236608 - Coding and Algorithms for Memories
Winter Semester 2018/2019

Write-Once Memory (WOM) Codes

The memory elements, called cells, have two states: 0 and 1. At the beginning, all the cells are in the 0 state. A cell can change its state from 0 to 1. This operation is irreversible in the sense that a cell cannot change its state from 1 to 0. The memory-state vectors are all the binary vectors \(c = (c_1, c_2, \ldots, c_n)\) of length \(n\). For two memory-state vectors \(c, c' \in \{0, 1\}^n\), we denote by \(c \geq c'\), if and only if \(c_i \geq c'_i\) for all \(1 \leq i \leq n\).

**Definition 1.** An \([n; t; M_1, \ldots, M_t]t\)-write WOM-code \(C\) is a coding scheme which consists of \(n\) cells and \(t\) pairs of encoding and decoding maps, denoted by \(E_i\) and \(D_i\) for \(1 \leq i \leq t\). The \(t\)-write WOM-code \(C\) satisfies the following properties:

1. \(E_1 : \{1, \ldots, M_1\} \rightarrow \{0, 1\}^n\),
2. For \(2 \leq i \leq t\), \(E_i : \{1, \ldots, M_i\} \times \{0, 1\}^n \rightarrow \{0, 1\}^n\), such that, for all \((m, c) \in \{1, \ldots, M_i\} \times \{0, 1\}^n\), \(E_i(m, c) \geq c\).
3. For \(1 \leq i \leq t\), \(D_i : \{0, 1\}^n \rightarrow \{1, \ldots, M_i\}\), such that \(D_i(E_i(m)) = m\) for all \(m \in \{1, \ldots, M_i\}\), and for \(2 \leq i \leq t\), \(D_i(E_i(m, c)) = m\) for all \((m, c) \in \{1, \ldots, M_i\} \times \{0, 1\}^n\).

The **sum-rate** of a \(t\)-write WOM-code \(C\) is defined to be

\[
R_{\text{sum}}(C) = \frac{\sum_{i=1}^t \log_2 M_i}{n}.
\]

**Proposition 2** Knowing the write number does not affect the set of achievable rates.

**Proof:** Assume that there exists an \([n; t; M_1, \ldots, M_t]t\)-write WOM-code \(C\) where the write number is known. Assume also that the sum-rate of \(C\) is \(R_{\text{sum}}(C) = \frac{\sum_{i=1}^t \log_2 M_i}{n}\). It is possible to change this WOM-code to an \([Nn + t; t; M_1^N, \ldots, M_t^N]t\)-write WOM-code \(C'\) by having \(N\) blocks of the \(t\)-write WOM-code \(C\) and \(t\) more cells indicating the write number. Then, the sum-rate of \(C'\) is

\[
R_{\text{sum}}(C') = \frac{\sum_{i=1}^t \log_2 M_i^N}{Nn + t} = \frac{N \sum_{i=1}^t \log_2 M_i}{Nn + t} = \frac{Nn}{Nn + t} \cdot \frac{R_{\text{sum}}(C)}{1 + \frac{t}{Nn}}.
\]
Therefore, for $N$ large enough it is possible to achieve the sum-rate of the $t$-write WOM-code $C$. The capacity region of a binary $t$-write WOM-code is

$$C_t = \left\{ (R_1, \ldots, R_t) \mid R_1 \leq h(p_1), R_2 \leq (1 - p_1) h(p_2), \ldots, R_{t-1} \leq \left( \prod_{i=1}^{t-2} (1 - p_i) \right) h(p_{t-1}), R_t \leq \prod_{i=1}^{t-1} (1 - p_i), \right\}.$$

The sum-rate of the WOM-code is given by

$$R = \sum_{j=1}^{t} R_j = h(p_1) + \sum_{j=2}^{t-1} \left( \prod_{i=1}^{j-1} (1 - p_i) h(p_j) \right) + \prod_{i=1}^{t-1} (1 - p_i).$$

The sum-rate is maximized when

$$p_j = \frac{1}{2 + t - j},$$

for $1 \leq j \leq t - 1$, and the maximum sum-rate is $\log_2(t + 1)$. For example, for $t = 2$, the maximum sum-rate, $\log_2 3$, is achieved for $p_1 = 1/3$.

For fixed-rate WOM-codes all individual rates are the same, so we consider those points on the boundary of the capacity region $C_t$ satisfying $R_1 = \cdots = R_t$. The maximum sum-rate, denoted by $R^F_U(t)$, is calculated recursively as stated as follows.

**Theorem 3.** The values of $R^F_U(t)$ for $t \geq 1$ satisfy the following recursive formula:

$$R^F_U(1) = 1$$

$$R^F_U(t + 1) = (t + 1) \cdot \text{root} \left\{ h \left( \frac{z}{R^F_U(t)/t} \right) - z \right\},$$

where $\text{root}\{f(z)\}$ is the minimum positive value of $z$ such that $f(z) = 0$.

Using the recursion in the theorem, the following results are obtained for $R^F_U(t)$ in Table 1.

<table>
<thead>
<tr>
<th>$t$</th>
<th>$R^F_U(t)$</th>
<th>$t$</th>
<th>$R^F_U(t)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td>2.712</td>
</tr>
<tr>
<td>2</td>
<td>1.546</td>
<td>7</td>
<td>2.9001</td>
</tr>
<tr>
<td>3</td>
<td>1.9368</td>
<td>8</td>
<td>3.0664</td>
</tr>
<tr>
<td>4</td>
<td>2.2436</td>
<td>9</td>
<td>3.2157</td>
</tr>
<tr>
<td>5</td>
<td>2.4965</td>
<td>10</td>
<td>3.352</td>
</tr>
</tbody>
</table>
A WOM Code Example

The following is an example of a $[n, n/2-1; n/2, n/2-1, n/2-2, \ldots, 2]$ WOM code. The $n$ cells are partitioned into two groups of $n/2$ cells each. The encoding maps are given as follows.

1. First write: write a message $m_1 \in \{1, 2, \ldots, n/2\}$ with $n/2$ values by programming the $m_1$-th cell in the first group of $n/2$ cells.

2. On the $i$-write: let $c_1, c_2$ be the memory-state vector of the first, second group of cells, respectively. Before this write, it holds that $w_H(c_1) = i - 1, w_H(c_2) = i - 2$, and $c_1 \geq c_2$. Write a message $m_i \in \{1, 2, \ldots, n/2 - i + 1\}$ with $n/2 - i + 1$ values as follows. Copy the memory-state vector $c_1$ to the second group of cells. Among the $n/2$ non-programmed cells in the first group, program the $m_i$-th cell.

The decoding map receives the $n$ cells as two vectors $c_1, c_2$ each of size $n/2$. The write number $i$ is given by the weight of the vector $c_1$, i.e. $i = w_H(c_1)$. The decoded value $m_i \in \{1, \ldots, n/2 - i + 1\}$ is calculated as follows. It holds that $c_1 \geq c_2$ and $d_H(c_1, c_2) = 1$, that is, there exists exactly one cell which is programmed in $c_1$ and not in $c_2$. The value of $m_i$ is the the location of this cell among the non-programmed cells in $c_2$.

The sum-rate of this construction is

$$\sum_{i=2}^{n/2} \frac{\log i}{n} = \frac{\log((n/2)!)}{n} \approx \frac{n/2 \log(n/2)}{n} = \frac{\log(n) - 1}{2}.$$

The Coset Coding Scheme

Given an $r \times n$ binary matrix $H$ (usually this will be the parity-check matrix of some linear code), the coset coding scheme works as follows:

1. First write: assume the message is $s \in \{0, 1\}^r$. Find a binary vector $c \in \{0, 1\}^n$ of minimum Hamming weight such that $H \cdot c^T = s$ and program the cells with the vector $c$.

2. On every consecutive write: if the memory-state vector is $c \in \{0, 1\}^n$ and the new message is $s \in \{0, 1\}^r$, find a binary vector $c' \in \{0, 1\}^n$ of minimum Hamming weight such that $H \cdot c'^T = s$ and $c' \geq c$ and program the cells with the vector $c'$.

The decoding on each write is given by $s = H \cdot c^T$, where $c$ is the cell-state vector.

Let $H_m$ be the $m \times (2^m - 1)$ parity check matrix of a Hamming code of length $2^m - 1$, dimension $2^m - m - 1$, and minimum distance 3.

Problem 1. Prove that the matrix $H_3$ provides a $[7, 3; 8, 8, 8]$ WOM code using the coset coding scheme.
Solution: On the first write at most one cell is programmed and on the second write at most two more cells are programmed. The third write succeeds since every four columns left provide a matrix of full rank. This is true since every 4 non-zero vectors (of any length) cannot belong to a subspace of dimension 2 (which has exactly 4 vectors, one of them is the zero vector).

Problem 2. Prove that the matrix $H_m$ for $m \geq 3$ provides a $[2^m - 1, 2^{m-2} + 1; 2^m, 2^{m-2}, \ldots, 2^m]$ WOM code using the coset coding scheme.

Solution: Every non-zero length-$m$ vector appears in the matrix and can also be represented as the sum of two columns in $2^{m-1} - 1$ different ways. On the first write at most one cell is programmed. On each consecutive write it is possible to write every syndrome $s$ of $m$ bits using at most two cells. After $i \leq 2^{m-2}$ writes the number of programmed cells is at most $1 + 2(i - 1) \leq 1 + 2 \cdot (2^{m-2} - 1) = 2^{m-1} - 1$. Given the syndrome of the following write, it can still be written if its corresponding cell was not programmed or by one of its $2^{m-1} - 1$ options to be written as a sum of two columns from the matrix.

$\epsilon$-Error Two-Write WOM Codes

An $\epsilon$-error two-write WOM code is a WOM code which its second write does not succeed in the worst case, but only with high probability.

Let $p \in (0, 1)$ and let $H$ be an $r \times n$ binary matrix for $r = \lceil pn \rceil$, which is chosen uniformly in random. On the first write, one of $\sum_{i=0}^{n-r} \binom{n}{i}$ messages is written such that at most $n - r$ cells are programmed from zero to one. Let $c_1$ be the memory-state vector after the first write. On the second write, the user seeks to write a message $s$ of $r$ bits by choosing a vector $c_2$ such that $H \cdot c_2^T = s$ and $c_2 \geq c_1$. Assume the messages on the first and second write are chosen uniformly in random and define by $P(n, p)$ to be the probability that the second write succeeds.

Problem 3. Prove that for every $p \in (0, 1)$, $\lim_{n \to \infty} P(n, p) \geq 0.287$

Solution: After the first write, at most $n - r$ cells are programmed, so for the second write it is possible to write at most $r$ information bits, and this will be possible if the sub-matrix corresponding to the (at least) $r$ columns of the available cells will be of full rank. This probability is at least the probability that an $r \times r$ matrix is of full rank. According to Problem 3 in HW 1, this probability is $\prod_{i=0}^{r-1} \frac{2^r - 2^i}{2^r} = \prod_{i=0}^{r-1} \left(1 - 2^{i-r}\right) = \prod_{i=1}^{r} \left(1 - 2^{-i}\right)$. It holds that $\lim_{r \to \infty} \prod_{i=1}^{r} \left(1 - 2^{-i}\right) \approx 0.2887$, and the last part can be calculated using Wolfram Alpha.
Two-Write Non-Binary WOM Codes

In this problem, we construct two-write WOM codes for \( q = 8 \). We let the length of the code be \( N = n + 3 \).

**First write:** we get a word \( w_1 \in \{0, 1, 2, 3, 4\}^n \). Let \( \alpha \leq \beta \in \{0, \ldots, 4\} \) be the two most common values appearing in \( w_1 \). Define a new word \( w_1' \) as follows. Whenever \( w_{1,i} = \alpha \) we set \( w_{1,i}' = 0 \) and when \( w_{1,i} = \beta \) we write \( w_{1,i}' = \alpha \). In the same way we replace the value 1 with \( \beta \). The rest of the coordinates are unchanged. We now write \( w_1' \) to the first \( n \) memory cells as is. In order to recover \( w_1 \) we write \( \alpha \) in the \((n+1)\)-th cell and \( \beta \) in the \((n+2)\)-th cell. It is clear how we can recover \( w_1 \) by reading the memory cells.

**Second write:** Let \( w_2 \in \{0, 5, 6, 7\}^n \) and \( w_3 \in \{1, 2, 3\}^{n/10} \) be the input words to be written. Let \( I' \) be the set of “small” coordinates of \( w_1' \), i.e., \( I' = \{ i \mid w_{1,i}' \leq 1 \} \). Note that \( |I'| \geq \frac{2}{5} \cdot n \). Let \( \gamma \in \{0, 5, 6, 7\} \) be the most common symbol appearing in \((w_2)_I\). That is, we only consider the coordinates \( I' \) and among them we check which value was the most popular in \( w_2 \). (We break ties arbitrarily.) Let \( I'' = \{ i \mid i \in I' \text{ and } w_{2,i} = \gamma \} \), and note that

\[
|I''| \geq \frac{1}{4} \cdot |I'| \geq \frac{n}{10}.
\]

Let \( I \) be the first \( n/10 \) coordinates of \( I'' \). We now write \( \gamma \) on the \( N \)-th memory cell and define a new word \( w_2' \) as follows. Whenever \( w_2 \) had zero we change it to \( \gamma \) and whenever it had \( \gamma \) we change it to zero. We now write the rest of the memory cells as follows.

1. If \( w_{2,i}'' \neq 0 \) then we write its value to the \( i \)-th cell.
2. If \( w_{2,i}'' = 0 \) and \( i \notin I \) then we write the value 4 in the \( i \)-th cell.
3. If \( w_{2,i}'' = 0 \) and \( i \in I \) and it is the \( j \)-th element in \( I \) (according to the order \( 0 < 1 < 2 < \cdots < n/10 \)) then we write the value \( w_{3,j} \) in the \( i \)-th cell (alternatively, we can think of \( w_3 \) as \( w_3 \in \{1, 2, 3\}^{|I|} \) and write its \( j \)-th coordinate). Notice that for \( i \in I \), \( w_{1,i}' \leq 1 \leq w_{3,i} \) and so this is a “legal” write.

In order to decode the word \( w_2 \), we read all cell levels, while treating levels 1, 2, 3, and 4 to be zero and converting the \( \gamma \) symbol. The word \( w_3 \) is decoded according to the first \( n/10 \) cells of level less than 4.

The sum-rate of the construction is calculated as follows. In the first write we wrote an arbitrary word in \( \{0,1,2,3,4\}^n \) so we store \( \log(5^n) = \log(5) \cdot n \) bits of information. In the second write we stored a pair \((w_2, w_3) \in \{0,5,6,7\}^n \times \{1,2,3\}^{n/10} \) so overall we store in the second write \( 2n + \left( \frac{\log(3)}{10} \right) \cdot n \) bits. Thus, the sum-rate, given by

\[
(\log 5 + 2 + \frac{\log 3}{10}) \cdot \left( \frac{n}{N} \right) = (\log 5 + 2 + \frac{\log 3}{10}) \cdot \left( 1 - \frac{3}{n + 3} \right),
\]

approaches \( 4.4804 \) for \( n \) large enough. Note that the upper bound on the sum-rate in this case is \( \log(\binom{10}{2}) = \log 36 \approx 5.1699 \).