极化码的编码方法和译码方法、及编码装置和译码装置

本申请要求于2020年09月17日提交俄罗斯专利局、申请号为2020130551、申请名称为“极化码的编码方法和译码方法、及编码装置和译码装置”的专利申请的优先权，其全部内容通过引用结合在本申请中。

技术领域

本申请涉及信道编码领域，更具体地，涉及一种极化码的编码方法和译码方法、及编码装置和译码装置。

背景技术

极化码（polar codes）已能够被严格证明达到信道容量的信道编码方案，具有高性能，较低复杂度，匹配方式灵活等特点，目前已经被第三代合作伙伴计划（3rd generation partnership project，3GPP）确定成为第五代（the 5thgeneration, 5G）的一些通信场景中控制信道编码方案。

原始的极化码基于核矩阵 $F_2=\begin{pmatrix}1 & 0\\ 1 & 1\end{pmatrix}$，它的 n 次克罗内克指数 $F_2^{\otimes m}$ 对应一个码长为 $N=2^m$ 的线性码。推广到高维核矩阵，研究者们证明，采用某些设计的比较好的高维核矩阵对极化码进行编码能够带来更高的极化速度，即更好的译码性能。但是高维核矩阵会带来较高的译码复杂度，如何降低译码复杂度是亟需解决的问题。

发明内容

本申请提供一种极化码的编码方法和译码方法、及编码装置和译码装置，以期降低译码复杂度。

第一方面，提供了一种编码方法。该方法可以由编码端设备执行，或者，也可以由配置于编码端设备中的芯片或芯片系统或电路执行，本申请对此不作限定。

该方法可以包括：获取信息比特；确定目标核矩阵，所述目标核矩阵是通过调整原始核矩阵获得的；基于所述目标核矩阵，对所述信息比特进行极化编码。

示例地，获取信息比特，包括内部产生信息比特，也可以包括从外部接收信息比特。

基于上述技术方案，采用极化编码时，可以基于原始核矩阵构造出多个核矩阵，基于构造出的核矩阵，根据实际通信情况，选择合适的目标核矩阵进行编码。通过该方式，在核矩阵构造或选择时，就综合考虑性能和译码复杂度，能够实现通过选择合适的编码时所使用的核矩阵，降低译码复杂度。

结合第一方面，在第一方面的某些实现方式中，所述目标核矩阵的维度小于所述原始核矩阵的维度。

基于上述技术方案，在已知维度的核矩阵的基础上，构造出多个相对较小维度的核矩阵。基于小维度的核矩阵进行编码，能在保证相同性能的情况下，提供更低的译码复杂度。

结合第一方面，在第一方面的某些实现方式中，所述目标核矩阵是通过调整所述原始核矩阵的部分距离 partial distance 获得的。

示例地，所述目标核矩阵是通过调整所述原始核矩阵的 partial distance profile (PDP) 获得的。

示例地，调整所述原始核矩阵的 partial distance 时，可以按照接近 Arikan 矩阵的 partial distance 进行调整。

结合第一方面，在第一方面的某些实现方式中，所述目标核矩阵是通过调整所述原始核矩阵的 partial distance，并且基于深度优先搜索算法获得的。

结合第一方面，在第一方面的某些实现方式中，在基于所述深度优先搜索算法进行处理时，使用以下一项或多项约束条件进行处理：

候选行集合 M 满足：$M[i]=\left\{v_0^{l-1}\in F_2^l \mid wt\left(v_0^{l-1}\right)=D_i\right\}$；或者，

从所述候选行集合 M 中选择倒数第 i 行时，选择出来的行 v 汉明重量大于或等于 D_i；或者，

在从所述候选行集合 M 中选择出所述行 v，并将其跟已构建好的行进行 partial distance 计算，如果 $c\in C_{\overline{K}}^{(\varphi+1)}\oplus\overline{K}[\varphi]$，$wt(c)<D_\varphi$，那么直接抛弃所述行 v，从所述候选行集合 M 中选择新的行；或者，

属于同一陪集的行不会再次被选中；

其中，i 用于表示行号，$i\in[0,l-1]$；$wt(v)$和$wt(c)$均表示汉明重量；$\overline{K}$表示当前构建的核矩阵 $\overline{K}$，$\overline{K}[\varphi]$表示核矩阵 $\overline{K}$ 的第 φ 行；D_i和D_φ均表示 partial distance；$C_{\overline{K}}^{(\varphi+1)}$表示核矩阵 $\overline{K}$ 对应的码；$\oplus$表示逻辑运算。

示例地，当前构建的核矩阵 $\overline{K}$，可以表示核 $\overline{K}$ 正在构建；或者核 $\overline{K}$ 构建好后，可以表示已构建完成的核 $\overline{K}$。

基于上述技术方案，可以通过设计关于候选行和/或所选行的约束条件，以便加快收敛速度，可以进一步节省计算量。

结合第一方面，在第一方面的某些实现方式中，所述原始核矩阵为维度为(24×24)的核矩阵 K_{24}，所述 K_{24} 为：

1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0
1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

$$\begin{pmatrix}
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0
\end{pmatrix}$$

结合第一方面，在第一方面的某些实现方式中，所述目标核矩阵为以下任一项：

$$K_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&1&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&1&1&0&0&1&1&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&1&1&1&1&1&1&1&1
\end{pmatrix}$$

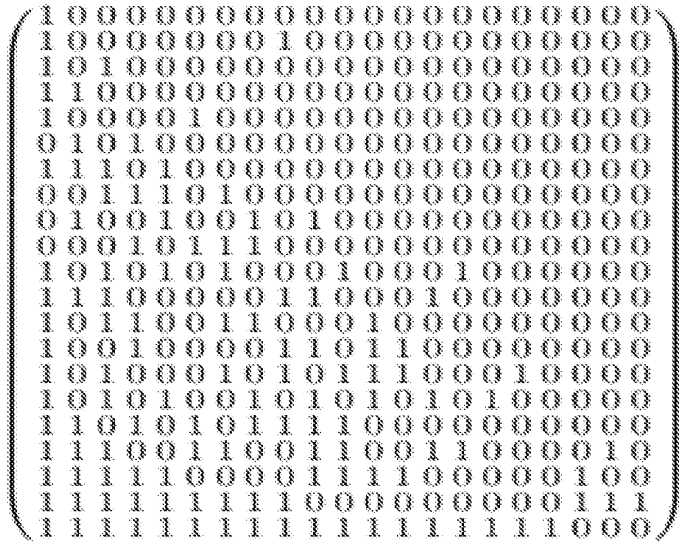
；或者，

$$K'_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&0&0&1&0&1&0&1&0&1&0&0&0&0&0\\
1&0&1&0&1&1&0&0&0&0&0&0&1&0&1&0&1&0&1&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&1&1\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0
\end{pmatrix}$$

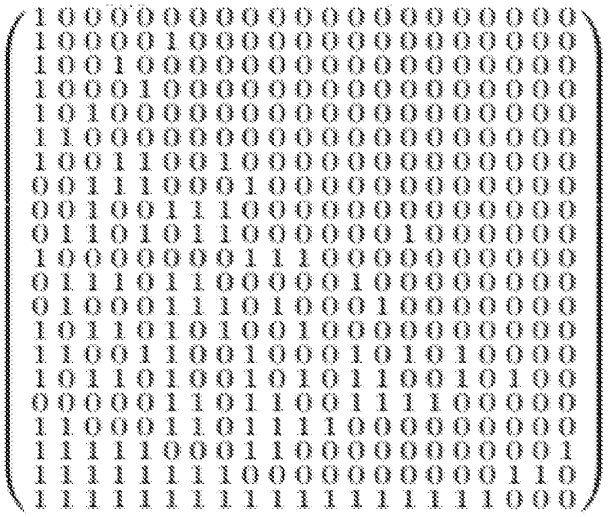
；或者，

$$K''_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&1&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&1&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0\\
1&1&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&0&1&0&0&1&0&0&0&1&0&0&0&1&0&1&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&0&0&0&0\\
0&0&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0
\end{pmatrix}$$

；或者，

K_{21}= 

；或者，

K'_{21}= 

；或者，

$$K_{24} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0
\end{pmatrix}$$

；或者，

$$K'_{24} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&1&0&1&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&1&1&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0
\end{pmatrix}$$

；或者，

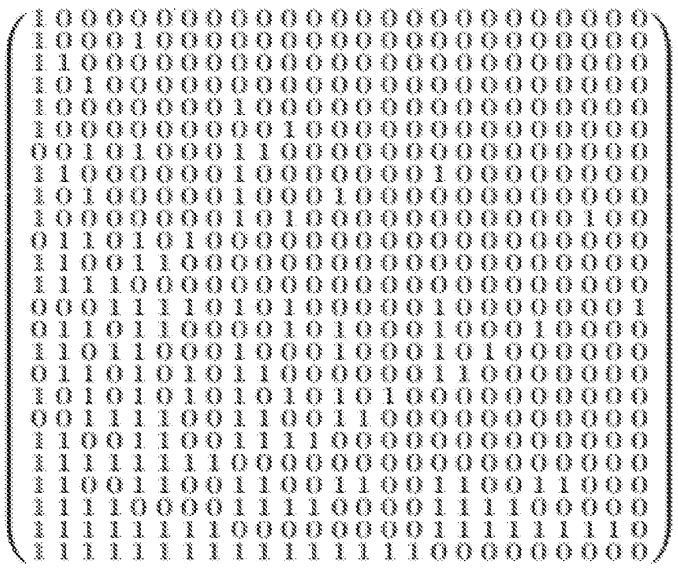
$$K_{24}'' = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&1&1&0&1&0&0&0&1&0&0&0&1&0&0&0\\
0&1&1&1&0&0&1&0&1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0
\end{pmatrix}$$

；或者，

$$K_{25} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&1&0&0&0&0&1&0&0&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&1&1&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0\\
0&0&1&0&1&1&0&1&1&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&0&1&0&1&1&0&1&0&1&1&0&0&0&0&1&0&0&0&0\\
1&0&1&1&0&1&1&1&1&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0\\
1&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&0&0&1&0&1&0&1&0\\
1&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&0&1&0&0&1&1&0&0\\
1&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0
\end{pmatrix}$$

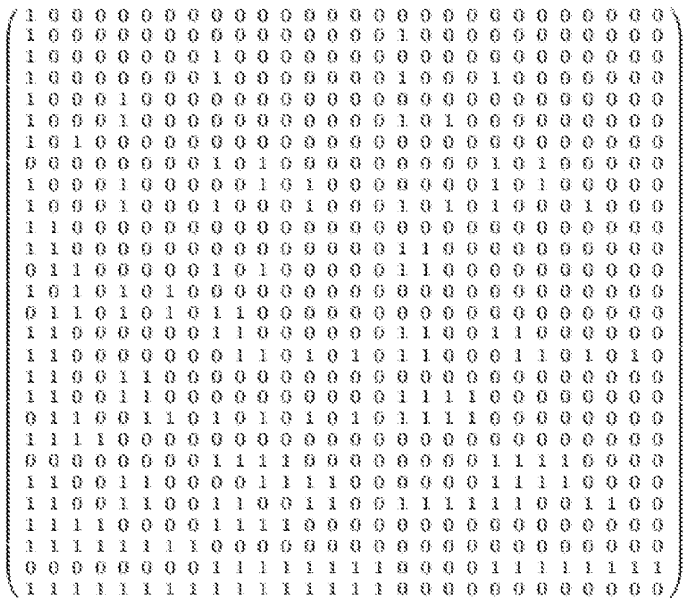
；或者，

$$K'_{25}=$$



；或者，

$$K_{28}=$$



；或者，

$$K_{32} = \left(\begin{array}{cccccccccccccccccccccccccccccccc}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&0&0&0&0&1&0&1&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&1&1&0&0&0&0&0&0&1&1&0&1&0&1&0&1&1&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&0&1&1&0&1&0&1&0&1&0&1&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1
\end{array}\right)$$

。

第二方面，提供了一种译码方法。该方法可以由译码端设备执行，或者，也可以由配置于译码端设备中的芯片或芯片系统或电路执行，本申请对此不作限定。

该方法可以包括：获取待译码序列；基于多个网格，对所述待译码序列进行译码，基于所述多个网格中的第二网格译码时，复用所述多个网格中的第一网格译码的中间结果，其中，所述第一网格用于对所述待译码序列译码的第 t 阶段，所述第二网格用于对所述待译码序列译码的第 t+i 阶段，t 为大于 0 或等于 0 的整数，i 为大于 1 或等于 1 的整数。

示例地，获取待译码序列，可以表示从外部接收待译码序列。

示例地，第一网格对应译码第 t 阶段，也就是说，在译码第 t 阶段，基于第一网格进行译码处理；或者说在对待译码序列的某一信息比特进行译码时，基于第一网格进行译码处理。示例地，第二网格对应译码第 t+i 阶段，也就是说，在译码第 t+i 阶段，基于第二网格进行译码处理；或者说在对待译码序列的又一信息比特进行译码时，基于第二网格进行译码处理。

示例地，第一网格译码的中间结果，即表示基于第一网格处理得到的结果，或者说，在译码第 t 阶段译码得到的结果。其中，中间结果例如可以包括：基于所述第一网格译码的译码路径和对应的路径度量值。

示例地，更小一级的网格图中所有可能路径拼接后得到多个组合，通过选择这些可能组合中路径分数最大的那个得出更大一级的网格图最优路径及路径值。

基于上述技术方案，采用网格 Trellis 的译码算法对待译码序列进行译码时，不同译码阶段得到的中间结果可复用。也就是说，在不同的网格上进行计算时，有些计算步骤可能相同，所以可以多次复用，这使得译码复杂度得到了很大程度的降低，且提高了译码速度。

结合第二方面，在第二方面的某些实现方式中，所述多个网格中的每个网格被分割为多个子网格。

基于上述技术方案，采用网格 Trellis 的译码算法对待译码序列进行译码时，可以先对

网格图进行分割，把在完整的 Trellis 上进行译码的过程分割成对子网格 (sub trellis)上进行译码的过程，从而可以降低译码复杂度。

结合第二方面，在第二方面的某些实现方式中，所述多个子网格的分割点的位置与译码复杂度相关。

也就是说，将网格进行分割的分割点的位置与译码复杂度相关。

基于上述技术方案，可以根据实际情况，选择合适的分割点的位置。分割点的位置可能会影响译码复杂度，因此，可以选择某些分割点，以便降低译码复杂度。

结合第二方面，在第二方面的某些实现方式中，所述方法还包括：根据所述第一网格和所述第二网格的关联度，确定基于所述第二网格译码时是否复用所述第一网格译码的中间结果。

示例地，第一网格和第二网格的关联度，可以表示第一网格和第二网格是否相同或者相似；或者，可以体现第一网格对应的多个子网格和第二网格对应的多个子网格是否相同或者相似；或者，也可以表示基于第一网格和第二网格处理时，是否有相同的计算步骤。

基于上述技术方案，可以根据网格之间的关联度，例如第一网格和第二网格是否相同或相似，确定在译码不同阶段得到的中间结果是否可被复用。

结合第二方面，在第二方面的某些实现方式中，所述译码方法还包括：根据以下信息，判断基于所述第二网格译码时是否复用所述第一网格译码的中间结果：所述译码的第 t+i 阶段通过对线性码 C 进行缩短操作得到的码与所述译码的第 t 阶段通过对线性码 C 进行缩短操作得到的码；和/或，所述译码的第 t+i 阶段通过对线性码 C 进行打孔操作得到的码与所述译码的第 t 阶段通过对线性码 C 进行打孔操作得到的码。

基于上述技术方案，可以通过比较在不同译码阶段对线性码 C 进行缩短操作得到的码和/或对线性码 C 进行打孔操作得到的码，确定在译码不同阶段得到的中间结果是否可被复用。

结合第二方面，在第二方面的某些实现方式中，如果 $s_{x,y}(\mathcal{C}^{(t+i)})=s_{x,y}(\mathcal{C}^{(t)})$，$p_{x,y}(\mathcal{C}^{(t+i)})\subset p_{x,y}(\mathcal{C}^{(t)})$，那么基于所述第二网格译码时，复用所述第一网格译码的中间结果；或者；如果 $s_{x,y}(\mathcal{C}^{(t+i)})\subset s_{x,y}(\mathcal{C}^{(t)})$，那么基于所述第二网格译码时，复用所述第一网格译码的中间结果；其中，$s_{x,y}(\mathcal{C}^{(t+i)})$表示译码的第 t+i 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行缩短操作得到的码，$s_{x,y}(\mathcal{C}^{(t)})$表示译码的第 t 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行缩短操作得到的码，$p_{x,y}(\mathcal{C}^{(t+i)})$表示译码的第 t+i 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行打孔操作得到的码，$p_{x,y}(\mathcal{C}^{(t)})$表示译码的第 t 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行打孔操作得到的码。

基于上述技术方案，可以根据上述条件，确定在译码不同阶段得到的中间结果是否可被复用。

结合第二方面，在第二方面的某些实现方式中，所述基于所述第二网格译码时，复用所述第一网格译码的中间结果，包括：在所述译码的第 t+i 阶段基于所述第二网格处理时，直接将基于所述第一网格译码的中间结果，作为所述第二网格译码的中间结果；或者，在

所述译码的第 t+i 阶段基于所述第二网格处理时，基于所述第一网格处理过程中的最大化操作的结果进行处理

结合第二方面，在第二方面的某些实现方式中，所述译码为极化译码。

也就是说，所述待译码序列为获取通过极化编码后的待译码序列。

基于上述技术方案，采用高维核矩阵进行编码时，采用常规网格算法进行译码时，polar 码译码器译码复杂度较高，通过本申请实施例，不仅可以通过在子网格上译码降低译码复杂度，而且该方案和采用 Arikan 核进行编码，SCL 进行译码的传统方案对比，能在保证相同性能的情况下，提供更低的译码复杂度。

第三方面，本申请提供一种编码装置，该编码装置具有实现上述第一方面及其任意可能的实现方式中的方法的功能。所述功能可以通过硬件实现，也可以通过硬件执行相应的软件实现。所述硬件或软件包括一个或多个与上述功能相对应的单元。

在一种可能的设计中，当所述功能的部分或全部通过硬件实现时，该编码装置包括：输入接口电路，用于获取信息比特；逻辑电路，用于执行上述第一方面中的编码方法，对所述信息比特进行极化编码；输出接口电路，用于输出编码后的序列。

可选的，该编码装置可以是芯片或者集成电路。

在一种可能的设计中，当所述功能的部分或全部通过软件实现时，该编码装置包括：存储器，用于存储计算机程序；处理器，用于执行存储器中存储的计算机程序，当所述计算机程序被执行时，该编码装置可以实现如上述第一方面所述的编码方法。

可选的，存储器可以是物理上独立的单元，也可以与处理器集成在一起。

在一种可能的设计中，当所述功能的部分或全部通过软件实现时，该编码装置包括仅包括处理器。用于存储程序的存储器位于编码装置之外，处理器通过电路/电线与存储器连接，用于读取并运行存储器中存储的程序，以执行上述第一方面中所述的编码方法。

在具体实现时，该编码装置可以为芯片或集成电路。

第四方面，本申请提供一种译码装置，该译码装置具有实现上述第二方面及其任意可能的实现方式中的方法的功能。所述功能可以通过硬件实现，也可以通过硬件执行相应的软件实现。所述硬件或软件包括一个或多个与上述功能相对应的单元。

在一种可能的设计中，当所述功能的部分或全部通过硬件实现时，该译码装置包括：输入接口电路，用于获取待译码序列；逻辑电路，用于执行上述第二方面中的译码方法，对所述待译码序列进行译码，得到译码结果；输出接口电路，用于输出译码结果。

可选的，该译码装置可以是芯片或者集成电路。

在一种可能的设计中，当所述功能的部分或全部通过软件实现时，该译码装置包括：存储器，用于存储计算机程序；处理器，用于执行存储器中存储的计算机程序，当所述计算机程序被执行时，该译码装置可以实现如上述第二方面所述的译码方法。

可选的，存储器可以是物理上独立的单元，也可以与处理器集成在一起。

在一种可能的设计中，当所述功能的部分或全部通过软件实现时，该译码装置包括仅包括处理器。用于存储程序的存储器位于译码装置之外，处理器通过电路/电线与存储器连接，用于读取并运行存储器中存储的程序，以执行上述第二方面中所述的译码方法。

在具体实现时，该译码装置可以为芯片或集成电路。

第五方面，本申请提供一种网络设备，包括收发器、处理器和存储器。处理器用于控

制收发器收发信号，存储器用于存储计算机程序，处理器用于调用并运行存储器中存储的计算机程序，使得网络设备执行第一方面任意可能的实现方式中的方法。

具体地，在网络设备作为信息和/或数据的发送端时，网络设备执行上述第一方面的编码方法，对发送的信息比特进行编码。

第六方面，本申请提供一种网络设备，包括收发器、处理器和存储器。处理器用于控制收发器收发信号，存储器用于存储计算机程序，处理器用于调用并运行存储器中存储的计算机程序，使得网络设备执行第二方面任意可能的实现方式中的方法。

具体地，在网络设备作为信息和/或数据的接收端时，网络设备执行上述第二方面的译码方法，对从发送端接收到的待译码序列进行译码。

第七方面，本申请提供一种终端设备，包括收发器、处理器和存储器。处理器用于控制收发器收发信号，存储器用于存储计算机程序，处理器用于调用并运行存储器中存储的计算机程序，使得终端设备执行第一方面任意可能的实现方式中的方法。

具体地，在终端设备作为信息和/或数据的发送端时，终端设备执行上述第一方面的编码方法，对发送的信息比特进行编码。

第八方面，本申请提供一种终端设备，包括收发器、处理器和存储器。处理器用于控制收发器收发信号，存储器用于存储计算机程序，处理器用于调用并运行存储器中存储的计算机程序，使得终端设备执行第二方面任意可能的实现方式中的方法。

具体地，在终端设备作为信息和/或数据的接收端时，终端设备执行上述第二方面的译码方法，对从发送端接收到的待译码序列进行译码。

第九方面，本申请提供一种计算机可读存储介质，该计算机可读存储介质中存储有指令，当其在计算机上运行时，使得计算机执行第一方面或第二方面中任意可能的实现方式中的方法。

第十方面，本申请提供一种计算机程序产品，该计算机程序产品包括计算机程序代码，当计算机程序代码在计算机上运行时，使得计算机执行上述第一方面或第二方面中任意一种可能的实现方式中的方法。

第十一方面，本申请提供一种芯片，包括逻辑电路和通信接口，所述通信接口用于接收待处理的数据和/或信息，并将所述待处理的数据和/或信息传输至所述逻辑电路，所述逻辑电路用于执行编码的处理，以及，所述通信接口还用于输出处理后的数据和/或信息。

具体地，所述芯片可以为配置在发送端中的芯片，所述待处理的数据和/或信息可以为信息比特。所述芯片通过通信接口获取信息比特，并将其传输至逻辑电路；所述逻辑电路采用第一方面描述的编码方法，对该信息比特进行编码；以及，所述芯片通过所述通信接口输出编码后的极化码字。

可选地，所述通信接口可以包括输入接口和输出接口。输入接口用于获取信息比特，所述输出接口用于输出编码后的极化码字。

可选地，所述芯片可以为配置在发送端中的芯片，所述逻辑电路用于生成信息比特，并采用第一方面描述的编码方法，对该信息比特进行编码；以及，所述芯片通过所述通信接口输出编码后的极化码字。

可选地，所述通信接口可以包括输入输出接口。输入输出接口用于输出编码后的极化码字。

第十二方面，本申请提供一种芯片，包括逻辑电路和通信接口，所述通信接口用于接收待处理的数据和/或信息，并将所述待处理的数据和/或信息传输至所述逻辑电路，所述逻辑电路用于执行译码的处理，以及，所述通信接口还用于输出处理后的数据和/或信息。

具体地，所述芯片可以为配置在接收端中的芯片，所述待处理的数据和/或信息可以为待译码序列。所述芯片通过通信接口接收待译码序列，并将其传输至逻辑电路；所述逻辑电路采用第二方面描述的译码方法，对该待译码序列进行译码；以及，所述芯片通过所述通信接口输出译码结果。

可选地，所述通信接口可以包括输入接口和输出接口。输入接口用于接收所述待译码序列，所述输出接口用于输出译码结果。

可选地，所述芯片可以为配置在接收端中的芯片，所述逻辑电路用于获取待译码序列，并采用第二方面描述的译码方法，对该待译码序列进行译码；以及，所述芯片通过所述通信接口输出译码结果。

可选地，所述通信接口可以包括输入输出接口。输入输出接口用于输出译码结果。

第十三方面，本申请提供一种通信系统，包括编码端设备和译码端设备。

附图说明

图 1 示出了适用于本申请实施例的无线通信系统的一示意图。

图 2 示出了采用无线技术进行通信的基本流程示意图。

图 3 示出了 polar 码编码的一示意图。

图 4 示出了 Trellis 表示的一可能示意图。

图 5 示出了用于 Arikan 核矩阵的扩展 Trellis 的一可能示意图。

图 6 示出了 (4096，2048)的 polar subcode 在编码侧采用不同极化核时，译码时的性能和复杂度对比。

图 7 示出了核 K_{16} 矩阵。

图 8 示出了核 K'_{16} 矩阵。

图 9 示出了(1024，512)的 polar subcode 在编码侧采用不同极化核时，译码时的性能和复杂度对比。

图 10 示出了核 K_{32} 矩阵。

图 11 示出了核 K'_{32} 矩阵。

图 12 示出了基于 Viterbi 算法的网格图译码的一可能示意图。

图 13 是根据本申请实施例提供的译码方法的一示意性框图。

图 14 是根据本申请实施例提供的编码方法的一示意性框图。

图 15 至图 18 示出了在核 K_{24} 的基础上构建维度为 20×20 核的示意图。

图 19 示出了构造出的维度为 20×20 核矩阵的示意图。

图 20 至图 22 示出了维度为 20 的核矩阵的示意图。

图 23 和图 24 示出了维度为 21 的核矩阵的示意图。

图 25 至图 27 示出了维度为 24 的核矩阵的示意图。

图 28 和图 29 示出了维度为 25 的核矩阵的示意图。

图 30 示出了维度为 28 的核矩阵的示意图。

图 31 示出了(8,4)码的一般性网格图和分割网格图的示意图。

图 32 至图 34 示出了利用网格图分割的方法对(8，4)汉明码进行译码的过程的示意图。

图 35 至图 37 示出了针对 2 次迭代的 Arikan 矩阵核处理过程的示意图。

图 38 示出了核矩阵 K_1 的一示意图。

图 39 示出了适用于本申请实施例的递归网格图。

图 40 和图 41 示出了基于核 K_{32}，核 K'_{32} 以及 Arikan 核的(1024，512)polar subcode 性能和复杂度对比的示意图。

图 42 和图 43 示出了与 5G LDPC 码进行性能和复杂度上的对比的示意图。

图 44 为本申请提供的装置的示意性框图。

图 45 为本申请提供的装置的一示意性结构图。

图 46 为本申请提供的装置的又一示意性结构图。

具体实施方式

下面将结合附图，对本申请中的技术方案进行描述。

本申请实施例的技术方案可以应用于各种无线通信系统，例如该无线通信系统可以包括但不限于：无线局域网（wireless local access network, WLAN）系统、窄带物联网（narrow band-internet of things, NB-IoT）系统、第五代（5th generation，5G）系统或新无线（new radio，NR）、长期演进（long term evolution，LTE）系统或者 5G 之后的通信系统等。本申请实施例的技术方案还可以应用于设备到设备（device to device，D2D）通信，机器到机器（machine to machine，M2M）通信，机器类型通信（machine type communication，MTC），卫星通信以及车联网系统中的通信。其中，车联网系统中的通信方式统称为 V2X（X 代表任何事物），例如，该 V2X 通信包括：车辆与车辆（vehicle to vehicle，V2V）通信，车辆与路边基础设施（vehicle to infrastructure，V2I）通信、车辆与行人之间的通信（vehicle to pedestrian，V2P）或车辆与网络（vehicle to network，V2N）通信等。

例如，本申请实施例的技术方案可以应用于 5G 移动通信系统的应用场景中，如增强移动带宽（enhance mobile broadband，eMBB）、高可靠性低延迟通信（ultra reliable low latency communication，URLLC）、增强海量机器连接通信（massive machine type communication，eMTC）等。

本申请提及的网络设备，网络设备可以是任意一种具有无线收发功能的设备。该设备包括但不限于：演进型节点 B（evolved Node B，eNB）、无线网络控制器（Radio Network Controller，RNC）、节点 B（Node B，NB）、基站控制器（Base Station Controller，BSC）、基站收发台（Base Transceiver Station，BTS）、家庭基站（例如，Home evolved NodeB，或 Home Node B，HNB）、基带单元（BaseBand Unit，BBU），无线保真（Wireless Fidelity，WIFI）系统中的接入点（Access Point，AP）、无线中继节点、无线回传节点、传输点（transmission point，TP）或者发送接收点（transmission and reception point，TRP）等，还可以为 5G，如，NR，系统中的 gNB，或，传输点（TRP 或 TP），5G 系统中的基站的一个或一组（包括多个天线面板）天线面板，或者，还可以为构成 gNB 或传输点的网络节点，如基带单元（BBU），或，分布式单元（distributed unit，DU）等。

在一些部署中，gNB 可以包括集中式单元（centralized unit，CU）和 DU。gNB 还可

以包括有源天线单元（active antenna unit，简称 AAU）。CU 实现 gNB 的部分功能，DU 实现 gNB 的部分功能。比如，CU 负责处理非实时协议和服务，实现无线资源控制（radio resource control，RRC），分组数据汇聚层协议（packet data convergence protocol，PDCP）层的功能。DU 负责处理物理层协议和实时服务，实现无线链路控制（radio link control，RLC）层、媒体接入控制（media access control，MAC）层和物理（physical，PHY）层的功能。AAU 实现部分物理层处理功能、射频处理及有源天线的相关功能。由于 RRC 层的信息最终会变成 PHY 层的信息，或者，由 PHY 层的信息转变而来，因而，在这种架构下，高层信令，如 RRC 层信令，也可以认为是由 DU 发送的，或者，由 DU+AAU 发送的。可以理解的是，网络设备可以为包括 CU 节点、DU 节点、AAU 节点中一项或多项的设备。此外，可以将 CU 划分为接入网（radio access network，RAN）中的网络设备，也可以将 CU 划分为核心网（core network，CN）中的网络设备，本申请对此不做限定。

本申请提及的终端设备也可以称为用户设备（user equipment，UE）、接入终端、用户单元、用户站、移动站、移动台、远方站、远程终端、移动设备、用户终端、终端、无线通信设备、用户代理或用户装置。本申请的实施例中的终端设备可以是手机（mobile phone）、平板电脑（Pad）、带无线收发功能的电脑、虚拟现实（virtual reality，VR）终端设备、增强现实（augmented reality，AR）终端设备、工业控制（industrial control）中的无线终端、无人驾驶（self driving）中的无线终端、远程医疗（remote medical）中的无线终端、智能电网（smart grid）中的无线终端、运输安全（transportation safety）中的无线终端、智慧城市（smart city）中的无线终端、可穿戴的无线终端、智慧家庭（smart home）中的无线终端等等。本申请的实施例对应用场景不做限定。

为便于理解本申请实施例，首先结合图 1 和图 2 详细说明适用于本申请实施例的通信系统。

图 1 是适用于本申请实施例的无线通信系统的一示意图。如图 1 所示，无线通信系统可以包括至少一个网络设备 110 和至少一个终端设备（例如，图 1 中所示的 111, 112 以及 113）。网络设备 110 和终端设备进行无线通信。当网络设备 110 向终端设备发送信号时，网络设备 110 为编码端，终端设备为译码端。当终端设备向网络设备 110 发送信号时，终端设备为编码端，网络设备为译码端。

在如图 1 所示的无线通信系统中，原始的信息经过编码端（encoder）的编码后，经过信道的传输，被解码端（decoder）接收，经过解码端的译码，恢复出原始的信息。

其中，编码端也可以称为发送端（或者说，发送设备），解码端也可以称为信息或数据的接收端（或者说，接收设备）。该发送设备可以为网络设备，该接收设备可以为终端设备。或者，该发送设备可以为终端设备，该接收设备可以为网络设备。或者，该发送设备可以为终端设备，该接收设备可以为终端设备。或者，该发送设备可以为网络设备，该接收设备可以为网络设备。

应理解，上述图 1 仅是示例性说明，本申请并未限定于此。例如，本申请实施例还可以应用于 5G eMBB 场景上行/下行控制信道编解码的任何通信场景。

图 2 是采用无线技术进行通信的基本流程图。发送端的信源依次经过信源编码、信道编码、速率匹配和调制后在信道上发出。接收端收到信号后依次经过解调、解速率匹配、信道解码和信源解码后获得信宿。本申请实施例提供的技术方案可以用于如图 2 中虚线框

所示的信道编码和信道解码模块中。

信道编解码是无线通信领域的核心技术之一，其性能的改进将直接提升网络覆盖及用户传输速率。目前，极化码（polar codes）已能够被严格证明达到信道容量的信道编码方案，具有高性能，较低复杂度，匹配方式灵活等特点，目前已经被 3GPP 确定成为 5G 控制信道 eMBB 场景（上行/下行）控制信道编码方案。

为便于理解本申请实施例，下面首先对本申请中涉及的几个术语做简单介绍。

1、polar 码的编码方式

参见图 3，图 3 示出了 polar 码编码的一示意图。如图 3 所示，符号“⊕”代表二进制相加，其输入为左侧和下侧，输出为右侧。图 3 中每条实线代表 1 比特。我们将{u_0, u_1, u_2, u_4}设置为固定比特（或者说冻结比特），将{u_3, u_5, u_6, u_7}共 4 位信息比特进行 polar 编码，得到 8 位编码比特。编码之后，再将该 8 位编码比特进行调制后经过噪声信道发送。

待编码比特根据他们各自的可靠度不同排序。一般地，可靠度较高的比特设置为信息比特（data），可靠度较低的比特设置为固定比特（frozen），固定比特（frozen）的值通常设置为 0，在实际传输中发送端和接收端都已知。如图 3 所示，u_7, u_6, u_5, u_3 为可靠度靠前的四位比特，设置为信息比特（data），u_4, u_2, u_1, u_0 为可靠度靠后的四位比特，设置为固定比特（frozen）。

2、线性分组码网格（Trellis）表示

Trellis 译码为卷积码的传统译码方式，对于线性分组码，也可以通过 Trellis 进行译码。Trellis 译码为一种最大似然（maximum likelihood，ML）/MAP 译码方式，通过该方式译码能够带来好的译码性能，但是相应的，其译码复杂度很高。给定一个线性分组码，其校验矩阵为 H，则所有的码字都需要符合：$0 = cH^T = \sum_{i=0}^{n-1} c_i H^{(i)}$。

其中，$H^{(i)}$ 表示校验矩阵 H 的第 i 列。网格图中在第 j 时刻的节点，被标注为：$S^{(j)} = \sum_{i=0}^{j-1} c_i H^{(i)}$。

图 4 示出了 Trellis 表示的一可能示意图。参见图 4，Trellis 表示的基本组成单元为 Trellis 节点（如图 4 中的节点 1, 2, …,20）。例如，校验矩阵 $H = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$，其 Trellis 表示如图 4 所示。关于采用 Trellis 进行译码的详细方案可以参考现有的描述，对此不作限定。

3、信道参数

polar 码是一种可达二进制输入离散无记忆信道容量的可构造性信道编码方式。

令 W：{0,1}→Y 为一个对称二进制（binary）离散无记忆信道（discrete memoryless channel）（B-DMC），其输入字母集合为 x={0,1}，输出字母集为 Y，在给定输入 x，信道输出是 y 的情况下，信道条件转移概率表示为：W(y|x)。

可以定义 B-DM 信道（B-DMC）W 的容量为：

$$I(W) = \sum_{y \in Y} \sum_{x \in \{0,1\}} \frac{1}{2} W(y|x) \log \frac{W(y|x)}{\frac{1}{2} W(y|0) + \frac{1}{2} W(y|1)}。$$

可以定义 B-DM 信道巴氏参数为:

$$Z(W)=\sum_{y\in Y}\sqrt{W(y|0)+\frac{1}{2}W(y|1)}。$$

4、核矩阵

Arikan 在文献里推断极化是一种普遍现象，并不单单受限于其给出的 $G=F_2^{\otimes m}$。其中 $F_2=\begin{pmatrix}1 & 0\\ 1 & 1\end{pmatrix}$，“$\otimes m$”代表 m 次克罗内克（Kronecker）乘积。Korada 等人证明了该结论并推广至高维核矩阵，证明了任意 $l\times l$ 的矩阵能够发生极化的条件是该矩阵的任意列置换矩阵不是上三角矩阵。其次，Korada 等人还证明了当 $l\leqslant 15$ 时，任意核矩阵的极化指数 β 都不能超过 1/2。在相同码长下，更大的极化指数 β 代表相应的极化码的译码错误概率更低，相应的译码性能更好。然而高维核矩阵极化码主要的问题在于其译码算法复杂度较高，这是高维核矩阵研究的瓶颈问题之一。

5、极化指数（polarization exponent）

关于极化指数简单介绍几个定理，具体的可以参考现有技术的描述，此处不作限定。

定理 1：给定一个 B-DM 信道 W（即 B-DWC W），对于 F_2 和任意 $\beta<1/2$，可以得到:

$$\liminf_{m\to\infty}P\left(Z_m\leq 2^{-n^{\beta}}\right)=\liminf_{m\to\infty}P\left(Z_m\leq 2^{-2^{m\beta}}\right)=I(W)。$$

相应地，如果 $I(W)<1$，对于任意 $\beta>1/2$，可以得到:

$$\liminf_{m\to\infty}P\left(Z_m\geq 2^{-n^{\beta}}\right)=\liminf_{m\to\infty}P\left(Z_m\geq 2^{-2^{m\beta}}\right)=1。$$

其中，β 称为极化指数，码长 $n=2^m$。

Korada 等人将上述定理推广至任意的高维核矩阵，此时，码长 $n=l^m$，即:

定理 2：对于任意 B-DM 信道 W（即 B-DWC W）满足 0< I(W)<1，满足以下条件，我们可以称一个 $l\times l$ 维的核矩阵 K 有极化指数 E(K):

（1）对于任意 $\beta<E(K)$:

$$\liminf_{m\to\infty}P(Z_m\leq 2^{-l^{m\beta}})=I(W)。$$

（2）对于任意 $\beta>E(K)$:

$$\liminf_{m\to\infty}P(Z_m\geq 2^{-l^{m\beta}})=1。$$

在这里，用 E(K)来表示极化指数。对于 Arikan 的核，其极化指数为 1/2。

给定一个矩阵 K 和相应的 E(K)，定理 2 表明:

1）当 m 趋于无穷时，如果 $0<R<I(W)$，$\beta<E(K)$，那么存在一个大小为 NR 的集合 A 满足：$\sum_{i\in A}Z^{(i)}\leq 2^{-n^{\beta}}$，A 为信息比特集合，通过逐比特消除（successive cancellation, SC）译码，在 SC 译码下误帧率的界限为:

$$P_e(n)\leq 2^{-n^{\beta}}。$$

2）当 m 趋于无穷时，如果 $R>0$，$\beta>E(K)$，那么任意大小为 NR 的集合 A 满足：$\max_{i\in A}Z^{(i)}>2^{-n^{\beta}}$，A 为信息比特集合，在 SC 译码下误帧率的界限为:

$$P_e(n)\geq 2^{-n^{\beta}}。$$

由上可知，极化指数可以衡量译码性能的收敛速度，更大的极化指数代表着更快的收敛速度，从而使得在码长较短的情况下也能获得更优的译码性能。

6、标度指数（scaling exponent）

极化核的另外一个关键特性是 scaling exponent，可以理解，scaling exponent 可以用于表征核矩阵性能。应理解，本申请实施例关心的是 scaling exponen 的功能，关于 scaling exponent 对应的中文表达，可以为标度指数，也可以为其他表达，对此不作限定。

举个示例，给定信道容量为 I(W)的信道 W，期望错误概率 P_e，假设对应（N，k）的极化码，其核矩阵为 K。如果希望以 R=I(W)-Δ的速率进行传输，那么码长 N 满足:

$$N=O\left(\frac{1}{\left(I(W)-R\right)^{\mu(K)}}\right)=Q\left(\Delta^{-\mu(K)}\right)$$

其中，$\mu(K)$为 scaling exponent。

可以理解，scaling exponent 是一个和信道有关的参数。对于随机码来说，$\mu=2$；对于 polar 码来说，当 $l\to\infty$ 时，令 $K=F_l$，可以得到:

$$\lim_{l\to\infty}\mu(F_l)=2$$

在 BEC 信道下，采用 Arikan 核的 polar 码，其 scaling exponent 可以为 3.627。

7、部分距离（partial distance）。

一个 $l\times l$ 的矩阵 K，$K[i]$表示为该矩阵的第 i 行，其部分距离 $D_i(i=0,\cdots,l-1)$ 为:

$$D_i\triangleq d_H\left(K[i],\left\langle K[i+1],\cdots,K[l-1]\right\rangle\right),i=0,\cdots,l-2$$

$$D_{l-1}\triangleq d_H\left(K[l-1],0\right)。$$

其中，$d_H(a,b)$表示向量 a 和 b 之间的汉明距离，向量 D 可以定义为 partial distance profile (PDP)。对任意 B-DMC W，任意 $l\times l$ 的极化矩阵 K 及其部分距离 $\{D_i\}_{i=0}^{l-1}$，相应的

极化指数E(K)可以表示为:

$$E(K)=\frac{1}{l}\sum_{i=0}^{l-1}\log_l D_i \text{。}$$

事实上，随着$l\to\infty$，我们可以得到一个$l\times l$的核使得极化指数无限接近于1。

应理解，上述定理仅是为理解做的简单说明，并不对本申请实施例的保护范围造成限定，具体地可以参考现有的描述，此处不作限定。

8、核矩阵编码

一个(n,k)的 polar 码(其中码长 n 为: $n=l^m$),给定一个$l\times l$的核矩阵 K, $G=M^{(m)}K^{\otimes m}$，其中，$M^{(m)}$为数字翻转矩阵，相应的映射关系为: $\sum_{i=0}^{m-1}t_i l^i\to\sum_{i=0}^{m-1}t_{m-1-i}l^i$，$t_i\in[l]$，$[l]$代表集合{0,1,…,$l$-1}，相应的编码可以表示为$C_0^{n-1}=u_0^{n-1}G$。其中，$u_i, i\in F$是冻结比特，$|F|=n-k$，剩余比特为信息比特。

9、核处理（Kernel Processing）

polar 码译码时，需要计算后验概率$W_m^{(j)}(u_0^j\mid y_0^l)$。$W_m^{(j)}(u_0^j\mid y_0^l)$可以通过递归得到，为描述简单，令 m=1，相应的任务可以被称为核处理，那么:

$$W_1^{(j)}(u_0^j\mid y_0^l)=\sum_{u_{j+1}^{l-1}}W_1^{(l-1)}(u_0^{l-1}\mid y_0^{l-1})=\sum_{u_{j+1}^{l-1}}\prod_{i=0}^{l-1}W\left((u_0^{l-1}K)_i\mid y_i\right)\text{。}$$

上式也可以近似表示为:

$$\tilde{W}_1^{(j)}(u_0^j\mid y_0^l)=\max_{u_{j+1}^{l-1}}W_1^{(l-1)}(u_0^{l-1}\mid y_0^{l-1})=\max_{u_{j+1}^{l-1}}\prod_{i=0}^{l-1}W\left((u_0^{l-1}K)_i\mid y_i\right)\text{。}$$

上式概率对应了在译码数中u_0^j可能性的最大的路径。对数似然比(log-likelihood ratio，LLR）可以近似地表示为:

$$S_{1j}(u_0^{i-1}\mid y_0^{n-1})=\max_{u_{i+1}^{l-1}}\ln W_1^{(l-1)}(u(0)^i\mid y_0^{l-1})-\max_{u_{i+1}^{l-1}}\ln W_1^{(l-1)}(u(1)^i\mid y_0^{l-1})\text{。}$$

其中，$u(a)^i=\left(u_0^{i-1},a,u_{i+1}^{l-1}\right)$。假设对于所有$u_j$，i<j<$l$都是等概的，核 K 最后$l$-i+1 行构成的矩阵可以生成对应的码字，那么$S_{1j}$可以通过对这些码字的陪集执行 ML 译码得到，陪集首由u_0^{i-1}和核 K 前i行构成的矩阵相乘得到。

应理解，在本申请实施例中，多次提及陪集（coset），本领域技术人员应理解其含义。例如，如果 G 是一个群，H 是 G 的一个子群，g 是 G 的一个元素，那么 gH = {gh: 对于所有 h ∈ H}表示 H 的左陪集，Hg = {hg: 对于所有 h ∈ H}表示 H 的右陪集。下文对此不再赘述。

10、Extended Kernel codes

考虑一个由$K^{(i)}$生成的（l+1，l-i）码$\overline{\mathbb{C}}^{(i)}$，$K^{(i)}$可以由以下规则从$K$得到:

1）$K^{(i)}$ 的前 l 列通过获取 K 的最后 l-i 行得到；

2）$K^{(i)}$ 的最后一列第一行置为 1，其余行全部置为 0。

例如，Arikan 的核为 $\begin{pmatrix}1 & 0\\ 1 & 1\end{pmatrix}$，可以得到 $K^{(0)}=\begin{pmatrix}1 & 0 & 1\\ 1 & 1 & 0\end{pmatrix}$，$K^{(1)}=\begin{pmatrix}1 & 1 & 1\end{pmatrix}$，其相应的 Trellis 如图 5 所示。

11、相关偏差（correlation discrepancy）

在采用 Trellis 进行译码时，需要对不同路径（即代表不同的译码结果）进行选择和对比，对于路径值（c 的分数）的定义可以由多种方式给出，例如：

1）概率的对数。例如，可以表示为：$\log W(c\,|\,y)$

2）相关性(correlation) 。例如，可以表示为：$\sum_i(-1)^{c_i}S(y_i)$。其中，$S(y_i)$为第 i 个 LLR。

3）correlation discrepancy。例如，可以表示为：$\sum_i\tau(S(y_i),c_i)$。

定义 c_0^{l-1} 关于 y_0^{l-1} 的 correlation discrepancy 为：$\varepsilon(c_0^{l-1},y_0^{l-1})=\sum_{i=0}^{l-1}\tau(c_i,y_i)$。其中，

$$\tau(c,y)=\begin{cases}-|S(y)|, & (-1)^c\neq sign\,S(y)\\ 0 & otherwise\end{cases}。$$

且，$S(y)=\log\dfrac{W(0\,|\,y)}{W(1\,|\,y)}$，可以得到：$S_{1,t}(u_0^{t-1}\,|\,y_0^{n-1})=\varepsilon(c^{(0)},y_0^{l-1})-\varepsilon(c^{(1)},y_0^{l-1})$。其中，$c^{(ut)}=(u_0^t,U(u_0^{t-1},u_t))K$，$U(u_0^{t-1},u_t)=u_{t+1}^{l-1}$。对于上式的求解，可以等效于寻找码字 $\overline{\mathbb{C}}^{(i)}$ 的陪集中最后一位为 0 和 1 时最可能的码字，这可以通过在码字 $\overline{\mathbb{C}}^{(i)}$ 的 Trellis 执行 Viterbi 算法实现。假设码字的最后一位被擦除，那么相应的 Trellis 可以被定义为第 i 个扩展 Trellis（extended Trellis）。一般来说，Viterbi 算法的复杂度为 $O(2^{\min(i+1,l-i)})$。

在本申请实施例中，在选择极化核时，同时考虑译码时对核处理的复杂度。有的时候性能更好的核可能会带来更高的译码复杂度，因此核矩阵构造或者选择时还需要综合考虑。

一示例，图 6 示出了 (4096，2048)的 polar 子码（polar subcode）在编码侧采用不同极化核 K_{16} 和 $K_{16}^{'}$ 时，译码侧采用 SCL 译码时的性能和复杂度对比，核 K_{16} 和 $K_{16}^{'}$ 如图 7 和图 8 所示。

可以知道：$E(K_{16})=E(K_{16}^{'})=0.51828$；$\mu(K_{16})=3.346$，$\mu(K_{16}^{'})=3.45$；$\phi(K_{16})=472$，$\phi(K_{16}^{'})=183$。其中，$\phi(K)$ 表示核 K 的处理复杂度。需要说明的是，此处主要以基于递归的核处理算法为例进行示例性说明。

可以看出，采用 K_{16} 相比于 $K_{16}^{'}$ 能带来更好的性能，但是其译码时复杂度更高，所以此时对于核矩阵的选择，还需要考虑性能和复杂度的折衷。

又一示例，图 9 示出了 (1024，512)的 polar subcode 在编码侧采用 $K_{32}^{'}$ 以及极化核 K_{32} 时，译码侧采用 SCL 译码时的性能和复杂度对比，核 K_{32} 和 $K_{32}^{'}$ 如图 10 和图 11 所示。

可以知道： $E\left(K_{32}\right)=0.5219$， $E\left(K_{32}^{'}\right)=0.529$； $\mu\left(K_{32}\right)=3.421$， $\mu\left(K_{32}^{'}\right)=3.207$； $\phi\left(K_{32}\right)=532$， $\phi\left(K_{32}^{'}\right)=6770$。可以看出，采用 $K_{32}^{'}$ 相比于 K_{32} 能带来更好的性能，但是其译码时复杂度更高，采用 $K_{32}^{'}$ 译码时的复杂度可能是采用 K_{32} 译码时的复杂度的十倍以上。

由上可知，在编码时，核矩阵的选择可以影响译码复杂度。

此外，前面提到线性分组码可以用基于 Trellis 的方法进行译码，polar 码也是一种线性分组码，也可以采用这样的方法进行译码。具体译码步骤例如可以如下所示。

1）令 l(j,s)为在时刻 j-1 时连接到时刻 j 的状态 s 的所有状态集合；

2）令 $c_{j-1,s',s}$ 为相应的连接边上的值；

3）定义节点值为 $\varepsilon_{j,s}=\max\limits_{s'\in l(j,s)}\left(\varepsilon_{j-1,s'}-\tau\left(S\left(y_{i-1}\right),c_{j-1,s',s}\right)\right)$，$\varepsilon_{0,0}=0$；

4）LLR 可以根据 $S_1^{(t)}\left(u_0^{t-1},y_0^{t-1}\right)=\varepsilon_{l,0}-\varepsilon_{l,1}$ 计算得出。

一示例，针对 Arikan 核采用 Viterbi 算法时，已知 Arikan 的核为 $\begin{pmatrix}1&0\\1&1\end{pmatrix}$，可以得到 $K^{(0)}=\begin{pmatrix}1&0&1\\1&1&0\end{pmatrix}$，其相应的网格图如图 12 所示。通过计算可以得到：

$$\begin{aligned}\varepsilon_{2,0}-\varepsilon_{2,1}&=\max\left(\tau(S_0,0)+\tau(S_1,0),\tau(S_0,1)+\tau(S_1,1)\right)-\\&\quad\max\left(\tau(S_0,0)+\tau(S_1,1),\tau(S_0,1)+\tau(S_1,0)\right)\\&=\max\left(\tau(S_0,0)-\tau(S_0,1)+\tau(S_1,0)-\tau(S_1,1),0\right)-\\&\quad\max\left(\tau(S_0,0)-\tau(S_0,1),\tau(S_1,0)-\tau(S_1,1)\right)\\&=\max\left(S_0+S_1,0\right)-\max\left(S_0,S_1\right)=\operatorname{sgn}\left(S_0\right)\operatorname{sgn}\left(S_1\right)\min\left(\left|S_0\right|,\left|S_1\right|\right)\end{aligned}$$

。

上述方法是一种 ML/MAP 类算法，在码长较长的时候，译码复杂度很高，很难被接受的。

有鉴于此，本申请实施例提出，可以从编码端和/或译码端进行改进，以便可以降低译码复杂度。例如，通过根据实际情况综合考虑，从构造出的多个核矩阵中选择需要的核矩阵，从而可以兼顾性能和译码复杂度。又如，采用网格码译码，并对网格图进行分割，通过将各个分割后的结果进行处理，进而得到最终的网格优化路径，可以降低译码复杂度。

下面将结合附图详细说明本申请提供的各个实施例。

图 13 是根据本申请实施例提供的编码方法 1300 的一示意性框图。

1310，获取信息比特。

也就是说，编码端设备获取待编码的信息比特。该信息比特可以是内部产生；或者，也可以是从外部接收，对此不作限定。

1320，确定目标核矩阵，该目标核矩阵是通过调整原始核矩阵获得的。

在本申请实施例中，调整原始核矩阵，至少可以包括以下两种方案。

方案 1，通过调节原始核矩阵的 PD，得到的一个或多个核矩阵。基于方案 1 得到一个或多个核矩阵，可以从该一个或多个核矩阵中选择合适的目标核矩阵。

方案 2，通过调整原始核矩阵的维度，构造出多个与原始核矩阵维度不同的核矩阵。基于方案 2 得到一个或多个核矩阵，可以从该一个或多个核矩阵中选择合适的目标核矩阵。

应理解，在方案 2 中，调整原始核矩阵的维度，表示的是通过调制原始核矩阵改变核矩阵的维度。例如，对原始核矩阵的行和/或列进行相应地处理，以使得构造出的核矩阵的行小于原始核矩阵的行，构造出的核矩阵的列小于原始核矩阵的列。

关于上述两种方案，下文详细介绍。

1330，基于目标核矩阵，对信息比特进行极化编码。

在本申请实施例中，采用极化编码时，可以先构造出多个核矩阵，基于构造出的核矩阵，根据实际通信情况，选择合适的核矩阵进行编码。通过该方式，在核矩阵构造或选择时，就综合考虑性能和译码复杂度，能够实现通过选择合适的编码时所使用的核矩阵，降低译码复杂度。

例如，在实际通信中，通过调整原始核矩阵，构造出一个或多个核矩阵，并且从该一个或多个核矩阵中选择出合适的目标核矩阵。然后将数据矢量和相应构造出来的目标核矩阵的克罗内克乘积来实现编码。然后经过速率匹配，调制等模块后经过信道传输。

关于编码侧的具体方案，下文详细介绍。

图 14 是根据本申请实施例提供的译码方法 1400 的一示意性框图。

1410，获取待译码序列；

1420，基于多个网格，对待译码序列进行译码，基于多个网格中的第二网格译码时，复用多个网格中的第一网格译码的中间结果，其中，第一网格用于对待译码序列译码的第 t 阶段，第二网格用于对待译码序列译码的第 t+i 阶段，t 为大于 0 或等于 0 的整数，i 为大于 1 或等于 1 的整数。

示例地，第一网格对应译码第 t 阶段，第二网格对应译码第 t+i 阶段。也就是说，在译码第 t 阶段，基于第一网格进行译码处理，在译码第 t+i 阶段，基于第二子网格进行译码处理。例如，在译码第 t 阶段，基于第一网格进行译码，在译码第 t+1 阶段，基于第二网格进行译码。

此外，在各个译码阶段处理得到的结果可以记为中间结果。例如，在译码第 t 阶段，基于第一网格译码时，得到的结果可以记为第一网格译码的中间结果。中间结果，例如可以包括在译码第 t 阶段译码得到的译码路径和对应的路径度量值。下文实施例中，为简洁，用复合分支表（composite branch table，CBT）表示。也就是说，CBT 存储的内容可以包括译码时对应的路径和路径值。在译码的每个阶段，都可以对应一个 CBT。

在本申请实施例中，采用网格 Trellis 的译码算法对待译码序列进行译码时，不同译码阶段得到的中间结果可复用。也就是说，在不同的网格上进行计算时，有些计算步骤可能相同，所以可以多次复用，这使得译码复杂度得到了很大程度的降低，且提高了译码速度。

应理解，本申请实施例主要以第一网格和第二网格为例进行示例性说明，对于多个子网格中的每个网格，或者说在译码的每个阶段，都可以判断该译码阶段是否可以复用其他译码阶段译码的中间结果。

可选地，多个网格中的每个网格被分割为多个子网格。

在本申请实施例中，采用网格 Trellis 的译码算法对待译码序列进行译码时，可以先对网格图进行分割，把在完整的 Trellis 上进行译码的过程分割成对子网格 (sub trellis)上进行译码的过程，从而可以降低译码复杂度。

示例地，本申请实施例提供的译码方法也可以记为递归网格译码或者递归网格分割的译码。

例如，在实际通信中，经过解速率匹配，解调后，通过本申请实施例提出的基于递归网格分割的算法对其进行译码，译码复杂度低。

关于译码侧（即递归网格分割的算法）的具体方案，下文详细介绍。

下面从编码侧和译码侧分别介绍本申请实施例的方案。

编码侧

可选地，在本申请实施例中，可以通过方案 1 和/或方案 2，构造出多个核矩阵。在核矩阵构造或选择时，可以根据实际情况，选择合适的核矩阵用于编码。

方案 1，通过调节原始核矩阵的 PD，构造出多个核矩阵；

方案 2，通过调整原始核矩阵的维度，构造出多个与原始核矩阵维度不同的核矩阵。

下面分别介绍这两种方案。

方案 1，通过调节原始核矩阵的 PD，构造出多个核矩阵。

可选地，基于已有的 PDP，通过调节该 PDP，进而构造出多个核矩阵。

可选地，调节给定 PDP，根据调节后的 PDP 进行深度优先搜索。

应理解，对于其他与深度优先搜索类似的算法，或者说可以实现同样功能的算法，都适用于本申请实施例。下文，为便于理解，主要以深度优先算法为例进行示例性说明。

一种可能的实现方式，在进行深度优先搜索时，可以通过设计一些约束条件，来降低搜索空间和搜索次数。约束条件，例如可以从以下一个或多个方面考虑。

方面 1，设计关于候选行集合 M 的约束条件。也就是说，可以限定候选行集合满足一定的条件。

一可能条件，候选行集合 M 需要满足：$M[i]=\left\{v_0^{l-1}\in F_2^l \mid wt\left(v_0^{l-1}\right)=D_i\right\}$。

方面 2，设计关于所选行的约束条件。也就是说，可以限定选择出来的行满足一定的条件。

一可能条件，从候选行集合 M 中选择倒数第 i 行时，选择出来的行 v 需要满足其汉明重量大于或等于 D_i，即 $wt(v)\geq D_i$。

又一可能条件，在从 M 中选择出某行 v，如果 $c\in C_{\overline{K}}^{(\varphi+1)}\oplus\overline{K}[\varphi]$，$wt(c)<D_\varphi$，那么此

时无需再继续往下运算，可以直接抛弃当前 v，从 M 中选择新的行。其中，⊕表示逻辑运算，或者说异或逻辑运算。

又一可能条件，不能选择同一陪集的行，或者说属于同一陪集的行不会再次被选中。

应理解，上述针对候选行集合 M 和所选行列举的约束条件仅是示例性说明，对此不作限定。例如，针对候选行集合 M 或所选行的其他约束条件，如属于上述约束条件的变形形式，也适用于本申请实施例。

为理解，下面介绍适用于方案 1 的算法流程。一可能的算法流程可以包括如下步骤。

步骤 1，调节 PDP，得到一个或多个新 PDP。

在该步骤 1 中，对于给定的一个 PDP，可以对其进行手动调节。示例地，在调节时，可以使得这个 PDP 尽可能接近 Arikan 矩阵的 PDP，例如：

例如，一个 16×16 的核 K_{16}，其相应的 PDP 为[1,2,2,2,2,4,4,4,4,6,6,8,8,8,8,16]，$E(K_{16})=0.51828$，$\mu(K_{16})=3.346$，$\phi(K_{16})=472$。

如前所述，在本申请实施例中，用 E(*K*)表示极化指数，极化指数可以衡量译码性能的收敛速度，更大的极化指数代表着更快的收敛速度，从而使得在码长较短的情况下也能获得更优的译码性能。用 $\mu(K)$ 表示标度指数 scaling exponent，scaling exponent 可以用于表征核矩阵性能。用 $\phi(K)$ 表示复杂度，即核 K 的处理复杂度。对此，下文不再介绍。

对其 PDP 进行手动调节，可以得到核 K'_{16} 的 PDP：[1,2,2,4,2,2,4,4,6,6,8,8,4,8,8,16]，$E(K'_{16})=0.51828$，$\mu(K'_{16})=3.45$，$\phi(K'_{16})=183$。

又如，一个 32×32 的核 K'_{32}，其相应的 PDP 为[1,2,2,2,2,4,4,4,6,2,4,4,6,6,8,4,8,4,8,8,8,12,12,12,12,12,16,16,16,16,16,32]，$E(K'_{32})=0.529$，$\mu(K'_{32})=3.207$，$\phi(K'_{32})=6770$。

对其 PDP 进行手动调节，可以得到核 K_{32} 的 PDP：[1,2,2,4,2,4,2,4,6,8,2,4,6,8,4,6,8,12,4,8,12,16,4,8,12,16,8,16,8,16,16,32]，$E(K_{32})=0.5219$，$\mu(K_{32})=3.421$，$\phi(K_{32})=532$。

应理解，上述列举的构造出的新的 PDP 仅是示例性说明，还可以构造出更多数量的 PDP。

步骤 2，针对每个新的 PDP，执行深度优先搜索的核构造方法。

通过对给定的 PDP 进行手动调节，能够得到一个或多个的新 PDP，在这些新的 PDP 的基础上，针对每一个新的 PDP，可以执行以下深度优先搜索的核矩阵构造方法。

（1）选择一个给定的 PDP：D 以及一个行向量候选集合 M（或者说候选行集合 M）。

（2）从倒数第一行 $\overline{K}[l-1]$ 开始构造核矩阵，这一行需要满足其行重和 D 最后一个值

相等，即：$D_{l-1}=d_H\left(\overline{K}[l-1],0\right)$。

（3）接着构造倒数第二行$\overline{K}[l-2]$，这一行需要满足：$D_{l-2}=d_H\left(\overline{K}[l-2],\left\langle\overline{K}[l-1]\right\rangle\right)$。

（4）接着构造倒数第三行$\overline{K}[l-3]$，这一行需要满足：

$D_{l-3}=d_H\left(\overline{K}[l-3],\left\langle\overline{K}[l-2],\overline{K}[l-1]\right\rangle\right)$。

5）继续以上步骤，类似地，倒数第 i 行需要满足：

$D_{l-i}=d_H\left(\overline{K}[l-i],\left\langle\overline{K}[l-i+1],\ldots,\overline{K}[l-1]\right\rangle\right)$。

在构造核矩阵的过程中，如果在候选行集合 M 中找不到满足以上条件的行，那么可以回退到上一步，重新找一个新的行$\overline{K}[l-i+1]$，然后继续接下来的流程。

在构造核矩阵的过程中，在满足以下任一条件时，可以停止该算法或者说停止核构造：找到合适的核矩阵时；或者，搜索了所有候选集合的行之后仍然无法找到满足条件的行使得算法继续进行下去。

通过上述步骤，可以得到多个构造出来的核矩阵，接下来可以通过对比这些核矩阵的一些参数，如极化指数（如 E(K)）、scaling exponent（如$\mu(K)$）以及通过对基于这些核矩阵编码后的码字进行译码的译码复杂度（如$\phi(K)$）等，根据实际需求，选择出最好最合适的核矩阵。此外，该方法算法简单，计算也简单易实现。

进一步地，还可以增加以下一项或多项约束条件，如上文关于方面 1 和方面 2 的约束条件，以便降低尝试次数和时长。

条件 1，任何核 K 都可以通过加法将第 i 行加到第 j 行(j<i)，从而核 K 转化为$D_i=wt\left(K[i]\right)$的核$\overline{K}$。此外，还可以把候选行集合 M 设置为：

$M[i]=\left\{v_0^{l-1}\in F_2^l \mid wt\left(v_0^{l-1}\right)=D_i\right\}$。

条件 2，从候选行集合 M 中选择倒数第 i 行时，选择出来的行 v 需要满足其汉明重量大于或等于D_i，即$wt(v)\geq D_i$。

条件 3，在从候选行集合 M 中选择出某行 v，将其跟已经构建好的行进行 partial distance 计算时，如果$c\in C_{\overline{K}}^{(\varphi+1)}\oplus\overline{K}[\varphi]$，$wt(c)<D_\varphi$，那么此时无需再继续往下运算，直接抛弃当前 v，从候选行集合 M 中选择新的行。

例如，假设构建一个 16×16 的核$\overline{K}$（即当前构建的核$\overline{K}$为一个 16×16 的核），目前已经构建好的行：

$\overline{K}[11]$=[1111111100000000]

$\overline{K}[12]$=[1111000011110000]

$\overline{K}[13]$=[1100110011001100]

$\overline{K}[14]$=[1010101010101010]

$\overline{K}[15]$=[1111111111111111]

假设希望找到行$\overline{K}[10]$满足$d_H\left(\overline{K}[10],\left\langle\overline{K}[11],\overline{K}[12],\ \overline{K}[13],\ \overline{K}[14],\ \overline{K}[15]\right\rangle\right)$=6。那么，首先从候选行集合 M 中选择 v=[1111110000000000]。可以发现，c=v+ $\overline{K}[11]$ =[0000001100000000]，$wt(c)=2<6$，此时，没有必要继续后续的$d_H\left(v,\left\langle\overline{K}[11],\overline{K}[12],\ \overline{K}[13],\ \overline{K}[14],\ \overline{K}[15]\right\rangle\right)$计算，当前的 v 直接抛弃。

条件 4，属于同一陪集的行不会再次被选中。

在从候选行集合 M 中选择行进行核 $\overline{K}$ 的构建时，如果某行 v 一旦被选中，计算陪集$C_{\overline{K}}^{(\varphi)}\oplus v$，即$\left\{c\oplus v\,|\,c\in C_{\overline{K}}^{(\varphi)}\right\}$。假设已经计算了$d_H\left(v,\left\langle C_{\overline{K}}^{(\varphi)}\right\rangle\right)$，接下来需要计算$d_H\left(v',\left\langle C_{\overline{K}}^{(\varphi)}\right\rangle\right)$。如果$v'\in C_{\overline{K}}^{(\varphi)}\oplus v$，那么$d_H\left(v',\left\langle C_{\overline{K}}^{(\varphi)}\right\rangle\right)=d_H\left(v,\left\langle C_{\overline{K}}^{(\varphi)}\right\rangle\right)$。即属于同一陪集的行不会再次被选中。

应理解，上述理解的条件 1 至条件 4 仅是示例性说明，对此不作限定。例如，针对候选行集合 M，或者针对选择的行，还可以设计其他约束条件，以便可以降低尝试次数和时长。

此外，可选地，一种可能的实现方式，可以在调节 PDP 时，可以按照使得这个 PDP 接近 Arikan 矩阵的 PDP 的方向调节。应理解，按照接近 Arikan 矩阵的 PDP 的方向调节 PDP 仅是一种可能的调节方式，对此不作限定。例如，对于构造出来的最好的核矩阵的 PDP 不一定是满足最接近 arikan 核矩阵的 PDP。又如，在选择最好最合适的核矩阵时，不一定是选择 PDP 最接近 arikan 核矩阵的 PDP 的核矩阵。

上文详细介绍了方案 1，通过方案 1 可以构造出多个核矩阵。下面介绍方案 2。

方案 2，通过调整原始核矩阵的维度，构造出多个与原始核矩阵维度不同的核矩阵。

可选地，在已有的相对较大维度的核矩阵的基础上，构造出多个相对小维度的核矩阵。基于小维度的核矩阵进行编码，能在保证相同性能的情况下，提供更低的译码复杂度。

一种可能的实现方式，可以通过缩短（shortening）方法，在已有的相对较大维度的核矩阵的基础上，构造出多个相对小维度的核矩阵。应理解，对于其他可以在已有维度的核矩阵的基础上构造出小维度的核矩阵的算法，或者说可以实现同样功能的算法，都适用于本申请实施例。下文，为便于理解，主要以 shortening 算法为例进行示例性说明。

为理解，下面介绍适用于方案 2 的算法流程。一可能的算法流程可以包括如下步骤。

步骤 1，考虑一个 $l\times l$ 的核，即确定原始核矩阵。

步骤 2，选择列 $j\in[l]$。

步骤 3，将第 i 行添加到所有第 j 列中带有 1 的行，i 为这一列中最后一个 1 对应的行。

步骤 4，删除第 i 行和第 j 列，得到$(l\text{-}1)\times(l\text{-}1)$的核 K'。可以知道，$\phi(K')<\phi(K)$，即小维度的核对应的译码复杂度更低。

通过上述方式构建的核矩阵，其译码复杂度会更低，通过重复上述方式 t 次，可以得到$(l\text{-}t)\times(l\text{-}t)$的核。

为便于理解，下面以核 K_{24} 为例，结合图 15 至图 19，列举一具体示例。

一可能的方案，原始核矩阵为图 15 所示的核 K_{24}。对于维度为 24×24 的核 K_{24}，其极化指数为 $E(K_{24})=0.502911$，$\mu(K_{24})=3.61903$，$\phi(K_{24})=374$。假设要在该核 K_{24} 的基础上构建一个新的维度为 20×20 核，具体过程可以如下。在本申请实施例中，假设行序号和列序列均是从 0 开始。

（1）选择第 15 列，可以看出，此列为 1 的行只有最后一行（即第 23 行）。可以删除第 15 列和第 23 行，得到 K_{23}，如图 15 所示。如图 15，将第 15 列和第 23 行用虚框框出。

（2）选择第 14 列，这一列 1 所在位置对应的行为第 13 行和第 14 行。可以将第 14 行添加到第 13 行，第 13 行的结果改变，删除第 14 列和第 14 行，得到 K_{22}，如图 16 所示。在图 16 中，将第 14 列和第 14 行用虚框框出。

（3）选择第 13 列，这一列 1 所在位置对应的行为第 16 行和第 18 行。可以将第 18 行添加到第 16 行，第 16 行的结果改变，删除第 13 列和第 18 行，得到 K_{21}，如图 17 所示。在图 17 中，将第 13 列和第 18 行用虚框框出。

（4）选择第 12 列，这一列 1 所在位置对应的行为第 5 行。可以删除第 12 列和第 5 行，得到 K_{20}，如图 18 所示。在图 18 中，将第 12 列和第 5 行用虚框框出。

通过上述处理，可以得到如图 19 所示的核矩阵 K_{20}。

或者，在该核 K_{24} 的基础上构建的核矩阵，可以包括以下一项或多项：如图 16 所示的核 K_{23}，如图 17 所示的核 K_{22}，如图 18 所示的核 K_{21}，如图 19 所示的核 K_{20}。在实际通信过程中，可以根据实际情况，综合考虑，如综合考虑性能和复杂度，从上述多个核矩阵中选择合适的核矩阵。

应理解，上述仅是为便于理解做的示例性说明，对于中间的处理过程，也可能有其他变形，对此不作限定。

通过以上方法够到得到的核矩阵，在采用相同译码算法的情况下，其译码复杂度会更低，如 $\phi(K_{20})<\phi(K_{24})$。基于此，可以根据实际情况对核矩阵的选择进行综合考虑，如综合考虑性能和复杂度，选择合适的核矩阵。

上文结合方案 1 和方案 2，介绍了本申请实施例提供的核矩阵构造方法。应理解，方案 1 和方案 2 可以单独使用，也可以结合使用，对此不作限定。

作为示例，下面列举通过本申请实施例构造出的几个核矩阵的形式。

一示例，维度为 20 的核矩阵，可以如图 20 至图 22 所示。

又一示例，维度为 21 的核矩阵，可以如图 23 和图 24 所示。

又一示例，维度为 24 的核矩阵，可以如图 25 至图 27 所示。

又一示例，维度为 25 的核矩阵，可以如图 28 和图 29 所示。

又一示例，维度为 28 的核矩阵，可以如图 30 所示。

又一示例，维度为 32 的核矩阵，可以如图 10 所示。

应理解，上述图 20 至图 30 列举的核矩阵仅是示例性说明，对此不作限定。

基于本申请实施例提供的编码方法，采用极化编码时，可以先构造出多个核矩阵，如通过上述方案 1 和/或方案 2 构造多个核矩阵。基于构造出的核矩阵，根据实际通信情况，选择合适的核矩阵进行编码。通过该方式，在核矩阵构造或选择时，就综合考虑性能和译码复杂度，能够实现通过选择合适的编码时所使用的核矩阵，降低译码复杂度。

译码侧

如前所述，采用网格码译码时，可以先对网格图进行分割，把在完整的 Trellis 上进行译码的过程分割成对子网格图(sub trellis)上进行译码的过程。然后可以通过对每一个小的网格图译码的中间结果进行组合，最终选出最优路径。

例如，更小一级的网格图中所有可能路径拼接后得到多个组合，通过选择这些可能组合中路径分数最大的那个得出更大一级的网格图最优路径及路径值。

为便于理解，下面示出了可能的算法流程。一可能的算法流程可以包括如下几个方面的内容。

方面 1，针对线性分组码的网格分割。

在本申请实施例中，针对 polar 码译码时每个阶段 t 的网格图进行多级分割。

假设给定一个(8, 4)的线性码 C，其生成矩阵为 $G=\begin{pmatrix}1&1&1&1&1&1&1&1\\1&1&1&1&0&0&0&0\\1&1&0&0&1&1&0&0\\1&0&1&0&1&0&1&0\end{pmatrix}$。令 $C_{h,h'}$ 为其子码，且非 0 位只会出现在 h≤i<h’的位置；令 $p_{h,h'}(C)$ 为通过对线性码 C，在除开 h≤i<h’的位置进行打孔（puncturing）操作得到的线性码；令 $s_{h,h'}(C)=p_{h,h'}(C_{h,h'})$ 为通过对线性码 C，在除开 h≤i<h’的位置进行缩短（shortening）操作得到的码。例如，当 0≤i<4，即 h=0，h’=4 时，可以得到:

$$p_{0,4}(C):G_{0,4}^{(p)}=\begin{pmatrix}1&1&1&1\\1&1&0&0\\1&0&1&0\end{pmatrix}$$

$$s_{0,4}(C):G_{0,4}^{(s)}=\begin{pmatrix}1&1&1&1\end{pmatrix}$$

在网格图上，从时刻 h 到时刻 h'的路径对应于陪集 $p_{h,h'}(C)/s_{h,h'}(C)$。

如图 31 所示，左图给出了该(8,4)码的一般性网格图示意，右图给出了该(8,4)码的分割网格图，通过对陪集的计算 $p_{0,4}(C)/s_{0,4}(C)$，可以得到 4 个陪集，分别是:

①{(0000),(1111)}

②{(0011),(1100)}

③{(1010),(0101)}

④{(0110),(1001)}

根据图 31 中左图左半部分和右图左半部分的对应关系，可以看出，它们又各自代表了从 0 时刻状态到 4 时刻状态的路径。

方面 2，线性分组码的网格译码。

对于每个陪集 $D\in p_{h,h'}(C)/s_{h,h'}(C)$，需要找到最可能的那个元素 l(D)和它对应的分数，对此，可以定义一个复合分支表（composite branch table，CBT）用于存储这两个值。需要说明的是，关于元素对应的分数，例如可以通过前面术语解释中的 correlation discrepancy 表示，例如可以记为 E(D)。作为示例，$CBT_{(x,y)}$=(l(D),E(D))。需要说明的是，对(n,k)码进行 ML 译码，即 $p_{0,n}(C)/s_{0,n}(C)$ 只包含一个元素，所以 $CBT_{(0,n)}$存储的是采用 ML 译码时对应的路径和路径值。

应理解，关于 CBT 仅是为便于描述做的命名，其命名不对本申请实施例的保护范围造成限定。此外，本申请实施例并不限定 CBT 中只能包括两个元素，在实际通信中，CBT 中可能还包括其他元素，对此不作限定。

还应理解，此处用 correlation discrepancy 表示元素（即路径 l(D)）对应的分数，其他可以用于表征该元素对应的分数（或者说路径值）的方案，也适用于本申请实施例。

方面 3，CBT 的构建。

假设构建 $CBT_{x,y}$，y-x⩾2。那么，对于某个 z（x<z<y），可以构建 $CBT_{x,z}$和 $CBT_{z,y}$。对 $D_y\in p_{x,y}(C)/s_{x,y}(C)$，可以枚举所有 $D'\in p_{x,z}(C)/s_{x,z}(C)$和 $D''\in p_{z,y}(C)/s_{z,y}(C)$的陪集。可以得到：$E(D)=\max_{\substack{D',D''\\ D'.D''=D_y}}\left(E(D')+E(D'')\right)$。

可以看出，路径[x,y]截断成了两部分。此时，对于 $CBT_{x,y}$的构建，可以不必枚举所有 $D_y\in p_{x,y}(C)/s_{x,y}(C)$，可以通过上述截断处理的方式得到，这能够带来复杂度的降低。

为便于理解，作为示例，通过图 32 至图 34 示例地介绍利用网格图分割的方法对（8，4）汉明码进行译码的过程。假设对于一个（8，4）的汉明码，经过信道传输后，在接收端的接收信号为 y_0^7=(-1,2,1,3,-1,2,3,5)。对该长度为 8 的码对应的网格图进行分割，如图 32

所示，可以分割为 4 个长度为 2 的子码，该 4 个长度为 2 的子码分别对应一子网格图。

图 32 示出了[0,1]这一段相应的路径及路径值计算过程。如图 32 所示，对于[0,1]，可以通过如下计算，获得[0,1]这一段相应的路径及路径值:

对于 ε_{20}， $\varepsilon_{20}=\varepsilon_{10}+\tau(S_2,0)=-1$;

对于 ε_{21}， $\varepsilon_{21}=\varepsilon_{10}+\tau(S_2,1)=-3$;

对于 ε_{22}， $\varepsilon_{22}=\varepsilon_{11}+\tau(S_2,0)=0$;

对于 ε_{23}， $\varepsilon_{23}=\varepsilon_{11}+\tau(S_2,1)=-2$。

类似地，[2,3]，[4,5]，[6,7]和其相应的子网格图都可以表示出来，并用类似的方法计算出各种可能路径即路径值。

接下来，[0,3]这一段可以通过[0,1]和[2,3]相应路径组合起来得到。例如，可以选出其中路径分数最大的路径，作为[0,3]这一段的路径，如图 33 中虚线框起来的部分。类似地，右边[4,7]这一段也可以采用相同的操作得到。最终[0,7]整个部分可以通过[0,3]和[4,7]组合得到，如可以通过[0,3]和[4,7]各种可能组合中路径分数对应最大的那条路径得到，如图 34 所示。

在本申请实施例中，可以根据实际情况，选择合适的分割点的位置。分割点的位置可能会影响译码复杂度。以一个 24×24 的核为例，对于 z 的选择（z: x<z<y）:

一示例，均匀分割。即 z=(x+y)/2，在该情况下，可能需要: 715 次加法（summations）、以及 449 次比较（comparison）。

又一示例，优化算法分割。例如: [0,24）分成[0,16）和[16,24），并进一步均匀分割。在该情况下，可能需要: 250 次加法以及 124 次比较。

因此，在实际通信中，可以根据复杂度选择分割点的位置，如可以选择某些分割点，以便降低译码复杂度。

方面 4，嵌套码及嵌套生成矩阵。

根据前面的定义，可以知道 $s_{x,y}(C)\in p_{x,y}(C)$，令 $p_{x,y}(C)$ 的生成矩阵表示为:

$$G_{x,y}^{(p)}=\left(\begin{array}{cc} G_{x,z}^{(s)} & 0 \\ 0 & G_{z,y}^{(s)} \\ \hline G_{x,y}^{(00)} & G_{x,y}^{(01)} \\ G_{x,y}^{(10)} & G_{x,y}^{(11)} \end{array}\right)$$

令 $s_{x,y}(C)$ 的生成矩阵表示为:

$$G_{x,y}^{(s)}=\begin{pmatrix} G_{x,z}^{(s)} & 0 \\ 0 & G_{z,y}^{(s)} \\ G_{x,y}^{(00)} & G_{x,y}^{(01)} \end{pmatrix}$$

方面 5，递归网格译码。

陪集 $D\in p_{x,y}(C)/s_{x,y}(C)$ 和向量 $vG_{x,y}^{'}$ 存在一一对应关系。其中 $G_{x,y}^{'}=\begin{pmatrix} G_{x,y}^{(10)} & G_{x,y}^{(11)} \end{pmatrix}$ 是一

个 $k'_{x,y}\times(y-x)$ 的矩阵。令 $CBT_{x,y}[v].l := l(D)$， $CBT_{x,y}[v].e := E(D)$，那么：

$$CBT_{x,y}[v].e = \max_{w\in F_2^{k''_{x-y}}} \left(CBT_{x,z}[a].e + CBT_{z,y}[b].e\right),\ v\in F_2^{k''_{x,y}}。$$

其中，a,b 分别代表陪集 $D'\in p_{x,z}(C)/s_{x,z}(C)$ 和陪集 $D''\in p_{z,y}(C)/s_{z,y}(C)$ 中对应元素的序号(如可以用二进制表示)。令 w,v 代表[x,y]这一段可能路径对应的序号，那么：

$$(w,v)\begin{pmatrix}G_{x,y}^{(00)}\\ G_{x,y}^{(10)}\end{pmatrix}\in D',\ (w,v)\begin{pmatrix}G_{x,y}^{(01)}\\ G_{x,y}^{(11)}\end{pmatrix}\in D''。$$

那么，a,b 可以通过以下映射关系得到：

$$(a',a)\begin{pmatrix}G_{x,z}^{(s)}\\ G_{x,z}^{'}\end{pmatrix}=(w,v)\begin{pmatrix}G_{x,y}^{(00)}\\ G_{x,y}^{(10)}\end{pmatrix}\Rightarrow a=(w\quad v)\hat{G}_{x,y}$$

$$(b',b)\begin{pmatrix}G_{z,y}^{(s)}\\ G_{z,y}^{'}\end{pmatrix}=(w,v)\begin{pmatrix}G_{x,y}^{(01)}\\ G_{x,y}^{(11)}\end{pmatrix}\Rightarrow b=(w\quad v)\tilde{G}_{x,y}$$

其中，a', b'为不重要的参数。根据上式，可以得到：

$$CBT_{x,y}[v].e = \max_{w\in F_2^{k''_{x-y}}} \left(CBT_{x,z}\left[(w\quad v)\hat{G}_{x,y}\right].e + CBT_{z,y}\left[(w\quad v)\tilde{G}_{x,y}\right].e\right),\ v\in F_2^{k''_{x,y}}。$$

上述计算的复杂度大概为 $O\left(2^{k'_{x,y}+k''_{x,y}}\right)$。这种方法的复杂度和分割策略有关，即如何在 x,y 中选择分割点 z，可能会影响译码复杂度。

方面 6，基于递归网格的核处理方法。

作为示例，令 $u_0^{t-1}=0$。由前面可以知道，$\varepsilon\left(c^{(0)},y_0^{l-1}\right)$，$\varepsilon\left(c^{(1)},y_0^{l-1}\right)$ 为 $CBT_{0,l}\left(\overline{C}^{(t)}\right)$ 的元素（entries），且可以通过下面的式子递归得到：

$$\begin{aligned}CBT_{x,y}[v].e &= \max_{w\in F_2^{k''_{x-y}}} \left(CBT_{x,z}[a].e + CBT_{z,y}[b].e\right)\\ &= \max_{w_0}\max_{w_1}\cdots\max_{w_{k''_{x-y}}}\left(CBT_{x,z}[a].e + CBT_{z,y}[b].e\right)\end{aligned}$$

其中，$a=(w\quad v)\hat{G}_{x,y}$，$b=(w\quad v)\tilde{G}_{x,y}$。CBT 的元素的序号用 v，a，b 来表示。$k'=\dim p_{x,y}(C)$，$k''=\dim s_{x,y}(C)$。

可选地，在译码过程中可以通过一些处理，进一步简化译码复杂度。

方式 1，复用译码的中间结果。

一种可能的设计，根据各个网格的关联度，确定基于网格译码得到的中间结果是否可被其他网格复用。

一示例，各个网格的关联度较大时，如当各个网格相似或相同时，或者在满足一定条件时，基于网格译码得到的译码结果可被其他网格译码时直接使用。例如，在 polar 码译

码阶段 t 时的各个部分的组合 CBT 和阶段 t+1 时的 CBT 相同，那么在阶段 t+1 时，无需再次计算。

可选地，可以根据不同译码阶段对线性码 C 进行缩短操作得到的码和/或不同译码阶段对线性码 C 进行打孔操作得到的码，确定各译码阶段的结果是否可复用。或者说，可以根据不同译码阶段对线性码 C 进行缩短操作得到的码和/或不同译码阶段对线性码 C 进行打孔操作得到的码，确定各个网格的关联度，进而可以确定各译码阶段的结果是否可复用。

一种可能的实现方式，可以通过检查 $p_{x,y}(\mathcal{C}^{(t+1)})$ 与 $p_{x,y}(\mathcal{C}^{(t)})$ 的关系，和/或，$s_{x,y}(\mathcal{C}^{(t+1)})$ 与 $s_{x,y}(\mathcal{C}^{(t)})$ 的关系来确定译码的中间结果是否可被复用。

情况 1，$s_{x,y}(\mathcal{C}^{(t+1)})=s_{x,y}(\mathcal{C}^{(t)})$，且 $p_{x,y}(\mathcal{C}^{(t+1)})\subset p_{x,y}(\mathcal{C}^{(t)})$ 时，译码的中间结果可被复用。也就是说，如果 $s_{x,y}(\mathcal{C}^{(t+1)})=s_{x,y}(\mathcal{C}^{(t)})$，$p_{x,y}(\mathcal{C}^{(t+1)})\subset p_{x,y}(\mathcal{C}^{(t)})$，那么阶段 t+1 时的 CBT 可为阶段 t 时的 CBT 的子集，可以直接获得，无需再次计算。

情况 2，$s_{x,y}(\mathcal{C}^{(t+1)})\subset s_{x,y}(\mathcal{C}^{(t)})$ 时，译码的中间结果可被复用。例如，可以保存中间过程中的最大化操作结果来降低阶段 t+1 的计算量。

方式 1 可以用于 polar 码译码的每一阶段。例如，在 polar 码译码的每一阶段，都可以判断上述两种情况是否满足。

方式 2，简化计算量。

一种可能的设计，可以从所有 CBT 的元素（entry）中减去相同的值，该处理不会影响最终结果。为理解，下面列举具体示例。

一示例，对于 $k'=2,\ k''=1$，考虑计算：

$$CBT_{x,y}[0].e=\max\left(CBT_{x,z}[0].e+CBT_{z,y}[0].e, CBT_{x,z}[1].e+CBT_{z,y}[1].e\right)$$

$$CBT_{x,y}[1].e=\max\left(CBT_{x,z}[0].e+CBT_{z,y}[1].e, CBT_{x,z}[1].e+CBT_{z,y}[0].e\right)$$

$$\Delta=CBT_{x,y}[0].e-CBT_{x,y}[1].e=L_{x,z}\boxplus L_{z,y}$$

可以看出：

$$\Delta=CBT_{x,y}[0].e-CBT_{x,y}[1].e=L_{x,z}\boxplus L_{z,y}$$

其中，$L_{p,q}=CBT_{p,q}[0].e-CBT_{p,q}[1].e$；$a\boxplus b=\mathrm{sgn}(a)\mathrm{sgn}(b)\min\left(|a|,|b|\right)$。

例如，可以设置 $CBT_{x,y}[0].e=\Delta$，$CBT_{x,y}[1].e=0$。在该情况下，仅需要 3 次操作，相比于直接的做法（直接的做法需要 6 次操作），可以减少很多计算量，降低译码时长。

又如，如果 $CBT_{p,q}[1].e=0$，还可以进一步减少计算量，如可以消除 2 次减法。

又一示例，对于 $k'=1,\ k''=0$，考虑计算：

$$CBT_{x,y}[v].e=CBT_{x,z}[v].e+CBT_{z,y}[v].e,\quad v\in\{0,1\}$$

如果可以保证对于任何信道向量 $CBT_{x,z}[1].e=CBT_{z,y}[v].e=0$，那么可以设置 $CBT_{x,z}[1].e=0$，在这种情况下，仅需一次加法。

应理解，在核处理过程中，通过一些计算上的处理，以降低计算量的方式，都适用于

本申请实施例。

在对很多不同的核处理时，通过上述方式2，可以大大降低计算量，降低译码时长。

在上文方面6中，主要以$u_0^{t-1}=0$为例进行了示例性说明。当 $u_0^{t-1}\neq 0$的时，关于$CBT_{x,y}[v].e$的公式替换为下面公式即可：

$$CBT_{x,y}[v].e=\max_{w\in F_2^{k^*_{x-y}}}\left(CBT_{x,z}[a+\delta'].\varepsilon+CBT_{z,y}[b+\delta''].\varepsilon\right)$$

其中，δ'、δ''用于表示偏移量，即δ'、δ''为取决于u_0^{t-1}的偏移量。

上文结合方面1至方面6详细介绍了译码侧的方案，下面为理解，结合具体示例举例说明。

示例一，针对2次迭代的 Arikan 矩阵核处理过程。

令$K=M(2)F_2^{\otimes 2}=\begin{pmatrix}1&0&0&0\\1&0&1&0\\1&1&0&0\\1&1&1&1\end{pmatrix}$。

以图35为例。对于$C^{(0)}$，$K^{(0)}=\begin{pmatrix}1&0&0&0&1\\1&0&1&0&0\\1&1&0&0&0\\1&1&1&1&0\end{pmatrix}$。

可以知道：$p_{0,4}\left(C^{(0)}\right)$是一个(4，4)码，$s_{0,4}\left(C^{(0)}\right)$是一个(4，3)码。可以计算得出陪集$p_{0,4}\left(C^{(0)}\right)/s_{0,4}\left(C^{(0)}\right)$的陪集首为：(0,0,0,0)和(1,0,0,0)。

$p_{0,2}\left(C^{(0)}\right)$、$p_{2,4}\left(C^{(0)}\right)$为(2，2)码，$s_{0,2}\left(C^{(0)}\right)$、$s_{2,4}\left(C^{(0)}\right)$为(2，1)码。在时刻2时的状态为：$S_2$ (0,0,0,0)={(0,0)}和 S_2 (1,0,0,0)={(1,0)}。因此，可以得到下式：

$$CBT_{0,2}^{(0)}=\left[\max\left(e_{0,0}+e_{0,1},e_{1,0}+e_{1,1}\right),\max\left(e_{1,0}+e_{0,1},e_{0,0}+e_{1,1}\right)\right]$$

$$CBT_{2,4}^{(0)}=\left[\max\left(e_{0,2}+e_{0,3},e_{1,2}+e_{1,3}\right),\max\left(e_{1,2}+e_{0,3},e_{0,2}+e_{1,3}\right)\right]$$

$$CBT_{0,4}^{(0)}=[\max\left(CBT_{0,2}^{(0)}[0]+CBT_{2,4}^{(0)}[0],CBT_{0,2}^{(0)}[1]+CBT_{2,4}^{(0)}[1]\right),$$
$$\max\left(CBT_{0,2}^{(0)}[0]+CBT_{2,4}^{(0)}[1],CBT_{0,2}^{(0)}[1]+CBT_{2,4}^{(0)}[0]\right)]$$

其中，$e_{i,j}=\tau\left(S\left(y_j\right),i\right)$。

以图36为例。对于$C^{(1)}$，$K^{(1)}=\begin{pmatrix}1&0&1&0&1\\1&1&0&0&0\\1&1&1&1&0\end{pmatrix}$。

可以知道：$p_{x,x+2}\left(C^{(1)}\right)=p_{x,x+2}\left(C^{(0)}\right),s_{x,x+2}\left(C^{(1)}\right)=s_{x,x+2}\left(C^{(0)}\right),x\in\{0,2\}$，$p_{0,4}\left(C^{(1)}\right)$是一

个(4，3)码，$s_{0,4}\left(C^{(1)}\right)$是一个(4，2)码，考虑其陪集中的向量(0,0,0,0)和(1,0,1,0)，假设$u_0=0$，可以得到下式:

$$CBT_{0,4}^{(0)}=\left[\max\left(CBT_{0,2}^{(0)}[0]+CBT_{2,4}^{(0)}[0],CBT_{0,2}^{(0)}[1]+CBT_{2,4}^{(0)}[1]\right)\right]$$

以图 37 为例。对于$C^{(2)}$，$K^{(2)}=\begin{pmatrix}1&1&0&0&1\\1&1&1&1&0\end{pmatrix}$。

可以知道：$p_{0,2}\left(C^{(2)}\right)$、$p_{2,4}\left(C^{(2)}\right)$为 (2，1)码；$s_{0,2}\left(C^{(2)}\right)$、$s_{2,4}\left(C^{(2)}\right)$为 (2，0)码；$p_{0,4}\left(C^{(2)}\right)$为(4，2)码，$s_{0,4}\left(C^{(2)}\right)$是一个(4，1)码，考虑其陪集中的向量(0,0,0,0)和(1,1,0,0)，所以 S_2(0,0,0,0)={(0,0)}和 S_2(1,1,0,0)={(1,1)}，所以仅需考虑(2,0)码的陪集(0,0)和(1,1)。假设$u_0=u_1=0$时，可以得到:

$$CBT_{0,2}^{(0)}=\left[e_{0,0}+e_{0,1},e_{1,0}+e_{1,1}\right],\quad CBT_{2,4}^{(0)}=\left[e_{0,2}+e_{0,3},e_{1,2}+e_{1,3}\right]$$

$$CBT_{0,4}^{(2)}=[\max\left(CBT_{0,2}^{(2)}[0]+CBT_{2,4}^{(2)}[0],CBT_{0,2}^{(2)}[1]+CBT_{2,4}^{(2)}[1]\right),$$
$$\max\left(CBT_{0,2}^{(2)}[0]+CBT_{2,4}^{(2)}[1],CBT_{0,2}^{(2)}[1]+CBT_{2,4}^{(2)}[0]\right)]$$

对于$C^{(3)}$，$K^{(3)}=\begin{pmatrix}1&1&1&1&1\end{pmatrix}$。

可以知道：$p_{0,2}\left(C^{(3)}\right)=p_{0,2}\left(C^{(2)}\right)=p_{2,4}\left(C^{(2)}\right)=p_{2,4}\left(C^{(3)}\right)$为(2，1)码，$s_{0,2}\left(C^{(2)}\right)$、$s_{2,4}\left(C^{(2)}\right)$为(2，0)码，$p_{0,4}\left(C^{(3)}\right)$为(4，1)码，$s_{0,4}\left(C^{(4)}\right)$为(4，0)码，可以得到:

$$CBT_{0,4}^{(3)}=\left[CBT_{0,2}^{(2)}[0]+CBT_{2,4}^{(2)}[0],CBT_{0,2}^{(2)}[1]+CBT_{2,4}^{(2)}[1]\right]$$

上文示例地介绍了针对 2 次迭代的 Arikan 矩阵核处理过程。上述具体的核处理过程仅是示例性说明，对此不作严格限定。

示例二，针对高维核矩阵的核处理过程。

以 SC 译码为例，将上文所述的基于递归的网格译码方法可以应用于 polar 码 SC 译码的每一阶段。假设核矩阵 K_1 如图 38 所示，对此 polar 码进行基于本申请实施例提供的译码方法，图 39 显示了与此内核相对应的部分递归网格图。

在一些情况下，如 $s_{x,y}(\mathcal{O}^{(t+1)})=s_{x,y}(\mathcal{O}^{(t)})$，$p_{x,y}(\mathcal{O}^{(t+1)})\subset p_{x,y}(\mathcal{O}^{(t)})$，那么阶段 t+1 时的 CBT 可为阶段 t 时的 CBT 的子集，可以直接获得，无需再次计算。由图 39（1）和图 39（2）可以看出，在译码时 t=1 阶段，除了加粗箭头指示的部分，其余部分的网格和 t=0 阶段完全一致，这说明当译码在 t=1 阶段时，中间的结果都可以复用 t=0 阶段时的结果，可以无需再次计算，可以看出大量的计算量都节省了。

在一些情况下，如 $s_{x,y}(\mathcal{O}^{(t+1)})\subset s_{x,y}(\mathcal{O}^{(t)})$ 时，译码的中间结果可被复用。例如，可以保存中间过程中的最大化操作结果来降低阶段 t+1 的计算量。由图 39（3）和图 39（4）可以看出，译码在 t=5 阶段时中间的比较操作(最大化选择操作的结果)，在 t=6 阶段可以被

复用，由此又能带来计算量的降低。

通过本申请实施例提供的译码方法，在 polar 码译码的不同阶段，如果前序阶段已经计算过的步骤在后续阶段重新出现，此时直接复用之前的计算结果，这能使得 polar 的译码复杂度大大降低。

表 1 示出了在编码侧采用不同的核矩阵时，采用本申请实施例提供的译码算法和 Viterbi 译码算法的复杂度比较。

表 1

核（Kernel）	Viterbi 译码算法		本申请实施例的译码算法	
	summation	comparison	summation	comparison
K_{20}	1189	2392	497	309
K'_{20}	3346	6714	1685	1279
K''_{20}	7524	15054	4051	3341
K_{24}	2102	4218	715	449
K'_{24}	4705	9424	2039	1483
K''_{24}	10043	20100	5523	4371
K_{28}	2865	5776	1005	671
K_{32}	4536	9072	341	191

由表 1 可以看出，本申请实施例提供的译码算法相比于现有算法，复杂度得到了大幅降低。

可选地，本申请实施例不仅可以用于二进制域，还可以用于非二进制域。表 2 示出了在非二进制域上，采用本申请实施例提供的译码算法和 Viterbi 译码算法的复杂度比较。其中，Reed-Solomon 表示里德-所罗门码，Hermitian 表示埃尔米特。

表 2

核（Kernel）	Viterbi 译码算法		本申请实施例的译码算法	
	summation	comparison	summation	comparison
4x4 Reed-Solomon over GF（2^2）	177	232	140	116
8x8 Reed-Solomon over GF（2^3）	66017	75440	46264	45896
8x8 Hermitian over GF（2^2）	2484	3304	1256	1072

从表 2 可以看出，采用本申请实施例提供的译码算法，在非二进制域依旧能带来大幅的复杂度降低。

此外，图 40 和图 41 示出了 (1024，512)的 polar 子码 polar subcode 在编码侧采用核 K_{32}、

核 K'_{32} 以及 Arikan 核时，译码侧译码时的性能和复杂度对比。图 40 和图 41 中的纵坐标 FER 表示误帧率（frame error ratio），图 40 中的横坐标表示空间的大小或者长度（list size），图 41 中的横坐标表示加法和比较的数量（number of summations and comparisons）。从图 40 可以看出，在相同 list size 的情况下，采用了核 K_{32} 和核 K'_{32} 时，采用本申请实施例提供的译码算法，能带来性能的提升。在复杂度方面，可以看出，表征 Arikan 核的线与表征核 K_{32} 和核 K'_{32} 的线，在不同的 FER 处有交叉。具体来说，可以看出，核 K_{32} 对应的曲线和采用 Arikan 的核的曲线在 FER=3×10^{-4} 交叉处，K_{32} 对应的曲线需要的 list size 为 L=32，Arikan 核对应的曲线需要 list size 为 L=200。采用核 K'_{32} 时，达到和 Arikan 核相同 FER 时，需要的 list size 为 L=96，此时采用 Arikan 的核需要的 list size 为 L=4096。

图 42 和图 43 示出了 polar 码和低密度奇偶校验（Low-density Parity-check，LDPC 码），性能和复杂度上的对比结果。图 42 和图 43 中的横坐标 E_b/N_0 表示误比特率，E_b 表示每比特（bit）信号能量，N_0 表示噪声的功率谱密度，图 42 中的纵坐标 FER 表示误帧率（frame error ratio），图 43 中的纵坐标表示操作的平均数量（average number of operations）。从图 42 和图 43 可以看出，polar 码在性能和复杂度上都更有优势。

应理解，在本申请中的各个实施例中的公式仅是示例性说明，其不对本申请实施例的保护范围造成限定。上述各个实施例中的公式主要是结合当前技术中的设计给出的示例，各个参数的定义可以是一般意义上的定义。在计算上述各个涉及的参数的过程中，也可以根据上述公式进行计算，或者基于上述公式的变形进行计算，也可以根据其它方式进行计算以满足公式计算的结果。

还应理解，编码侧和译码侧的方案可以单独使用，也可以结合使用，对此不作限定。例如，可以采用本申请实施例提供的编码方法进行编码，采用现有的译码方法进行译码。又如，可以采用现有的编码方法进行编码，采用本申请实施例提供的译码方法进行译码。又如，可以采用本申请实施例提供的编码方法进行编码，并且采用本申请实施例提供的译码方法进行译码。一示例，在将数据矢量与某些矩阵（内核）的克罗内克乘积相乘来实现编码（polar 码编码），并且在接收端采用基于递归网格的译码方法进行译码。

还应理解，在上述一些示例中，以 SC 译码为例进行示例性说明，对此不作限定。例如，本申请实施例还可以用于 SCL，序列译码（sequential decoding），SC-Flip 等译码方式中。

基于上述技术方案，在一些实施例中，采用网格 Trellis 的译码算法对待译码序列进行译码时，可以先对网格图进行分割，把在完整的 Trellis 上进行译码的过程分割成对子网格上进行译码的过程，从而可以降低译码复杂度。此外，在不同的子网格上进行计算时，有些计算步骤可能相同，所以可以多次复用，这使得译码复杂度得到了很大程度的降低，且提高了译码速度。

此外，基于上述技术方案，在一些实施例中，采用极化编码时，可以先构造出多个核矩阵，基于构造出的核矩阵，根据实际通信情况，选择合适的核矩阵进行编码。通过该方式，在核矩阵构造或选择时，就综合考虑性能和译码复杂度，能够实现通过选择合适的编

码时所使用的核矩阵，降低译码复杂度。

本文中描述的各个实施例可以为独立的方案，也可以根据内在逻辑进行组合，这些方案都落入本申请的保护范围中。例如，编码侧和译码侧的方案可以单独使用，也可以结合使用。

可以理解的是，上述各个方法实施例中，由译码端设备（如网络设备或终端设备）实现的方法和操作，也可以由可用于译码端设备的部件（例如芯片或者电路）实现，由编码端设备（如网络设备或终端设备）实现的方法和操作，也可以由可用于编码端设备的部件（例如芯片或者电路）实现。

以上详细说明了本申请实施例提供的方法。以下，结合图 44 至图 46 详细说明本申请实施例提供的装置。应理解，装置实施例的描述与方法实施例的描述相互对应，因此，未详细描述的内容可以参见上文方法实施例，为了简洁，这里不再赘述。

本申请实施例可以根据上述方法示例对编码端设备或者译码端设备进行功能模块的划分，例如，可以对应各个功能划分各个功能模块，也可以将两个或两个以上的功能集成在一个处理模块中。上述集成的模块既可以采用硬件的形式实现，也可以采用软件功能模块的形式实现。需要说明的是，本申请实施例中对模块的划分是示意性的，仅仅为一种逻辑功能划分，实际实现时可以有另外的划分方式。下面以采用对应各个功能划分各个功能模块为例进行说明。

图 44 是本申请实施例提供的装置的示意性框图。该装置 4400 包括收发单元 4410 和处理单元 4420。收发单元 4410 可以实现相应的通信功能，处理单元 4420 用于进行数据处理。收发单元 4410 还可以称为通信接口或通信单元。

可选地，该装置 4400 还可以包括存储单元，该存储单元可以用于存储指令和/或数据，处理单元 4420 可以读取存储单元中的指令和/或数据，以使得装置实现前述方法实施例。

该装置 4400 可以用于执行上文方法实施例中编码端设备所执行的动作，这时，该装置 4400 可以为编码端设备或者可配置于编码端设备的部件，收发单元 4410 用于执行上文方法实施例中编码端设备侧的收发相关的操作，处理单元 4420 用于执行上文方法实施例中编码端设备侧的处理相关的操作。

或者，该装置 4400 可以用于执行上文方法实施例中译码端设备所执行的动作，这时，该装置 4400 可以为译码端设备或者可配置于译码端设备的部件，收发单元 4410 用于执行上文方法实施例中译码端设备侧的收发相关的操作，处理单元 4420 用于执行上文方法实施例中译码端设备侧的处理相关的操作。

作为另一种设计，该装置 4400 用于执行上文图 14 所示实施例中编码端设备所执行的动作，收发单元 4410 用于：获取信息比特；处理单元 4420 用于：确定目标核矩阵，目标核矩阵是通过调整原始核矩阵获得的；处理单元 4420 还用于：基于目标核矩阵，对信息比特进行极化编码。

作为一示例，目标核矩阵的维度小于原始核矩阵的维度。

作为又一示例，目标核矩阵是通过调整原始核矩阵的部分距离 partial distance 获得的。

作为又一示例，目标核矩阵是通过调整原始核矩阵的 partial distance，并且基于深度优先搜索算法获得的。

作为又一示例，在基于深度优先搜索算法进行处理时，使用以下一项或多项约束条件

进行处理：候选行集合 M 满足：$M[i]=\left\{v_0^{l-1}\in F_2^l \mid wt\left(v_0^{l-1}\right)=D_i\right\}$；或者，从所述候选行集合 M 中选择倒数第 i 行时，选择出来的行 v 汉明重量大于或等于 D_i；或者，在从所述候选行集合 M 中选择出所述行 v，并将其跟已构建好的行进行 partial distance 计算，如果 $c\in C_{\overline{K}}^{(\varphi+1)}\oplus\overline{K}[\varphi]$，$wt(c)<D_\varphi$，那么直接抛弃所述行 v，从所述候选行集合 M 中选择新的行；或者，属于同一陪集的行不会再次被选中；其中，i 用于表示行号，$i\in[0,l-1]$；$wt(v)$ 和 $wt(c)$ 均表示汉明重量；$\overline{K}$ 表示当前构建的核矩阵 $\overline{K}$，$\overline{K}[\varphi]$ 表示核矩阵 $\overline{K}$ 的第 φ 行；D_i 和 D_φ 均表示 partial distance；$C_{\overline{K}}^{(\varphi+1)}$ 表示核矩阵 $\overline{K}$ 对应的码；$\oplus$ 表示逻辑运算。

作为又一示例，原始核矩阵为维度为 (24×24) 的核矩阵 K_{24}，K_{24} 为：

1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0
1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0
1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 1 0 1 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0
0 1 1 0 1 0 1 0 0 1 1 0 1 0 1 0 0 0 0 0 0 0 0 0
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 1 1 1 0 0 1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0
1 1 0 0 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0
1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0

作为又一示例，目标核矩阵为以下任一项：

$$K_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&1&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&1&1&0&0&1&1&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&1&1&1&1&1&1&1&1
\end{pmatrix}$$

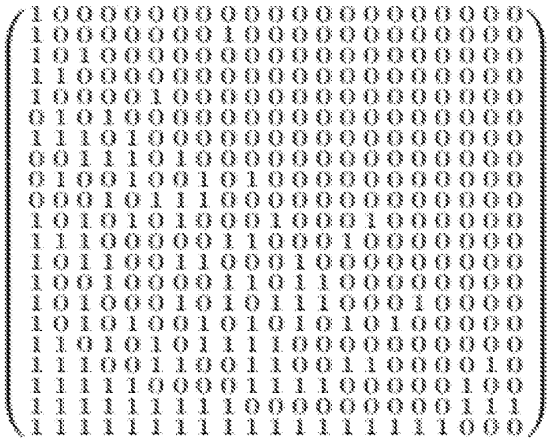
；或者，

$$K'_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&0&0&1&0&1&0&1&0&1&0&0&0&0&0\\
1&0&1&0&1&1&0&0&0&0&0&0&1&0&1&0&1&0&1&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&1&1\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0
\end{pmatrix}$$

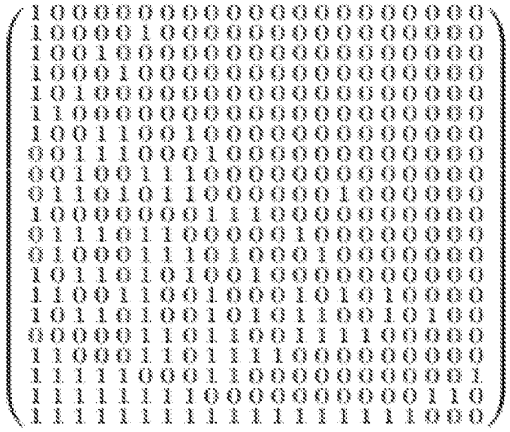
；或者，

$$K''_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&1&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&1&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0\\
1&1&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&0&1&0&0&1&0&0&0&1&0&0&0&1&0&1&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&0&0&0&0\\
0&0&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0
\end{pmatrix}$$

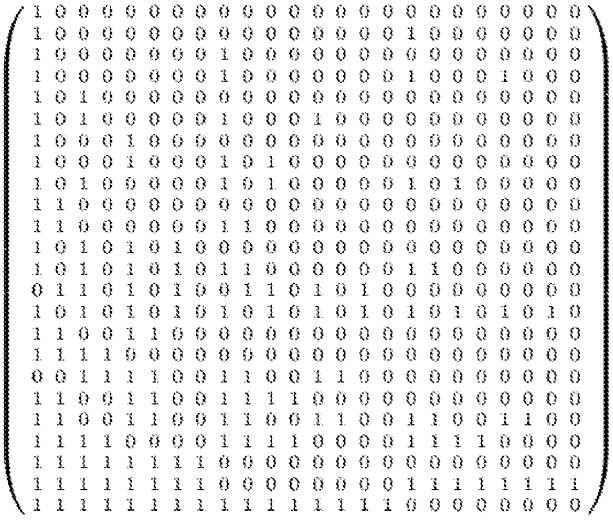
；或者，

$K_{21}=$ 

；或者，

$K'_{21}=$ 

；或者，

$K_{24}=$ 

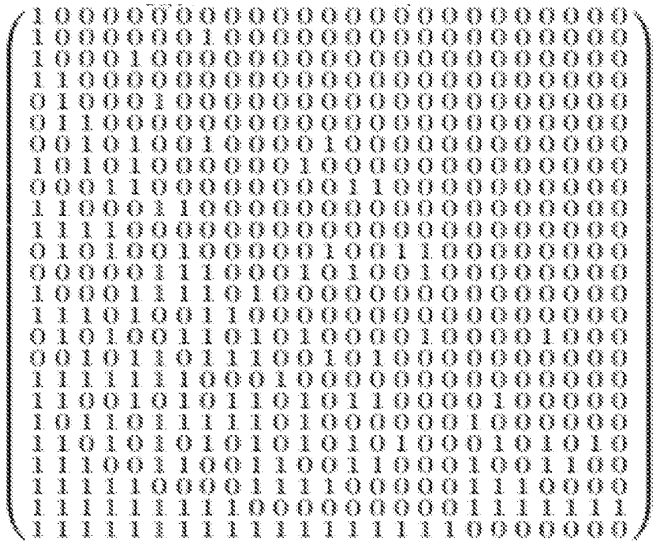
；或者，

$$K'_{24} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&1&1&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0
\end{pmatrix}$$

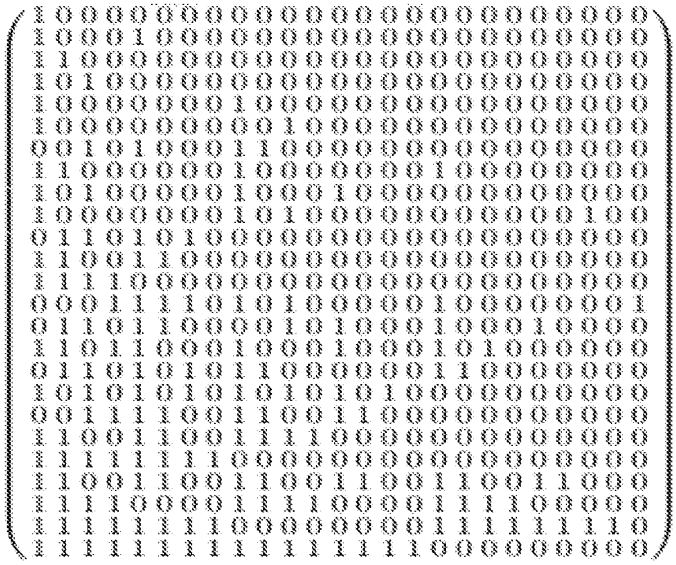
；或者，

$$K''_{24} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&1&1&0&1&0&0&0&1&0&0&0&1&0&0&0\\
0&1&1&1&0&0&1&0&1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0
\end{pmatrix}$$

；或者，

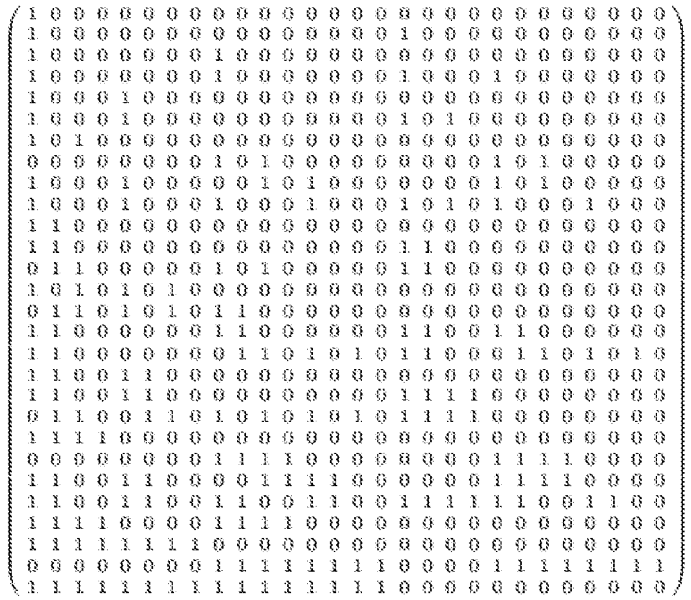
$$K_{25}=$$ 

；或者，

$$K'_{25}=$$ 

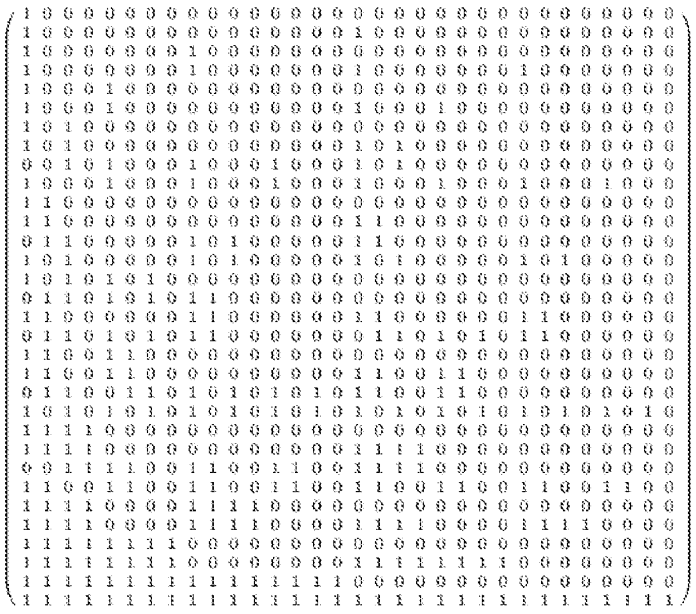
；或者，

$$K_{28}=$$



；或者，

$$K_{32}=$$



。

该装置 4400 可实现对应于根据本申请方法实施例中的编码端设备执行的步骤或者流程，该装置 4400 可以包括用于执行图 13 至图 43 的方法实施例中的编码端设备执行的方法的单元。并且，该装置 4400 中的各单元和上述其他操作和/或功能分别为了实现图 13 至图 43 的方法实施例中的相应流程。

其中，当该装置 4400 用于执行图 13 中的方法 1300 时，收发单元 4410 可用于执行方法 1300 中的步骤 1310，处理单元 4420 可用于执行方法 1300 中的步骤 1320 和 1330。

应理解，各单元执行上述相应步骤的具体过程在上述方法实施例中已经详细说明，为了简洁，在此不再赘述。

作为另一种设计，该装置 4400 用于执行上文图 14 所示实施例中译码端设备所执行的动作，收发单元 4410 用于：获取待译码序列；处理单元 4420 用于：基于多个网格，对待译码序列进行译码，基于多个网格中的第二网格译码时，复用多个网格中的第一网格译码的中间结果，其中，第一网格用于对待译码序列译码的第 t 阶段，第二网格用于对待译码序列译码的第 t+i 阶段，t 为大于 0 或等于 0 的整数，i 为大于 1 或等于 1 的整数。

作为一示例，多个网格中的每个网格被分割为多个子网格。

作为又一示例，多个子网格的分割点的位置与译码复杂度相关。

作为又一示例，处理单元 4420 还用于：根据第一网格和第二网格的关联度，确定基于第二网格译码时是否复用第一网格译码的中间结果。

作为又一示例，处理单元 4420 具体用于：根据以下信息，判断基于第二网格译码时是否复用第一网格译码的中间结果：译码的第 t+i 阶段通过对线性码 C 进行缩短操作得到的码与译码的第 t 阶段通过对线性码 C 进行缩短操作得到的码；和/或，译码的第 t+i 阶段通过对线性码 C 进行打孔操作得到的码与译码的第 t 阶段通过对线性码 C 进行打孔操作得到的码。

作为又一示例，如果 $s_{x,y}(\mathcal{C}^{(t+i)})=s_{x,y}(\mathcal{C}^{(t)})$，$p_{x,y}(\mathcal{C}^{(t+i)})\subset p_{x,y}(\mathcal{C}^{(t)})$，那么基于第二网格译码时，复用第一网格译码的中间结果；或者；如果 $s_{x,y}(\mathcal{C}^{(t+i)})\subset s_{x,y}(\mathcal{C}^{(t)})$，那么基于第二网格译码时，复用第一网格译码的中间结果；其中，$s_{x,y}(\mathcal{C}^{(t+i)})$表示译码的第 t+i 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行缩短操作得到的码，$s_{x,y}(\mathcal{C}^{(t)})$表示译码的第 t 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行缩短操作得到的码，$p_{x,y}(\mathcal{C}^{(t+i)})$表示译码的第 t+i 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行打孔操作得到的码，$p_{x,y}(\mathcal{C}^{(t)})$表示译码的第 t 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行打孔操作得到的码。

作为又一示例，处理单元 4420 具体用于：在译码的第 t+i 阶段基于第二网格处理时，直接将基于第一网格译码的中间结果，作为第二网格译码的中间结果；或者，在译码的第 t+i 阶段基于第二网格处理时，基于第一网格处理过程中的最大化操作的结果进行处理。

该装置 4400 可实现对应于根据本申请方法实施例中的译码端设备执行的步骤或者流程，该装置 4400 可以包括用于执行图 13 至图 43 的方法实施例中的译码端设备执行的方法的单元。并且，该装置 4400 中的各单元和上述其他操作和/或功能分别为了实现图 13 至图 43 的方法实施例中的相应流程。

其中，当该装置 4400 用于执行图 14 中的方法 1400 时，收发单元 4410 可用于执行方法 1400 中的步骤 1410，处理单元 4420 可用于执行方法 1400 中的步骤 1420。

上文实施例中的处理单元 4420 可以由至少一个处理器或处理器相关电路实现。收发单元 4410 可以由收发器或收发器相关电路实现。收发单元 4410 还可称为通信单元或通信接口。存储单元可以通过至少一个存储器实现。

在一个可能的设计中，装置 4400 的上述功能可以通过硬件实现，或者通过硬件执行相应的软件实现。

作为一个实施例，装置 4400 可以包括一个或多个处理器，所述一个或多个处理器用

于执行存储器中存储的计算机程序，以使装置 4400 执行本申请提供的任意一个方法实施例。

可选地，所述用于存储计算机程序的存储器位于装置 4400 之外，所述一个或多个处理器通过电路和/或电线与所述存储器连接。所述存储器可以为一个或多个。

可选地，装置 4400 还包括一个或多个存储器。

进一步可选地，装置 4400 还包括一个或多个通信接口。

作为一些示例，所述一个通信或多个通信接口可以为输入输出接口，或者输出输出电路，本申请对此不作限定。

作为另一个实施例中，装置 4400 还可以通过硬件实现。

如图 45 所示，本申请实施例还提供一种装置 4500。该装置 4500 包括处理器 4510，处理器 4510 与存储器 4520 耦合，存储器 4520 用于存储计算机程序或指令和/或数据，处理器 4510 用于执行存储器 4520 存储的计算机程序或指令和/或数据，使得上文方法实施例中的方法被执行。

可选地，该装置 4500 包括的处理器 4510 为一个或多个。

可选地，如图 45 所示，该装置 4500 还可以包括存储器 4520。

可选地，该装置 4500 包括的存储器 4520 可以为一个或多个。

可选地，该存储器 4520 可以与该处理器 4510 集成在一起，或者分离设置。

可选地，如图 45 所示，该装置 4500 还可以包括收发器 4530，收发器 4530 用于信号的接收和/或发送。例如，处理器 4510 用于控制收发器 4530 进行信号的接收和/或发送。

作为一种方案，该装置 4500 用于实现上文方法实施例中由编码端设备执行的操作。

例如，处理器 4510 用于实现上文方法实施例中由编码端设备执行的处理相关的操作，收发器 4530 用于实现上文方法实施例中由编码端设备执行的收发相关的操作。

作为另一种方案，该装置 4500 用于实现上文方法实施例中由译码端设备执行的操作。

例如，处理器 4510 用于实现上文方法实施例中由译码端设备执行的处理相关的操作，收发器 4530 用于实现上文方法实施例中由译码端设备执行的收发相关的操作。

本申请实施例还提供一种装置 4600，该装置 4600 可以用于执行上述方法实施例中由编码端设备所执行的操作，或者，该装置 4600 可以用于执行上述方法实施例中由译码端设备所执行的操作。

装置 4600 可以包括输入接口电路 4610、逻辑电路 4620 和输出接口电路 4620。

作为一种方案，该装置 4600 可以用于执行上述方法实施例中由译码端设备所执行的操作。

可选地，装置 4600 也可以为译码端中用于实现信道解码的器件或模块。例如，信道解码器或信道解码电路等。或者，装置 4600 也可以为配置在接收端中的芯片。

在一种可能的实现方式中，输入接口电路 4610，用于获取待译码序列；逻辑电路 4620，用于采用本申请提供的译码算法，对待译码序列进行译码；输出接口电路 4630，用于输出译码结果。

在又一种可能的实现方式中，逻辑电路 4620，用于获取待译码序列，并采用本申请提供的译码算法，对该待译码序列进行译码；以及，输出接口电路 4630 输出译码结果。

作为另一种方案，该装置 4600 可以用于执行上述方法实施例中由编码端设备所执行

的操作。

可选地，装置 4600 也可以为编码端中用于实现编码的器件或模块。例如，编码器或编码电路等。或者，装置 4600 也可以为配置在发送端中的芯片。

在一种可能的实现方式中，输入接口电路 4610，用于获取信息比特；逻辑电路 4620，用于采用本申请提供的编码算法，对信息比特进行编码；输出接口电路 4630，用于输出编码后的极化码字。

在又一种可能的实现方式中，逻辑电路 4620，用于生成信息比特，并采用本申请提供的编码算法，对该信息比特进行编码；以及，输出接口电路 4630 输出编码后的极化码字。

可选地，装置 4600 可以是芯片或者集成电路。例如，所述芯片可以为片上系统（system on chip，SOC），也可以是基带芯片等。

可选地，上述主要以输入接口电路 4610 和输出接口电路 4620 分别为单独的接口电路为例进行了示例性说明，对此不作限定。例如，输入接口电路 4610 和输出接口电路 4620 可以集成为一个接口。

本申请实施例还提供一种计算机可读存储介质，其上存储有用于实现上述方法实施例中由编码端设备执行的方法，或由译码端设备执行的方法的计算机指令。

例如，该计算机程序被计算机执行时，使得该计算机可以实现上述方法实施例中由编码端设备执行的方法，或由译码端设备执行的方法。

本申请实施例还提供一种包含指令的计算机程序产品，该指令被计算机执行时使得该计算机实现上述方法实施例中由编码端设备执行的方法，或由译码端设备执行的方法。

本申请还提供一种芯片，包括一个或多个存储器以及一个或多个处理器。所述一个或多个存储器用于存储计算机程序，所述一个或多个处理器用于从所述一个或多个存储器中调用并运行所述计算机程序，使得安装有所述芯片的设备执行使本申请的方法。

本申请还提供一种通信设备，包括上述装置 4500 或装置 4600。

本申请实施例还提供一种通信系统，该通信系统包括上文实施例中的编码端设备与译码端设备。

本文中的译码端，也即信号和/或数据的接收端。相应地，发送信号和/或数据的一方为发送端。可选地，译码端可以为通信系统中的网络设备（例如，5G 的 gNB），也可以为终端设备，本申请的方案不作限定。

本文中的编码端，也即信号和/或数据的发送端。相应地，接收信号和/或数据的一方为接收端。可选地，编码端可以为通信系统中的网络设备（例如，5G 的 gNB），也可以为终端设备，本申请的方案不作限定。

所属领域的技术人员可以清楚地了解到，为描述方便和简洁，上述提供的任一种通信装置中相关内容的解释及有益效果均可参考上文提供的对应的方法实施例，此处不再赘述。

在本申请实施例中，编码端设备或译码端设备可以包括硬件层、运行在硬件层之上的操作系统层，以及运行在操作系统层上的应用层。其中，硬件层可以包括中央处理器（central processing unit，CPU）、内存管理单元（memory management unit，MMU）和内存（也称为主存）等硬件。操作系统层的操作系统可以是任意一种或多种通过进程（process）实现

业务处理的计算机操作系统，例如，Linux 操作系统、Unix 操作系统、Android 操作系统、iOS 操作系统或 windows 操作系统等。应用层可以包含浏览器、通讯录、文字处理软件、即时通信软件等应用。

本申请实施例并未对本申请实施例提供的方法的执行主体的具体结构进行特别限定，只要能够通过运行记录有本申请实施例提供的方法的代码的程序，以根据本申请实施例提供的方法进行通信即可。例如，本申请实施例提供的方法的执行主体可以是编码端设备或译码端设备，或者，是编码端设备或译码端设备中能够调用程序并执行程序的功能模块。

本申请的各个方面或特征可以实现成方法、装置或使用标准编程和/或工程技术的制品。本文中使用的术语“制品”可以涵盖可从任何计算机可读器件、载体或介质访问的计算机程序。

其中，计算机可读存储介质可以是计算机能够存取的任何可用介质或者是包含一个或多个可用介质集成的服务器、数据中心等数据存储设备。可用介质（或者说计算机可读介质）例如可以包括但不限于：磁性介质或磁存储器件(例如，软盘、硬盘（如移动硬盘）、磁带)、光介质(例如，光盘、压缩盘（compact disc，CD）、数字通用盘（digital versatile disc，DVD）等)、智能卡和闪存器件（例如，可擦写可编程只读存储器（erasable programmable read-only memory，EPROM）、卡、棒或钥匙驱动器等）、或者半导体介质（例如固态硬盘（solid state disk，SSD）等、U 盘、只读存储器（read-only memory，ROM）、随机存取存储器（random access memory，RAM）等各种可以存储程序代码的介质。

本文描述的各种存储介质可代表用于存储信息的一个或多个设备和/或其它机器可读介质。术语“机器可读介质”可以包括但不限于：无线信道和能够存储、包含和/或承载指令和/或数据的各种其它介质。

应理解，本申请实施例中提及的处理器可以是中央处理单元（central processing unit，CPU），还可以是其他通用处理器、数字信号处理器（digital signal processor，DSP）、专用集成电路（application specific integrated circuit，ASIC）、现成可编程门阵列（field programmable gate array，FPGA）或者其他可编程逻辑器件、分立门或者晶体管逻辑器件、分立硬件组件等。通用处理器可以是微处理器或者该处理器也可以是任何常规的处理器等。

还应理解，本申请实施例中提及的存储器可以是易失性存储器或非易失性存储器，或可包括易失性和非易失性存储器两者。其中，非易失性存储器可以是只读存储器（read-only memory，ROM）、可编程只读存储器（programmable ROM，PROM）、可擦除可编程只读存储器（erasable PROM，EPROM）、电可擦除可编程只读存储器（electrically EPROM，EEPROM）或闪存。易失性存储器可以是随机存取存储器（random access memory，RAM）。例如，RAM 可以用作外部高速缓存。作为示例而非限定，RAM 可以包括如下多种形式：静态随机存取存储器（static RAM，SRAM）、动态随机存取存储器（dynamic RAM，DRAM）、同步动态随机存取存储器（synchronous DRAM，SDRAM）、双倍数据速率同步动态随机存取存储器（double data rate SDRAM，DDR SDRAM）、增强型同步动态随机存取存储器（enhanced SDRAM，ESDRAM）、同步连接动态随机存取存储器（synchlink DRAM，SLDRAM）和直接内存总线随机存取存储器（direct rambus RAM，DR RAM）。

需要说明的是，当处理器为通用处理器、DSP、ASIC、FPGA 或者其他可编程逻辑器

件、分立门或者晶体管逻辑器件、分立硬件组件时，存储器（存储模块）可以集成在处理器中。

还需要说明的是，本文描述的存储器旨在包括但不限于这些和任意其它适合类型的存储器。

在本申请所提供的几个实施例中，应该理解到，所揭露的装置和方法，可以通过其它的方式实现。例如，以上所描述的装置实施例仅是示意性的，例如，上述单元的划分，仅仅为一种逻辑功能划分，实际实现时可以有另外的划分方式，例如多个单元或组件可以结合或者可以集成到另一个系统，或一些特征可以忽略，或不执行。此外，所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口，装置或单元的间接耦合或通信连接，可以是电性，机械或其它的形式。

上述作为分离部件说明的单元可以是或者也可以不是物理上分开的，作为单元显示的部件可以是或者也可以不是物理单元，即可以位于一个地方，或者也可以分布到多个网络单元上。可以根据实际的需要选择其中的部分或者全部单元实现本申请提供的方案。

另外，在本申请各个实施例中的各功能单元可以集成在一个单元中，也可以是各个单元单独物理存在，也可以两个或两个以上单元集成在一个单元中。

在上述实施例中，可以全部或部分地通过软件、硬件、固件或者其任意组合来实现。

当使用软件实现时，可以全部或部分地以计算机程序产品的形式实现。该计算机程序产品包括一个或多个计算机指令。在计算机上加载和执行计算机程序指令时，全部或部分地产生按照本申请实施例所述的流程或功能。计算机可以是通用计算机、专用计算机、计算机网络、或者其他可编程装置。例如，计算机可以是个人计算机，服务器，或者网络设备等。计算机指令可以存储在计算机可读存储介质中，或者从一个计算机可读存储介质向另一个计算机可读存储介质传输，例如，计算机指令可以从一个网站站点、计算机、服务器或数据中心通过有线（例如同轴电缆、光纤、数字用户线（DSL））或无线（例如红外、无线、微波等）方式向另一个网站站点、计算机、服务器或数据中心进行传输。关于计算机可读存储介质，可以参考上文描述。

以上所述，仅为本申请的具体实施方式，但本申请的保护范围并不局限于此，任何熟悉本技术领域的技术人员在本申请揭露的技术范围内，可轻易想到的变化或替换，都应涵盖在本申请的保护范围之内。因此，本申请的保护范围应以权利要求和说明书的保护范围为准。

权 利 要 求 书

1. 一种编码方法，其特征在于，包括：

获取信息比特；

确定目标核矩阵，所述目标核矩阵是通过调整原始核矩阵获得的；

基于所述目标核矩阵，对所述信息比特进行极化编码。

2. 根据权利要求 1 所述的编码方法，其特征在于，

所述目标核矩阵的维度小于所述原始核矩阵的维度。

3. 根据权利要求 1 所述的编码方法，其特征在于，

所述目标核矩阵是通过调整所述原始核矩阵的部分距离 partial distance 获得的。

4. 根据权利要求 3 所述的编码方法，其特征在于，所述目标核矩阵是通过调整所述原始核矩阵的 partial distance，并且基于深度优先搜索算法获得的。

5. 根据权利要求 4 所述的编码方法，其特征在于，在基于所述深度优先搜索算法进行处理时，使用以下一项或多项约束条件进行处理：

候选行集合 M 满足：$M[i]=\left\{v_0^{l-1}\in F_2^l \mid wt\left(v_0^{l-1}\right)=D_i\right\}$；或者，

从所述候选行集合 M 中选择倒数第 i 行时，选择出来的行 v 汉明重量大于或等于 D_i；或者，

在从所述候选行集合 M 中选择出所述行 v，并将其跟已构建好的行进行 partial distance 计算，如果 $c\in C_{\overline{K}}^{(\varphi+1)}\oplus\overline{K}[\varphi]$，$wt(c)<D_\varphi$，那么直接抛弃所述行 v，从所述候选行集合 M 中选择新的行；或者，

属于同一陪集的行不会再次被选中；

其中，i 用于表示行号，$i\in[0,l-1]$；$wt(v)$和$wt(c)$均表示汉明重量；$\overline{K}$表示当前构建的核矩阵$\overline{K}$，$\overline{K}[\varphi]$表示核矩阵$\overline{K}$的第φ行；D_i和D_φ均表示 partial distance；$C_{\overline{K}}^{(\varphi+1)}$表示核矩阵$\overline{K}$对应的码；$\oplus$表示逻辑运算。

6. 根据权利要求 1 至 5 中任一项所述的编码方法，其特征在于，

所述原始核矩阵为维度为(24×24)的核矩阵K_{24}，所述K_{24}为：

1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0
1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0

1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 1 0 1 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0
0 1 1 0 1 0 1 0 0 1 1 0 1 0 1 0 0 0 0 0 0 0 0 0
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 1 1 1 0 0 1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0
1 1 0 0 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0
1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 。

7. 根据权利要求 1 至 6 中任一项所述的编码方法，其特征在于，所述目标核矩阵为以下任一项：

$$K_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&1&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&1&1&0&0&1&1&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&1&1&1&1&1&1&1&1
\end{pmatrix}$$

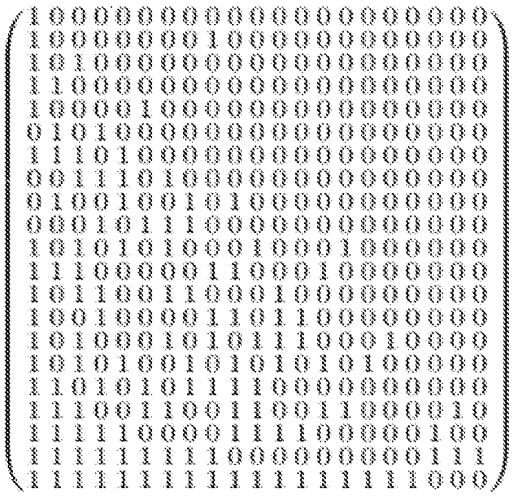
；或者，

$$K'_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&0&0&1&0&1&0&1&0&1&0&0&0&0&0\\
1&0&1&0&1&1&0&0&0&0&0&0&1&0&1&0&1&0&1&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&1&1\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0
\end{pmatrix}$$

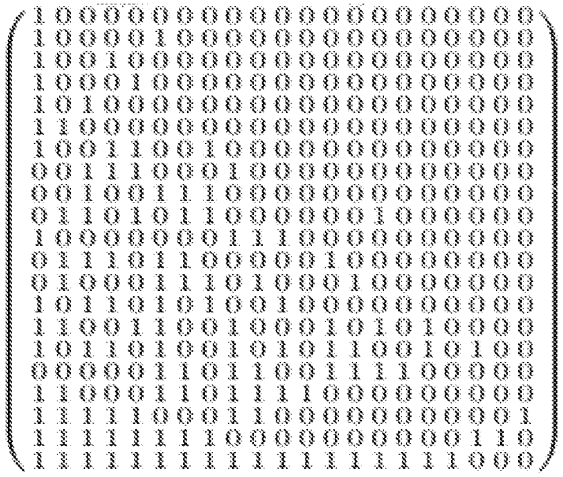
；或者，

$$K''_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&1&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&1&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0\\
1&1&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&0&1&0&0&1&0&0&0&1&0&0&0&1&0&1&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&0&0&0&0\\
0&0&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0
\end{pmatrix}$$

；或者，

K_{21}= 

；或者，

K'_{21}= 

；或者，

$$K_{24} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0
\end{pmatrix}$$

；或者，

$$K'_{24} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&1&0&1&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&1&1&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0
\end{pmatrix}$$

；或者，

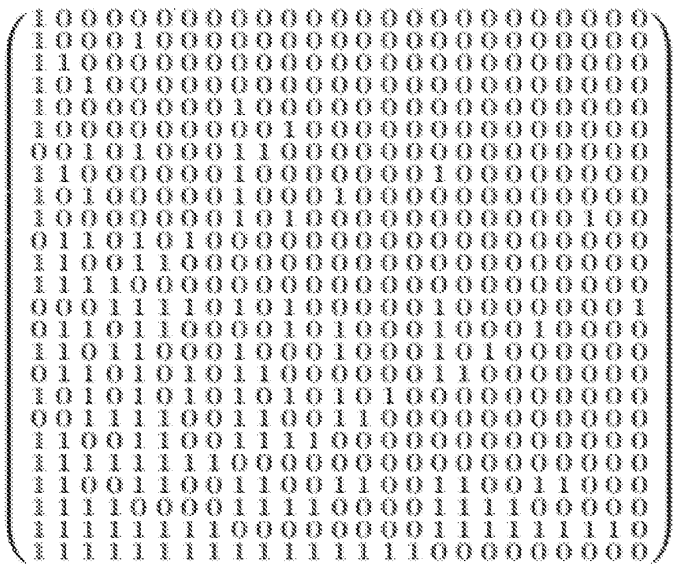
$$K_{24}'' = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&1&1&0&1&0&0&0&1&0&0&0&1&0&0&0\\
0&1&1&1&0&0&1&0&1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0
\end{pmatrix}$$

；或者，

$$K_{25} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&1&0&0&0&0&1&0&0&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&1&1&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0\\
0&0&1&0&1&1&0&1&1&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&0&1&0&1&1&0&1&0&1&1&0&0&0&0&1&0&0&0&0\\
1&0&1&1&0&1&1&1&1&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0\\
1&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&0&0&1&0&1&0&1&0\\
1&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&0&1&0&0&1&1&0&0\\
1&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0
\end{pmatrix}$$

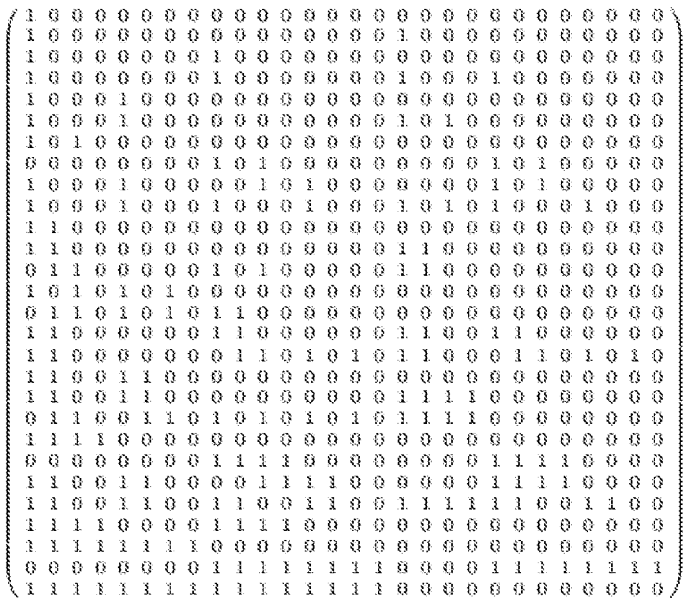
；或者，

$$K'_{25}=$$

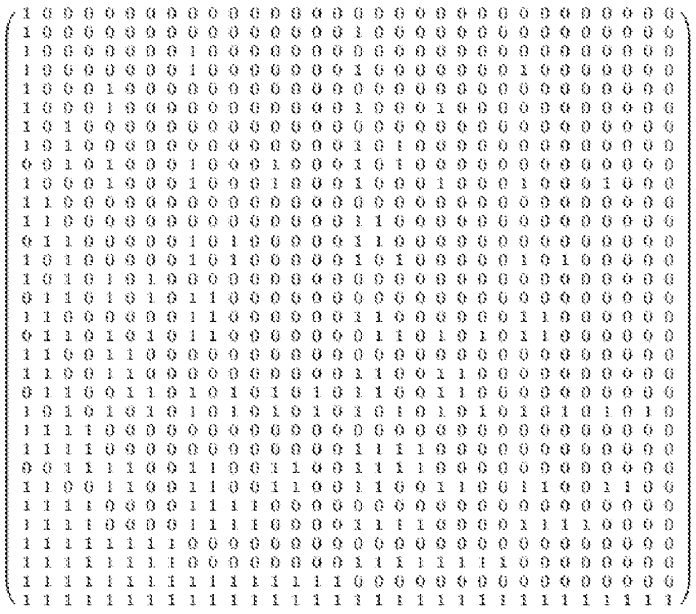


；或者，

$$K_{28}=$$



；或者，

$$K_{32}=$$ 

8. 一种译码方法，其特征在于，包括:

获取待译码序列;

基于多个网格，对所述待译码序列进行译码，基于所述多个网格中的第二网格译码时，复用所述多个网格中的第一网格译码的中间结果，

其中，所述第一网格用于对所述待译码序列译码的第 t 阶段，所述第二网格用于对所述待译码序列译码的第 t+i 阶段，t 为大于 0 或等于 0 的整数，i 为大于 1 或等于 1 的整数。

9. 根据权利要求 8 所述的译码方法，其特征在于，所述多个网格中的每个网格被分割为多个子网格。

10. 根据权利要求 9 所述的译码方法，其特征在于，所述多个子网格的分割点的位置与译码复杂度相关。

11. 根据权利要求 8 至 10 中任一项所述的译码方法，其特征在于，所述译码方法还包括:

根据所述第一网格和所述第二网格的关联度，确定基于所述第二网格译码时是否复用所述第一网格译码的中间结果。

12. 根据权利要求 8 至 11 中任一项所述的译码方法，其特征在于，所述译码方法还包括:

根据以下信息，判断基于所述第二网格译码时是否复用所述第一网格译码的中间结果:

所述译码的第 t+i 阶段通过对线性码 C 进行缩短操作得到的码与所述译码的第 t 阶段通过对线性码 C 进行缩短操作得到的码; 和/或，

所述译码的第 t+i 阶段通过对线性码 C 进行打孔操作得到的码与所述译码的第 t 阶段通过对线性码 C 进行打孔操作得到的码。

13. 根据权利要求 12 所述的译码方法，其特征在于，

如果 $s_{x,y}(\mathcal{C}^{(t+i)})=s_{x,y}(\mathcal{C}^{(t)})$，$p_{x,y}(\mathcal{C}^{(t+i)})\subset p_{x,y}(\mathcal{C}^{(t)})$，那么基于所述第二网格译码时，复用所述第一网格译码的中间结果；或者

如果 $s_{x,y}(\mathcal{C}^{(t+i)})\subset s_{x,y}(\mathcal{C}^{(t)})$，那么基于所述第二网格译码时，复用所述第一网格译码的中间结果；

其中，$s_{x,y}(\mathcal{C}^{(t+i)})$表示译码的第 t+i 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行缩短操作得到的码，$s_{x,y}(\mathcal{C}^{(t)})$表示译码的第 t 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行缩短操作得到的码，$p_{x,y}(\mathcal{C}^{(t+i)})$表示译码的第 t+i 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行打孔操作得到的码，$p_{x,y}(\mathcal{C}^{(t)})$表示译码的第 t 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行打孔操作得到的码。

14. 根据权利要求 8 至 13 中任一项所述的译码方法，其特征在于，

所述基于所述第二网格译码时，复用所述第一网格译码的中间结果，包括：

在所述译码的第 t+i 阶段基于所述第二网格处理时，直接将基于所述第一网格译码的中间结果，作为所述第二网格译码的中间结果；或者，

在所述译码的第 t+i 阶段基于所述第二网格处理时，基于所述第一网格处理过程中的最大化操作的结果进行处理。

15. 一种编码装置，其特征在于，包括：收发单元和处理单元，

所述收发单元，用于获取信息比特；

所述处理单元，用于确定目标核矩阵，所述目标核矩阵是通过调整原始核矩阵获得的；

所述处理单元，还用于基于所述目标核矩阵，对所述信息比特进行极化编码。

16. 根据权利要求 15 所述的编码装置，其特征在于，

所述目标核矩阵的维度小于所述原始核矩阵的维度。

17. 根据权利要求 15 所述的编码装置，其特征在于，

所述目标核矩阵是通过调整所述原始核矩阵的部分距离 partial distance 获得的。

18. 根据权利要求 17 所述的编码装置，其特征在于，所述目标核矩阵是通过调整所述原始核矩阵的 partial distance，并且基于深度优先搜索算法获得的。

19. 根据权利要求 18 所述的编码装置，其特征在于，在基于所述深度优先搜索算法进行处理时，所述处理单元，还用于使用以下一项或多项约束条件进行处理：

候选行集合 M 满足：$M[i]=\left\{v_0^{l-1}\in F_2^l \mid wt\left(v_0^{l-1}\right)=D_i\right\}$；或者，

从所述候选行集合 M 中选择倒数第 i 行时，选择出来的行 v 汉明重量大于或等于 D_i；或者，

在从所述候选行集合 M 中选择出所述行 v，并将其跟已构建好的行进行 partial distance 计算，如果 $c\in C_{\bar{K}}^{(\varphi+1)}\oplus\bar{K}[\varphi]$，$wt(c)<D_\varphi$，那么直接抛弃所述行 v，从所述候选行集合 M 中选择新的行；或者，

属于同一陪集的行不会再次被选中；

其中，i 用于表示行号，$i\in[0,l-1]$；$wt(v)$和$wt(c)$均表示汉明重量；$\bar{K}$表示当前构

建的核矩阵 $\overline{K}$，$\overline{K}[\varphi]$ 表示核矩阵 $\overline{K}$ 的第 φ 行；D_i 和 D_φ 均表示 partial distance；$C_{\overline{K}}^{(\varphi+1)}$ 表示核矩阵 $\overline{K}$ 对应的码；$\oplus$ 表示逻辑运算。

20. 根据权利要求 15 至 19 中任一项所述的编码装置，其特征在于，

所述原始核矩阵为维度为 (24×24) 的核矩阵 K_{24}，所述 K_{24} 为：

1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0
1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0
1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 1 0 1 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0
0 1 1 0 1 0 1 0 0 1 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 1 1 1 0 0 1 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0
1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 。

21. 根据权利要求 15 至 20 中任一项所述的编码装置，其特征在于，所述目标核矩阵为以下任一项：

$$K_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&1&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&1&1&0&0&1&1&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&1&1&1&1&1&1&1&1
\end{pmatrix}$$

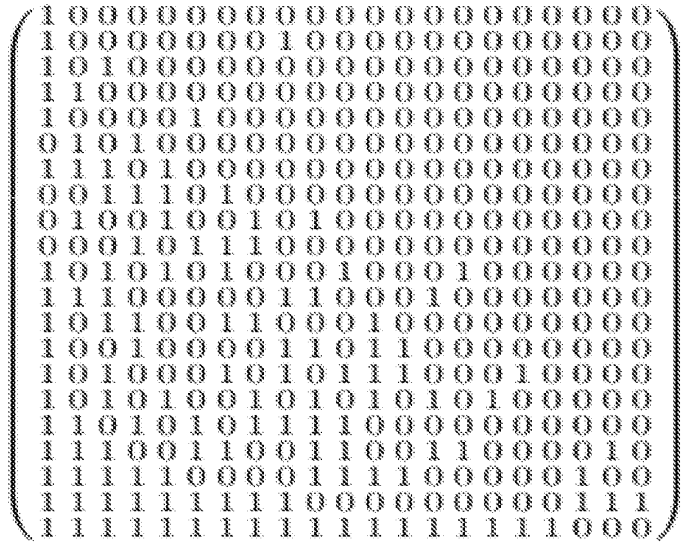
；或者，

$$K'_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&0&0&1&0&1&0&1&0&1&0&0&0&0&0\\
1&0&1&0&1&1&0&0&0&0&0&0&1&0&1&0&1&0&1&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&1&1\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0
\end{pmatrix}$$

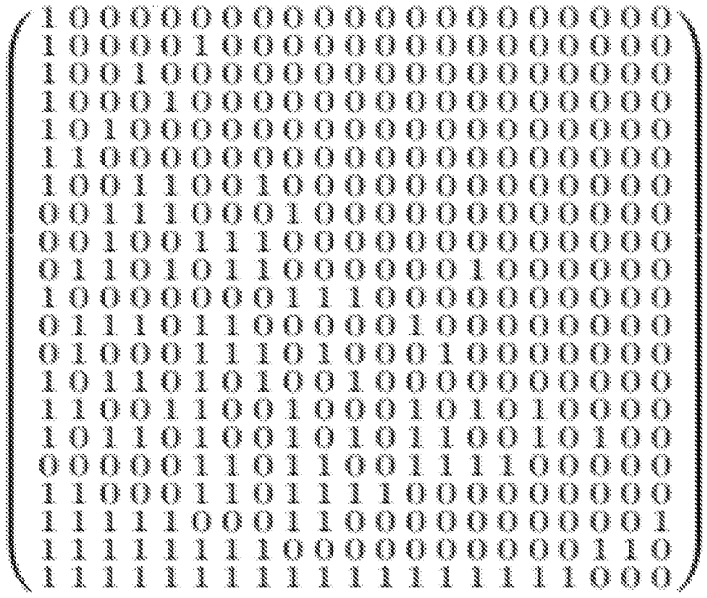
；或者，

$$K''_{20}=\begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&1&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&1&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0\\
1&1&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&0&1&0&0&1&0&0&0&1&0&0&0&1&0&1&0\\
0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&0&1&1&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&0&0&0&0\\
0&0&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0
\end{pmatrix}$$

；或者，

K_{21}= 

；或者，

K'_{21}= 

；或者，

$$K_{24} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0
\end{pmatrix}$$

；或者，

$$K'_{24} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&1&0&1&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&1&1&0&1&0&0&0&1&0&0&0&1&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&1&1&0&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0
\end{pmatrix}$$

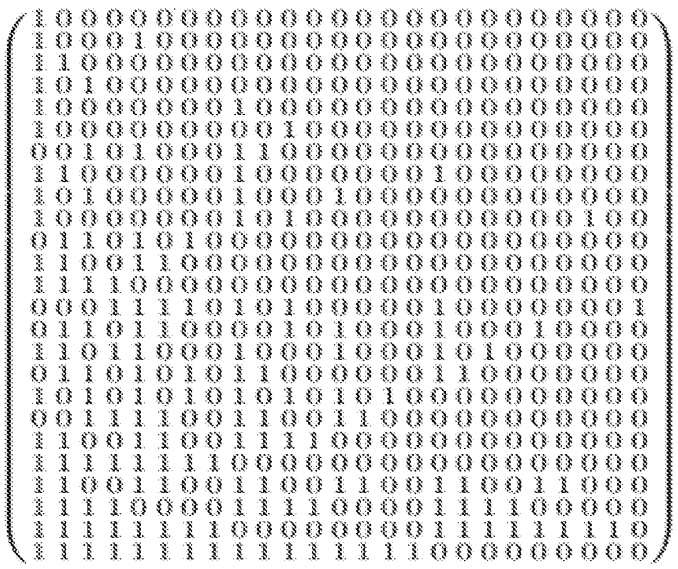
；或者，

$$K_{24}'' = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&1&1&0&1&0&0&0&1&0&0&0&1&0&0&0\\
0&1&1&1&0&0&1&0&1&0&0&0&1&0&0&0&1&0&1&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&1&1&0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0\\
1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0\\
1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0
\end{pmatrix}$$

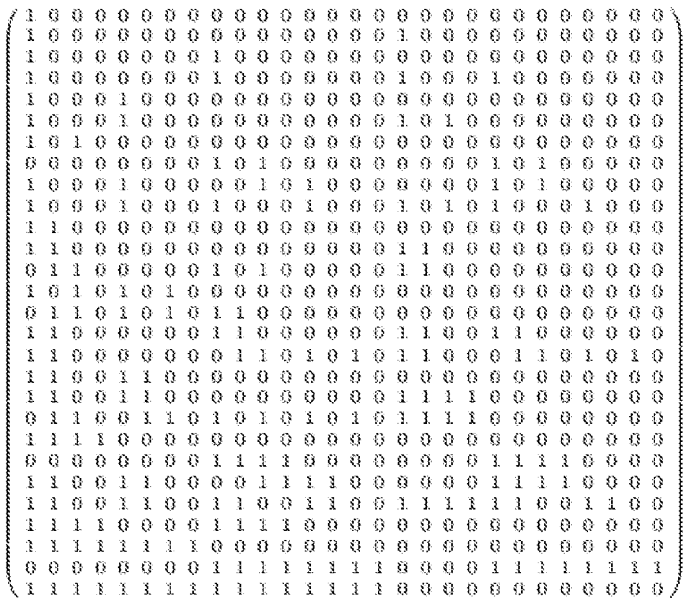
；或者，

$$K_{25} = \begin{pmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&1&0&0&0&0&1&0&0&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0&0&0&0&0&0&0\\
1&0&0&0&1&1&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&1&1&0&1&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0\\
0&0&1&0&1&1&0&1&1&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
1&1&1&1&1&1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&0&1&0&1&1&0&1&0&1&1&0&0&0&0&1&0&0&0&0\\
1&0&1&1&0&1&1&1&1&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0\\
1&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&0&0&1&0&1&0&1&0\\
1&1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0&0&1&0&0&1&1&0&0\\
1&1&1&1&1&1&0&0&0&1&1&1&1&0&0&0&0&1&1&1&0&0&0&0\\
1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1\\
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&0&0&0&0&0&0&0
\end{pmatrix}$$

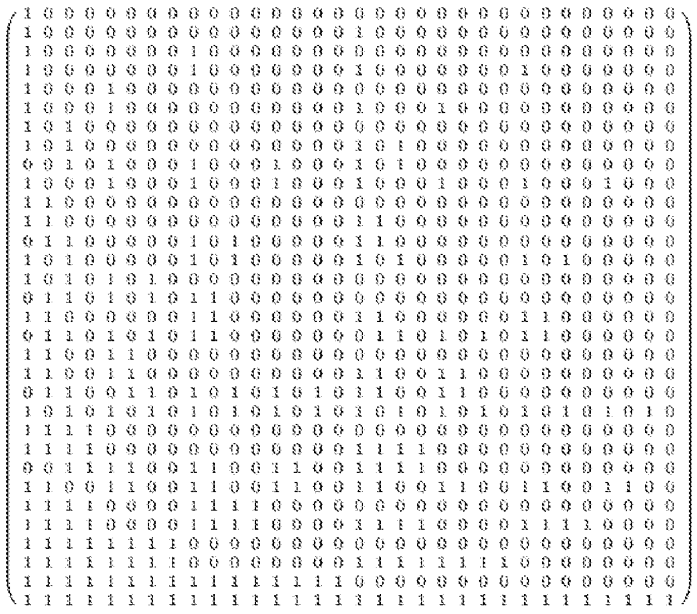
；或者，

$$K'_{25}=$$ 

；或者，

$$K_{28}=$$ 

；或者，

K_{32} = 

。

22. 一种译码装置，其特征在于，包括：收发单元和处理单元，

所述收发单元，用于获取待译码序列；

所述处理单元，用于基于多个网格，对所述待译码序列进行译码，基于所述多个网格中的第二网格译码时，复用所述多个网格中的第一网格译码的中间结果，

其中，所述第一网格用于对所述待译码序列译码的第 t 阶段，所述第二网格用于对所述待译码序列译码的第 t+i 阶段，t 为大于 0 或等于 0 的整数，i 为大于 1 或等于 1 的整数。

23. 根据权利要求 22 所述的译码装置，其特征在于，所述多个网格中的每个网格被分割为多个子网格。

24. 根据权利要求 23 所述的译码装置，其特征在于，所述多个子网格的分割点的位置与译码复杂度相关。

25. 根据权利要求 22 至 24 中任一项所述的译码装置，其特征在于，所述处理单元，还用于：

根据所述第一网格和所述第二网格的关联度，确定基于所述第二网格译码时是否复用所述第一网格译码的中间结果。

26. 根据权利要求 22 至 25 中任一项所述的译码装置，其特征在于，所述处理单元，还用于：

根据以下信息，判断基于所述第二网格译码时是否复用所述第一网格译码的中间结果：

所述译码的第 t+i 阶段通过对线性码 C 进行缩短操作得到的码与所述译码的第 t 阶段通过对线性码 C 进行缩短操作得到的码；和/或，

所述译码的第 t+i 阶段通过对线性码 C 进行打孔操作得到的码与所述译码的第 t 阶段通过对线性码 C 进行打孔操作得到的码。

27. 根据权利要求 26 所述的译码装置，其特征在于，

如果 $s_{x,y}(\mathcal{C}^{(t+i)})=s_{x,y}(\mathcal{C}^{(t)})$，$p_{x,y}(\mathcal{C}^{(t+i)})\subset p_{x,y}(\mathcal{C}^{(t)})$，那么基于所述第二网格译码时，复用所述第一网格译码的中间结果；或者

如果 $s_{x,y}(\mathcal{C}^{(t+i)})\subset s_{x,y}(\mathcal{C}^{(t)})$，那么基于所述第二网格译码时，复用所述第一网格译码的中间结果；

其中，$s_{x,y}(\mathcal{C}^{(t+i)})$表示译码的第 t+i 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行缩短操作得到的码，$s_{x,y}(\mathcal{C}^{(t)})$表示译码的第 t 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行缩短操作得到的码，$p_{x,y}(\mathcal{C}^{(t+i)})$表示译码的第 t+i 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行打孔操作得到的码，$p_{x,y}(\mathcal{C}^{(t)})$表示译码的第 t 阶段通过对线性码 C 在除开 $x\le z<y$ 的位置进行打孔操作得到的码。

28. 根据权利要求 22 至 27 中任一项所述的译码装置，其特征在于，

所述基于所述第二网格译码时，复用所述第一网格译码的中间结果，包括：

在所述译码的第 t+i 阶段基于所述第二网格处理时，直接将基于所述第一网格译码的中间结果，作为所述第二网格译码的中间结果；或者，

在所述译码的第 t+i 阶段基于所述第二网格处理时，基于所述第一网格处理过程中的最大化操作的结果进行处理。

29. 一种芯片，其特征在于，包括逻辑电路和通信接口，所述通信接口用于接收待处理的数据和/或信息，并将所述待处理的数据和/或信息传输至所述逻辑电路，所述逻辑电路用于执行如权利要求 1-7 中任一项所述的编码的处理，以及，所述通信接口还用于输出编码后的极化码字。

30. 一种芯片，其特征在于，包括逻辑电路和通信接口，所述通信接口用于接收待处理的数据和/或信息，并将所述待处理的数据和/或信息传输至所述逻辑电路，所述逻辑电路用于执行如权利要求 8-14 中任一项所述的译码的处理，以及，所述通信接口还用于输出译码结果。

31. 一种编码装置，其特征在于，包括至少一个处理器，所述至少一个处理器与至少一个存储器耦合，所述至少一个处理器用于执行所述至少一个存储器中存储的计算机程序或指令，以使所述通信装置执行如权利要求 1-7 中任一项所述的编码方法。

32. 根据权利要求 31 所述的编码装置，其特征在于，所述至少一个处理器与所述至少一个存储器集成在一起。

33. 一种译码装置，其特征在于，包括至少一个处理器，所述至少一个处理器与至少一个存储器耦合，所述至少一个处理器用于执行所述至少一个存储器中存储的计算机程序或指令，以使得所述通信装置执行如权利要求 8-14 中任一项所述的译码方法。

34. 根据权利要求 33 所述的译码装置，其特征在于，所述至少一个处理器与所述至少一个存储器集成在一起。

35. 一种计算机可读存储介质，其特征在于，所述计算机可读存储介质中存储有计算机指令，当计算机指令在计算机上运行时，如权利要求 1-7 中任一项所述的编码方法被实现。

36. 一种计算机可读存储介质，其特征在于，所述计算机可读存储介质中存储有计算

机指令，当计算机指令在计算机上运行时，如权利要求 8-14 中任一项所述的译码方法被实现。