

Truth-Functional Simplification (with NNF)

[tfreduce] $\text{or}([\dots \text{false}, \langle \text{args} \rangle]) = \text{or}([\langle \text{args} \rangle])$
 $\text{or}([\dots \text{true}, \langle \text{args} \rangle]) = \text{true}$

$\text{and}([\dots \text{true}, \langle \text{args} \rangle]) = \text{and}([\langle \text{args} \rangle])$
 $\text{and}([\dots \text{false}, \langle \text{args} \rangle]) = \text{false}$

$\text{or}([]) = \text{false}$ $\text{and}([]) = \text{true}$

[doublered] $\text{or}([\mathbf{X} \text{ or}([\mathbf{Y}, \mathbf{Z}, \dots])]) = \text{or}([\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \dots])$

$\text{and}([\mathbf{X} \text{ and}([\mathbf{Y}, \mathbf{Z}, \dots])]) = \text{and}([\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \dots])$

[multired] $\text{or}([\mathbf{X}, \dots \langle \text{args} \rangle \dots, \mathbf{X}]) = \text{or}([\dots \langle \text{args} \rangle \dots, \mathbf{X}])$

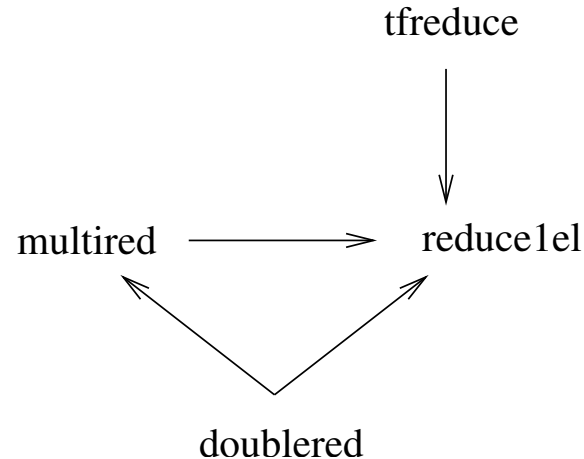
$\text{and}([\mathbf{X}, \dots \langle \text{args} \rangle \dots, \mathbf{X}]) = \text{and}([\dots \langle \text{args} \rangle \dots, \mathbf{X}])$

[reduce1el] $\text{or}([\mathbf{X}]) = \mathbf{X}$ $\text{and}([\mathbf{X}]) = \mathbf{X}$

- Each simplification applies only to a conjunction or disjunction.
- We do not worry about the arguments.
- We do not worry about any containing expression
of which the conjunction or disjunction may be an argument.

Now let's program `tfreduce`, `doublered`, `multired`, and `reduce1el`

Truth-Functional Simplification (with NNF)



If $\text{rule}A \longrightarrow \text{rule}B$,

then $\text{rule}A$ must be applied prior to $\text{rule}B$.

```
reduce(Wff, Wff) :- literal(Wff), !.
```

```
reduce(Wff, RWff) :- tfreduce(Wff, Wff1),
```

```
    doublered(Wff1, Wff2),
```

```
    multired(Wff2, Wff3),
```

```
    reduce1el(Wff3, RWff).
```

```
literal(-X) :- atom(X).    % atom is built-in
```

```
literal(X) :- atom(X).
```

tfreduce(X, X) :- literal(X), !.

tfreduce(and(Args), false) :- member(false, Args), !.

tfreduce(and(Args), true) :- remove(true, Args, []), !.

tfreduce(and(Args), and(Argsp)) :- remove(true, Args, Argsp).

tfreduce(or(Args), true) :- member(true, Args), !.

tfreduce(or(Args), false) :- remove(false, Args, []), !.

tfreduce(or(Args), or(Argsp)) :- remove(false, Args, Argsp).

member(X, [X | Rest]) :- !.

member(X, [_ | Rest]) :- member(X, Rest).

remove(E, [], []) :- !.

remove(E, [E | T], R) :- remove(E, T, R), !.

remove(E, [H | T], [H | R]) :- remove(E, T, R).

```
doublered(X, X) :- literal(X), !.  
doublered(X, Y) :- redandand(X, Y), !.  
doublered(X, Y) :- redoror(X, Y).
```

```
redandand(and([]), and([])) :- !.  
redandand(and([and(Args) | Rest]), and(Result)) :- !,  
    redandand(and(Rest), and(RedRest)),  
    append(Args, RedRest, Result).  
redandand(and([Arg1 | Rest]), and([Arg1 | RedRest])) :-  
    redandand(and(Rest), and(RedRest)).
```

```
redoror(or([]), or([])) :- !.  
redoror(or([or(Args) | Rest]), or(Result)) :- !,  
    redoror(or(Rest), or(RedRest)),  
    append(Args, RedRest, Result).  
redoror(or([Arg1 | Rest]), or([Arg1 | RedRest])) :-  
    redoror(or(Rest), or(RedRest)).
```

```
append([], L, L).  
append([H | T], L, [H | TL]) :- append(T, L, TL).
```

multired(X, X) :- literal(X), !.

multired(and(Args), and(Argsc)) :- collapse(Args, Argsc).

multired(or(Args), or(Argsc)) :- collapse(Args, Argsc).

collapse([], []) :- !.

collapse([H | T], [H | Tc]) :- remove(H, T, Tp), collapse(Tp, Tc).

reduce1el(and([X]), X) :- !.

reduce1el(or([X]), X) :- !.

reduce1el(X, X).

`simplify(E, E) :- literal(E), !.`

`simplify(and(Args), Result) :- !,
 simplist(Args, SimpArgs),
 reduce(and(SimpArgs), Result).`

`simplify(or(Args), Result) :- !,
 simplist(Args, SimpArgs),
 reduce(or(SimpArgs), Result).`

`simplist([], []) :- !.`

`simplist([H | T], [Hs | Ts]) :- simplify(H, Hs), simplist(T, Ts).`

?- tfreduce(and([]), Ans).

Ans = true

Yes

?- tfreduce(and([a]), Ans).

Ans = and([a])

Yes

?- tfreduce(and([a,b,true,c,a,true,s]), Ans).

Ans = and([a, b, c, a, s])

Yes

?- tfreduce(and([a,b,true,c,a,true,false]), Ans).

Ans = false

Yes

?- tfreduce(and([a,b,true,c,or([r,e,w,true]),true,false]), Ans).

Ans = false

Yes

?- tfreduce(and([true]), Ans).

Ans = true

Yes

?- tfreduce(and([false]), Ans).

Ans = false

Yes

?- multired(or([a,-b,c,-b,-c,d,a]), Ans).

Ans = or([a, -b, c, -c, d])

Yes

?- reduce1el(and([a]), Ans).

Ans = a

Yes

?- doublered(or([and([a,b,-c]), -a, d, or([-b,-a]), q, w]), Ans).

Ans = or([and([a, b, -c]), -a, d, -b, -a, q, w])

Yes

?- simplify(or([-a,-b]), Ans).

Ans = or([-a, -b])

Yes

?- simplify(or([-a, -b, and([])]), Ans).

Ans = true

Yes

?- reduce(or([false, false, false, true]), Ans).

Ans = true

Yes

?- reduce(or([false, false, and([a,b]), or([a,b,-c]), -c]), Ans).

Ans = or([and([a, b]), a, b, -c])

Yes

```
?- reduce(or([false, false, a, a, a, a]), Ans).
```

```
Ans = a
```

```
Yes
```

```
?- simplify(or([-a,
```

```
|           -b,  
|           and([false,  
|               a,  
|               or([d,e,r,p]),  
|               true ] ),  
|           or([x,y]),  
|           and([a, and([b,c]])),  
|           or([-a, c, d, -q, or([-b,r,p]])),  
|           or([a,b, and([false, true]) ]),  
|           and([e, true, true]),  
|           false, false, false,  
|           or([]) ] ),  
|           Ans).
```

```
Ans = or([-a, -b, x, y, and([a, b, c]), c, d, -q, r, p, a, b, e])
```

```
Yes
```