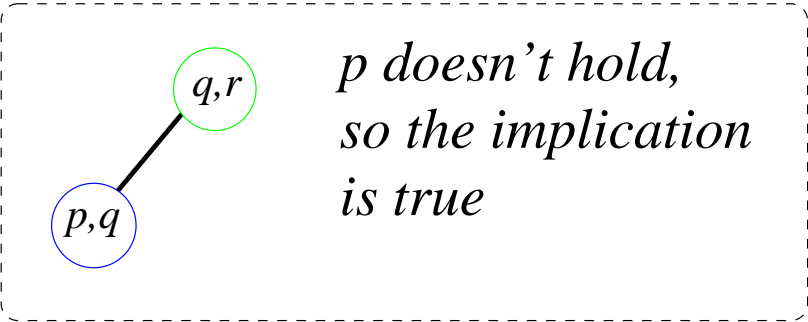
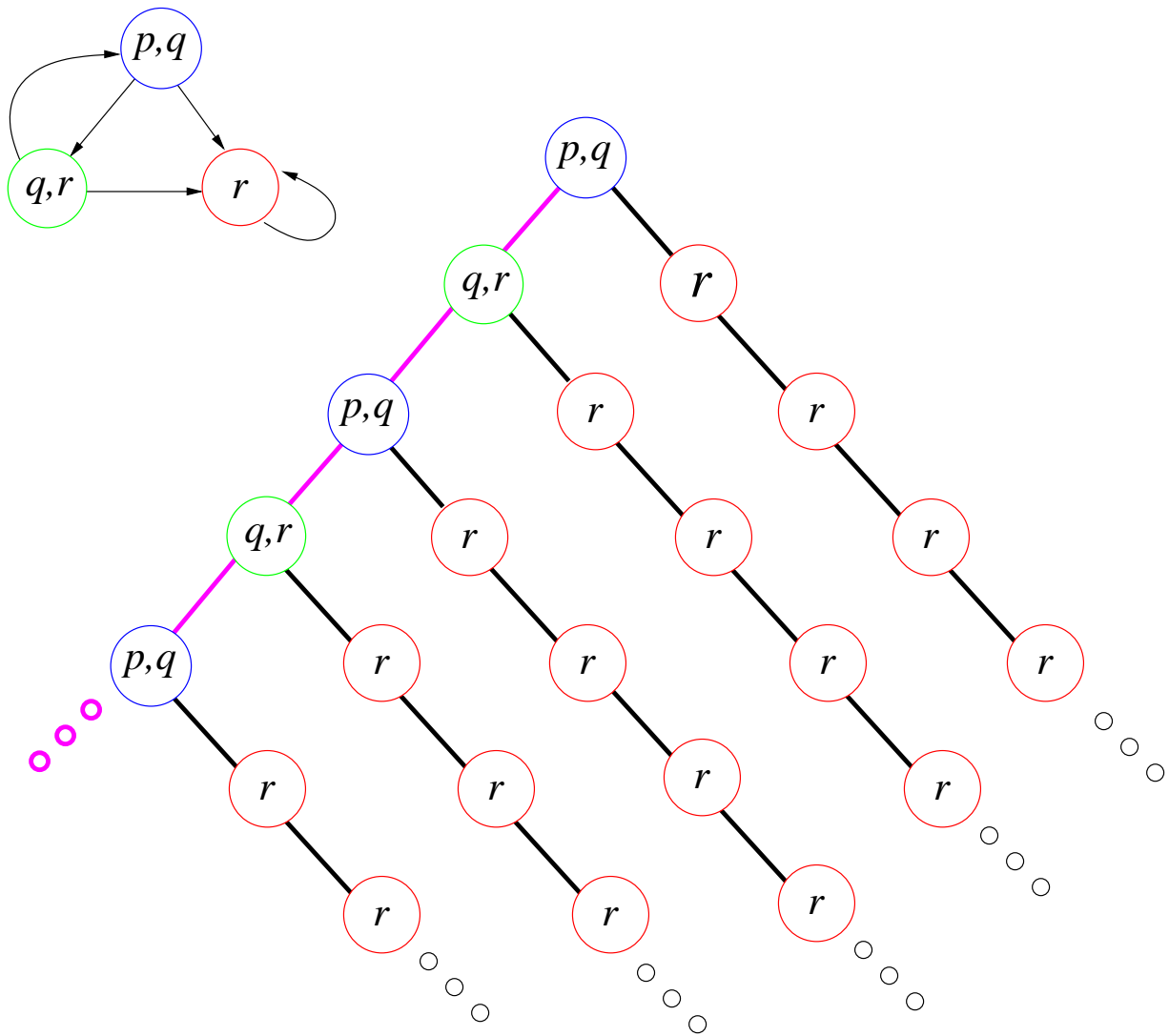


$AG[ p \Rightarrow AX r ]$

*"Whenever p holds, r is true in the next state"*







**AF[AG  $r$ ]**  
*false*

"At some point,  
 $r$  holds forever"

```
MODULE main
```

```
VAR
```

```
S: {red, blue, green};
```

```
ASSIGN
```

```
init(S) := blue;
```

```
next(S) := case S = blue: {green, red};
```

```
           S = red: {red};
```

```
           S = green: {red, blue};
```

```
           1: S;
```

```
        esac;
```

```
DEFINE
```

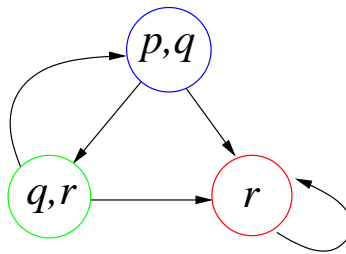
```
p := S = blue;
```

```
q := S = blue | S = green;
```

```
r := S = green | S = red;
```

```
SPEC EG(q)
```

```
SPEC AG(p -> AX(r))
```



```
-- specification EG q is true
-- specification AG (p -> AX r) is true
-- specification AF AG r is false
-- as demonstrated by the following execution sequence
-- loop starts here --
state 1.1:
r = 0
q = 1
p = 1
S = blue

state 1.2:
r = 1
p = 0
S = green

state 1.3:
r = 0
p = 1
S = blue

resources used:
user time: 0.01 s, system time: 0.01 s
BDD nodes allocated: 258
Bytes allocated: 933888
BDD nodes representing transition relation: 7 + 1
```