Integrating a node in a dag

procedure *integrate*(n_1, p, n_2, D)

parameters: global dag D

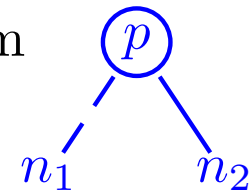
input: nodes n_1, n_2 in D representing formulas F_1, F_2, variable p

output: node n in (modified) D representing *if* p *then* F_2 *else* F_1

begin

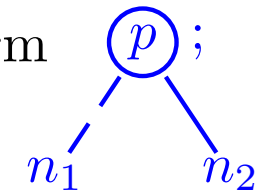
if $n_1 = n_2$ then return n_1;

if D contains a node n having the form



then return n;

add to D a new node n of the form p;



return n

end

Building OBDDs

procedure $obdd(F)$

input: propositional formula F

parameters: global dag D; total order on variables

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

begin

F := $simplify(F)$

if $F = \bot$ then return $\boxed{0}$

if $F = \top$ then return $\boxed{1}$

p := $max\_atom(F)$

n_1 := $obdd(F_p^{\bot})$

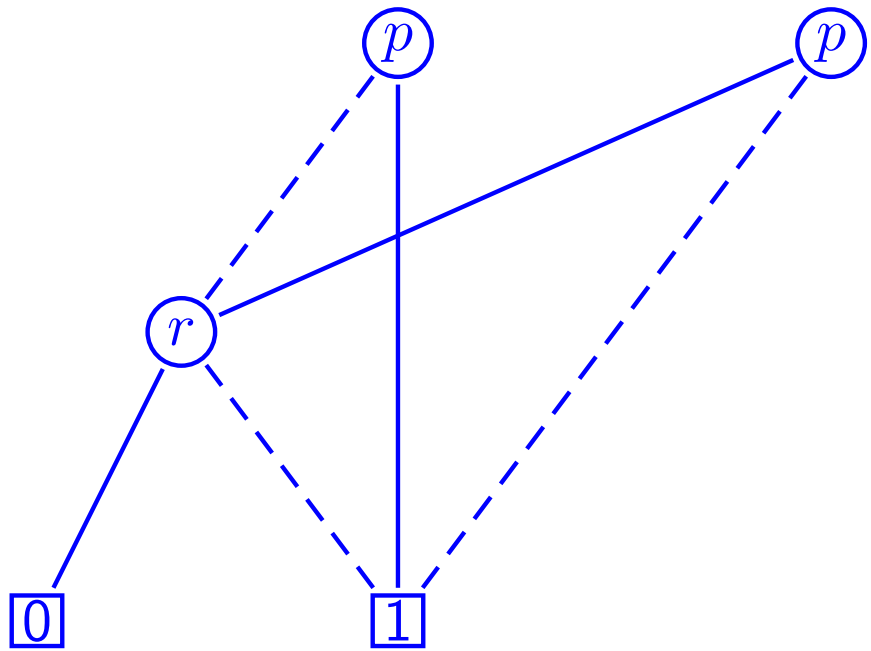
n_2 := $obdd(F_p^{\top})$

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

end

Example

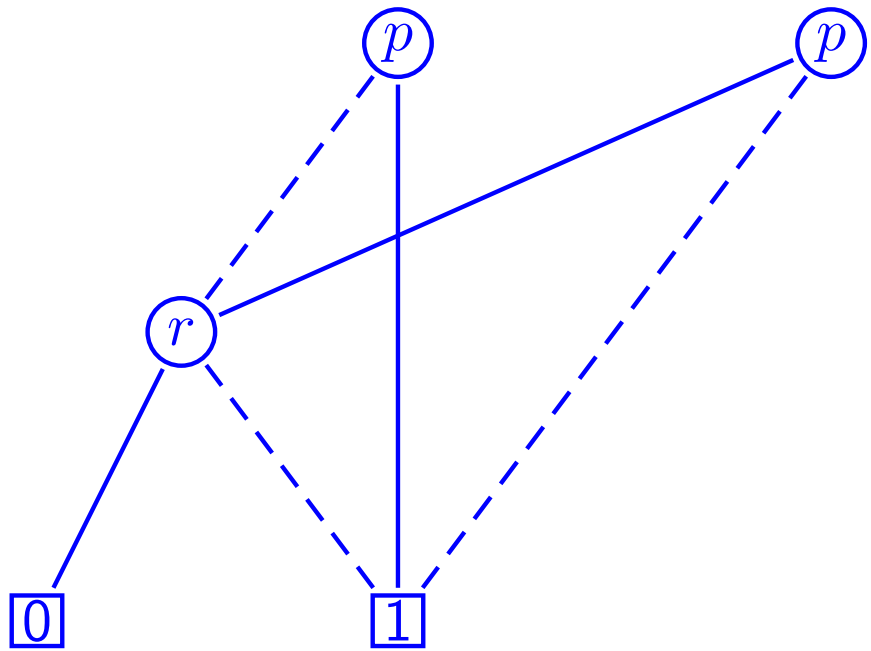
$obdd((q \to p) \wedge r \to (p \leftrightarrow r) \wedge q)$; order $p > q > r$



Example

(p) $\quad (q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$

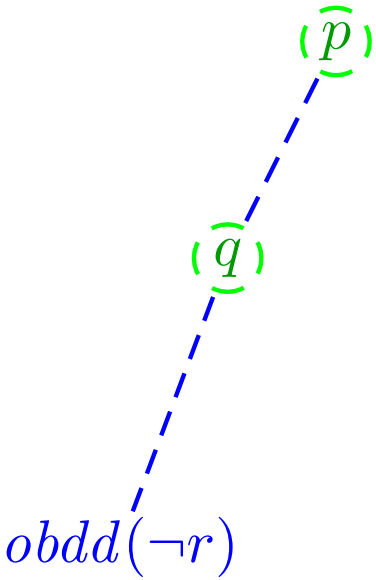
$obdd(\neg q \wedge r \to \neg r \wedge q)$

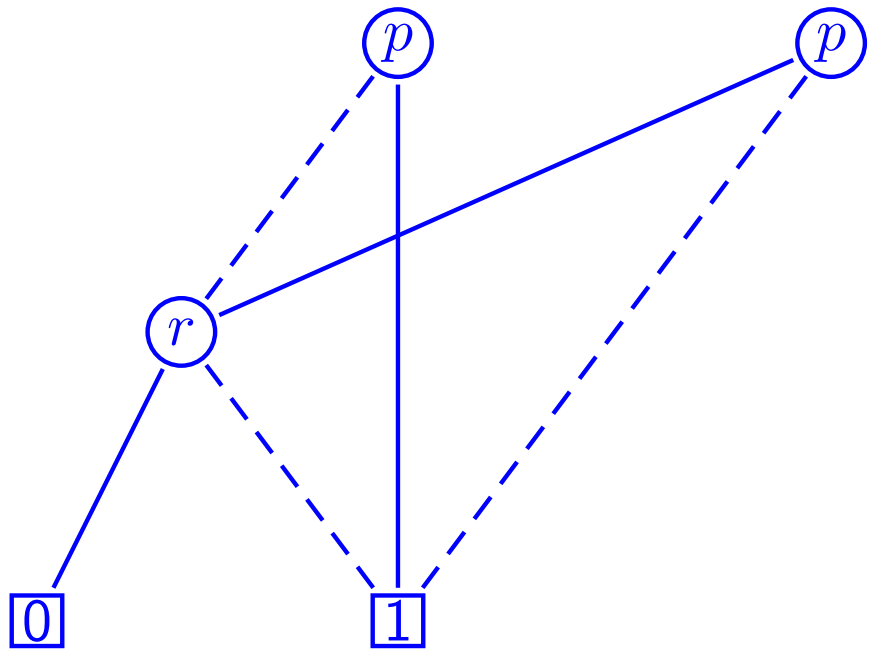


Example

$(q \rightarrow p) \wedge r \rightarrow (p \leftrightarrow r) \wedge q$

$\neg q \wedge r \rightarrow$
$\neg r \wedge q$

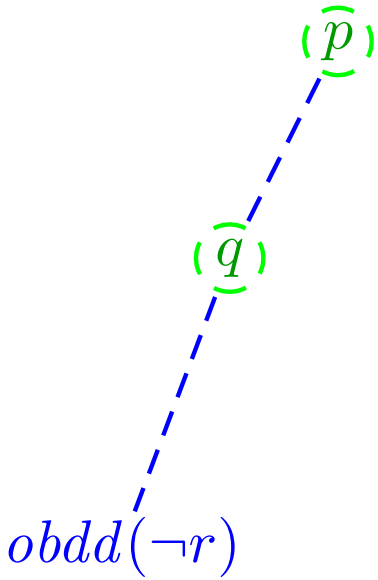


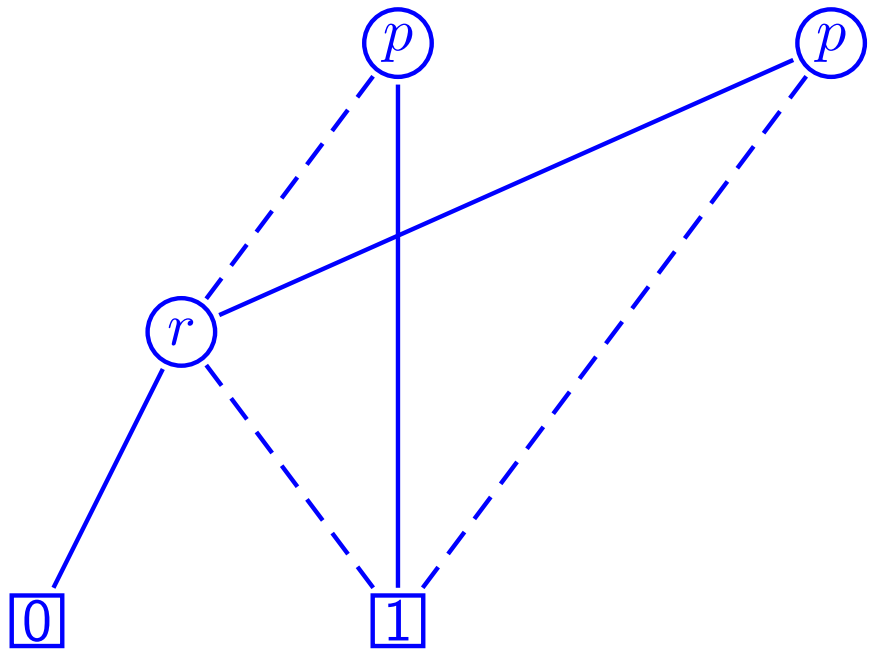


Example

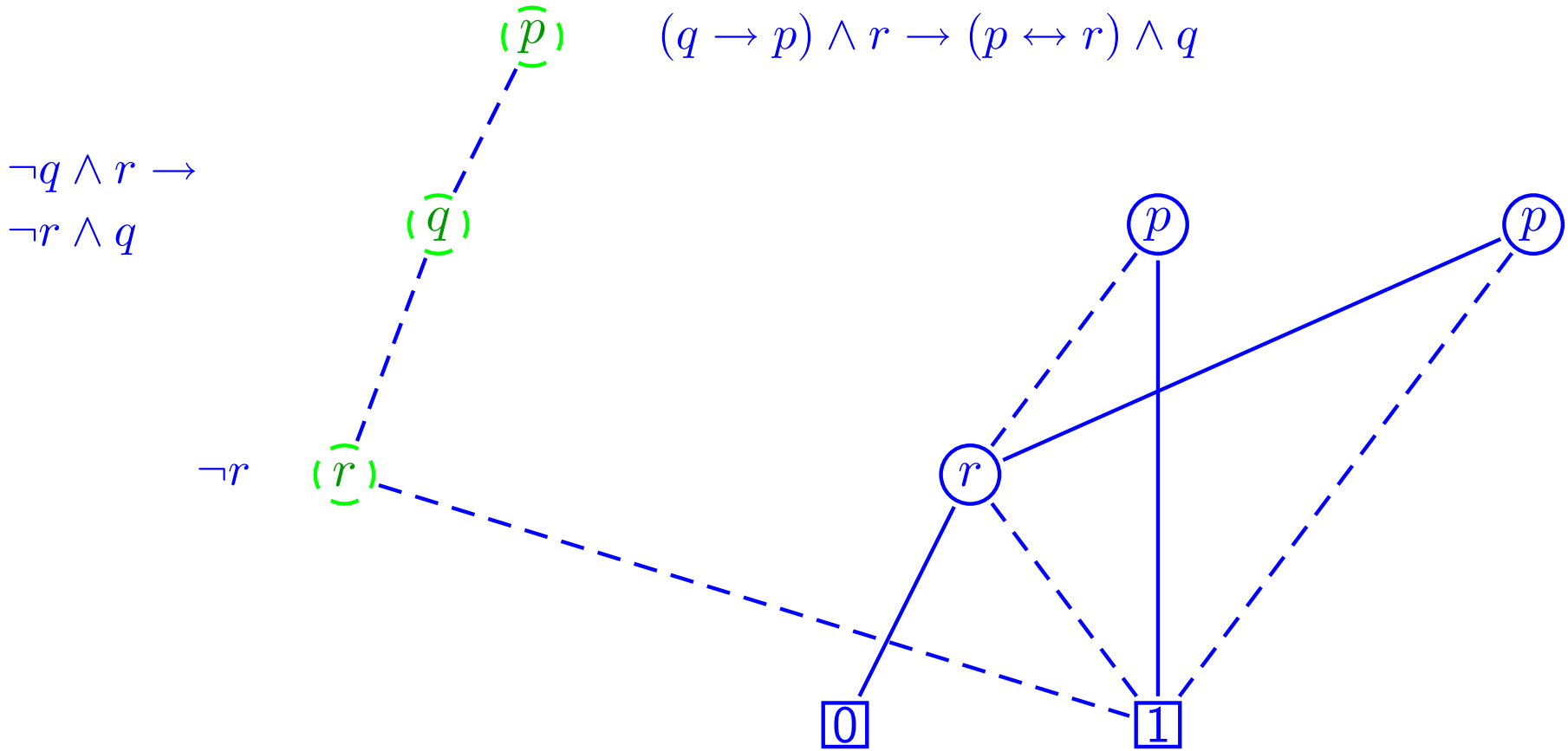
$(q \rightarrow p) \wedge r \rightarrow (p \leftrightarrow r) \wedge q$

$\neg q \wedge r \rightarrow$
$\neg r \wedge q$





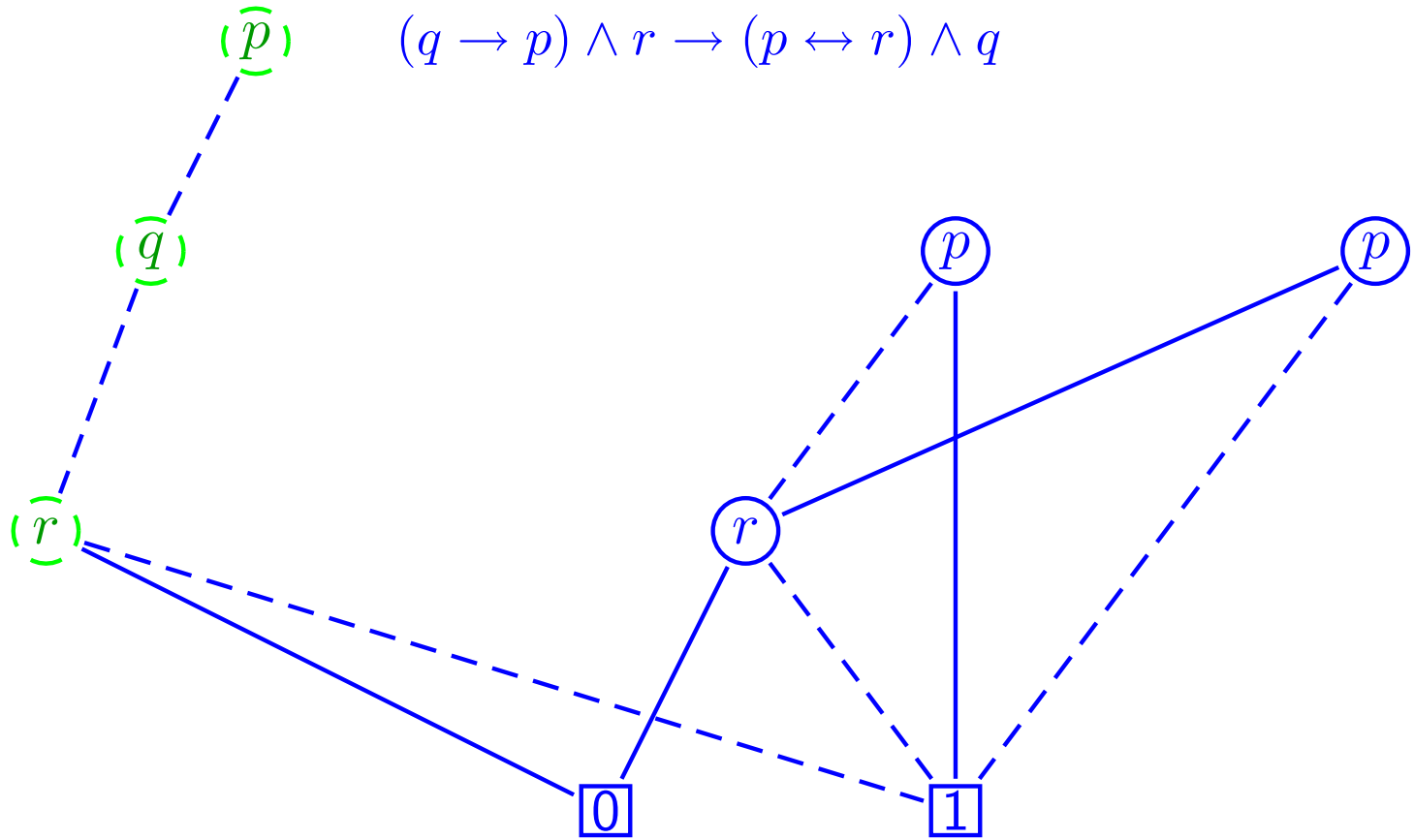
Example


$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$
$\neg q \wedge r \to$
$\neg r \wedge q$
$\neg r$
p
q
r
p
r
p
0
1

Example

$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$

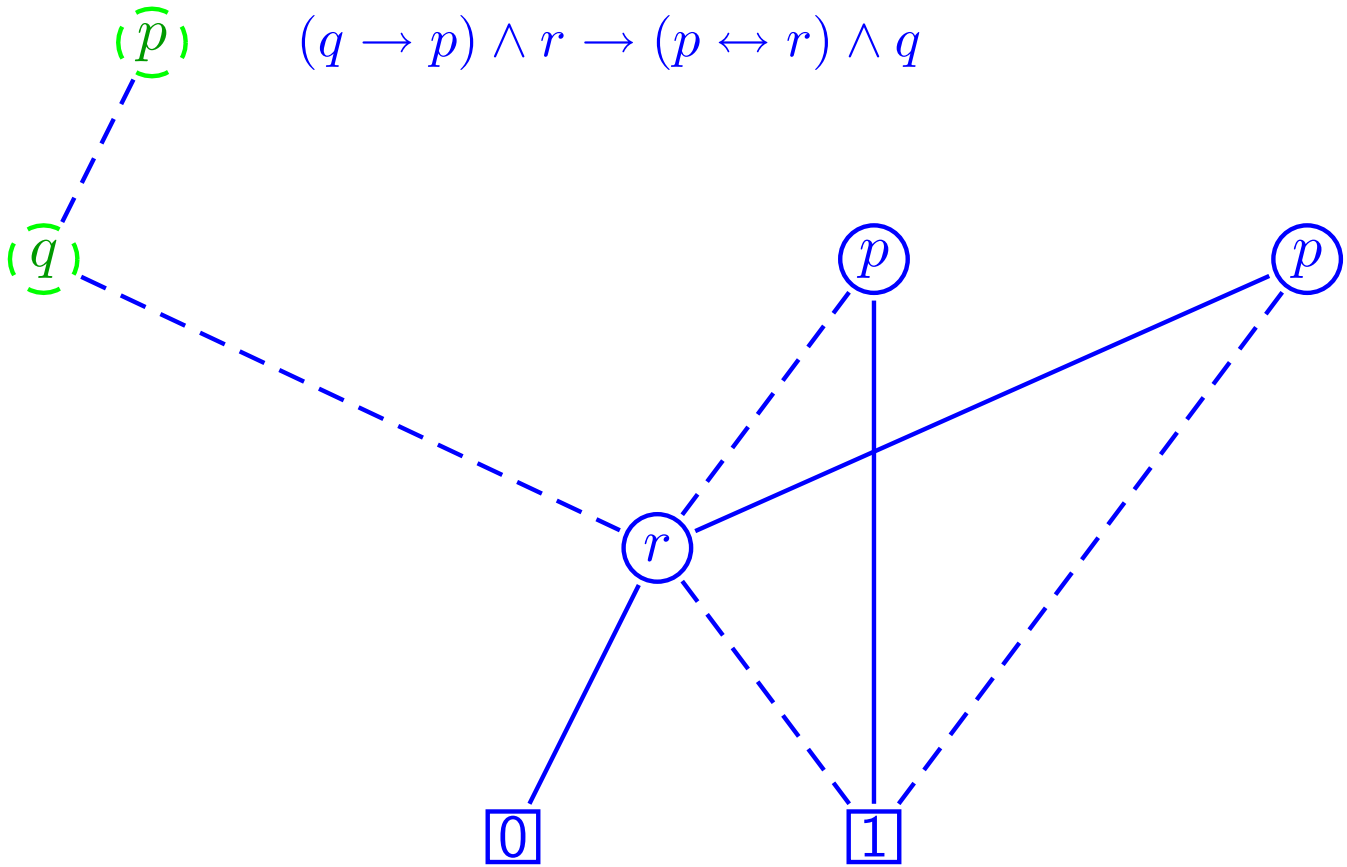
$\neg q \wedge r \to$
$\neg r \wedge q$



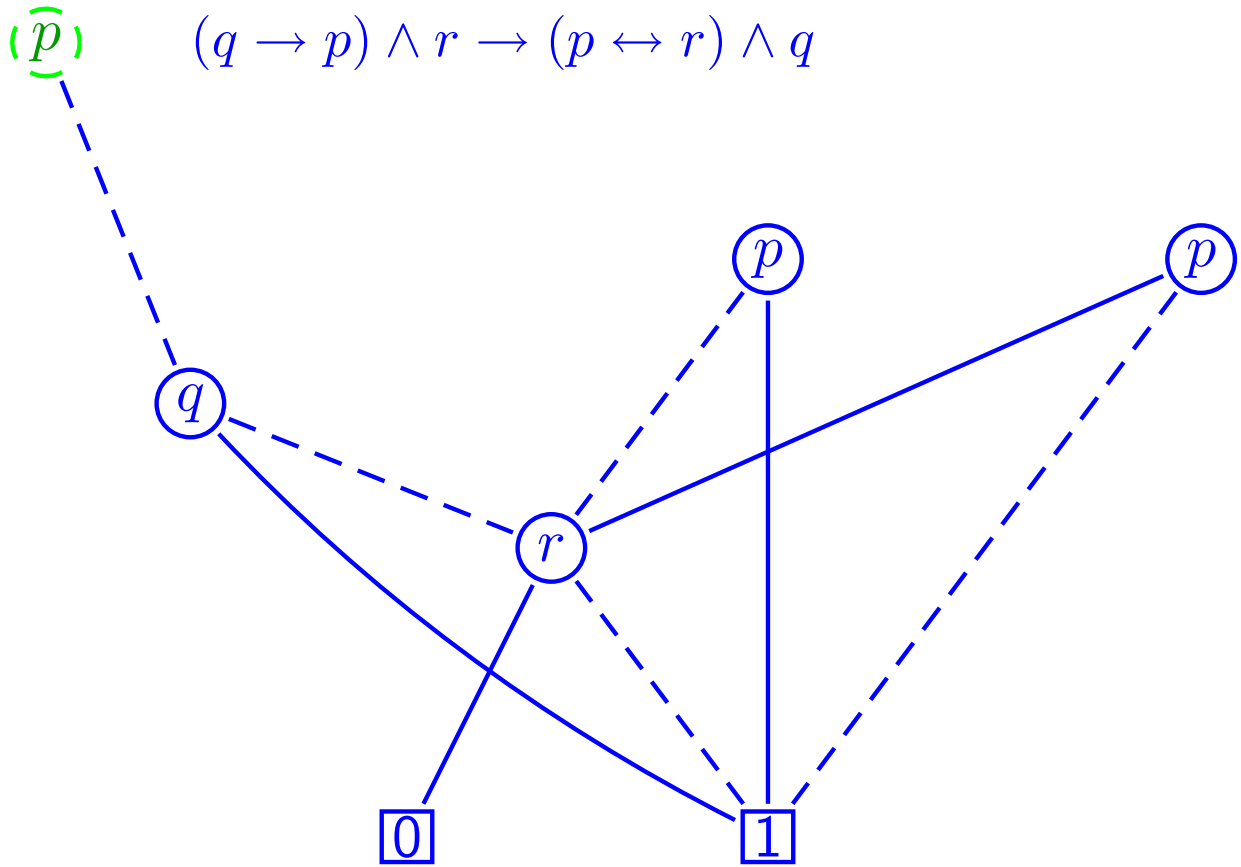
Example

$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$

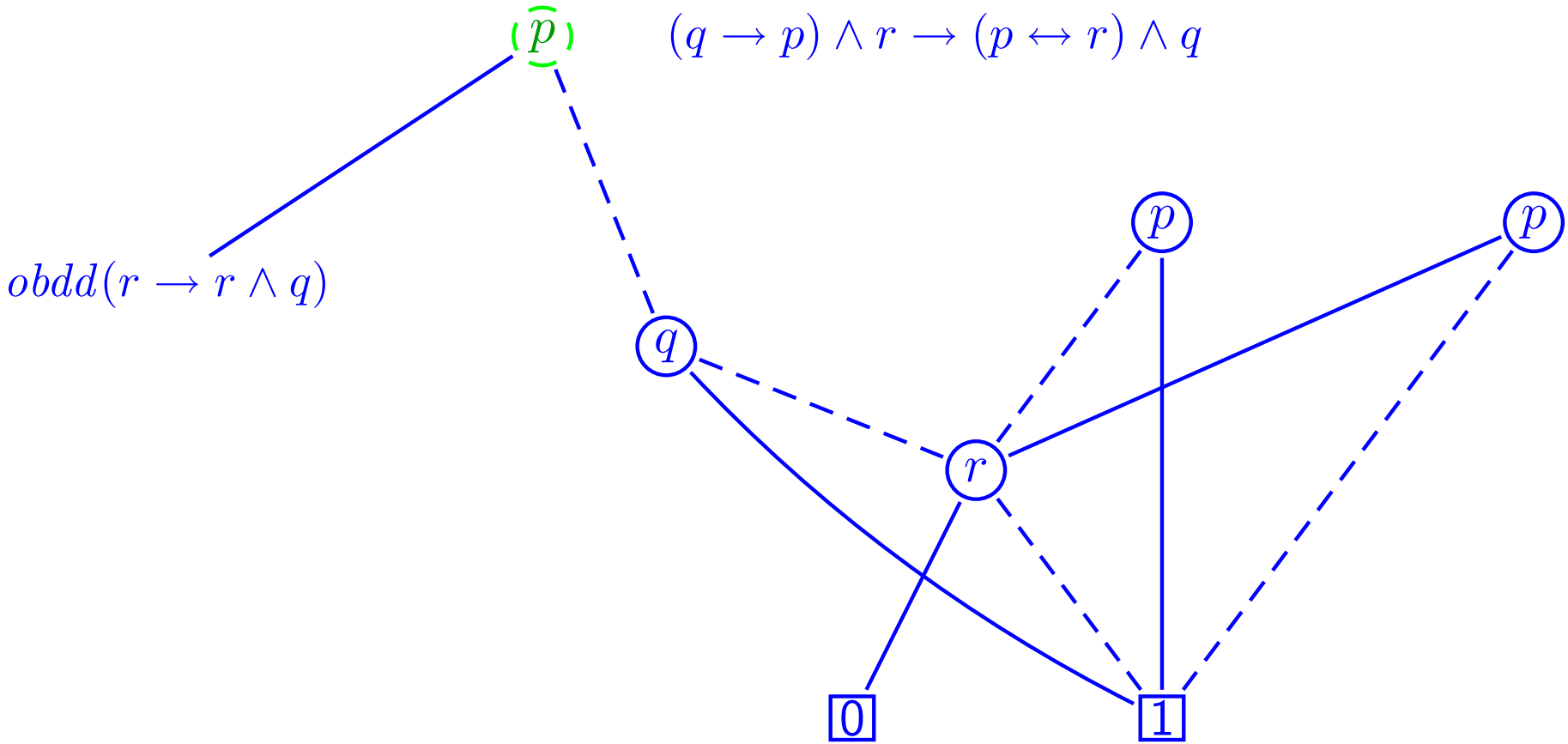
$\neg q \wedge r \to$
$\neg r \wedge q$



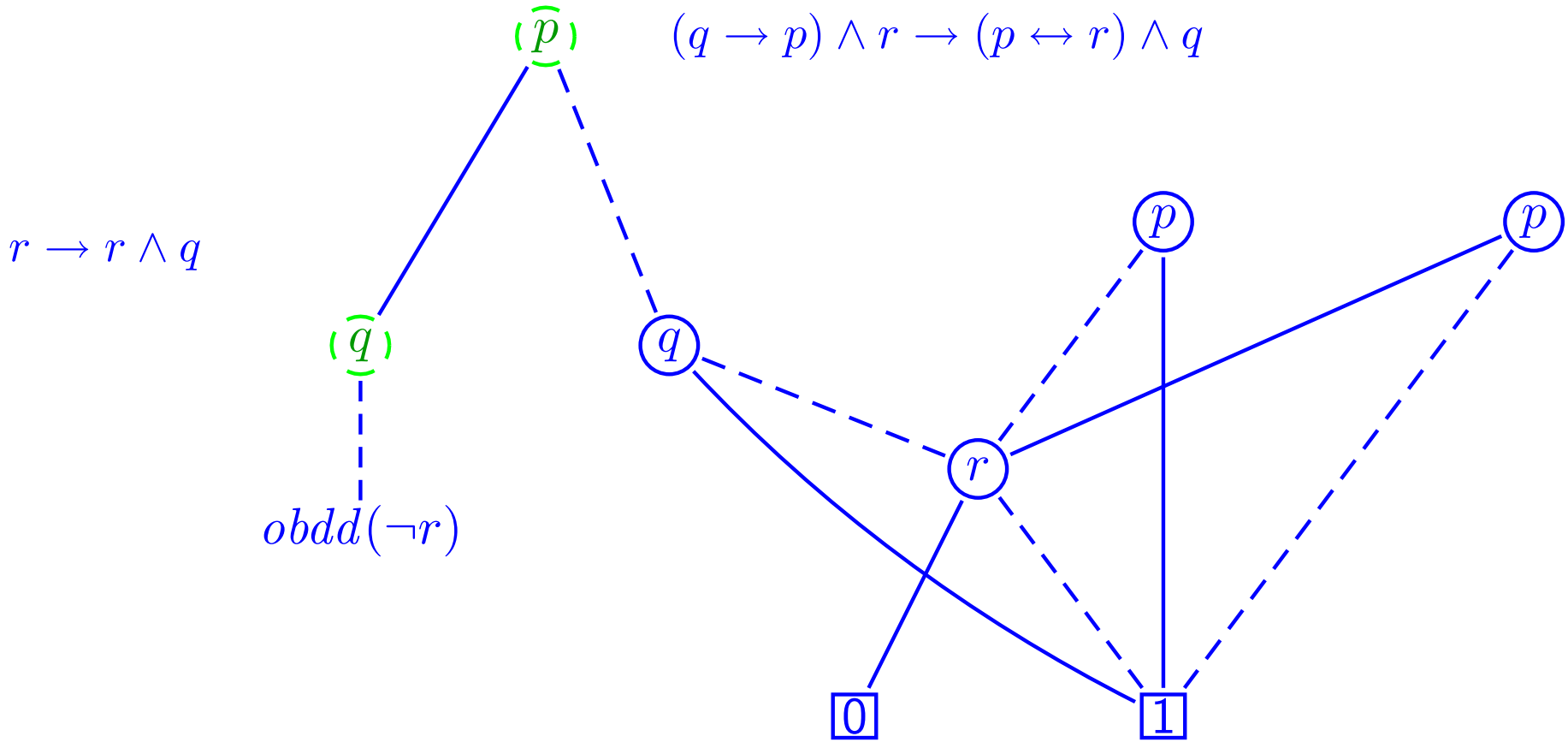
Example



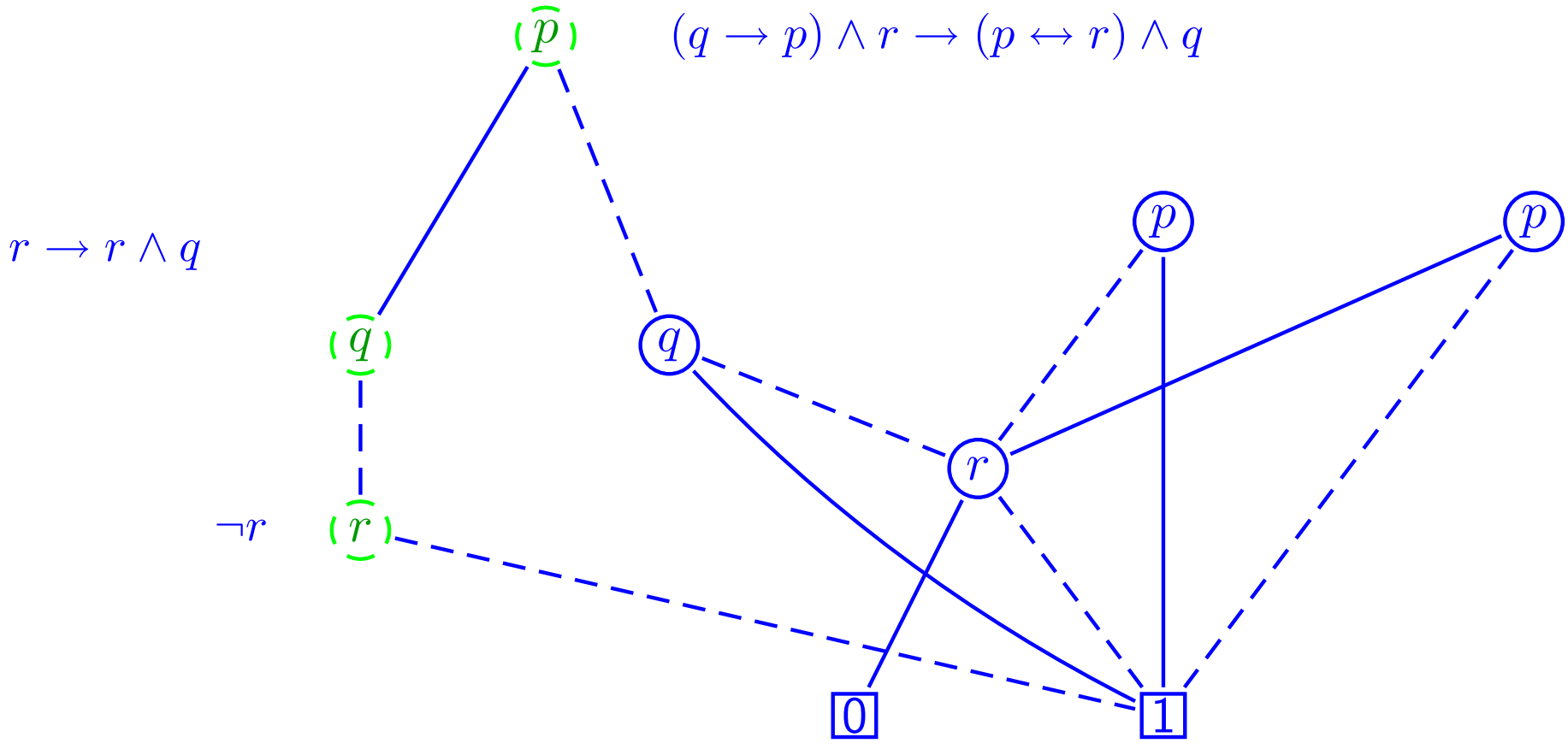
Example


$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$
$obdd(r \to r \wedge q)$
p
q
r
p
p
0
1

Example

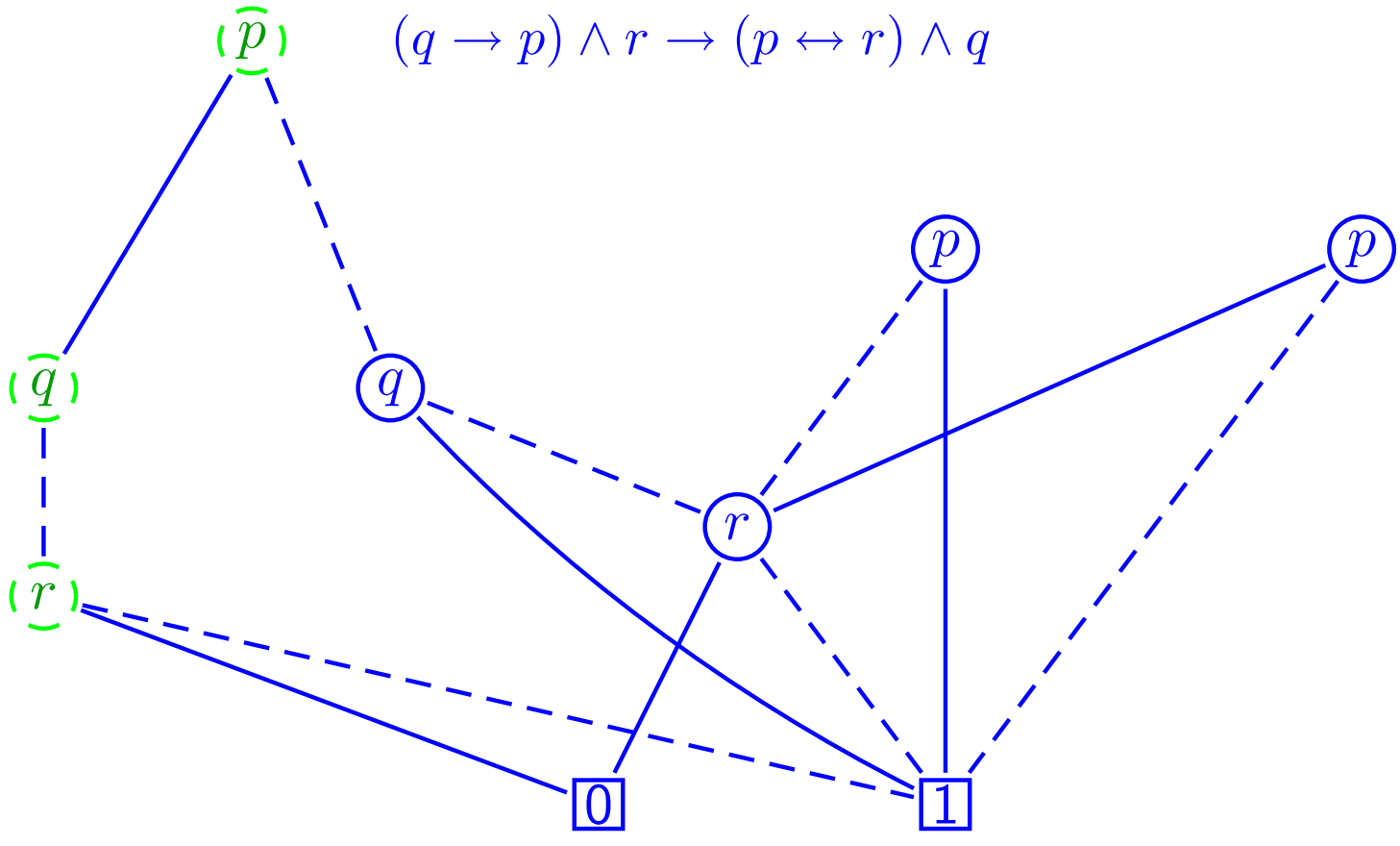

$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$
$r \to r \wedge q$
p
q
$obdd(\neg r)$
p
q
r
p
0
1

Example


$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$
$r \to r \wedge q$
$\neg r$
p
q
r
0
1

Example

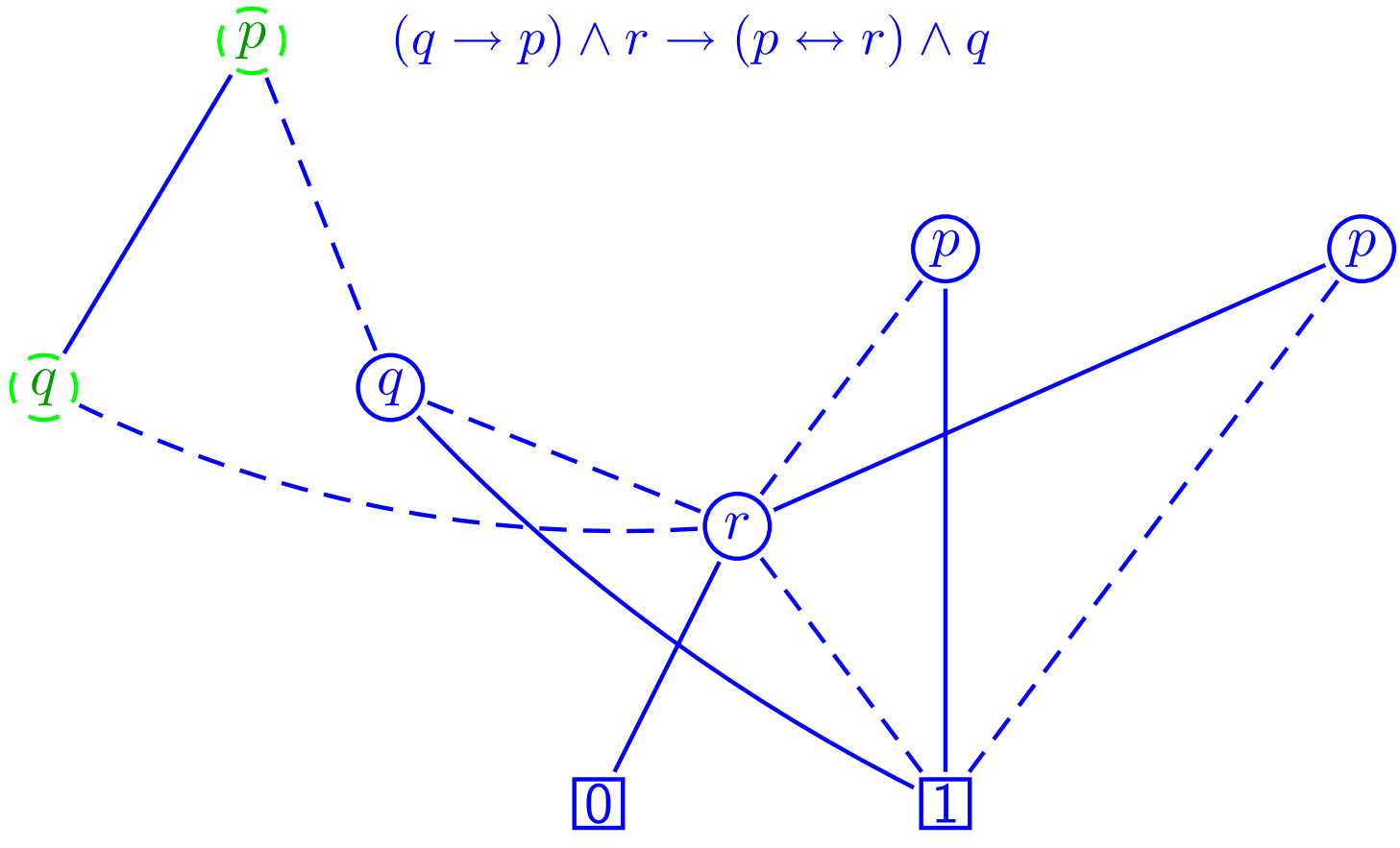
$r \to r \wedge q$



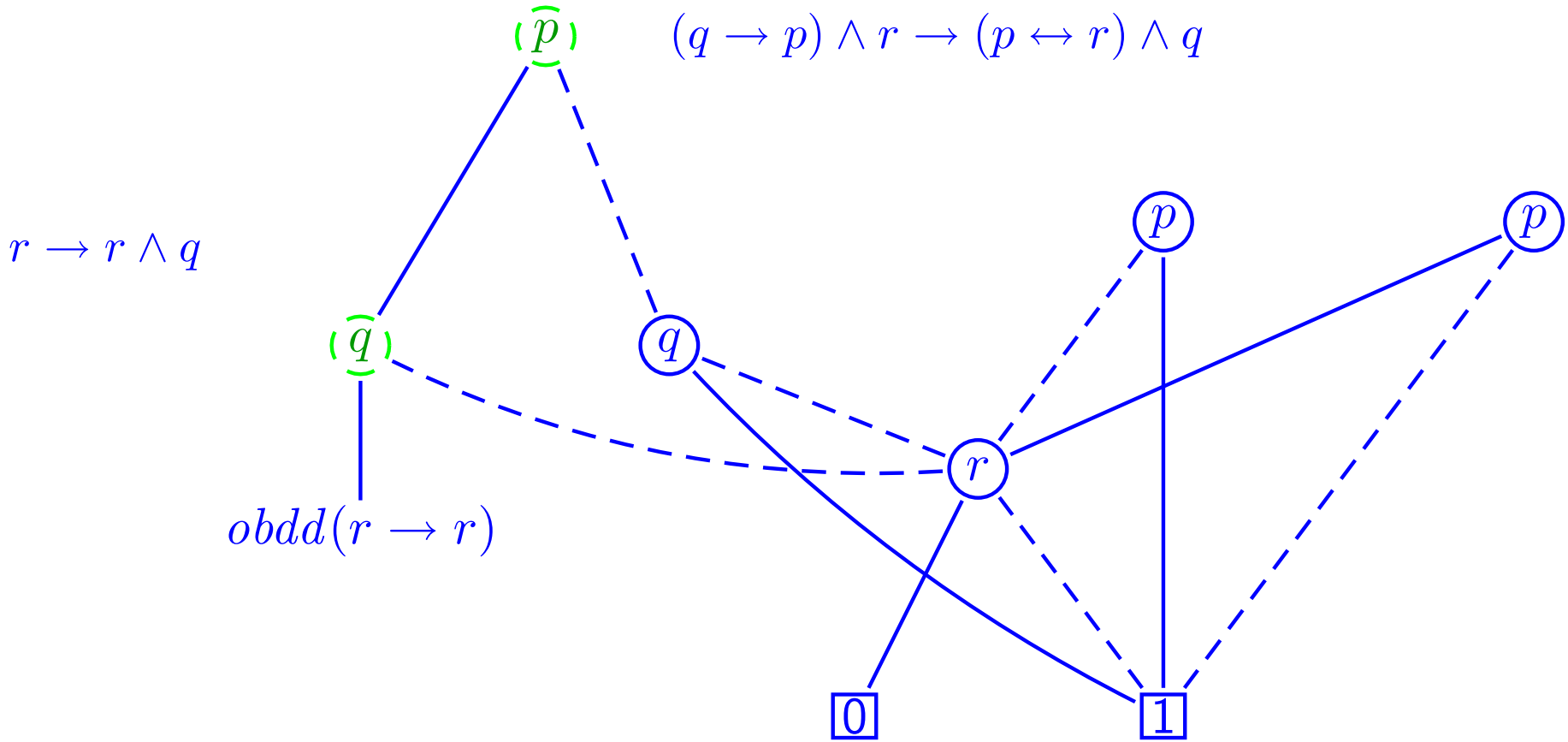
Example

$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$

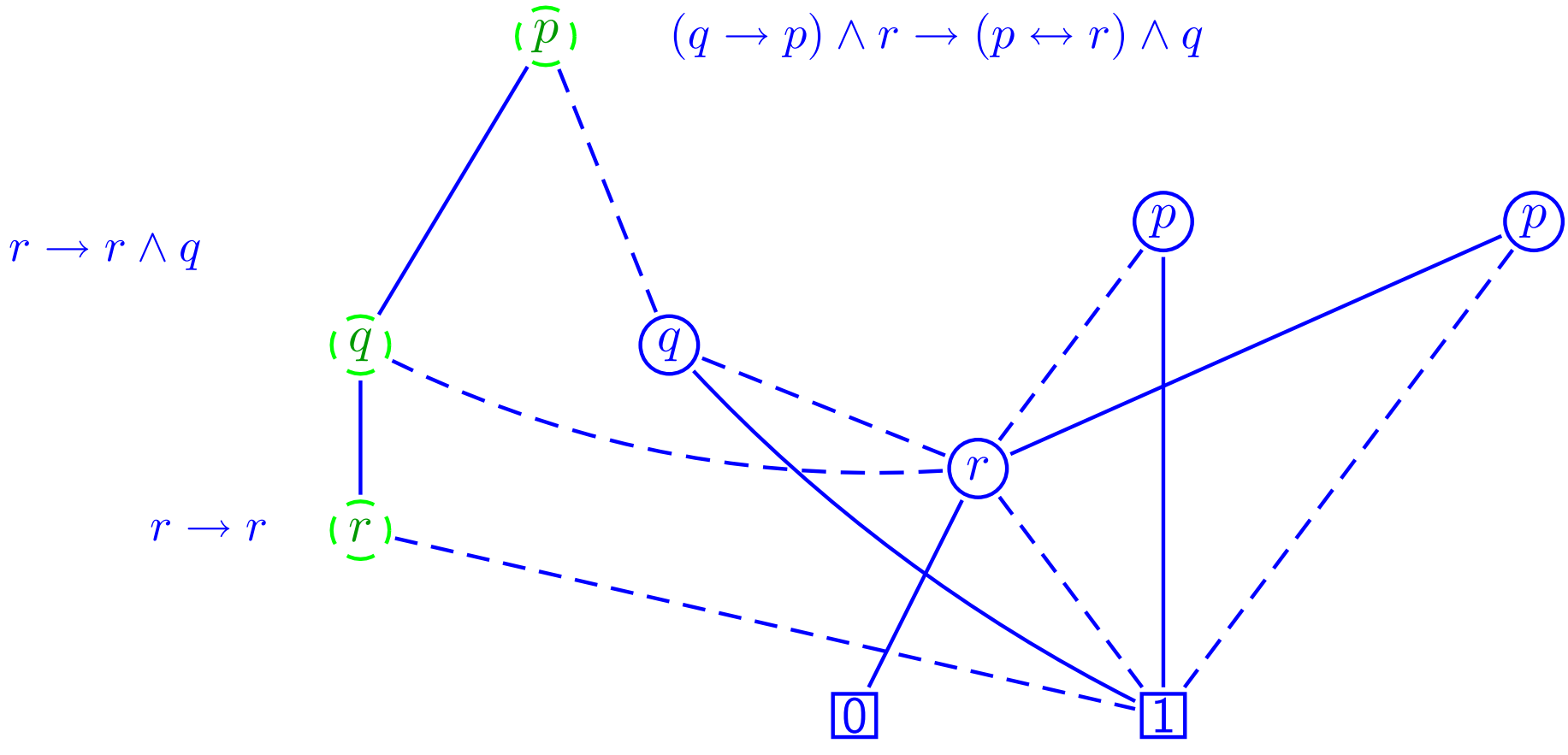
$r \to r \wedge q$



Example


$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$
$r \to r \wedge q$
p
q
$obdd(r \to r)$
p
q
r
p
0
1

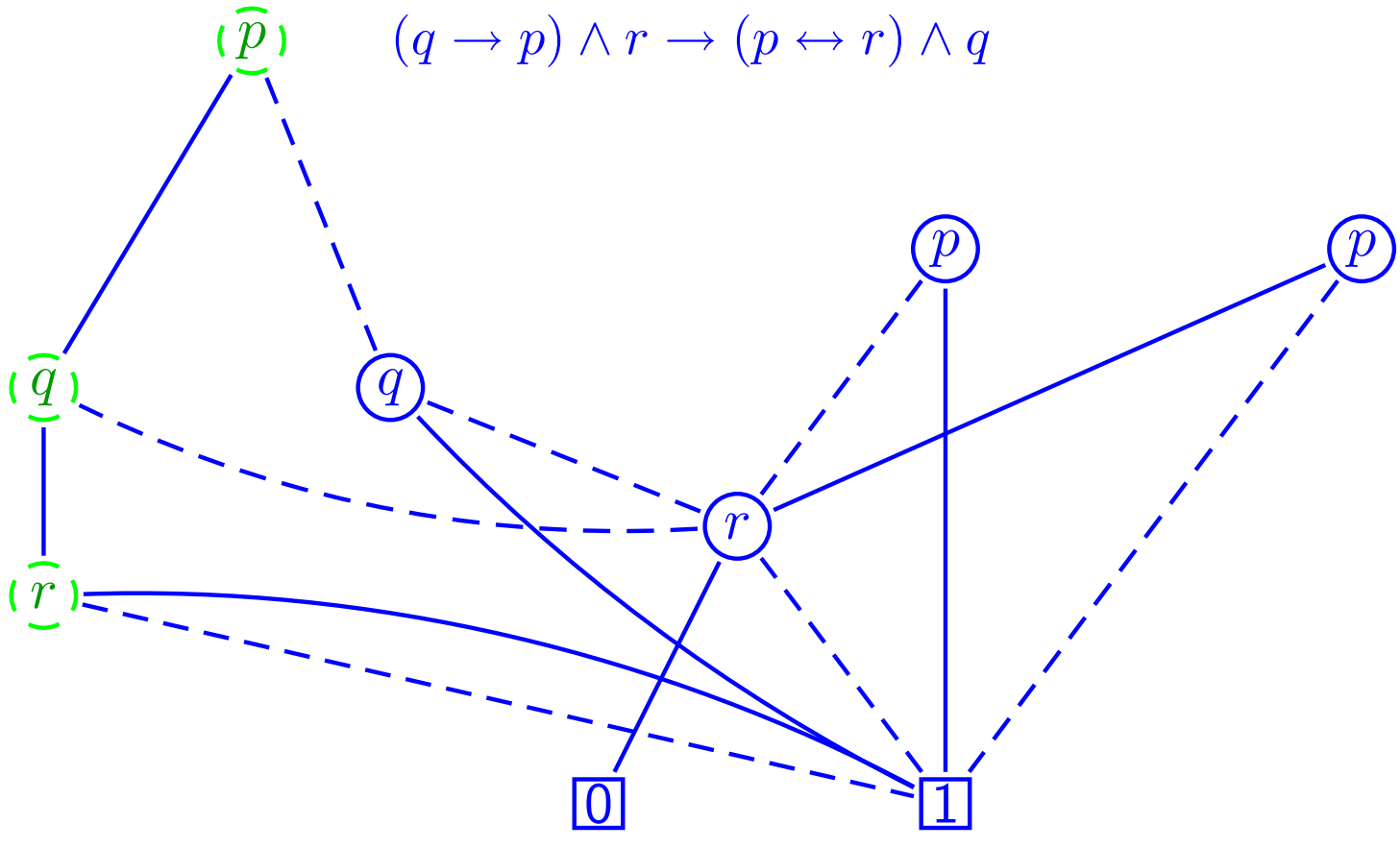
Example


$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$
$r \to r \wedge q$
$r \to r$
p
q
r
p
q
r
p
0
1

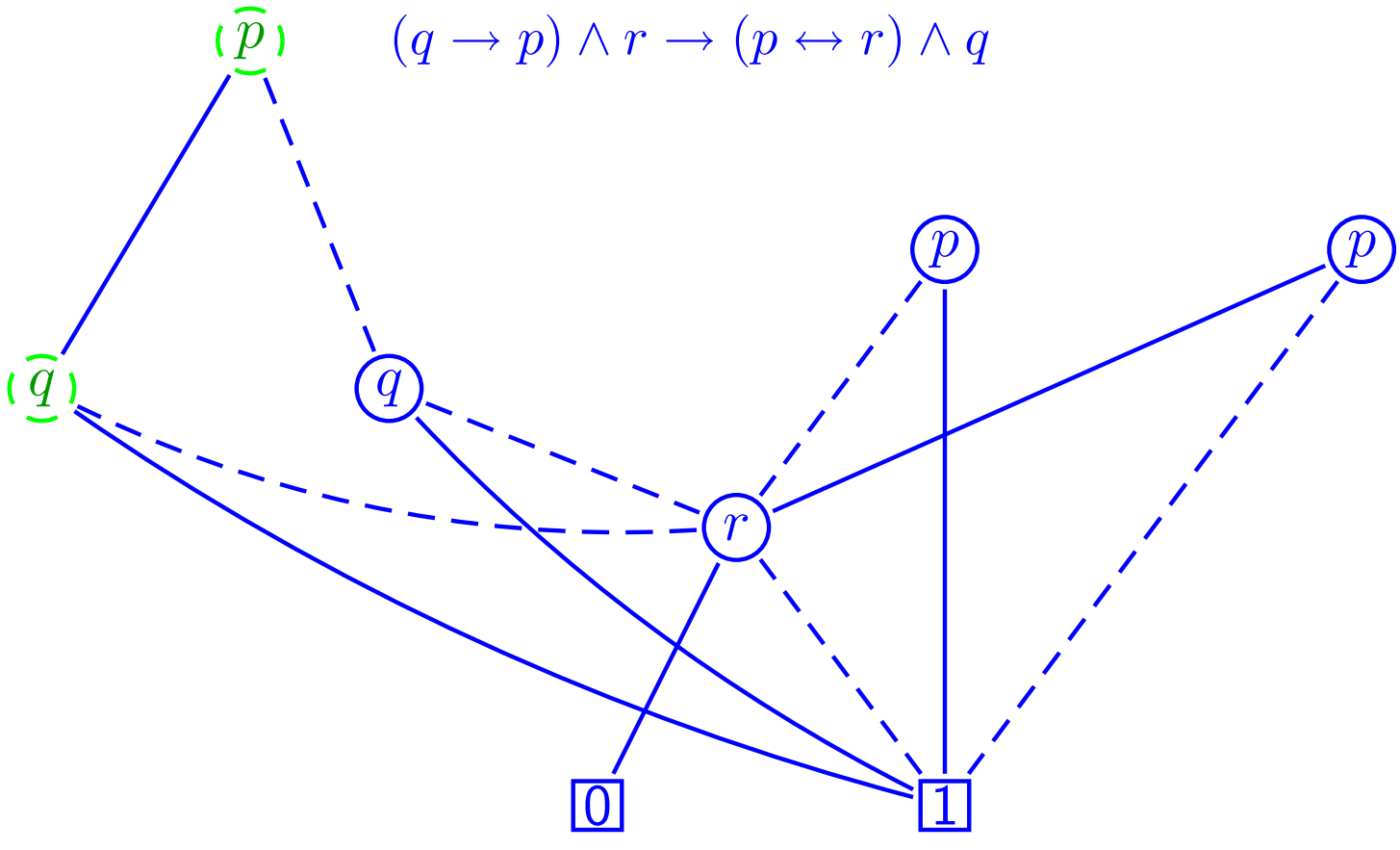
Example

$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$

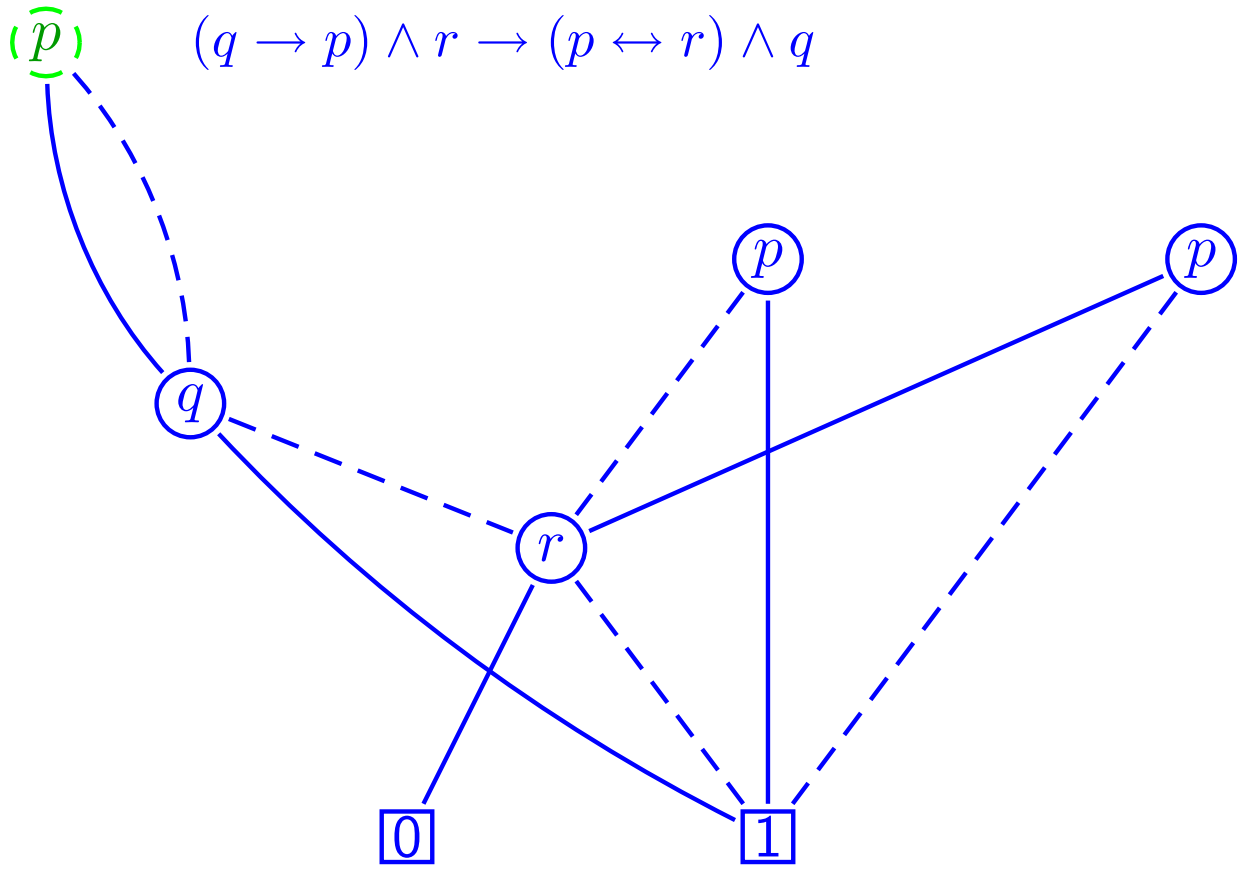
$r \to r \wedge q$



Example

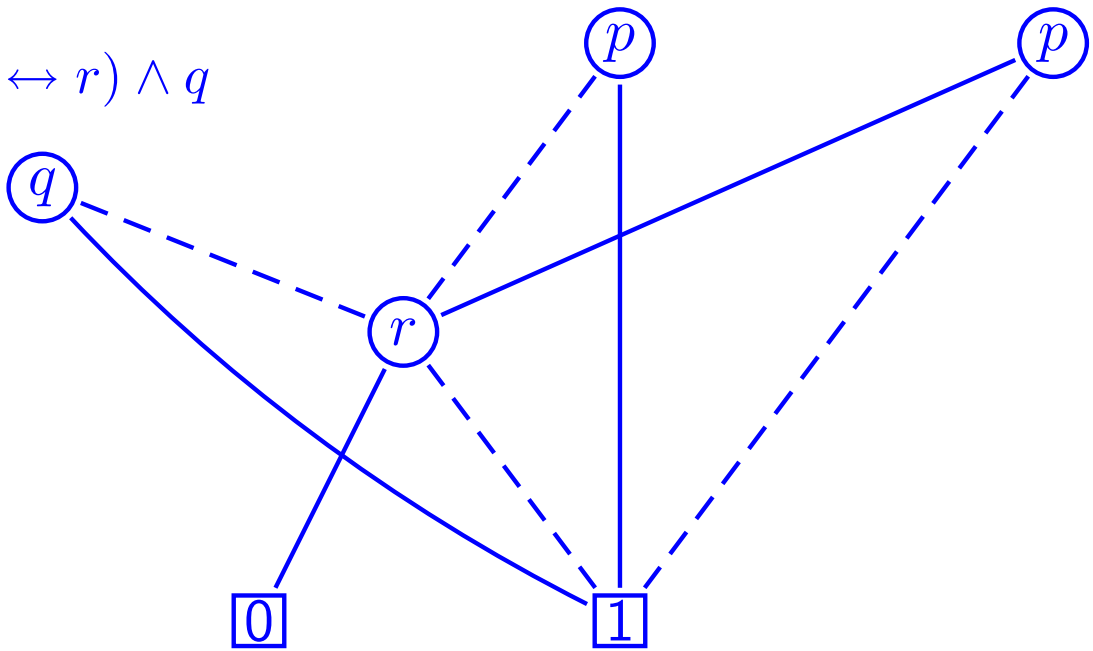


Example



Example

$(q \to p) \wedge r \to (p \leftrightarrow r) \wedge q$



Applying Boolean Functions to OBDDs

Suppose we have to compute an OBDD that represents a boolean function $f(b_1, \ldots, b_n)$, for example $b_1 \vee \ldots \vee b_n$ and we have already built OBDDs for $b_1, \ldots, b_n$.

Use the property:

$$
\begin{aligned}
& f(\ \textit{if } p \textit{ then } l_1 \textit{ else } r_1, \\
& \quad \ldots, \\
& \quad \textit{if } p \textit{ then } l_n \textit{ else } r_n\ ) = \\
& \textit{if } p \textit{ then } f(l_1, \ldots, l_n) \textit{ else } f(r_1, \ldots, r_n).
\end{aligned}
$$

Disjunction

Suppose we have OBDD's for A and B and we need to compute OBDD for $A \vee B$.

Let $A = \mathit{if}\ p\ \mathit{then}\ A'\ \mathit{else}\ A''$ and

$B = \mathit{if}\ p\ \mathit{then}\ B'\ \mathit{else}\ B''$

Then $A \vee B \equiv \mathit{if}\ p\ \mathit{then}\ A' \vee B'\ \mathit{else}\ A'' \vee B''$

Let $A = \mathit{if}\ p\ \mathit{then}\ A'\ \mathit{else}\ A''$ and

$B = \mathit{if}\ q\ \mathit{then}\ B'\ \mathit{else}\ B''$, where $p > q$

Then $A \vee B \equiv \mathit{if}\ p\ \mathit{then}\ A' \vee B\ \mathit{else}\ A'' \vee B$

We can have any boolean function in place of disjunction above.

Disjunction

Suppose we have OBDD's for A and B and we need to compute OBDD for $A \vee B$.

Let $A = \mathit{if}\ p\ \mathit{then}\ A'\ \mathit{else}\ A''$ and

$B = \mathit{if}\ p\ \mathit{then}\ B'\ \mathit{else}\ B''$

Then $A \vee B \equiv \mathit{if}\ p\ \mathit{then}\ A' \vee B'\ \mathit{else}\ A'' \vee B''$

Let $A = \mathit{if}\ p\ \mathit{then}\ A'\ \mathit{else}\ A''$ and

$B = \mathit{if}\ q\ \mathit{then}\ B'\ \mathit{else}\ B''$, where $p > q$

Then $A \vee B \equiv \mathit{if}\ p\ \mathit{then}\ A' \vee B\ \mathit{else}\ A'' \vee B$

We can have any boolean function in place of disjunction above.

Disjunction

procedure $disjunction(n_1, \ldots, n_m)$
parameters: global dag D
input: nodes $n_1, \ldots, n_m$ representing $F_1, \ldots, F_m$ in D
output: a node n representing $F_1 \vee \ldots \vee F_m$ in (modified) D
begin
  if some n_i is $\boxed{1}$ then return $\boxed{1}$
  if $m = 1$ then return n_1
  if some n_i is $\boxed{0}$ then
    return $disjunction(n_1, \ldots, n_{i-1}, n_{i+1}, \ldots, n_m)$
  p := $max\_atom(n_1, \ldots, n_m)$
  forall $i = 1 \ldots m$
    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 := $disjunction(l_1, \ldots, l_m)$
  k_2 := $disjunction(r_1, \ldots, r_m)$
  return $integrate(k_1, p, k_2, D)$
end

Summary

OBDD is a data structure which gives:

- a compact representation of boolean functions.
- facilitates efficient checking of satisfiability, validity, equivalence.
- facilitates efficient algorithms for boolean operations on OBDDs: disjunction, conjunction, negation...

Next: Quantified Boolean Formulas: (QBF)