Integrating a node in a dag

```
procedure integrate(n1, p, n2, D)
input: dag D, nodes n1, n2 in D, atom p.
begin
  if D contains a node n labelled by p such that n1 = left(n)
                                            and n2 = right(n)
    then return n
    else
      add to D a node n labelled by p and arcs from n to n1 and n2
                              labelled by 0 and 1 respectively
      return n
end
```

Building OBDDs

procedure $obdd(A)$

input: propositional formula A

parameters: a dag D containing N_0 and N_1

output: a node n in (modified) D which represents A

begin

A := $simplify(A)$

if $A = \bot$ then return N_0

if $A = \top$ then return N_1

p := $max\_atom(A)$

n_1 := $obdd(A_p^\bot)$

n_2 := $obdd(A_p^\top)$

if $n_1 = n_2$ then return n_1

return $integrate(n_1, p, n_2, D)$

end

Composing OBDDs

```
procedure compose(A, q_1, ..., q_m, n_1, ..., n_m)
parameters: a dag D containing n_1, ..., n_m and N_0, N_1 as nodes
input: propositional formula A(q_1, ..., q_m) of atoms q_1, ..., q_m
       nodes n_1, ..., n_m representing A_1, ..., A_m in D
output: a node n representing A(A_1, ..., A_m) in (modified) D
begin
  A := simplify(A)
  if A = ⊥ then return N_0
  if A = ⊤ then return N_1
  if some n_i is a leaf labelled 0 then
    return compose(A^⊥_{q_i}, q_1, ..., q_{i-1}, q_{i+1}, ..., q_m, n_1, ..., n_{i-1}, n_{i+1}, ..., n_m)
  if some n_i is a leaf labelled 1 then
    return compose(A^⊤_{q_i}, q_1, ..., q_{i-1}, q_{i+1}, ..., q_m, n_1, ..., n_{i-1}, n_{i+1}, ..., n_m)
  p := max_atom(n_1, ..., n_m)
  forall i = 1 ... m
    if n_i is labelled by p
      then (l_i, r_i) := (left(n_i), right(n_i))
      else (l_i, r_i) := (n_i, n_i)
  k_1 := compose(A, q_1, ..., q_m, l_1, ..., l_m)
  k_2 := compose(A, q_1, ..., q_m, r_1, ..., r_m)
  if k_1 = k_2 then return k_1
  return integrate(k_1, p, k_2, D)
end
```

Special case: disjunction

```
procedure disjunction(n_1, ..., n_m)
parameters: a dag D containing n_1, ..., n_m and N_0, N_1 as nodes
input: nodes n_1, ..., n_m representing A_1, ..., A_m in D
output: a node n representing A_1 ∨ ... ∨ A_m in (modified) D
begin
  if m = 1 then return n_1
  if some n_i is a leaf labelled 0 then
    return disjunction(n_1, ..., n_{i-1}, n_{i+1}, ..., n_m)
  if some n_i is a leaf labelled 1 then return N_1
  p := max_atom(n_1, ..., n_m)
  forall i = 1 ... m
    if n_i is labelled by p
      then (l_i, r_i) := (left(n_i), right(n_i))
      else (l_i, r_i) := (n_i, n_i)
  k_1 := disjunction(l_1, ..., l_m)
  k_2 := disjunction(r_1, ..., r_m)
  if k_1 = k_2 then return k_1
  return integrate(k_1, p, k_2, D)
end
```