2018-2019春季
信息隐藏课程
第11讲 校验子格编码

中国科学院信息工程研究所
INSITUTE OF INFORMATION ENGNEERING CAS

SKLOIS
信息安全国家重点实验室

赵险峰

中国科学院信息工程研究所
信息安全国家重点实验室

2017年12月

纲要

1. 最优嵌入问题
2. STC基本思想
3. STC算法
4. 文献阅读推荐
5. 作业

1-1 嵌入失真(代价)

- 载体样本为 $\boldsymbol{x}=(x_1,\cdots x_n)\in \mathrm{X}\triangleq \mathrm{I}^n$，典型地，$\mathrm{I}=\{0,1,\cdots,255\}$；$\boldsymbol{y}=(y_1,\cdots y_n)\in \mathrm{Y}\subset \mathrm{X}$ 是相应的隐写样本，$\pi(\boldsymbol{y})\triangleq P(\boldsymbol{y}|\boldsymbol{x})$ 表示其分布，可认为对应不同的修改方法；可以记 $\mathrm{Y}=\mathrm{I}_1\times \mathrm{I}_2\times\cdots\times \mathrm{I}_n$，$\mathrm{I}_i\subset \mathrm{I}$
- 例如，对二元嵌入LSBR有 $I_i=\{x_i,\bar{x}_i\}$，$\bar{x}_i$ 表示修改LSB；对三元嵌入LSBM有 $\mathrm{I}_i=\{x_i-1,x_i,x_i+1\}$，$x_i$是样点值
- 对应特定 $\boldsymbol{x}$，不同的 $\boldsymbol{y}$ 代表不同的嵌入方法（假定其分布为 $\pi(\boldsymbol{y})$），因此可以定义 $D(\boldsymbol{y})\triangleq D(\boldsymbol{x},\boldsymbol{y})$ 为该修改方式下的总体影响，可称为总失真（Distortion）或者代价（Cost）函数
- 隐写的信息传输量 $H(\pi)=-\sum_{\boldsymbol{y}\in\mathrm{Y}}\pi(\boldsymbol{y})\log_2\pi(\boldsymbol{y})$
- 平均总失真为 $E_\pi(D)=\sum_{\boldsymbol{y}\in\mathrm{Y}}D(\boldsymbol{y})\pi(\boldsymbol{y})$

1-2 限负载发送（PLS）的最优嵌入问题

- 在隐写中，一般需要发送给定数量的消息，设 m 表示需隐藏的消息量，此时最优嵌入问题归结为，通过设计隐写算法，寻求解决最小嵌入失真的问题——限负载发送（Payload Limited Sender，PLS）问题

$$\min_{\pi(\boldsymbol{y})} E_\pi(D) = \sum_{\boldsymbol{y}\in \mathrm{Y}} D(\boldsymbol{y})\pi(\boldsymbol{y})$$

$$\text{s.t.} \quad H(\pi) = -\sum_{\boldsymbol{y}\in \mathrm{Y}} \pi(\boldsymbol{y}) \log_2 \pi(\boldsymbol{y}) = m, \qquad \sum_{\boldsymbol{y}\in \mathrm{Y}} \pi(\boldsymbol{y}) = 1$$

- 即在有传输一定消息量m的能力下，使得平均总代价最小

SKLOIS
信息安全国家重点实验室

1-3 加性模型

- 在具体的分析与计算中，往往需要给出$D(\boldsymbol{y})$的具体形式。但是，由于每个样点的嵌入影响相互干扰，精确计算 $D(\boldsymbol{y})$ 非常困难。设 ρ_i 表示仅仅 x_i被修改为 y_i 引起的失真，可以表示为单点失真函数：

$$\rho_i \triangleq \rho_i(y_i) \triangleq \rho_i(\boldsymbol{x}, y_i) \triangleq \rho_i(\boldsymbol{x}, \boldsymbol{x}_{\sim i}y_i)$$

- 其中，$\boldsymbol{x}_{\sim i}y_i$表示 $\boldsymbol{x}$中只有x_i被修改为y_i。简单的做法是在以下定义的模型下估计总失真（称该模型为嵌入影响的加性模型）：

$$D(\boldsymbol{y}) = \sum_{i=1}^{n} \rho_i(\boldsymbol{x}, \boldsymbol{x}_{\sim i}y_i)$$

- 加性模型假设各个位置上的隐写修改之间不相互影响，距离现实有一定偏差，但加性模型下对失真的处理比较简单，在自适应隐写中得到了广泛的应用，也取得了很好的效果

1-4 加性模型下的最优嵌入理论结论

在加性模型下，以上优化问题理论上（它不关心嵌入什么信息和怎么提取，只关心信息量和失真）可解（λ 为拉格朗日乘数，可基于约束求得）

- 最优二元嵌入有

$$\pi(\boldsymbol{y}) = \prod_{i=1}^{n} \frac{e^{-\lambda\rho_i(y_i)}}{e^{-\lambda\rho_i(y_i=x_i)} + e^{-\lambda\rho_i(y_i=\bar{x}_i)}} = \prod_{i=1}^{n} \pi_i(y_i)$$

- 最优三元嵌入有

$$\pi(\boldsymbol{y}) = \prod_{i=1}^{n} \frac{e^{-\lambda\rho_i(y_i)}}{e^{-\lambda\rho_i(y_i=x_i+1)} + e^{-\lambda\rho_i(y_i=x_i)} + e^{-\lambda\rho_i(y_i=x_i-1)}} = \prod_{i=1}^{n} \pi_i(y_i)$$

- 以上分布一般用于模拟给定 ρ_i 下加性模型下的最优嵌入效果，用于与实际算法进行比较，表征实际算法距离最优嵌入的差距
- 实际算法用的 ρ_i 与以上分析相同，但一般用STC编码实现最优

2-1 STC之前的相关技术在优化上的局限

- 矩阵编码仅仅减少修改次数，在载体分组中不参考载体样点的特性；分组优化，未进行全局优化
- MME在载体分组中仅参考了隐写幅度特性，但是受到的代数制约较大；分组优化，未进行全局优化
- 湿纸编码实现了依据原始载体局部特性的动态位置嵌入，并且这种动态性不影响消息的正常接收，优化范围大。但是，湿纸编码较难构造，尤其是，可修改位置选择缺乏优化控制
  - 在湿纸编码中，已经出现嵌入失真（Embedding Distortion）的概念，但失真是用“干”或“湿”仅仅2个级别来简单衡量的，在能够使提取方程满足的多种位置组合中，在干点范围内也缺乏优化选择；全局优化计算困难，实际只能分组优化

2-2 线性提取方程的多解利用

- 为在现实算法中实现最小失真嵌入，直观的想法是利用线性提取方程的多解性质，在解空间中选择总体失真最少的嵌入方法，并避免接收者需要知道动态变化的嵌入位置
- 典型地，设$H_{m\times n}$为二元域上的校验矩阵，原始载体为 $\boldsymbol{x}$ ，相应的藏密载体为 $\boldsymbol{y}$，$P(\boldsymbol{y})$ 表示载体的LSB序列，$\boldsymbol{m}$ 表示嵌入的消息，满足提取方程的解集（解空间）可以描述为

$$C(\boldsymbol{m}) = \{\boldsymbol{z} \in \{0,1\}^n | H\boldsymbol{z} = \boldsymbol{m}\}$$

- 最小代价嵌入直观上是求解以下问题

$$\mathrm{Emb}(\boldsymbol{x}, \boldsymbol{m}) = \arg \min_{P(\boldsymbol{y}) \in C(\boldsymbol{m})} D(\boldsymbol{x}, \boldsymbol{y}), \quad \mathrm{Ext}(\boldsymbol{y}) = H \times P(\boldsymbol{y}) = \boldsymbol{m}$$

- 其中，Emb 与 Ext 分别表示嵌入与提取算法。但是，如果基于类似前面的湿纸编码方法，只能采取逐次不同方式嵌入，嵌完后比较每次代价的搜索方法（在满足提取方程的解空间中逐一比较），但在计算上是困难的

2-3 STC基本思想与带状校验矩阵

- STC采用**带状校验矩阵**。**提取方程求解可逐块进行**，有利于进行提取方程的构造（即逐步完成嵌入消息与优化）。STC校验矩阵取小矩阵

$$\hat{H}_{h\times w}=\begin{pmatrix}\hat{h}_{1,1} & \cdots & \hat{h}_{1,w}\\ \vdots & \ddots & \vdots\\ \hat{h}_{h,1} & \cdots & \hat{h}_{h,w}\end{pmatrix}$$

- 按照矩阵对角线方向不断重复，每次的重复方法是，靠上一个小矩阵的右侧向下一行摆放，最后形成近似带状矩阵（最后几次复制不完整，后面将会发现这不影响嵌入和提取）求解$H\times P(\boldsymbol{y})=\boldsymbol{m}$

$$H_{m\times n}$$

$$=\begin{pmatrix}
\hat{h}_{1,1} & \cdots & \hat{h}_{1,w} & 0 & 0 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0\\
\vdots & \ddots & \vdots & \hat{h}_{1,1} & \cdots & \hat{h}_{1,w} & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0\\
\hat{h}_{h,1} & \cdots & \hat{h}_{h,w} & \vdots & \ddots & \vdots & \cdots & 0 & \cdots & \cdots & \cdots & \cdots & 0\\
0 & 0 & 0 & \hat{h}_{h,1} & \cdots & \hat{h}_{h,w} & \cdots & 0 & \cdots & \cdots & \cdots & \cdots & 0\\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \hat{h}_{1,1} & \cdots & \hat{h}_{1,w} & 0 & \cdots & 0\\
0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \hat{h}_{2,1} & \ddots & \hat{h}_{2,w} & \hat{h}_{1,1} & \cdots & \hat{h}_{1,w}
\end{pmatrix}$$

2-4 STC提取方程的逐步求解（嵌入）

- 如果 H 为校验矩阵，$\boldsymbol{m}$ 就是校验子，因此STC编码可看作是通过逐步修改$\boldsymbol{x}$ 或者通过逐步构造 $\boldsymbol{y}$ 使得 $H\boldsymbol{y}=\boldsymbol{m}$ 并使总体失真代价的和最小；为方便，下面也假设$\boldsymbol{x}$ 和$\boldsymbol{y}$ 为LSB序列
- 在逐步构造提取方程$H\boldsymbol{y}=\boldsymbol{m}$时，STC每次通过新加入 w 个 $\boldsymbol{y}$ 中元素 $\left(y_{(i-1)w+1},\cdots,y_{iw}\right)$ 确定一个消息比特 m_i 。依次构造以下等式（可以认为，每行对藏密载体 $\boldsymbol{y}$ 的运算，仅仅约束一个比特消息的嵌入方式，显然**每步都是多解的，形成多条发展路径，可按积累的失真不断选择求解路径，提前消除没有前途的路径**）：

$$
\begin{gathered}
\left(h_{1,1},\cdots,h_{1,w}\right)\left(y_1,\cdots,y_w\right)^{\mathrm{T}}=\left(h_{1,1},\cdots,h_{1,w}\right)\left(x_1+e_1,\cdots,x_w+e_w\right)^{\mathrm{T}}=m_1\\
\left(h_{2,1},\cdots,h_{2,2w}\right)\left(y_1,\cdots,y_{2w}\right)^{\mathrm{T}}=\left(h_{2,1},\cdots,h_{2,2w}\right)\left(x_1+e_1,\cdots,x_{2w}+e_{2w}\right)^{\mathrm{T}}=m_2\\
\vdots\\
\left(h_{i,(i-h)w+1},\cdots,h_{i,iw}\right)\left(y_{(i-h)w+1},\cdots,y_{iw}\right)^{\mathrm{T}}=m_i\\
\vdots\\
\left(h_{m,(m-h)w+1},\cdots,h_{m,mw}\right)\left(y_{(m-h)w+1},\cdots,y_{mw}\right)^{\mathrm{T}}=m_m
\end{gathered}
$$

信息安全国家重点实验室

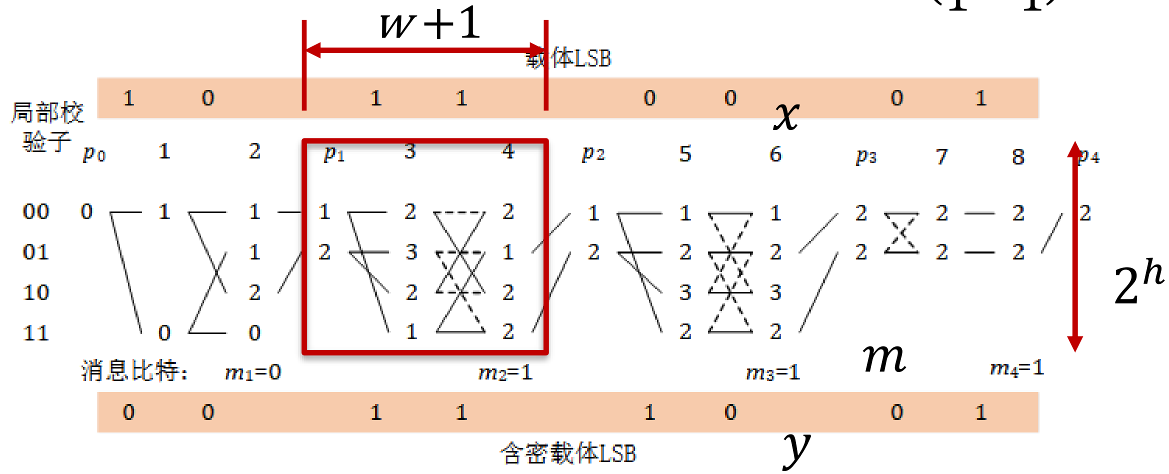
2-5 STC的一些性质

- 若 α 为负载率，则 $\alpha = 1/w$，w为小矩阵的列数
- 由于 H 每列最多有 h 个非零元素，y_i 仅影响 h 个消息比特
- 校验矩阵 H 的列数 $n = m \cdot w = m/\alpha$
- 校验矩阵 H 的尺寸可以表达为 $m \times n = [\alpha n] \times n = m \times (m \cdot w)$
- 由于 α 为小数而 w 为整数，因此按照以上方式排列子矩阵不能获得任意的负载率。在分别采用宽度 w 与 $w+1$ 两个子矩阵时，由于 $1/(w+1) < \alpha < 1/w$，可以通过混合排列子矩阵逼近 α
- 在描述STC中，我们假设原始载体 $\boldsymbol{x}$ 与相应的藏密载体 $\boldsymbol{y}$ 已经被置乱，这是隐写编码的基本原则：它使得STC运行平稳，每次面临基本类似的数据情况进行优化处理，也有利于嵌入短消息，并保护嵌入数据与应用协议的安全
- **根据前面的“路径”描述，STC的求解与路径优化过程可以通过格图表达**

3-1 STC算法（二元编码格图结构）

- 在格图中，任何一个仍在发展的路径代表仍可能满足 $\boldsymbol{Hy}=\boldsymbol{m}$ 并且总代价和可能将是最小的。这里给出二元编码格图的组成：
  - **子块。**格图包含 m 个子块，依次对应 $H_{m\times n}$中对角线方向相应小矩阵提取方程求解；第 i 个子块图通过 w 层的发展确保了 m_i 可通过 $\boldsymbol{Hy}=\boldsymbol{m}$ 提取，y_i 决定了是否将 H 的第 i 列加到当前校验子（即当前嵌入的消息）中；每个子块有 2^h 行与 $w+1$列，子块中两个相邻列上的节点由连线从左向右连接
  - **节点。**每个子块节点表示一种阶段情况，连线通过的节点为可达节点，对应一个状态（State）；在2个子块间，按照可满足 $\boldsymbol{Hy}=\boldsymbol{m}$ 的原则选择可达节点进入下一子块的第一列，当前状态低位删除高位补0，因此子块首列节点表示开始或者在上轮基础上开始新的子块计算
  - **失真。**节点上标记的数字表示编码代价，即至目前，前面修改积累的失真总和（右图仅以修改次数和为失真和）

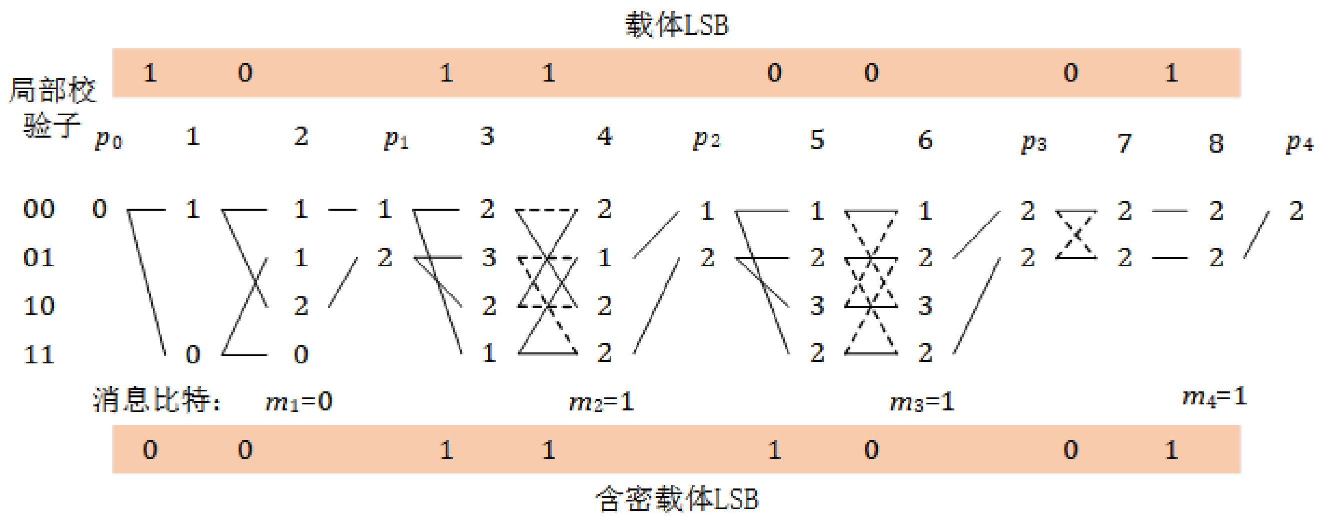
$$\hat{H}_{2\times 2}=\begin{pmatrix}1 & 0\\ 1 & 1\end{pmatrix}$$



3-2 STC算法（二元编码格图结构）（续）

- **位置（列编号）**。除了每个子块的第一列，整个格图每列依次编号为 $\{1,2,\cdots,n\}$，表示当前考虑修改的载体位置；第 i 个子块的首列用 p_i 表示
- **局部校验子（消息）状态（也即行编号或之前的State）**。每行依次编号为 $\{0,1,\cdots,2^h-1\}$，一般用二进制表示当前得到的局部校验子，注意局部校验子值的二进制是低位靠右，对应校验子向量的上部（逆时针旋转）
- **连线**。子块内每个可达节点的2个分支代表不同的嵌入方法，即令 y_i为1或0，连线的方向并不表示特定的修改方式；子块间连线表示有效状态节点进入下一子块，状态进入下一子块时当前状态低位删除高位补0

$$\hat{H}_{2\times 2}=\begin{pmatrix}1 & 0\\ 1 & 1\end{pmatrix}$$

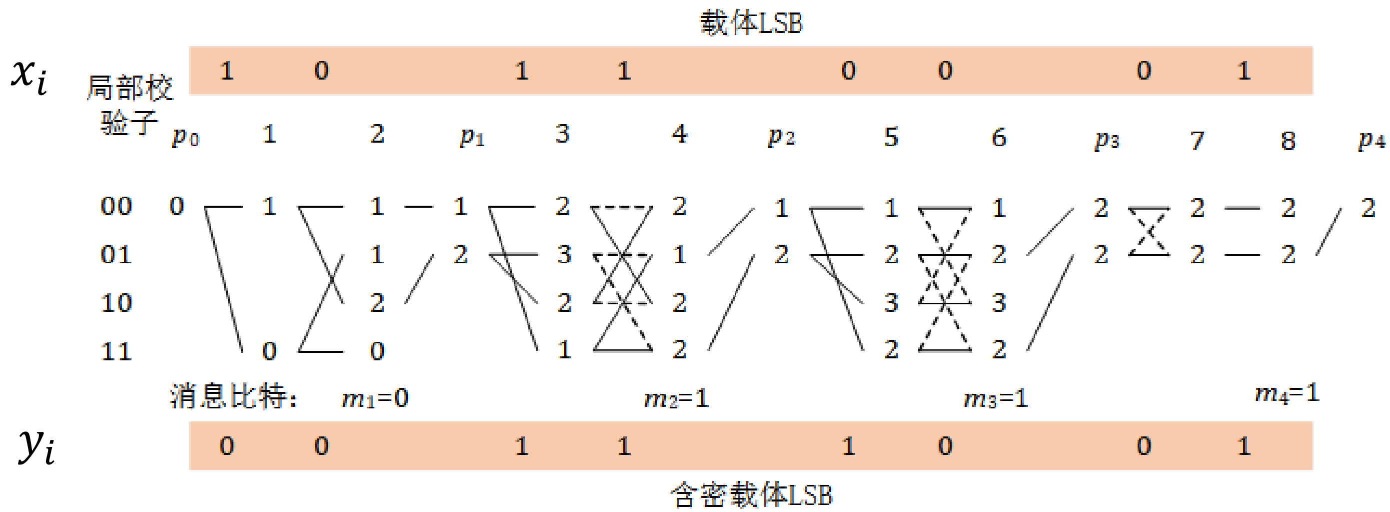


3-3 二元STC嵌入例子（问题描述）

- 在$\boldsymbol{x}$ = 10110001中嵌入$\boldsymbol{m}$ = 0111得到$\boldsymbol{y}$ = 00111001的过程，失真代价为修改次数
- $\hat{H}$ 与H 分别为：$\hat{H}_{2\times2}=\begin{pmatrix}1 & 0\\1 & 1\end{pmatrix}$，$H_{4\times8}=$
$$\begin{pmatrix}1&0&0&0&0&0&0&0\\1&1&1&0&0&0&0&0\\0&0&1&1&1&0&0&0\\0&0&0&0&1&1&1&0\end{pmatrix}$$
- 即通过修改$\boldsymbol{x}$为$\boldsymbol{y}$使得以下成立且总修改次数最少：

$$\begin{pmatrix}1&0&0&0&0&0&0&0\\1&1&1&0&0&0&0&0\\0&0&1&1&1&0&0&0\\0&0&0&0&1&1&1&0\end{pmatrix}\begin{pmatrix}y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\\y_8\end{pmatrix}=\begin{pmatrix}0\\1\\1\\1\end{pmatrix}$$

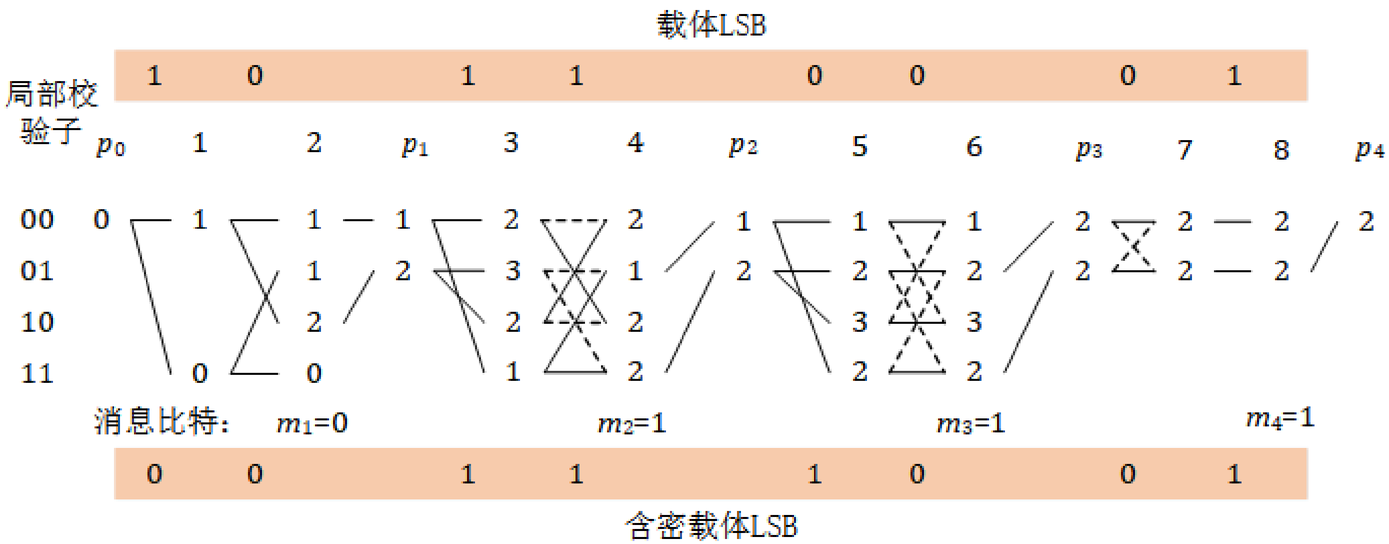
3-4 二元STC嵌入例子（第1子块）



$$\hat{H}_{2\times 2} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

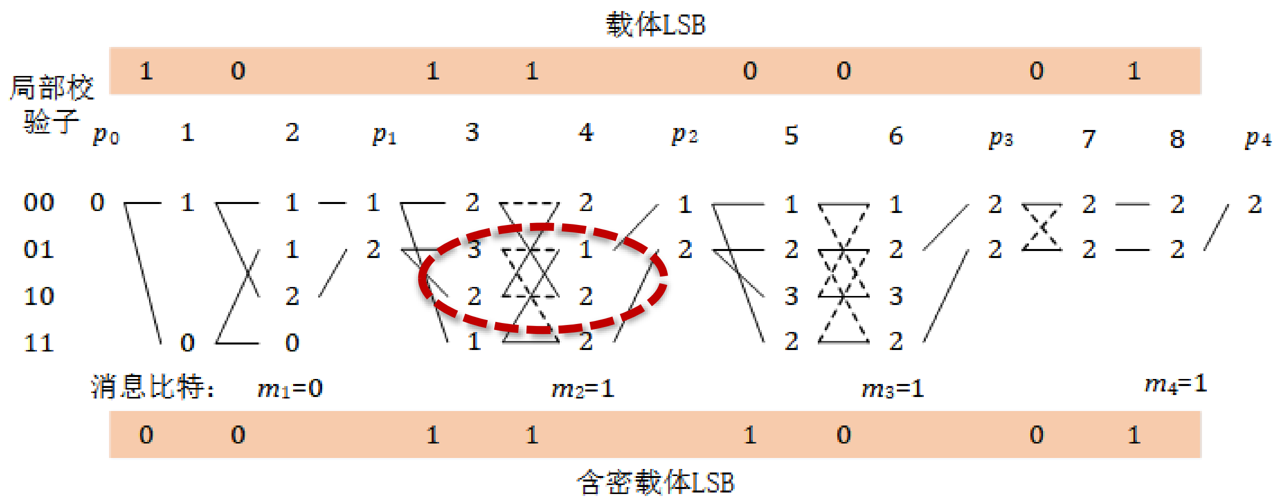
- 开始，初态为00；对应 $y_1 = 1$ （无修改）可认为要在00初态上加第一列的11，进入$00 + 11 = 11$态；对应 $y_1 = 0$ （修改）则不加入，保留在$00 + 00 = 00$状态，但代价为1次修改
- 从第2层的11态开始，若保持$y_2 = 0$ （无修改），则留在$11 + 00 = 11$态，若改$y_2 = 1$ （修改），则进入$11 + 10 = 01$态，其中，10为 H 的第2列元素；从第二层的00态开始，若保持$y_2 = 0$ （无修改），则留在$00 + 00 = 00$态，若$y_2 = 1$ （修改），则进入$00 + 10 = 10$态

3-5 二元STC嵌入例子（第1子块进入第2子块）



- 由于要求$m_1 = 0$，删除第1个子块的结束状态01与11上的路径
- 由于经过以上处理后$m_1 = 0$已经确保，在第1个子块的结束状态00与10中删除此低位0，得到0与1，再在高位上补0，则进入第下一子块的局部校验子初态为00与01
- 显然，新加入的当前校验子状态有待下一个块的运算被更改

3-6 二元STC嵌入例子（第2至第4子块）



注意是大矩阵最后一行

$$\widehat{H}_{2\times 2}=\begin{pmatrix}1 & 0\\ 0 & 0\end{pmatrix}$$

- 第2、3子块的处理类似第1子块，因此省略（注意高代价路径的削减）
- 进入第4子块的局部校验子初态为00与01，代价和均为2
  - 第一层从00出发，若$y_7=0$（无修改），保留在00，若$y_7=1$（修改），进入00+01=01，但代价和为3；第一层从01出发，若$y_7=0$（无修改），保留在01，若$y_7=1$（修改），进入01+01=00，但代价和为3 (显然代价为3的都可丢弃)
  - 第二层从01出发，若$y_8=1$（无修改），则进入01+00=01，代价和仍为2，满足$m_4=1$的要求；其他代价和为3的路径删除；而从第二层从00出发，若$y_8=1$（无修改），则进入00+00=00，代价和仍为2，不满足$m_4=1$的要求

3-6 STC嵌入算法

- 输入得到 $\boldsymbol{x}, \boldsymbol{m}, H$；
- 对每个可嵌入样点计算代价 $\rho_i(\boldsymbol{x}, \boldsymbol{x}_{\sim i} y_i)$；对$N$元编码，每个样点的代价数量一般是$N$个（一般其中一个是0，即不修改情况）
- 将载体 $\boldsymbol{x}$ 分段，每段长度是小矩阵 $\hat{H}$ 的宽度；对每个分段进行以下3个步骤直至嵌入结束；
  - 在第一段的初始状态为全零，其他分段的初始状态是上一段遗留状态最低位删除并且最高位补0的结果
  - 考察当前状态通过修改或不修改 x_i 的变化可能，根据 x_i 的可能修改情况记录每条路径上的代价和，路径逐位置向后推进
  - 每个分段结束后，删除不可能满足提取方程的路径；对代价明显多的路径提前删除
- 在最后一个分段处理后，在满足提取方程的路径中，选择代价和最小的，依此路径回溯确定实际的修改方法，得到并输出 $\boldsymbol{y}$

5 文献阅读推荐

- [1] 教材第11章
- [2] T. Filler, J. Judas, J. Fridrich. Minimizing embedding impact in steganography using trellis-coded quantization. SPIE Electronic Imaging, Media Forensics and Security, SPIE Proceedings, Vol. 7541, Page 05-1-05-14. 2010.
- [3] T. Filler, J. Judas, J. Fridrich. Minimizing additive distortion in steganography using syndrome-trellis codes, IEEE Trans. Information Forensics and Security, 6(3): 920-935, 2011

6 作业

- 自选8个消息比特和一段LSB载体，用修改次数为代价，用STC将前者嵌入后者，给出全部STC参数配置以及嵌入过程（画出格图并给出每步说明，推荐用Visio制图），给出最后得到的LSB输出

谢谢！

中国科学院信息工程研究所
INSITUTE OF INFORMATION ENGNEERING CAS

SKLOIS
信息安全国家重点实验室