Towards Efficient Random-Order Enumeration for Join Queries Technical Report

0.1 Proof of Lemma 2.

Proof. We prove the contrapositive: if $\varphi_Q^*(i) \neq \mathit{false}$, then $i \leq upp(\psi_r)$. By the definition of φ^*, for each integer $i > 0$ satisfying $\varphi_Q^*(i) \neq \mathit{false}$, there is a tuple $t \in Res(Q)$ along with a root-to-leaf path $u_1, \ldots, u_h$ ($u_1 = r, u_h = u_t$) such that (11) holds. Then, let

$$\begin{aligned} \mathrm{sum}_j &= \sum_{\substack{\psi \prec \psi_{u_{j+1}} \\ \psi \in children(\psi_{u_j})}} upp(\psi) \\ &\leq \sum_{\psi \in children(\psi_{u_j})} upp(\psi) - upp(\psi_{u_{j+1}}) \\ &\leq upp(\psi_{u_j}) - upp(\psi_{u_{j+1}}), \end{aligned} \tag{1}$$

therefore

$$\begin{aligned} i &= \sum_{j=1}^{h-1} \mathrm{sum}_j + 1 \leq \sum_{j=1}^{h-1} \Big(upp(\psi_{u_j}) - upp(\psi_{u_{j+1}}) \Big) + 1 \\ &= upp(\psi_{u_1}) - upp(\psi_{u_h}) + 1 = upp(\psi_r). \end{aligned} \tag{2}$$

□

0.2 Proof of Lemma 4.

Proof. For each tuple $t \in R$ satisfying ψ, we have $l_i \leq t(x_i) \leq h_i$ for $\forall i \in \{r_1, \ldots, r_d\}$. This implies $t_l^\psi \preceq t \preceq t_h^\psi$, and consequently $t \in R[t_l^\psi, t_h^\psi]$. Therefore, $R|_\psi \subseteq R[t_l^\psi, t_h^\psi]$. For each tuple $t \in R[t_l^\psi, t_h^\psi]$, we have $t_l^\psi \preceq t \preceq t_h^\psi$. We need to prove that for each $x_i \in \mathrm{att}(R)$, $l_i \leq t(x_i) \leq h_i$. Let s be a split position of ψ, then:

(1) For $\forall x_i \in \mathrm{att}(R)$ with $\forall 1 \leq i < s$, since $l_i = h_i$, it follows that $l_i = t(x_i) = h_i$. Otherwise, one of the conditions $t \succeq t_l^\psi$ or $t \preceq t_h^\psi$ would be violated.
(2) If $x_s \in \mathrm{att}(R)$ then $l_s \leq t(x_s) \leq h_s$, otherwise $t_l^\psi \preceq t \preceq t_h^\psi$ would be violated.
(3) For $\forall x_i \in \mathrm{att}(R)$ with $\forall s < i \leq n$, since $l_i = \min_Q(x_i)$ and $h_i = \max_Q(x_i)$, it follows that $l_i \leq t(x_i) \leq h_i$.

Then t satisfies ψ, *i.e.*, $t \in R|_\psi$. Therefore, $R[t_l^\psi, t_h^\psi] \subseteq R|_\psi$ and then we have $R[t_l^\psi, t_h^\psi] = R|_\psi$, which implies that $R.cnt(\psi) = |R|_\psi|$. □

0.3 Proof of Lemma 6.

Proof. Since $upp(\psi) \leq 1$, two cases need to be discussed. Firstly, for the case $upp(\psi) = 0$, we have $|Res(Q|_\psi)| \leq upp(\psi) = 0$, which implies that $Res(Q|_\psi) = \emptyset$. Secondly, for the case $upp(\psi) = 1$, we have $|Res(Q|_\psi)| \leq 1$. Let $s = \max\{i | l_i \neq h_i\}$, by Property 1, there exists at most one integer $p \in \mathbb{N}[l_s, h_s]$ such that

- $upp([[l_1, h_1], \ldots, [l_s, p-1], \ldots, [l_n, h_n]]) = 0$,
- $upp([[l_1, h_1], \ldots, [p, p], \ldots, [l_n, h_n]]) = 1$,
- $upp([[l_1, h_1], \ldots, [p+1, h_s], \ldots, [l_n, h_n]]) = 0$.

If $upp([[l_1, h_1], \ldots, [p, p], \ldots, [l_n, h_n]]) = 0$ follows for every $p \in \mathbb{N}[l_s, h_s]$, we have $Res(Q|_\psi) = \emptyset$. Otherwise, it is able to calculate the only integer $p \in \mathbb{N}[l_s, h_s]$ by a multi-head binary search (introduced in Section 4.4) in $O(\log |Q|)$ time. Then apply the same procedure to the next unfixed position of $[[l_1, h_1], \ldots, [p, p], \ldots, [l_n, h_n]]$, and continue this process iteratively on other positions as needed. If this process finally obtains $\psi^* = [[l_1^*, h_1^*], \ldots, [l_n^*, h_n^*]]$ such that $upp(\psi^*) = 1$ and $\forall 1 \leq i \leq n, l_i^* = h_i^*$, then $(l_1^*, \ldots, l_n^*)$ is the only tuple in $Res(Q|_\psi)$. Otherwise, we have $Res(Q|_\psi) = \emptyset$. The time it takes to decide whether $Res(Q|_\psi) = \emptyset$ and calculate ψ^* (if exists) is at most $O(n \log |Q|)$, which is $O(\log |Q|)$ in data complexity. □

0.4 Proof of Lemma 8.

Proof. For the convenience of the proof, a dummy element $+\infty$ is appended to the end of each sequence, that is, $A_i[|A_i|] = +\infty$ for each $1 \leq i \leq k$. If $F(|A_1|, \ldots, |A_k|) \leq T$, the algorithm obviously returns p^*. Then for the case where $F(|A_1|, \ldots, |A_k|) > T$, we show that Algorithm 4 maintains the invariant

$$A_i[l_i] \leq p^* \leq A_i[r_i], \quad \forall\, 1 \leq i \leq k, \tag{3}$$

and then prove that Algorithm 4 correctly returns p^* after at most $O(\log \sum_{i=1}^k |A_i|)$ iterations.

Base case. Initially, for each $1 \leq i \leq k$ we set $l_i = 0$ and $r_i = |A_i|$. Since each A_i is sorted,

$$A_i[0] \leq \min A_i \leq p^* \leq \max A_i \leq A_i[n], \tag{4}$$

then the invariant holds before the first iteration.

Inductive step. Suppose at the start of some iteration we have $A_i[l_i] \leq p^* \leq A_i[r_i]$ for any $1 \leq i \leq k$. We aim to demonstrate that the invariant is preserved after the update performed in the current iteration. Without loss of generality, we assume that for any $1 \leq i \leq k$, $r_i - l_i > 1$ satisfies. (If some i already has $r_i - l_i \leq 1$ and the algorithm has not yet terminated, then by the invariant we have $A_i[l_i] < p^* \leq A_i[r_i]$ and $N_i(A_i[r_i]) \leq N_i(A_i[l_i] + 1) = r_i$, then $N_i(p^*) = r_i = m_i$, and the problem reduces to the remaining $k-1$ sequences.) In each iteration, the algorithm modifies exactly one of the two bounds: it either updates the lower bound $l_{i_{\min}}$ for some index $i_{\min}$, or updates the upper bound $r_{i_{\max}}$ for some index $i_{\max}$. We discuss the two cases:

Case 1: $F(m_1, \ldots, m_k) \leq T$.

Let $p = A_{i_{\min}}[m_{i_{\min}}] = \min_{r_i - l_i > 1} A_i[m_i]$. Since

$$N_{i_{\min}}(p) = \left|\{x \in A_{i_{\min}} \mid x < A_{i_{\min}}[m_{i_{\min}}]\}\right| \leq m_{i_{\min}}, \tag{5}$$

and for each $i \neq i_{\min}$,

$$N_i(p) \leq N_i\big(A_i[m_i]\big) = \left|\{x \in A_i \mid x < A_i[m_i]\}\right| \leq m_i, \tag{6}$$

it follows that

$$F\big(N_1(p), \ldots, N_k(p)\big) \leq F(m_1, \ldots, m_k) \leq T. \tag{7}$$

Hence $p^* \geq p$, then after updating $l_{i_{\min}} \leftarrow m_{i_{\min}}$, we still have $A_{i_{\min}}[l_{i_{\min}}] \leq p^* \leq A_{i_{\min}}[r_{i_{\min}}]$. For all $i \neq i_{\min}$, neither l_i nor r_i changes, so the invariant is preserved after the update.

Case 2: $F(m_1, \ldots, m_k) > T$.

Let $p = A_{i_{\max}}[m_{i_{\max}}] = \max_{r_i - l_i > 1} A_i[m_i]$. Since

$$N_{i_{\max}}(p+1) = |\{x \in A_{i_{\max}} | x \leq A_{i_{\max}}[m_{i_{\max}}]\}| \geq m_{i_{\max}}, \quad (8)$$

and for each $i \neq i_{\min}$,

$$N_i(p+1) \geq N_i(A_i[m_i]+1) = |\{x \in A_i | x \leq A_i[m_i]\}| \geq m_i, \quad (9)$$

it follows that

$$F(N_1(p+1), \ldots, N_k(p+1)) \geq F(m_1, \ldots, m_k) > T. \quad (10)$$

Thus $p^* < p+1$ which implies $p^* \leq p$, then after updating $r_{i_{\max}} \leftarrow m_{i_{\max}}$, we still have $A_{i_{\max}}[l_{i_{\max}}] \leq p^* \leq A_{i_{\max}}[r_{i_{\max}}]$. For all $i \neq i_{\max}$, neither l_i nor r_i changes, so the invariant is preserved after the update.

Convergence and termination. In each iteration, an update is performed on sequence A_i only if $r_i - l_i > 1$, and every such update reduces the length of the interval by at least one. Consequently, the length of each interval $[l_i, r_i]$ shrinks to at most 2 after at most $O(\log |A_i|)$ updates on A_i. Since the number of sequences is a constant, the total number of iterations of the algorithm is bounded by $O(\log \sum_{i=1}^{k} |A_i|)$.

Now we show that when Algorithm 4 terminates, it returns p^* correctly. If it terminates at Line 4 or Line 14, which implies

$$F\big(N_1(A_i[l_i]+1), \ldots, N_k(A_i[l_i]+1)\big) > T, \quad (11)$$

it follows that $p^* \leq A_i[l_i]$, then by the invariant $p^* \geq A_i[l_i]$, we have $p^* = A_i[l_i]$.

If Algorithm 4 terminates at line 16, which implies $r_i - l_i \leq 1$ and $F(N_1(A_i[l_i]+1), \ldots, N_k(A_i[l_i]+1)) \leq T$ satisfies for every $1 \leq i \leq k$. Let $p = \min_{1 \leq i \leq k} A_i[r_i]$ and $q = \max_{1 \leq i \leq k} A_i[l_i]$, then

$$\begin{aligned} F(N_1(p), \ldots, N_k(p)) &\leq F(N_1(A_1[r_1]), \ldots, N_k(A_k[r_k])) \\ &\leq F(N_1(A_1[l_1]+1), \ldots, N_k(A_k[l_k]+1)) \\ &\leq F(N_1(q+1), \ldots, N_k(q+1)) \leq T. \end{aligned} \quad (12)$$

Thus $p^* \geq p$, and by the invariant, $p^* \leq p$, we conclude $p^* = p = \min_{1 \leq i \leq k} A_i[r_i]$.

In summary, since $i_{\min}$, $i_{\max}$, and F can be computed in $O(1)$ time, and computing each N_i takes at most $O(\log |A_i|)$ time, Algorithm 4 returns p^* in $O(\log \sum_{i=1}^{k} |A_i|)$ time. □

0.5 Proof of Lemma 9.

Proof. Let $\psi = [[l_1, h_1], \ldots, [l_n, h_n]]$, since $s = \min\{i | l_i \neq h_i\}$ is a split position of ψ, it follows that

(1) $\forall 1 \leq i < s, l_i = h_i$,
(2) $\forall s < i \leq n, l_i = \min_Q(x_i)$ and $h_i = \max_Q(x_i)$.

If ψ_{left} is not an empty range, then $l_p \leq mid_p - 1$, which implies that ψ_{left} is a prefix range filter with split position s. Similarly, if ψ_{left} is not an empty range, then $mid_p + 1 \leq h_p$, which implies that ψ_{right} is a prefix range filter with split position s. Finally, since $mid_p \leq mid_p$, ψ_{mid} is a prefix range filter with split position s. □

0.6 Proof of Lemma 11.

Proof. If $\text{att}(Q_s) = \text{att}(Q)$, then $Res(Q|_\psi) \subseteq Res(Q_s|_\psi)$, which implies $|Res(Q|_\psi)| \leq |Res(Q_s|_\psi)|$. If $\text{att}(Q_s) \subsetneq \text{att}(Q)$, then

$$\begin{aligned} |Res(Q|_\psi)| &= |Res(Q_s|_\psi) \bowtie Res(Q_r|_\psi)| \\ &\leq \sum_{t \in Res(Q_s|_\psi)} |Res((Q_r \ltimes t)|_\psi)| \\ &\leq |Res(Q_s|_\psi)| \max_{t \in Res(Q_s|_\psi)} |Res((Q_r \ltimes t)|_\psi)| \\ &\leq |Res(Q_s|_\psi)| \cdot |Res(Q_r^*|_\psi)| \\ &\leq |Res(Q_s|_\psi)| \cdot \text{AGM}_c(Q_r^*|_\psi). \end{aligned} \quad (13)$$

□