Using types when compiling unification

- This table describes how unification is compiled in Aquarius.
- unify(X,Y) generates code for the unification X=Y using type T.
- "T $\Rightarrow$ var(X)" tests whether type T implies var(X).

Name	Condition	Actions
unify(X, Y)	var(X), var(Y)	var_var(X, Y)
	var(X), nonvar(Y)	var_nvar(X, Y)
	nonvar(X), var(Y)	var_nvar(Y, X)
	nonvar(X), nonvar(Y)	nvar_nvar(X, Y)
nvar_nvar(X, Y)		$\forall$ args X_i, Y_i: unify(X_i, Y_i)
var_nvar(X, Y)	T $\Rightarrow$ new(X)	new_old(X, Y)
	T $\Rightarrow$ ground(X)	old_old(X, Y)
	otherwise	old_old(X, Y) (depth limited)
var_var(X, Y)	T $\Rightarrow$ (old(X), old(Y))	oldv_oldv(X, Y)
	T $\Rightarrow$ (old(X), new(Y))	Generate store instruction
	T $\Rightarrow$ (new(X), old(Y))	Generate store instruction
	T $\Rightarrow$ (new(X), new(Y))	new_new(X, Y)
new_new(X, Y)		Generate store and move instructions
new_old(X, Y)	compound(Y)	W_seq(X, Y)
	atomic(Y)	Generate store instruction
	var(Y)	var_var(X, Y)
old_old(X, Y)	compound(Y), (T$\Rightarrow$nonvar(X))	Test Y type, then old_old_R(X, Y)
	compound(Y)	Generate switch, R & W branches
	atomic(Y), (T $\Rightarrow$ nonvar(X))	old_old_R(X, Y)
	atomic(Y)	Generate unify_atomic instruction
	nonvar(Y), (T $\Rightarrow$ var(X))	old_old_W(X, Y)
	var(Y)	var_var(X, Y)
oldv_oldv(X, Y)	A = atomic_value(T, X)	unify(Y, A)
	A = atomic_value(T, Y)	unify(X, A)
	T $\Rightarrow$ (atomic(X), atomic(Y))	Generate comparison instruction
	T $\Rightarrow$ (var(X), nonvar(Y))	Generate store instruction
	T $\Rightarrow$ (nonvar(X), var(Y))	Generate store instruction
	otherwise	Generate unify instruction
old_old_W(X, Y)	compound(Y)	W_seq(X, Y)
	atomic(Y)	Generate store instruction
old_old_R(X, Y)	compound(Y)	$\forall$ args X_i, Y_i: old_old(X_i, Y_i)
	atomic(Y)	Generate comparison instruction
W_seq(X, Y)		Generate W mode sequence