1 VOUW: A Framework

1.1 Encoding Models and Instantiations

Definition 1. *Given a set of instantiations $A|H_A$, we take* $\texttt{usage}(X)$ *of a pattern X to be the prevalence of X^* in $A|H_A$. More precisely*

$$\texttt{usage}(X) = |\{R \mid R = (Y, t, \boldsymbol{p}) \in \{A|H_A\}, Y^* = X^*\}|$$

From this definition we see that the *usage* of a pattern is sum a of how often any of its variants occur as an instance. Using this function we find the probability that a certain pattern occurs simply by $P(X) = \frac{\texttt{usage}(X)}{|\{A|H_A\}|}$. The optimal length of a code word L^C can then be found by Shannon's Entropy.

Pattern dimensions The number of bits required for the spatial offsets depend directly on the *area* the pattern covers in A. We define this area informally as the difference in rows and columns between the smallest and the largest offset in a pattern X. Furthermore, instead of computing the area, we compute the *width* and *height* of a pattern separately. These are defined in two steps: first we define $\texttt{rowMax}(X) = i \iff \big((i,j), \boldsymbol{w}\big) \in X \wedge \nexists \big((i',j'), \boldsymbol{w'}\big) \in X$ s.t. $i' > i$, and analogously $\texttt{rowMin}(X)$, $\texttt{colMax}(X)$ and $\texttt{colMin}(X)$ as the largest and smallest row and column occurring in an offset of X respectively. We can then simple say that $\texttt{width}(X) = \texttt{colMax}(X) - \texttt{colMin}(X)$ and define $\texttt{height}(X)$ analogously for the row offsets.

A problem with this approach is that it only looks at the surface area of the pattern. For example, patterns measuring 2×8 and 4×4 have equal code lengths while the latter *may* be favourable.

Pattern code length

$$L(X) = |X| \cdot \Bigg(-\log\Big(\texttt{height}(X)^{-1}\Big) - \log\Big(\texttt{width}(X)^{-1}\Big) - \log\Big(b(A)^{-1}\Big)\Bigg)$$

There the term $-\log\Big(b(A)^{-1}\Big)$ has a big influence on the encoding performance while $b(A)$ says little about the distribution of values in A.

Variant encoding Each region simply refers to its pattern X by using the code word we computed earlier and we already know its length. The pivot is again a fixed number that depends on the total number of instantiations $|\{A|H_A\}|$. Encoding the variant is harder because we have never clearly defined $|X^*|$. In principle the upper bound for the number of variants for a given pattern X is $b(A)^{|X|}$, since we established X^* is at least finite. This is not a practical figure however, given that there probably are far less elements in A. A solution is to limit the total number of variants for X to the number that *we know about* at a given moment. We can find this number in a similar way to the $\texttt{usage}$ function.

Definition 2. *Given a set of instantiations $A|H_A$, we define*

$$\texttt{variants}(X) = |\{Y \circ t \mid R = (Y, t, \boldsymbol{p}) \in \{A|H_A\}, Y^* = X^*\}|$$

Which basically defines $\texttt{variants}(X)$ as the number of distinct variants of X that occur within $A|H_A$.

Region code length

$$L(R) = -log\big(|\{A|H_A\}|^{-1}\big) - \log\left(\texttt{variants}(X)^{-1}\right) + L^C(X)$$

Where X is the pattern in R. Note that the size of the instance set $|\{A|H_A\}|$ is used both here as well as to compute the code length of a single pattern, giving a bias to the size of the instance set (i.e. favouring many patterns and few regions).

Total code length sums

$$L(H_A) = \sum_{X \in H_A} L^C(X) + L(X)$$
$$L(A|H_A) = \sum_{R \in A|H_A} L(R)$$