prefix property; we need $\gamma'(x) + 1 = \lceil \log \gamma(x) \rceil + 1$ bits for this. All in all, we end up with a code concatenating the trivial encoding of $\gamma'(x)$, the uniform encoding of $\gamma(x)$ given $\gamma'(x)$, and a uniform encoding of x given $\gamma(x)$, with:

$$L(x) = \lceil \log m_{\gamma(x)} \rceil + 2\lceil \log \gamma(x) \rceil + 1 \text{ bits.} \tag{3.4}$$

Since for all γ, $m_{\gamma+1} \geq m_\gamma + 1$, we have $\gamma \leq m_\gamma$ so that $L(x) \leq 3\lceil \log m_{\gamma(x)} \rceil + 1$. We are often interested in situations with $m_\gamma = |\mathcal{M}_\gamma|$ increasing exponentially, e.g. $\mathcal{M}_\gamma = \{1, \ldots, 2^{\gamma-1}\}$. Then $\gamma = \log m_\gamma + 1$ and (3.4) gives

$$L(x) \leq \lceil \log m_{\gamma(x)} \rceil + 2\lceil \log \lceil \log m_{\gamma(x)} + 1 \rceil \rceil + 1 \leq \\ \log m_{\gamma(x)} + 2 \log(\log m_{\gamma(x)} + 1) + 3. \tag{3.5}$$

We call the resulting code with lengths (3.5) a *quasi-uniform description method*; see also the box on page 91. We refine this description method and provide a precise definition in Section 3.2.3. One possible rationale for using general quasi-uniform description methods is given by the *luckiness principle*, described in the box on page 92. The name "luckiness principle" is not standard in the MDL area. I have adopted it from the computational learning theory community, where it expresses a superficially distinct but on a deeper level related idea, introduced by Shawe-Taylor, Bartlett, Williamson, and Anthony (1998); see also (Herbrich and Williamson 2002).

3.1.3 Assessing the Efficiency of Description Methods

We frequently need to compare description methods in terms of how well they compress particular sequences.

Definition 3.2 Let C_1 and C_2 be description methods for a set $\mathcal{A}$.

1. We call C_1 *more efficient* than C_2 if for all $x \in \mathcal{A}$, $L_1(x) \leq L_2(x)$ while for at least one $x \in \mathcal{A}$, $L_1(x) < L_2(x)$.
2. We call a code C for set $\mathcal{A}$ *complete* if there does not exist a code C' that is more efficient than C.

We note that a "complete" description method must always be a code; a code can be complete or not, as the case may be.

3.2 The Most Important Section of This Book: Probabilities and Code Lengths, Part I

All theory about MDL that we will develop builds on the following simple but crucial observation: