As we discussed in previous lectures, the output size of a join query often dominates the running time, since the algorithm has to enumerate all the output tuples. Thus, being able to compute the output size, or even provide a good upper bound on the output size becomes an important task. In this lecture, we discuss the following question: given a conjunctive query $q$, where each relation $R_j$ has size $N_j$, what is the largest possible output?

**Example 6.1.** Consider the join query $q(x, y, z) = R_1(x, y), R_2(y, z)$ where the sizes of the relations are $N_1, N_2$ respectively. It is easy to see that the largest possible output is $N_1 \cdot N_2$, which occurs when the join behaves like a cartesian product.

**Example 6.2.** Consider the triangle query $T(x, y, z) = R(x, y), S(y, z), T(z, x)$, where relations have sizes $N_R, N_S, N_T$. A first straightforward bound is $N_R \cdot N_S \cdot N_T$. We can get a better bound by noticing that the join of any two relations is an upper bound on the total size, so we get an improved bound of $\min\{N_R \cdot N_S, N_R \cdot N_T, N_T \cdot N_S\}$.

Can we do any better? We will see that another upper bound on the size of the query is $\sqrt{N_R \cdot N_S \cdot N_T}$. Notice that, depending on the relation between $N_R, N_S, N_T$, this can be a better or worse bound than the above three quantities.

### 6.1 The AGM Bound

We start by introducing some notation.

**Definition 6.3.** The fractional edge cover of a conjunctive query $q$ is a vector $u$, which assigns a weight $u_j$ to relation $R_j$, such that for every variable $x \in \text{vars}(q)$, $\sum_{j: x \in \text{vars}(R_j)} u_j \geq 1$.

We say fractional edge cover to distinguish from the (integral) edge cover, which assigns to each relation a weight of 0 or 1. The value of the minimum fractional edge cover of a CQ $q$ is denoted by $\rho^*(q)$. The AGM inequality, first proved in [AGM08], bounds the output size of a join w/o projections using any fractional edge cover.

**Theorem 6.4.** Let $q$ be a full conjunctive query. For every fractional edge cover $u$ of $q$, we have:

$$|q| \leq \prod_{j=1}^{\ell} N_j^{u_j}$$

**Example 6.5.** For the triangle query, a fractional edge cover is $u_R = u_S = u_T = 1/2$, which gives the $\sqrt{N_R \cdot N_S \cdot N_T}$ bound. Observe that this is the edge cover with the minimum value (3/2) as well. Notice
that \((u_R, u_S, u_T) = (1, 1, 0), (1, 0, 1), (0, 1, 1)\) are also valid fractional edge covers, which give the \(N_R \cdot N_S, N_R \cdot N_T\) and \(N_S \cdot N_T\) upper bounds respectively.

**Example 6.6.** As an exercise, compute the optimal fractional edge covers for the cycle query \(C_k\), and the Loomis Whitney join \(LW_k\), where:

\[
LW_k = R_1(x_2, \ldots, x_k), R_2(x_1, x_3, \ldots, x_k), \ldots, R_k(x_1, \ldots, x_{k-1})
\]

The AGM bound gives us an infinite number of upper bounds on the output size. Given the cardinalities of each relation, how can we find the best (minimum) possible bound? We can achieve this by minimizing the quantity \(\prod_{j=1}^\ell N_j^{u_j}\) by solving the following linear program:

\[
\begin{align*}
\text{min} & \quad \sum_j \log_2(N_j) \cdot u_j \\
\text{s.t.} & \quad \forall x \in \text{vars}(q) : \sum_{j : x \in \text{vars}(R_j)} u_j \geq 1 \\
& \quad \forall R_j : u_j \geq 0
\end{align*}
\]

The AGM bound is tight; in other words, we can always find a database instance \(I\), such that \(|q(I)|\) is equal to the the worst-case upper bound.

### 6.2 Worst-Case Optimal Joins

All of the join processing algorithms we have seen so far (e.g. for acyclic queries) have used Select-Project query plans. Consider the triangle query with relations of equal size \(N\): the worst-case output size is \(N^{3/2}\). Consider the three standard query plans to compute this query: \((R \bowtie S) \bowtie T\), \((R \bowtie T) \bowtie S\) and \((T \bowtie S) \bowtie R\). We can construct an instance such that any such plan needs time \(\Omega(N^2)\) to run (since the intermediate size of the join will be that large). In fact, even if we add projections to the plan, we can show that the running time will always be \(\Omega(N^2)\) in the worst case. The question is: can we design an algorithm that always runs in time linear w.r.t. the worst-case output, which in our case is \(O(N^{3/2})\)?

The answer to this question is yes; there exists a worst-case optimal algorithm that matches the worst-case output in running time [NRR13,V14].

### References

