

---

**Rule 0.15** *⟨Reduce Mathematical Expressions⟩*


---

<pre> [...]  <b>contract</b> A {    [...]    <b>function</b> f(pds) {      [...]      result = expr<sub>1</sub>;      stmts    }    [...]  } </pre>	=	<pre> [...]  <b>contract</b> A' {    [...]    <b>function</b> f(pds) {      [...]      result = expr<sub>2</sub>;      stmts    }    [...]  } </pre>
---	---	--

**where**

- expr*<sub>1</sub> is a complex mathematical or logical expression in contract *A*;
- expr*<sub>2</sub> is the reduced form of *expr*<sub>1</sub> 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*<sub>1</sub> and *expr*<sub>2</sub> are semantically equivalent (evaluate to the same value);
- expr*<sub>2</sub> requires fewer operations than *expr*<sub>1</sub>;
- 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

$$\forall s_i, s'_i . (s_i = s'_i) \rightarrow (s_f = s'_f)$$


---