

# An Extended GHKM Algorithm for Inducing $\lambda$ -SCFG

Peng Li, Yang Liu and Maosong Sun

THUNLP&CSS

Tsinghua University, China



# Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

# Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

# Semantic Parsing

- Semantic parsing: mapping a natural language sentence into its computer executable meaning representation

**NL** : Every boy likes a star



**MR**:  $\forall x.(boy(x) \rightarrow \exists y (human(y) \wedge pop(y) \wedge like(x, y)))$

# Related Work

- **Hand-build systems** (e.g., Woods et al., 1972; Warren & Pereira, 1982)
- **Learning for semantic parsing**
  - **Supervised methods** (e.g., Wong & Mooney, 2007; Lu et al., 2008)
  - **Semi-supervised methods** (e.g., Kate & Mooney, 2007)
  - **Unsupervised methods** (e.g., Poon & Domingos, 2009 & 2010; Goldwasser et al., 2011)

# Related Work

- Hand-build systems (e.g., Woods et al., 1972; Warren & Pereira, 1982)
- Learning for semantic parsing
  - Supervised methods (e.g., Wong & Mooney, 2007; Lu et al., 2008)
  - Semi-supervised methods (e.g., Kate & Mooney, 2007)
  - Unsupervised methods (e.g., Poon & Domingos, 2009 & 2010; Goldwasser et al., 2011)

# Supervised Methods

- Inductive logic programming based methods (e.g., Zelle & Mooney, 1996; Tang & Mooney, 2001)
- String kernel based methods (e.g., Kate & Mooney, 2006)
- Grammar based methods
  - PCFG (e.g., Ge & Mooney, 2005)
  - SCFG (e.g., Wong & Mooney, 2006 & 2007)
  - CCG (e.g., Zettlemoyer & Collins, 2005 & 2007; Kwiatkowski et al., 2010 & 2011)
  - Hybrid tree (e.g., Lu et al., 2008)
  - Tree transducer (Jones et al., 2012)

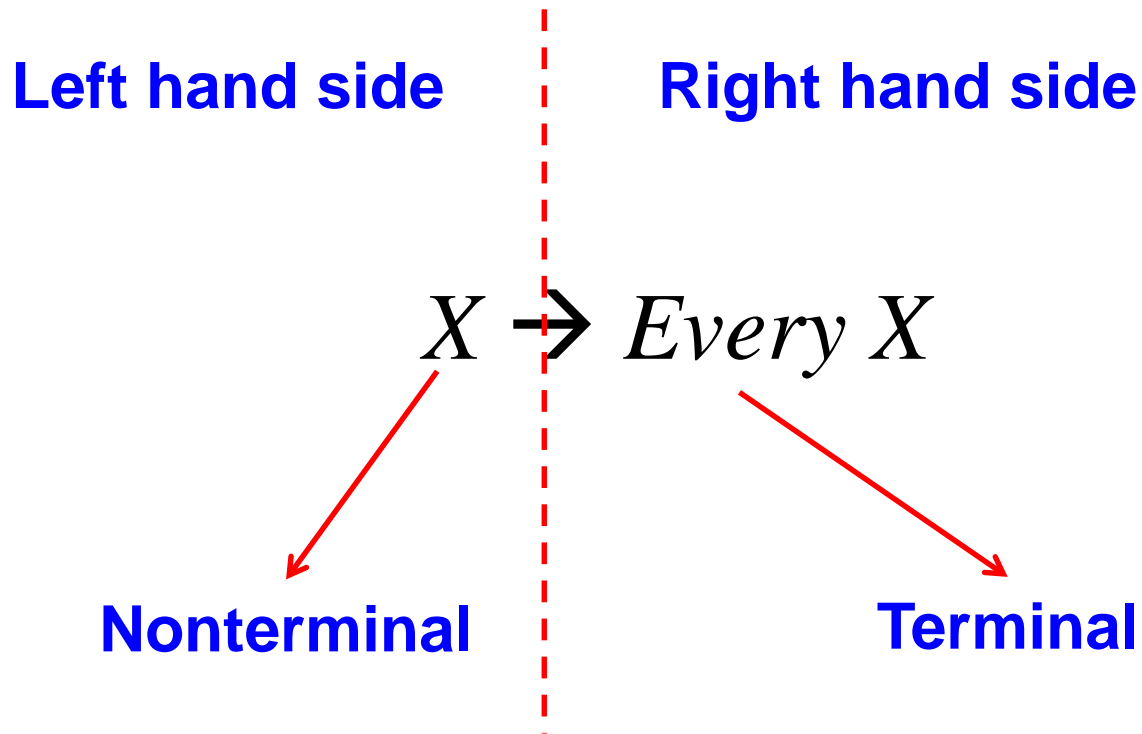
# Supervised Methods

- Inductive logic programming based methods (e.g., Zelle & Mooney, 1996; Tang & Mooney, 2001)
- String kernel based methods (e.g., Kate & Mooney, 2006)
- Grammar based methods
  - PCFG (e.g., Ge & Mooney, 2005)
  - SCFG (e.g., Wong & Mooney, 2006 & 2007)
  - CCG (e.g., Zettlemoyer & Collins, 2005 & 2007; Kwiatkowski et al., 2010 & 2011)
  - Hybrid tree (e.g., Lu et al., 2008)
  - Tree transducer (Jones et al., 2012)



# Context Free Grammar (CFG)

- A formal grammar in which every production rule is of the following form



# Context Free Grammar (CFG)

- Derivation example

## Derivation

$S \rightarrow X$   
 $\rightarrow \textit{Every } X$   
 $\rightarrow \textit{Every } X_1 X_2$   
 $\rightarrow \textit{Every boy } X_2$   
 $\rightarrow \textit{Every boy } X_2 \textit{ a star}$   
 $\rightarrow \textit{Every boy likes a star}$

## CFG Rules

$r_1: S \rightarrow X$   
 $r_2: X \rightarrow \textit{Every } X$   
 $r_3: X \rightarrow X_1 X_2$   
 $r_4: X \rightarrow \textit{boy}$   
 $r_5: X \rightarrow \textit{X a star}$   
 $r_6: X \rightarrow \textit{likes}$

# Synchronous Context Free Grammar (SCFG)

Left hand side

Right hand  
side 1

Right hand  
side 2

$X \rightarrow \langle \textit{Every } X_1, \textit{ 每个 } X_1 \rangle$

Rewritten synchronously

One nonterminal

# Synchronous Context Free Grammar (SCFG)

- Two strings can be generated synchronously

$S \rightarrow \langle X, X \rangle$

$\rightarrow \langle \textit{Every X}, \textit{每个 X} \rangle$

$\rightarrow \langle \textit{Every X}_1 X_2, \textit{每个 X}_1 X_2 \rangle$

.....

$\rightarrow \langle \textit{Every boy likes a star},$   
 $\textit{每个男孩都喜欢一个明星} \rangle$

**How to use SCFG to handle logical forms?**

# $\lambda$ -calculus

- A formal system in mathematical logic for expressing computation by way of **variable binding** and **substitution**
  - $\lambda$ -expression:  $\lambda x.\lambda y.borders(y, x)$
  - $\beta$ -conversion: bound variable substitution
$$\lambda x.\lambda y.borders(y, x)(texas) = \lambda y.borders(y, texas)$$
  - $\alpha$ -conversion: bound variable renaming
$$\lambda x.\lambda y.borders(y, x) = \lambda z.\lambda y.borders(y, z)$$

# $\lambda$ -SCFG: SCFG+ $\lambda$ -calculus

- Reducing semantic parsing problem to SCFG parsing problem
- Using  $\lambda$ -calculus to handle semantic specific phenomenon
- Rule example
  - $X \rightarrow \langle \textit{Every } X_1, \lambda f. \forall x (f(x)) \triangleleft X_1 \rangle$

(Wong & Mooney, 2007)

# $\lambda$ -SCFG: SCFG+ $\lambda$ -calculus

**NL** : Every boy likes a star

$$r_1: S \rightarrow \langle X_1, X_1 \rangle$$

$$r_2: X \rightarrow \langle \text{Every } X_1, \lambda f. \forall x (f(x)) \triangleleft X_1 \rangle$$

$$r_3: X \rightarrow \langle X_1 X_2, \lambda f. \lambda g. \lambda x. f(x) \rightarrow g(x) \triangleleft X_1 \triangleleft X_2 \rangle$$

$$r_4: X \rightarrow \langle \text{boy}, \lambda x. \text{boy}(x) \rangle$$

$$r_5: X \rightarrow \langle X_1, \lambda f. \lambda x. \exists y (f(x, y)) \triangleleft X_1 \rangle$$

$$r_6: X \rightarrow \langle X_1 \text{ a star}, \lambda f. \lambda x. \lambda y. \text{human}(y) \wedge \text{pop}(y) \wedge f(x, y) \triangleleft X_1 \rangle$$

$$r_7: X \rightarrow \langle \text{like}, \lambda x. \lambda y. \text{like}(x, y) \rangle$$

$$\langle S_1, S_1 \rangle$$

$$\rightarrow \langle X_2, X_2 \rangle$$

$$\rightarrow \langle \text{Every } X_3, \lambda f. \forall x. (f(x)) \triangleleft X_3 \rangle$$

$$\rightarrow \langle \text{Every } X_4 X_5, \lambda f. \lambda g. \forall x. (f(x) \rightarrow g(x)) \triangleleft X_4 \triangleleft X_5 \rangle$$

$$\rightarrow \langle \text{Every boy } X_5, \lambda g. \forall x. (\text{boy}(x) \rightarrow g(x)) \triangleleft X_5 \rangle$$

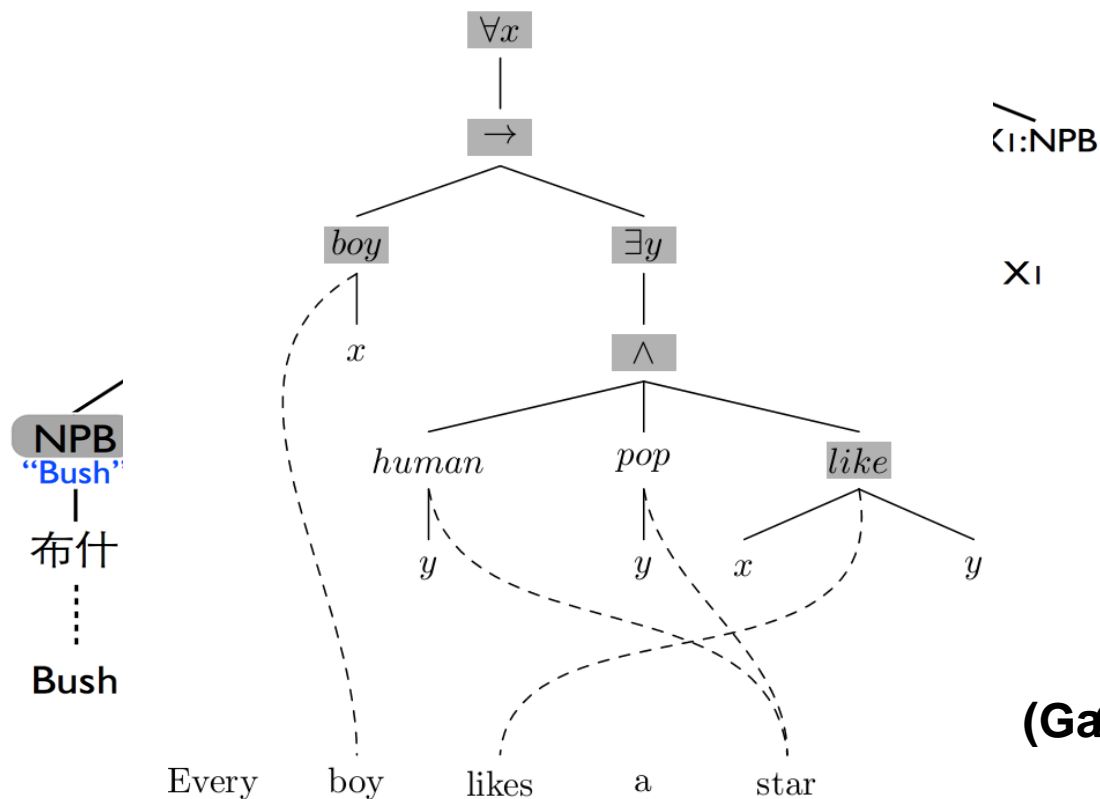
$$\rightarrow \langle \text{Every boy } X_6, \lambda f. \forall x. (\text{boy}(x) \rightarrow \exists y (f(x, y))) \triangleleft X_6 \rangle$$

$$\rightarrow \langle \text{Every boy } X_7 \text{ a star}, \lambda f. \forall x. (\text{boy}(x) \rightarrow \exists y (\text{human}(y) \wedge \text{pop}(y) \wedge f(x, y))) \triangleleft X_7 \rangle$$

$$\rightarrow \langle \text{Every boy likes a star}, \forall x. (\text{boy}(x) \rightarrow \exists y (\text{human}(y) \wedge \text{pop}(y) \wedge \text{like}(x, y))) \rangle$$

# GHKM

- The GHKM algorithm extracts STSG rules from aligned tree-string pairs





# Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

# Overview

**NL** : Every boy likes a star

**MR**:  $\forall x.(boy(x) \rightarrow \exists y (human(y) \wedge pop(y) \wedge like(x, y)))$



**GHKM Rule Extractor**



$X \rightarrow \langle \textit{Every } X_1, \lambda f.\forall x (f(x)) \triangleleft X_1 \rangle$

$X \rightarrow \langle \textit{boy}, \lambda x.boy(x) \rangle$



**Parameter estimation**

**Semantic Parser**

# Rule Extraction Algorithm

- Outline

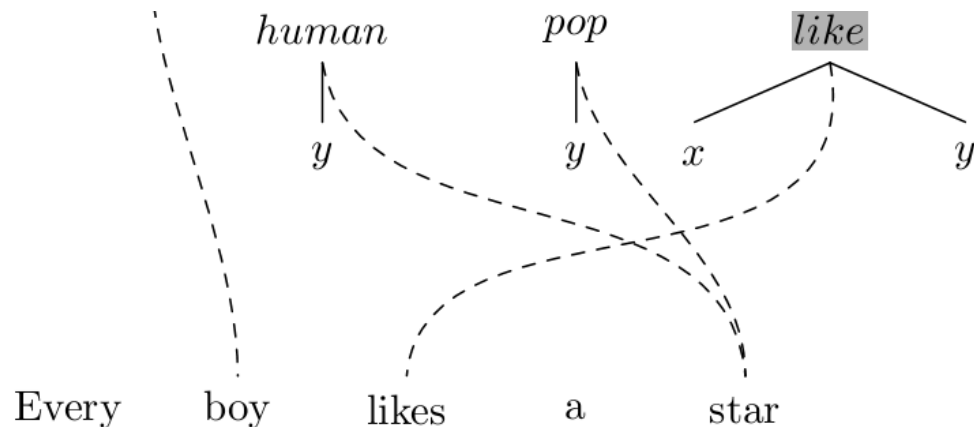
1. Building training examples
  1. Transforming logical forms to trees
  2. Aligning trees with sentences
2. Identifying frontier nodes
3. Extracting minimal rules
4. Extracting composed rules

# Building Training Examples



**NL** : Every boy likes a star

**MR** :  $\forall x.(boy(x) \rightarrow \exists y (human(y) \wedge pop(y) \wedge like(x, y)))$

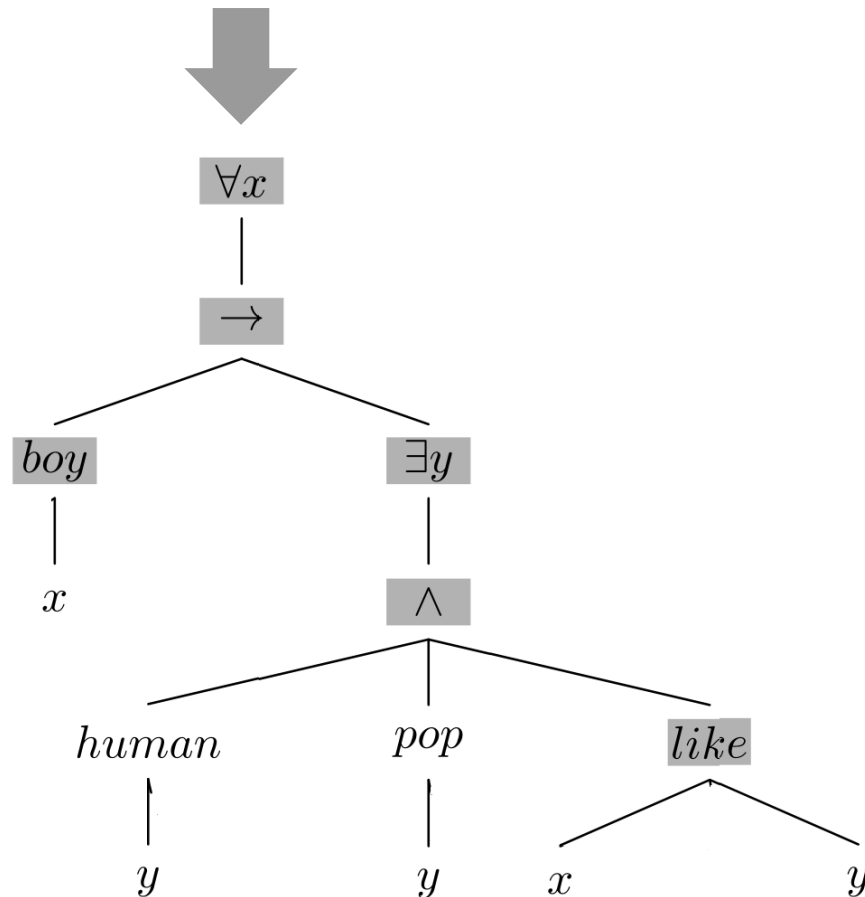


# Building Training Examples

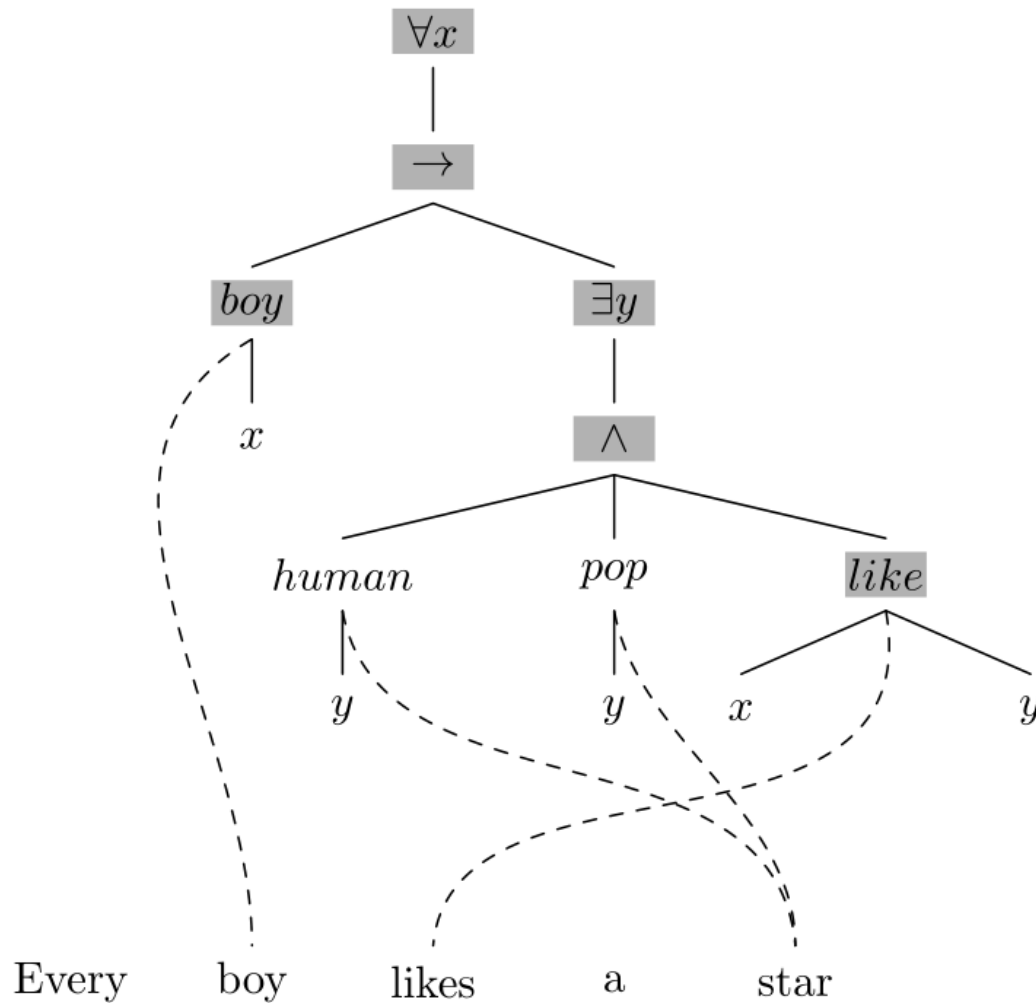
type	expression	tree
variable	$x$	$(x)$
atomic predicate	$p(x_1, \dots, x_n)$	$(p(x_1) \dots (x_n))$
complex predicate	$p(P_1(x), \dots, P_n(x))$	$(p(P_1(x)) \dots (P_n(x)))$
universal quantifier	$\forall x P(x)$	$(\forall x (P(x)))$
existential quantifier	$\exists x P(x)$	$(\exists x (P(x)))$
$\lambda$ operator	$\lambda x.P(x)$	$(\lambda x (P(x)))$
negation operator	$\neg P(x)$	$(\neg (P(x)))$
conjunction operator	$P_1(x) \wedge \dots \wedge P_n(x)$	$(\wedge (P_1(x)) \dots (P_n(x)))$
disjunction operator	$P_1(x) \vee \dots \vee P_n(x)$	$(\vee (P_1(x)) \dots (P_n(x)))$
implication operator	$P_1(x) \rightarrow P_2(x)$	$(\rightarrow (P_1(x)) (P_2(x)))$
biconditional operator	$P_1(x) \leftrightarrow P_2(x)$	$(\leftrightarrow (P_1(x)) (P_2(x)))$

# Building Training Examples

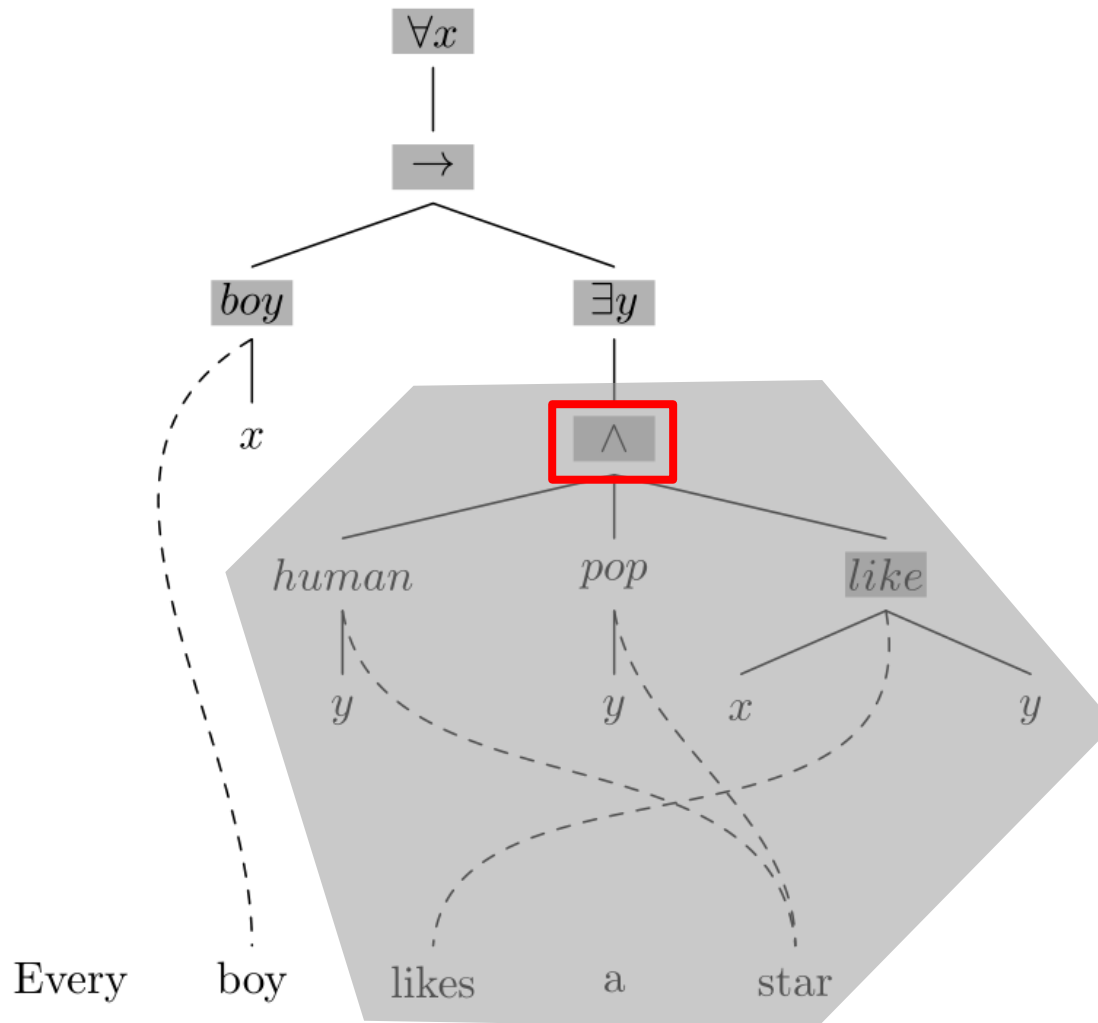
$$\forall x.(boy(x) \rightarrow \exists y (human(y) \wedge pop(y) \wedge like(x, y)))$$



# Building Training Examples

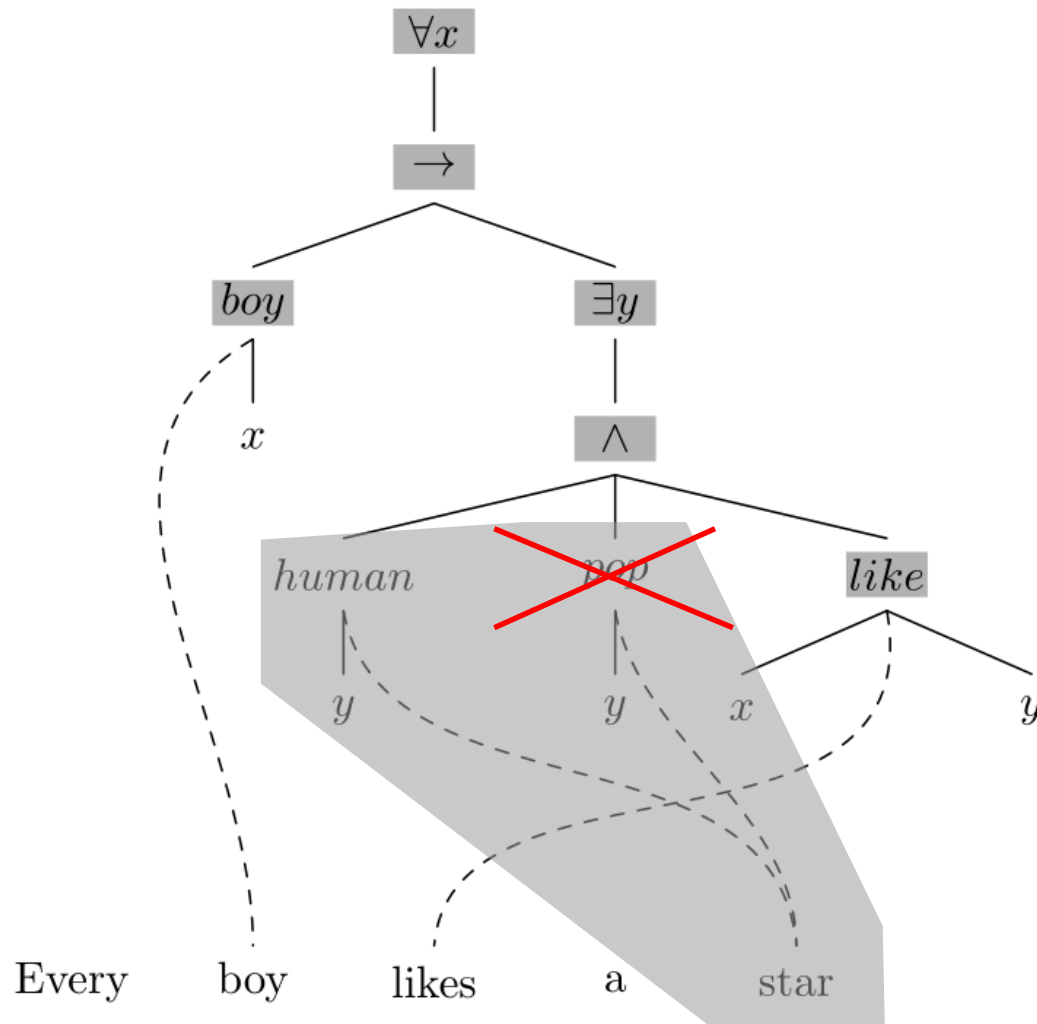


# Identifying Frontier Nodes

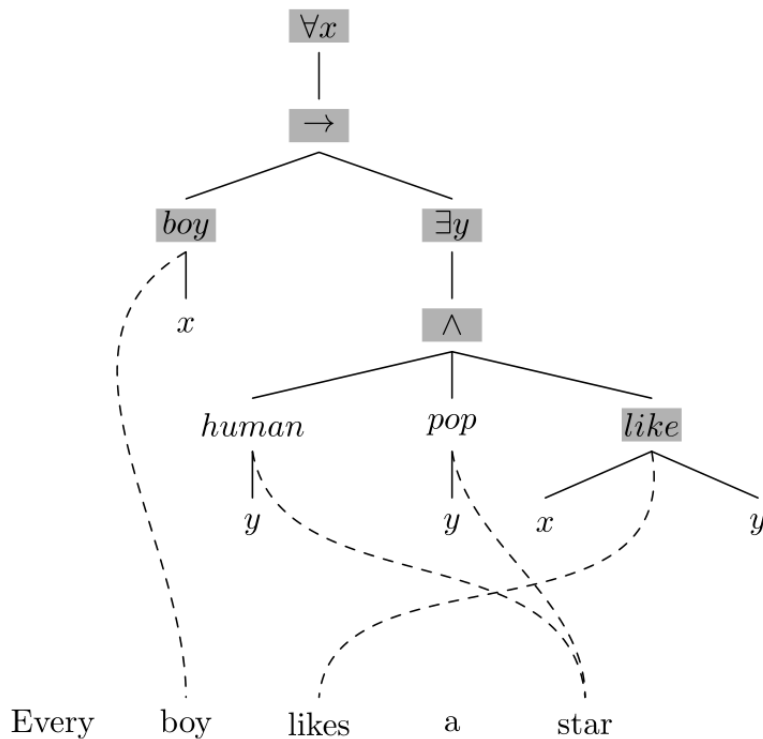




# Identifying Frontier Nodes



# Extracting minimal rules



- $\forall x: X \rightarrow \langle \text{Every } X_1, \lambda f. \forall x (f(x)) \triangleleft X_1 \rangle$
- $\rightarrow: X \rightarrow \langle X_1 X_2, \lambda f. \lambda g. \lambda x. f(x) \rightarrow g(x) \triangleleft X_1 \triangleleft X_2 \rangle$
- $\text{boy}: X \rightarrow \langle \text{boy}, \lambda x. \text{boy}(x) \rangle$
- $\exists y: X \rightarrow \langle X_1, \lambda f. \lambda x. \exists y (f(x, y)) \triangleleft X_1 \rangle$
- $\wedge: X \rightarrow \langle X_1 \text{ a star},$   
 $\lambda f. \lambda x. \lambda y. \text{human}(y) \wedge \text{pop}(y) \wedge f(x, y) \triangleleft X_1 \rangle$
- $\text{like}: X \rightarrow \langle \text{like}, \lambda x. \lambda y. \text{like}(x, y) \rangle$

# Composed Rule Extraction

$$X \rightarrow \langle X_1 X_2, \lambda f. \lambda g. \lambda x. f(x) \rightarrow g(x) \triangleleft X_1 \triangleleft X_2 \rangle$$

$$X \rightarrow \langle \textit{boy}, \lambda x. \textit{boy}(x) \rangle$$

$$+ X \rightarrow \langle X_1, \lambda f. \lambda x. \exists y (f(x, y)) \triangleleft X_1 \rangle$$

---

$$= X \rightarrow \langle \textit{boy} X_1, \lambda f. \lambda x. \textit{boy}(x) \rightarrow \exists y (f(x, y)) \triangleleft X_1 \rangle$$

# Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

# Modeling

- Log-linear model + MERT training

$$w(D) = \prod_{i=1}^3 \prod_{r \in D} h_i(r)^{\lambda_i} \times h_4(D)^{\lambda_4} \times h_5(D)^{\lambda_5}$$

$$h_1(X \rightarrow \langle s, e \rangle) = p(e|s) \quad h_4(X \rightarrow \langle s, e \rangle) = p_s(e(D))$$

$$h_2(X \rightarrow \langle s, e \rangle) = p_{lex}(s|e) \quad h_5(X \rightarrow \langle s, e \rangle) = \exp(|D|)$$

$$h_3(X \rightarrow \langle s, e \rangle) = p_{lex}(e|s)$$

- Target

$$\hat{e} = e \left( \underset{D \text{ s.t. } s(D) \equiv s}{\operatorname{argmax}} w(D) \right)$$

# Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

# Experiments

- Dataset: GEOQUERY
  - 880 English questions with corresponding Prolog logical forms



*Which rivers run through the states bordering Texas?*

Semantic Parsing

```
answer(traverse(next_to(stateid('texas'))))
```

(Kate & Wong, ACL 2010 Tutorial)

Arkansas, Canadian, Cimarron,  
Gila, Mississippi, Rio Grande ...

Answer

Query



# Experiments

- Dataset: GEOQUERY
  - 880 English questions with corresponding Prolog logical forms
- Evaluation metrics

$$\textit{precision} = \frac{|C|}{|G|}, \textit{recall} = \frac{|C|}{|T|},$$

$$\textit{F-measure} = \frac{2 \cdot \textit{precision} \cdot \textit{recall}}{\textit{precision} + \textit{recall}}$$



# Experiments

System	P	R	F
Independent Test Set			
Z&C 2005	<b>96.3</b>	79.3	87.0
Z&C 2007	95.5	83.2	88.9
Kwiatkowksi, <i>et al.</i> (2010)	94.1	85.0	89.3
Cross Validation Results			
Kate <i>et al.</i> (2005)	89.0	54.1	67.3
Wong and Mooney (2006)	87.2	74.8	80.5
Kate and Mooney (2006)	93.3	71.7	81.1
Lu <i>et al.</i> (2008)	89.3	81.5	85.2
Ge and Mooney (2005)	95.5	77.2	85.4
Wong and Mooney (2007)	92.0	86.6	89.2
<i>this work</i>	<b>93.0</b>	<b>87.6</b>	<b>90.2</b>

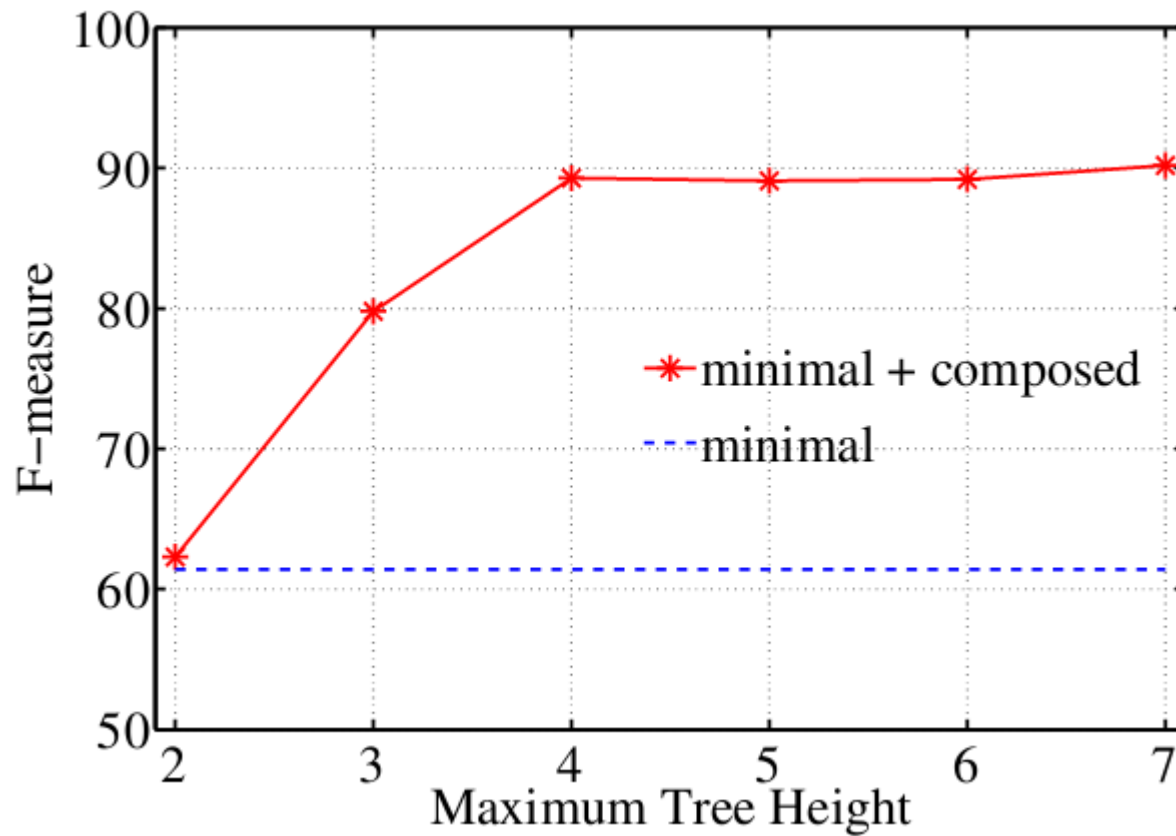
# Experiments

- F-measure for different languages

System	en	ge	el	th
Wong and Mooney (2006)	77.7	74.9	78.6	75.0
Lu <i>et al.</i> (2008)	81.0	68.5	74.6	76.7
Kwiatkowski, <i>et al.</i> (2010)	82.1	<b>75.0</b>	73.7	66.4
Jones <i>et al.</i> (2005)	79.3	74.6	75.4	<b>78.2</b>
<i>this work</i>	<b>84.2</b>	74.6	<b>79.4</b>	76.7

\* en - English, ge - German, el - Greek, th - Thai

# Experiments



# Advantages

- Feasible to extract rules with varying granularities in a principled way
  - The widely used dataset only has 880 training examples
- Alleviating the data sparseness problem
  - Treating atomic logical form tokens as tree nodes instead of context free grammar (CFG) production
- Robust to the nonisomorphism between NL sentences and logical forms

# Outline

- Background
- Rule extraction algorithm
- Modeling
- Experiments
- Conclusion

# Conclusion

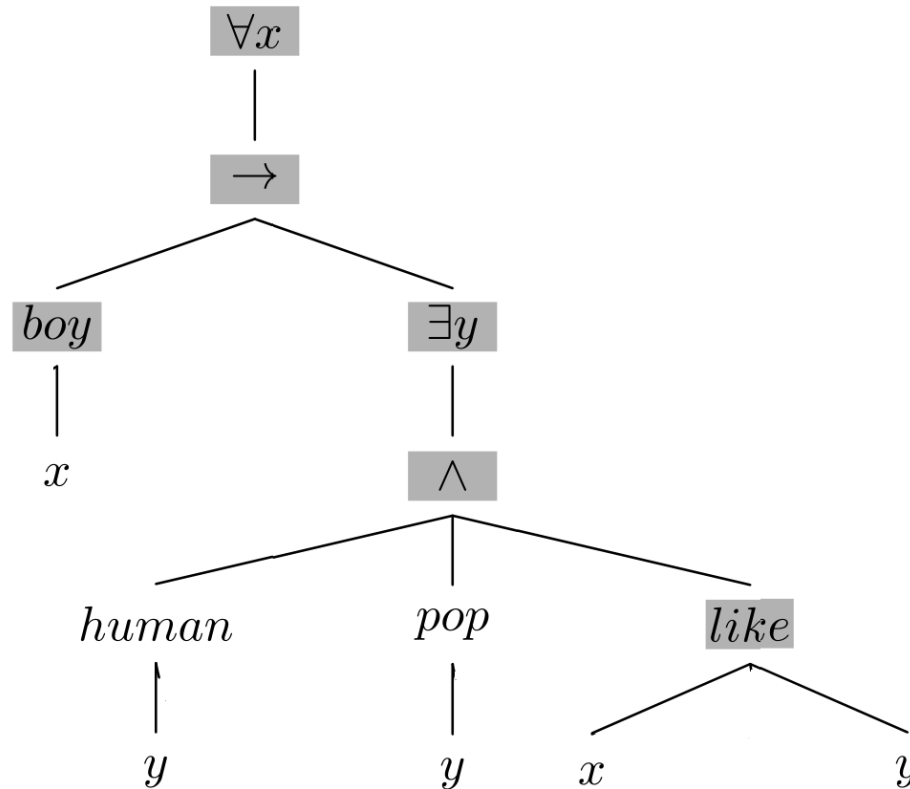
- We have presented an extended GHKM algorithm for inducing  $\lambda$ -SCFG and achieved state-of-the-art performance
- Future work
  - Better alignment model
  - Investigate tree binarization to further improve rule coverage
  - Use EM or Monte Carlo methods to better estimate  $\lambda$ -SCFG rule probabilities

**Welcome for discussion  
in poster session!**

# Backup Slides



# Building Training Examples

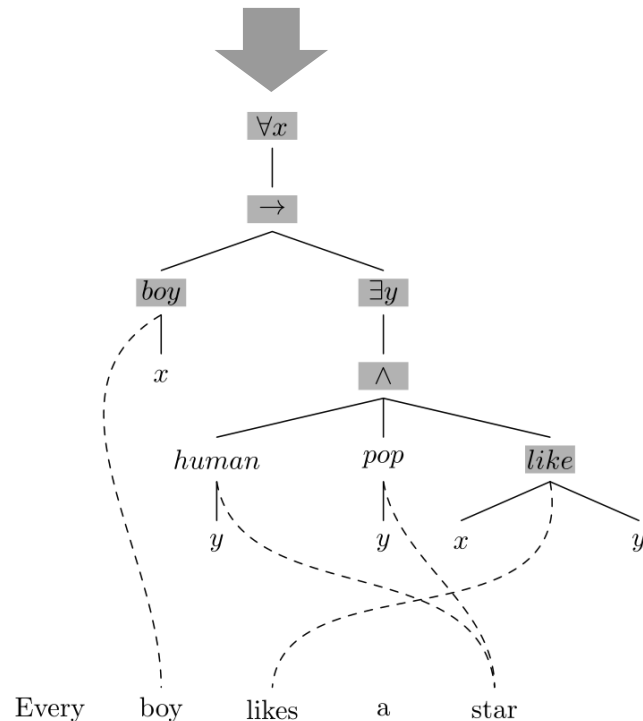


*boy human pop like*

# Building Training Examples

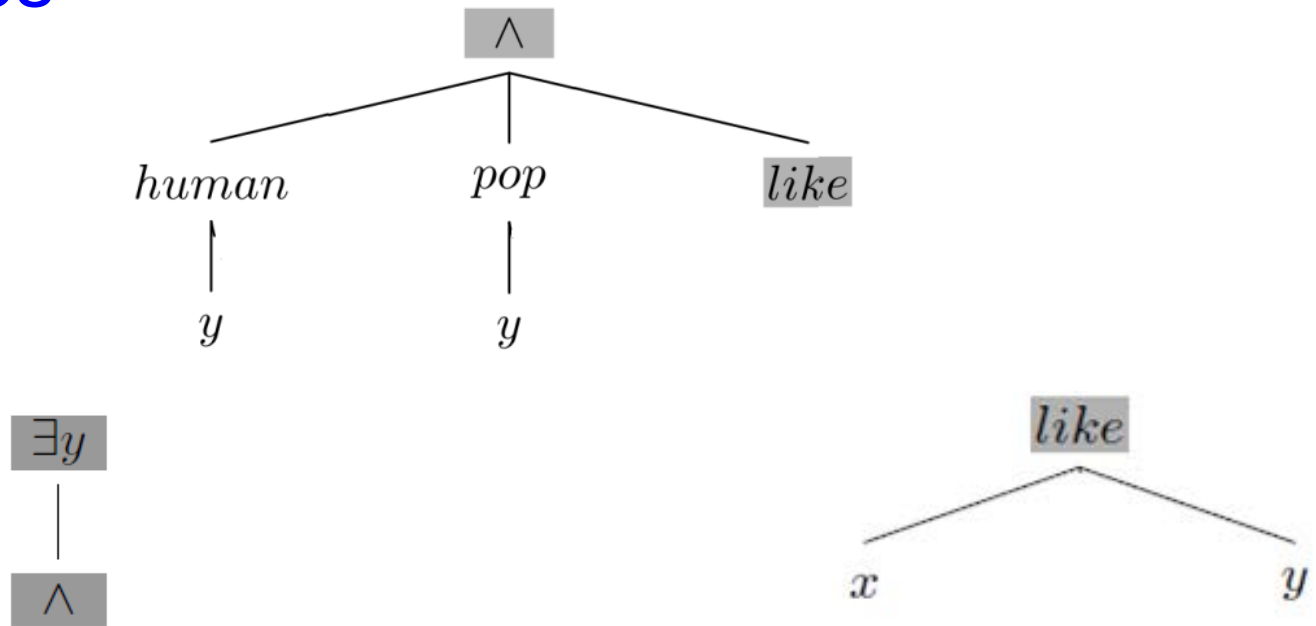
*boy human pop like*

**Every boy likes a star**

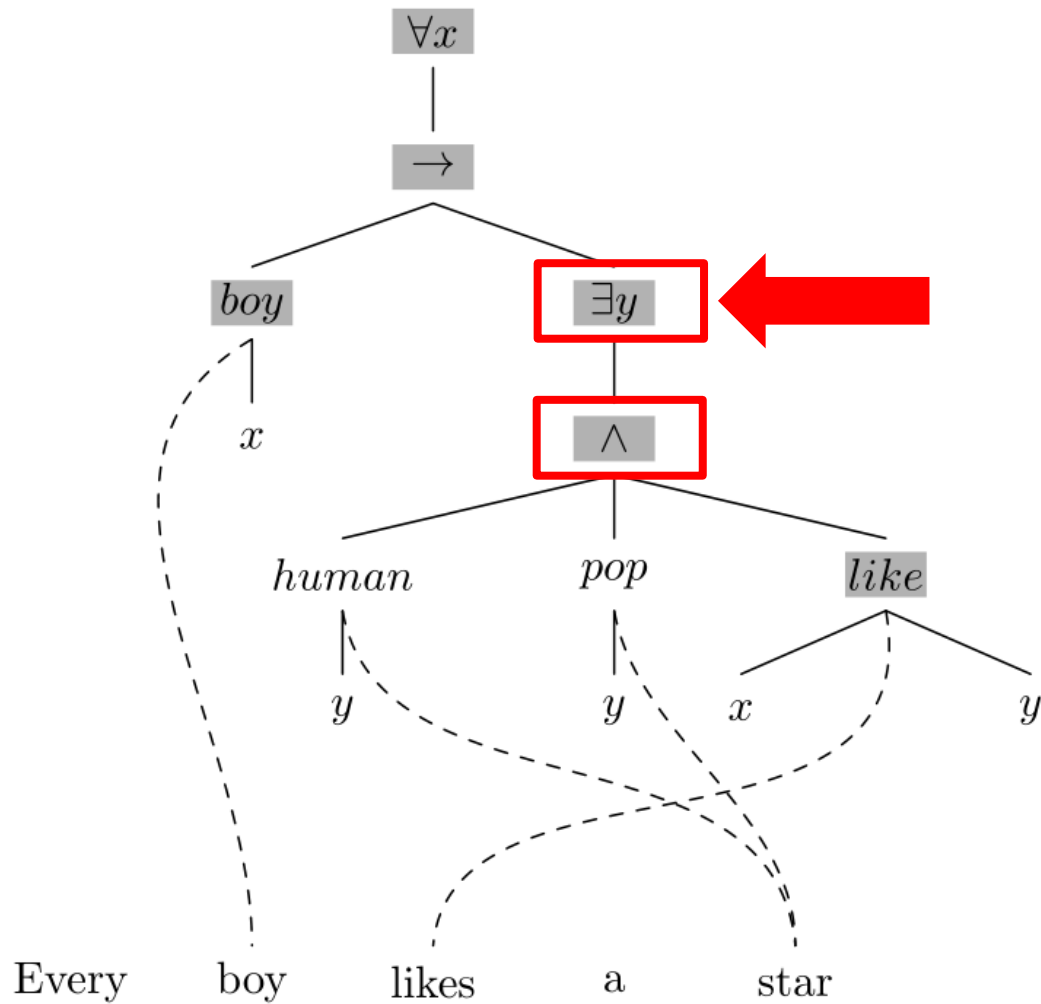


# Identifying Minimal Frontier Tree

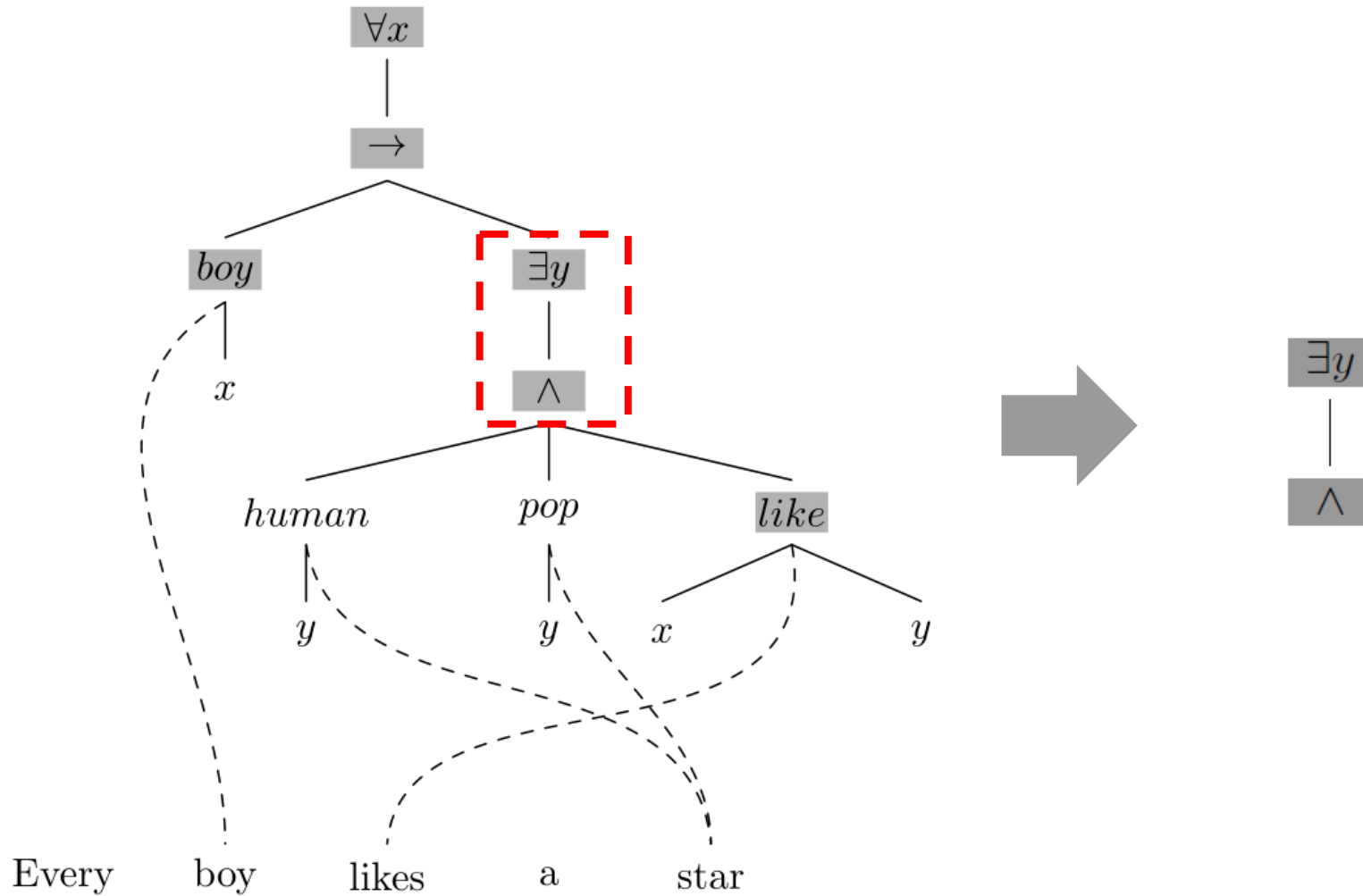
- Minimal frontier tree: all the nodes in the tree are **non-frontier** node **except** the **root** and **leaves**



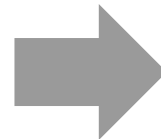
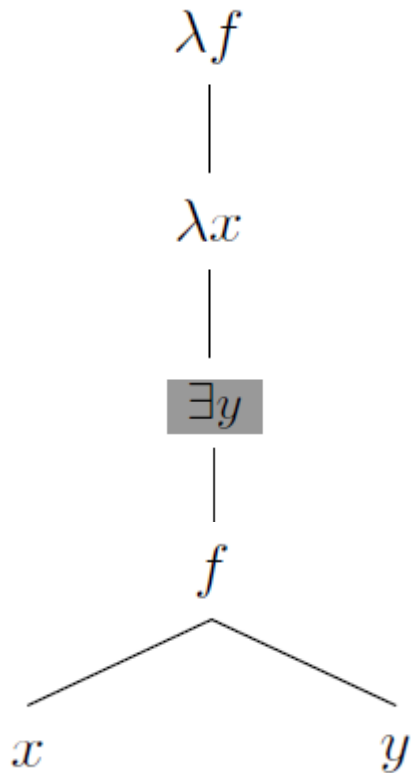
# Identifying Minimal Frontier Tree



# Identifying Minimal Frontier Tree

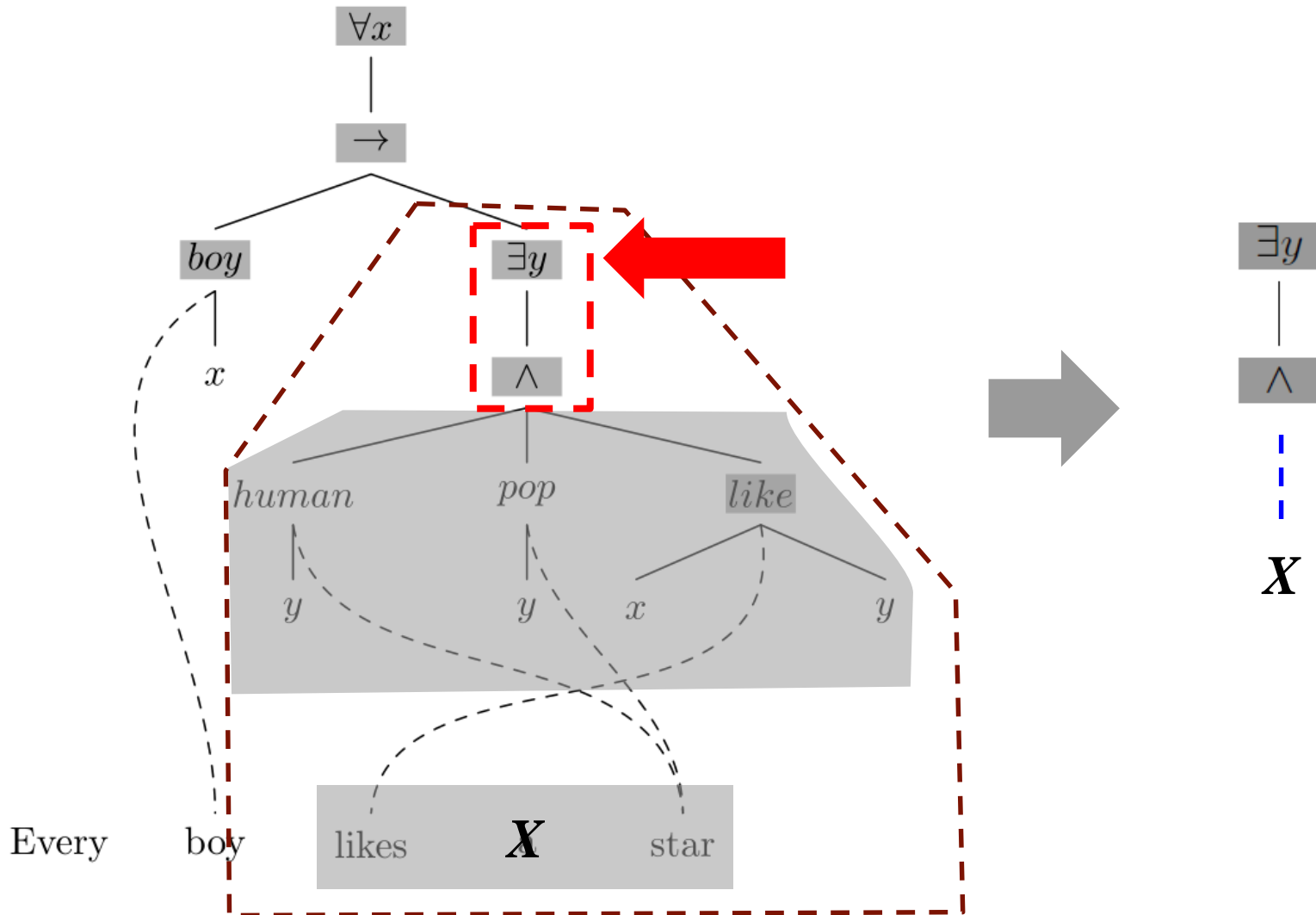


# Minimal Frontier Tree to $\lambda$ -Expression

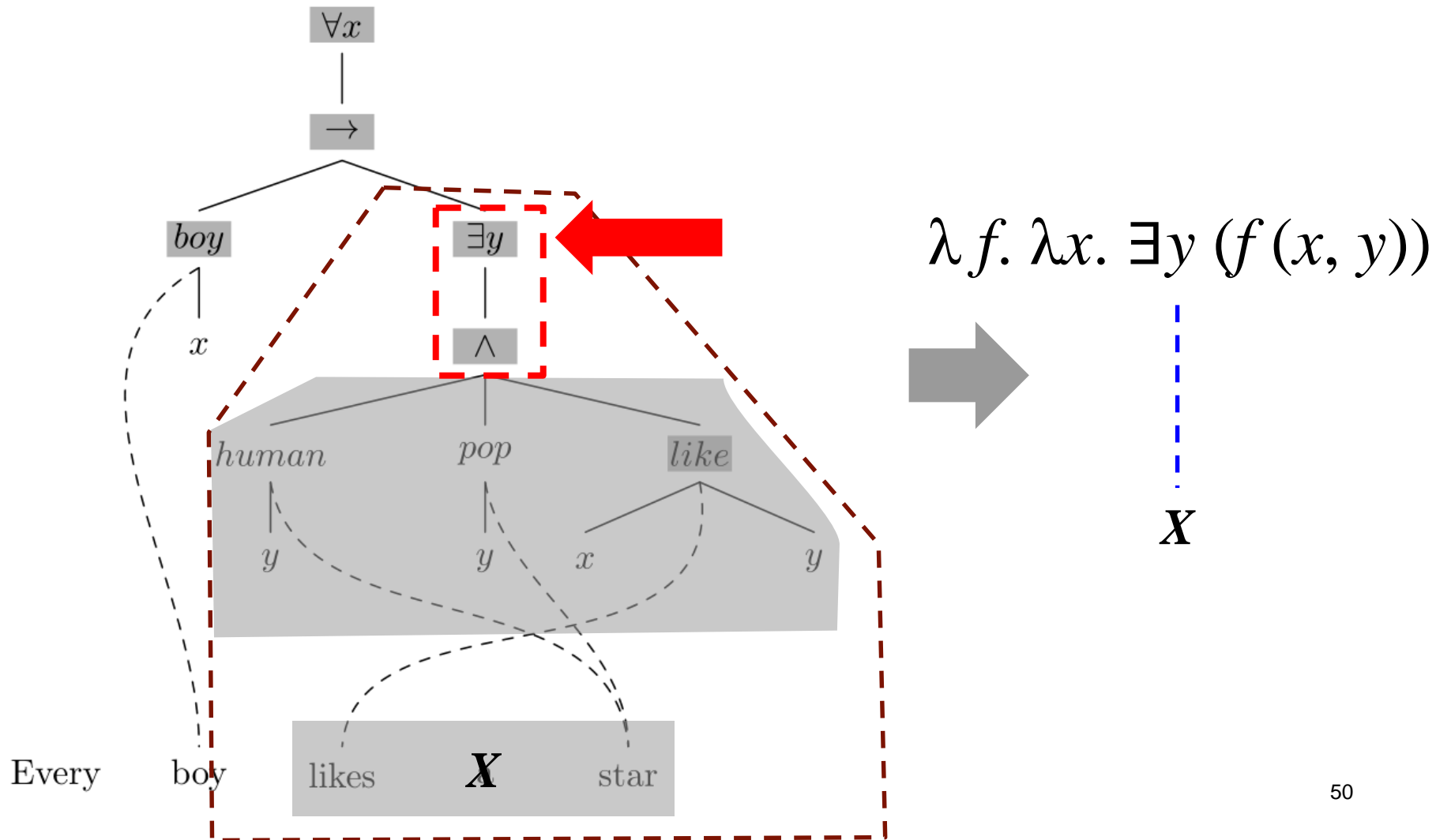


$\lambda f. \lambda x. \exists y (f(x, y))$

# Minimal Rule Extraction



# Minimal Rule Extraction





# Minimal Rule Extraction

$X \rightarrow \langle X_1, \lambda f. \lambda x. \exists y (f(x, y)) \triangleleft X_1 \rangle$

$\lambda f. \lambda x. \exists y (f(x, y))$

$X$

Every boy likes  $X$  star

# Experiments

