**1) Integrate a Unit-Preference Heuristic**

A well-known strategy in resolution is to prioritize resolving with **unit clauses** (clauses containing just one literal). The rationale is that resolving against a single literal tends to shorten the length of resolvents more dramatically, pushing you faster toward a contradiction (or a simpler clause set).

**2) Penalize Complex Unifiers (or Reward Simpler Ones)**

Sometimes you do not only care how many total literals end up in the resolvent but also **how complicated the unifier** is:

* **Count the number of variable bindings** the unifier introduces.
* **Count the arity or “nested function depth”** that appears in the resolvent.
* **Count how many variables in the unifier unify with function terms** (this can introduce more subsequent complexity).

## **3) Prioritize Shorter Resolvents More Aggressively**

Currently, you measure the sum of all literals in the new set of clauses after performing one resolution step. One simple extension is to measure not just the sum, but the *difference* in literal count before and after. Pairs that yield a large literal reduction might be more valuable than ones that barely reduce the total.

**4) Consider a Frequency-Based Literal Selection Heuristic**

In classical resolution theorem proving, a common tactic is to pick for resolution the literal(s) that appear *most frequently* or *least frequently* in the entire set of clauses, aiming either to maximize the chance of quickly deriving contradictions or to reduce duplication.

* **Most-frequent-literal heuristic**: The idea is that if a literal or predicate appears in many clauses, resolving it might help “simplify” multiple places.
* **Least-frequent-literal heuristic**: Alternatively, focusing on “rare” literals might zero in on specialized contradictions.