

## アルゴリズムとデータ構造Ⅲ 4回目:11月04日

構文解析 CYK法の続き

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

## 授業の予定(中間試験まで)

1	10/07	スタック(後置記法で書かれた式の計算)
2	10/14	チューリング機械, 文脈自由文法
3	10/21	構文解析 CYK法
4	11/04	構文解析 CYK法
5	11/11	構文解析(チャート法), グラフ(ダイクストラ法)
6	11/18	構文解析(チャート法), グラフ(ダイクストラ法, DPマッチング)
7	11/19 4時限 B2-41	グラフ(DPマッチング, A*アルゴリズム)
8	11/25	グラフ(A*アルゴリズム), 前半のまとめ
9	12/02	中間試験

## 授業の予定(中間試験以降)

10	12/09	全文検索アルゴリズム(simple search, KMP)
11	12/16	全文検索アルゴリズム(BM, Aho-Corasick)
12	01/06	全文検索アルゴリズム(Aho-Corasick), データ圧縮
13	01/13	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	01/20	テキスト圧縮(zip), 音声圧縮(ADPCM, MP3, CELP), 画像圧縮(JPEG)
15	02/03	期末試験

## 10月28日の補講日

木3時限	11/11	11/18	11/25	12/02	12/09	12/16
木4時限	11/11	11/18	11/25	12/02	12/09	12/16
金1時限	11/12	11/19	11/26	12/03	12/10	12/17
金4時限	11/12	11/19 補講日	11/26	12/03	12/10	12/17

教室: B2-41

## 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.)

## 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.

## 構文解析とは(Wikipediaより)

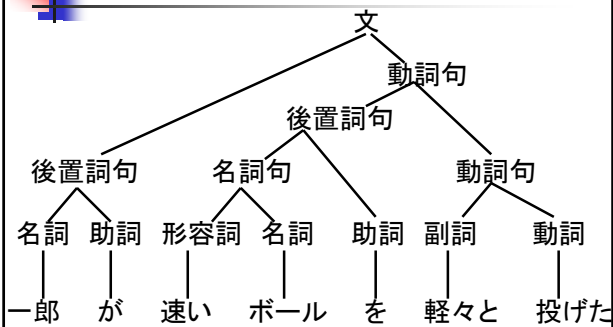
- ある文章の文法的な関係を説明すること(parse)。計算機科学の世界では、構文解析は字句解析(Lexical Analysis)とともに、おもにプログラミング言語などの形式言語の解析に使用される。また、自然言語処理に応用されることもある。
- コンパイラにおいて構文解析を行う機構を構文解析器(Parser)と呼ぶ。
- 構文解析は入力テキストを通常、木構造のデータ構造に変換し、その後の処理に適した形にする。字句解析によって入力文字列から字句を取り出し、それらを構文解析器の入力として、構文木や抽象構文木のようなデータ構造を生成する。

## 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

## 構文木

(一郎が速いボールを軽々と投げた)



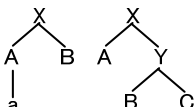
## CYK (Cocke-Younger-Kasami) 法

- 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
    - $A \rightarrow BC$
    - $A \rightarrow a$
  - $A \rightarrow BCD$ ,  $A \rightarrow aB$ などは直接は扱えない
- CYK表
  - 構文解析の途中経過を保持するための表

## CYKアルゴリズム

- チョムスキーの標準形で表される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



## 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

- |   |  |
|---|--|
| <ul style="list-style-type: none"> <li>(1) <math>s \rightarrow pp\ v</math></li> <li>(2) <math>s \rightarrow adv\ vp</math></li> <li>(3) <math>vp \rightarrow pp\ v</math></li> <li>(4) <math>vp \rightarrow adv\ v</math></li> <li>(5) <math>np \rightarrow vp\ n</math></li> <li>(6) <math>np \rightarrow v\ n</math></li> <li>(7) <math>pp \rightarrow np\ p</math></li> <li>(8) <math>pp \rightarrow n\ p</math></li> </ul> | <ul style="list-style-type: none"> <li>(9) <math>adv \rightarrow</math>急いで</li> <li>(10) <math>n \rightarrow</math>一郎</li> <li>(11) <math>p \rightarrow</math>を</li> <li>(12) <math>v \rightarrow</math>走る</li> <li>(13) <math>v \rightarrow</math>見る</li> </ul> |
|---|--|

## CYK構文解析の概要

T2,5: 走る一郎を見た

3種類の部分木 (次ページ)

T2,2: 走る | T3,5: 一郎を見た  
T2,3: 走る一郎 | T4,5 を見た  
T2,4: 走る一郎を | T5,5 見た

1. 急いで	T1,1	T1,2	T1,3	T1,4	T1,5
2. 走る		T2,2	T2,3	T2,4	T2,5
3. 一郎			T3,3	T3,4	T3,5
4. を				T4,4	T4,5
5. 見た					T5,5

急いで 走る 一郎 を 見た

CYK表は構文木を表している

## T2,5までの部分木 (3種類)

急いで 走る 一郎 を 見た  
T2,2: 走る | T3,5: 一郎を見た  
T2,3: 走る一郎 | T4,5 を見た  
T2,4: 走る一郎を | T5,5 見た

## CYKアルゴリズム

1.  $A \rightarrow a$ の生成規則を用いて, 主対角線上の要素を計算  
for  $i=1$  to  $N$   
 $T_{i,i} = \{A \mid A \rightarrow w_i\}$

2.  $A \rightarrow BC$ 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  $N-n$   
 $T_{i,i+n} = \bigcup_{j=1}^n \{A \mid A \rightarrow BC, B \in T_{i,i+j-1}, C \in T_{i+j,i+n}\}$

3.  $S \in T_{1,N}$ 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	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

- A→BC型
  - (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
- A→a型
  - (9) adv→急いで
  - (10) n→一郎
  - (11) p→を
  - (12) v→走る
  - (13) v→見る

## CYK構文解析表(1/5) A→a型

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

- A→BC型
  - (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
- A→a型
  - (9) adv→急いで
  - (10) n→一郎
  - (11) p→を
  - (12) v→走る
  - (13) v→見る

## CYK構文解析表(2/5)

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

- A→BC型
  - (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
- A→a型
  - (9) adv→急いで
  - (10) n→一郎
  - (11) p→を
  - (12) v→走る
  - (13) v→見る

### 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	<ul style="list-style-type: none"> <li>(1) s→pp v</li> <li>(2) s→adv vp</li> </ul>		n→一郎	pp→n p	vp→pp v s→pp v
4. を	<ul style="list-style-type: none"> <li>(3) vp→pp v</li> <li>(4) vp→adv v</li> <li>(5) np→vp n</li> </ul>			p→を	
5. 見た	<ul style="list-style-type: none"> <li>(6) np→v n</li> <li>(7) pp→np p</li> <li>(8) pp→n p</li> </ul>	<ul style="list-style-type: none"> <li>(9) adv→急いで</li> <li>(10) n→一郎</li> <li>(11) p→を</li> <li>(12) v→走る</li> <li>(13) v→見る</li> </ul>			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	<ul style="list-style-type: none"> <li>(1) s→pp v</li> <li>(2) s→adv vp</li> </ul>		n→一郎	pp→n p	vp→pp v s→pp v
4. を	<ul style="list-style-type: none"> <li>(3) vp→pp v</li> <li>(4) vp→adv v</li> <li>(5) np→vp n</li> </ul>			p→を	
5. 見た	<ul style="list-style-type: none"> <li>(6) np→v n</li> <li>(7) pp→np p</li> <li>(8) pp→n p</li> </ul>	<ul style="list-style-type: none"> <li>(9) adv→急いで</li> <li>(10) n→一郎</li> <li>(11) p→を</li> <li>(12) v→走る</li> <li>(13) v→見る</li> </ul>			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	<ul style="list-style-type: none"> <li>(1) s→pp v</li> <li>(2) s→adv vp</li> </ul>		n→一郎	pp→n p	vp→pp v s→pp v
4. を	<ul style="list-style-type: none"> <li>(3) vp→pp v</li> <li>(4) vp→adv v</li> <li>(5) np→vp n</li> </ul>			p→を	
5. 見た	<ul style="list-style-type: none"> <li>(6) np→v n</li> <li>(7) pp→np p</li> <li>(8) pp→n p</li> </ul>	<ul style="list-style-type: none"> <li>(9) adv→急いで</li> <li>(10) n→一郎</li> <li>(11) p→を</li> <li>(12) v→走る</li> <li>(13) v→見る</li> </ul>			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

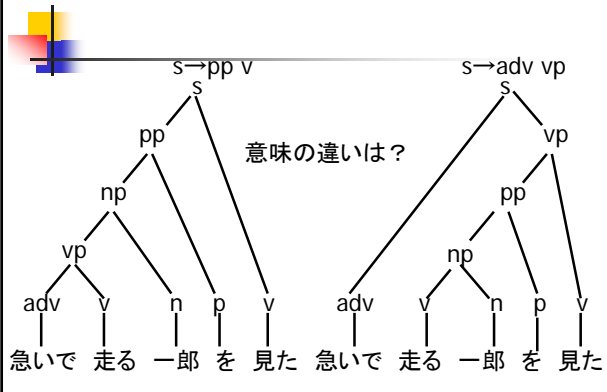
### CYK構文解析表 → 構文木 (s→pp v の構文木)

	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	vp→pp v
3. 一郎			n→一郎	pp→n p	vp→pp v
4. を				p→を	vp→pp v
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	vp→adv v	np→vp n	pp→np p	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	vp→pp v
4. を				p→を	vp→pp v
5. 見た					v→見た

### 文脈自由文法に基づく構文木(2種類)



### 質問: CYK法ではなぜチョムスキーの標準形しか扱えない?

### 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
  - (7) N → I
  - (8) N → Nana
  - (9) N → pizza
  - (10) V → eat
  - (11) P → with
- A→BC型 (1-6)  
A→a型 (辞書規則) (7-11)

### 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

A→BC型 (1-6)      A→a型 (7-11)

### 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

A→BC型 (1-6)      A→a型 (7-11)

### 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

A→BC型 (1-6)      A→a型 (7-11)

### 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	A→BC型			P → with	PP → P N
5. Nana	A→a型				N → Nana

(1) S → N V (2) S → S PP (3) S → V N (4) V → V N (5) PP → P N (6) N → N PP  
 (7) N → I (8) N → Nana (9) N → pizza (10) V → eat (11) P → with

### 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	A→BC型			P → with	PP → P N
5. Nana	A→a型				N → Nana

(1) S → N V (2) S → S PP (3) S → V N (4) V → V N (5) PP → P N (6) N → N PP  
 (7) N → I (8) N → Nana (9) N → pizza (10) V → eat (11) P → with

### CYK法で構文解析 S → N V の構文木

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			V → V N
3. pizza			N → pizza		N → N PP
4. with	A→BC型			P → with	PP → P N
5. Nana	A→a型				N → Nana

(1) S → N V (2) S → S PP (3) S → V N (4) V → V N (5) PP → P N (6) N → N PP  
 (7) N → I (8) N → Nana (9) N → pizza (10) V → eat (11) P → with

### CYK法で構文解析 S → S PP の構文木

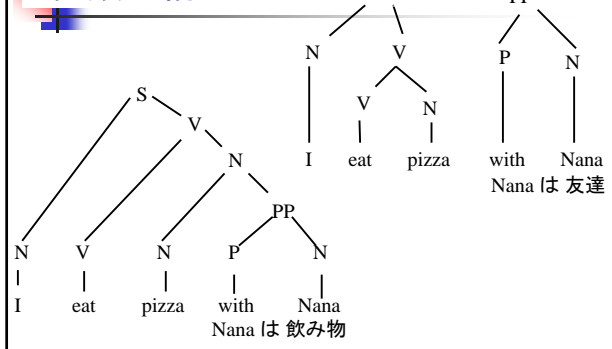
I eat pizza with Nana.

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

(1) S → N V (2) S → S PP (3) S → V N (4) V → V N (5) PP → P N (6) N → N PP  
 (7) N → I (8) N → Nana (9) N → pizza (10) V → eat (11) P → with

### I eat pizza with Nana.

2種類の構文木



### CYK法のアルゴリズム

adv → 急いで	vp → adv v			
B	A	np → v n		
	v → 見る	n → 一郎	pp → n p	
		p → を		
				v → 見た

- BとCの結果をAのマスを埋めるために利用
  - A→BC
- 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

## CYKアルゴリズム $T_{1,2}$ の計算

1.  $A \rightarrow a$ の生成規則を用いて、主対角線上の要素を計算

for  $i = 1$  to  $N$

$$T_{i,i} = \{A \mid A \rightarrow w_i\}$$

2.  $A \rightarrow BC$ 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  $N-n$

$$T_{i,i+n} = \bigcup_{j=1}^n \{A \mid A \rightarrow BC, B \in T_{i,i+j-1}, C \in T_{i+j,j+n}\}$$

3.  $S \in T_{1,N}$ であれば、 $w_1 \dots w_N$ は開始記号 $S$ から導出可能

1,1	1,2	1,3	1,4	1,5
2,2	2,3	2,4	2,5	
	3,3	3,4	3,5	
		4,4	4,5	
			5,5	

$T_{1,2}$ の計算

$i = 1, n = 1, j = 1 (1 \leq j \leq n)$

$$T_{1,2} = A \rightarrow BC, B = T_{1,1+1-1} = T_{1,1}, C = T_{1+1,1+1} = T_{2,2}$$

## CYKアルゴリズム $T_{2,4}$ の計算

2.  $A \rightarrow BC$ 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  $N-n$

$$T_{i,i+n} = \bigcup_{j=1}^n \{A \mid A \rightarrow BC, B \in T_{i,i+j-1}, C \in T_{i+j,j+n}\}$$

$T_{2,4}$ の計算

$i = 2, n = 2 (4 = 2 + n), j = 1, 2 (1 \leq j \leq n)$

$$T_{2,4} = A \rightarrow BC, B = T_{2,2+j-1}, C = T_{2+j,2+2}$$

$$j = 1 \text{ の時 } T_{2,4} = A \rightarrow BC, B = T_{2,2}, C = T_{3,4}$$

$$j = 2 \text{ の時 } T_{2,4} = A \rightarrow BC, B = T_{2,3}, C = T_{4,4}$$

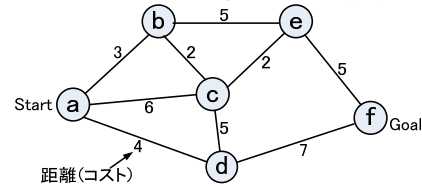
1,1	1,2	1,3	1,4	1,5
	2,2	2,3	2,4	2,5
		3,3	3,4	3,5
			4,4	4,5
				5,5

## 本日のまとめ

- 構文解析 (先週の復習)
  - 構文木 (急いで走る一郎を見た)
  - 代表的な構文解析アルゴリズム
- CYK法
  - CYKアルゴリズムの説明
  - 解析例 (急いで走る一郎を見た)
  - Step by step で解析

## 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

## 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 (ナップザック問題)
- ビタビアルゴリズム (音声認識など)

## 構文解析アルゴリズム

ここまで

### 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  $\rightarrow$  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  $\rightarrow$  解析終了
- 代表的なアルゴリズム
  - アーリー法 (Earley parser)
  - LL法