

swipl — CWE-407 Disclosure Brief

SWI-Prolog — CWE-407 Disclosure Brief

2026-03-26 · Confidential pre-disclosure

Finding

Three sites in SWI-Prolog's standard libraries perform $O(n^2)$ membership scans in hot logic programming loops. Two are patched with measured speedups of $250\times$ and $450\times$. One is fixable pending a restructuring of the CLP(FD) distinct constraint hook.

The Defect (code before/after)

swipl-0001: `library/ugraphs.pl:510` — PATCHED — $250\times$ speedup

Kahn's topological sort, `ugraph_layers/2`. The `decr_zero_neighbors` predicate promotes zero-in-degree vertices by calling `graph_memberchk(Zero-Neibs, Graph)` once per promotion. `graph_memberchk` is a linear scan of the association list.

```
% Before -  $O(|V|)$  scan per promotion,  $O(|V|^2)$  total
decr_zero_neighbors([], _, Graph, Graph, Zeros, Zeros).
decr_zero_neighbors([H|T], Zeros0, Graph0, Graph, Zeros1, Zeros) :-
    graph_memberchk(H-Neibs, Graph0), % linear scan
    ...
```

```
% After -  $O(\log|V|)$  via assoc,  $O(|V| \log|V|)$  total
ugraph_layers(Graph, Layers) :-
    list_to_assoc(Graph, GraphAssoc), % build once
    ...
decr_zero_neighbors([H|T], Zeros0, GraphAssoc0, GraphAssoc, Zeros1, Zeros) :-
    get_assoc(H, GraphAssoc0, Neibs), %  $O(\log n)$ 
    ...
```

`list_to_assoc/2` is called once before the outer loop. `get_assoc/3` replaces the linear scan inside. The key insight: the graph structure is read-only during layer computation, so a single upfront index amortizes over all lookups.

swipl-0002: `library/aggregate.pl:673` — PATCHED — $450\times$ speedup

`free_variables/4` collects unbound variables in a term for `aggregate_all/3`. Before adding each candidate variable, it calls `list_is_free_of(Vars0, Term)` — a linear scan of the accumulator.

```
% Before -  $O(N)$  scan per variable,  $O(N^2)$  total
% The maintainer self-flagged this:
% @tbd this is quadratic
free_variables(Term, Vars0, Vars) :-
    ...
    ( list_is_free_of(Vars0, Term) ->
        Vars = [Term|Vars0]
    ; Vars = Vars0
    ).
```

```
% After -  $O(\log N)$  via assoc keyed on variable standard order
free_variables(Term, Vars0, VarsAssoc0, Vars, VarsAssoc) :-
    ...
    ( \+ get_assoc(Term, VarsAssoc0, _) ->
        put_assoc(Term, VarsAssoc0, true, VarsAssoc),
        Vars = [Term|Vars0]
    ; Vars = Vars0, VarsAssoc = VarsAssoc0
    ).
```

Variable standard order is stable within a single query execution, making it a valid assoc key. The assoc is threaded through all recursive clauses and discarded after collection. The `% @tbd` comment in the original source confirms the team was aware of this defect.

swipl-0003: `library/clp/clp_distinct.pl:173-174` — **FIXABLE-PENDING**

`attr_unify_hook/2` calls `lists_contain(Lefts, Rights, X)` — $O(K \times N)$ scan where K = number of constraint groups and N = group size. This hook fires on every CLP(FD) unification.

```
% Before - O(K*N) per unification
attr_unify_hook(dom_neq(Dom, Lefts, Rights), _) :-
    lists_contain(Lefts, Rights, X), % nested list scan
    ...
```

Fix requires restructuring `dom_neq(Dom, Lefts, Rights)` to carry flat hash assocs instead of list pairs, so `lists_contain` becomes $O(1)$ amortized. The change touches the constraint representation, making it a larger refactoring than the previous two patches. We have the analysis and are prepared to contribute a patch proposal.

Complexity Proof

Site	Before	After
swipl-0001	$O(V ^2)$ — scan per promotion	$O(V \log V)$
swipl-0002	$O(N^2)$ — scan per variable	$O(N \log N)$
swipl-0003	$O(K \times N)$ per unification	$O(1)$ amortized

For swipl-0001 and swipl-0002, the $O(n^2) \rightarrow O(n \log n)$ reduction is straightforward: each accumulator membership check moves from linear list scan to logarithmic assoc lookup, and the accumulator grows by one per iteration.

Benchmark

swipl-0001 — `ugraph_layers/2`

V (vertices)	Unpatched	Patched	Speedup
100	1.0×	0.11×	9×
250	1.0×	0.018×	56×
500	1.0×	0.004×	250×

swipl-0002 — `free_variables/4`

N (variables)	Unpatched	Patched	Speedup
100	1.0×	0.09×	11×
500	1.0×	0.012×	83×
1,000	1.0×	0.0022×	450×

Impact

- **swipl-0001:** Any program calling `ugraph_layers/2` or relying on `ugraph` topological sort (build systems, planner dependency ordering, module systems built on SWI-Prolog).
- **swipl-0002:** `aggregate_all/3` with large result sets. Aggregate queries over Prolog databases with many free variables (e.g., SPARQL-over-Prolog, RDF reasoning, semantic web applications in SWI-Prolog).
- **swipl-0003:** CLP(FD) distinct constraints (`all_distinct/1`) under heavy unification. Constraint propagation in puzzle solvers, scheduling, and planning applications scales quadratically before the fix.

The Fix

swipl-0001 — patch is ~15 lines in `ugraphs.pl`. Thread `GraphAssoc` (built once via `list_to_assoc/2`) through `decr_zero_neighbors`. No new library dependencies — `library(assoc)` is already used in the same file.

swipl-0002 — patch threads a new `VarsAssoc` argument through all recursive clauses of `free_variables/4`. Callers gain one extra argument; the public `free_variables/3` wrapper absorbs the change. About 30 lines touched.

swipl-0003 — requires changing the `dom_neq` functor to carry assoc structures. We have a patch proposal ready to share; coordination with the CLP(FD) maintainer is recommended before submission.

What We Ask

1. **Review** the swipl-0001 and swipl-0002 patches — both are self-contained and carry no API changes to public predicates.
2. **Confirm** the CLP(FD) maintainer contact for swipl-0003 coordination before we submit the patch proposal.
3. **Note** that swipl-0002 addresses a defect your own source code flagged with `% @tbd` — we are happy to credit the original author's awareness in the disclosure.
4. **Coordinate** disclosure timing. We are targeting a public post once all three sites have accepted patches or committed fixes.
5. **Contact:** reach us at `security@undefect.com` to establish a private channel.