我们将（a_1，a_2）$_z$−索引编码问题中的文件划分为 $m = \lfloor \frac{K}{a_1+a_2+1} \rfloor$ 个子文件。对于这些子文件，它表示 m 个（a_1，a_2）$_z$−索引编码问题的并集，如下表所示。

$x^1_{1,1}$	$x^1_{1,2}$	$x^2_{1,1}$	$x^2_{1,2}$	...	$x^p_{1,1}$	$x^p_{1,2}$
$x^1_{2,1}$	$x^1_{2,2}$	$x^2_{2,1}$	$x^2_{2,2}$	...	$x^p_{2,1}$	$x^p_{2,2}$
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$x^1_{K,1}$	$x^1_{K,2}$	$x^2_{K,1}$	$x^2_{K,2}$	...	$x^p_{K,1}$	$x^p_{K,2}$

表 II m （a_1、a_2）$_z$−索引编码问题的并集的表格表示。$x^m_{k,\ j}$ 表示用户 k请求的 j^{th} 文件的 m^{th} 子文件。在表中，奇数列对应（a_1、a_2）$_z$–索引编码问题，偶数列对应（a_2、a_1）$_z$–索引编码问题。

请注意，表II 表示一个包含 K 个用户的统一索引编码问题，且每个用户请求 $2m$ 个不同的子文件。我们将该统一索引编码问题转换为一个包含 $2Km$ 个虚拟用户的对称统一索引编码问题，其中每个虚拟用户请求一个不同的子文件。具体而言，统一索引编码问题中的每个用户都映射为对称统一索引编码问题中的 $2m$ 个虚拟用户，使得每个虚拟用户请求该用户所

请求文件中的一个子文件。虚拟用户的侧信息与其对应的原始用户相同。请注意，在表II中，每个节点代表一个虚拟用户。我们给出对称统一索引编码问题的一个上界，该上界也适用于表II中的原始统一索引编码问题。

一种合法着色方案：我们取 K 种颜色，并为节点 $(k,\ p)$ 分配颜色 $<k+\binom{p-1}{2}\ (a_1+a_2+2)\ >_K$，当$p$为奇数时；若 p 为偶数，则分配颜色 $<k-\binom{p}{2}-1\ (a_1+a_2+2)\ +a_2+1>_K$，$\forall k\in[K],\ p\in[2m]$。现在，我们验证这是一种合法着色方案。

考虑第 1 列中的一个节点，例如节点 $(k,1)$。注意，其颜色为 k，它在奇数列$p\in[2m]$ 中的干扰节点为节点 $(<k-j>_K,p)\ \forall j\in[a_2]$、节点 (k,p)（当 $p\neq 1$时），以及节点 $(<k+j>_K,p)\ \forall j\in[a_1]$；它们的颜色分别为 $<k-j+\binom{p-1}{2}\ a_1+a_2+2\ >_K$ $\forall j\in[a_2]$ $<k+\binom{p-1}{2}\ a_1+a_2+2\ >_K$ $p\neq 1$（）、（）（）（当 时），以及$<k+j+\binom{p-1}{2}\ (a_1+a_2+2)\ >_K\ \forall j\in[a_1]$。若 $m=\lfloor\frac{K}{a_1+a_2+1}\rfloor$，我们可以轻易验证这些颜色均不与颜色 k重合。因此，节点 $(k,1)$ 的颜色与其奇数列干扰节点的颜色不同。节点 $(k,1)$ 在偶数列 $p\in[2m]$ 中的干扰节点为节点 $(<k-j>_K,2)$ $\forall j\in[a_1]$、节点 $(k,2)$ 以及节点 $(<k+j>_K,2)$ $\forall j\in[a_2]$；它们的颜色分别为 $<k-j+\binom{p}{2}-1\ (a_1+a_2+2)\ +a_2+1>_K\ \forall j\in[a_1]$、$<k+\binom{p}{2}-1\ (a_1+a_2+2)\ +a_2+1>_K$ 以及 $<k+j+\binom{p}{2}-1\ (a_1+a_2+2)\ +a_2+1>_K\forall j\in[a_2]$。同样，我们可以轻易验证这些颜色均不等于颜色 k。因此，节点 $(k,1)$ 的颜色也与其偶数列干扰节点的颜色不同。因此，在所提出的着色方案下，第 1 列中的所有节点的颜色均不同于各自对应的干扰节点的颜色。通过类似的论证，我们可以证明其他列中的所有节点也都满足这一性质。因此，该着色方案是合法的。

根据 $(a_1,a_2)_z$–索引编码问题的定义，如果我们将用户按 $(m,n)_z$–索引编码问题排列在圆上，那么一个用户之后依次是其 m 干扰用户、z 侧信息用户，最后是 n 干扰用户，详情请参见 [19]。我们将前 m 个干扰用户称为顶部干扰用户，将最后n 个干扰用户称为底部干扰用户。现在，我们按列依次计算节点 (1,1) 的闭反外邻域中的颜色数。由于对称性，所有用户对应的颜色数都相同。根据我们的放置策略，$\forall p\in[2:\ 2m]$，某列中底部干扰节点的颜色与其前一列中顶部干扰节点的颜色 p 相同。因此，它们不会计入节点 (1,1) 的闭反外邻域。因此，对节点 (1,1) 的闭反外邻域有贡献的列数为：第 1 列中有 a_1+a_2+1，除第 1 列外的奇数列中有 a_1+1，偶数列中有 a_2+1。因此，节点 (1,1) 的闭反外邻域中的颜色总数为$\lfloor\frac{K}{a_1+a_2+2}\rfloor(a_1+a_2+2)+a_2$。因此，

$$\mathcal{X}_l\leq\lfloor\frac{K}{a_1+a_2+2}\rfloor(a_1+a_2+2)+a_2.$$

由于我们只使用 K 种颜色，

$$\mathcal{X}_l\leq K.$$

因此，

$$\mathcal{X}_l\leq\min\{\lfloor\frac{K}{a_1+a_2+2}\rfloor(a_1+a_2+2)+a_2,K\}.$$

由于我们将每个文件分成 $\lfloor\frac{K}{a_1+a_2+2}\rfloor$ 个大小相等的子文件，

$$\overline{R}^*\leq\overline{R}_3=\frac{\min\{\lfloor\frac{K}{a_1+a_2+2}\rfloor(a_1+a_2+2)+a_2,K\}}{\lfloor\frac{K}{a_1+a_2+2}\rfloor}.$$

□

附录 C
证明第节定理VI 的证明

证明。我们先给出证明概要，再详述细节。首先，我们按照放置策略存储文件。在传输阶段，我们构造一个包含 K 个用户的索引编码问题实例，其中每个用户 j 想要文件 $\mathcal{F}_{d_j}$ 中不可用的 $K-iL$ 子文件。用户 j 的侧信息是其余 $K-1$ 个用户所需、并存储在用户 j 的可用缓存 $[j$ 中的子文件：$j+L-1]$。我们证明所构造的索引编码问题是若干个 $(a_1,a_2)_L$ – 索引编码问题的并集，并利用第 IV 节的结果得到这些索引编码问题最优传输率的上界。最后，我们的 MACC ICP 的最优传输率不超过各个 $(a_1,a_2)_z$ – 索引编码问题上界之和。第 IV 节中的不同结果给出不同的上界。具体而言，定理 5 的结果推出定理7，引理 1 的结果推出定理 8，而定理 6 的结果推出定理 9。详情如下。

回顾一下，在放置策略中，我们将每个文件$\mathcal{F}_n\ n\in[N]$ 划分为 K 个大小相同且互不相交的子文件$\mathcal{F}_{n,\ m}\ m\in[K]$子文件$\mathcal{F}_{n,\ m}$ 存储在缓存 $m,\ m+L,\ .....\ m+(i-1)L$中。因此，缓存 $k\ k\in[K]$ 存储的子文件为 $\mathcal{M}_k=\{\mathcal{F}_{n,<k-(r-1)L>_K}:\ n\in[N],\ r\in[i]\}$。

放置策略的一些有用性质如下：

- 每个缓存都存储每个文件的 i 个子文件。因此，满足内存约束 $M=iN/K$。
- 任意两个缓存存储的子文件集合都不相同
- 每个子文件恰好可供 iL 个连续用户使用，且任意两个子文件可用的缓存集合都不相同
- 每个用户都可以访问 iL 个子文件，且任意两个用户可用的子文件集合都不相同

由于子文件 $\mathcal{F}_{n,\ m}$ 存储在缓存 $m,\ m+L,\ .....,\ m+(i-1)L$中，因此，索引属于集合 $[<m-L+1>_K:\ <m+(i-1)L>_K]$的用户都可以获取该子文件。因此，$\mathcal{F}_{n,\ m}$ 表示为 $\mathcal{F}_{n,\ [<m-L+1>_K:\ <m+(i-1)L>_K]}$，其中，下标集合中的索引表示相应用户处可用的子文件。对于集合 $[K]$ 的每个包含 iL 个循环连续元素的子集，都存在一个子文件，反之亦然。

令请求模式为 $\mathbf{d}=(d_1,d_2,...,d_K)\in[N]^K$，即用户 j 请求文件 $\mathcal{F}_{d_j}$。换言之，用户 $j\ j\in[K]$ 请求文件 $\mathcal{F}_{d_j}$ 的 K 个子文件。根据我们的放置策略，用户 j 已经接入了 iL 个子文件，因此只需要文件 $\mathcal{F}_{d_j}$剩余的 $K-iL$ 个子文件。特别地，用户 j 已经接入了 $\mathcal{F}_{d_j,<j+l-1-(r-1)L>_K,\ r\in[i],\ l\in[L]}$，还需要剩余的 $K-iL$ 个子文件。用户 j 所需的子文件为 $\mathcal{F}_{d_j,[<j+a>_K:<j+a+iL-1>_K]}$，

$\forall a \in [K-iL]$。在考虑最坏情况时，假设每个用户请求不同的文件，则所有用户总共需要 $K(K-iL)$ 个子文件。

现在，我们将这里的问题映射为一个包含 K 个用户的索引编码问题实例，使得每个用户 j 需要文件 $\mathcal{F}_{d_j}$ 中不可用的 $K-iL$ 个子文件。用户 j 的侧信息是其余 $K-1$ 个用户所需、并存储在用户 j 可用缓存 $[j$ 中的子文件：$j+L-1]$。为了理解索引编码问题的结构性质，我们构造一个 $K \times K-iL$ 表，使得

1）每个单元格代表一个虚拟用户，

2） p^{th} 行代表用户 p 所需的子文件，

3）节点（p，q）代表子文件 $\mathcal{F}_{d_p,\ [<p+q>_K:\ <p+q+iL-1>_K]}$。见表 III 中的示例。

$\mathcal{F}_{d_1,\ [2:\ iL+1]}$	$\mathcal{F}_{d_1[3:\ iL+2]}$	...	$\mathcal{F}_{d_1[K-iL+1:\ K]}$
$\mathcal{F}_{d_2[3:\ iL+2]}$	$\mathcal{F}_{d_2[4:\ iL+3]}$	...	$\mathcal{F}_{d_2,\ [K-iL+2:\ 1]}$
⋮	⋮	⋮	⋮
$\mathcal{F}_{d_K,\ [1:\ iL]}$	$\mathcal{F}_{d_K,\ [2:\ iL+1]}$	.....	$\mathcal{F}_{d_K,\ [K-iL:\ K-1]}$

表III由 *MACC*问题形成的索引编码问题的表格表示。

表 III 表示一个包含 K 个用户且每个用户请求 $K-iL$ 个子文件的统一索引编码问题。我们按如下方式将该统一索引编码问题转换为对称统一索引编码问题：在表 III中，每个单元格表示由索引编码问题用户请求的一个子文件。现在，将每个节点作为对称统一索引编码问题中的一个虚拟用户，请求相应的子文件；其侧信息与统一索引编码问题中由该行表示的用户相同。

由于一行中的所有节点均由同一个用户请求，因此该行中所有节点的侧信息节点都相同。特别地，所有下标中包含 l 的节点在用户 l 处均可用，因而它们是行 l 中节点的侧信息节点。上述性质 3意味着，一列中所有节点的可用子文件的相对位置都相同，因此每一列单独表示一个对称索引编码问题 [25]。注意：(i) 每个子文件的下标集合都是 $[K]$的子集，其中包含 iL 个连续元素；(ii) 在每一列中，下标从第 j 行到第 $j+1$行循环移位一个位置。因此，对于任意一列，每个元素$k \in [K]$ 都会出现在 iL 个连续条目中。由此，一列中任意节点的侧信息也遵循相同的模式。因此，每一列表示某些 a_1、$a_2 \in \mathbb{N}$ 和 $a_1+a_2 = K-iL-1$对应的 $(a_1,\ a_2)_{iL}$−索引编码问题。特别地，第 j列表示 $(K-iL-j,\ j-1)_{iL}$−索引编码问题。因此，根据 $(a_1,\ a_2)_z$−索引编码问题的定义，第 j 列和第 $K-iL-j+1$ 列共同表示 $(K-iL-j,\ j-1)_{iL}$−索引编码问题 $\forall j \le \frac{K-iL}{2}$。现在，我们利用第 IV 节中的结果，给出每个 $(K-iL-j,\ j-1)_{iL}$−索引编码问题的最优传输率上界。表 III 中索引编码问题的最优传输率由各个 $(K-iL-j,\ j-1)_{iL}$−索引编码问题的上界之和作为上界。

令 R_j^* 为 $(K-iL-j,\ j-1)_{iL}$−索引编码问题 $\forall j \le \frac{K-iL}{2}$ 的最优传输率，并令 R_T^* 为表III中索引编码问题的最优传输率。那么，R_T^* 有如下上界：

- 如果 $K-iL$ 为偶数，则表III 中的索引编码问题可表示为恰好 $\frac{K-iL}{2}$ $(a_1,\ a_2)_z$−个索引编码问题的并集，具体为 $(K-iL-j,\ j-1)_{iL}$−索引编码问题 $\forall j \le \frac{K-iL}{2}$。因此，

$$R_T^* \le \sum_{j=1}^{\frac{K-iL}{2}} R_j^*. \tag{6}$$

- 如果 $K-iL$ 为奇数，则表 III 中的索引编码问题可以表示为以下各项的并集：$\frac{K-iL-1}{2}$ $(a_1,\ a_2)_z$−个索引编码问题，分别由 $(K-iL-j,\ j-1)_L$−索引编码问题 $\forall j \le \frac{K-iL}{2}$和一个由 $(a_1,\ a_2)_z$−给出的 $(\frac{K-iL-1}{2},\ \frac{K-iL-1}{2})_{iL}$−索引编码问题组成。因此，

$$R_T^* \le \sum_{j=1}^{\frac{K-iL-1}{2}} R_j^* + R'_{\frac{K-iL-1}{2}}, \tag{7}$$

其中，$R'_{\frac{K-iL-1}{2}}$ 是以下对象的最优传输率 $(\frac{K-iL-1}{2}, \frac{K-iL-1}{2})_{iL}$−ICP.

现在，我们证明定理 7。

在一个 $(K-iL-j,\ j-1)_{iL}$−索引编码问题中，$a_1 = K-iL-j$ 且$a_2 = j-1$。然后，根据定理 5，如果存在一个整数 X 使得 $X \ge a_1+a_2+2 = K-iL+1$且 $X|K$，则 $R_j^* \le X$。因此，

- 如果 $K-iL$ 是偶数，那么根据（6），我们有

$$R_T^* \le \sum_{j=1}^{\frac{K-iL}{2}} R_j^* = \frac{(K-iL)X}{2}.$$

- 如果 $K-iL$ 为奇数，则我们可以通过将每个文件拆分为两部分，从 $(\frac{K-iL-1}{2}, \frac{K-iL-1}{2})_L$−索引编码问题构造一个 $(\frac{K-iL-1}{2}, \frac{K-iL-1}{2})_L$−索引编码问题，并证明 $(\frac{K-iL-1}{2}, \frac{K-iL-1}{2})_L$−索引编码问题的最优传输率有界于 $X/2$，若存在一个 X 使得 $X \ge a_1+a_2+2 = K-iL+1$ 且 $X|K$。因此，根据（7），我们有

$$R_T^* \le \sum_{j=1}^{\frac{K-iL-1}{2}} R_j^* + R_{\frac{K-iL-1}{2}} = \frac{(K-iL-1)X}{2} + \frac{X}{2}.$$

请注意，由于我们将 $(\frac{K-iL-1}{2}, \frac{K-iL-1}{2})_L$−索引编码问题中的每个子文件分成 2 个相等的部分，奇数情形下的子分包为 $K+1$，仍然是线性的。

由于每个子文件的大小为 $1/K$ 单位，因此 MACC问题的最优传输率上界为 R_T^*/K，由此完成定理7的证明。

现在，我们证明定理8。

- $(K-iL+1)|K$ 情形可直接由定理7得出。

- $K-iL=1$ 情形构成一个索引编码问题，其中每个用户需要一个子文件，并拥有所有其他用户所需的子文件。因此，侧信息图构成一个团，只需发送一条大小为 $1/K$ 单位的消息即可解决 MACC问题。

- 如果 $K-iL$ 是偶数，则根据（6），我们有

$$R_T^* \le \sum_{j=1}^{\frac{K-iL}{2}} R_j^*.$$

根据引理4，我们有

$$R_j^* = \min\{a_1+2a_2+2, K\} = \min\{K-iL+j, K\}.$$

将 R_j^* 代入（6），经过简单的代数运算，可得 $\frac{R_T^*}{K} = R_e(M)$。 —

- 如果 $K-iL$ 为奇数，则根据（7），有

$$R_T^* \le \sum_{j=1}^{\frac{K-iL-1}{2}} R_j^* + R'_{\frac{K-iL-1}{2}}.$$

根据引理4，有

$$R_j^* = \min\{a_1+2a_2+2, K\} = \min\{K-iL+j, K\},$$

and

$$+R'_{\frac{K-iL-1}{2}} = K-iL.$$

将 R_j^* 和 $+R'_{\frac{K-iL-1}{2}}$ 代入（7），经过简单的代数运算，可得 $\frac{R_T^*}{K} = R_o(M)$。定理8 的证明至此完成。

同样地，如果使用定理5 的结果来界定表R_T^*中 MACC ICP 的传输速率（R_T^*），可得定理9 的上界。由于我们将每个文件划分为 a_1+a_2+2 个子文件以得到5，因此子分包为~$F_5(M)$。

□

附录D 示例

下面的示例说明了我们的策略。

例5。 考虑一个包含 $N=K=8$、$L=2$ 和 $M=i=3$ 的 *MACC*问题。放置策略：

- 将每个文件 $\mathcal{F}_n$ $\forall n \in [8]$ 分成 $K=8$个子文件$\mathcal{F}_{n,1}$, $\mathcal{F}_{n,2}$, $\mathcal{F}_{n,3}$, $\mathcal{F}_{n,4}$, $\mathcal{F}_{n,5}$,$\mathcal{F}_{n,6}$, $\mathcal{F}_{n,7}$ 和 $\mathcal{F}_{n,8}$。
- 缓存 k 存储每个文件的 $i=3$ 子文件，它们由 $\mathcal{F}_{n,\ k}$、$\mathcal{F}_{n,\ <k-2>_8}$ 和 $\mathcal{F}_{n,\ <k-4>_8}$、 $\forall n \in [8]$ 给出。

注意，根据我们的放置策略，$\mathcal{F}_{n,m} \forall n, m \in [8]$ 存储在缓存 m, $< m+2 >_8$ 和 $< m+4 >_8$ 中，并可供集合 $[< m-1 >_8 : < m+4 >_8]$中的用户使用：$< m-1 >_8, m$ 连接到缓存 m，用户$< m+1 >_8$, $< m+2 >_8$ 连接到缓存$< m+2 >_8$，而用户 $< m+2 >_8$, $< m+4 >_8$连接到缓存 $< m+3 >_8$。因此，$\mathcal{F}_{n,m}$ 也可以表示为 $\mathcal{F}_{n,[<m-1>_8 : <m+4>_8]}$，其中带下标的集合 $[< m-1 >_8 : < m+4 >_8]$ 表示子文件$\mathcal{F}_{n,\ [<m-1>_8 : <m+4>_8]}$ 可供集合 $[< m-1 >_8 : < m+4 >_8]$中的用户使用。注意，集合$[< m-1 >_8 : < m+4 >_8]$ 是由 $iL=6$个循环连续的[4] 元素组成的集合。

传输策略：

•设请求模式（$d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8$）。 *be*

- 每个用户 j 需要文件 $\mathcal{F}_{d_j}$ 的 *8* 个子文件，并且已经接入了 $iL=6$个子文件。因此，用户 j 还需要文件 $\mathcal{F}_{d_j}$ 的 *2* 个子文件，具体如下。

[4]在我们的论文中，循环连续指的是通过循环回绕形成的连续。$\mathcal{F}_{d_j,\ \{<j+1>_8, <j+2>_8, <j+3>_8, <j+4>_8, <j+5>_8, <j+6>_8\}}$，以及 $\mathcal{F}_{d_j,\ \{<j+2>_8, <j+3>_8, <j+4>_8, <j+5>_8, <j+6>_8, <j+7>_8\}}$，即，

- *User 1 needs* $\mathcal{F}_{d_1,\{2,3,4,5,6,7\}}$, *and* $\mathcal{F}_{d_1,\{3,4,5,6,7,8\}}$,
- *User 2 needs* $\mathcal{F}_{d_2,\{3,4,5,6,7,8\}}$, *and* $\mathcal{F}_{d_2,\{4,5,6,7,8,1\}}$,
- *User 3 needs* $\mathcal{F}_{d_3,\{4,5,6,7,8,1\}}$, *and* $\mathcal{F}_{d_3,\{5,6,7,8,1,2\}}$,
- *User 4 needs* $\mathcal{F}_{d_4,\{5,6,7,8,1,2\}}$, *and* $\mathcal{F}_{d_4,\{6,7,8,1,2,3\}}$,
- *User 5 needs* $\mathcal{F}_{d_5,\{6,7,8,1,2,3\}}$, *and* $\mathcal{F}_{d_5,\{7,8,1,2,3,4\}}$,
- *User 6 needs* $\mathcal{F}_{d_6,\{7,8,1,2,3,4\}}$, *and* $\mathcal{F}_{d_6,\{8,1,2,3,4,5\}}$,
- *User 7 needs* $\mathcal{F}_{d_7,\{8,1,2,3,4,5\}}$, *and* $\mathcal{F}_{d_7,\{1,2,3,4,5,6\}}$, *and*
- *User 8 needs* $\mathcal{F}_{d_8,\{1,2,3,4,5,6\}}$, *and* $\mathcal{F}_{d_8,\{2,3,4,5,6,7\}}$.

- 现在，我们将编码缓存问题映射为一个索引编码问题，并将该索引编码问题表示为如下表格形式：

$\mathcal{F}_{d_1,\{2,3,4,5,6,7\}}$	$\mathcal{F}_{d_1,\{3,4,5,6,7,8\}}$
$\mathcal{F}_{d_2,\ \{3,4,5,6,7,8\}}$	$\mathcal{F}_{d_2,\ \{4,5,6,7,8,1\}}$
$\mathcal{F}_{d_3,\ \{4,5,6,7,8,1\}}$	$\mathcal{F}_{d_3,\ \{5,6,7,8,1,2\}}$
$\mathcal{F}_{d_4,\ \{5,6,7,8,1,2\}}$	$\mathcal{F}_{d_4,\ \{6,7,8,1,2,3\}}$
$\mathcal{F}_{d_5,\{6,7,8,1,2,3\}}$	$\mathcal{F}_{d_5,\{7,8,1,2,3,4\}}$
$\mathcal{F}_{d_6,\{7,8,1,2,3,4\}}$	$\mathcal{F}_{d_6,\{8,1,2,3,4,5\}}$
$\mathcal{F}_{d_7,\{8,1,2,3,4,5\}}$	$\mathcal{F}_{d_7,\{1,2,3,4,5,6\}}$
$\mathcal{F}_{d_8,\{1,2,3,4,5,6\}}$	$\mathcal{F}_{d_8,\{2,3,4,5,6,7\}}$
$(1,\ 0)_6$−索引编码问题	$(0,\ 1)_6$−索引编码问题

表 IV $N=8$ $K=8$-CCDN 的索引编码问题。行 p 包含用户 $p, \forall p \in [8]$ 所需的子文件。最后一行注明了由相应列中的节点构成的索引编码问题类型。我们用红色标出用户 1 所需的子文件，用蓝色标出其侧信息子文件。请注意，所有蓝色子文件的下标集合中都包含 1。因此，蓝色节点是行 1 节点的侧信息节点。

服务器传输中涉及总共 *16* 个子文件。我们可以将此处的问题映射为一个索引编码问题实例，其中有 $K=16$ 个虚拟用户，每个虚拟用户请求一个不同的子文件。代表（并请求）某个子文件 X 的虚拟用户所拥有的旁信息，是请求子文件 X的用户可用的子文件。例如，代表子文件 $\mathcal{F}_{d_1,\ \{2,3,4,5,6,7\}}$ 的虚拟用户的旁信息，是用户 *1* 可用的子文件，即$\mathcal{F}_{d_2,\ \{4,5,6,7,8,1\}}$,$\mathcal{F}_{d_3,\ \{5,6,7,8,1,2\}}$, $\mathcal{F}_{d_3,\ \{6,7,8,1,2,3\}}$, ...。我们可以求解这个索引编码问题，以获得所提出方案的可达传输率。请注意，*CCDN* 中中央服务器传输的消息数量等于索引编码问题中的消息数量，且每条消息的大小等于一个子文件的大小。

为了理解上述索引编码问题的结构性质，我们构造一个 8×2 表（见表*IV*），其中 p^{th} 行和 q^{th} 列包含子文件$F_{d_p,\ [<p+q>_K : <p+q+5>_K]}$，即用户 p 请求且用户 $< p+q >$、$< p+q+1 >$、$< p+q+2 >$、 $< p+q+4 >_K$可用的子文件，

$< p+q+5 >_K$ 和 $< p+q+6 >_K$。我们将此项称为节点（p, q），其中，$p \in [8], q \in [2]$。注意，第 p 行中的项是用户 p所需的子文件。

因此，我们将编码缓存问题映射为一个索引编码问题，并在表*IV*中以表格形式表示该索引编码问题。所构成的索引编码问题是对称 *ICP*，其定义如下：若节点 (p_1, q_1) 是节点 (p_2, q_2) 的侧信息节点（或干扰节点），则节点 $(< p_1 + 1 >_K, q_1)$ 是节点 $(< p_2 + 1 >_K, q_2)$ 的侧信息节点（或干扰节点），反之亦然。为说明所构成的索引编码问题具有对称性，我们在表$(N = 8, K = 8)-$中展示对应于 *CCDN* 的索引编码问题：表 *IV* 突出显示用户 *1* 请求的文件及其侧信息文件，表 *V* 则突出显示用户 *2* 请求的文件及其侧信息文件。注意，在表格的第 *1* 列中，用户 *1* 需要子文件 $\mathcal{F}_{d_1,\ \{2,3,4,5,6,7\}}$ 其后是 *1* 个干扰节点，再后面是用户 *1* 的 *6* 个侧信息节点。表*IV*对此进行了突出显示。同样，用户 *2* 需要子文件 $\mathcal{F}_{d_2,\ \{3,4,5,6,7\}}$ 其后是 *1* 个干扰节点，再后面是用户 *2* 的 *6* 个侧信息节点。表 *V* 对此进行了突出显示。由于对称性，在这些表格的第 *1* 列中，用户 i 请求 $\mathcal{F}_{d_i,\ [<i+1>_8:\ <i+6>_8]}$其后是 *1* 个干扰节点，再后面是用户 i 的 *6* 个侧信息节点。因此，第 *1* 列表示 $(a_1 = 1, a_2 = 0)_6-$索引编码问题。同样，我们也可以检查第 *2* 列中的

这些表格中，用户 i 请求 $\mathcal{F}_{d_i,\ [<i+2>_8:\ <i+7>_8]}$ 其后有 *6* 个侧信息节点，再后是用户 i的 *1* 个干扰节点。因此，第 *2* 列表示 $(a_1 = 0, a_2 = 1)_6-$索引编码问题。这两个索引编码问题的并集为 $(a_1 = 1, a_2 = 0)_6-$索引编码问题。因此，表 *IV* 和 *V* 表示 $(a_1 = 1, a_2 = 0)_6-$索引编码问题。

$\mathcal{F}_{d_1,\ \{2,3,4,5,6,7\}}$	$\mathcal{F}_{d_1,\ \{3,4,5,6,7,8\}}$
$\mathcal{F}_{d_2,\ \{3,4,5,6,7,8\}}$	$\mathcal{F}_{d_2,\ \{4,5,6,7,8,1\}}$
$\mathcal{F}_{d_3,\ \{4,5,6,7,8,1\}}$	$\mathcal{F}_{d_3,\ \{5,6,7,8,1,2\}}$
$\mathcal{F}_{d_4,\ \{5,6,7,8,1,2\}}$	$\mathcal{F}_{d_4,\ \{6,7,8,1,2,3\}}$
$\mathcal{F}_{d_5,\ \{6,7,8,1,2,3\}}$	$\mathcal{F}_{d_5,\ \{7,8,1,2,3,4\}}$
$\mathcal{F}_{d_6,\ \{7,8,1,2,3,4\}}$	$\mathcal{F}_{d_6,\ \{8,1,2,3,4,5\}}$
$\mathcal{F}_{d_7,\{8,1,2,3,4,5\}}$	$\mathcal{F}_{d_7,\{1,2,3,4,5,6\}}$
$\mathcal{F}_{d_8,\{1,2,3,4,5,6\}}$	$\mathcal{F}_{d_8,\{2,3,4,5,6,7\}}$
$(1,\ 0)_6-$索引编码问题	$(0,\ 1)_6-$索引编码问题

表 V：$(N = 8,\ K = 8)$-CCDN 的索引编码问题。用户 2 所需的子文件以红色标出，其侧信息子文件以蓝色标出。请注意，所有蓝色子文件的下标集合中都包含 2。因此，蓝色节点是第 2 行节点的侧信息节点。

根据引理 *4*，$R_1 = 3$ *3* 条消息就足以解决给定的多接入编码缓存问题。由于每个子文件的大小为 $1/K$ 单位，因此我们的传输速率为 $3/8$单位。因此，我们的传输速率为 $3/8$ 单位，子分包数为 $F = K = 8$；而根据引理 *2*，传输速率为 $R_2(M) = 0.5$ 单位，子分包数为 $F = 4K = 32$。