Hypergraphs

*Hypergraph* = nodes plus (hyper)edges that are sets of any number of nodes.

- Applications include optimizing queries that are joins and representing “universal relations” (a useful data-modeling concept).
- Typically, nodes represent attributes and hyperedges are sets of attributes.

Example

Suppose we have relations with schemas *ABC*, *ACD*, and *BE*. This database schema could be represented by the hypergraph

![Hypergraph Diagram](image)

Acyclic Hypergraphs

These have some useful properties that make query optimization easier than the general case. Most “natural” queries correspond to acyclic hypergraphs.

Definition depends on *GYO reduction*; GYO = Graham-Yu-Ozsoyoglu.

- An *ear* is a hyperedge *H* such that we can divide its nodes into two groups: those that appear in *H* and no other hyperedge and those that are contained in another hyperedge *G*.
  - Note that an isolated edge is an ear; no *G* is needed.

- GYO reduction of a hypergraph is the process of repeatedly finding ears and removing them. That is, we remove those nodes that are in the ear and no other hyperedge; then we remove the hyperedge itself, leaving the other nodes.
  - We say that ear *H* is *consumed* by *G*, if all the nodes that are not unique to *H* are in *G*.
  - If a hypergraph is reduced to nothing
by GYO reduction, then it is said to be *acyclic*.

✦ Aside: “acyclic” makes sense: if the hypergraph is an ordinary graph, it is acyclic iff it is a tree.

---

**Example**

Here is an acyclic hypergraph

![Diagram of an acyclic hypergraph]

- The central hyperedge $DEF$ can consume each of the other three hyperedges.
- At that time, the remaining hyperedge is trivially an ear, since all of its nodes are unique to it.

---

**Formal GYO Reduction**

The original definition of GYO reduction consisted of the following two steps:

1. Eliminate a node that is in only one hyperedge.
2. Delete a hyperedge that is contained in another.

The goal is to reduce a hypergraph to a single, empty hyperedge.

- You need to look at GYO reduction this way to show that there is a unique GYO reduction of any hypergraph, acyclic or not.
  
  ✦ Key idea of proof: candidates for step (1) remain candidates, no matter what other steps are taken.

---

**Dangling Tuple Elimination**

- Useful as a first step in optimizing large joins.
• A collection of relations $R_1, R_2, \ldots, R_n$ is *locally join consistent* if for each $i$ and $j$ there are no tuples that dangle between $R_i$ and $R_j$. Formally: $\pi_{R_i}(R_i \bowtie R_j) = R_i$, and similarly when $i$ and $j$ are reversed.

• These relations are *globally join consistent* if there are no dangling tuples when considered as a group. Formally, for all $i$:
  $$\pi_{R_i}(R_1 \bowtie R_2 \bowtie \cdots \bowtie R_n) = R_i$$

• Easy to check global consistency implies local consistency.
  ✦ What about the opposite?

---

**Theorem**

If the relation schemas $R_1, R_2, \ldots, R_n$ form an acyclic hypergraph, then whenever relations for these schemas are locally consistent, they are globally consistent.

**Proof**

Induction on $n$, the number of hyperedges (relations in the join).

**Basis**: For $n = 1$ there is nothing to check.

**Induction**: Assume for $n - 1$ hyperedges, and prove for $n$.

• Let $E$ be the first ear in a GYO reduction, and let $G$ be the remaining hypergraph.

• Since $G$ has local consistency and $n - 1$ hyperedges, by the inductive hypothesis, $G$ is globally consistent.
  ✦ That is, every tuple of every relation of $G$ appears in the result of the join.

• $E$ was consumed by some hyperedge $H$, and $E$ is locally consistent with $H$. Therefore, each tuple $t$ of $E$ joins with some tuple $s$ of $H$.

• $s$ appears as part of some tuple $r$ in the join of the relations in $G$. Since attributes of $E$ are either unique to it, or in $H$, $t$ joins with $r$.
  ✦ Thus, $t$ participates in the join of all $n$ relations.

• However, if the hypergraph is not acyclic, we can always find relations that are locally consistent but not globally consistent.
Example
Consider $AB = \{00,11\}$, $BC = \{00,11\}$, and $AC = \{01,10\}$.

- Any two relations are join-consistent. E.g., $AB \Join AC = \{001,110\}$, which projected onto $AB$ is $\{00,11\}$.
- But $AB \Join BC \Join AC = \emptyset$, so the relations are not globally consistent.

Reduction by Semijoins
If we are to take the join of several relations, it is often efficient to first remove the dangling tuples.

- It guarantees that whatever order we join in, the result never shrinks. Thus, the total work is proportional to the output, and we can’t do more than a constant factor better than that.
- To reduce relations to globally consistent subsets, we can use the semijoin operation:
  $$ R :\leftarrow R \Join S = \pi_R (R \Join S) = R \Join (\pi_R (S)) $$
- Sometimes, semijoins don’t help eliminating dangling tuples.
  - For example, $AB$, $BC$, and $AC$ above are not changed by any semijoin.

- However, if the hypergraph is acyclic, the following algorithm produces a full reducer for a set of relations.
  - That is, the result is a set of globally join-consistent relations.

1. Pick an ear $E$ that can be consumed by hyperedge $H$. Execute the semijoin $H :\leftarrow H \Join E$.
2. Recursively generate a full reducer for the hypergraph with $E$ removed.
3. Append the semijoin $E :\leftarrow E \Join H$.

Example
Consider the relation schemas $ABC$, $ACD$, and $DE$.

- $ACD$ is an ear that is consumed by $ABC$.
- In the resulting hypergraph, $ABC$ can be consumed by $BE$.
- The full reducer:
Proof It Works

- After step (1), it is impossible for the join of the remaining hyperedges to have a tuple that doesn’t join with any tuple of $E$.
- Inductively, step (2) leaves the relations other than $E$ in a globally join-consistent state.
- Then, step (3) eliminates from $E$ any tuples that do not join with the other relations.