describe the first approach.

Consider a system with 7^d states, where each state is a vector $v \in \{1,2,3,4,5,6,*\}^d$, where $v_i \in \{1,2,3,4,5,6\}$ means that the i'th dice is currently showing its v_i'th side, and $v_i = *$ means that we are currently re-rolling the i'th dice. Let us refer to states containing $*$'s as undetermined states, and the remaining states as fixed states.

Initially, we are in the undetermined state $(*,*,\ldots,*)$ corresponding to our first roll, and our goal is to reach one of the fixed states that correspond to a word in the dictionary (after possibly permuting the coordinates).

From a fixed state, we can for a cost of 0 choose to replace any non-empty subset of the coordinates by $*$ and move to that undetermined state, indicating which dice we choose to reroll. From an undetermined state, the $*$'s will be replaced uniformly at random by digits 1-6 and we will move to that fixed state for a cost of 1 (corresponding to the re-rolling of the dice).

Now we want to make the right choices at all the fixed states in such a way that we reach one of the goal states with minimum possible total expected cost. We can do this by running a variant of Dijkstra's algorithm backwards from the goal nodes. Initially, all states x have a distance of $\text{dist}[x] = \infty$, except the goal states which have a distance of 0, and we process the states in increasing order of distance.

Whenever we process a fixed state x, we have to update the distance of all undetermined states y that match x. Suppose that y has a total of $s > 1$ $*$'s, and let D be the set of fixed states matching y that have been processed. Then we can update $\text{dist}[y]$ to $\frac{6^s + \sum_{z \in D} \text{dist}[z]}{|D|}$ (if this is better than the current distance for y); this corresponds to the strategy "keep rerolling these $*$'s until we get to one of the states in D – it takes in expectation $6^s/|D|$ re-rolls to reach a state in D, and since this will be uniformly random its expected distance will be $\frac{1}{|D|}\sum_{x \in D} \text{dist}[z]$. To make this fast, we should keep track of $|D|$ and $\sum_{x \in D} \text{dist}[z]$ for each y and not recompute them every time.

Whenever we process an undetermined state x, the situation is simpler and we can simply update each fixed state y that matches x, updating $\text{dist}[y]$ to $\text{dist}[x]$ (if this is better than current distance for y); this corresponds to the strategy which given y moves to x by changing the corresponding coordinates to $*$.

After running this, the distance for $(*,*,\ldots,*)$ contains the answer.