Toward the construction of A linguistically valid Japanese CCG treebank

Asa Tomita[1], Hitomi Yanaka[2], Daisuke Bekki[1]

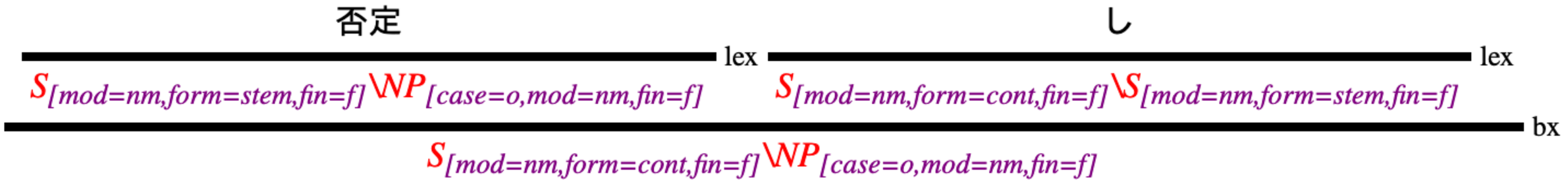
[1]Ochanomizu University [2]University of Tokyo

Tokyo Workshop on Theoretical and Computational Semantics
2023/07/21

Treebank:

a corpus that annotates the syntactic structures of sentences

ex: Japanese CCGbank [Uematsu et al. 2013]



Combinatory categorial grammar (CCG)
[Steedman 1996, 2001]

Lexicon

Contains words:
their phonological, syntactic and semantic information

Keats $\vdash NP : k$
eats $\vdash (S \setminus NP) / NP : eat$
apples $\vdash NP : apple$

+

Combinatory Rules

ex:
function application rules
function composition rules

Background

Japanese CCG parsers (ex: **depccg** [Yoshikawa et al. 2017], **jigg** [Noji & Miyao 2016])

- Take Japanese sentences as input and then output CCG trees
- Use **CCG treebanks** as training/evaluation data
- Rely on CCG treebanks for their **linguistic validity**

→ We constructed a **linguistically valid** CCG treebank based on Bekki (2010), which we regard as the standard Japanese CCG

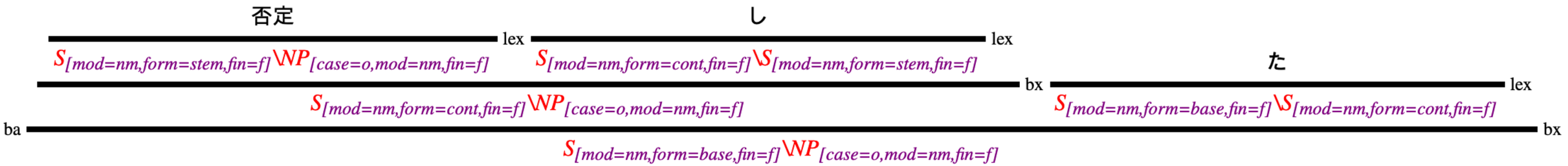
Japanese CCGbank [Uematsu et al. 2013]

Kyoto corpus [Kawahara et al. 2002]

NAIST text corpus [Iida et al. 2007]

Japanese particle corpus [Hanaoka et al. 2010]

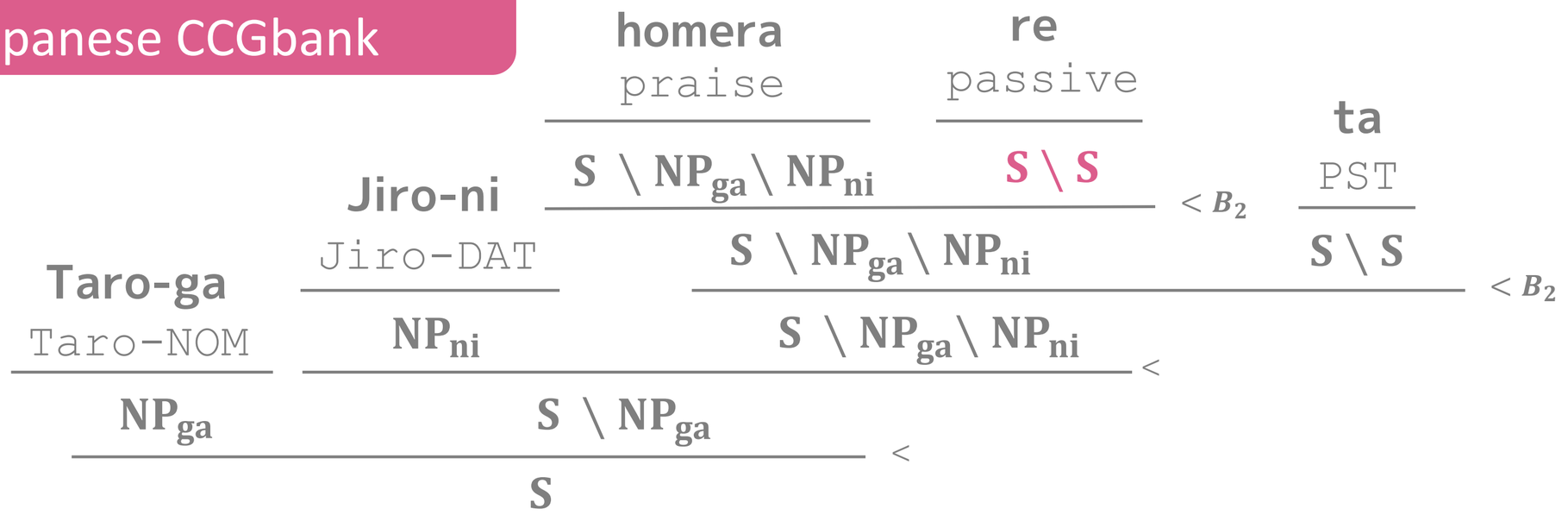
Japanese CCGbank was obtained through automatic conversion of these corpora.



Japanese CCGbank [Uematsu et al. 2013]

Problem: produces false predictions on passive and causative nestings [Bekki & Yanaka 2023]

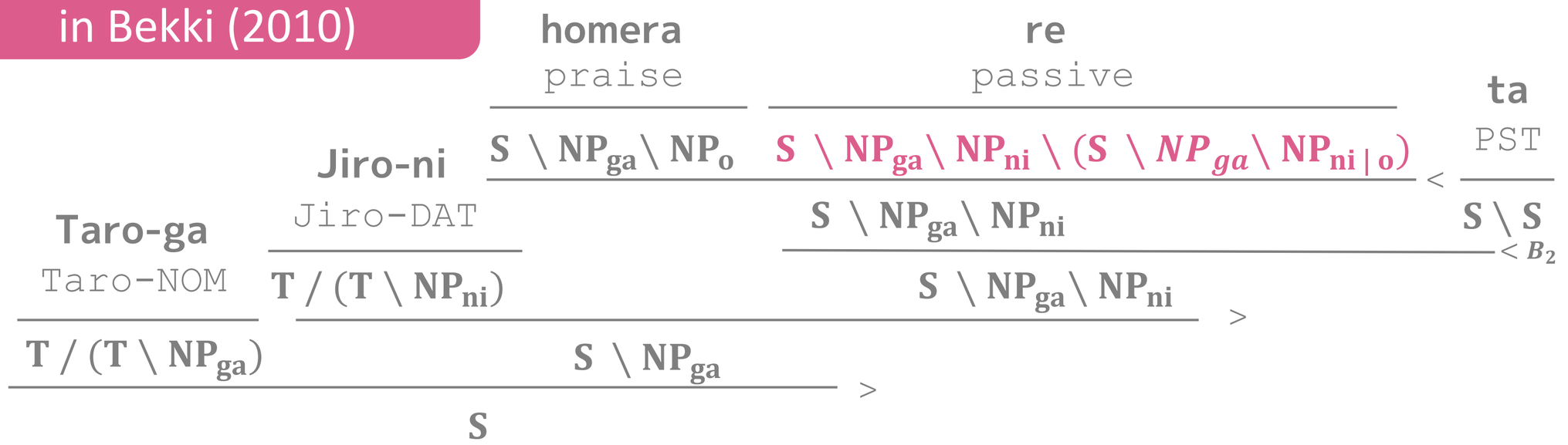
Syntactic structure in Japanese CCGbank



Japanese CCGbank [Uematsu et al. 2013]

Problem: produces false predictions on passive and causative nestings [Bekki & Yanaka 2023]

Syntactic structure in Bekki (2010)



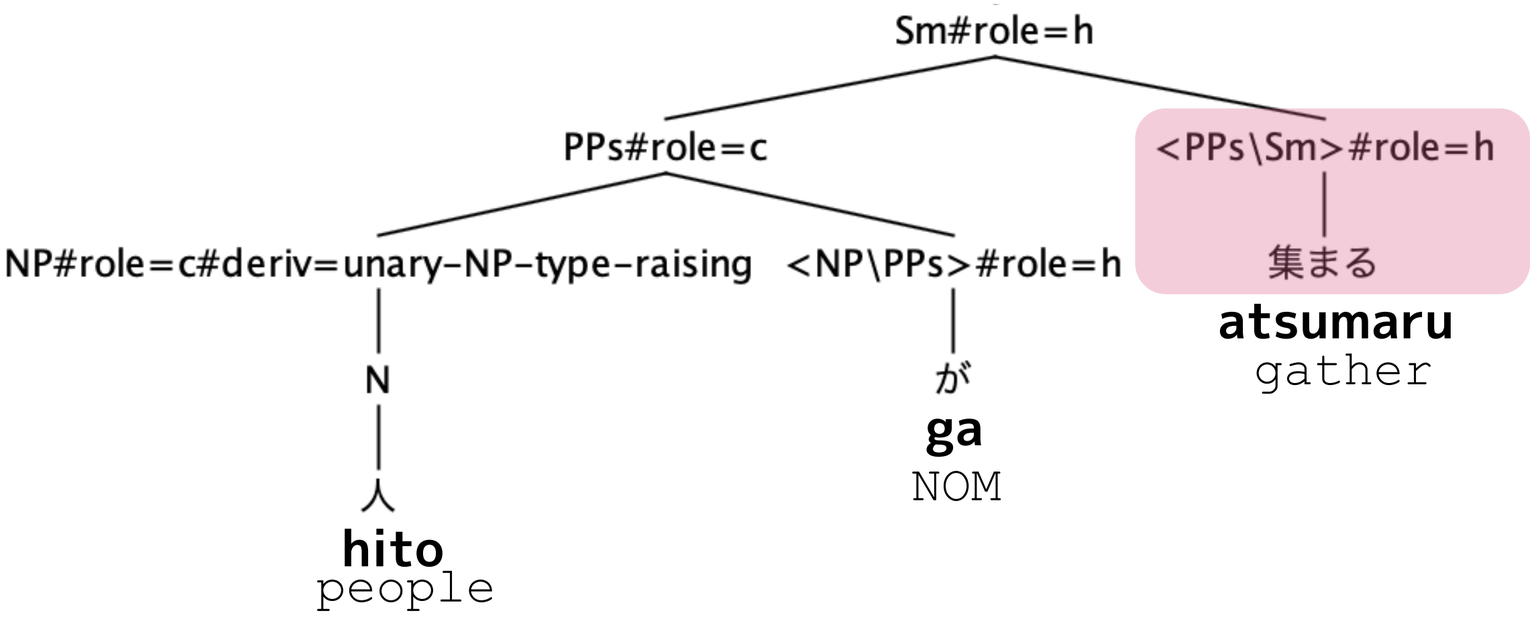
ABCTreebank [Kubota et al. 2019]

- Constructed by converting the Keyaki Treebank [Butler 2012] to **ABC grammar** trees
  (ABC grammar: function application rules + function composition rules)

- Can be easily converted to a CCG or type-logical grammar (TLG) [Morrill 1994; Moortgat 1997] treebank

ABCTreebank [Kubota et al. 2019]

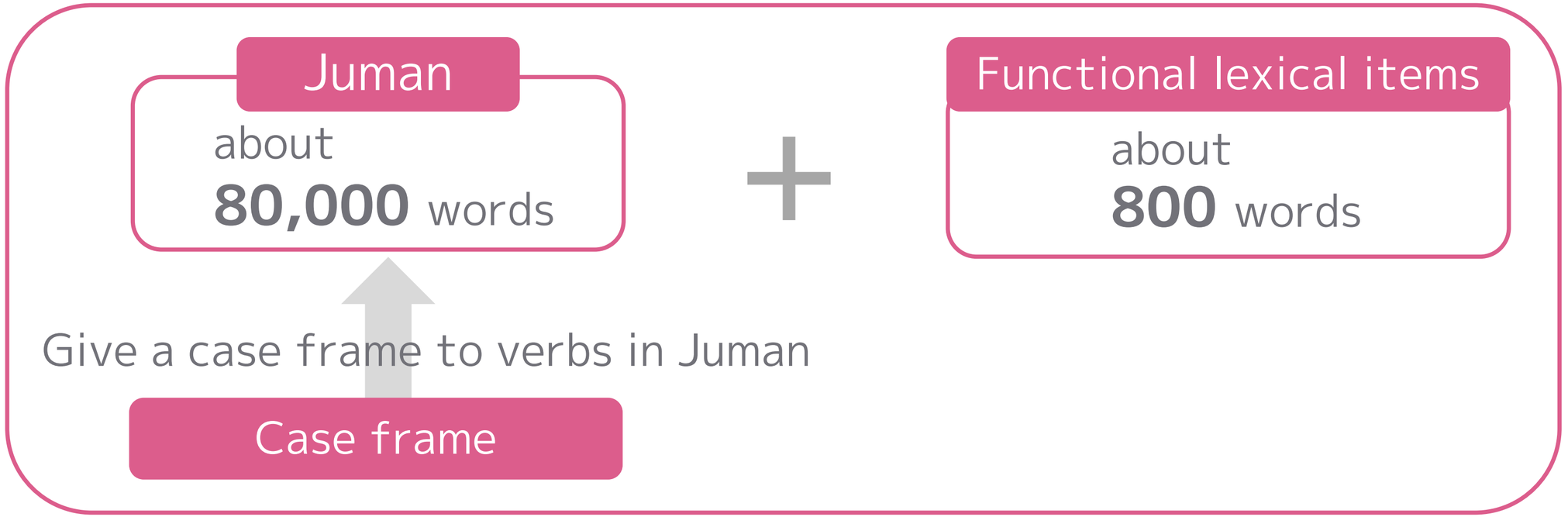
Pros: Argument structures are manually annotated
Cons: Syntactic information such as the part of speech is not included



Ra-column five row conjugation form

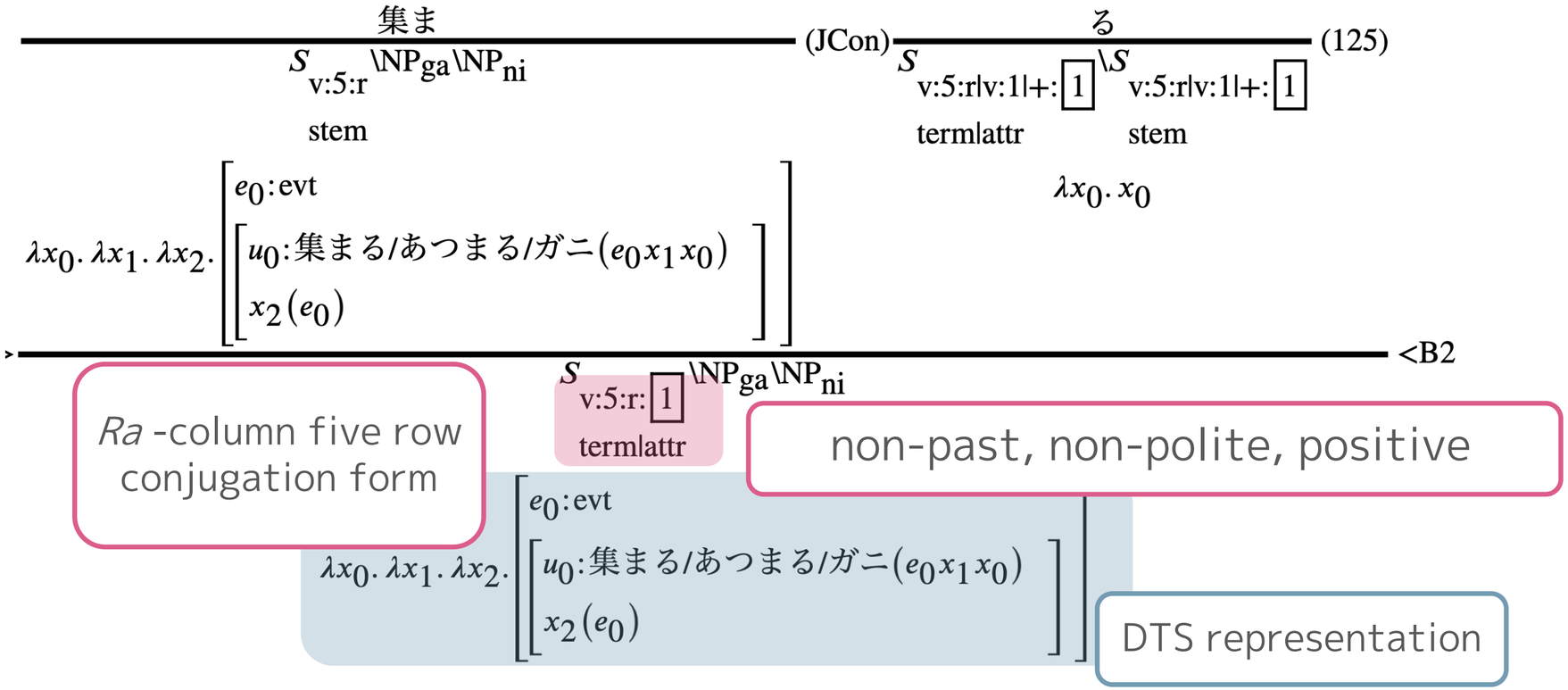
lightblue [Bekki & Kawazoe 2016]

- Japanese CCG parser that utilizes a **lexicon** consisting of 80,000 Juman words with case frames



lightblue [Bekki & Kawazoe 2016]

Pros: provides CCG trees with detailed syntactic features and semantic representations in DTS



lightblue [Bekki & Kawazoe 2016]

Pros: provides CCG trees with detailed syntactic features and semantic representations in DTS

Cons: contains errors in argument structure
→ sometimes gives an unnatural disambiguation in some contexts

Research aim and proposal

Research aim

Construction of a **linguistically valid** Japanese CCG treebank with **detailed syntactic features**

Research aim and proposal

Research aim

Construction of a **linguistically valid** Japanese CCG treebank with **detailed syntactic features**

Proposed Method

ABCTreebank

argument structures are manually annotated

+

lightblue

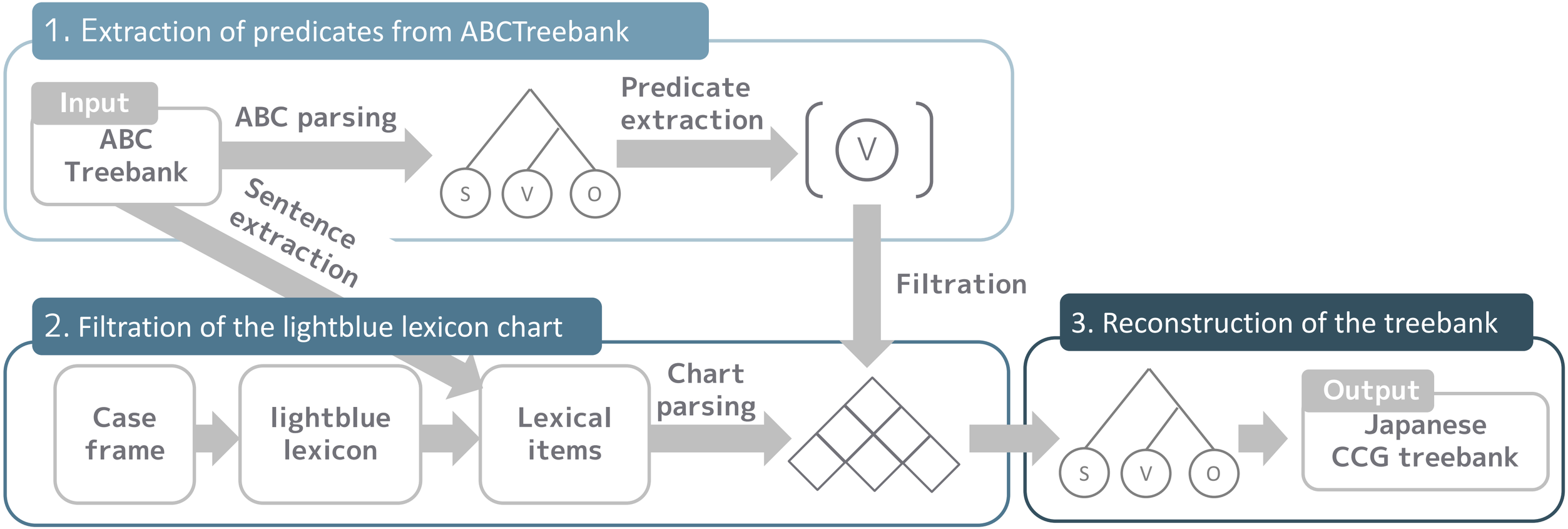
provides CCG trees with **detailed syntactic features and DTS representations**

ABCTreebank reforging

Reforging: a method of **decomposing** and **reconstructing** a treebank

1. Extraction of predicates from ABCTreebank
2. Filtration of the lightblue lexicon chart
3. Reconstruction of the treebank

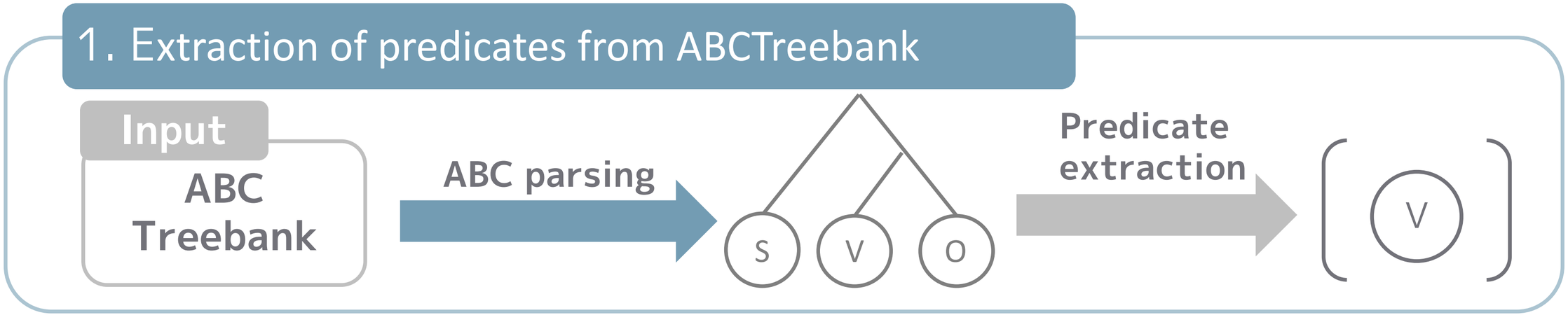
ABCTreebank reforging



Reforging – extraction of predicates from ABCTreebank

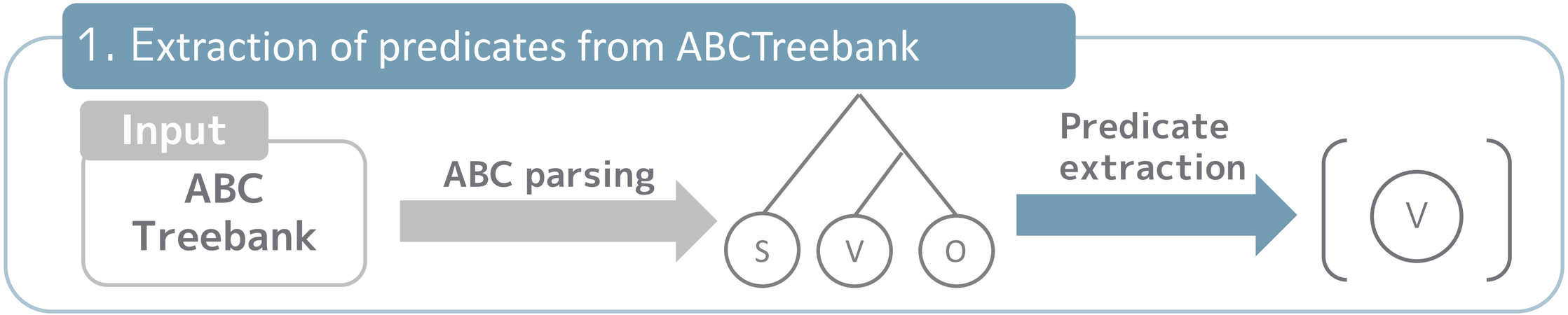
1. Extraction of predicates from ABCTreebank
2. Filtration of the lightblue lexicon chart
3. Reconstruction of the treebank

Reforging – extraction of predicates from ABCTreebank



Step 1-1:
Obtain tree-structured data from ABCTreebank by parsing the ABCTreebank

Reforging – extraction of predicates from ABCTreebank



Step 1-2:

Extract the syntactic information of predicates from ABCTreebank as a list of tuples with **four elements**

Four elements of the tuple:

phonetic form, category, starting position, ending position

Reforging – extraction of predicates from ABCTreebank

example:

Input

「 嫌いな 人 も いる 」

don't like / people / also / exist

(Some people don't like it.)

Output

[("嫌い", PPs \ SreI , 0 , 1),

phonetic form / category / start / end

(“いる”, PPs \ Sm , 5 , 6)]

Step 1-2:

Extract the syntactic information of predicates from the ABCTreebank as a list of tuples with **four elements**

Four elements of the tuple:

phonetic form, category, starting position, ending position

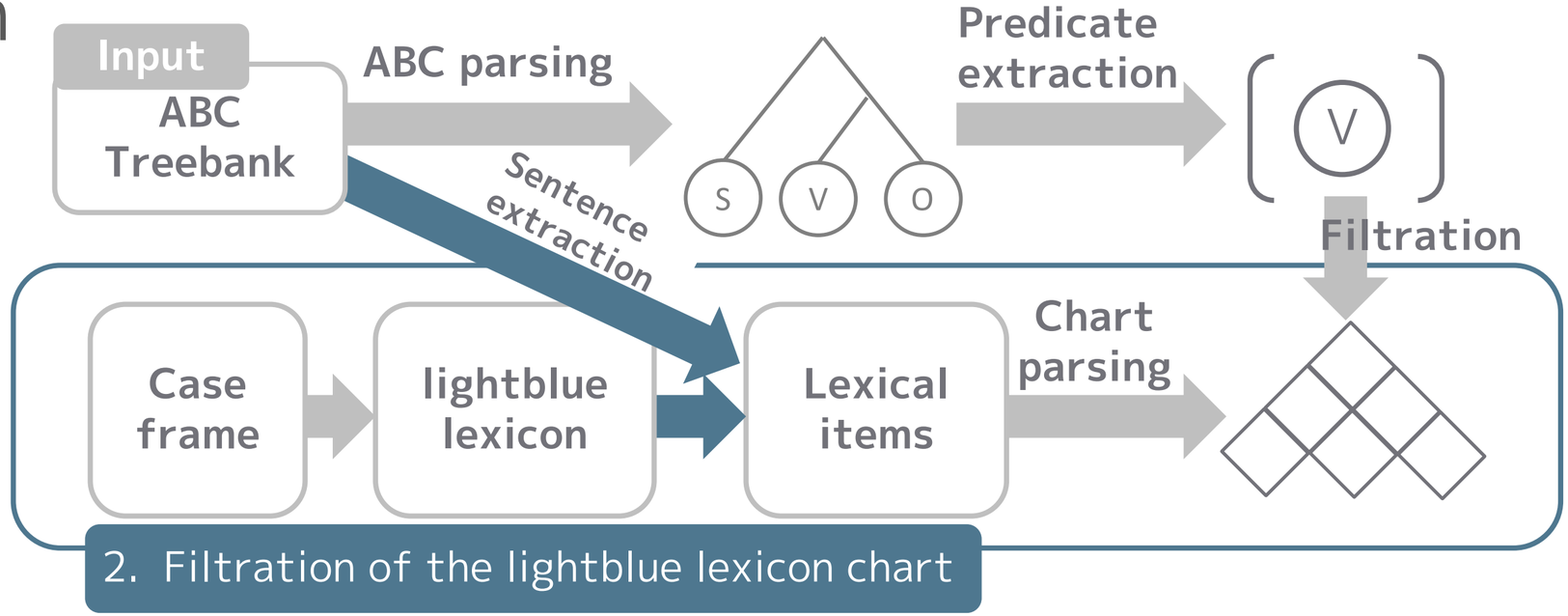
Reforging – filtration of the lightblue lexicon chart

1. Extraction of predicates from ABCTreebank
2. Filtration of the lightblue lexicon chart
3. Reconstruction of the treebank

Reforging – filtration of the lightblue lexicon chart

Step 2-1:

Extract the lexical items of all combinations of the substrings in the ABCTreebank sentence from the lightblue lexicon



Reforging – filtration of the lightblue lexicon chart

Step 2-1:
Extract the lexical items of all combinations of the substrings in the ABCTreebank sentence from the lightblue lexicon

Example:

Input

「 嫌いな 人 も いる 」
don’t like　people　also　exist

(Some people don't like it.)

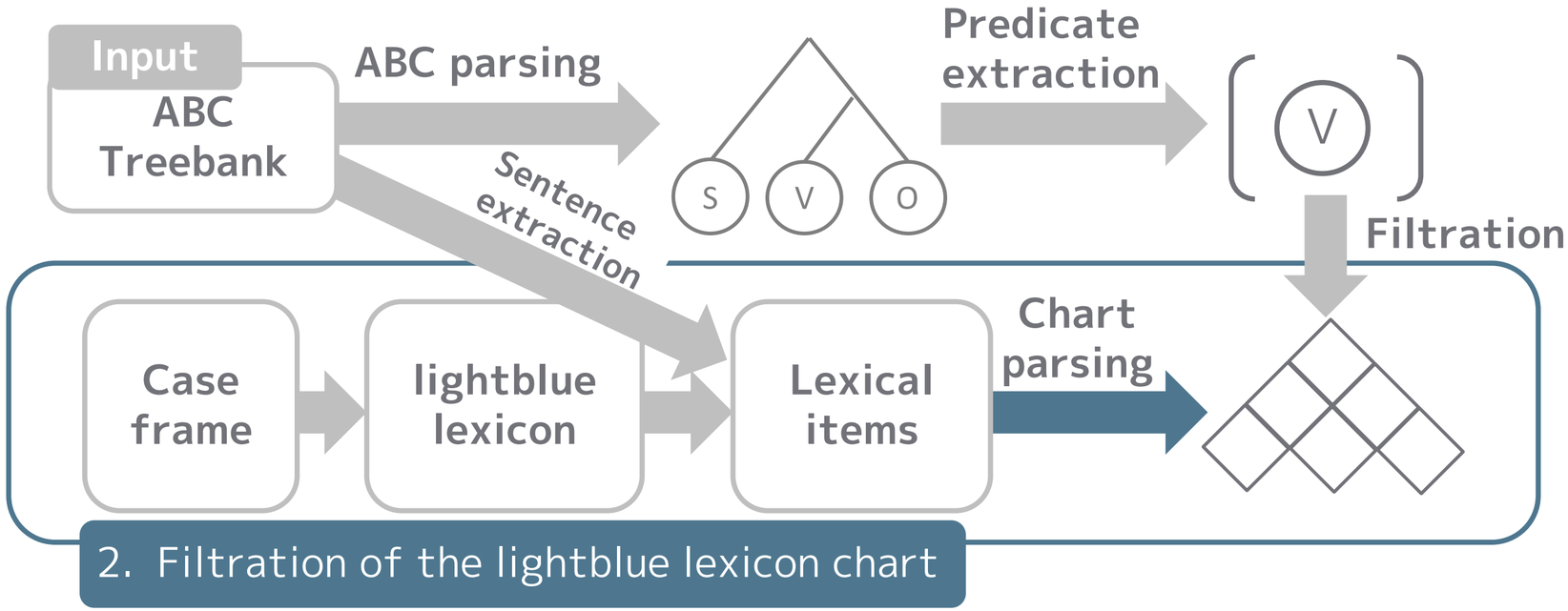
Output

Lexical items of words such as
嫌、い、な、人、も、い、る、嫌い、いな、いる、嫌いな

Reforging – filtration of the lightblue lexicon chart

Step 2-2:

Perform chart parsing for the sentence, using the lexical items extracted in Step2-1

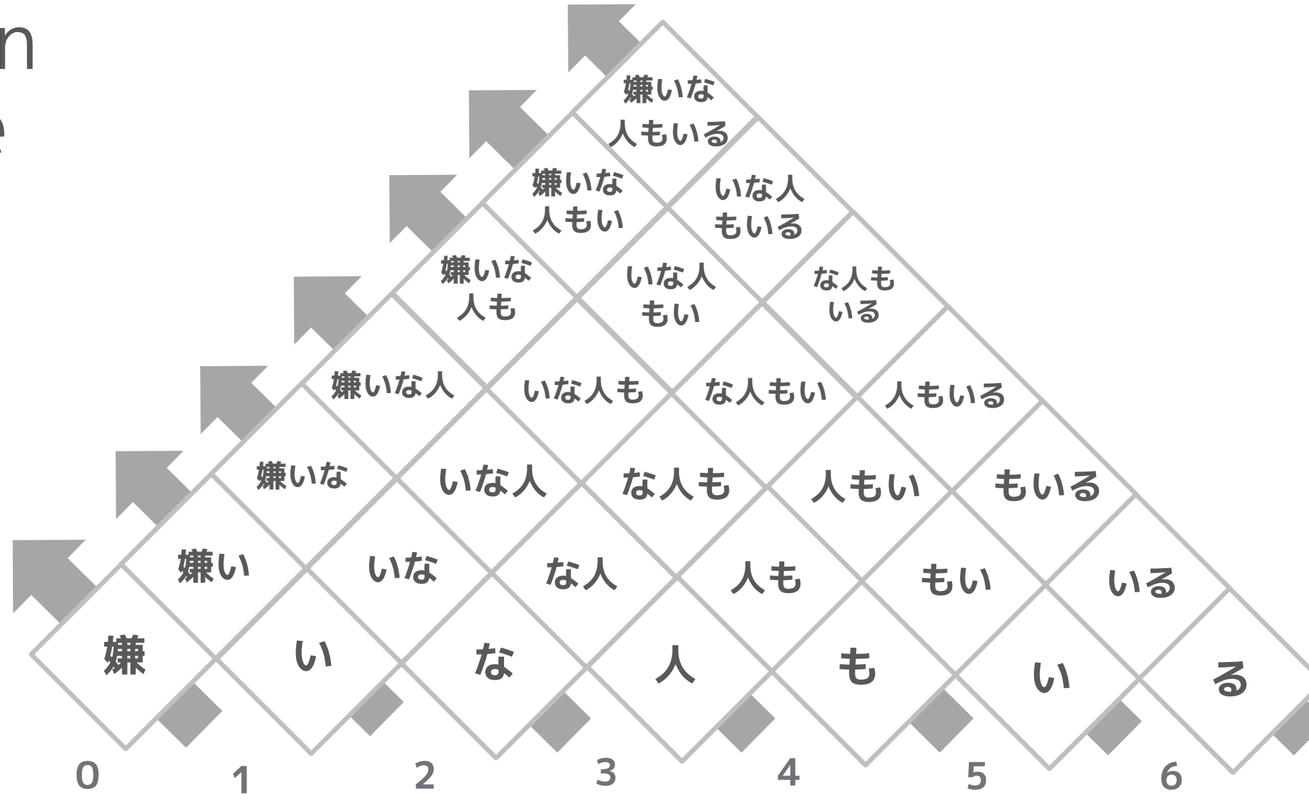


Left-corner chart parsing

Node data are calculated when building a syntactic structure of the word by compositing daughter nodes.

Node data include:

- syntactic category
- semantic representation
- score
- and so on ...



Left-corner chart parsing

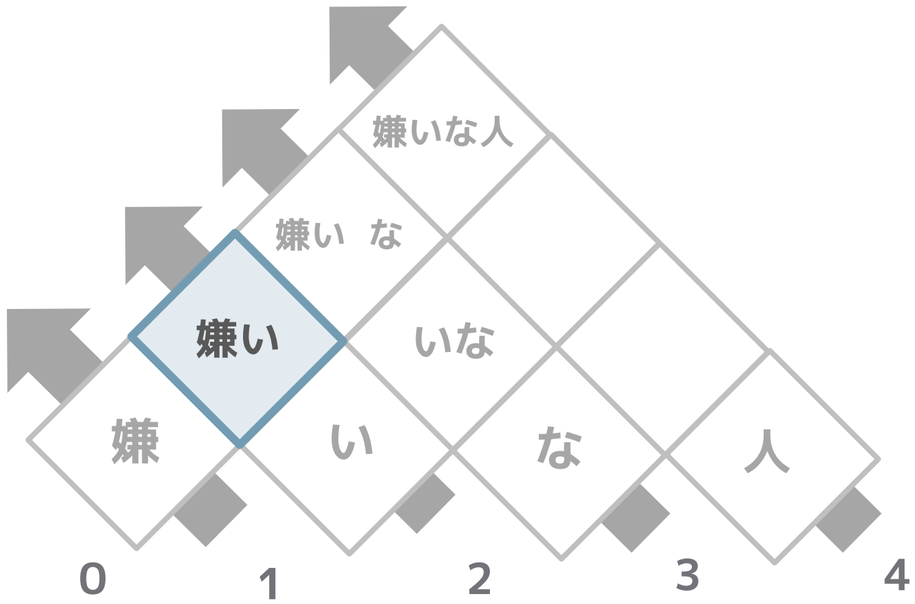
Node data of 「嫌い (don't like)」 cell (0,2) are

1. **Syntactic information of 「嫌い」 (nominal predicate)**
2. Syntactic information composited from 「嫌」 and 「い」 (continuous form of the verb)

Lexical items of (0,2)

("嫌い/きらい", $S_{[n:da|n:na|n:ni|+][nstem]} \setminus NP_{ga}$)

("嫌う/きらう/ガオ", $S_{[v:5:w<1>][cont|mod:m]} \setminus NP_{ga} \setminus NP_{o}$)



Left-corner chart parsing

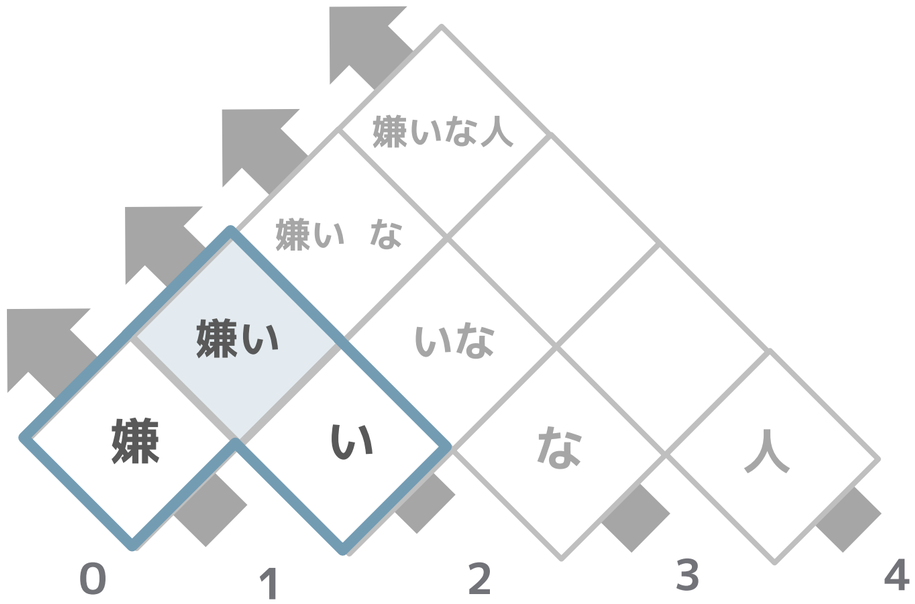
Node data of 「嫌い (don't like)」 cell (0,2) are

1. Syntactic information of 「嫌い」 (nominal predicate)
2. **Syntactic information composited from 「嫌」 and 「い」 (continuous form of the verb)**

Lexical items of (0,2)

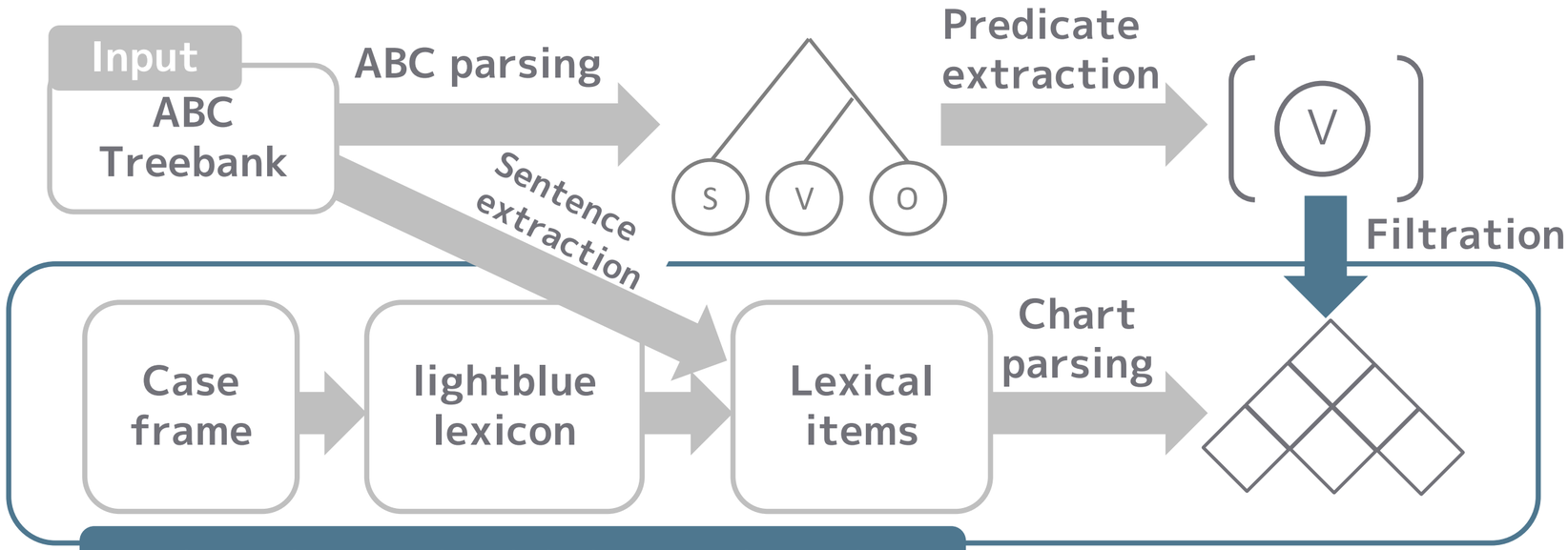
("嫌い/きらい",
$S_{[n:da|n:na|n:ni|+][nstem]} \setminus NP_{ga}$)

("嫌う/きらう/ガオ",
$\boldsymbol{S_{[v:5:w<1>][cont|mod:m]} \setminus NP_{ga} \setminus NP_{o}}$)

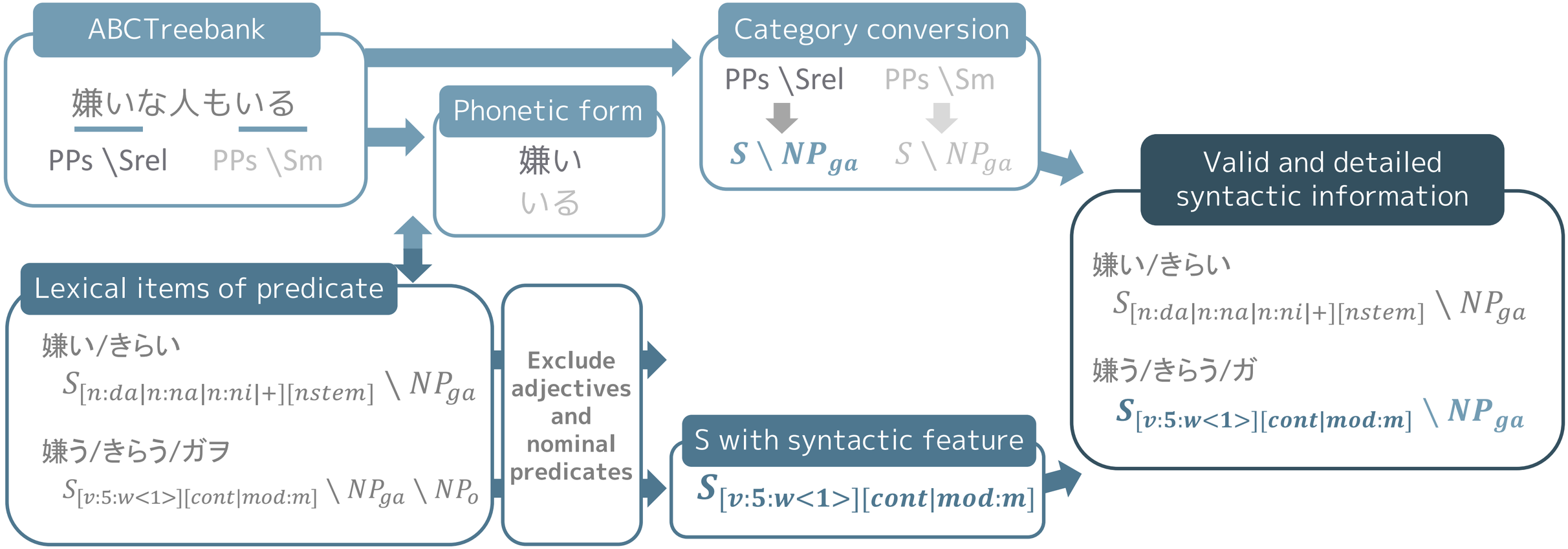


Reforging – filtration of the lightblue lexicon chart

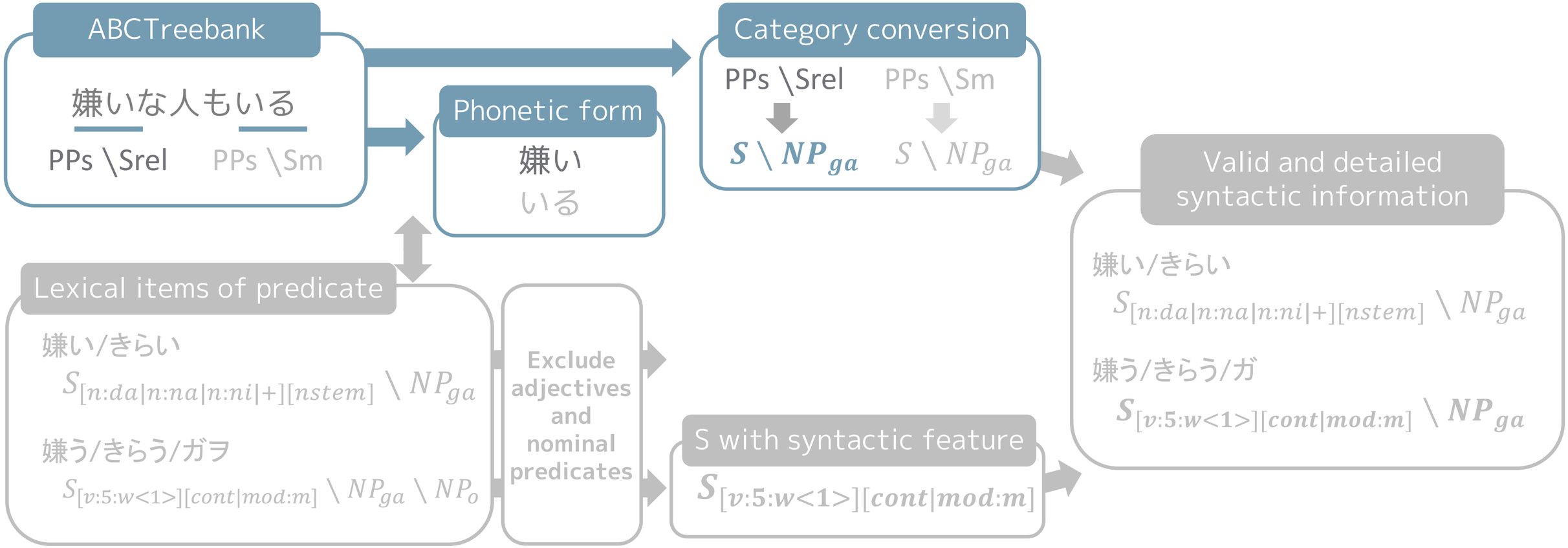
Step 2-3:
Filter the chart with the argument-structure information of the verb extracted from ABCTreebank



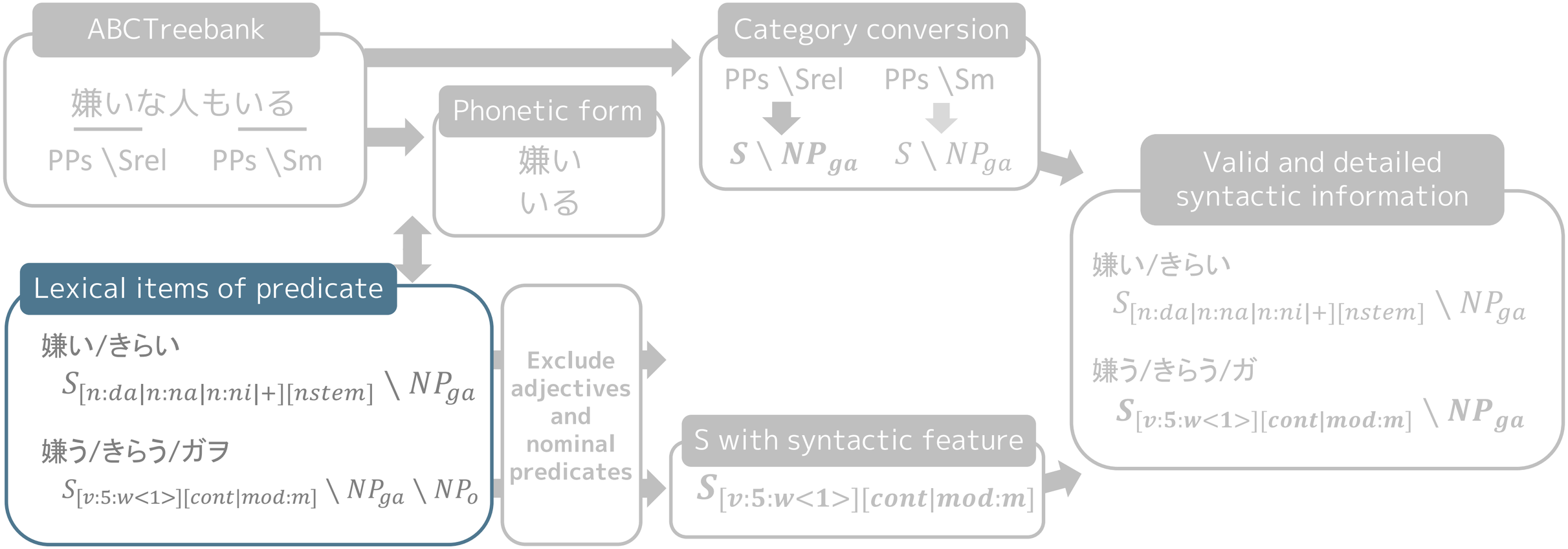
Filtering algorithm



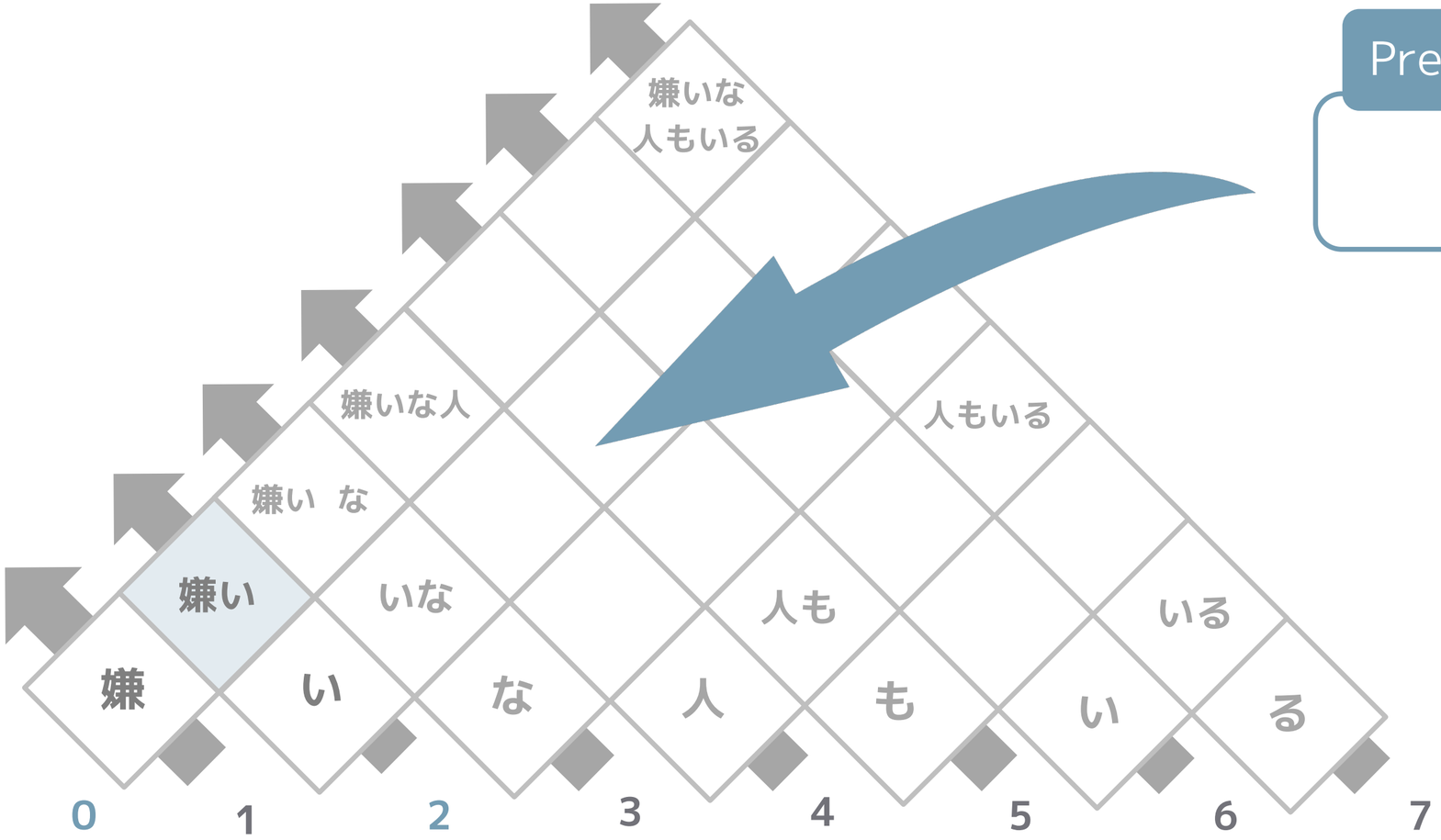
Filtering algorithm



Filtering algorithm



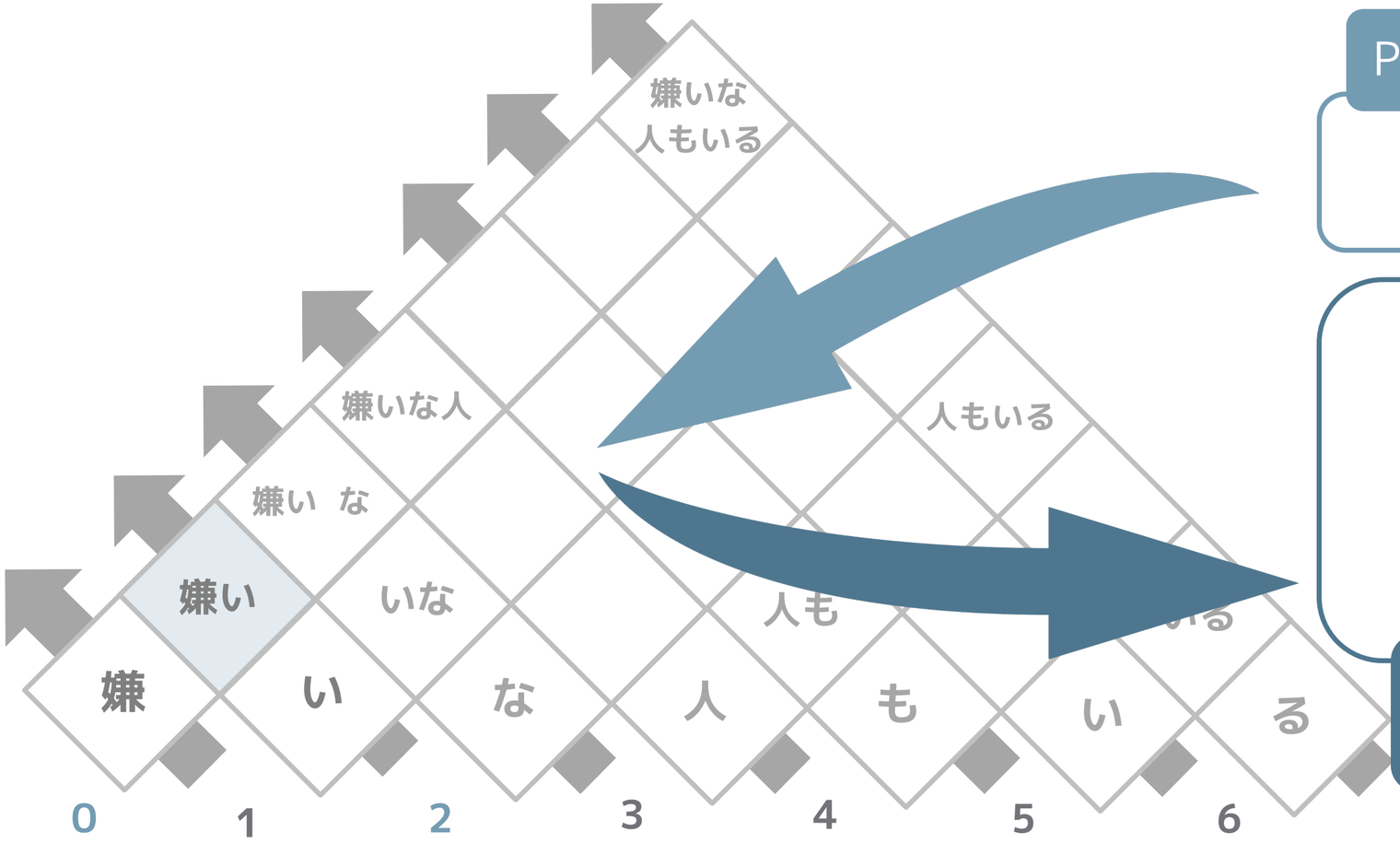
Filtering algorithm – obtaining lexical items



Predicate data from ABCTreebank

("嫌い", PPs \ SreI ,0,1)

Filtering algorithm – obtaining lexical items



Predicate data from ABCTreebank

("嫌い", PPs \ SreI ,0,1)

嫌い/きらい

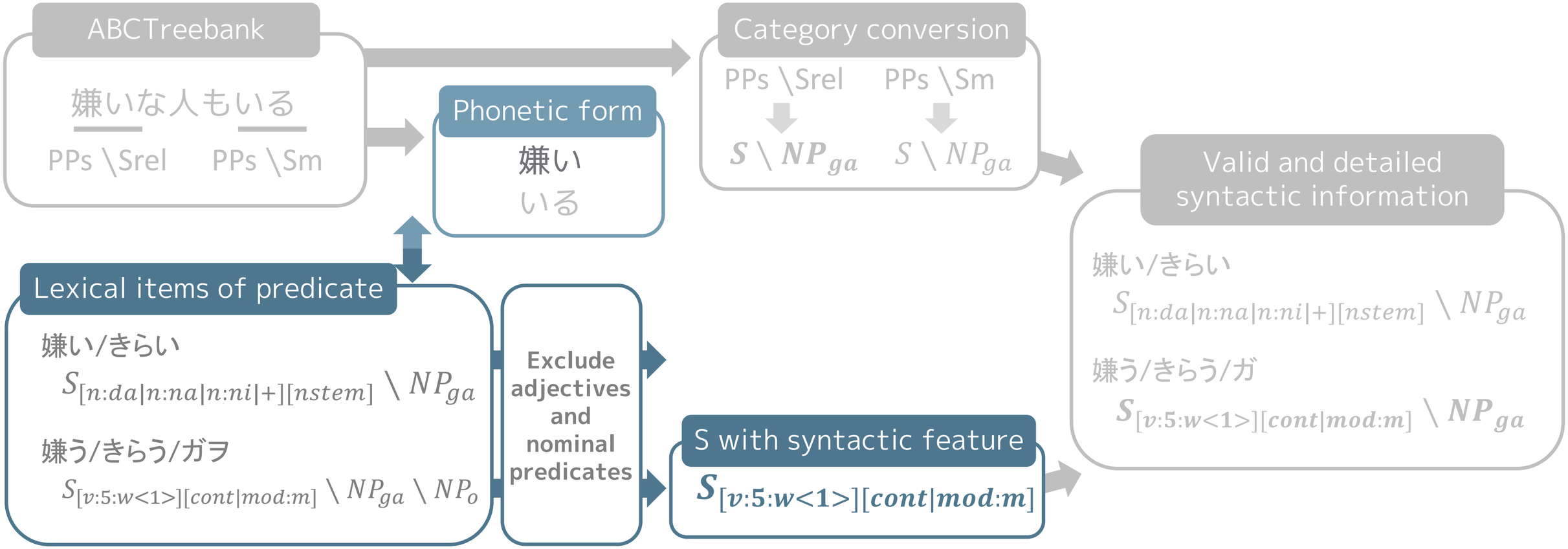
$S_{[n:da|n:na|n:ni|+][nstem]} \setminus NP_{ga}$

嫌う/きらう/ガヲ

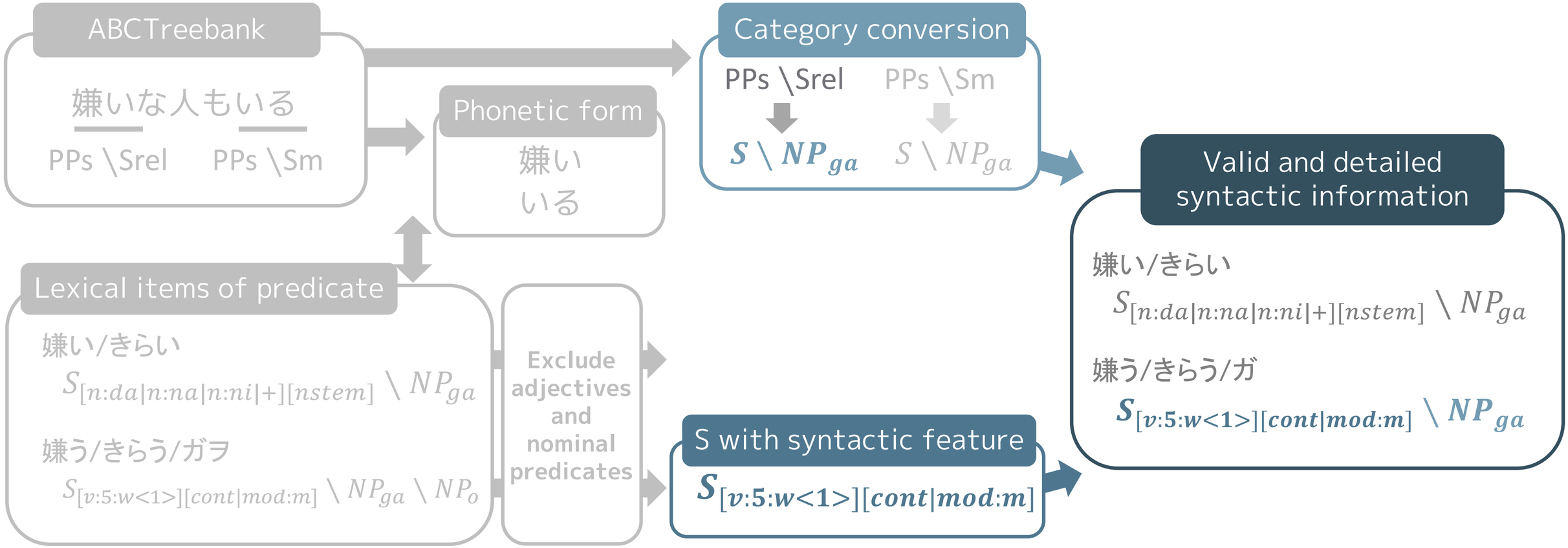
$S_{[v:5:w<1>][cont|mod:m]} \setminus NP_{ga} \setminus NP_{o}$

Syntactic information of lexical items in the lightblue lexicon

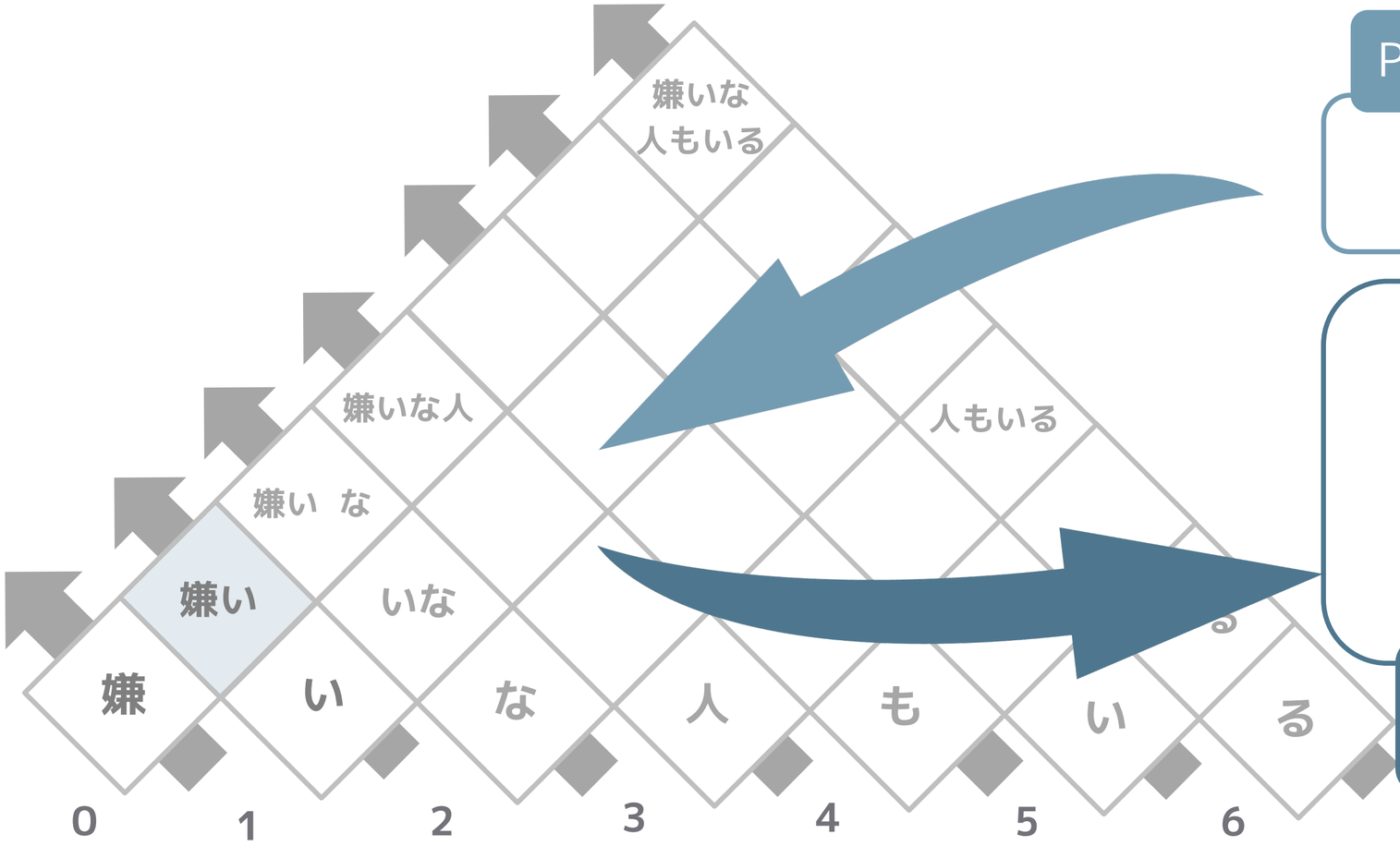
Filtering algorithm



Filtering algorithm



Reforging – filtration of the lightblue lexicon chart



Predicate data from ABCTreebank

[("嫌い", PPs \ SreI ,0,1)]

嫌い/きらい

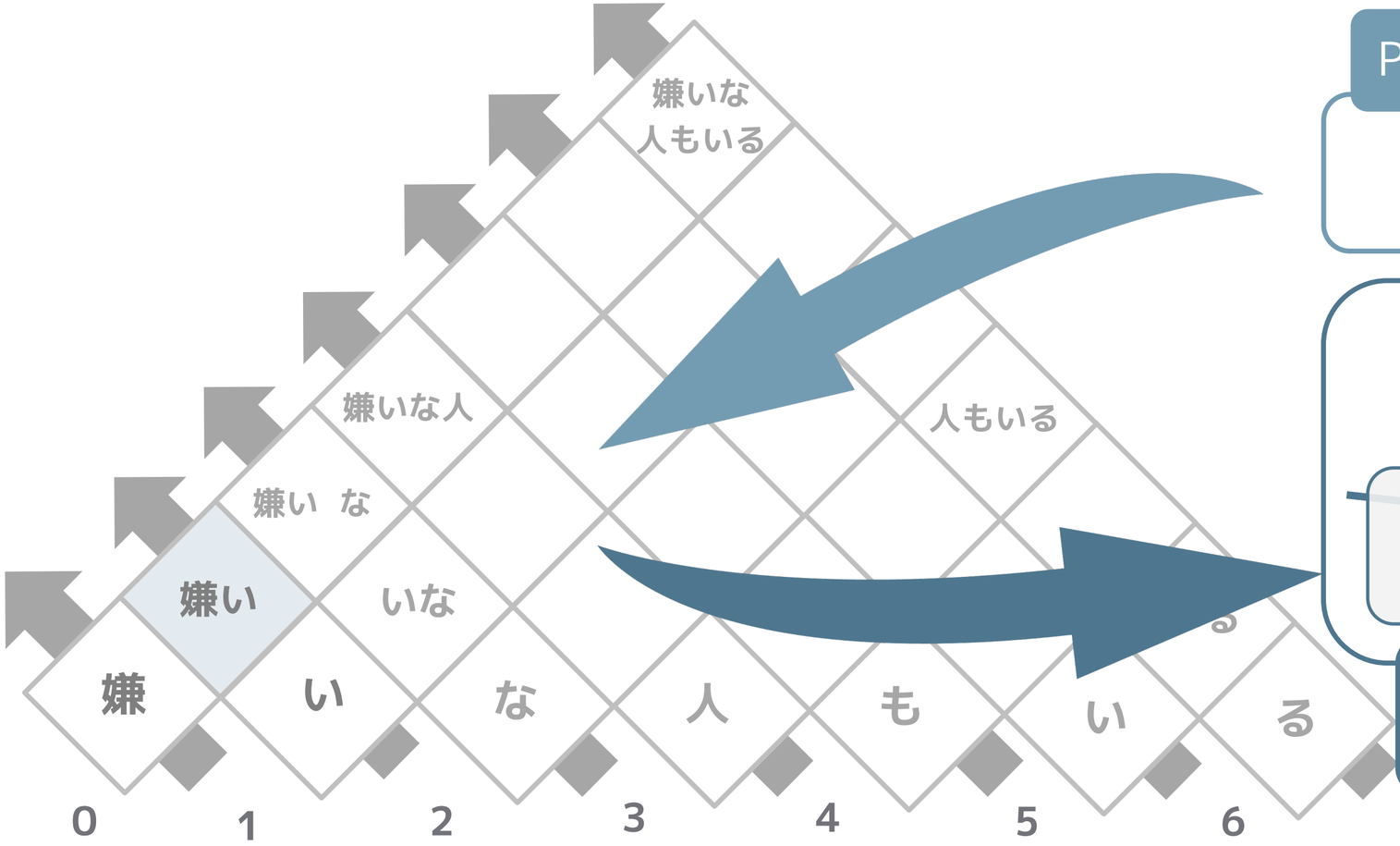
$S_{[n:da|n:na|n:ni|+][nstem]} \setminus NP_{ga}$

嫌う/きらう/ガヲ

$S_{[v:5:w<1>][cont|mod:m]} \setminus NP_{ga} \setminus NP_{o}$

Syntactic information of lexical items in lightblue lexicon

Reforging – filtration of the lightblue lexicon chart



Predicate data from ABCTreebank

[("嫌い", PPs \ SreI ,0,1)]

嫌い/きらい

$S_{[n:da|n:na|n:ni|+][nstem]} \setminus NP_{ga}$

嫌う/きらう/ガ

$S_{[v:5:w<1>][cont|mod:m]} \setminus NP_{ga}$

Syntactic information of lexical items in lightblue lexicon

Reforging – reconstruction of the treebank

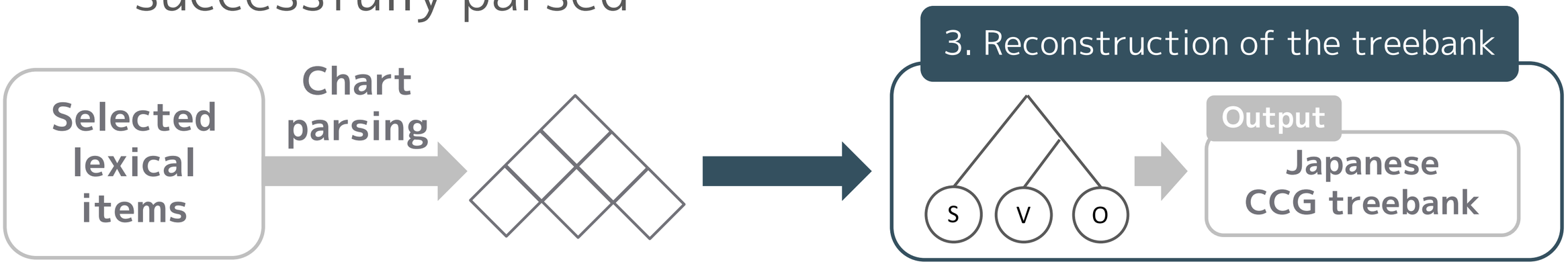
1. Extraction of predicates from ABCTreebank
2. Filtration of the lightblue lexicon chart
3. Reconstruction of the treebank

Reforging – reconstruction of the treebank

Step 3-1:

Parse the sentence in ABCTreebank, using the filtered chart

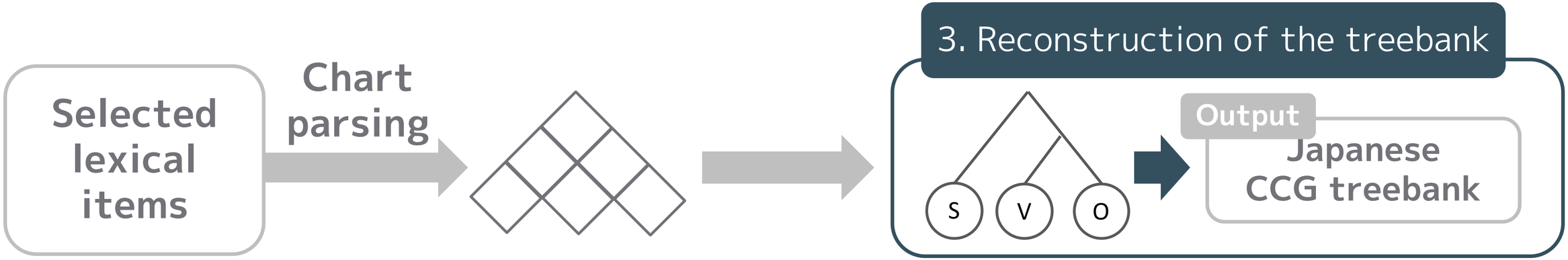
- The CCG syntactic structure is output when successfully parsed



Reforging – reconstruction of the treebank

Step 3-2:

Parse all sentences in ABCTreebank and convert them to a treebank format



Example of successful reforging

Input: 会議が始まった

(The meeting began)

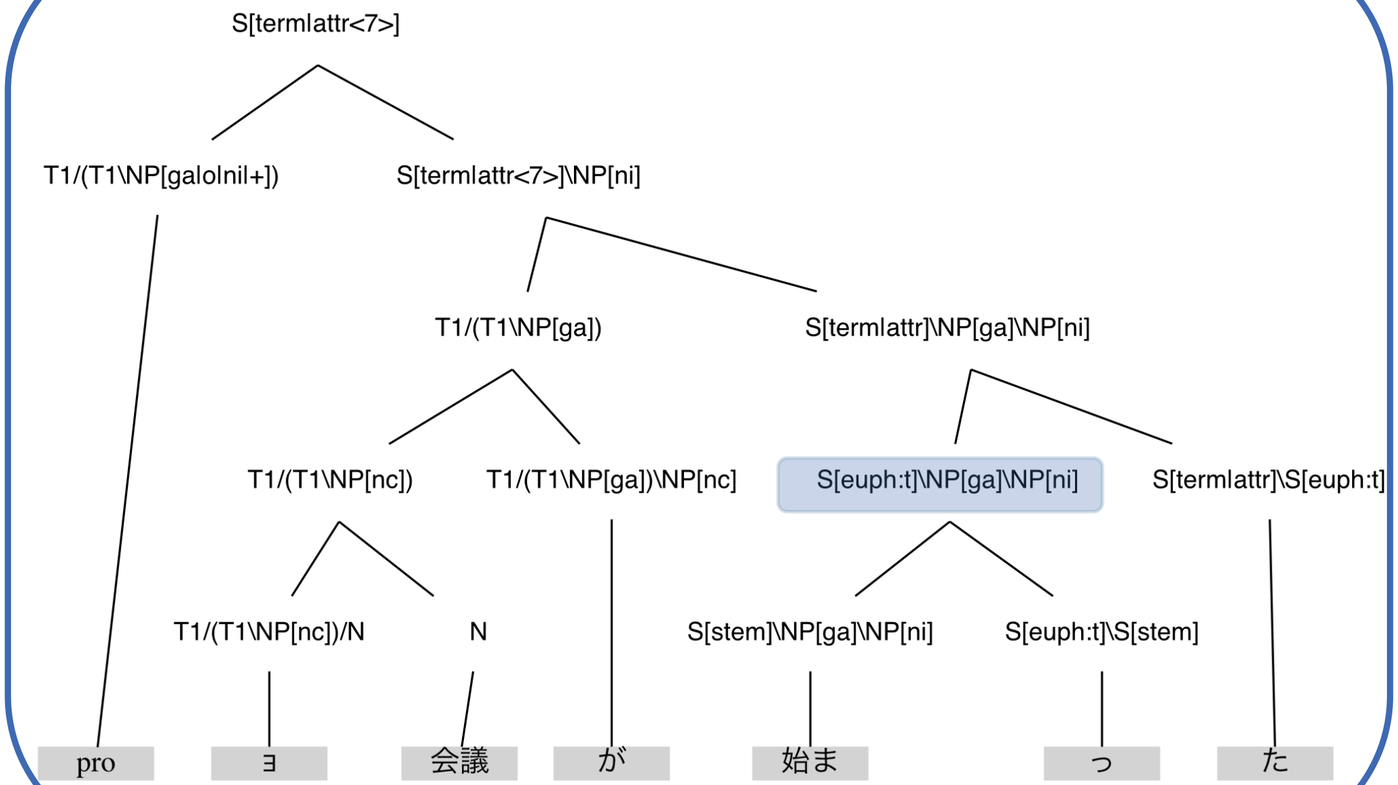
- lightblue analyzes **Ga-case and Ni-case NPs** as arguments

Syntactic information of lexical items in the lightblue lexicon

$S_{[v:5:r<1>][euph:t]} \setminus NP_{ga} \setminus NP_{ni}$

$S_{[v:5:r<1>][euph:t]} \setminus NP_{ga}$

Before reforging



Example of successful reforging

Input: 会議が始まった

(The meeting began)

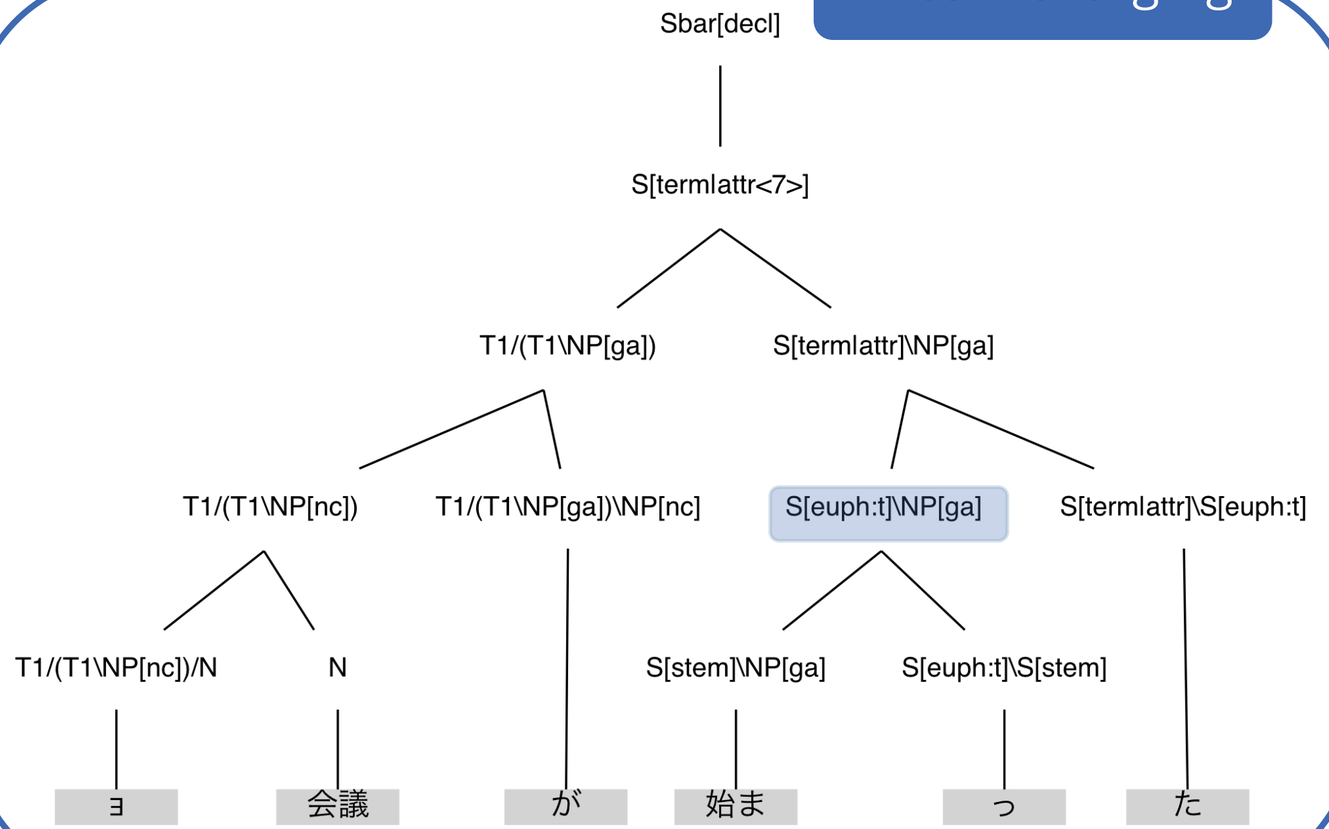
- An argument was changed to **Ga-case NP**
- One “pro” was removed

Syntactic information of lexical items in the lightblue lexicon

$S_{[v:5:r<1>][euph:t]} \setminus NP_{ga}$

$S_{[v:5:r<1>][euph:t]} \setminus NP_{ga}$

After reforging



Error analysis

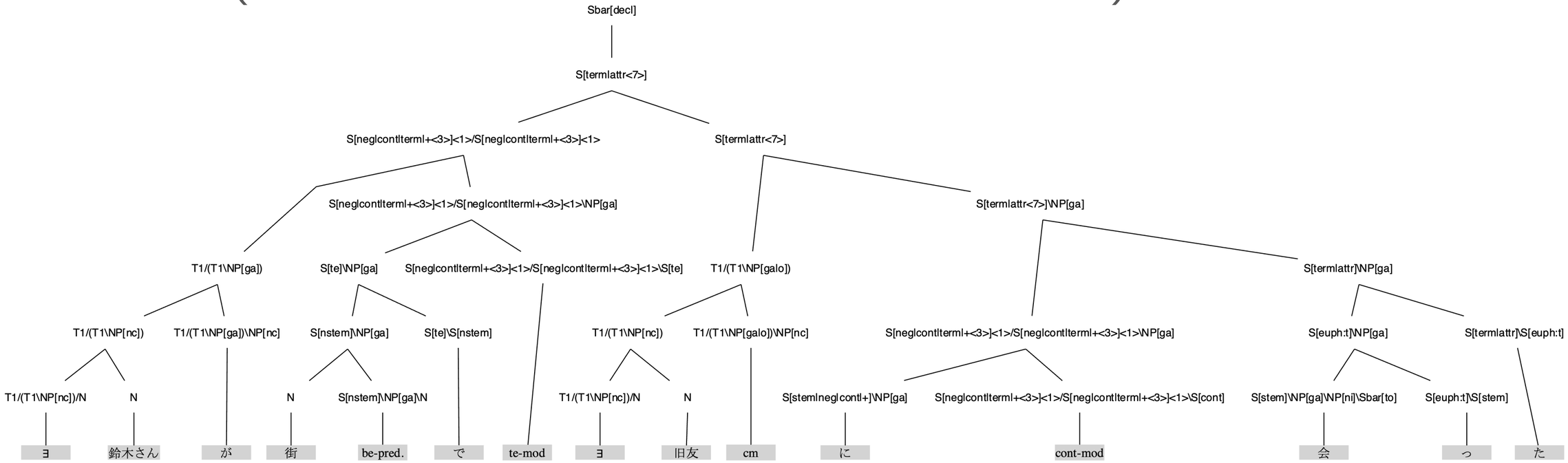
Possible causes of error:

1. Errors caused by **incorrect argument structures** in ABCTreebank
2. Errors caused by **missing entries** in the lightblue lexicon
3. Errors caused by incorrect analysis of adnominal clause in lightblue

Error analysis
– Incorrect argument structure in ABCTreebank

Input: 鈴木さんが街で旧友に会った

(Mr. Suzuki met an old friend in town)

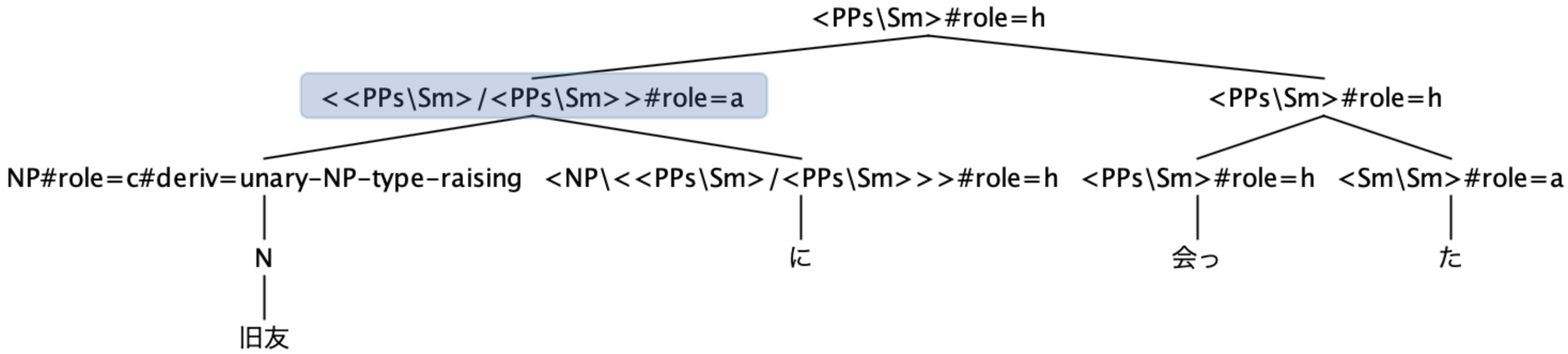


Error analysis
– Incorrect argument structure in ABCTreebank

Input: 鈴木さんが街で旧友に会った

(Mr. Suzuki met an old friend in town)

Argument structure in ABCTreebank



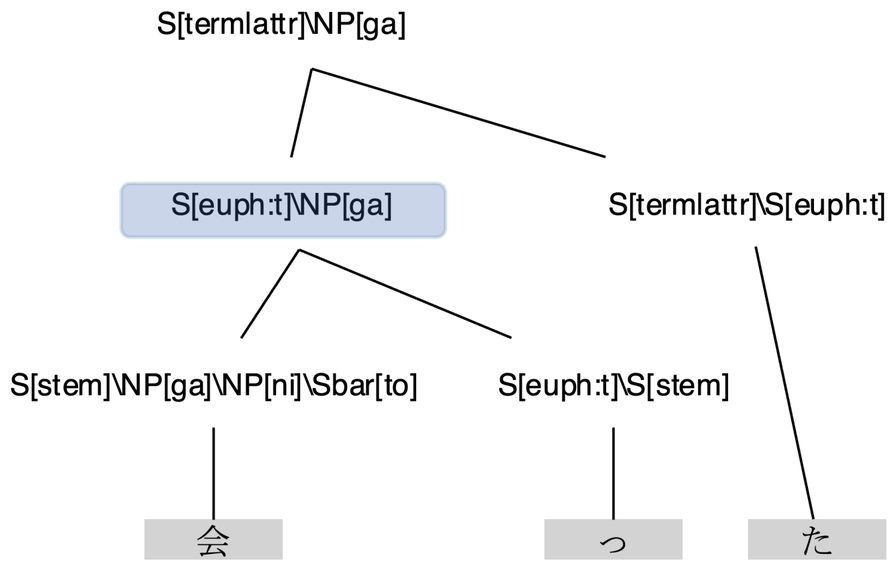
- ABCTreebank analyzed「旧友に」as an **adverb phrase**, but it should be analyzed as a **Ni-case noun phrase (PPo2)**

Error analysis
– Incorrect argument structure in ABCTreebank

Input: 鈴木さんが街で旧友に会った

(Mr. Suzuki met an old friend in town)

Output of lightblue



lightblue analyzed **Ga-case NP** as the argument.

However,
the argument should be **Ga-case and Ni-case NPs.**

Conclusion

- We proposed a **"reforging"** method for constructing a linguistically valid Japanese CCG treebank with detailed syntactic features
- Using this method, we succeeded in outputting some correct Japanese CCG syntactic structures
- This study started with the assumption that argument structures of ABCTreebank are valid
  - However, the validity of argument structures of ABCTreebank is an upper bound

Conclusion

Future work

- Output a greater number of valid CCG syntactic structures
  → improve the filtering algorithm
- Obtain linguistically valid argument structures
  → investigate sources other than ABCTreebank

This research was supported by the
JST CREST project「知識と推論に基づいて言語で説明できるAIシステム」,
JSPS科研費 JP20K19868

References

1. Daisuke Bekki and Ai Kawazoe. Implementing variable vectors in a CCG parser.
   In Logical Aspects of Computational Linguistics. Celebrating 20 Years of LACL (1996–2016), pp. 52–67, Berlin, Heidelberg, 12 2016. Springer Berlin Heidelberg.
2. Daisuke Bekki and Hitomi Yanaka. Is Japanese CCGBank empirically correct? A case study of passive and causative constructions.
   In Proceedings of Treebanks and Linguistic Theories (TLT) 2023 (to appear), the workshop in the Georgetown University Round Table on Linguistics 2023 (GURT2023), forthcoming.
3. Daisuke Kawahara and Sadao Kurohashi. A fullylexicalized probabilistic model for Japanese syntactic and case structure analysis. In Proc. of the Human Language Technology Conference of the NAACL, Main Conference, June 2006.
4. Yoshikawa Masashi, Noji Hiroshi, and Matsumoto Yuji. A* CCG parsing with a supertag and dependency factored model.
   In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pp. 277–287, Vancouver, Canada, 2017. Association for Computational Linguistics.
5. Ryu Iida, Mamoru Komachi, Kentaro Inui, and Yuji Matsumoto. 2007. Annotating a Japanese text corpus with predicate-argument and coreference relations. In Proceedings of Linguistic Annotation Workshop, pages 132–139.
6. Mark Steedman. The Syntactic Process. MIT Press, 2000.
7. Mark J. Steedman. Surface Structure and Interpretation. The MIT Press, Cambridge, 1996.
8. 植松すみれ, 松崎拓也, 花岡洋輝, 宮尾祐介, 美馬秀樹. 統語・意味コーパスの統合と再解釈による大規模な日本語CCG文法の開発.
   人工知能学会全国大会論文集, Vol. JSAI2013, pp. 4B11–4B11, 2013.
9. 戸次大介. 日本語文法の形式理論. くろしお出版, 東京, 2010.
10. 窪田悠介, 峯島宏次, 林則序, 岡野伸哉. 汎用的な範疇文法 ツリーバンクの構築.
    言語処理学会 第 25 回年次大会 発 表論文集 (2019 年 3 月), pp. 143–146. 一般社団法人 言 語処理学会, 2019.
11. 窪田悠介, 峯島宏次, 林則序, 岡野伸哉. ABC ツリーバン ク:学際的な言語研究のための基盤資源.
    言語処理学会 第 27 回年次大会 発表論文集 (2021 年 3 月), pp. 1529–1534. 一般社団法人 言語処理学会, 2021.
12. 花岡洋輝, 増田勝也, 植松すみれ, 美馬秀樹. 日本語助詞「と」コーパスの構築.
    言語処理学会 第 18 回年次大会 発 表論文集 (2012 年 3 月), pp. 247–250. 一般社団法人 言語処理学会, 2012.