Combining Axiom Injection and Knowledge Base Completion for Efficient Natural Language Inference

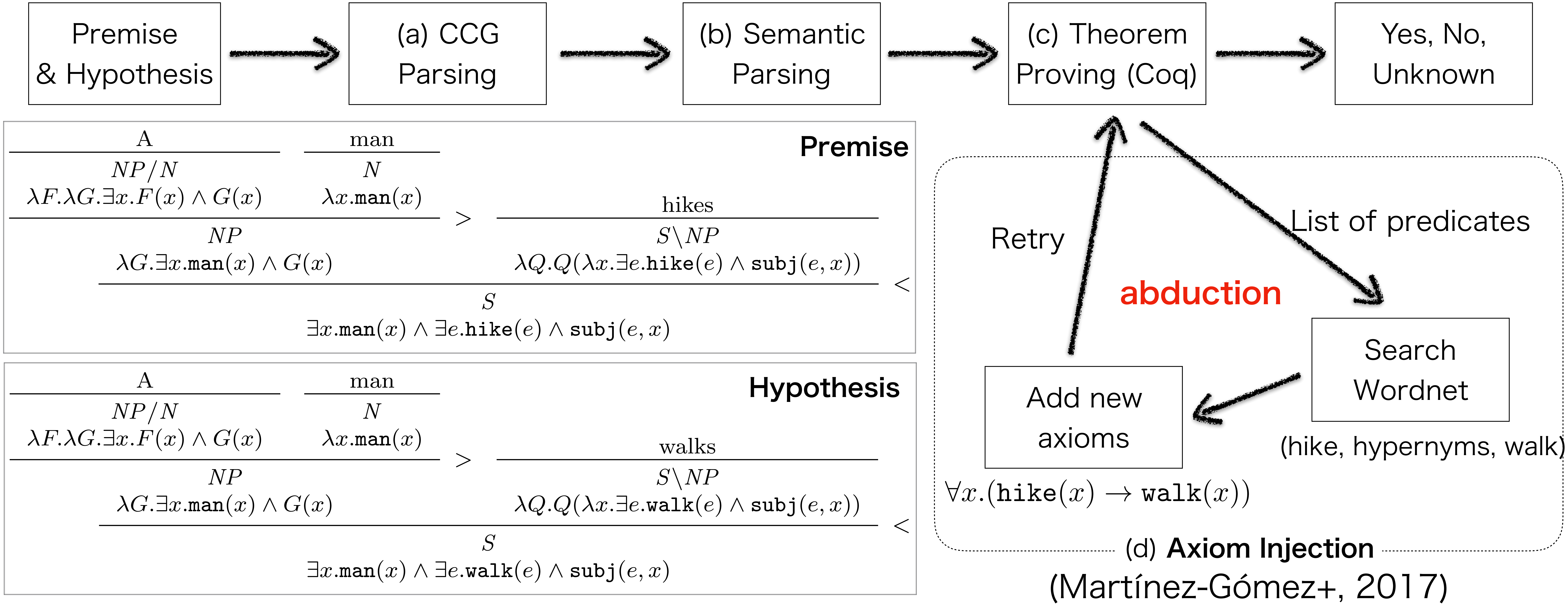
Masashi Yoshikawa◆, Koji Mineshima★, Hiroshi Noji❤, Daisuke Bekki★
◆Nara Institute of Science and Technology
★Ochanomizu University
❤Artificial Intelligence Research Center, AIST
AAAI-33 2019/XX/XX

Today's Talk

- Goal:
  - Developing **Accurate** & **Efficient** RTE system based on higher-order logic
  - Proof Assistant Coq: Automated higher-order theorem proving
- Issue: How to deal the massive amount of knowledge (axioms) ??
  - **Extending knowledge base** & **efficient reasoning** lie in a strained relation
- We solve the issue by combining logic-based RTE system with **knowledge base completion**!
  - We develop a **Coq plugin** for the further efficiency

$$\forall x.\mathtt{hike}(x) \rightarrow \mathtt{walk}(x)$$
$$\forall x.\mathtt{walk}(x) \rightarrow \neg\mathtt{ride}(x)$$
$$\vdots$$

Background: ccg2lambda (Mineshima+, 2015)



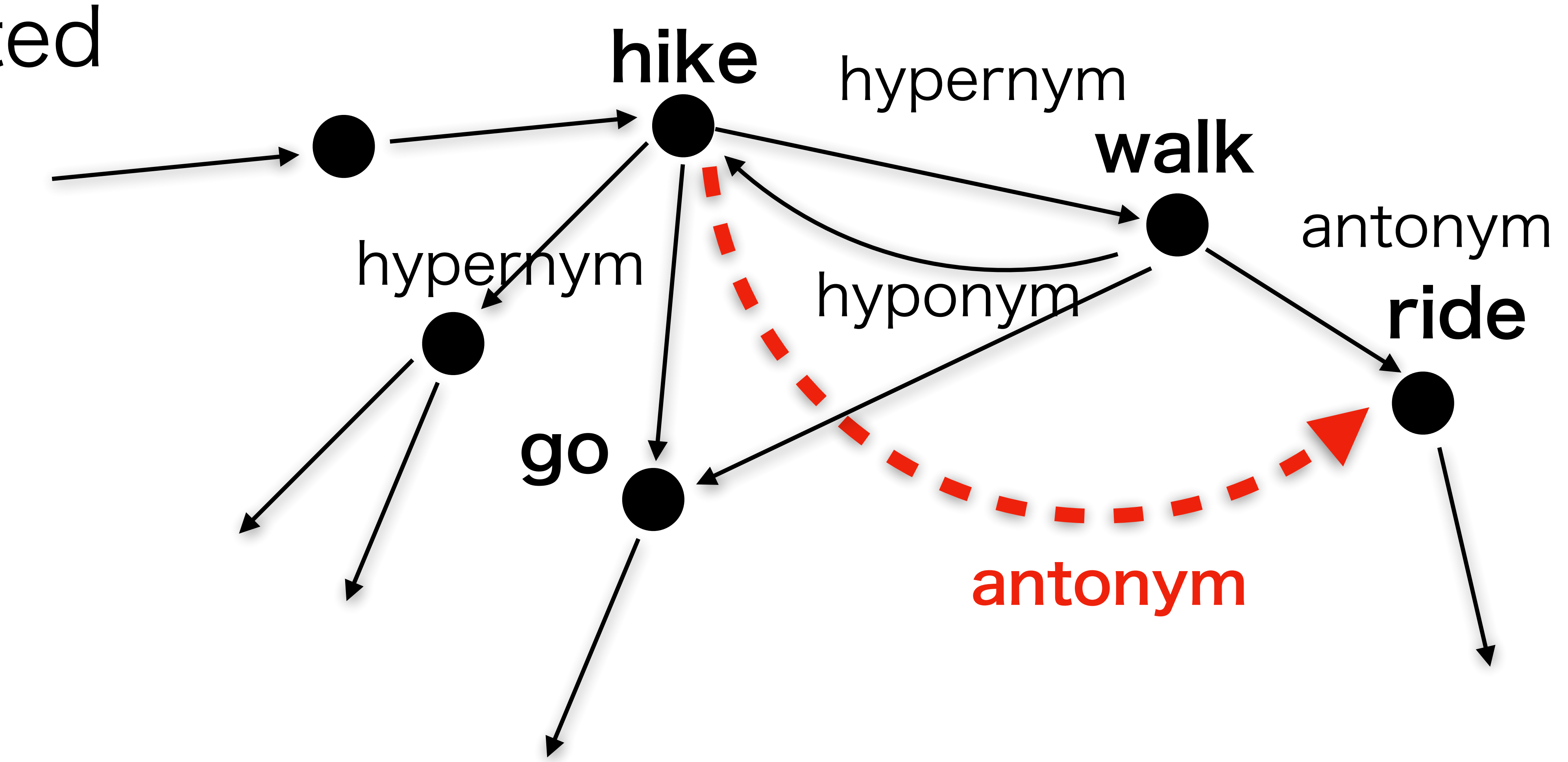
- **1st Issue**: Tensed relation between extending knowledge and efficient reasoning
- **2nd ssue**: Needs to exit a Coq proving session to run abduction

Knowledge Base Completion

- Wordnet
  - 0.1 million nodes, 0.5 million edges
- Freebase
  - 2 billion edges
- Knowledge base is not complete
  - Not all true facts (relations) are annotated

  → Infer missing knowledge using ML

(hike, hypernym, walk)
(walk, antonym, ride)
(hike, hypernym, go)
…



Knowledge Base Completion

- Design a function f to evaluate a triplet (s, r, o) is true or false
  - Simple models with the efficiency in mind (e.g. use of diagnoal matrices)
  - Higher-order relations (e.g. transitivity) are modeled with loss function (Bouchard+, 2015)
  - Achieves high scores on WordNet benchmark datasets(WN18, more than 94% in Hits@1)
- TransE (Bordes+, 2013)

$$f(s, r, o) = ||\mathbf{e}_s + \mathbf{e}_r - \mathbf{e}_o||_2^2, \mathbf{e}_i \in \mathbb{R}^n$$

Very simple additive model

- ComplEx (Trouillon+, 2016)

$$f(s, r, o) = \sigma(Re(\langle \boldsymbol{e}_s, \boldsymbol{e}_r, \overline{\boldsymbol{e}_o} \rangle)), \boldsymbol{e}_s, \boldsymbol{e}_r, \boldsymbol{e}_o \in \mathbb{C}^n$$

Models anti-symmetricity in complex vector space

Proposed: Axiom Injection using Knowledge Base Completion

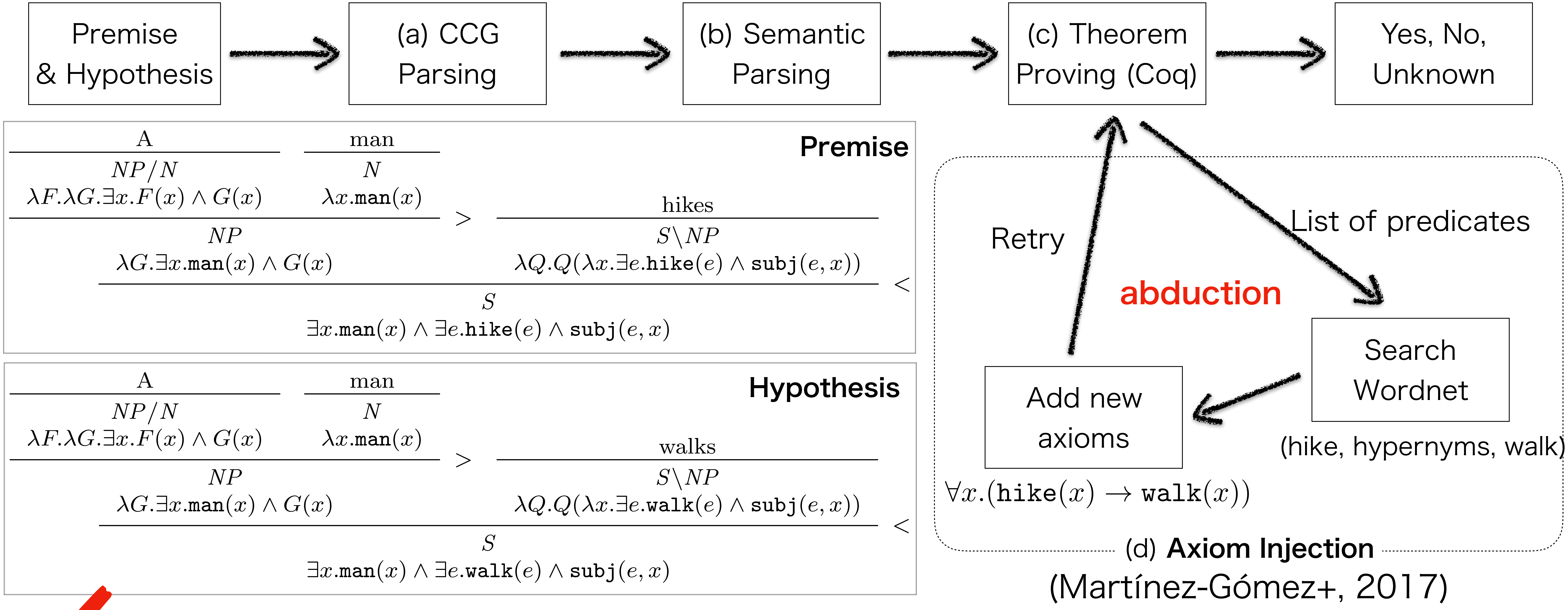
- We replace abduction method based on search on KBs (Martínez-Gómez+, 2017)
- Instead, our method uses a KBC function f
  - Convert triplets whose scores are above a predefined threshold
  - Conversion is defined in terms of relations r:

Relation r	Generated Axiom	Example
`synonym`, `hypernym`, `derivationally-related`	$\forall x.(s(x) \rightarrow o(x))$	$(\texttt{make}, r, \texttt{build}) \leadsto \forall e.(\texttt{make}(e) \rightarrow \texttt{build}(e))$
`antonym`	$\forall x.(s(x) \rightarrow \neg o(x))$	$(\texttt{parent}, r, \texttt{child}) \leadsto \forall x.(\texttt{parent}(x) \rightarrow \neg \texttt{child}(x))$
`hyponym`	$\forall x.(o(x) \rightarrow s(x))$	$(\texttt{talk}, r, \texttt{advise}) \leadsto \forall e.(\texttt{advise}(e) \rightarrow \texttt{talk}(e))$

Proposed: Axiom Injection using Knowledge Base Completion

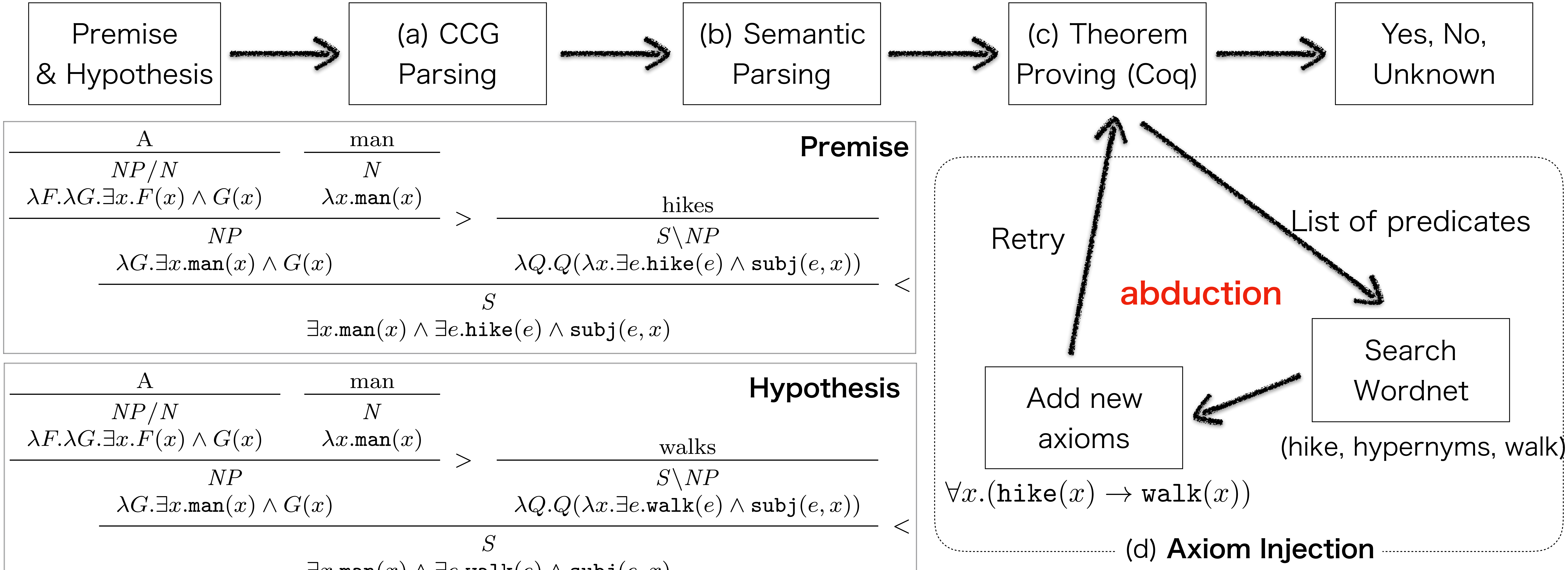
- **Very Fast**: traversal on KB → matrix operation of $O(n)$
  - Large gap in computing transitive closure (hypernym → hypernym → $\cdots$)
- **Scalable**: Adding new triplets with low cost
  - KB Graph: The time of a traversal is increased as # of entities does…
  - If s, r, and o exist already, no increase of # of parameters!
    - In the experiment, we add knowledge from VerbOcean(Chklovski+, 2004)
- **Latent knowledge** obtained in KBC model is also available (e.g. (hike, antonym, ride))

ccg2lambda (Mineshima+, 2015)



- **1st Issue**: Tensed relation between extending knowledge and efficient reasoning
- **2nd ssue**: Needs to exit a Coq proving session to run abduction

ccg2lambda (Mineshima+, 2015)



(Martínez-Gómez+, 2017)

- **1st Issue**: Tensed relation between extending knowledge and efficient reasoning
- **2nd ssue**: Needs to exit a Coq proving session to run abduction

Even Faster Reasoning with our Coq plugin (`abduction`)

```
1 subgoal

  ============================
   (exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)) ->
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)
t < intro.
1 subgoal

  H : exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)
  ============================
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)

t < abduction.
```

Even Faster Reasoning with our Coq plugin (`abduction`)

(man, walk)
(man, hike)
(hike, walk)

```
1 subgoal

  ==============================
  (exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)) ->
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)
t < intro.
1 subgoal

  H : exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)
  ==============================
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)

t < abduction.
```

Even Faster Reasoning with our Coq plugin (`abduction`)

(man, walk)
(man, hike)
(hike, walk)

Construct a list of predicate pairs from context and goal

```
1 subgoal

  ============================
  (exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)) ->
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)
t < intro.
1 subgoal

  H : exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)
  ============================
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)

t < abduction.
```

Even Faster Reasoning with our Coq plugin (`abduction`)

```
1 subgoal

  ==============================
  (exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)) ->
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)
t < intro.
1 subgoal

  H : exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)
  ==============================
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)

t < abduction.
```

(man, walk)
(man, hike)
(hike, walk)

Construct a list of predicate pairs from context and goal

↓

Send them to Python server

↓

Evaluate all the predicate pairs using ComplEx

↓

Filter them by score

Even Faster Reasoning with our Coq plugin (`abduction`)

```
1 subgoal

  ==============================
  (exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)) ->
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)
t < intro.
1 subgoal

  H : exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)
  ==============================
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)

t < abduction.
```

(man, walk)
(man, hike)
(hike, walk)

Construct a list of predicate pairs from context and goal

↓

Send them to Python server

↓

Evaluate all the predicate pairs using ComplEx

↓

Filter them by score

↓

Add them as axioms

(hike, hypernym, walk)

convert

$$\forall x \,.\, \mathbf{hike}(x) \rightarrow \mathbf{walk}(x)$$

Even Faster Reasoning with our Coq plugin (`abduction`)

(man, walk)
(man, hike)
(hike, walk)

```
1 subgoal

  ==============================
   (exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)) ->
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)
t < intro.
1 subgoal

  H : exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)
  ==============================
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)

t < abduction.
1 subgoal
  H : exists x : Entity, man x /\ (exists e : Event, hike e /\ subj e x)
  NLax1 : forall x : Event, hike x -> walk x
  ==============================
  exists x : Entity, man x /\ (exists e : Event, walk e /\ subj e x)
```

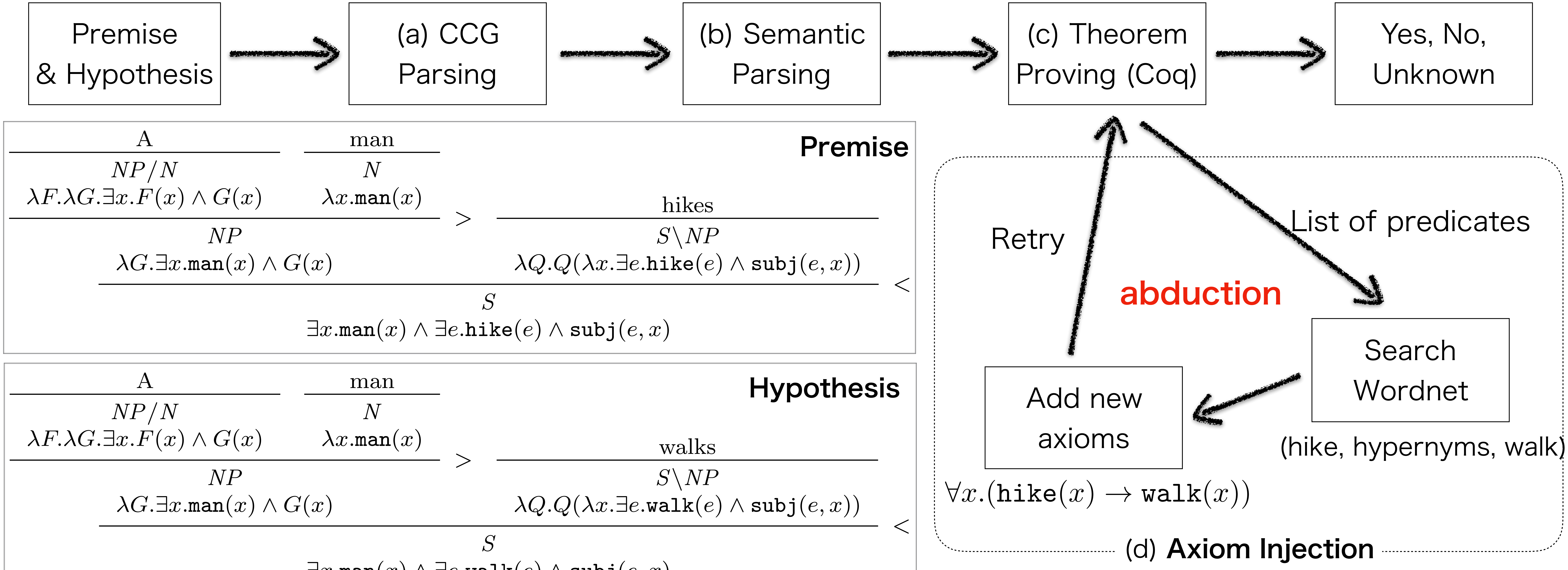
1. Construct a list of predicate pairs from context and goal
2. Send them to Python server
3. Evaluate all the predicate pairs using ComplEx
4. Filter them by score
5. Add them as axioms

$$\forall x\,.\,\mathbf{hike}(x) \rightarrow \mathbf{walk}(x)$$

(hike, hypernym, walk)

convert

Even Faster Reasoning with our Coq plugin (`abduction`)



(Martínez-Gómez+, 2017)

- **1st Issue**: Tensed relation between extending knowledge and efficient reasoning
- **2nd ssue**: Needs to exit a Coq proving session to run abduction

Even Faster Reasoning with our Coq plugin (`abduction`)

Premise & Hypothesis → (a) CCG Parsing → (b) Semantic Parsing → **(c) Theorem Proving (Coq)** → Yes, No, Unknown

Premise

$$\dfrac{\dfrac{\dfrac{\text{A}}{NP/N \quad \lambda F.\lambda G.\exists x.F(x) \wedge G(x)} \quad \dfrac{\text{man}}{N \quad \lambda x.\mathtt{man}(x)}}{NP \quad \lambda G.\exists x.\mathtt{man}(x) \wedge G(x)}> \quad \dfrac{\text{hikes}}{S\backslash NP \quad \lambda Q.Q(\lambda x.\exists e.\mathtt{hike}(e) \wedge \mathtt{subj}(e,x))}}{S \quad \exists x.\mathtt{man}(x) \wedge \exists e.\mathtt{hike}(e) \wedge \mathtt{subj}(e,x)}<$$

Hypothesis

$$\dfrac{\dfrac{\dfrac{\text{A}}{NP/N \quad \lambda F.\lambda G.\exists x.F(x) \wedge G(x)} \quad \dfrac{\text{man}}{N \quad \lambda x.\mathtt{man}(x)}}{NP \quad \lambda G.\exists x.\mathtt{man}(x) \wedge G(x)}> \quad \dfrac{\text{walks}}{S\backslash NP \quad \lambda Q.Q(\lambda x.\exists e.\mathtt{walk}(e) \wedge \mathtt{subj}(e,x))}}{S \quad \exists x.\mathtt{man}(x) \wedge \exists e.\mathtt{walk}(e) \wedge \mathtt{subj}(e,x)}<$$

Coq combined with `abduction` tactic

- No need to quit Coq
- Python server
  - existing DL libraries (and use of GPUs!)
- The mechanism can be extended for other problems of theorem proving (e.g. premise selection (Alemi+, 2016))

- **1st Issue**: Tensed relation between extending knowledge and efficient reasoning
- **2nd ssue**: Needs to exit a Coq proving session to run abduction

Experiments

- We evaluate our model on SICK(Marelli+, 2014) dataset
  - Metrices: accuracy and processing time of an RTE problem
    - (macro average of 5 runs on trial set, timeout 100 sec.)
- We use ComplEx (Trouillon+ 2016) as our KBC model.
  - Logistic loss: $\sum_{((s,r,o),t)\in\mathcal{D}} t \log f(s,r,o) + (1-t)\log(1-f(s,r,o))$
- In this work, we create training data as well
  - by extracting triplets from WordNet: (syno-, ant-, hypo-, hyper- nyms ..)
  - VerbOcean(Chklovski+, 2004): relations among verbs

Experimental Results on SICK

	Accuracy	Speed (Sec. / Problem)
no abduction	77.30	3.79
search-based (WordNet)	83.55	9.12
search-based (+VerbOcean)	83.68	9.42
KBC-based (WordNet)	83.55	4.03
KBC-based (+VerbOcean)	83.45	3.84

- Baseline: axiom injection based on search on KBs (Martínez-Gómez+, 2017)

Experimental Results on SICK

	Accuracy	Speed (Sec. / Problem)
no abduction	77.30	3.79
search-based (WordNet)	83.55	9.12
search-based (+VerbOcean)	83.68	9.42
KBC-based (WordNet)	83.55	4.03
KBC-based (+VerbOcean)	83.45	3.84

- Baseline: axiom injection based on search on KBs (Martínez-Gómez+, 2017)
- **Accuracy**: comparable / slightly lower compared to baseline
- **Proccessing Speed**: greatly improve; come close to "no abduction" case
- **Augmentation with VerbOcean**: further improvement in speed

Analyzing Latent Knowledge on More Challenging LexSICK dataset

- We construct a new RTE problems to evaluate latent knowledge of KBC in terms of RTE.
  - 60 problems of combination of syntactic & lexical phenomena
    - quantification coordination, passive-active alteration, etc.
    - use thesaurus.com and Merriam-Webster (not WordNet)
- NN-based ResEncoder (Nie and Bansal, 2007): #correct is 18/58
- search- and KBC-based ccg2lambda: 20/58 vs. 21/58
  - showing the dataset is difficult for both NN- and logic-based methods.

P: Someone is dropping the meat in a pan.	**P:** The man is singing and playing the guitar.
-> H: The meat is being thrown into a pan.	**-> H:** The guitar is being performed by a man.

We observe interesting results with KBC-based system.

- on LexSICK dataset:

P: A man and a woman are *walking* together through the wood.

-> H: A man and a woman are *staying* together.

- KBC: $\forall x\,.\,\mathbf{walk}(x) \rightarrow \neg\mathbf{stay}(x)$
- search: (none)

P: A man is *emptying* a container made of plastic completely.

-> H: A man is *clearing* a container made of plastic completely.

- KBC: $\forall x\,.\,\mathbf{empty}(x) \rightarrow \neg\mathbf{clear}(x)$
- search: (none)

We observe interesting results with KBC-based system.

- on SICK dataset:

P: A <u>*couple*</u> of white dogs are <u>*running*</u> and jumping <u>*along*</u> a beach.
-> H: <u>*Two*</u> dogs are <u>*playing*</u> <u>*on*</u> a beach.

- KBC:

$$\forall x\,.\,\mathbf{couple}(x) \rightarrow \mathbf{two}(x) \qquad \forall x\,.\,\mathbf{run}(x) \rightarrow \mathbf{play}(x)$$

$$\forall x\,.\,\mathbf{along}(x) \rightarrow \mathbf{on}(x)$$

Summary:

- A KBC-based injection method of lexical knowledge to logic-based RTE systems
  - Efficient, Scalable, Provides latent knowledge
  - `abduction` tactic enables even faster reasoning
- Future work
  - Phrase-level lexical axioms $\forall x.(\mathtt{have}(x) \wedge \mathtt{fun}(x)) \rightarrow \mathtt{enjoy}(x)$
    - Compositionally compute their vectors from word's ones
  - Lexical axioms from multiword expressions

$$\forall x.\mathtt{make\_up\_one's\_mind}(x) \rightarrow \mathtt{determine}(x)$$