Algorithm 1 Rewriting algorithm

```
1: function APPLY_MODEL(tree, model)
2:     rewrite_ht ← new hashtable
3:     compute_rewrite_rules(tree, model, rewrite_ht)
4:     new_tree ← rewrite_tree(tree, rewrite_ht)
5:     return new_tree
6: end function
7:
8: function COMPUTE_REWRITE_RULES(tree, model, rewrite_ht)
9:     if tree.sequent.is_leaf then
10:        compute_rewrite_sequent(tree.sequent, model, rewrite_ht)
11:    else
12:        for each premise p in tree.premisses do
13:            compute_rewrite_rules(p, model, rewrite_ht)
14:        end for
15:        compute_rewrite_sequent(tree.sequent, model, rewrite_ht)
16:    end if
17: end function
18:
19: function COMPUTE_REWRITE_SEQUENT(sequent, model, rewrite_ht)
20:    Let K be the set of keys in rewrite_ht
21:    for each Γ in sequent.contexts do
22:        if sequent.is_leaf then
23:            for each constraint cstr where Γ is found in model do
24:                if cstr = EMP (Γ) then
25:                    if Γ was not rewritten yet then
26:                        rewrite_ht.add Γ ([·], [·])
27:                    end if
28:                end if
29:                if cstr = IN (F, Γ, n) then
30:                    if Γ was not rewritten yet then
31:                        if sequent.is_closed_leaf and Γ is bounded then
32:                            rewrite_ht.add Γ ([·], [F_1, ..., F_n])
33:                        else
34:                            rewrite_ht.add Γ ([Γ_k], [F_1, ..., F_n])
35:                            where Γ_k is a fresh context variable
36:                        end if
37:                    else
38:                        Let ([Γ_1, ..., Γ_i], [t_1, ..., t_n]) be the rewriting of Γ
39:                        if F is different from every t in [t_1, ..., t_n] then
40:                            rewrite_ht.replace Γ ([Γ_1, ..., Γi], [t_1, ..., t_n, F])
41:                        end if
42:                    end if
43:                end if
44:            end for
```

Algorithm 2 Rewriting algorithm (continuation)

```
45:         else
46:             for each constraint cstr where Γ is found in model do
47:                 if cstr = EMP (Γ) then
48:                     if Γ was not rewritten yet then
49:                         rewrite_ht.add Γ ([·], [·])
50:                     else if rewriting (Γ) is different from ([·], [·]) then
51:                         Let ([Γ_1, ..., Γi], [·]) be the rewriting of Γ
52:                         for each Γ_k in [Γ_1, ..., Γi] do
53:                             rewrite_ht.replace Γk ([·], [·])
54:                         end for
55:                     end if
56:                 end if
57:                 if cstr = UNION (Γ_1, Γ_2, Γ) then
58:                     if Γ was not rewritten yet then
59:                         if Γ_1 is not a key in K then
60:                             rewriting (Γ_1) ← ([Γ_1], [·])
61:                         end if
62:                         if Γ_2 is not a key in K then
63:                             rewriting (Γ_2) ← ([Γ_2], [·])
64:                         end if
65:                         rewrite_ht.add Γ (rewriting (Γ_1) + rewriting (Γ_2))
66:                     end if
67:                 end if
68:                 if cstr = SETMINUS (Γ_0, F, Γ) then
69:                     if Γ was not rewritten yet then
70:                         rewrite_ht.add Γ (rewriting (Γ_0) − F)
71:                     else
72:                         This is the case where rewriting (Γ) and rewriting (Γ_0)
73:                         must have exactly one context variable each, so
74:                         let ([Γ_1], [t_1, ..., t_i]) be the rewriting of Γ
75:                         and ([Γ_1′], [t_1′, ..., t_j′]) be the rewriting of Γ_0
76:                         if [t_1′, ..., t_j′] = ([t_1, ..., t_i] − F) then
77:                             if Γ_1 > Γ_1′ then
78:                                 for each Γ_k in K do
79:                                     Replace every occurrence of Γ_1′
80:                                     with Γ_1 in rewriting (Γ_k)
81:                                 end for
82:                             else
83:                                 for each Γ_k in K do
84:                                     Replace every occurrence of Γ_1
85:                                     with Γ_1′ in rewriting (Γ_k)
86:                                 end for
87:                             end if
88:                         end if
89:                     end if
90:                 end if
91:             end for
92:         end if
93:     end for
94: end function
```