Rule 0.15 ⟨*Reduce Mathematical Expressions*⟩

```
[...]
contract A {
  [...]
  function f(pds) {
    [...]
    result = expr1;
    stmts
  }
  [...]
}
```

=

```
[...]
contract A′ {
  [...]
  function f(pds) {
    [...]
    result = expr2;
    stmts
  }
  [...]
}
```

where

$expr_1$ is a complex mathematical or logical expression in contract A;

$expr_2$ is the reduced form of $expr_1$ with fewer operations;

result is a variable storing the expression result;

pds are the parameter declarations of function f;

stmts represents the sequence of statements following the expression.

provided

The expressions $expr_1$ and $expr_2$ are semantically equivalent (evaluate to the same value);

$expr_2$ requires fewer operations than $expr_1$;

The reduction may apply algebraic rules such as: factoring (e.g., $a \cdot x + b \cdot x = (a + b) \cdot x$), De Morgan's laws (e.g., $\neg x \wedge \neg y \equiv \neg(x \vee y)$), distributive properties, or constant folding;

No side effects are introduced or removed by the transformation;

The reduction maintains numerical precision and overflow behavior.

Invariant:

Let s_i and s_i' be the initial state of A and A', respectively.

Let s_f and s_f' be the state reached by A and A', respectively, after $A.f()$ and $A'.f()$ are executed from s_i and s_i', respectively.

Then, the coupling invariant is

$$\forall\, s_i, s_i' \;.\; (s_i = s_i') \rightarrow (s_f = s_f')$$