Proof. Let t be an output tuple in $Q_1(D) \setminus Q_2(D)$. Since Q_1 and Q_2 are SPJUD* queries that can be written as nested differences of queries like $q_1 - q_2 - (q_3 - (q_4 - q_5)) - ...$, where all q_i-s are SPJU queries, $Q_1 - Q_2$ is also an SPJUD* query. The output tuple t must be either in or not in the result of each q_i. We find the smallest witness by enumerating the minimal witnesses of t w.r.t. every q_i and D. If t is in the result of $q_i(D)$, let w_i be the set of minimal witnesses of t w.r.t. q_i and D. Then we pick one element from every $w_i \cup \{\emptyset\}$, and construct w as the union of all witnesses or the empty set we picked. We evaluate Q_1 and Q_2 on w to see whether it is a witness for t, and record the w of the smallest size. We finish this procedure until we enumerate all combinations.

This procedure will return the smallest witness because: (i) if $t \notin q_i(D)$, t will also not be in $q_i(w)$ for any $w \subseteq D$ due to monotonicity, so we don't need to consider such q_i-s; (ii) Assume that w' is a smallest witness of t w.r.t. $Q_1 - Q_2$ and D, for all q_i where $t \in q_i(w')$, w' must be a superset of a minimal witness of t w.r.t. q_i and D. Hence w' must be the union of minimal witnesses of t w.r.t. these q_i-s and D; otherwise, if w' is a strict superset of the union of minimal witnesses of t, we can always remove tuples not belong to any minimal witness of t w.r.t. q_i-s and D from w', without affecting t to be in or not in any q_i, which contradicts the assumption that w' is a smallest witness. Therefore a smallest witness of t w.r.t. $Q_1 - Q_2$ and D must be union of minimal witness of t w.r.t. q_i and D, and thus it must be enumerated during the enumeration procedure.

The time complexity of entire enumeration process is $O(\Pi_i m^{k_i}) = O(m^{kd})$, where d is the number of difference operators, m is the max size of relations, k_i is the max complexity of each SPJU query q_i (i.e., the number of joins in q_i is $k_i - 1$), $k = \max_i k_i$ and d is the number of q_i-s. When queries are of bounded sizes, i.e., if d and k are fixed, the SWP for two SPJUD queries that can be written as nested differences of SPJU queries is polynomial-time solvable. □

SWP is NP-hard in general even for bounded-size queries.

Theorem 8. *The SWP for two SPJUD queries Q_1 and Q_2 is NP-hard in data complexity.*

Proof. We again give a reduction from the vertex cover problem with vertex degree at most 3 (see Theorem 3).

Construction. Suppose in $G = (V, E)$, $V = \{v_1, ..., v_n\}$, $E = \{e_1, \cdots, e_m\}$. We construct two relations $R(A, Z, E_1, E_2, E_3)$ and $S(B, C, Z)$. For each vertex $v_i \in V$, R contains a tuple $t_i = (v_i, z, e_{i_1}, e_{i_2}, e_{i_3})$, where $e_{i_1}, e_{i_2}, e_{i_3}$ are the identifiers of edges adjacent to v_i, $i_1 < i_2 < i_3$. If the degree of v_i is less than 3, the identifiers are replaced by a null symbol "$*$". Here z is a constant. For each edge $e_i \in E$, S contains a tuple $(e_i, e_{(i\%m)+1}, z)$, where $e_{(i\%m)+1}$ is the identifier of the next edge in the edge list (the next edge of e_m is e_1). Let $D = (R, S)$ be the database instance.

Next, we construct an SPJUD query that consists of several subqueries as follows: Let q_1 (on S) = $\pi_Z(S)$; q_2 (on S) = $\pi_{B,Z}(S)$; q_3 (on R, S)=$\pi_{S.C,S.Z}(S \bowtie_{S.C=E_1 \vee S.C=E_2 \vee S.C=E_3} R)$. Then we construct $Q_1 = q_1$, hence $Q_1(D) = \{(z)\}$. We also construct $Q_2 = \pi_Z(q_2 - q_3)$ (assume C in q_3 is renamed to B). For edge $e_i = (v_j, v_\ell)$, the edge e_i appears for both tuples t_j, t_ℓ (in one of E_1, E_2, E_3 attributes), and therefore, (e_i, z) appears in the result of $q_3(D)$ for every $i \in [1, m]$. Hence $q_3(D) = \pi_{B,Z}(S)$. So $q_2(D) \setminus q_3(D) = \emptyset$. Then $(Q_1 - Q_2)(D) = \{(z)\}$, and the goal is to find the smallest witness for (z). For the vertex cover instance in Figure 11(a), R will be as given in Figures 11(c), and S will contain tuples $\{(e_1, e_2, z), (e_2, e_3, z), \cdots (e_7, e_1, z)\}$.

We now show that *there exists a vertex cover C of size at most p in the graph G if and only if there is a witness $D' = (R', S')$ where $|R'| + |S'| \le p + m$.*

The "Only If" direction. Suppose we are given a vertex cover C of G with at most k vertices. Construct $R' = \{t_i \mid v_i \in C\}$, and $S' = S$. $Q_1(D) = Q_1(D') = \{(z)\}$ since S is unchanged. Similarly, $q_2(D') = \pi_{B,Z}(S)$ is unchanged. Since C is a vertex cover, for every edge $e_i = (v_j, v_\ell)$ either t_i or t_ℓ is in R'; hence $q_3(D') = q_3(D)$, i.e., each (e_i, z), $i \in [1, m]$ appears in $q_3(D')$. Then $Q_1 - Q_2$ outputs (z) on D', $|R'| = |C| \le p$, $|S'| = |S| = m$, and we get a witness of at most $p + m$ tuples.

The "If" direction. Consider any witness $D' = (R', S')$ where $R' \subseteq R, S' \subseteq S, |R'| + |S'| \le p + m$, such that $(z) \in Q_1(D') \setminus Q_2(D')$. We construct $C = \{v_i \mid t_i \in R'\}$. Since (z) is in $Q_1(D') \setminus Q_2(D')$, (z) must be in the result of $q_1(S')$, and not in the result of $q_2(S') - q_3(R', S')$, hence S' must contain at least one tuple. Therefore, $q_2(S')$ outputs at least one tuple (e_i, z) since S' is not empty. In turn, $q_3(R', S')$ must output all tuples in $q_2(S')$ to make $q_2(S') - q_3(R', S')$ empty. (a) We argue that $S' = S$. Suppose S' contains at least one tuple, say wlog, (e_1, e_2, z). Then to remove (e_1, z) from $q_2(S') \setminus q_3(R', S')$, $q_3(R', S')$ must contain (e_1, z), which can generate only from $S(e_m, e_1, z)$. Hence $(e_m, e_1, z) \in S'$. In turn, $(e_m, z) \in q_2(S')$. To remove it, we need $S(e_{m-1}, e_m, z)$ in S'. Continuing this argument (by induction), we get $S = S'$. (b) Consider any tuple, say wlog., (e_1, e_2, z) in S'. Then to remove (e_1, z) from $q_2(S') \setminus q_3(R', S')$, not only the tuple $(e_m, e_1, z) \in S'$, it also has to satisfy the join condition with R. This will hold only if for one of the end points v_j, v_ℓ of $e_1 = (v_j, v_\ell)$, $t_j \in R'$ or $t_\ell \in R'$. This should hold for all edges, and therefore the set C we constructed is a vertex cover. Since $|S'| = |S| = m$, $|R'| = |C| \le p$, therefore, we get a vertex cover in G of size at most p.

The queries we constructed during the reduction are all of bounded size, therefore the SWP for two SPJUD queries is NP-hard in data complexity. □