第1回進捗報告

4年　上田亮

今週やったこと

- いくつか論文を読んでみました（チラ見を含む）
  a. 有限状態トランスデューサに関して
     i. **Mohri. 2009. Weighted Automata Algorithms. Handbook of weighted automata. Springer.**
     ii. Sproat et al. 1996. A Stochastic Finite-State Word-Segmentation Algorithm for Chinese. Computational Linguistics.
     iii. Kaplan and Kay. 1994. Regular Models of Phonological Rule Systems. Computational Linguistics.
  b. 知識グラフに関して
     i. **Friedman, Broeck. 2020. Symbolic Querying of Vector Spaces: Probabilistic Databases Meets Relational Embeddings. ArXiv.**

→「有限状態トランスデューサによる分かち書き」に挑戦してみることに

今週やったこと

- 実装を開始
  - https://github.com/wedddy0707/enshu3_1
  - とりあえず Python3 で動かせるようにしたい
  - 重み付き有限状態トランスデューサの Classを実装中
    - composition, closure などのメソッドを作った
  - 入力文字列トランスデューサ、辞書トランスデューサの簡単バージョンを実装

トピック

- 分かち書きとは
- 有限状態トランスデューサーとは
- 重み付き有限状態トランスデューサとは
- トランスデューサの演算
- トランスデューサーによる分かち書き

トピック

- 分かち書きとは
- 有限状態トランスデューサーとは
- 重み付き有限状態トランスデューサとは
- トランスデューサの演算
- トランスデューサーによる分かち書き

分かち書きとは

- 単語ごとの区切り

にほんご　の　ぶんしょう　に　おいて　ご　の　くぎり　に　くうはく　を　はさん　で　きじゅつする　こと

- 分節ごとの区切り

にほんごの　ぶんしょうに　おいて　ごの　くぎりに　くうはくを　はさんで　きじゅつする　こと

Wikipedia「分かち書き」

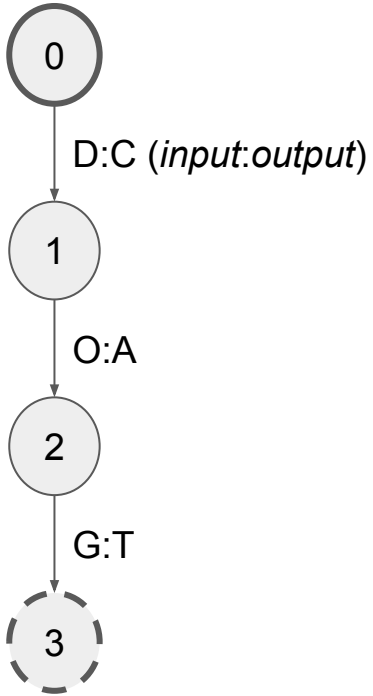
トピック

- 分かち書きとは
- 有限状態トランスデューサーとは
- 重み付き有限状態トランスデューサとは
- トランスデューサの演算
- トランスデューサーによる分かち書き

有限状態トランスデューサとは

- 有限状態オートマトンに、出力を備えたもの
- 入力記号を受けると同時に、出力記号を出す

例えば、右のトランスデューサは、“DOG”という文字列を受け取ると、”CAT”を出力して受理状態になる



トピック

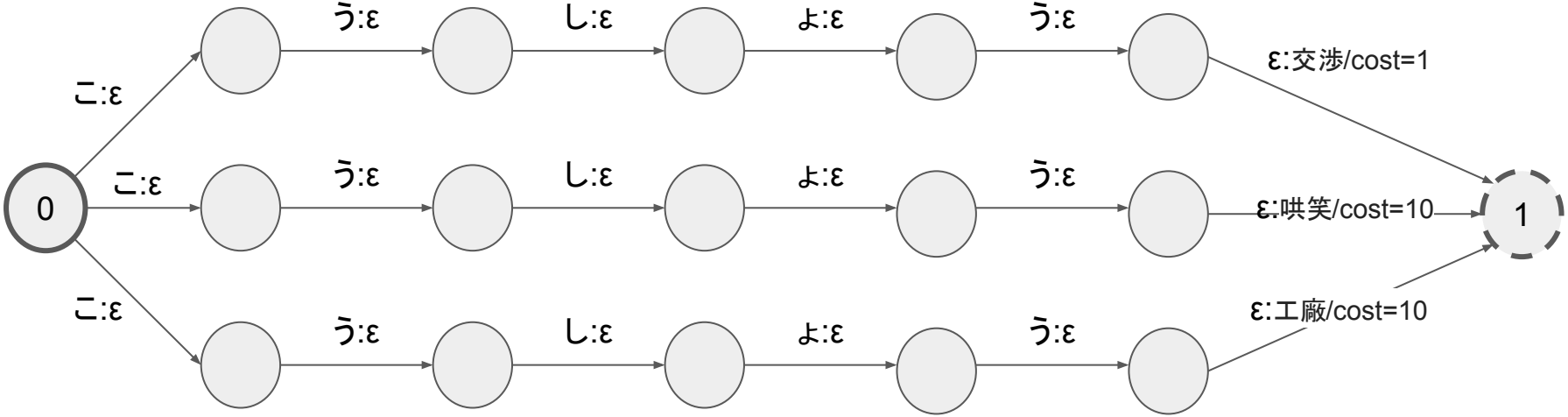
- 分かち書きとは
- 有限状態トランスデューサーとは
- 重み付き有限状態トランスデューサとは
- トランスデューサの演算
- トランスデューサーによる分かち書き

重み付き有限状態トランスデューサとは

- 有限状態トランスデューサに「辺の重み」を装備

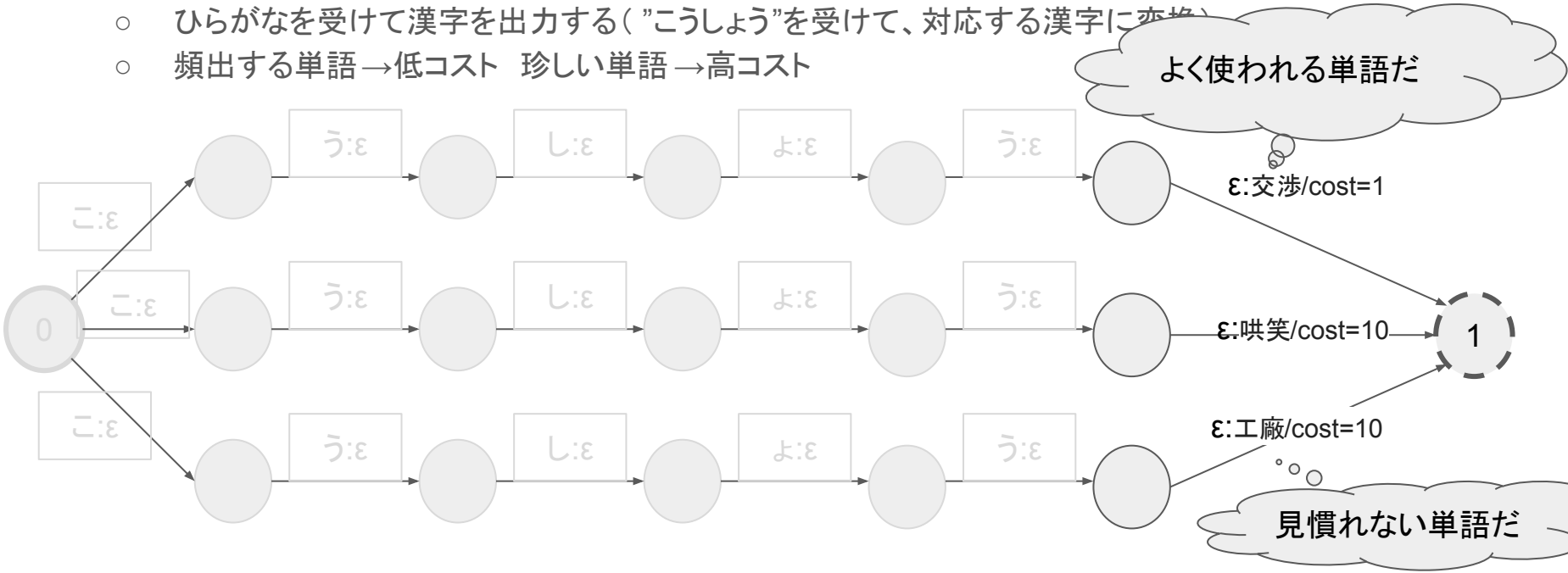
重み付き有限状態トランスデューサとは：例

- 有限状態トランスデューサに「辺の重み」を装備
- 例えば、下のε遷移付き非決定トランスデューサ
  - ひらがなを受けて漢字を出力する（ ”こうしょう”を受けて、対応する漢字に変換）
  - 頻出する単語→低コスト　珍しい単語→高コスト



重み付き有限状態トランスデューサとは：例

- 有限状態トランスデューサに「辺の重み」を装備
- 例えば、下のε遷移付き非決定トランスデューサ
  - ひらがなを受けて漢字を出力する（ ”こうしょう”を受けて、対応する漢字に変換）
  - 頻出する単語→低コスト　珍しい単語→高コスト



重み付き有限状態トランスデューサとは：数学的定義

数学的に定義するなら…

A weighted transducer T over a semiring S is an 8-tuple $T = (\sum, \triangle, Q, I, F, E, \lambda, \rho)$ where

- $\sum$: finite input alphabet
- $\triangle$: finite output alphabet
- Q : a finite set of states
- $I \subseteq Q$: the set of initial states
- $F \subseteq Q$: the set of final states
- E : a finite multi-set of transitions. $E \subseteq Q \times (\sum \cup \{\varepsilon\}) \times (\triangle \cup \{\varepsilon\}) \times S \times Q$
- $\lambda : I \rightarrow S$: an initial weight function
- $\rho : F \rightarrow S$: a final weight function

トピック

- 分かち書きとは
- 有限状態トランスデューサーとは
- 重み付き有限状態トランスデューサとは
- トランスデューサの演算
- トランスデューサーによる分かち書き

トランスデューサの演算

T, U を、それぞれ正規表現 *x, y* を受理するトランスデューサとします

- T⊕U : 正規表現 *x* | *y* を受理するトランスデューサ
- T⊗U : 正規表現 *xy* を受理するトランスデューサ
- T◦U　　: 合成。Tの出力を、Uが逐次入力として受け取る
- T*　　: Kleene Closure. 正規表現 x* = ε | x | xx | xxx | … を受理する

トランスデューサの演算：合成

- 合成 T◦U:Tの出力を、Uが逐次入力として受け取る

例えば、右のトランスデューサ

T:

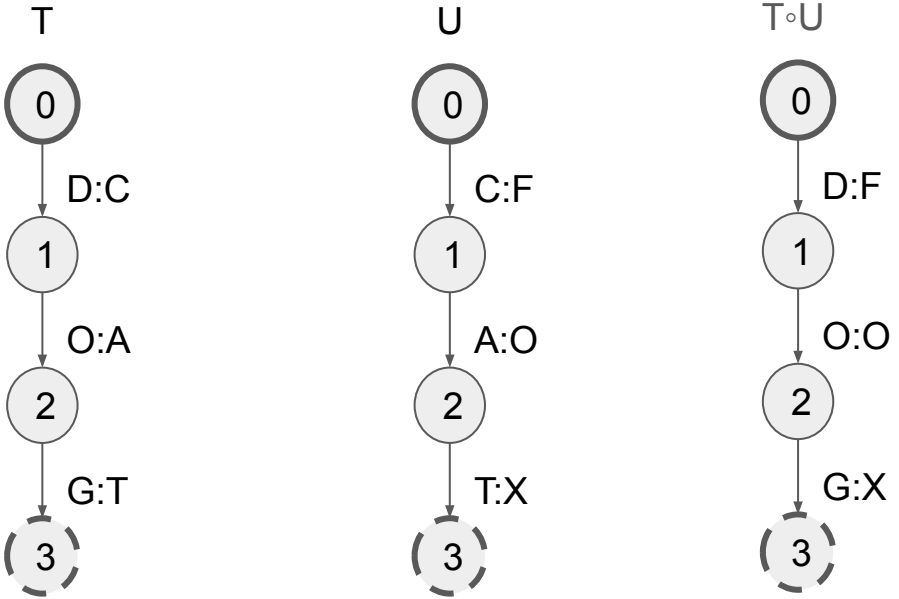
“DOG”を受け “CAT” を出力

U:

“CAT”を受け “FOX” を出力

T◦U:

“DOG”を受け “FOX” を出力



トピック

- 分かち書きとは
- 有限状態トランスデューサーとは
- 重み付き有限状態トランスデューサとは
- トランスデューサの演算
- トランスデューサーによる分かち書き

有限状態トランスデューサによる分かち書き

- 入力文字列を分かち書きしたものに変換したい

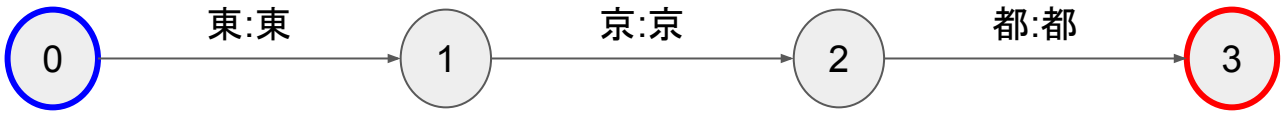
  ↓

- 入力文字列　と　辞書　を　それぞれトランスデューサで表現
- それらの合成によって、分かち書きを表現したい

有限状態トランスデューサによる分かち書き

入力文字列トランスデューサ I:

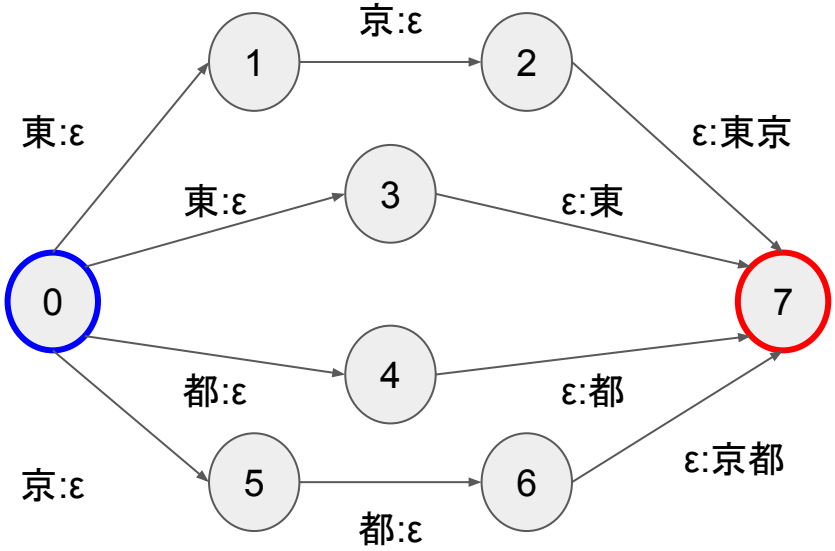
- 入力アルファベット : {“東”, “京”, “都”}
- 出力アルファベット : {“東”, “京”, “都”}



有限状態トランスデューサによる分かち書き

辞書トランスデューサ *D*

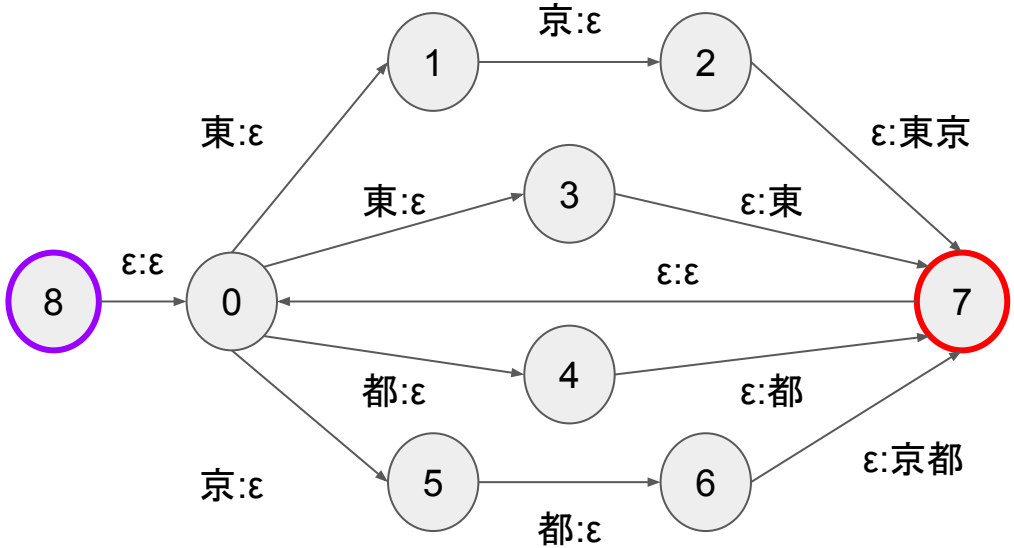
- 入力アルファベット　　: {“東”, “京”, “都”}
- 出力アルファベット(語彙)　: {“東”, “京都”, “東京”, “都”}



有限状態トランスデューサによる分かち書き

辞書トランスデューサ *D* のクロージャ *D**

- 入力アルファベット　　: {“東”, “京”, “都”}
- 出力アルファベット(語彙)　: {“東”, “京都”, “東京”, “都”}

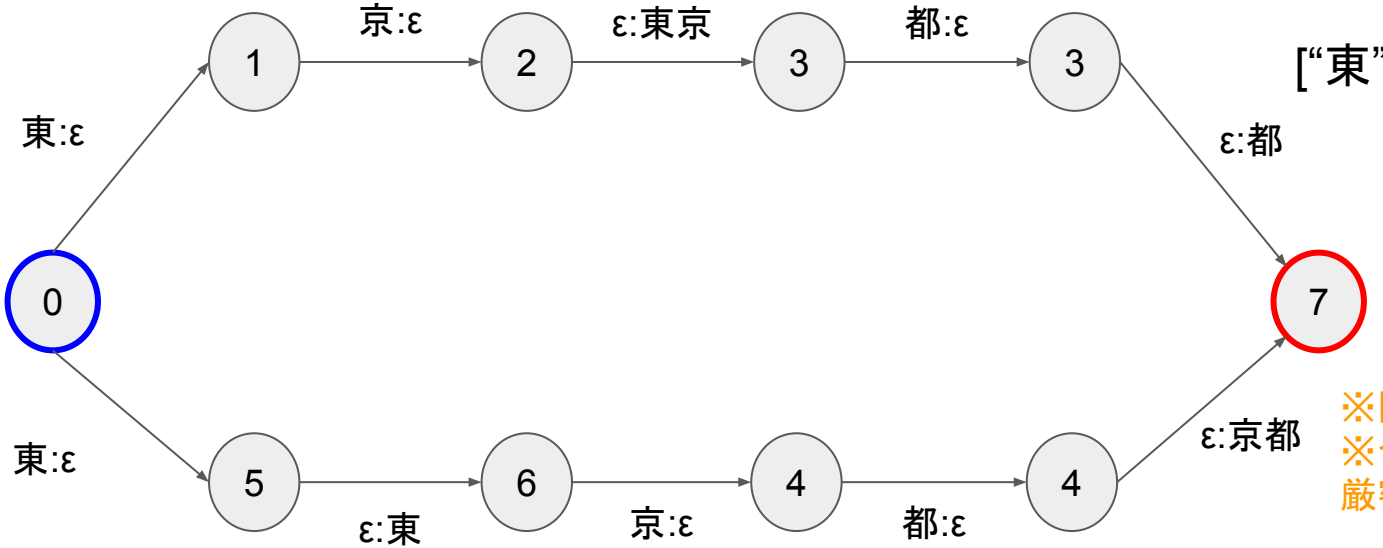


有限状態トランスデューサによる分かち書き

分かち書きトランスデューサ $I \circ D^*$

- 入力アルファベット(文字)　　: {“東”, “京”, “都”}
- 出力アルファベット(語彙)　　: {“東”, “京都”, “東京”, “都”}

入力列
[“東”, “京”, “都”]
↓
出力列
[“東京”, “都”]
or
[“東”, “京都”]



※図は雰囲気。
※合成の定義とは
厳密には違うかも

来週までにやりたいこと

- 実装を進める
  - トランスデューサを図として描画できるツールを探して利用する
  - ε-removal, determinize, minimize などの重要な演算を実装する
  - UniDic の情報をうまく取り出せるようにする
- 論文を読む
  - せっかくなので、知識グラフに関する論文にも目を通して、知見を得たい。
    - そもそも Conventional Method もよくわかっていないので、そこから。

参考文献

1. 有限状態トランスデューサに関して
    a. **Mohri. 2009. Weighted Automata Algorithms. Handbook of weighted automata. Springer.**
    b. Sproat et al. 1996. A Stochastic Finite-State Word-Segmentation Algorithm for Chinese. Computational Linguistics.
    c. Kaplan and Kay. 1994. Regular Models of Phonological Rule Systems. Computational Linguistics.
2. 知識グラフに関して
    a. **Friedman, Broeck. 2020. Symbolic Querying of Vector Spaces: Probabilistic Databases Meets Relational Embeddings. ArXiv.**
3. 実装にあたり参考にさせていただいたサイト
    a. http://e-yuki67.hatenablog.com/entry/2017/02/05/155312
    b. http://e-yuki67.hatenablog.com/entry/2017/02/26/212444