Tight Upper Bounds on the Redundancy of Optimal Binary AIFV codes

11 July 2016

Weihua Hu, Hirosuke Yamamoto, Junya Honda
The University of Tokyo
Outline

1. Background
Binary Almost instantaneous FV (AIFV) codes

2. Main result:
Tight upper bounds on the redundancy of binary AIFV codes (worst-case redundancy)

3. Comparison with Huffman codes

4. Idea & outline of proofs

5. Conclusion
Binary AIFV codes

Fixed-to-Variable length (FV) codes

Source symbol → Codeword

Uniquely Decodable

Instantaneous (Huffman)

Class of FV Codes
Binary AIFV codes

**Fixed-to-Variable length (FV) codes**

- Source symbol
- Codeword

Uniquely Decodable

- *Almost Instantaneous* [Yamamoto+ 2015]
- *Instantaneous* (Huffman)

Class of FV Codes
Almost Instantaneous (AI) FV codes

Generalization of instantaneous binary FV codes
[Yamamoto, Tsuchihashi, Honda, 2015]

<table>
<thead>
<tr>
<th></th>
<th>Instantaneous</th>
<th>Almost Instantaneous</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.(Code Trees)</td>
<td>One</td>
<td>Two</td>
</tr>
<tr>
<td>Source Symbols</td>
<td>Leaves</td>
<td>Leaves + incomplete node (master node)</td>
</tr>
<tr>
<td>Decoding Delay</td>
<td>None</td>
<td>At most 2 bits.</td>
</tr>
</tbody>
</table>

Given a source sequence \(x_1 x_2 x_3 \cdots\), an encoding procedure of a binary AIFV code goes as follows.

Procedure 1 (Encoding of a binary AIFV code).
1) Use \(T_0\) to encode the initial source symbol \(x_1\).
2) When \(x_i\) is encoded by a leaf (resp. a master node), then use \(T_0\) (resp. \(T_1\)) to encode the next symbol \(x_{i+1}\).

Using a binary AIFV code of Fig. 1, a source sequence ‘acdbaca’ is encoded to ‘0.11.1100.10.0.11.01’, where dots ‘.’ are inserted for the sake of human readability, but they are not in the actual codeword sequences. The code trees are visited in the order of \(T_0 \rightarrow T_0 \rightarrow T_1 \rightarrow T_0 \rightarrow T_0 \rightarrow T_0 \rightarrow T_1 \rightarrow T_0\).
Almost Instantaneous (AI) FV codes

Encoding (decoding) procedure use $T_0$ and $T_1$ iteratively. After using a master node, use $T_1$ for the next.

No ‘00’ from the root.

Only ‘00’ from the master node.

→The codes are uniquely decodable.
### Worst-case Redundancy of AIFV codes

<table>
<thead>
<tr>
<th>Huffman code</th>
<th>AIFV code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redundancy</td>
<td>$&lt; 1$</td>
</tr>
</tbody>
</table>

AIFV codes have **good empirical performance**. Even beat Huffman code for $\chi^2$ for some sources. [Yamamoto+ 2015]
Worst-case Redundancy of AIFV codes

<table>
<thead>
<tr>
<th>Huffman code</th>
<th>AIFV code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redundancy</td>
<td>$&lt; 1$</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$$p_{\text{max}} \equiv \max_{x \in \mathcal{X}} p_X(x).$$

Worst-case redundancy in terms of $p_{\text{max}}$ (Our result)
Worst-case Redundancy of AIFV codes

Theorem (Worst-case redundancy of AIFV codes)
For \( p_{\text{max}} = p \geq 1/2 \), the worst-case redundancy of AIFV codes is

\[
f(p) = \begin{cases} 
  p^2 - 2p + 2 - h(p) & \text{if } \frac{1}{2} \leq p \leq \frac{-1+\sqrt{5}}{2}, \\
  -\frac{2p^2+p+2}{1+p} - h(p) & \text{if } \frac{-1+\sqrt{5}}{2} \leq p < 1. 
\end{cases}
\]

Theorem (Redundancy upper bound of AIFV codes)
For \( p_{\text{max}} < 1/2 \), the worst-case redundancy is at most \( 1/4 \).
Comparison with Huffman codes
Comparison with Huffman codes

**Corollary (Worst-case Redundancy)**
Worst-case redundancy of binary AIFV codes is $\frac{1}{2}$. 

\[
\min_K \sum_{k=1}^{K} p(x_k)l(x_k) 
\leq 1 - l(x_k) \leq 1 
\leq 1 
\]

\[
K \sum_{k=1}^{K} 2^{-l(x_k)} \leq 1 
\]

\[
Y = \{0, 1\} 
\]

\[
p_{\text{max}} = \max_{x \in X} p(X) 
\leq \frac{1}{2} 
\]

\[
f(p) = \begin{cases} 
    p^2 - 2p + 2 - h(p) & \text{if } 0.5 \leq p \leq 1 - \frac{1}{\sqrt{5}} \\
    -2p^2 + p + 2 + h(p) & \text{if } 1 - \frac{1}{\sqrt{5}} < p < 1 \end{cases} 
\]

\[
O(|X|) \leq O(|X|^2) \leq O(|X|^m) \leq O(m|X|) 
\]

$\frac{1}{m}$, $\frac{1}{2}$, $1$.
Comparison with Huffman codes

<table>
<thead>
<tr>
<th></th>
<th>Huffman</th>
<th>AIFV</th>
<th>Huffman for (\mathcal{X}^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redundancy</td>
<td>&lt; 1</td>
<td>&lt; 1/2</td>
<td>&lt; 1/2</td>
</tr>
<tr>
<td>Storage for Code trees</td>
<td>(O(</td>
<td>\mathcal{X}</td>
<td>))</td>
</tr>
</tbody>
</table>

\(\mathcal{X}\) : Source alphabet

More memory efficient than Huffman codes for \(\mathcal{X}^2\).
Proof idea

Goal:
Prove bounds of optimal binary AIFV codes

Challenge:
No simple algorithm known to construct the optimal AIFV code.
→ Difficult to analyze optimal code directly...
Proof idea

Our approach:
Simple Construction of sub-optimal AIFV codes from Huffman codes.

Redundancy

Sub-optimal AIFV code

Optimal AIFV code

All kinds of sources with a fixed $p_{\text{max}}$
Proof idea

Our approach:
Simple Construction of sub-optimal AIFV codes from Huffman codes.

Redundancy

Tight in the worst-case!

Sub-optimal AIFV code

Optimal AIFV code

All kinds of sources with a fixed $p_{\text{max}}$
Proof outline (1/6)

- Simple **two-stage construction** of sub-optimal AIFV code trees from Huffman tree

Ex.)

![Huffman Tree Example]

$q_1 \geq q_2, \cdots \geq q_{2K-2}$.

$K$: size of source alphabet

Sibling pair: $(q_{2k-1}, q_{2k})$

Sibling property

[Gallager 1978]
Proof outline (2/6)

- **Two-stage construction**

1. From $T_{\text{Huffman}}$ to $T_{\text{base}}$
   - for sibling index $k = 2, \ldots, K - 1$ do
   - if $q_{2k-1}$ is a leaf and $2q_{2k} < q_{2k-1}$ then

   ![Diagram of tree transformations](image-url)
Proof outline (3/6)

- **Two-stage** construction

2. From $T_{base}$ to $T_0$ and $T_1$

Only if $\frac{-1 + \sqrt{5}}{2} \leq q_1$
Proof outline (4/6)

- Simple two-stage construction of sub-optimal AIFV code trees from Huffman tree

Ex.)
Proof outline (5/6)

• Upper bounds for sub-optimal AIFV code can be evaluated $\Rightarrow$ tight for $p_{\text{max}} \geq \frac{1}{2}$. Why?
Proof outline (6/6)

Optimal AIFV trees for \((p_{\text{max}}, 1 - p_{\text{max}} - \delta, \delta)\) coincides with worst-case trees of sub-optimal AIFV codes.

\(\begin{align*}
T_0 &\quad T_1 \\
0 &\quad 0 \\
1 &\quad 1 \\
1 - p_{\text{max}} - \delta &\quad 1 - p_{\text{max}} - \delta
\end{align*}\)

(a) \(\frac{1}{2} \leq p_{\text{max}} \leq \frac{\sqrt{5} - 1}{2}\).

(b) \(\frac{\sqrt{5} - 1}{2} \leq p_{\text{max}} \leq 1\).

→ The bound of sub-optimal trees is tight for \(p_{\text{max}} \geq \frac{1}{2}\).
Conclusion

1. Worst-case redundancy of binary AIFV codes is $1/2$.

2. Worst-case redundancy in terms of

$$p_{\text{max}} = p \geq 1/2.$$ 

**Theoretical justification** for superior performance of AIFV codes over Huffman codes.

**Further extension**

If the codes are allowed to use 3 and 4 code trees, worst-case redundancy is $1/3$ and $1/4$, respectively.