Appendix

A. Proof of Theorem 1

Proof sketch. In summary, we can build a reduction from *set covering* problem [14], one of the Karp's 21 NP-complete problems, to the decision version of Problem 1, thus showing the NP-hardness of Problem 1.

First, we give the decision version of Problem 1, which is to decide whether there exists a scheme to choose files, that have $\Delta C(F) = \kappa$. We first show that the decision version of Problem 1 is in NP. Given a scheme of selected files for compaction, we could easily compute the costs with the files following Formula 2, thus verifying the conditions within polynomial time. Therefore, the problem is in NP.

Next, we build a reduction from *set covering* problem. Given a set U of all the elements and V which is a collection of subsets in U, assuming that all the subsets in V are sufficient to cover U, the problem is to decide whether there exists a scheme of subsets whose union could cover U, and having a size less than or equal to a given number η.

For an arbitrary instance of *set covering* problem, we will construct an instance of our problem, to show their equivalence. Let $U = \{u_1, u_2, \ldots, u_{|U|}\}$, denote all the elements of *set covering* problem, and $V = \{V_1, V_2, \ldots, V_{|V|}\}$, having $V_j \subsetneq U, \forall j \in \{1, 2, \ldots, |V|\}$. We then construct the files in our problem and set them with only one dimension. First, let $c^k = c^v = 0$, i.e., we do not consider the space costs of the values and the keys. Let $F_i = \{f_1, f_2, \ldots, f_{|V|}\}$ denote the $|V|$ files in Level i, corresponding to V. Next, for each f_j, we set them containing $|U|$ keys, if $u_i \in V_j$, the i-th key of f_j has its value, and if $u_i \notin V_j$, the i-th key of f_j refers to null value, i.e., recording 0 in the bitmap. In such settings, we make f_j correspond to V_j respectively. We then let $M = \eta$, i.e., finding η files. The problem of finding a group of subsets from V to cover U with size no more than η, is clearly equivalent to the problem of finding a compaction scheme, with files no more than $M = \eta$ and $\Delta C(F) = 0$. To conclude, an arbitrary instance of *set covering* problem could be reduced to the decision version of our problem. Hence, the NP-hardness of Problem 1 is proved.

B. Proof of Proposition 2

Proof sketch. Let v_1, v_2 denote two versions of a value and v_2 is the newer version of v_1. Let $l(v_1), l(v_2)$ denote the current levels of v_1 and v_2 in LSM-tree, and $f(v_1), f(v_2)$ denote the files including v_1 and v_2 respectively. To ensure the correctness, we should ensure $l(v_1) \geq l(v_2)$ at any time.

First, the out-of-place updates and the structure of the LSM-tree ensure that $f(v_2)$ is flushed to Level 0 later than $f(v_1)$. That means initially, we have $l(v_1) \geq l(v_2)$. Next, we consider two scenarios:

(1) If $l(v_1) > l(v_2)$, it is safe to compact $f(v_2)$, since it does not incur problems.
(2) If $l(v_1) = l(v_2)$, the DAG constraint ensures that if $f(v_2)$ is selected, $f(v_1)$ must also be selected. The compaction of $f(v_1)$ is not restricted regarding $f(v_2)$. This also guarantees $l(v_1) \geq l(v_2)$.

To conclude, DAG constraint ensures the correctness.