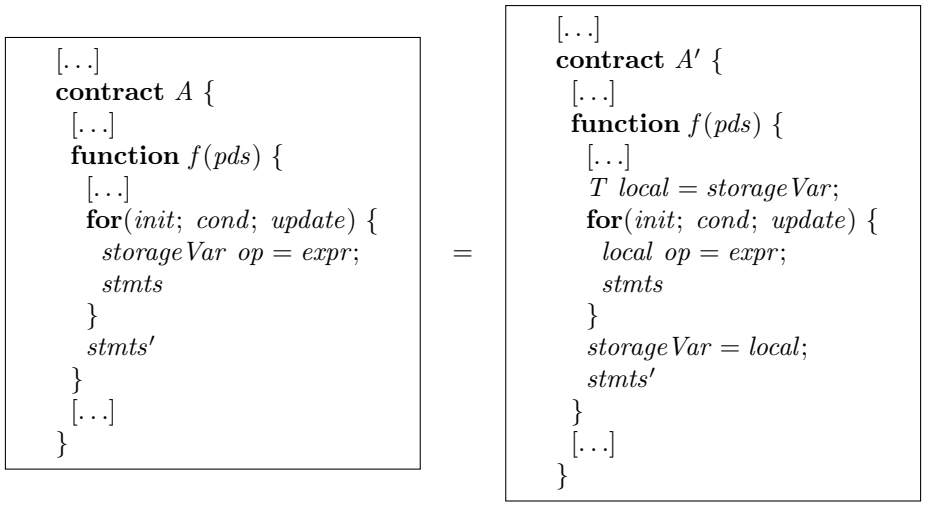
Rule 0.2 ⟨*Optimize Loops with Repeated Storage Calls*⟩



where

storageVar is a storage variable being accessed in each loop iteration;

T is the type of *storageVar*;

local is a local variable of type T used to cache the storage value;

op is a compound assignment operator (e.g., $+=, -=, *=$, *etc.*);

expr is an expression computed in each iteration;

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

stmts represents statements inside the loop body;

stmts′ represents statements following the loop.

provided

storageVar is only accessed within the loop via the compound assignment *storageVar op* = *expr*;

No external calls or state-modifying operations occur within the loop that could affect *storageVar*;

The loop does not modify *storageVar* through aliasing or indirect references;

stmts does not read or write *storageVar* except through the compound assignment.

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

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