漢文の依存文法解析にもとづく自動訓読システム

安岡孝一 (やすおかこういち)*

1　はじめに

日本における漢文 (古典中国語) の受容は、一つには訓読という形でおこなわれてきた。訓読は、言語処理という観点から見た場合、VO 型の孤立語である古典中国語を、OV 型の膠着語である日本語の読み下し文に変換する、という過程の一種だとみなせる。訓読を、返り点と送り仮名に分けたならば、VO 型から OV 型への変換を返り点が担い、孤立語から膠着語への変換を送り仮名が担っている、と考えることもできるだろう。

一方、コンピュータによる漢文の自動解析は、形態素解析→依存文法解析→直接構成鎖解析という手順によって、白文の統語構造を得ることができる、というのが、発表者の目論見[1]である。入力された白文に対し、形態素解析によって、単語切りをおこなうと同時に、各単語の品詞を得る。依存文法解析によって、単語と単語の間の係り受け関係を解析すると同時に、文の切れ目を得る。直接構成鎖解析によって、各文の統語構造を解析木の形で得る。

ただし、コンピュータによる返り点の自動付与に関しては、直接構成鎖解析をおこなう必要は無く、依存文法解析までで (ほぼ) 十分であることを、発表者は既に[2]示した。卑近な言い方をすれば、訓読の返り点は、漢文の統語構造ではなく、係り受け関係によって (ほぼ) 記述可能である。一方、送り仮名の自動付与に関しては、膠着語としての日本語を特徴づける格助詞・接続助詞・活用語尾などを、各単語ないし構成鎖の末尾に付与する、という問題に帰着できる。

これらのアイデアにもとづき、漢文自動訓読システム UD-Kundoku の開発をおこなった。UD-Kundoku は、入力された白文に対し、最初に依存文法解析をおこない、次に返り点の自動付与をおこなって語順を入れ替え、最後に送り仮名を自動付与する、という python3 モジュールである。

2　Universal Dependencies による漢文の依存文法解析

発表者が班長を務める京都大学人文科学研究所共同研究班「古典中国語のコーパスの研究」(班員: ウィッテルン クリスティアン、守岡知彦、池田巧、山崎直樹、二階堂善弘、鈴木慎吾、師茂樹、白須裕之、藤田一乘) では、現在、漢文の依存文法解析に精力を傾注し

*京都大学人文科学研究所附属東アジア人文情報学研究センター

[1]安岡孝一: 漢文の形態素解析・依存文法解析・直接構成鎖解析, 東方學報, 第 94 册 (2019 年 12 月), pp.330-322.

[2]安岡孝一: 漢文の依存文法解析と返り点の関係について, 日本漢字学会第 1 回研究大会予稿集 (2018 年 12 月 1 日), pp.33-48.

表 1: 古典中国語に対する UD 依存構造タグ

	Nominals	Clauses	Modifier Words	Function Words
Core arguments	nsubj 主語 ↪nsubj:pass [受動文] obj 目的語 iobj 間接目的語	csubj 節主語 ccomp 節目的語 xcomp 節補語		
Non-core arguments	obl 斜格補語 ↪obl:tmod [時] ↪obl:lmod [場所] vocative 呼称語 expl 形式語 dislocated 外置語	advcl 連用修飾節	advmod 連用修飾語 discourse 談話要素 ↪discourse:sp [文助詞]	aux 動詞補助成分 cop 繋辞 (copula) mark 標識 (marker)
Nominal dependents	nmod 体言による連体修飾語 nummod 数量による修飾語	acl 連体修飾節	amod 用言による連体修飾語	det 決定詞 clf 類別詞 case 格表示
Coordination	**MWE**	**Loose**	**Special**	**Other**
conj 接続 cc 接続詞	fixed 固着 compound 複合 (endocentric) ↪compound:redup [重畳] flat 並列 (exocentric) ↪flat:vv [動詞類]	list 細目 parataxis 隣接表現	orphan 親なし	root 親

ており、その道具立ての一つとして、Universal Dependencies [3](以下「UD」) の古典中国語への適用を研究している。依存文法解析それ自体は、Tesnière の構造的統語論[4]に源を発し、Мельчук の有向グラフ記述[5]によって、一応の完成を見た手法である。その最大の特長は、言語横断的な記述が可能だという点にあり、Мельчук の手法をコンピュータ向けに洗練した UD においても、言語に関わらない記述、という特長が前面に押し出されている。UD における文法構造記述は、句構造を考慮せず、全てを単語間のリンクとして表現する。これは、Мельчук の有向グラフ記述が、単語間のリンクという形態を取っていたからであり、そういう割り切りの結果として、言語横断的な文法構造記述を可能としているのである。

UD における係り受け関係の記述は、文中の単語をノードとする有向グラフにおいて、単語間の依存関係をリンクで表現する。各単語から出るリンクは複数ありうるが、各単語に入るリンクは必ず 1 本とする。また、リンクはループしない。リンクには、それぞれ UD 依存構造タグを付与する。古典中国語 UD においては、表 1 に示す 38 種類[6]のタグを使用している。タグのうち 32 種類は、もともと UD で規定されているものであり、6 種類 (nsubj:pass・obl:tmod・obl:lmod・discourse:sp・compound:redup・flat:vv) は、その派

[3]Joakim Nivre: Towards a Universal Grammar for Natural Language Processing, CICLing 2015: 16th International Conference on Intelligent Text Processing and Computational Linguistics (April 2015), pp.3-16.

[4]Lucien Tesnière: Éléments de Syntaxe Structurale, Paris: C. Klincksieck (1959).

[5]Igor A. Mel'čuk: Dependency Syntax: Theory and Practice, New York: State University of New York Press (1988).

[6]Koichi Yasuoka: Universal Dependencies Treebank of the Four Books in Classical Chinese, DADH2019: 10th International Conference of Digital Archives and Digital Humanities (December 2019), pp.20-28.

生形である。root はリンク元を持たないが、他のタグによるリンクは、リンク元の単語とリンク先の単語を 1 つずつ有する。たとえば、漢文の動賓構造は、動詞をリンク元、賓語をリンク先、とする obj というリンクで表現する。

白文に対する依存文法解析は、その前段階として、単語切りという処理を必要とする。白文では、単語と単語の間に区切りがないことから、単語というものを処理単位とする依存文法解析に際し、まず、白文を単語に区切る処理が必要となるのである。この処理をわれわれは、漢文の形態素解析[7]という形で実現し、白文の単語切りをおこなうと同時に、各単語に対して 4 階層の品詞を得ている。また、この際に、UD 向け品詞 (PROPN・NOUN・PRON・NUM・VERB・ADP・ADV・AUX・PART・SCONJ・CCONJ・INTJ・SYM) も同時に得ている。

依存文法解析のための手法は、これまでに数多く提案されているが、われわれの古典中国語 UD のように、複数の root を持ち (dependency forest)、UD 依存構造のリンクどうしが交差せず (planar)、root をまたぐリンクがない (projective)、という条件においては、arc-planar [8]という (非決定性) アルゴリズムが、有効だと考えられる。arc-planar は、単語列の先頭から末尾に向かって「垣根」(stack-buffer boundary) を移動していく、というイメージで処理をおこなう。「垣根」がおこなう遷移は、**Shift・Reduce・Left-Arc・Right-Arc** の 4 種類である。

- **Shift** 「垣根」を右に 1 単語分、移動する。
- **Reduce** 「垣根」のすぐ左の単語を除去して、解析結果へ移す。
- **Left-Arc** 「垣根」のすぐ右の単語から、すぐ左の単語へリンクを繋ぐ。
- **Right-Arc** 「垣根」のすぐ左の単語から、すぐ右の単語へリンクを繋ぐ。

単語が全て **Reduce** されて、「垣根」がポツンと取り残された時点で、arc-planar は終了である。arc-planar による「孟子見梁惠王王曰叟不遠千里而來」の依存文法解析の様子を、図 1 に示す。あとは、リンクが入っていない「見」「曰」「遠」に root を刺すことで、依存文法解析結果が得られるわけである。

ただし、arc-planar における「垣根」の遷移は、実際には非決定的である。図 1 では解析過程を一本道で示したが、現実には、各局面において複数の可能性が、枝分かれとして存在する。これら複数の可能性については、それぞれの遷移を選択した場合を、確率的に並行して解析することになる。

[7]安岡孝一, ウィッテルン クリスティアン, 守岡知彦, 池田巧, 山崎直樹, 二階堂善弘, 鈴木慎吾, 師茂樹: 古典中国語 (漢文) の形態素解析とその応用, 情報処理学会論文誌, Vol.59, No.2 (2018 年 2 月), pp.323-331.

[8]Carlos Gómez-Rodríguez, Joakim Nivre: A Transition-Based Parser for 2-Planar Dependency Structures, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (July 2010), pp.1492-1501.

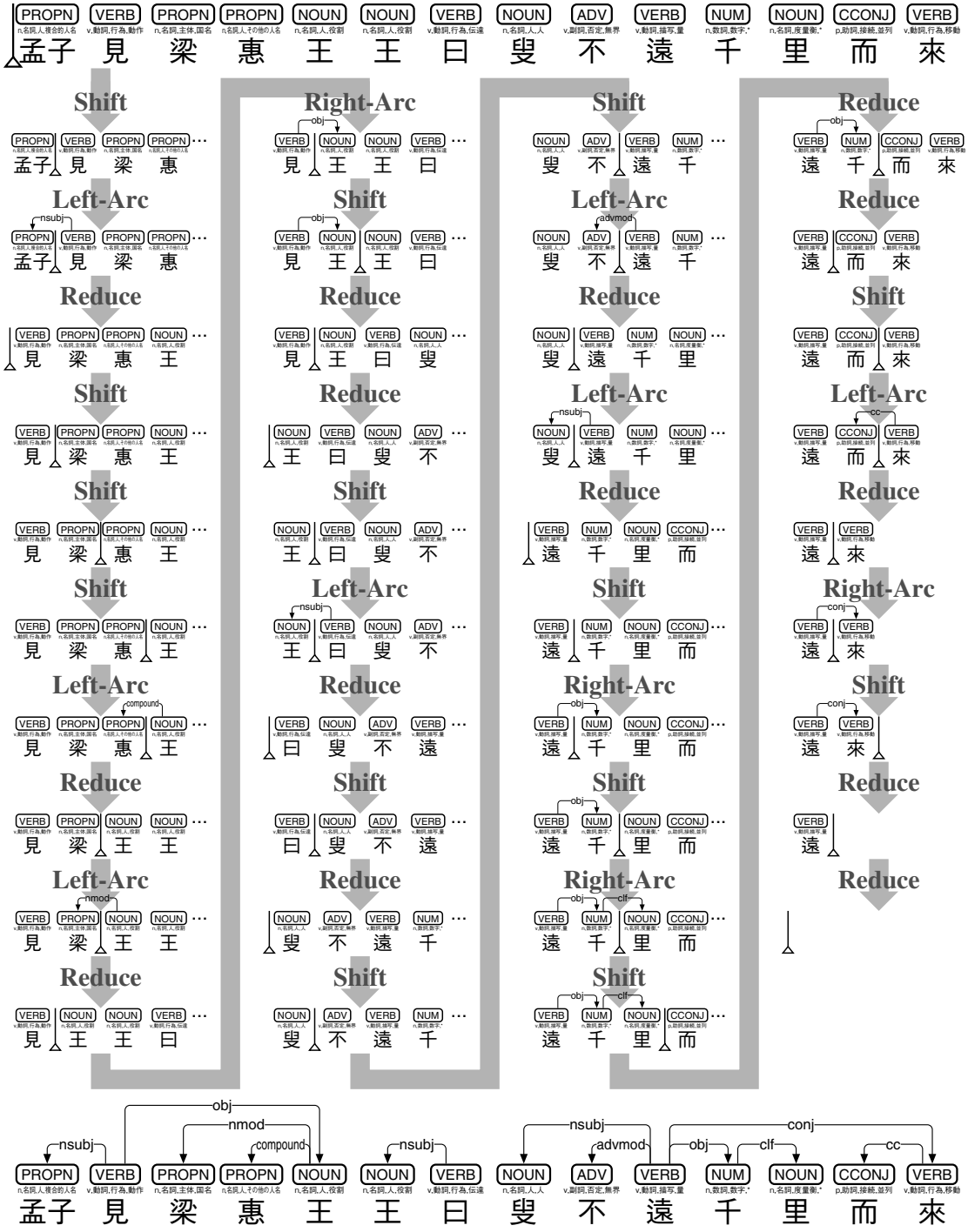

図 1: arc-planar による漢文の依存文法解析

3 漢文の依存文法解析にもとづく返り点の自動付与

漢文の依存文法解析結果から、以下に示す 24 のルールに従い、返り点を自動付与する。各ルールの詳細については、文献[2]を参照されたい。

3.1 右向きの UD リンクに対する返り点のルール

ルール① 右向きの obj リンクは、リンク先からリンク元へ返り点を打つ。

ルール② 右向きの obl リンクは、リンク先からリンク元へ返り点を打つ。

ルール③ 右向きの expl リンクは、リンク先からリンク元へ返り点を打つ。

ルール④ 右向きの ccomp および xcomp リンクは、リンク先からリンク元へ返り点を打つ。

3.2 左向きの UD リンクに対する返り点のルール

ルール⑤ 左向きの cop リンクは、リンク元からリンク先へ返り点を打つ。

ルール⑥ 左向きの case および mark リンクは、リンク元からリンク先へ返り点を打つ。ただし、リンク先の形態素解析結果が「v, 前置詞, 基盤」の場合は、返り点を打たない。

ルール⑦ 左向きの aux リンクは、リンク元からリンク先へ返り点を打つ。

ルール⑧ 左向きの advmod リンクは、リンク先の形態素解析結果が「v, 副詞, 否定」「v, 副詞, 判断, 逆接」「v, 副詞, 時相, 将来」の場合、あるいはリンク先が「難」「易」の場合に限って、リンク元からリンク先へ返り点を打つ。

ルール⑨ 左向きの cc リンクは、リンク先の形態素解析結果が「v, 前置詞, 関係」の場合に限って、リンク元からリンク先へ返り点を打つ。

3.3 個別の文字に対するルール

ルール⑩ ルール①〜④で打った返り点において、返り先が「況」の場合、返り点を削除する。ただし、返り点の返り元から、左向きの case あるいは mark リンクが出ている場合は、返り点を削除する代わりに、返り先をそのリンク先に移動する。

ルール⑪ ルール①〜④で打った返り点において、返り先が「謂」であり、かつ、その「謂」から「所」へ左向きの mark リンクが出ている場合、「謂」を返り先とする返り点を削除する。

ルール⑫ ルール①あるいは④で打った返り点において、返り先が「請」であり、かつ、その「請」から vocative リンクが出ている場合、返り点を削除する。

ルール⑬ ルール②で打った返り点において、返り元が「焉」であり、かつ、返り先の形態素解析結果が「v, 動詞, 描写」以外の場合、返り点を削除する。

ルール⑭ ルール④で打った返り点において、返り先が「如」であり、かつ、その「如」から obj リンクもしくは expl リンクが出ている場合、返り点を削除する。

ルール⑮ ルール④で打った返り点において、返り先が「助」であり、かつ、xcomp リンクによる場合、返り点を削除する。加えて、その「助」が他のルールによる返り点の返り元である場合、返り元を、削除した返り点の返り元へ移動する。

ルール⑯ ルール④で打った返り点において、返り先が「勸」であり、かつ、xcomp リンクによる場合、返り点を削除する。加えて、その「勸」が他のルールによる返り点の返り元である場合、返り元を、削除した返り点の返り元へ移動する。

ルール⑰ ルール⑥で打った「以」から「所」への返り点において、その「以」に左向きの advmod リンクが入っている場合、返り元を、advmod リンクのリンク元へ移動する。

ルール⑱ ルール⑥で打った「以」から「所」への返り点において、その「以」に左向きの advmod リンクが入っていない場合、返り点を削除する。加えて、その「以」が他のルールによる返り点の返り先・返り元である場合、それらの返り先・返り元を「所」へ移動する。

ルール⑲ ルール⑦で打った返り点において、返り先が「能」であり、かつ、その「能」が他のルールによる返り点の返り元でない場合、「能」を返り先とする返り点を削除する。

ルール⑳ ルール⑦で打った返り点において、返り先が「敢」の場合、返り点を削除する。加えて、その「敢」が他のルールによる返り点の返り元である場合、返り元を、削除した返り点の返り元へ移動する。

ルール㉑ ルール⑦で打った返り点において、返り先が「得」であり、かつ、その「得」から「而」へ右向きの advmod リンクが出ている場合、返り点を削除する。加えて、その「得」が他のルールによる返り点の返り元である場合、返り元を、削除した返り点の返り元へ移動する。

3.4 返り点の付け替えに対するルール

ルール㉒ ルール①〜㉑で打った返り点 (ルール⑧を除く) の返り元から、右向きの conj・clf・flat・case リンクが出ている場合、それらのリンク先のうち文末に最も近いものへ、返り元を移動する。

ルール㉓ ルール①〜㉒で打った返り点において、1 つの返り先へ複数の返り元から返り点が集中している場合、それらの返り元のうち文末に最も近いものを残し、他の返り元は削除する。

ルール㉔ ルール①〜㉓で打った返り点において、1 つの返り元から複数の返り先へ返り点がある場合、それらの返り先のうち文頭に最も近いもの以外に返り元を追加し、返り点を後ろから順に辿る形とする。

4 訓読における送り仮名の自動付与

返り点によって語順を入れ替えた古典中国語 UD に対し、送り仮名を付与して、書き下し文にする。訓読における送り仮名は、助詞と活用語尾に分けられることから、それぞれについて考えてみよう。

4.1 助詞の付与

目的語を表す格助詞

obj リンクおよび ccomp リンクについては、リンク先の単語を始点とする構成鎖の末尾に、目的語を表す格助詞を付与する。目的語を表す格助詞は「を」が代表的である。たとえば「王顧左右而言他」では、返り点「王顧二左右一而言レ他」にもとづいて語順を入れ替えた上で、「左右」(「左」を始点とする構成鎖) と「他」の後に「を」を付与する (図 2)。

ただし、動詞の種類によっては、目的語を表す格助詞に「を」以外が適切な場合もある。現時点での UD-Kundoku の実装を表 2 に示すが、今後さらなる改良をおこなう必要があるだろう。

表 2: 目的語を表す格助詞に「を」以外が適切な場合

格助詞	4 階層品詞	備考
に	v, 動詞, 行為, 移動	
に	v, 動詞, 行為, 伝達	obj リンク先が「n, 代名詞, 人称」「n, 名詞, 人」の場合
と	v, 動詞, 行為, 伝達	助動詞の「可」を伴う場合
の	v, 動詞, 行為, 分類	
が	v, 動詞, 行為, 分類	obj・ccomp リンク先が動詞の場合
をして	v, 動詞, 行為, 使役	
	v, 動詞, 存在, 存在	「有」「無」は格助詞を伴わない

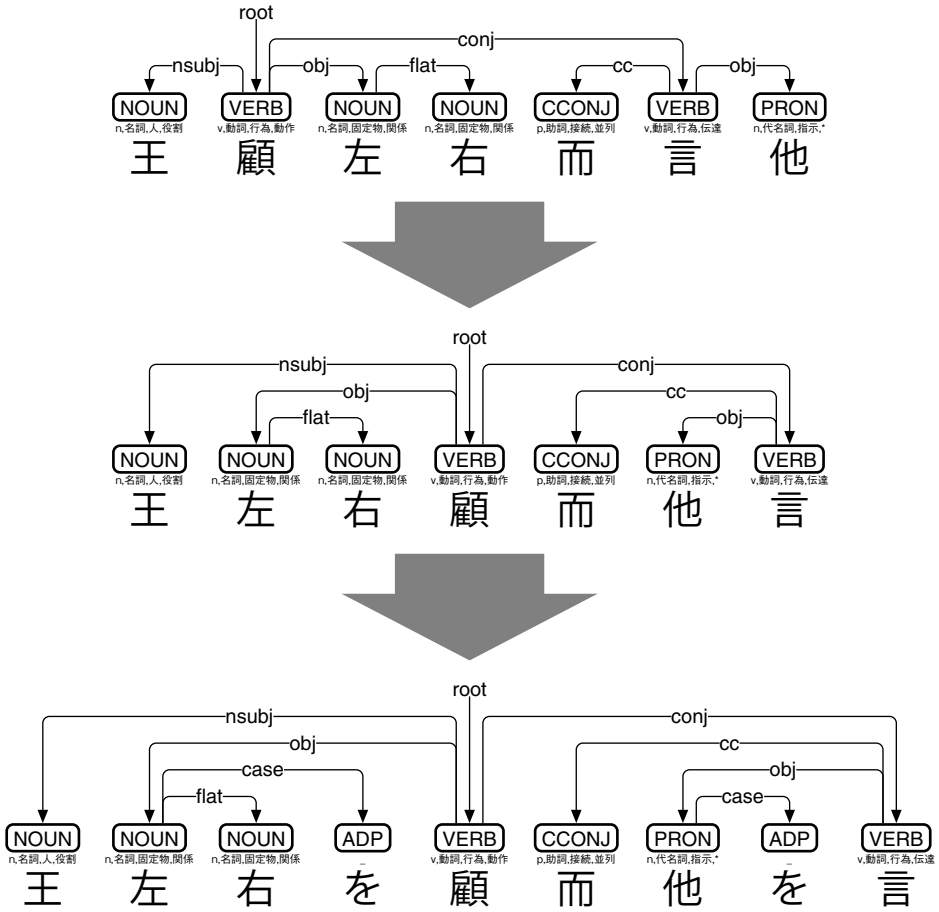

図 2: 「王顧左右而言他」に対する語順入れ替えと格助詞「を」の付与

斜格補語を表す格助詞

obl リンクについては、リンク先の単語を始点とする構成鎖の末尾に、斜格補語を表す格助詞を付与する。斜格補語を表す格助詞は「に」が代表的だが、case リンクを伴う場合は、その先にある前置詞の種類に応じて格助詞を変更する (表 3) とともに、前置詞を格助詞に同化させる。一方、case リンクを伴わない代名詞の「自」に対しては、格助詞を付与せずに「自」を「自ら」(みずから) とする。

表 3: 斜格補語を表す格助詞に「に」以外が適切な場合

格助詞	4 階層品詞	備考
より	v, 前置詞, 経由	
より	v, 前置詞, 基盤	obl リンク元が「v, 動詞, 描写, 量」の場合
ゆえ	v, 前置詞, 源泉	
と	v, 前置詞, 関係	

主語を表す格助詞

nsubj リンクおよび csubj リンクについては、リンク先の単語を始点とする構成鎖の末尾に、主語を表す格助詞「は」を付与する。ただし、主語を表す格助詞が複数並ぶ場合は、2 つ目以降は「が」を用いる。一方、case リンクに「之」(p, 助詞, 接続, 属格) を伴っている場合は、格助詞を付与せずに「之」を「の」とする。

所有を表す格助詞

nmod リンクおよび det リンクについては、リンク先の単語を始点とする構成鎖の末尾に、所有を表す格助詞「の」を付与する。ただし、case リンクに「之」(p, 助詞, 接続, 属格) を伴っている場合は、格助詞を付与せずに「之」を「の」とする。

接続助詞

左向きの advcl リンクが「則」(v, 副詞, 時相, 緊接) をまたいでいる場合は、リンク先の単語を始点とする構成鎖の末尾に、接続助詞「ば」を付与する。「而」(p, 助詞, 接続, 並列) は「而して」とする。

句末の助詞

句末の助詞 (p, 助詞, 句末) のうち、「乎」は「や」とする。「已」と「耳」は「のみ」とする。「兮」は「よ」とする。「也」は「なり」とした上で、ナリ活用 (表 5 参照) をおこなう。その他の句末の助詞は「か」とするが、句末の助詞が複数並ぶ場合は読まない。また、「也」(p, 助詞, 提示) は「なる」とする。

句頭の助詞

句頭の助詞 (p, 助詞, 句頭) のうち、「蓋」は「けだし」とする。その他の句頭の助詞は「それ」とする。

4.2 活用語尾の付与

文中の動詞 (助動詞や一部の副詞を含む) には、以下に示す手順により活用語尾を付与する。なお、活用語尾の付与順序は、文末の動詞から文頭の動詞へと、順におこなう。

動詞の活用語尾

動詞 (日本語では形容詞となる場合を含む) は、表 4 にもとづき、次の単語との間に活用語尾を付与する。ただし、動詞と次の単語の間に、左向きの amod リンクが繋がっている場合、あるいは右向きの flat:vv リンクが繋がっている場合は、活用語尾を付与しない。動詞と次の単語の間に、左向きの advmod・advcl リンクが繋がっている場合、右向きの parataxis リンクが繋がっている場合、あるいは直接のリンクが無い場合は、連用形に「て」

を付与する。動詞と次の単語の間に、右向きの aux リンクが繋がっている場合は、以下のとおりとする。

- 次の単語が「得」であれば、連体形に「を」を付与する。
- 次の単語が「欲」であれば、未然形に「んと」を付与する。
- 次の単語が「被」「見」であれば、未然形とする。
- その他は、連体形とする。

表 4 に載っていない動詞については、旧仮名口語 UniDic [9]に活用の種類 (表 5) を問い合わせる。旧仮名口語 UniDic に問い合わせても活用の種類が判明しない動詞は、文語サ行変格とみなす。flat:vv リンクのリンク先にあたる動詞も、文語サ行変格とみなす。

ただし、「況」(v, 動詞, 行為, 動作) は活用せず「況んや」とする。「所以」は「以てする所」ではなく「ゆゑん」とする。「所謂」は「いはゆる」とする。「如何」「奈何」「若何」は「いかん」とする。

表 4: 動詞活用表 (一部)

動詞		未然形	連用形	終止形	連体形	已然形	命令形
有	v, 動詞, 存在, 存在	有ら	有り	有り	有る	有れ	有れ
無	v, 動詞, 存在, 存在	無から	無く	無し	無き	無けれ	無かれ
於	v, 動詞, 存在, 存在	於てせ	於	於てす	於てする	於てせ	於てせよ
為	v, 動詞, 存在, 存在	たら	たり	たり	たる	たれ	たれ
為	v, 動詞, 行為, 生産	為さ	為し	為す	為す	為せ	為せ
來	v, 動詞, 行為, 移動	來	來	來る	來る	來れ	來よ
之	v, 動詞, 行為, 移動	ゆか	ゆき	ゆく	ゆく	ゆけ	ゆけ
行	v, 動詞, 行為, 移動	行か	行き	行く	行く	行け	行け
行	v, 動詞, 行為, 動作	行は	行ひ	行ふ	行ふ	行へ	行へ
以	v, 動詞, 行為, 動作	以てせ	以	以てす	以てする	以てせ	以てせよ
顧	v, 動詞, 行為, 動作	顧み	顧み	顧みる	顧みる	顧みれ	顧みよ
易	v, 動詞, 行為, 動作	易へ	易へ	易へる	易へる	易へれ	易へよ
易	v, 動詞, 描写, 形質	易から	易く	易し	易き	易けれ	易くせよ
同	v, 動詞, 描写, 形質	同じとせ	同じとし	同じ	同じ	同じとすれ	同じとせよ
重	v, 動詞, 描写, 量	重ね	重ね	重ねる	重ねる	重ねれ	重ねよ
如	v, 動詞, 行為, 分類	如くあら	如く	如し	如き	如くあれ	如くあれ
曰	v, 動詞, 行為, 伝達	曰は	曰ひ	曰く	曰ふ	曰へ	曰へ
說	v, 動詞, 行為, 伝達	說か	說き	說く	說く	說け	說け
說	v, 動詞, 行為, 態度	說ば	說び	說ぶ	說ぶ	說べ	說べ
可	v, 動詞, 描写, 態度	可とせ	可とし	可とす	可とする	可とすれ	可とせよ
興	v, 動詞, 行為, 態度	興じ	興じ	興ず	興じる	興じれ	興じよ
興	v, 動詞, 行為, 動作	興さ	興し	興す	興す	興せ	興せ

[9]小木曽智信: 旧仮名遣いの口語文を対象とした形態素解析辞書, 人文科学とコンピュータ「じんもんこん 2012」論文集 (2012 年 11 月), pp.25-32.

表 5: 動詞活用表 (旧仮名口語 UniDic 問い合わせ用)

活用の種類	未然形	連用形	終止形	連体形	已然形	命令形
五段-バ行	□ば	□び	□ぶ	□ぶ	□べ	□べ
文語上二段-バ行	□び	□び	□ぶ	□びる	□びれ	□びよ
文語下二段-バ行	□べ	□べ	□ぶ	□べる	□べれ	□べよ
五段-ダ行	□だ	□ぢ	□づ	□づ	□で	□で
文語上二段-ダ行	□ぢ	□ぢ	□づ	□ぢる	□ぢれ	□ぢよ
文語下二段-ダ行	□で	□で	□づ	□でる	□でれ	□でよ
五段-サ行	□さ	□し	□す	□す	□せ	□せ
文語ザ行変格	□ぜ	□じ	□ず	□ずる	□じれ	□ぜよ
五段-マ行	□ま	□み	□む	□む	□め	□め
文語上二段-マ行	□み	□み	□む	□みる	□みれ	□みよ
文語下二段-マ行	□め	□め	□む	□める	□めれ	□めよ
五段-ワア行	□は	□ひ	□ふ	□ふ	□へ	□へ
文語四段-ハ行	□は	□ひ	□ふ	□ふ	□へ	□へ
文語上二段-ハ行	□ひ	□ひ	□ふ	□ひる	□ひれ	□ひよ
文語下二段-ハ行	□へ	□へ	□ふ	□へる	□へれ	□へよ
五段-タ行	□た	□ち	□つ	□つ	□て	□て
文語上二段-タ行	□ち	□ち	□つ	□ちる	□ちれ	□ちよ
文語下二段-タ行	□て	□て	□つ	□てる	□てれ	□てよ
五段-ガ行	□が	□ぎ	□ぐ	□ぐ	□げ	□げ
文語上二段-ガ行	□ぎ	□ぎ	□ぐ	□ぎる	□ぎれ	□ぎよ
文語下二段-ガ行	□げ	□げ	□ぐ	□げる	□げれ	□げよ
文語形容詞-ク	□から	□く	□し	□き	□けれ	□くせよ
文語形容詞-シク	□しから	□しく	□し	□しき	□しけれ	□しくせよ
五段-カ行	□か	□き	□く	□く	□け	□け
文語上二段-カ行	□き	□き	□く	□きる	□きれ	□きよ
文語下二段-カ行	□け	□け	□く	□ける	□けれ	□けよ
五段-ラ行	□ら	□り	□る	□る	□れ	□れ
上一段	□	□	□る	□る	□れ	□よ
下一段	□	□	□る	□る	□れ	□よ
ナリ活用	□なら	□なり	□なり	□なる	□なれ	□なれ
文語サ行変格	□せ	□し	□す	□する	□すれ	□せよ

助動詞の活用語尾

助動詞は、表 6 にもとづいて、次の単語との間に活用語尾を付与する。「須」「宜」「儀」については再読文字とせず、「〜すべし」とだけ読んでいる。なお、「敢」(v, 助動詞, 願望) は活用せず「敢へて」とし、直後が動詞以外の場合に限って「敢へてす」(文語サ行変格) で活用する。同様に、「肯」(v, 助動詞, 願望) は活用せず「肯へて」とし、直後が動詞以外の場合に限って「肯へてす」(文語サ行変格) で活用する。「能」(v, 助動詞, 可能) は活用せず「能く」とし、直後が否定の場合に限って「能ふ」(文語四段-ハ行) で活用する。

表 6: 助動詞活用表

助動詞		未然形	連用形	終止形	連体形	已然形	命令形
得	v, 助動詞, 可能	得	得	得る	得る	得れ	得よ
可	v, 助動詞, 可能	べから	べく	べし	べき	べけれ	べけれ
須	v, 助動詞, 必要	べから	べく	べし	べき	べけれ	べけれ
宜	v, 助動詞, 必要	べから	べく	べし	べき	べけれ	べけれ
儀	v, 助動詞, 必要	べから	べく	べし	べき	べけれ	べけれ
欲	v, 助動詞, 願望	欲さ	欲し	欲す	欲する	欲せ	欲せよ

特殊な副詞の活用語尾

副詞のうち「v, 副詞, 否定」と「v, 副詞, 時相, 将来」については、表 7 にもとづいて活用する。ただし、「未」(v, 副詞, 否定, 有界) は「未だ」と「ず」で動詞を挟み込み、「ず」を表 7 にもとづいて活用することで、いわゆる再読文字とする。他の副詞については、ひらがなに置き換えるか「に」を付与することで、日本語の副詞に近づける。

表 7: 特殊な副詞の活用表

副詞		未然形	連用形	終止形	連体形	已然形	命令形
不	v, 副詞, 否定, 無界	ざら	ずし	ず	ざる	ざれ	ざれ
弗	v, 副詞, 否定, 無界	ざら	ずし	ず	ざる	ざれ	ざれ
未	v, 副詞, 否定, 有界	ざら	ずし	ず	ざる	ざれ	ざれ
毋	v, 副詞, 否定, 禁止	なから	なく	なかれ	なき	なけれ	なかれ
勿	v, 副詞, 否定, 禁止	なから	なく	なかれ	なき	なけれ	なかれ
莫	v, 副詞, 否定, 禁止	なから	なく	なかれ	なき	なけれ	なかれ
非	v, 副詞, 否定, 体言否定	非ざら	非ずし	非ず	非ざる	非ざれ	非ざれ
將	v, 副詞, 時相, 将来	んとせ	んとし	んとす	んとする	んとすれ	んとせよ
且	v, 副詞, 時相, 将来	んとせ	んとし	んとす	んとする	んとすれ	んとせよ

5 漢文自動訓読システム UD-Kundoku

ここまでに述べてきた手法を用いて、漢文自動訓読システム UD-Kundoku を python3 モジュールとして実装した。UD-Kundoku は、いわゆる encode-reorder-decode モデル[10]を採用しており、encode と reorder と decode の 3 つのステップから構成される。

[10]Josep M. Crego and José B. Mariño: Integration of POStag-based Source Reordering into SMT Decoding by an Extended Search Graph, AMTA2006: 7th Conference of the Association for Machine Translation in the Americas (August 2006), pp.29-36.

1. 白文を依存文法解析して、古典中国語 UD を生成する (encode)
2. 返り点にもとづいて、語順を入れ替える (reorder)
3. 送り仮名を付与する (decode)

UD-Kundoku の解析動作の概要を、図 3 に示す。UD-Kundoku は、内部に UD-Kanbun [6] と UniDic2UD [11] と旧仮名口語 UniDic を内蔵 (ダウンロード) しており、encode と返り点の付与は UD-Kanbun が、decode のうち活用語尾の付与は UniDic2UD ＋旧仮名口語 UniDic が、それぞれ一部分担している。

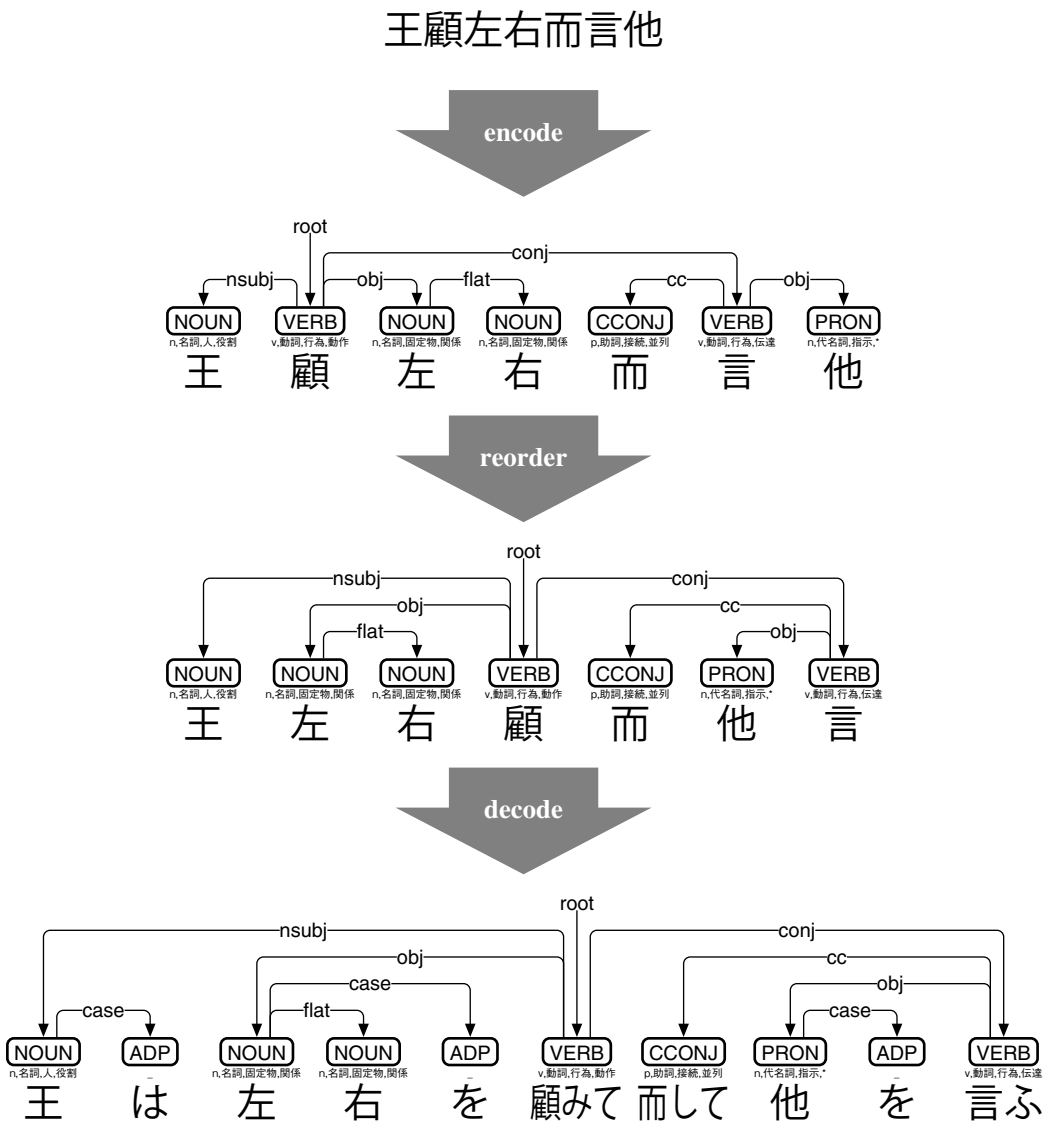

図 3: 「王顧左右而言他」に対する自動訓読の流れ

[11]安岡孝一: 形態素解析部の付け替えによる近代日本語 (旧字旧仮名) の係り受け解析, 情報処理学会研究報告, Vol.2020-CH-124 (2020 年 9 月), No.3, pp.1-8.

UD-Kundoku の簡単な使用法 (python コンソール上) を、以下に示す。

```
>>> import udkundoku
>>> lzh=udkundoku.load(Danku=False)
>>> s=lzh("王顧左右而言他")
>>> t=udkundoku.translate(s)
>>> print(t.sentence())
```

なお、オプションとして Danku=True を指定した場合は、古詩文斷句[12]で文切りを試し、それが上手くいかない場合は、UD-Kanbun 内蔵の UDPipe [13]で文切りをおこなう。詳細な使用法は、https://github.com/KoichiYasuoka/UD-Kundoku を参照されたい。

UD-Kundoku のインストールを簡便にすべく、PyPI (Python Package Index) によるパッケージ化をおこなった。python (3.6 以上) と pip と g++を搭載したシステム (Linux・Cygwin・macOS など) をインターネットに接続し、コンソールで

```
pip3 install -U udkundoku --user
```

を実行すれば、UD-Kundoku を一発でインストールできる。

これと合わせ、UD-Kundoku のデモンストレーション・ページを Google Colaboratory 上[14]で公開した。Google ID と Web ブラウザ (Edge・Safari・Chrome) があれば、こちらはインストール作業なしに UD-Kundoku を使えるので、ぜひアクセスしてみてほしい。

6 おわりに

古典中国語 Universal Dependencies にもとづく漢文自動訓読システム UD-Kundoku を、encode-reorder-decode モデルによる python3 モジュールとして実装した。encode (依存文法解析) は機械学習、reorder (返り点による語順入れ替え) と decode (送り仮名の付与) はルールベースである。

なお、本研究は、科学研究費補助金基盤研究 (B) 20H04481 『古典漢文依存文法コーパスにもとづく係り受け構造の自動抽出』の研究助成を受けている。

[12]胡韌奮, 李紳, 諸雨辰: 基於深層語言模型的古漢語知識表示及自動斷句研究, CCL2019: 18th China National Conference on Computational Linguistics (2019 年 10 月).

[13]Milan Straka and Jana Straková: Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe, Proceedings of the CoNLL 2017 Shared Task (August 2017), pp.88-99.

[14]https://colab.research.google.com/github/KoichiYasuoka/UD-Kundoku/blob/master/udkundoku.ipynb