I tried to work through this rewriting approach by hand for the simplest imaginable "real" example -- simplog. It's not working. We are indeed going to need to be more clever about how we handle bindings across subgoals. In the "brainstorming" section below, I tried to suss this out but can't, even for this one example (let alone a general rule). Let me know if this isn't making sense.

// "if someone has a log, and knows about someone else who doesn't….."
missing_log(A, Pl) :- log(X, Pl), node(X, A), notin log(A, Pl);
⇒

// problem 1: no bindings given for "A"
Bindings from forward execution : notin missing_log(_, "hello", 4) => dom_missing_log_Pl(hello)

^Hmmm...so post does not inform a domain for A since the first attribute of the missing_log subgoal in the post rule is a wildcard. One way to handle such a case is snatching domain info out of the missing_log relation after running P. Alternatively, since the values of A derive from node(X,A) in the missing_log rule, the domain of A could be the range of values in the second attribute of node.

A corresponding rule set for the latter approach would be :
not_missing_log(A, Pl) :- notin log(_, Pl), dom_missing_log_post_pl(Pl), dom_missing_log_node_att_a(A)
not_missing_log(A, Pl) :- notin node(_, Pl), dom_missing_log_post_pl(Pl), dom_missing_log_node_att_a(A)
not_missing_log(A, Pl) :- log(A, Pl), dom_missing_log_post_pl(Pl), dom_missing_log_node_att_a(A)

I'll compile an example graph and send it along in a bit.

// this will only fire if log has *no* entries with string hello.
not_missing_log(?, Pl) :- notin log(_, Pl), dom_missing_log_Pl(Pl)

// this tells us nothing.
not_missing_log(A, Pl) :- notin node(_, A), dom_missing_log_Pl(Pl)

// this fires too eagerly! True if a log exists with string "hello" -- between this and the first rule,
// not_missing_log() will always be made true.
not_missing_log(?, Pl) :- log(_, Pl), dom_missing_log_Pl(Pl)

Brainstorm country:

// "if someone has a log, and knows about someone else who doesn't….."
missing_log(A, Pl) :- log(X, Pl), node(X, A), notin log(A, Pl);

The real de-morgans-ification would invert the whole statement; something like:

"If no one has a log, or if someone has a log but does not know someone else who doesn't"

// no one has a log
not_missing_log(?, Pl) :- notin log(_, Pl), dom_missing_log_Pl(Pl);
// someone has a log, but she doesn't know anyone

```
not_missing_log(?, Pl) :- log(X, Pl), notin node(X, _);

// dang it, this last one isn't going to work since what we really want to do is universally quantify over all node.A.
not_missing_log(?, Pl) :- log(X, Pl), node(X, A),  /* ugh, not right */ log(A, Pl);
```