
Rule 0.23 *(Avoid Repetitive Arithmetic Operations in Loops)*

<pre> [...] contract A { [...] function f(pds) { [...] for(init; cond; upd) { T var = expr; stmts[var] } stmts' } [...] } </pre>	=	<pre> [...] contract A' { [...] function f(pds) { [...] T var = expr; for(init; cond; upd) { stmts[var] } stmts' } [...] } </pre>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

where

expr is an arithmetic or logical expression computed inside the loop;

var is a variable of type *T* storing the result of *expr*;

stmts[*var*] represents loop body statements that use *var*;

init, *cond*, and *upd* are the loop initialization, condition, and update expressions;

pds are the parameter declarations of function *f*;

stmts' represents statements following the loop.

provided

The expression *expr* does not depend on the loop variable or any value modified within the loop;

The value of *expr* remains constant throughout all loop iterations;

Moving *expr* outside the loop does not change program semantics;

expr is side-effect free (does not modify state or call external functions);

No variable in *expr* is modified between the pre-computation and loop execution.

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

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