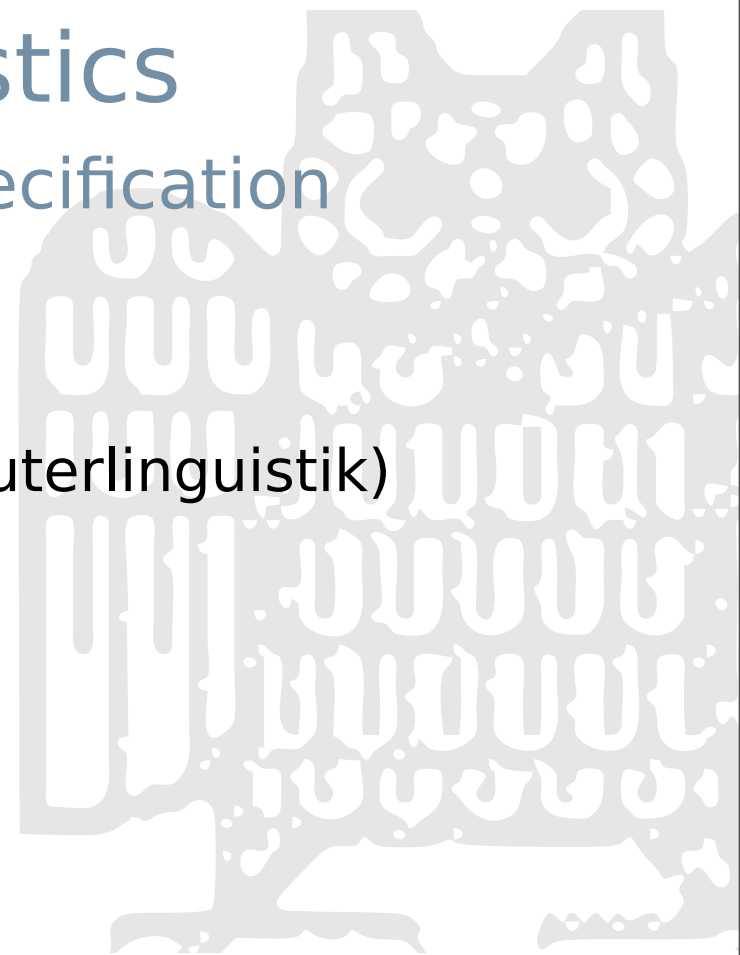


# Computational Linguistics

## Algorithms for Scope Underspecification

Dietrich Klakow & Stefan Thater  
FR 4.7 Allgemeine Linguistik (Computerlinguistik)  
Universität des Saarlandes

Summer 2014



# Today

- Refining the solver from the last lecture
- Hypernormally connected dominance graphs
- Tree Automata (in a nutshell)
- Redundancy elimination (very short)

# Solving dominance graphs

**solve( $G = \langle V, E \cup D \rangle$ ) =** (\*)

choose a free fragment  $F$  of  $G$  else fail

let  $G_1, \dots, G_k$  be the WCC's of  $G \setminus F$

let  $\langle V_i, E_i \cup D_i \rangle = \text{solve}(G_i)$

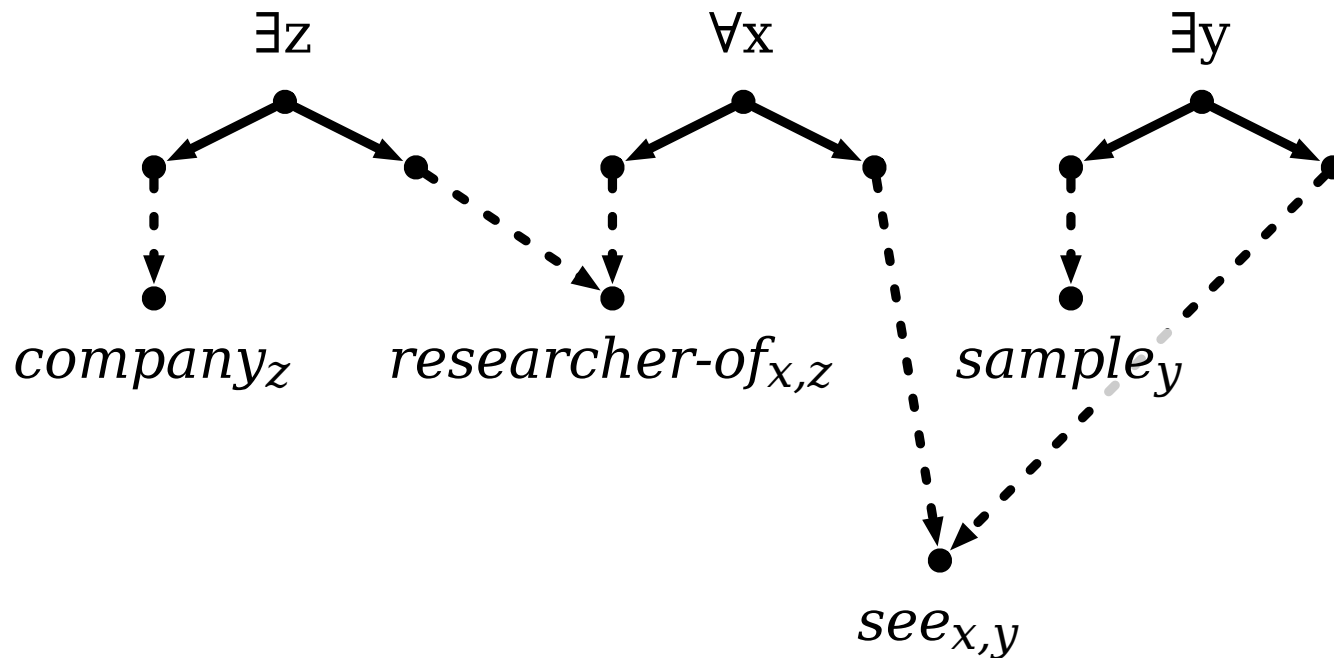
return  $\langle V, E \cup D_1 \cup \dots \cup D_k \cup D' \rangle$

where  $D'$  are dominance edges that connect the holes of  $F$  with the solved form of one of the corresponding  $G_i$

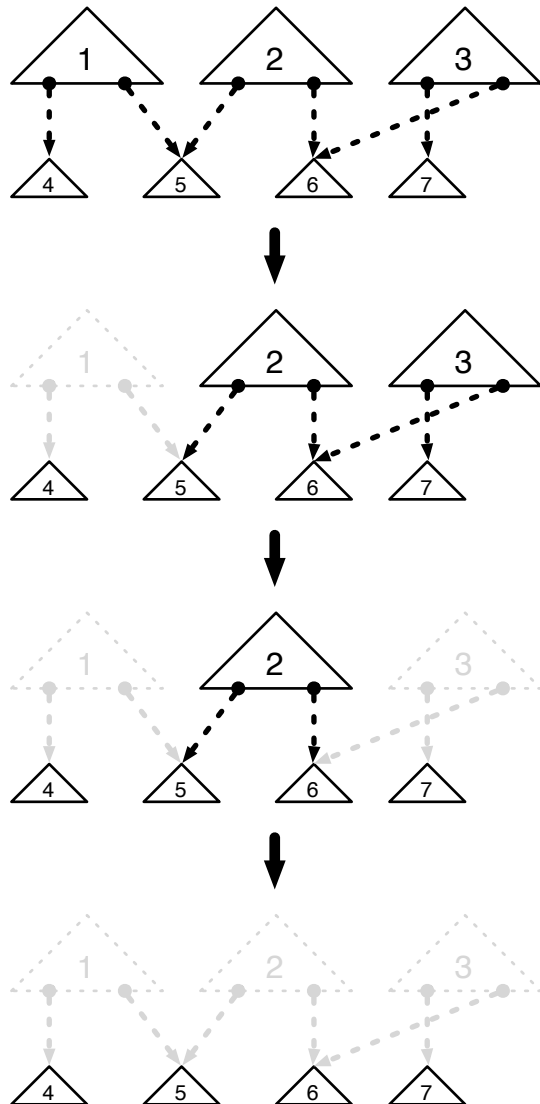
(\*) slightly simplified version, works only for connected normal dominance graphs

# An Example

- *Every researcher of a company saw a sample.*

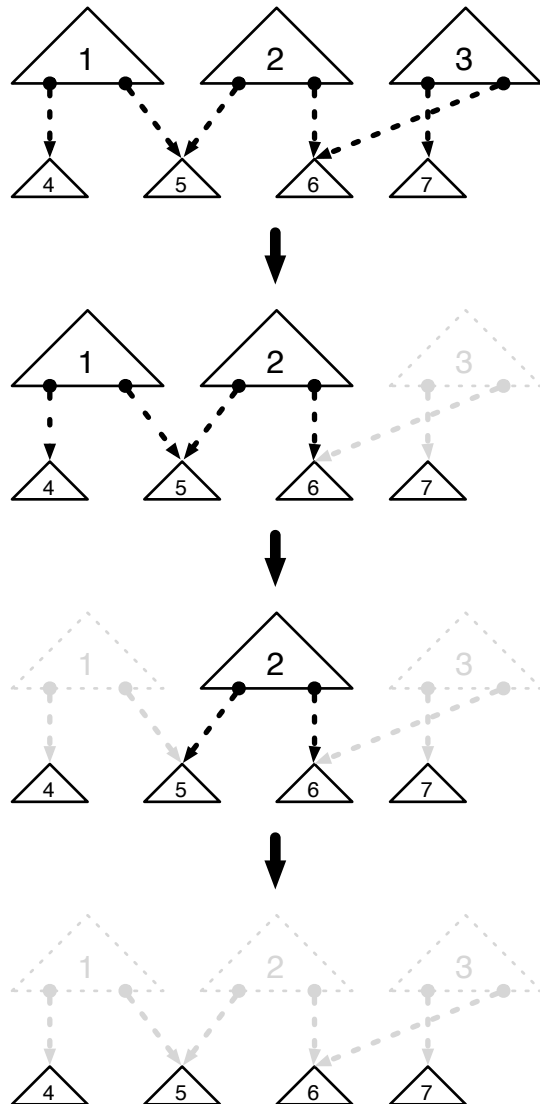


# An Example: Run #1



subgraph	free	wccs
$\{1, \dots, 7\}$	1	$\{4\}, \{2, 3, 5, 6, 7\}$
$\{2, 3, 5, 6, 7\}$	3	$\{7\}, \{2, 5, 6\}$
$\{2, 5, 6\}$	2	$\{5\}, \{6\}$

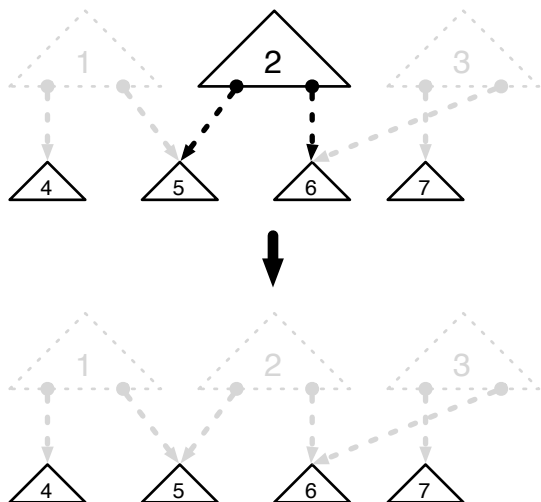
# An Example: Run #2



subgraph	free	wccs
$\{1, \dots, 7\}$	3	$\{7\}, \{1, 2, 4, 5, 6\}$
$\{1, 2, 4, 5, 6\}$	1	$\{4\}, \{2, 5, 6\}$
$\{2, 5, 6\}$	2	$\{5\}, \{6\}$

# An Example

**Problem:** The algorithm is applied twice to the subgraph  $\{2,5,6\}$



subgraph	free	wccs
$\{1, \dots, 7\}$	1	$\{4\}, \{2, 3, 5, 6, 7\}$
$\{2, 3, 5, 6, 7\}$	3	$\{7\}, \{2, 5, 6\}$
<b><math>\{2, 5, 6\}</math></b>	<b>2</b>	<b><math>\{5\}, \{6\}</math></b>

subgraph	free	wccs
$\{1, \dots, 7\}$	3	$\{7\}, \{1, 2, 4, 5, 6\}$
$\{1, 2, 4, 5, 6\}$	1	$\{4\}, \{2, 5, 6\}$
<b><math>\{2, 5, 6\}</math></b>	<b>2</b>	<b><math>\{5\}, \{6\}</math></b>

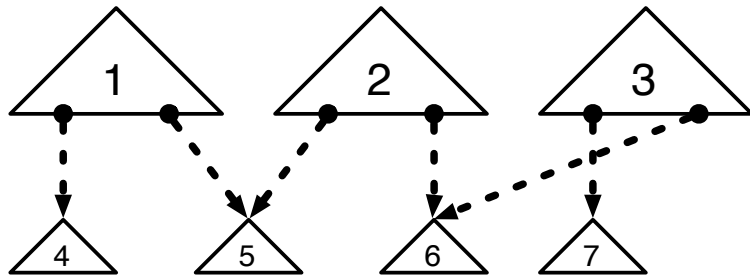
# Chart-based solver

- **Basic idea:** use dynamic programming techniques and store intermediate results in a chart-like datastructure
- The chart records how graphs are decomposed into smaller subgraphs if free fragments are removed
  - The chart assigns each subgraph a “split.”
  - Splits consist of (references to) a free fragment  $F$  and the weakly connected components of  $G \setminus F$ .
    - Notation:  $F(G_1, \dots, G_n)$

# The algorithm

```
GRAPH-SOLVER-CHART( $G'$ )
1  if there is an entry for  $G'$  in the chart
2    then return true
3   $free \leftarrow$  FREE-FRAGMENTS( $G'$ )
4  if  $free = \emptyset$ 
5    then return false
6  if  $G'$  contains only one fragment
7    then return true
8
9  for each  $F \in free$ 
10 do  $split \leftarrow$  SPLIT( $G', F$ )
11   for each  $S \in$  WCCS( $G' - F$ )
12   do if GRAPH-SOLVER-CHART( $S$ ) = false
13     then return false
14   add ( $G', split$ ) to the chart
15 return true
```

# An Example

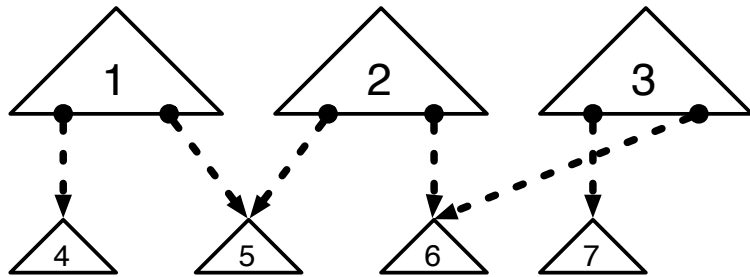


```

GRAPH-SOLVER-CHART( $G'$ )
1  if there is an entry for  $G'$  in the chart
2    then return true
3   $free \leftarrow$  FREE-FRAGMENTS( $G'$ )
4  if  $free = \emptyset$ 
5    then return false
6  if  $G'$  contains only one fragment
7    then return true
8
9  for each  $F \in free$ 
10 do  $split \leftarrow$  SPLIT( $G', F$ )
11   for each  $S \in$  WCCS( $G' - F$ )
12     do if GRAPH-SOLVER-CHART( $S$ ) = false
13       then return false
14   add ( $G', split$ ) to the chart
15 return true
    
```

subgraph	splits
$\{1, \dots, 7\}$	1( $\{4\}, \{2, 3, 5, 6, 7\}$ )
	2( $\{1, 4, 5\}, \{3, 6, 7\}$ )
	3( $\{1, 2, 4, 5, 6\}, \{7\}$ )
$\{2, 3, 5, 6, 7\}$	2( $\{5\}, \{3, 6, 7\}$ )
	3( $\{2, 5, 6\}, \{7\}$ )
$\{1, 2, 4, 5, 6\}$	1( $\{4\}, \{2, 5, 6\}$ )
	2( $\{1, 4, 5\}, \{6\}$ )
$\{2, 5, 6\}$	2( $\{5\}, \{6\}$ )
$\{3, 6, 7\}$	3( $\{7\}, \{6\}$ )
$\{1, 4, 5\}$	1( $\{4\}, \{5\}$ )

# An Example



```

GRAPH-SOLVER-CHART( $G'$ )
1  if there is an entry for  $G'$  in the chart
2    then return true
3   $free \leftarrow$  FREE-FRAGMENTS( $G'$ )
4  if  $free = \emptyset$ 
5    then return false
6  if  $G'$  contains only one fragment
7    then return true
8
9  for each  $F \in free$ 
10 do  $split \leftarrow$  SPLIT( $G', F$ )
11   for each  $S \in WCCS(G' - F)$ 
12     do if GRAPH-SOLVER-CHART( $S$ ) = false
13       then return false
14   add ( $G', split$ ) to the chart
15 return true
    
```

subgraph	splits
$\{1, \dots, 7\}$	1( $\{4\}, \{2, 3, 5, 6, 7\}$ )
	2( <b><math>\{1, 4, 5\}</math></b> , <b><math>\{3, 6, 7\}</math></b> )
	3( $\{1, 2, 4, 5, 6\}, \{7\}$ )
$\{2, 3, 5, 6, 7\}$	2( $\{5\},$ <b><math>\{3, 6, 7\}</math></b> )
	3( <b><math>\{2, 5, 6\}</math></b> , $\{7\}$ )
$\{1, 2, 4, 5, 6\}$	1( $\{4\},$ <b><math>\{2, 5, 6\}</math></b> )
	2( <b><math>\{1, 4, 5\}</math></b> , $\{6\}$ )
$\{2, 5, 6\}$	2( $\{5\}, \{6\}$ )
$\{3, 6, 7\}$	3( $\{7\}, \{6\}$ )
$\{1, 4, 5\}$	1( $\{4\}, \{5\}$ )

# Complexity

- Let  $G$  be a dominance graph with  $n$  nodes and  $m$  edges
- The computation of free fragments takes time  $O(n + m)$
- The time to compute the chart is  $O(n(n + m) \text{wcsg}(G))$ 
  - $\text{wcsg}(G)$  = the number of weakly connected subgraphs of  $G$
- Worst case complexity in the number of nodes is  $O(n^2 2^n)$ 
  - $\Rightarrow$  big improvement compared to  $O(n^2 n!)$  of the basic solver
  - ( $n!$  = upper bound for the number of solved forms)

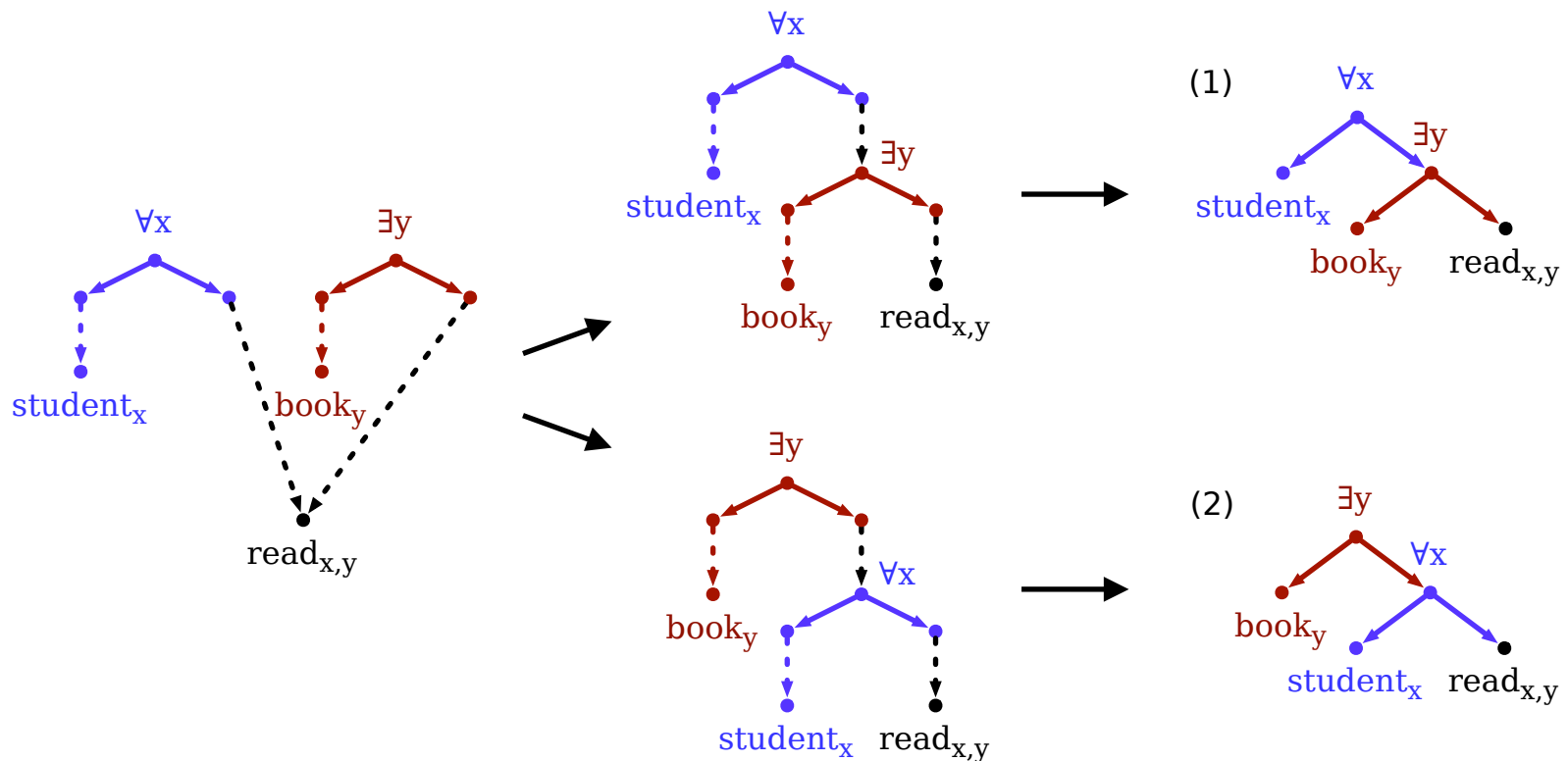
# Hypernormally Connected Dominance Graphs

# The big picture

- *Every student reads a book*

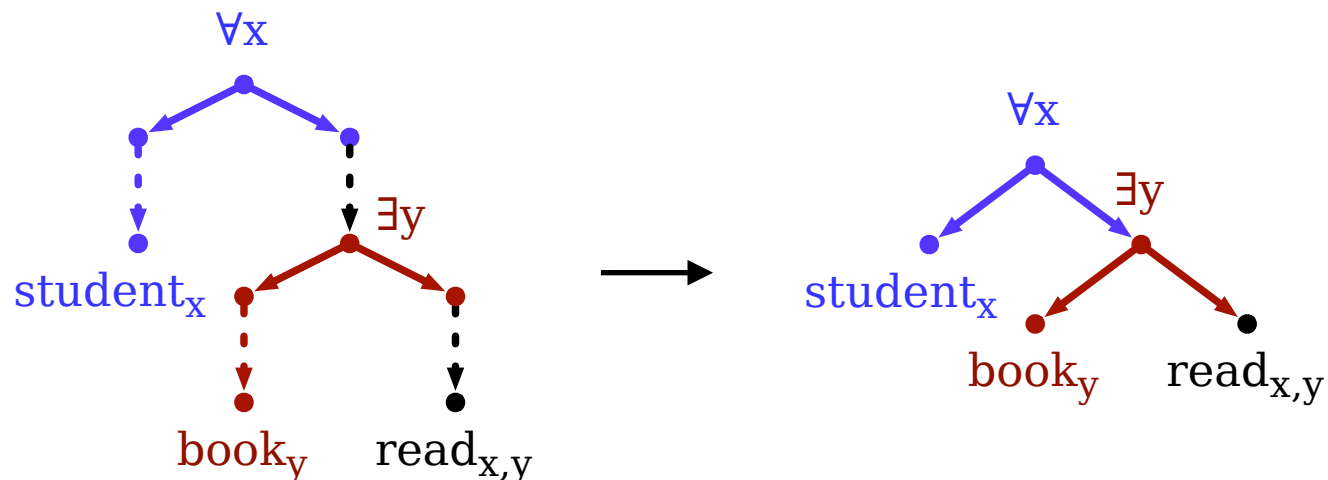
(1)  $\forall x(\text{student}(x), \exists y(\text{book}(y), \text{read}(x, y)))$

(2)  $\exists y(\text{book}(y), \forall x(\text{student}(x), \text{read}(x, y)))$



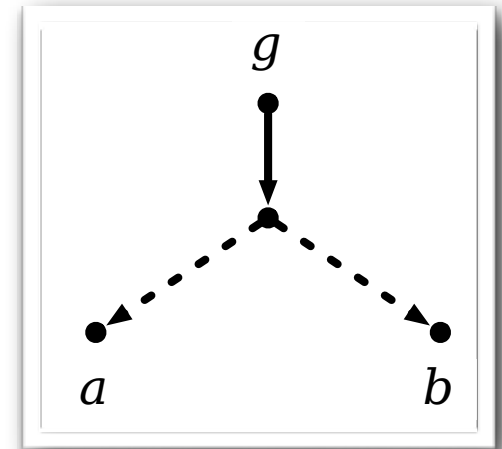
# Simple solved forms

- We are usually interested only in solved forms in which every hole is related to exactly one root
  - let's call such solved forms "simple"
- Simple solved forms can be mapped to "proper" trees simply by "plugging" the hole with the root connected to it by a dominance edge.



# Not all solved forms are simple

- **Problem:**  
Not all solved forms are simple
- **Solution:**  
Identify a class of dominance graphs that only have simple solved forms
- $\Rightarrow$  Hypernormally connected dominance graphs

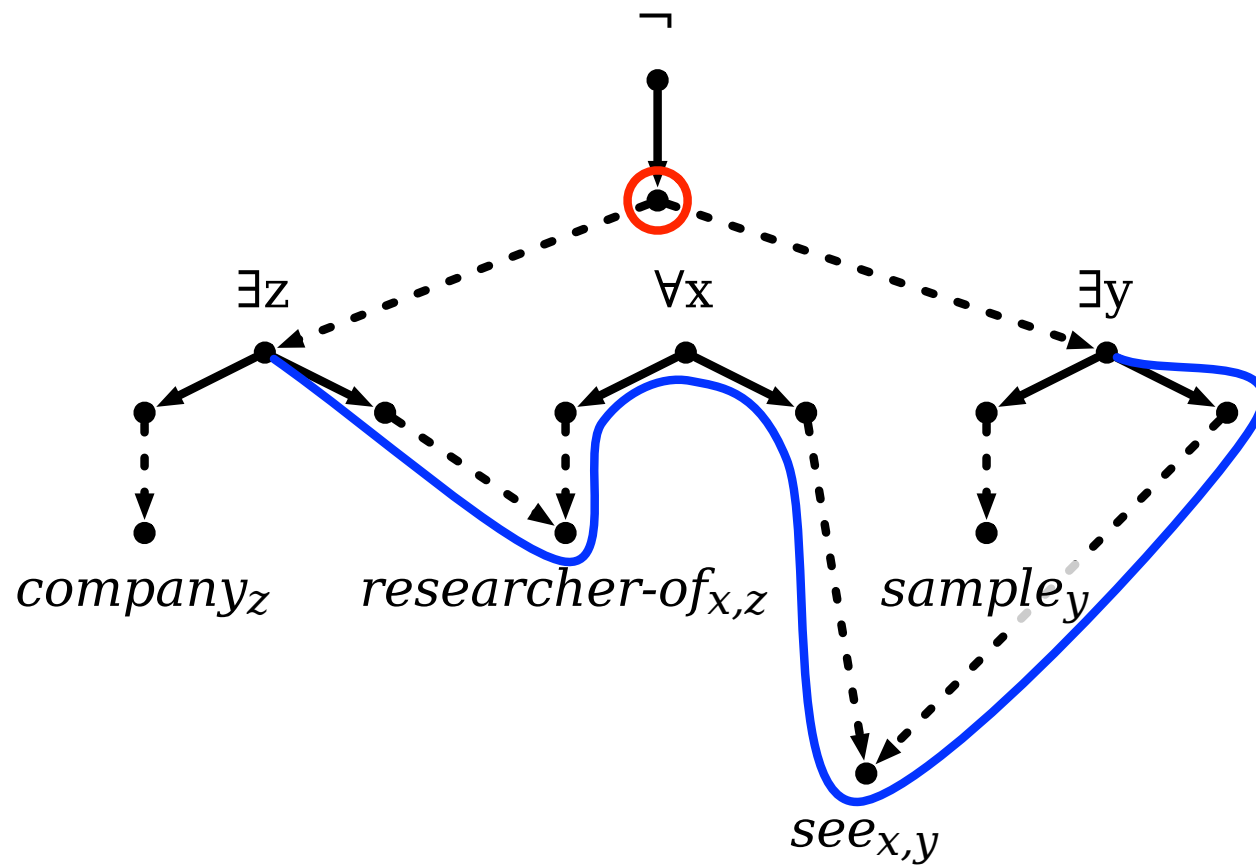


# Hypernormally connected dominance graphs

- A **hypernormal path** in a (normal) dominance graph  $G$  is a path in the undirected version of  $G$  that does not use two dominance edges incident to the same hole.
- A (normal) dominance graph  $G$  is **hypernormally connected** if each pair of nodes is connected by some hypernormal path.

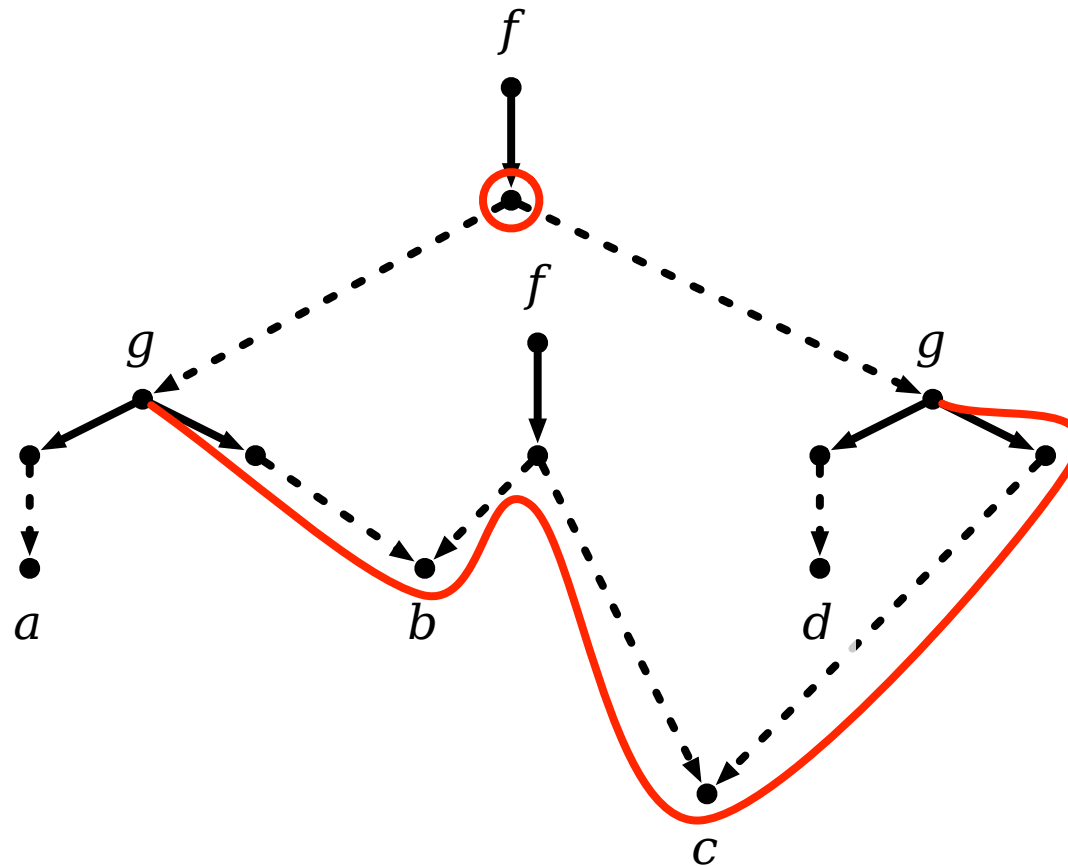
# Hypernormally connected dominance graphs

- Hypernormally connected:



# Hypernormally connected dominance graphs

- Not hypernormally connected:



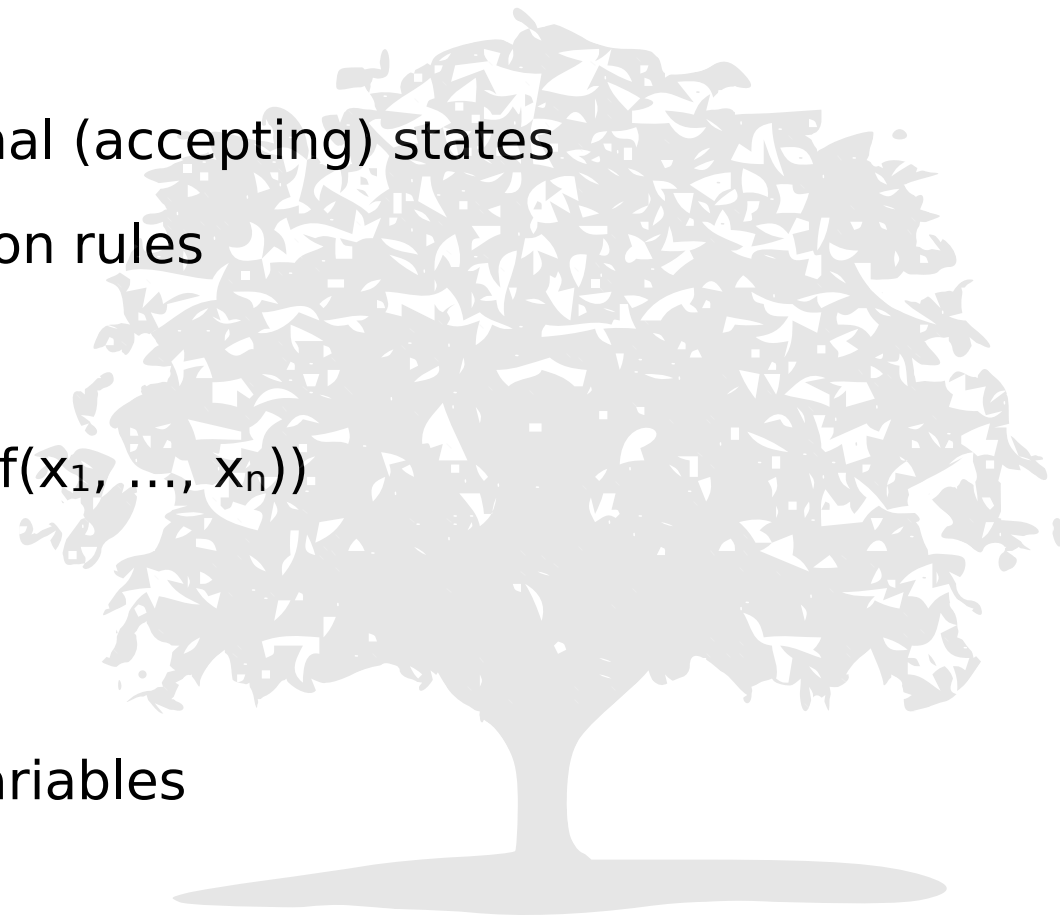
# Hypernormally connected dominance graphs

- **Lemma:** if  $G$  is a hypernormally connected normal dominance graph with free fragment  $F$ , then all WCCs of  $G \setminus F$  are hypernormally connected.
- **Proposition:** if a normal dominance graph is hypernormally connected, then all its (minimal) solved forms are simple.

# Tree Automata

# Bottom-up Tree Automaton

- **A tree automaton** is a tuple  $A = \langle Q, \Sigma, Q_f, \Delta \rangle$ 
  - $\Sigma$  a finite ranked signature
  - $Q$  a finite set of states
  - $Q_f \subseteq Q$  a finite set of final (accepting) states
  - $\Delta$  a finite set of transition rules
- **Transition rules:**
  - $f(q_1(x_1), \dots, q_n(x_n)) \rightarrow q(f(x_1, \dots, x_n))$ 
    - $f \in \Sigma$
    - $q, q_1, \dots, q_n \in Q$
    - $x_1, \dots, x_n$  different variables



# An Example Computation

- $Q = \{q_3, q_4, \dots\}$ ,  $\Sigma = \{\exists x|_2, \text{book}_y|_0, \dots\}$ ,  $Q_f = \{q_{12345}\}$

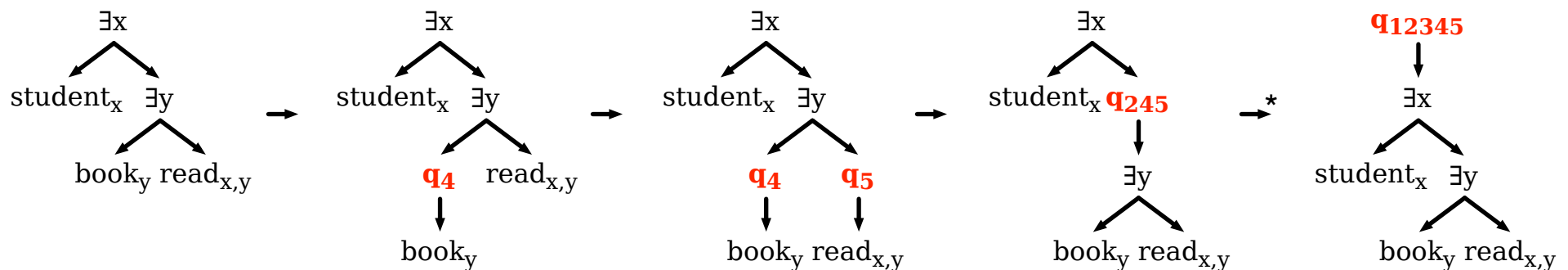
$$\exists x(q_3(x_1), q_{245}(x_2)) \rightarrow q_{12345}(\exists x(x_1, x_2))$$

$$\exists y(q_4(x_1), q_5(x_2)) \rightarrow q_{245}(\exists y(x_1, x_2))$$

$$\text{student}_x \rightarrow q_3(\text{student}_x)$$

$$\text{book}_y \rightarrow q_4(\text{book}_y)$$

$$\text{read}_{x,y} \rightarrow q_5(\text{read}_{x,y})$$



# An Example Computation

- $Q = \{q_3, q_4, \dots\}$ ,  $\Sigma = \{\exists x|_2, \text{book}_y|_0, \dots\}$ ,  $Q_f = \{q_{12345}\}$

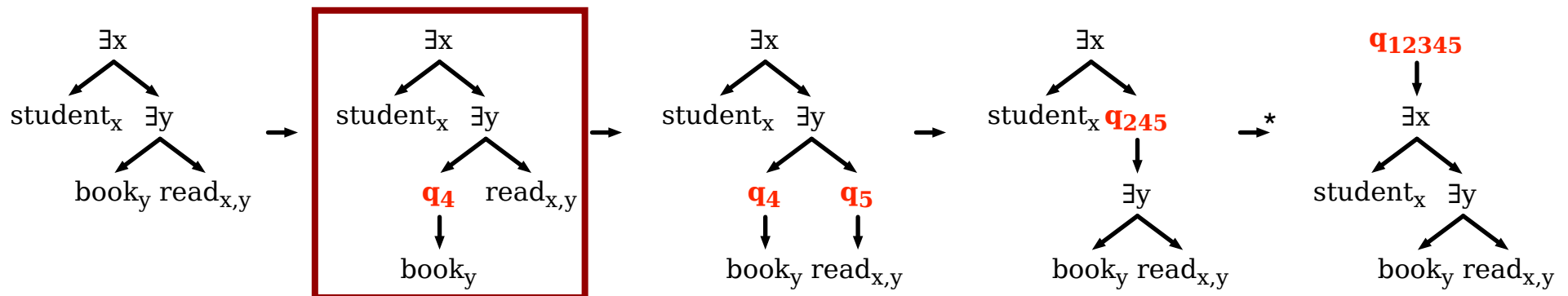
$$\exists x(q_3(x_1), q_{245}(x_2)) \rightarrow q_{12345}(\exists x(x_1, x_2))$$

$$\exists y(q_4(x_1), q_5(x_2)) \rightarrow q_{245}(\exists y(x_1, x_2))$$

$$\text{student}_x \rightarrow q_3(\text{student}_x)$$

$$\text{book}_y \rightarrow q_4(\text{book}_y)$$

$$\text{read}_{x,y} \rightarrow q_5(\text{read}_{x,y})$$



# An Example Computation

- $Q = \{q_3, q_4, \dots\}$ ,  $\Sigma = \{\exists x|_2, \text{book}_y|_0, \dots\}$ ,  $Q_f = \{q_{12345}\}$

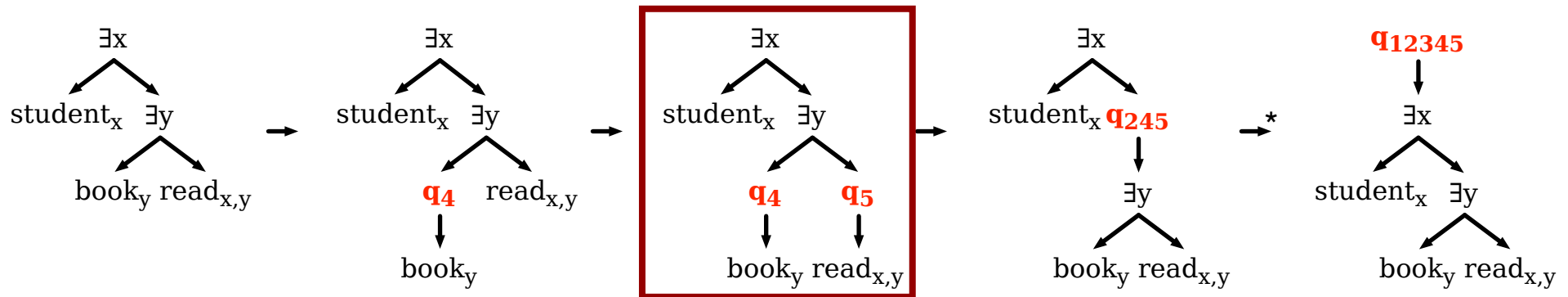
$$\exists x(q_3(x_1), q_{245}(x_2)) \rightarrow q_{12345}(\exists x(x_1, x_2))$$

$$\exists y(q_4(x_1), q_5(x_2)) \rightarrow q_{245}(\exists y(x_1, x_2))$$

$$\text{student}_x \rightarrow q_3(\text{student}_x)$$

$$\text{book}_y \rightarrow q_4(\text{book}_y)$$

$$\text{read}_{x,y} \rightarrow q_5(\text{read}_{x,y})$$



# An Example Computation

- $Q = \{q_3, q_4, \dots\}$ ,  $\Sigma = \{\exists x_{|2}, \text{book}_{y|0}, \dots\}$ ,  $Q_f = \{q_{12345}\}$

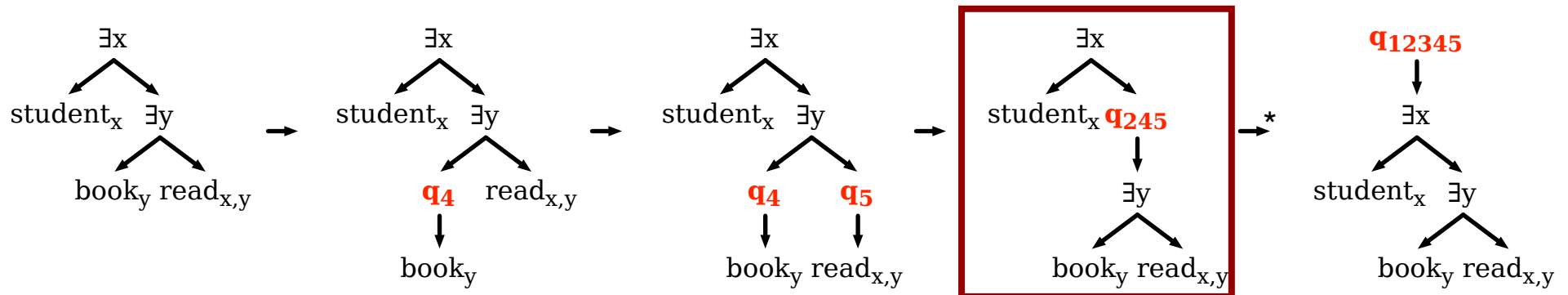
$$\exists x(q_3(x_1), q_{245}(x_2)) \rightarrow q_{12345}(\exists x(x_1, x_2))$$

$$\exists y(q_4(x_1), q_5(x_2)) \rightarrow q_{245}(\exists y(x_1, x_2))$$

$$\text{student}_x \rightarrow q_3(\text{student}_x)$$

$$\text{book}_y \rightarrow q_4(\text{book}_y)$$

$$\text{read}_{x,y} \rightarrow q_5(\text{read}_{x,y})$$



# An Example Computation

- $Q = \{q_3, q_4, \dots\}$ ,  $\Sigma = \{\exists x|_2, \text{book}_y|_0, \dots\}$ ,  $Q_f = \{q_{12345}\}$

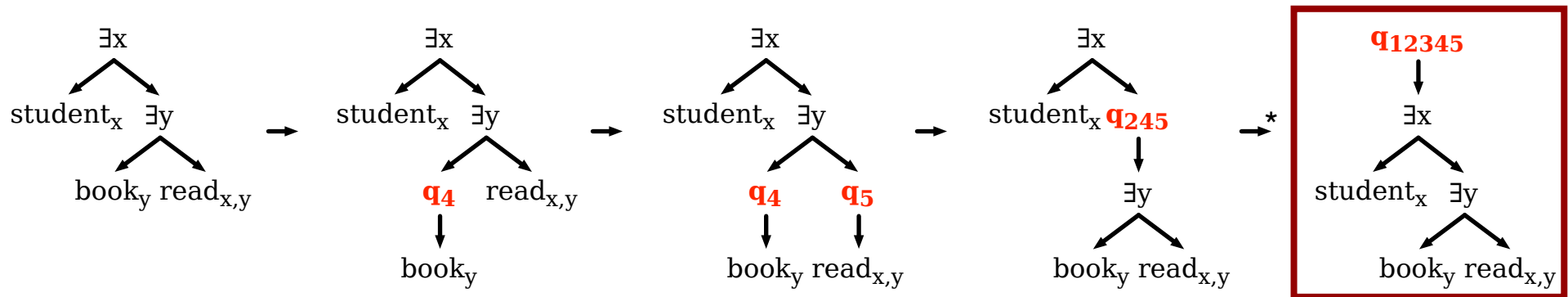
$$\exists x(q_3(x_1), q_{245}(x_2)) \rightarrow q_{12345}(\exists x(x_1, x_2))$$

$$\exists y(q_4(x_1), q_5(x_2)) \rightarrow q_{245}(\exists y(x_1, x_2))$$

$$\text{student}_x \rightarrow q_3(\text{student}_x)$$

$$\text{book}_y \rightarrow q_4(\text{book}_y)$$

$$\text{read}_{x,y} \rightarrow q_5(\text{read}_{x,y})$$



# Bottom-up Tree Automaton

- A tree  $t$  is **accepted** by an automaton  $A = \langle Q, \Sigma, Q_f, \Delta \rangle$  if
  - $t \rightarrow^* q(t)$
  - $q \in Q_f$
- **The language  $L(A)$**  of trees recognized by  $A$  is the set of trees accepted by  $A$ .

# Another Example

## ■ Automaton (rules)

$$\exists x(q_3(x_1), q_{245}(x_2)) \rightarrow q_{12345}(\exists x(x_1, x_2))$$

$$\text{student}_x \rightarrow q_3(\text{student}_x)$$

$$\exists y(q_4(x_1), q_{135}(x_2)) \rightarrow q_{12345}(\exists y(x_1, x_2))$$

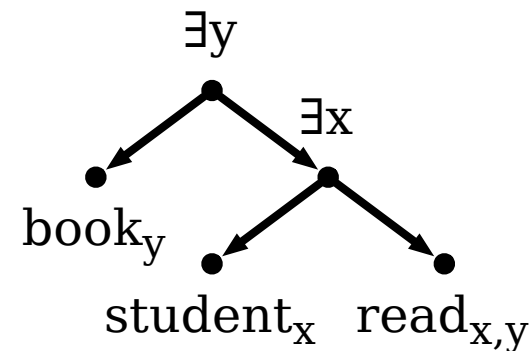
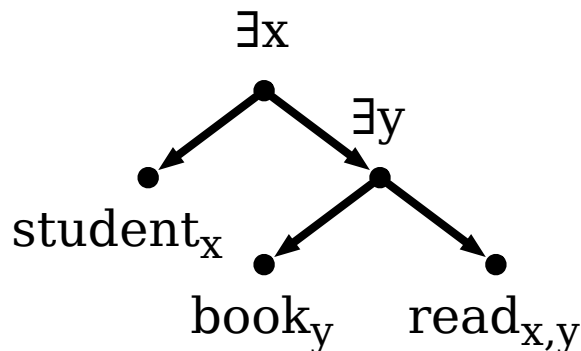
$$\text{book}_y \rightarrow q_4(\text{book}_y)$$

$$\exists y(q_4(x_1), q_5(x_2)) \rightarrow q_{245}(\exists y(x_1, x_2))$$

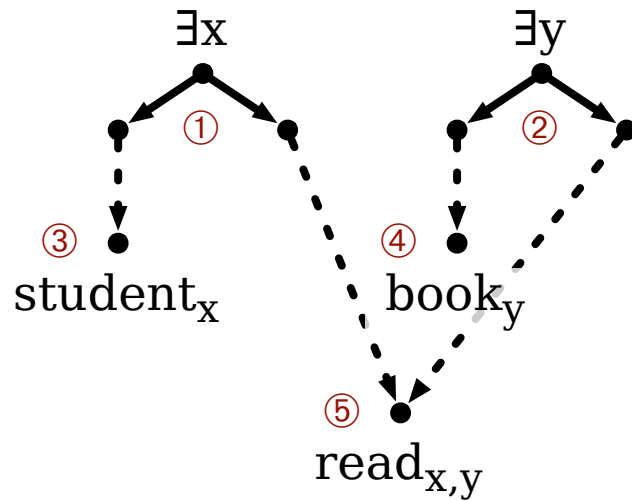
$$\text{read}_{x,y} \rightarrow q_5(\text{read}_{x,y})$$

$$\exists x(q_3(x_1), q_5(x_2)) \rightarrow q_{135}(\exists x(x_1, x_2))$$

## ■ Accepted Trees:



# Back to dominance charts



dominance chart	
{1,2,3,4,5}	1({3}, {2,4,5})
	2({4}, {1,3,5})
{2,4,5}	2({4}, {5})
{1,3,5}	1({3}, {5})

# Compare

## tree automaton

$\exists x(q_3(x_1), q_{245}(x_2)) \rightarrow q_{12345}(\exists x(x_1, x_2))$

$\exists y(q_4(x_1), q_{135}(x_2)) \rightarrow q_{12345}(\exists y(x_1, x_2))$

$\exists y(q_4(x_1), q_5(x_2)) \rightarrow q_{245}(\exists y(x_1, x_2))$

$\exists x(q_3(x_1), q_5(x_2)) \rightarrow q_{135}(\exists x(x_1, x_2))$

$\text{student}_x \rightarrow q_3(\text{student}_x)$

$\text{book}_y \rightarrow q_4(\text{book}_y)$

$\text{read}_{x,y} \rightarrow q_5(\text{read}_{x,y})$

## dominance chart

{1,2,3,4,5} 1({3}, {2,4,5})

2({4}, {1,3,5})

{2,4,5} 2({4}, {5})

{1,3,5} 1({3}, {5})

# Charts = Tree automata

- Dominance charts can be seen as (or translated into) tree automata
  - ... provided that the original dominance graph is hypernormally connected (why?)

# Charts = Tree automata

- The class of recognizable languages is closed under
  - Union
  - Complement
  - Intersection
- $\Rightarrow$  We can model certain inferences on the level of dominance charts by intersecting regular tree languages
  - Redundancy elimination
  - Weakest readings

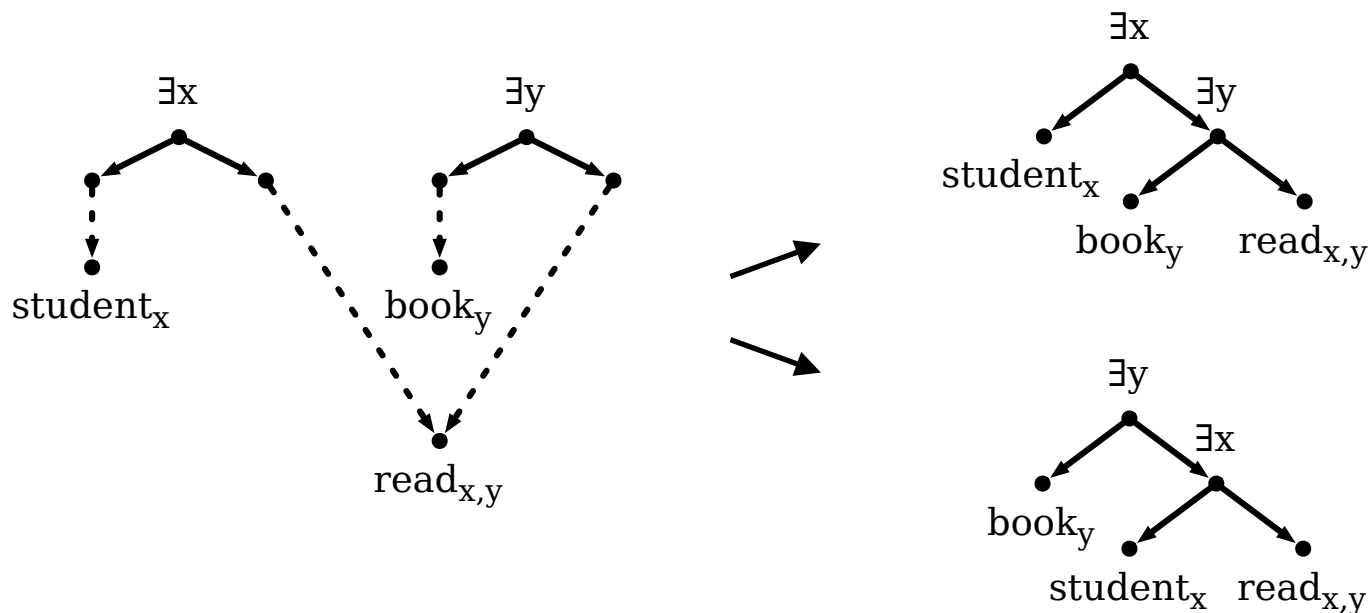
# Redundancy Elimination (very short)

# Ambiguity, revisited

- *A student reads a book*

(1)  $\exists x(\text{student}(x), \exists y(\text{book}(y), \text{read}(x, y)))$

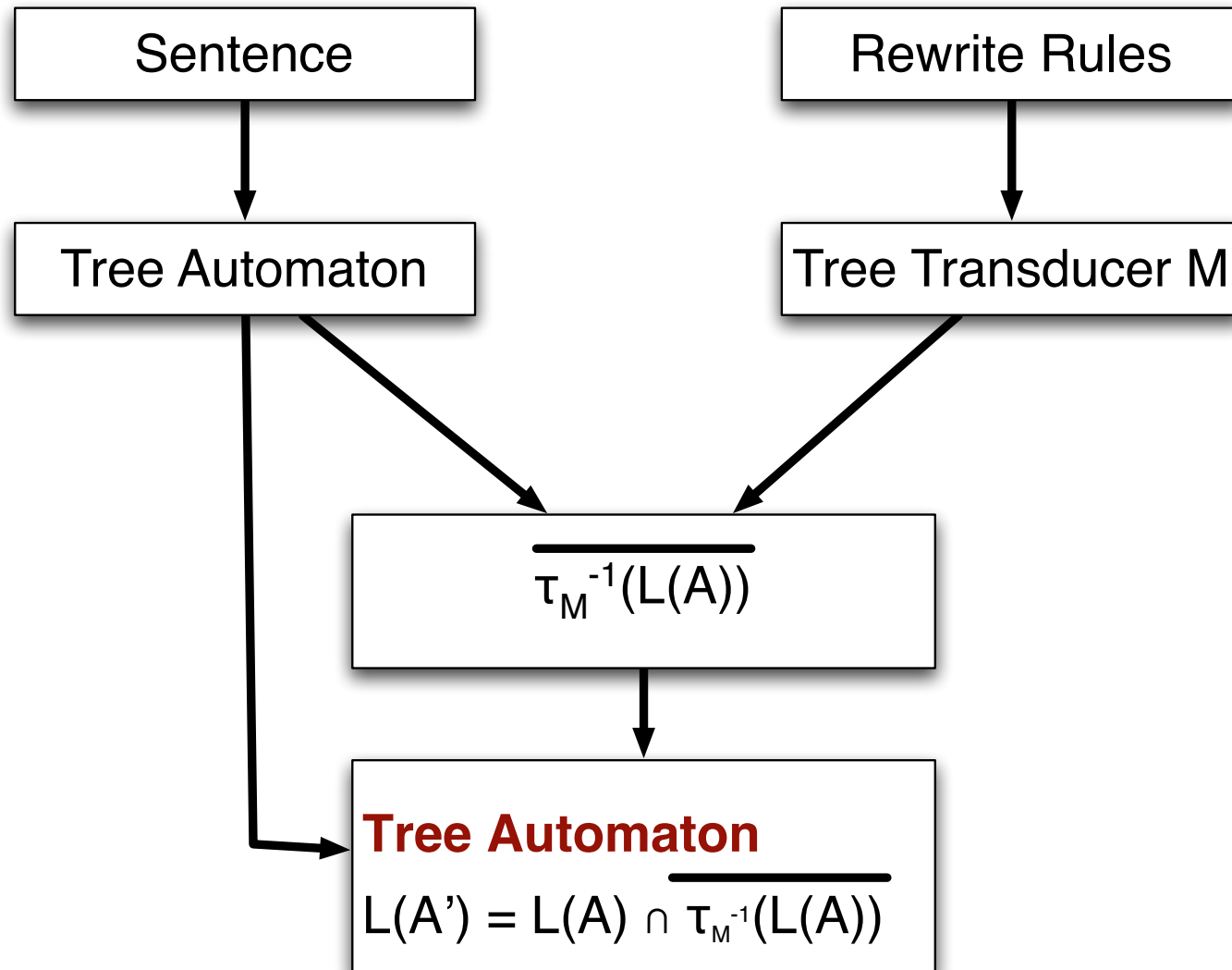
(2)  $\exists y(\text{book}(y), \exists x(\text{student}(x), \text{read}(x, y)))$



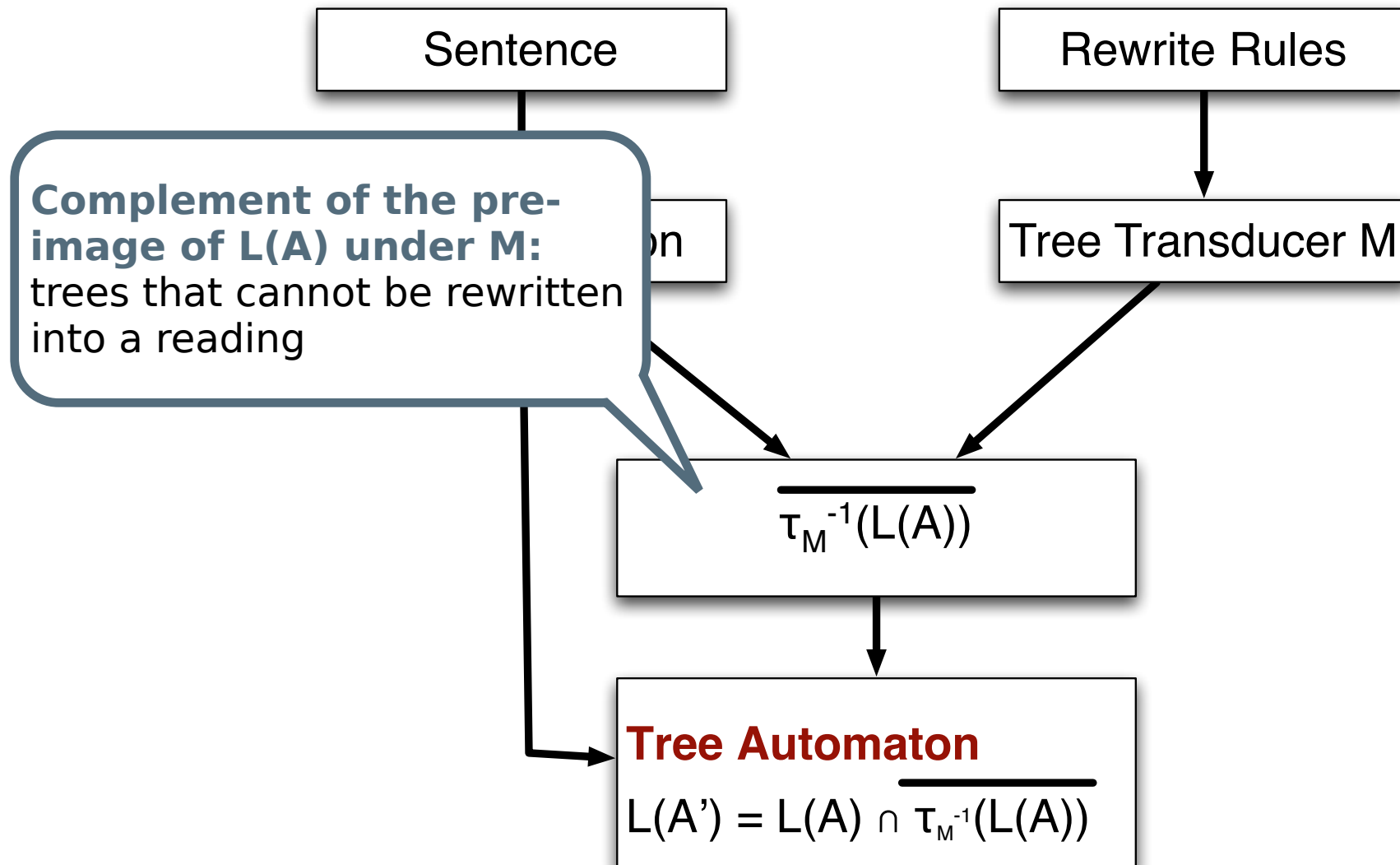
# Redundancy Elimination

- *A student reads a book*
  - (1)  $\exists x(\text{student}(x), \exists y(\text{book}(y), \text{read}(x, y)))$
  - (2)  $\exists y(\text{book}(y), \exists x(\text{student}(x), \text{read}(x, y)))$
- Readings (1) and (2) are logically equivalent!
- Basic idea:
  - Model the relation between (1) and (2) by rewrite rules
  - Translate the rewrite rules into a tree automaton
  - Redundancy elimination = Intersection of regular tree languages

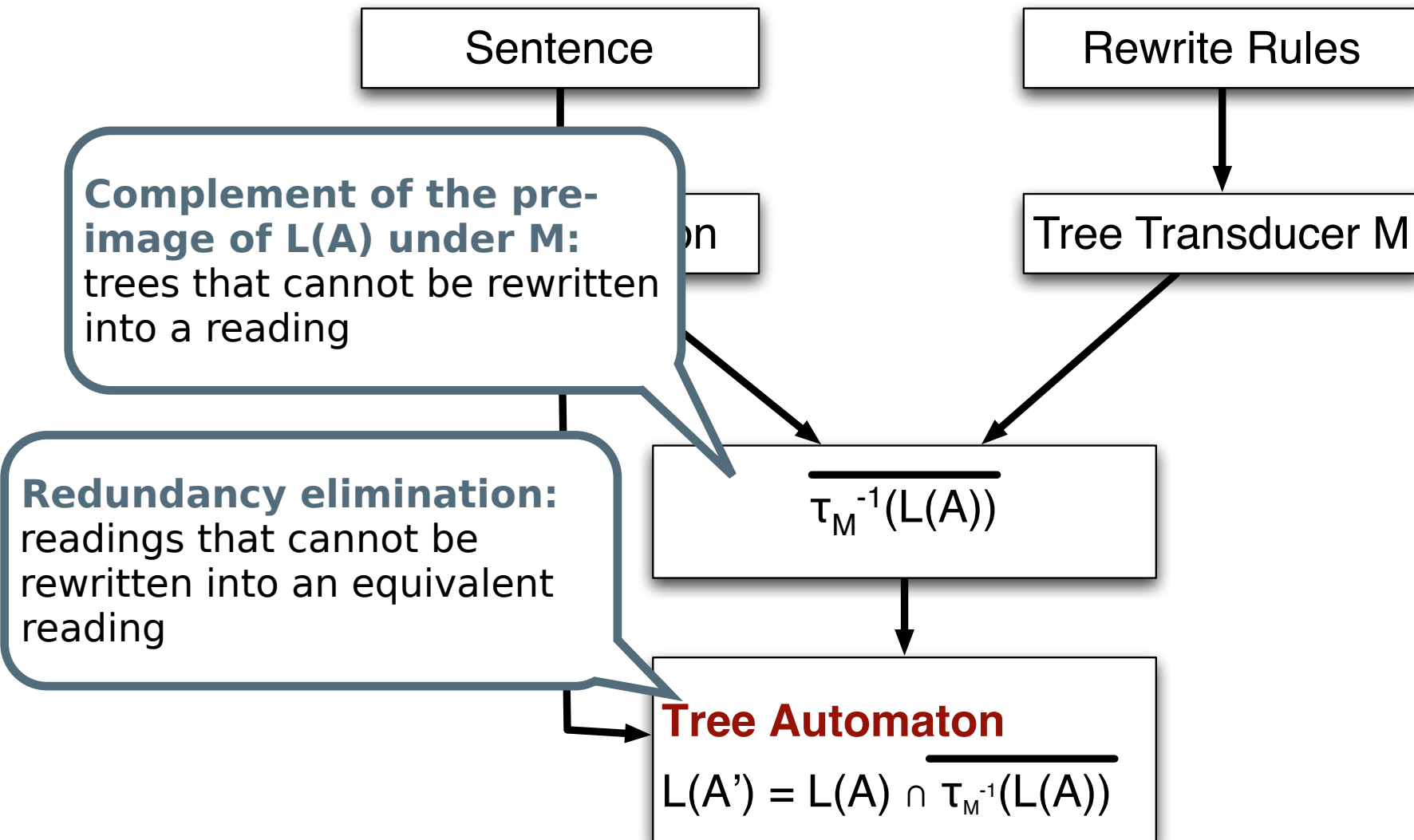
# Redundancy Elimination



# Redundancy Elimination



# Redundancy Elimination



# References

- Alexander Koller and Stefan Thater (2005). The evolution of dominance constraint solvers [\[PDF\]](#). In Proceedings of the ACL Workshop on Software.
- Alexander Koller and Stefan Thater (2010). Computing Weakest Readings [\[PDF\]](#). In Proceedings of the 48<sup>th</sup> Annual Meeting of the Association for Computational Linguistics.
- Hubert Comon, Max Dauchet, Remi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, Marc Tommasi. Tree Automata, Techniques and Applications. [\[PDF\]](#)