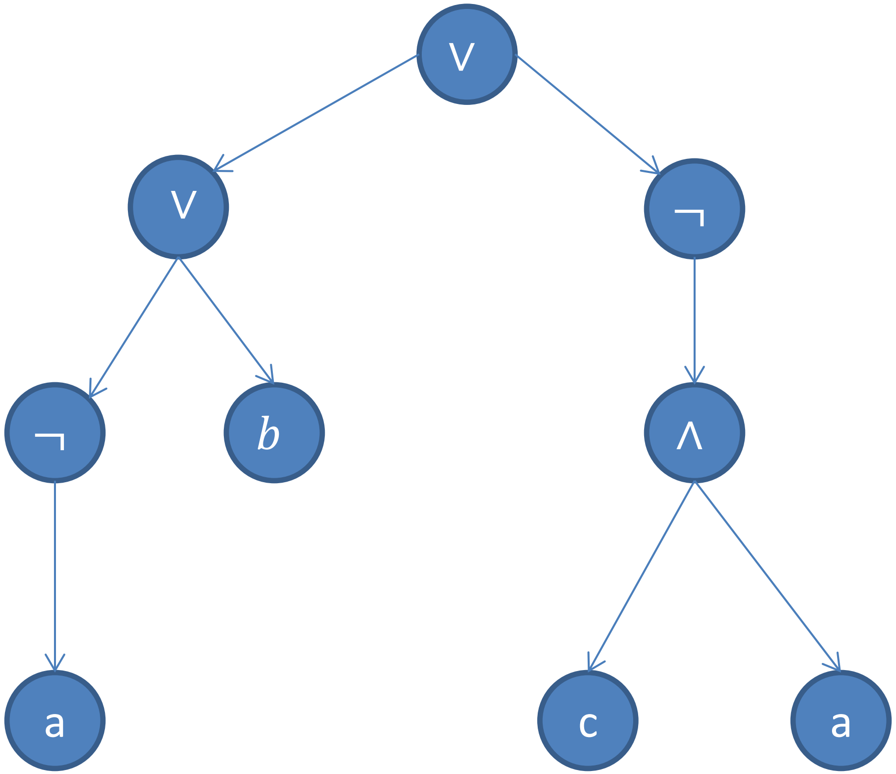
Rewrite System

- Example: need to prove the termination of the following rules:
  - $\neg \neg x \rightarrow x$
  - $\neg(x \lor y) \rightarrow (\neg x) \land (\neg y)$
  - $\neg(x \land y) \rightarrow (\neg x) \lor (\neg y)$
  - $x \land (y \lor z) \rightarrow (x \land y) \lor (x \land z)$
  - $(y \lor z) \land x \rightarrow (y \land x) \lor (z \land x)$

- Rule can be applied to top term or inner term

- We consider terms as labeled trees.
- $(\neg\, a \land b) \lor \neg(c \land d)$



- Suppose $\lesssim$ is a quasi-order on trees, $<$ the corresponding strict order
- $<$ is a simplification order if:
  - subterm property:
    $$f(\dots, s, \dots) > s$$
  - monotony:
    $$s > t \Rightarrow f(\dots, s, \dots) > f(\dots, t, \dots)$$
  - deletion property:
    $$f(\dots, s, \dots) > f(\dots \dots)$$

- In case of fixed arity the deletion property isn't necessary

- Claim: Suppose we have a finite alphabet Σ, and $<$ is a simplification order. Then $\lesssim$ is WQO
- Proof:
  Consider the identity WQO on Σ. (Why it's a WQO ?)
  Let $t_1, t_2, \dots, t_n, \dots$ be some infinite sequence of trees. According to Kruskal's theorem we know that $t_i \hookrightarrow t_j$ for some $i < j$ ($\hookrightarrow$ stands for "can be embedded")

- Claim:

$t = g(t_1, \dots, t_n) \hookrightarrow s = f(s_1, \dots, s_m) \Rightarrow t \lesssim s$

- Proof:

Since $t \hookrightarrow s$ we must have one of the following cases:

1. $t \hookrightarrow s_j$ for some j. Then $t \lesssim s_j$ (induction) hence $t \lesssim s$ (subterm property)
2. $g = f$ and $t_i \hookrightarrow s_{j_i}$ for $j_1 < j_2 < \dots < j_n$

   By induction $t_i \lesssim s_{j_i}$.

   By monotony and deletion:

   $\mathrm{t} = \mathrm{g}(\mathrm{t}_1, \dots, \mathrm{t}_\mathrm{n}) \lesssim f\left(s_{j_1}, \dots, s_{j_n}\right) \lesssim f(s_1, \dots, s_m) = s$

So $\lesssim$ is W.Q.O.

- How can we prove termination of rewriting system ?
- If we can define a simplification order such that for every rule $l \to r$ we have a decrease in the order, we're done !
- Since we have: $s < t \Rightarrow f(\ldots s \ldots) < f(\ldots t \ldots)$ it guaranties that inner substitutions will cause a decrease in the top term.

- We'll see several example for simplification orders.

multiset path Order

- $t = g(t_1, \dots, t_n) \quad s = f(s_1, \dots, s_m)$
- Recursive definition for $t \lesssim s$:

1. $t \lesssim s_i$ for some $1 \leq i \leq m$

2. $t_j < s$ for all $1 \leq j \leq m$

and

$(g, \{t_1, \dots, t_n\}) \lesssim_{lex} (f, \{s_1, \dots s_m\})$

- $\lesssim$ is reflexive
- $\lesssim$ is transitive (structure induction)
- $\Longrightarrow \quad \lesssim$ is quasi-order.

- When proving termination of rewrite systems we are mainly interested with $<$ relation rather then $\lesssim$.
- The following can be proved by structural induction:
- Suppose $t = g(t_1, \ldots, t_n) \quad s = f(s_1, \ldots, s_m)$. then $t < s$ iff:
  - $t \leq s_i$ for some $1 \leq i \leq m$

  or
  - $g < f$ and $t_j < s$ for all $1 \leq j \leq m$

  or
  - $g \approx f$ and $\{\mathrm{t}_1, \ldots, \mathrm{t}_\mathrm{n}\} <_{lex} \{s_1, \ldots, s_m\}$
- $\Rightarrow$ the multiset path order is a simplification order !!

- So we get that for finite alphabet Σ the multiset path order is WQO
- What about infinite alphabet ?
  We can prove directly from Kruskal's theorem that if the alphabet Σ is WQO, then the multiset path order $\lesssim$ is also WQO.
- Proof:
  Let $t_1, t_2, \dots, t_n, \dots$ be some infinite sequence of trees. According to Kruskal's theorem we know that $t_i \hookrightarrow t_j$ for some $i < j$ ($\hookrightarrow$ stands for "can be embedded")

- Claim:

$t = g(t_1, \dots, t_n) \hookrightarrow s = f(s_1, \dots, s_m) \Rightarrow t \lesssim s$

- Proof:

Since $t \hookrightarrow s$ we must have one of the following cases:

1. $t \hookrightarrow s_j$ for some j. Then $t \lesssim s_j$ (induction) hence $t \lesssim s$
2. $g \lesssim f$ and $t_i \hookrightarrow s_{j_i}$ for $j_1 < j_2 < \dots < j_n$

   By induction $t_i \lesssim s_{j_i} \implies t_i < s$ for all i.

   Also $\{t_1, \dots, t_n\} \lesssim \{s_1, \dots, s_m\} \implies$
   $(g, \{t_1, \dots, t_n\}) \lesssim_{lex} (f, \{s_1, \dots s_m\}) \implies t \lesssim s$

- So $\lesssim$ is W.Q.O.

- Back to the example:
  - *¬ ¬x -> x*
  - *¬(x ∨ y) -> (¬ x) ∧ (¬ y)*
  - *¬(x ∧ y) -> (¬ x) ∨ (¬ y)*
  - *x ∧ (y ∨ z) -> (x ∧ y) ∨ (x ∧ z)*
  - *(y ∨ z) ∧ x -> (y ∧ x) ∨ (z ∧ x)*
- The alphabet is $\Sigma = \{\neg, \vee, \wedge\}$.
- Define WQO on it: $\neg > \wedge > \vee$
- Easy to verify that all the above rules do cause reduction in the multiset path order

Lexicographic Path Order

- $t = g(t_1, \dots, t_n) \quad s = f(s_1, \dots, s_m)$
- Recursive definition for $t \lesssim s$:

1. $t \lesssim s_i$ for some $1 \leq i \leq m$
2. $t_j < s$ for all $1 \leq j \leq m$

   and

   $(g, t_1, \dots, t_n) \lesssim_{lex} (f, s_1, \dots, s_m)$

- $\lesssim$ is reflexive, and transitive hence $\lesssim$ is quasi-order.
- The following can be proved:

  Let $t = g(t_1, \ldots, t_n) \quad s = f(s_1, \ldots, s_m)$ then $t < s$ iff:

  1. $t \lesssim s_i$ for some $1 \leq i \leq m$
  2. $t_j < s$ for all $1 \leq j \leq m$

     and

     $(g, t_1, \ldots, t_n) <_{lex} (f, s_1, \ldots, s_m)$

- $\Rightarrow$ Easily follows that the lexicographic path ordering is a simplification ordering.

Recursive path order

- We can also mix multiset, lexicographic, and also avoid arguments.
- Always need to preserve the property:

For $t = g(t_1, \ldots, t_n) \lesssim s = f(s_1, \ldots, s_m)$
we must have $t_j < s$ for all $1 \leq j \leq m$

- Note: $t_j < s$ and not just $t_j \lesssim s$

- Example:
  - $\neg \neg x \rightarrow x$
  - $\neg(x \vee y) \rightarrow (\neg x) \wedge (\neg y)$
  - $\neg(x \wedge y) \rightarrow (\neg x) \vee (\neg y)$
  - $x \wedge (y \vee z) \rightarrow (x \wedge y) \vee (x \wedge z)$
  - $(y \vee z) \wedge x \rightarrow (y \wedge x) \vee (z \wedge x)$
  - $(x \vee y) \vee z \rightarrow x \vee (y \vee z)$
  - $x \wedge (y \wedge z) \rightarrow (x \wedge y) \wedge z$

- Multiset won’t work for the last 2 rules …
- Lexicographic won’t work for the last rule

- We take:

$\vee < \wedge < \neg$

For "$\vee$" we use lexicographic order

For "$\wedge$" we use reverse lexicographic order