AVM representation and DAG representation.

An AVM might look like this:

```
0   AGR 1 | NUMBER 2  SG |
    SUBJ 3 |          AGR 1 |
```

The corresponding DAG would look like this:

```
AGR   1   NUMBER   2
   ^     ^
   |     |
   0   AGR

SUBJ 3
```

The same structure, “expanded” for the unification algorithm:

```
0   PTR NULL   CT 4
    AGR 1   CT 5   PTR NULL
    SUBJ 3   PTR NULL
```

```
PTR NULL   CT 5   PTR NULL
AGR 1   NUMBER 2   CT 6   SG
```
The expanded structure as a DAG:

The expanded representations for the other argument of example (11.23)
Stepping through the algorithm:

Unify(f1 = 0, f2 = 10)
f1-real = 0, f2-real = 10 neither node has a ptr

Neither structure is null, they aren’t identical, so
10.PTR ← 0 make f2 point to f1

Loop through f2-real’s features
feature = 10’s SUBJ
other-feature = 0’s SUBJ

Unify(f1 = 11, f2 = 3)
f1-real = 11, f2-real = 3

Neither pointer is null and they aren’t identical
3.PTR ← 11

Loop through f2-real’s features
feature = 3’s AGR
other-feature = 11’s AGR

Unify(f1 = 8, f2 = 12)
f1-real = 1, f2-real = 12 8 has a pointer

Neither is null and they aren’t identical
12.PTR ← 8

Loop through f2-real’s features
feature = 12’s PERSON
other-feature ← 1’s PERSON  Make a new (19)

Unify(f1 = 13, f2 = 19)
f1-real = 13, f2-real = 19

f2-real is null
19.PTR ← 13
return 13

return 8

return 11

return 0