

アルゴリズムとデータ構造III 4回目:10月23日

構文解析 CYK法の続き

授業資料 <http://ir.cs.yamanashi.ac.jp/~ysuzuki/algorithm3/index.html>

1

授業の予定(中間試験まで)

1	10/02	スタック(後置記法で書かれた式の計算)
2	10/09	チューリング機械, 文脈自由文法
3	10/16	構文解析 CYK法
4	10/23	構文解析 CYK法
5	10/30	構文解析 CYK法, アーリー法
6	11/06	構文解析, グラフ
7	11/13	グラフ(動的計画法, ダイクストラ法, DPマッチング)
8	11/20	グラフ(DPマッチング, ビームサーチ, A*アルゴリズム)
9	11/27	中間試験

2

授業の予定(中間試験以降)

10	12/04	全文検索アルゴリズム(simple search, KMP)
11	12/11	全文検索アルゴリズム(BM, Aho-Corasick)
12	12/18	全文検索アルゴリズム(Aho-Corasick), データ圧縮
13	01/08	暗号(黄金虫, 踊る人形) 符号化(モールス信号, Zipfの法則, ハフマン符号)テキスト圧縮
14	01/15	テキスト圧縮(zip), 音声圧縮(ADPCM, MP3, CELP), 画像圧縮(JPEG)
15	01/29	期末試験

3

本日のメニュー

- CYK法の続き
 - CYKアルゴリズム
 - 解析例(急いで走る一郎を見た)
 - 練習問題(I eat pizza with Nana.)

4

構文解析 CYK法

- 先週勉強した文脈自由文法により, 文から自動的に構文木を生成する。

5

構文解析とは(Wikipediaより)

- ある文章の文法的な関係を説明すること(parse)。
計算機科学の世界では、構文解析は**字句解析(Lexical Analysis)**とともに、おもに**プログラミング言語などの形式言語**の解析に使用される。また、**自然言語処理**に応用されることもある。
- コンパイラにおいて構文解析を行う機構を**構文解析器(Parser)**と呼ぶ。
- 構文解析は入力テキストを通常、**木構造**のデータ構造に変換し、その後の処理に適した形にする。**字句解析**によって入力文字列から字句を取り出し、それらを構文解析器の入力として、**構文木**や**抽象構文木**のようなデータ構造を生成する。

6

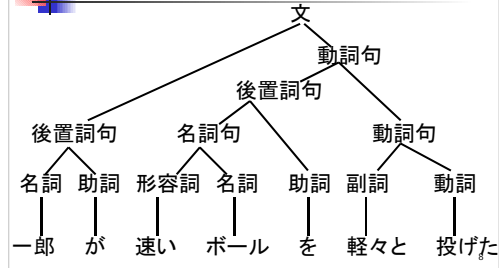
構文解析

- 入力文(記号列)が与えられたとき, 文法によってその文を解析し, その構造を明らかにする
- 代表的な構文解析アルゴリズム
 - CYK法
 - チャート法
 - アーク法
 - LR法

7

構文木

(一郎が速いボールを軽々と投げた)



CYK (Cocke-Younger-Kasami) 法

- ボトムアップアルゴリズム
- 扱える文法
 - チョムスキーの標準形
 - $A \rightarrow BC$
 - $A \rightarrow a$
- CYK表
 - 構文解析の途中経過を保持するための表

9

CYKアルゴリズム

- チョムスキーの標準形の文脈自由文法を対象とした構文解析法
 - チョムスキーの標準形
 - $A \rightarrow BC$ ($A, B, C \in V_n$)
 - $A \rightarrow a$ ($A \in V_n, a \in V_t$)
- $X \rightarrow aB$ はチョムスキーの標準形ではないが
 $X \rightarrow AB, A \rightarrow a$ に分解できる
 $X \rightarrow ABC$ はチョムスキーの標準形ではないが
 $X \rightarrow AY, Y \rightarrow BC$ に分解できる

10

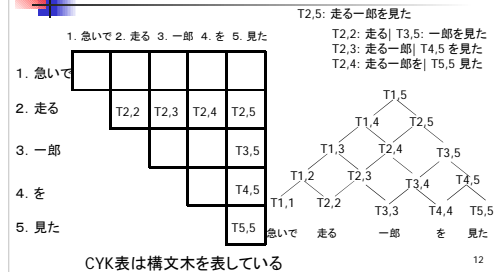
チョムスキーの標準形の例 「急いで走る一郎を見る」

A→BC型 A→a型

- | | |
|-----------------------------|---------------------------|
| (1) $s \rightarrow pp\ v$ | (9) $adv \rightarrow 急いで$ |
| (2) $s \rightarrow adv\ vp$ | (10) $n \rightarrow 一郎$ |
| (3) $vp \rightarrow pp\ v$ | (11) $p \rightarrow を$ |
| (4) $vp \rightarrow adv\ v$ | (12) $v \rightarrow 走る$ |
| (5) $np \rightarrow vp\ n$ | (13) $v \rightarrow 見る$ |
| (6) $np \rightarrow v\ n$ | |
| (7) $pp \rightarrow np\ p$ | |
| (8) $pp \rightarrow n\ p$ | |

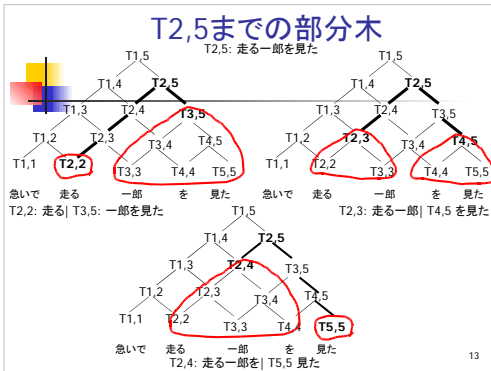
11

CYK構文解析の概要



CYK表は構文木を表している

12



CYKアルゴリズム

- $A \rightarrow a$ の生成規則を用いて、主対角線上の要素を計算
for $i = 1$ to N
 $T_{i,i} = \{A \mid A \rightarrow w_i\}$
- $A \rightarrow BC$ の生成規則を用いて、2番目以降の対角線上の要素を計算
for $n = 1$ to $N-1$
for $i = 1$ to $N-n$
 $T_{i,i+n} = \bigcup_{j=1}^n \{A \mid A \rightarrow BC, B \in T_{i,i+j-1}, C \in T_{i+j,i+n}\}$
- $S \in T_{1,N}$ であれば、 $w_1 \dots w_N$ は開始記号Sから導出可能

CYK構文解析表(完成)

	1. 急いで	2. 走る	3. 一郎	4. を	5. 見た
1. 急いで	adv→急いで	vp→adv v	np→vp n	pp→np p	vp→pp v s→pp v s→adv vp
2. 走る		v→走る	np→v n	pp→np p	vp→pp v s→pp v
3. 一郎	(1) s→pp v (2) s→adv vp (3) vp→pp v		n→一郎	pp→n p	vp→pp v s→pp v
4. を	(4) vp→adv v (5) np→vp n (6) np→v n (7) pp→np p (8) pp→n p			p→を	
5. 見た	(9) adv→急いで (10) n→一郎 (11) p→を (12) v→走る (13) v→見る				v→見た

CYK構文解析表(1/5)

	1. 急いで	2. 走る	3. 一郎	4. を	5. 見た
1. 急いで	adv→急いで				
2. 走る		v→走る			
3. 一郎	(1) s→pp v (2) s→adv vp (3) vp→pp v		n→一郎		
4. を	(4) vp→adv v (5) np→vp n (6) np→v n (7) pp→np p (8) pp→n p			p→を	
5. 見た	(9) adv→急いで (10) n→一郎 (11) p→を (12) v→走る (13) v→見る				v→見た

CYK構文解析表(2/5)

	1. 急いで	2. 走る	3. 一郎	4. を	5. 見た
1. 急いで	adv→急いで	vp→adv v			
2. 走る		v→走る	np→v n		
3. 一郎	(1) s→pp v (2) s→adv vp (3) vp→pp v		n→一郎	pp→n p	
4. を	(4) vp→adv v (5) np→vp n (6) np→v n (7) pp→np p (8) pp→n p			p→を	
5. 見た	(9) adv→急いで (10) n→一郎 (11) p→を (12) v→走る (13) v→見る				v→見た

CYK構文解析表(3/5)

	1. 急いで	2. 走る	3. 一郎	4. を	5. 見た
1. 急いで	adv→急いで	vp→adv v	np→vp n		
2. 走る		v→走る	np→v n	pp→np p	
3. 一郎	(1) s→pp v (2) s→adv vp (3) vp→pp v		n→一郎	pp→n p	vp→pp v s→pp v
4. を	(4) vp→adv v (5) np→vp n (6) np→v n (7) pp→np p (8) pp→n p			p→を	
5. 見た	(9) adv→急いで (10) n→一郎 (11) p→を (12) v→走る (13) v→見る				v→見た

CYK構文解析表(4/5)

1. 急いで 2. 走る 3. 一郎 4. を 5. 見た

1. 急いで	adv→急いで	vp→adv v	np→vp n	pp→np p	
2. 走る		v→走る	np→v n	pp→np p	vp→pp v s→pp v
3. 一郎			n→一郎	pp→n p	vp→pp v s→pp v
4. を				p→を	
5. 見た					v→見た

- (1) s→pp v
- (2) s→adv vp
- (3) vp→pp v
- (4) vp→adv v
- (5) np→vp n
- (6) np→v n
- (7) pp→np p
- (8) pp→n p
- (9) adv→急いで
- (10) n→一郎
- (11) p→を
- (12) v→走る
- (13) v→見た

CYK構文解析表(5/5)

1. 急いで 2. 走る 3. 一郎 4. を 5. 見た

1. 急いで	adv→急いで	vp→adv v	np→vp n	pp→np p	vp→pp v s→pp v s→adv vp
2. 走る		v→走る	np→v n	pp→np p	vp→pp v s→pp v
3. 一郎			n→一郎	pp→n p	vp→pp v s→pp v
4. を				p→を	
5. 見た					v→見た

- (1) s→pp v
- (2) s→adv vp
- (3) vp→pp v
- (4) vp→adv v
- (5) np→vp n
- (6) np→v n
- (7) pp→np p
- (8) pp→n p
- (9) adv→急いで
- (10) n→一郎
- (11) p→を
- (12) v→走る
- (13) v→見た

CYK構文解析表(完成！)

1. 急いで 2. 走る 3. 一郎 4. を 5. 見た

1. 急いで	adv→急いで	vp→adv v	np→vp n	pp→np p	vp→pp v s→pp v s→adv vp
2. 走る		v→走る	np→v n	pp→np p	vp→pp v s→pp v
3. 一郎			n→一郎	pp→n p	vp→pp v s→pp v
4. を				p→を	
5. 見た					v→見た

CYK構文解析表 → 構文木 (s→pp v の構文木)

1. 急いで 2. 走る 3. 一郎 4. を 5. 見た

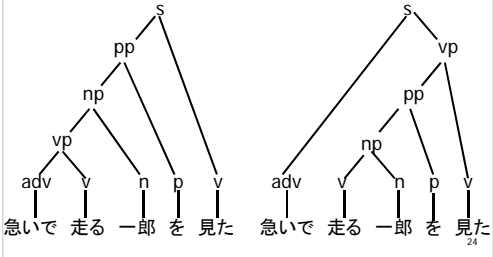
1. 急いで	adv→急いで	vp→adv v	np→vp n	pp→np p	s→pp v
2. 走る		v→走る	np→v n	pp→np p	
3. 一郎			n→一郎	pp→n p	
4. を				p→を	
5. 見た					v→見た

CYK構文解析表 → 構文木 (s→adv vp の構文木)

1. 急いで 2. 走る 3. 一郎 4. を 5. 見た

1. 急いで	adv→急いで				s→adv vp
2. 走る		v→走る	np→v n	pp→np p	vp→pp v
3. 一郎			n→一郎	pp→n p	
4. を				p→を	
5. 見た					v→見た

文脈自由文法に基づく構文木



練習問題1

- CYK法を使って“I eat pizza with Nana”の構文解析結果を作成しなさい。

	(1)	S	→	N V
	(2)	S	→	S PP
	(3)	S	→	V N
	(4)	V	→	V N
	(5)	PP	→	P N
	(6)	N	→	N PP
A→BC				

	(7)	N	→	I
	(8)	N	→	Nana
	(9)	N	→	pizza
	(10)	V	→	eat
	(11)	P	→	with
A→a (辞書規則)				

25

CYK法で構文解析

I eat pizza with Nana.

	1. I	2. eat	3. pizza	4. with	5. Nana
1. I	N → I				
2. eat		V → eat			
3. pizza			N → pizza		
4. with				P → with	
5. Nana					N → Nana

- S → N V
- S → S PP
- S → V N
- V → V N
- PP → P N
- N → N PP
- N → I
- N → Nana
- N → pizza
- V → eat
- P → with

26

CYK法で構文解析

I eat pizza with Nana.

	1. I	2. eat	3. pizza	4. with	5. Nana
1. I	N → I	S → N V			
2. eat		V → eat	S → V N V → V N		
3. pizza			N → pizza		
4. with				P → with	PP → P N
5. Nana					N → Nana

- S → N V
- S → S PP
- S → V N
- V → V N
- PP → P N
- N → N PP
- N → I
- N → Nana
- N → pizza
- V → eat
- P → with

27

CYK法で構文解析

I eat pizza with Nana.

	1. I	2. eat	3. pizza	4. with	5. Nana
1. I	N → I	S → N V	S → N V		
2. eat		V → eat	S → V N V → V N		
3. pizza			N → pizza		N → N PP
4. with				P → with	PP → P N
5. Nana					N → Nana

- S → N V
- S → S PP
- S → V N
- V → V N
- PP → P N
- N → N PP
- N → I
- N → Nana
- N → pizza
- V → eat
- P → with

28

CYK法で構文解析

I eat pizza with Nana.

	1. I	2. eat	3. pizza	4. with	5. Nana
1. I	N → I	S → N V	S → N V		
2. eat		V → eat	S → V N V → V N		S → S PP S → V N V → V N
3. pizza			N → pizza		N → N PP
4. with				P → with	PP → P N
5. Nana					N → Nana

- S → N V
- S → S PP
- S → V N
- V → V N
- PP → P N
- N → N PP
- N → I
- N → Nana
- N → pizza
- V → eat
- P → with

29

CYK法で構文解析

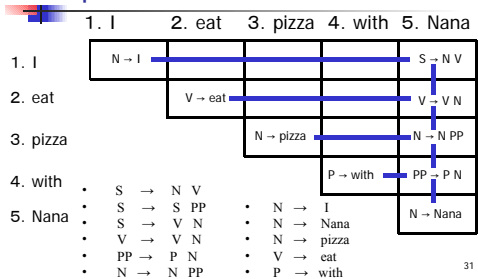
I eat pizza with Nana.

	1. I	2. eat	3. pizza	4. with	5. Nana
1. I	N → I	S → N V	S → N V		S → N V S → S PP
2. eat		V → eat	S → V N V → V N		S → S PP S → V N V → V N
3. pizza			N → pizza		N → N PP
4. with				P → with	PP → P N
5. Nana					N → Nana

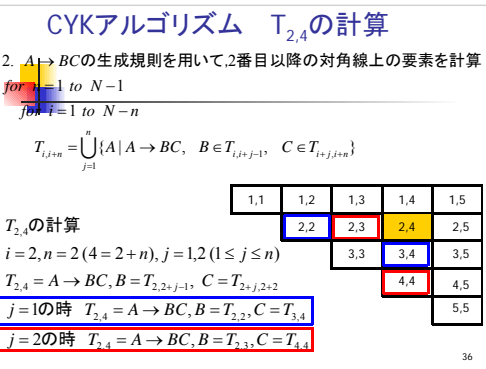
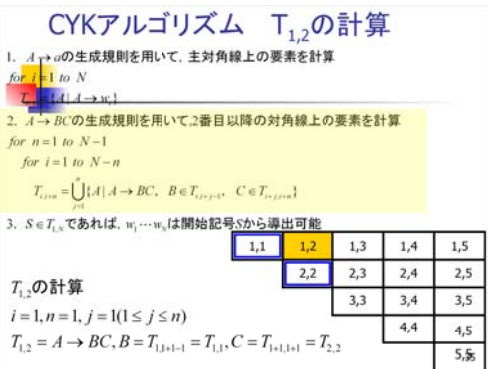
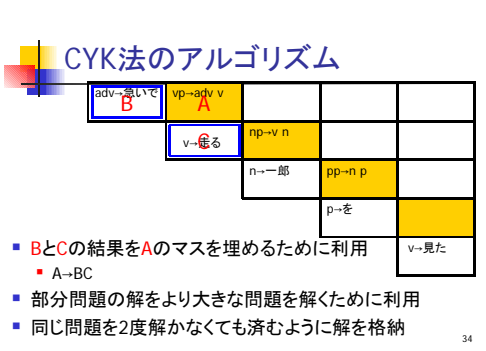
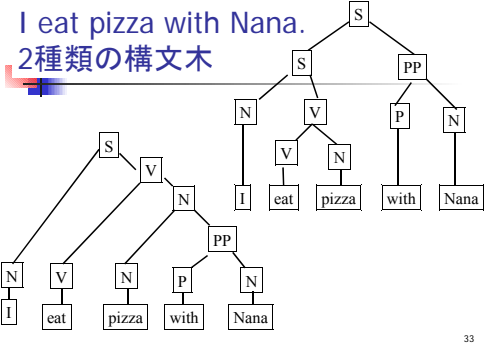
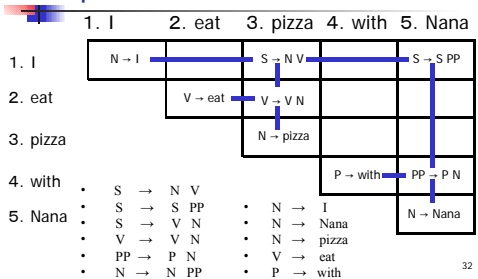
- S → N V
- S → S PP
- S → V N
- V → V N
- PP → P N
- N → N PP
- N → I
- N → Nana
- N → pizza
- V → eat
- P → with

30

CYK法で構文解析 S → N V の構文木 I eat pizza with Nana.

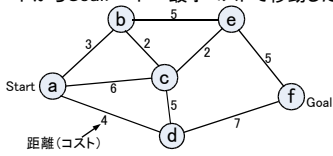


CYK法で構文解析 S → S PP の構文木 I eat pizza with Nana.



ダイクストラ法(最短経路問題用アルゴリズム)

- StartノードからGoalノードへ最小コストで移動したい



- a-e, a-dなどの最短距離をa-fの最短距離を見つけるために利用
- 部分問題の解をより大きな問題を解くために利用
- 同じ問題を2度解かなくても済むように解を格納

37

動的計画法(Dynamic Programming)

- 部分問題の解をより大きな問題を解くために利用
- 同じ問題を2度解かなくても済むように解を格納

アルゴリズムの例

- CYK法(構文解析)
- ダイクストラ法(最短経路問題)
- DPマッチング(パターンマッチング DNAの解析にも利用)
- DPを使った解法(ナップサック問題)
- ビタビアルゴリズム(音声認識など)

38

構文解析アルゴリズム

ボトムアップアルゴリズム

- 戦略
 - 単語列から出発
 - Sを導出 → 解析終了
- 代表的なアルゴリズム
 - CYK法
 - LR法

トップダウンアルゴリズム

- 戦略
 - Sから出発
 - 目的の単語列を導出 → 解析終了
- 代表的なアルゴリズム
 - アークリー法(Earley parser)
 - LL法

9