

正規表現の脆弱性

ReDoS 攻撃

を簡単に説明する回



xryuseix

@ryusei_ishika



1回生が入ったので、自己紹介

- 名前：石川琉聖(ID: xryuseix)
- 三回 情理 SNコースです
- セキュリティとアルゴリズムをやります
- いろんなイベントに出没します
- RiST元副団体長, RiPPro元団体長です
- C++とPythonをよく書きます

Event

年	内容
2019	セキュリティ・キャンプ全国大会2019 集中開発コース 暗号化通信ゼミ
2020-2021	若手セキュリティイノベーター育成プログラム SecHack365 研究駆動コース
2020	AVTOKYO 2020 Talks 情報通信システムセキュリティ研究会 (ICSS) ▼ 研究テーマ ↓ これICSS論文賞いただきました🏆
2021	<ul style="list-style-type: none"> 「仮想背景を使用したリモート会議映像における秘匿された背景の再構築手法」 ○辻知希, 石川琉聖 (立命館大) ・衛藤将史 (NICT) ・服部祐一 (セキュアサイクル) ・井上博之 (広島市大) 「プログラミングコンテストにおけるソースコードの盗作検知手法の実装と評価」 ○石川琉聖 (立命館大) ・服部祐一 (セキュアサイクル) ・井上博之 (広島市大) ・猪俣敦夫 (阪大)
2021	ICPC アジア地区横浜大会

Media

年	内容
2021	サイバーセキュリティII 第2回 情報セキュリティ教育と人材育成 BS231ch

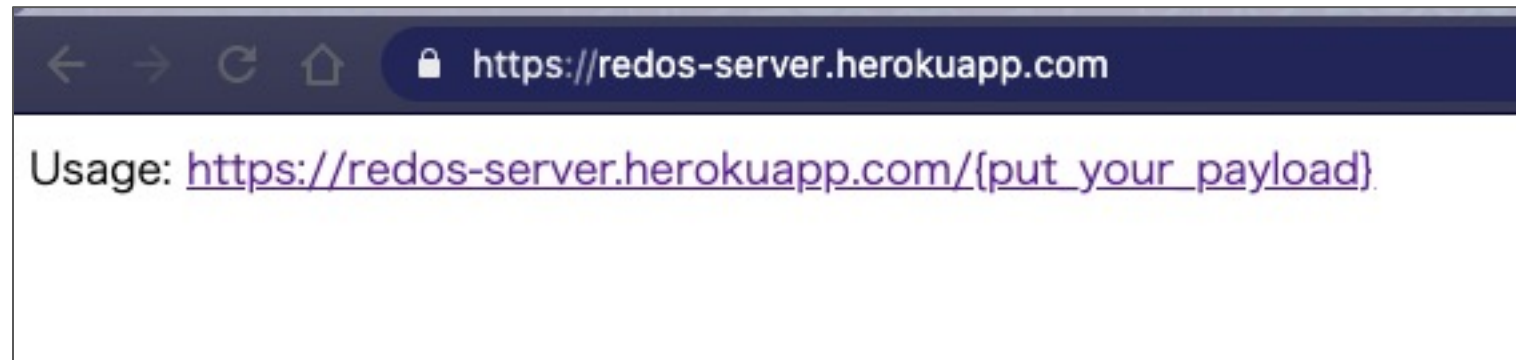
Hack

年	内容
2020-	IPA 脆弱性関連情報届出受理 42件 ▶ 取得番号一覧

<https://xryuseix.github.io/>

ここからが本題です

○ CTF の問題をプレゼント🎉



<https://redos-server.herokuapp.com/>

(開発ソースコード: https://github.com/xryuseix/ReDoS_server)

正規表現とは

- 文字列の集合を一つの文字列で表現する方法
- 言語によらず基本的には一緒

a から始まって z で終わる 3 桁の文字列

1. | a.z

a から始まって z で終わる 2 桁以上の文字列

1. | a.*z

4 桁以上の半角数字

1. | \d{4,}

section. 3

使用頻度の高い正規表現式

Email アドレス (RFC準拠ではない)

`^\w+([-+.]\w+)*@\w+([-+]\w+)*\.\w+([-+]\w+)*$`

URL

`^https?://([\w-]+\.)+[\w-]+(/[\w-./?%&=]*)?$`

ドメイン名

`^[a-zA-Z0-9][a-zA-Z0-9-]{1,61}[a-zA-Z0-9]\.[a-zA-Z-]{2,}$`

固定電話番号

`^0\d(-\d{4}|\d-\d{3}|\d\d-\d\d|\d{3}-\d)-\d{4}$`

携帯電話番号

`^0[789]0-\d{4}-\d{4}$`

IP 電話番号

`^050-\d{4}-\d{4}$`

フリーダイヤル

`^(0120|0800)-\d{3}-\d{3}$`

日付 (YYYY-MM-DD形式)

`^\d{4}-\d\d-\d\d$`

郵便番号

`^\d{3}-\d{4}$`

<https://murashun.jp/article/programming/regular-expression.html>

正規表現はどう使うのか

○ 基本的にできること

- 大量の文字から特定の文字列を探す
 - “This is a pen.”の中に“pen”があるかまた何文字目にあるか
- 文字列が正しい形式か判定する
 - 電話番号に0-0-0-0みたいなものを入力されないように
 - 入力に変な文字(制御文字など)を入力されないように

○ ↑を用いてすること

- 文字列の置換
 - “This is a pen. I like pen”の中にある“pen”を全て“apple”に変える
- 文字列の分割
 - 文字列をスペース, カンマ, タブのいずれかで分割

正規表現はどう使うのか

○ 基本的にできること(**Python re モジュール**)

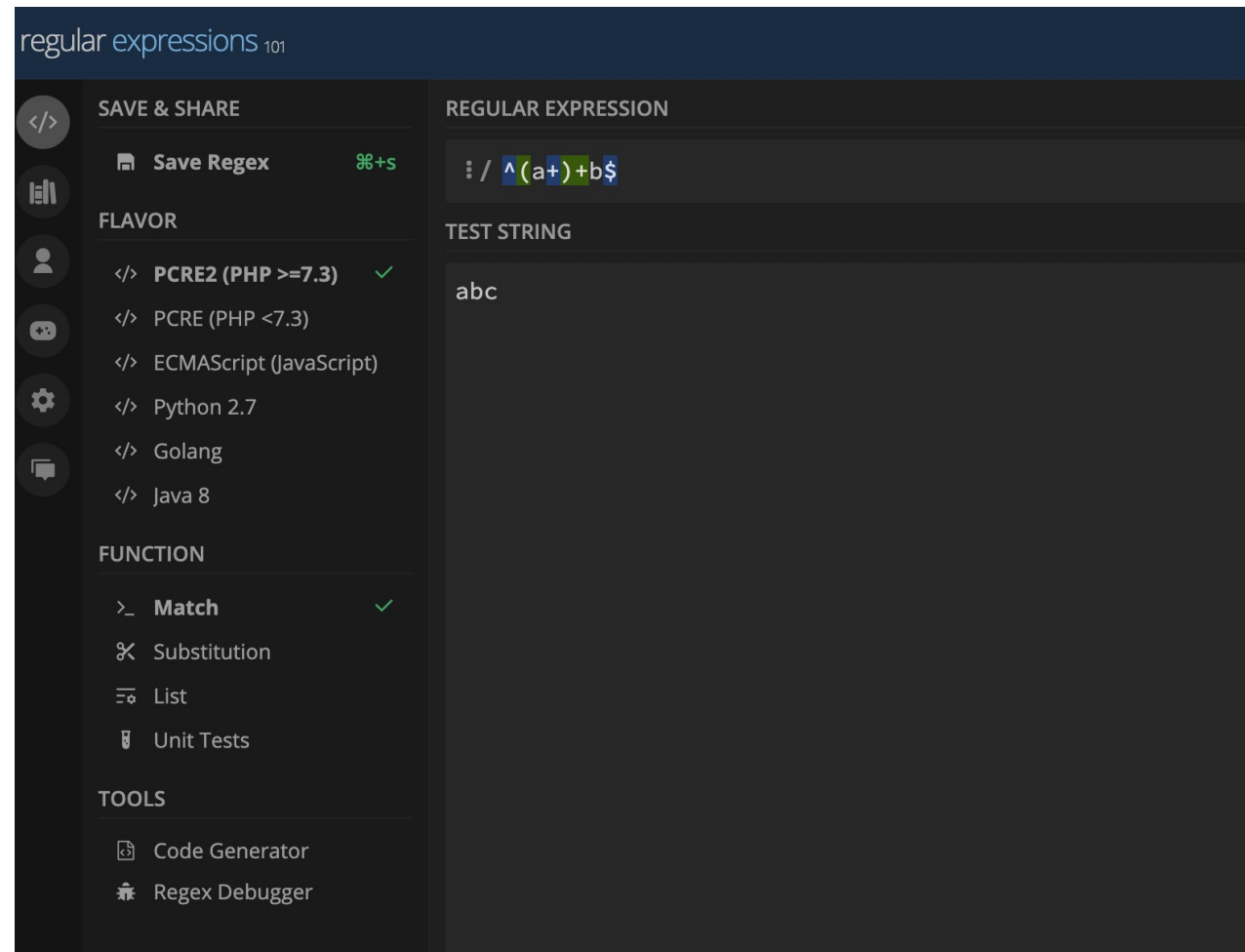
- 大量の文字から特定の文字列を探す (re.search)
 - “This is a pen.”の中に“pen”があるかまた何文字目にあるか
- 文字列が正しい形式か判定する (re.match)
 - 電話番号に0-0-0-0みたいなものを入力されないように
 - 入力に変な文字(制御文字など)を入力されないように

○ ↑を用いてすること

- 文字列の置換 (re.sub)
 - “This is a pen. I like pen”の中にある“pen”を全て“apple”に変える
- 文字列の分割 (re.split)
 - 文字列をスペース, カンマ, タブのいずれかで分割

実行の様子を確認する

○ 正常 1



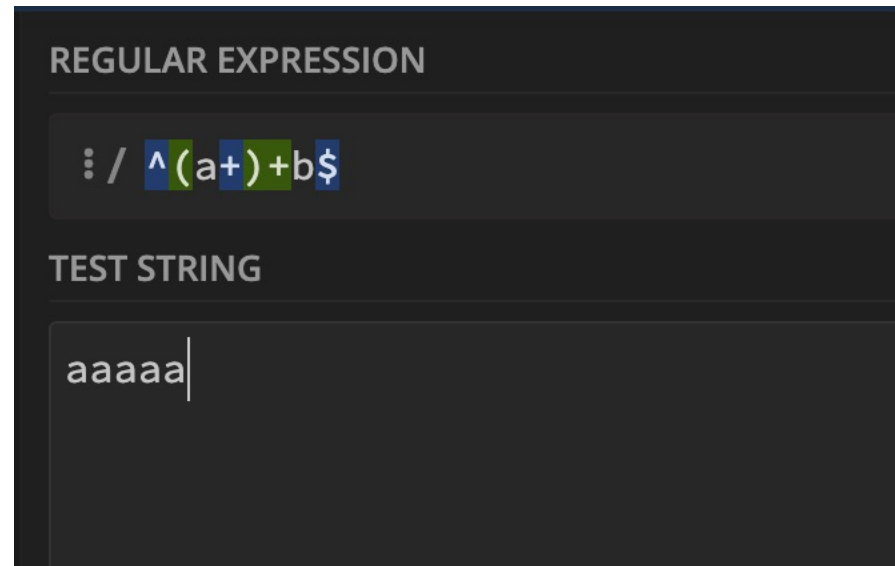
<https://regex101.com/>

○ 正常 1

The screenshot displays the regex101.com web application interface. At the top, the 'MATCH STEPS' section shows a progress bar with markers at 1, 3, 4, 6, and 7. The first step is highlighted, and a blue dot indicates the current position. Below this, the 'MATCH STEP 1' section shows the regex pattern `/^(a+)+b$/` and the input string `abc`. The input string is highlighted in green, indicating a successful match. The interface is dark-themed.

<https://regex101.com/>

○ 正常2



<https://regex101.com/>

○ 正常2

MATCH STEPS

1 1 19 38 58 77 96 96

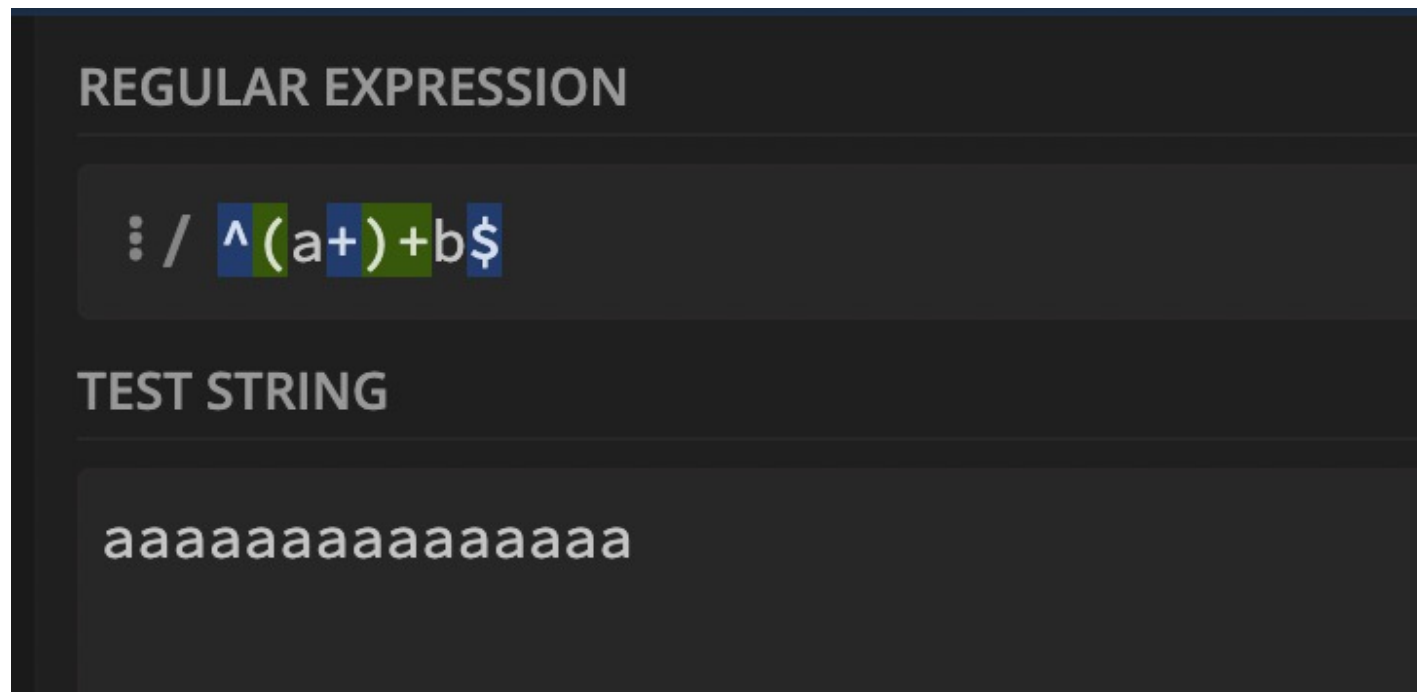
Available keyboard shortcuts

MATCH STEP 1

```
/^(a+)+b$  
aaaaa
```

<https://regex101.com/>

○ 異常



<https://regex101.com/>

実行の様子を確認する

○ 異常

最初の200手↓

MATCH STEPS

1 1 40 80 120 160 200 200

Available keyboard shortcuts

MATCH STEP 1

```
/^(a+)+b$  
aaaaaaaaaaaaaaaa
```

<https://regex101.com/>

実行の様子を確認する

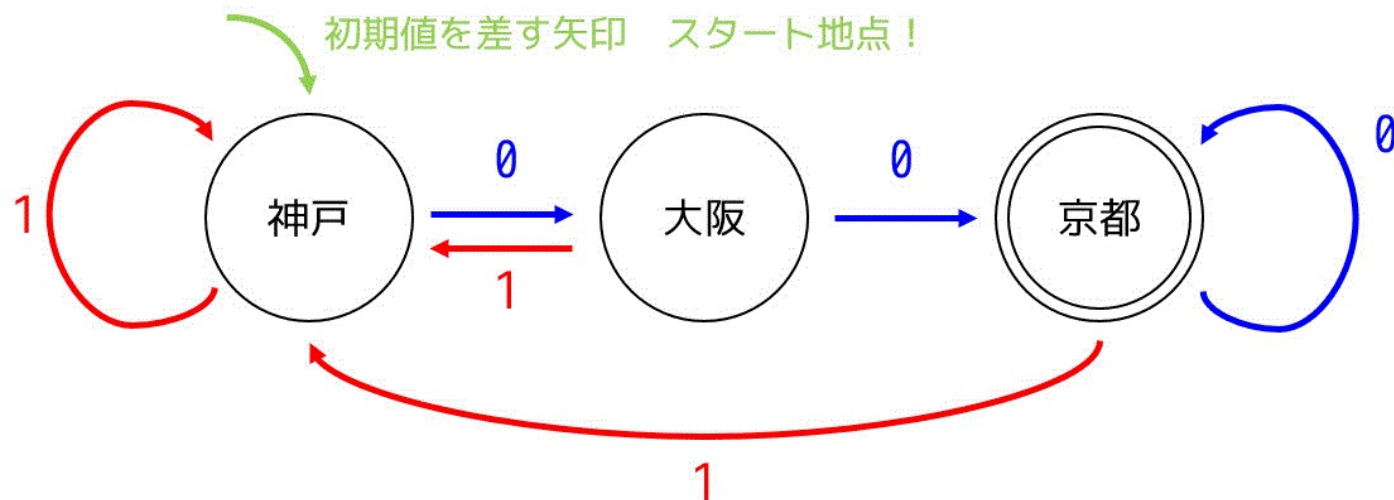
- aの個数が増えるとステップ数はどうなるのか

aの個数	ステップ数
3	24
5	96
10	3072
15	98304
20	119987

なぜこうなのか

- 正規表現はオートマトンによって文字列を評価しています
- オートマトンとは？ (↓こんな感じの)

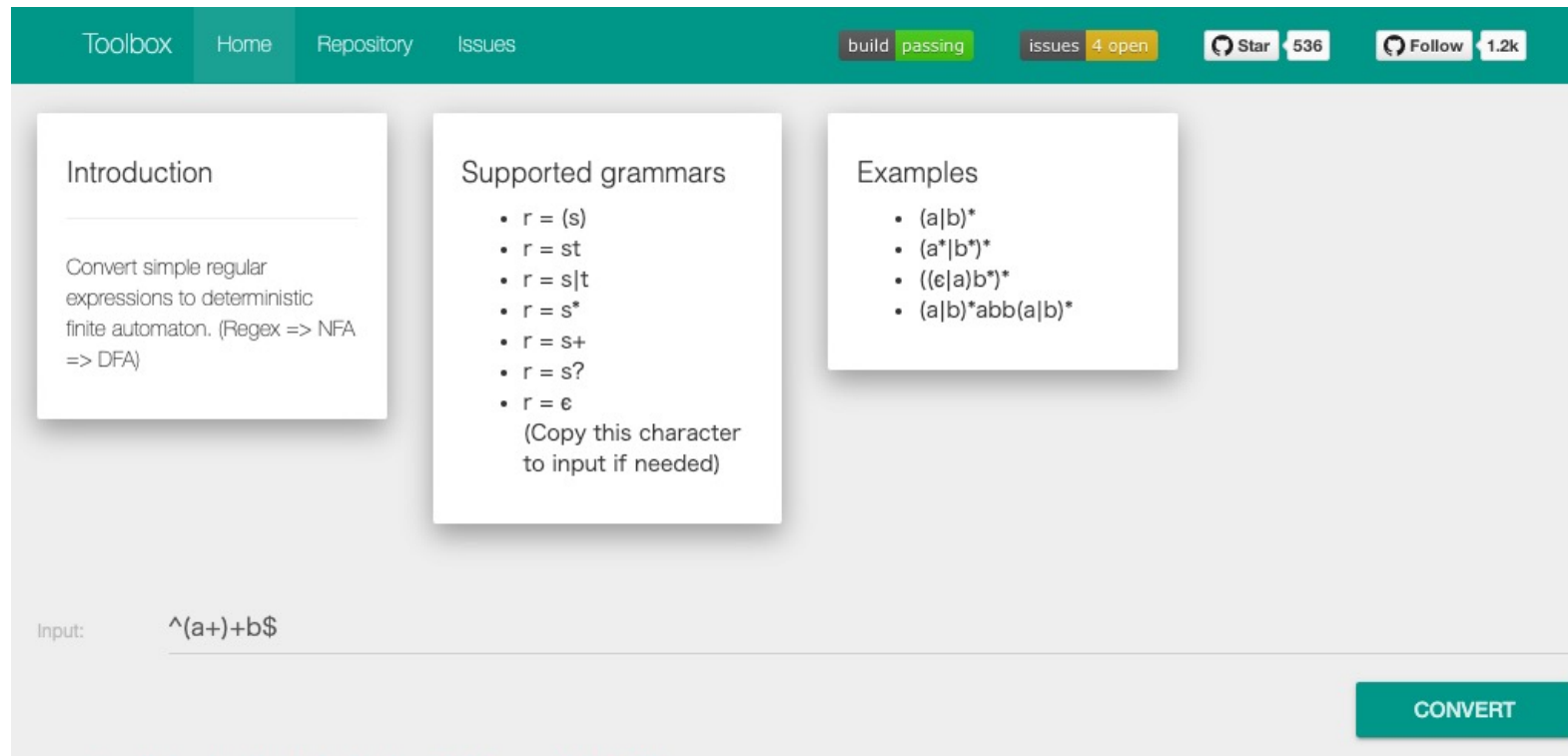
$\Sigma = \{0,1\}$: サイコロの目一覧



<https://www.momoyama-usagi.com/entry/info/automaton01>

なぜこうなるのか

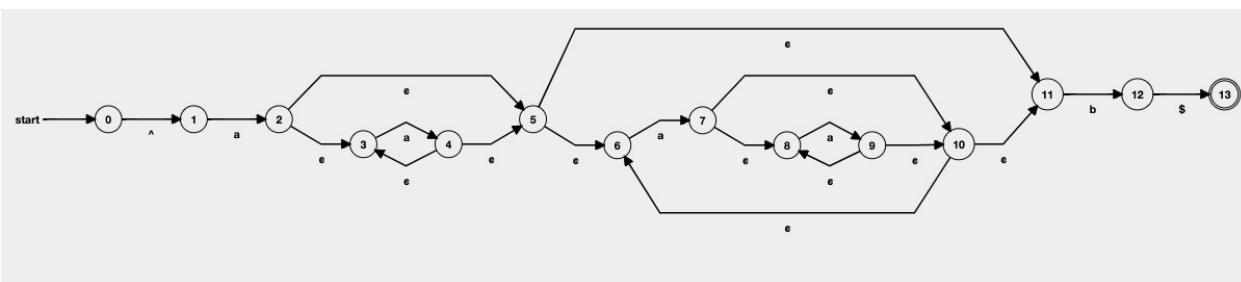
- 正規表現はオートマトンによって文字列を評価しています
- あの正規表現のオートマトンは？



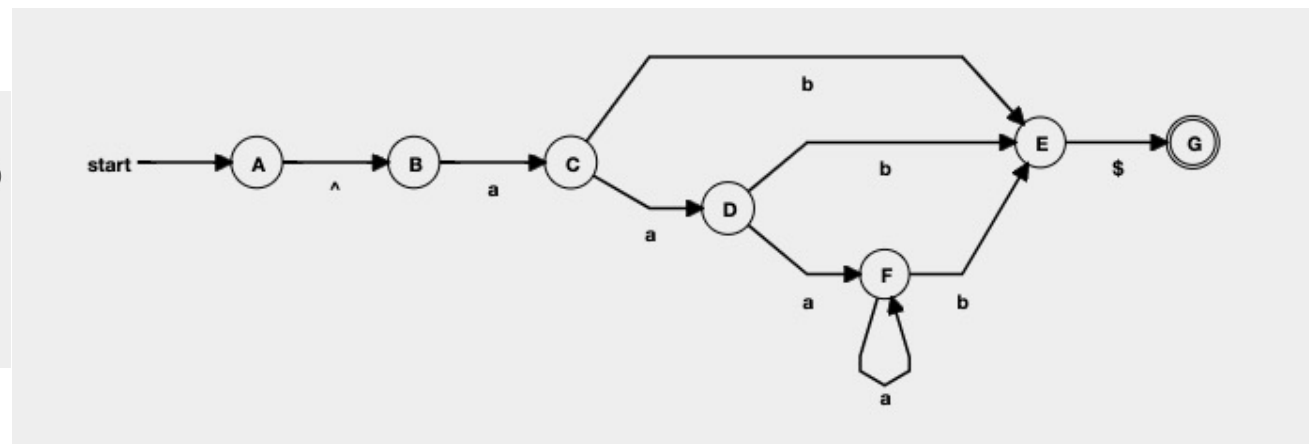
<https://cyberzhg.github.io/toolbox/nfa2dfa>

なぜこうなるのか

- 正規表現はオートマトンによって文字列を評価しています
- あの正規表現のオートマトンは？



Pythonのreはこっち
非決定性オートマトン(NFA)

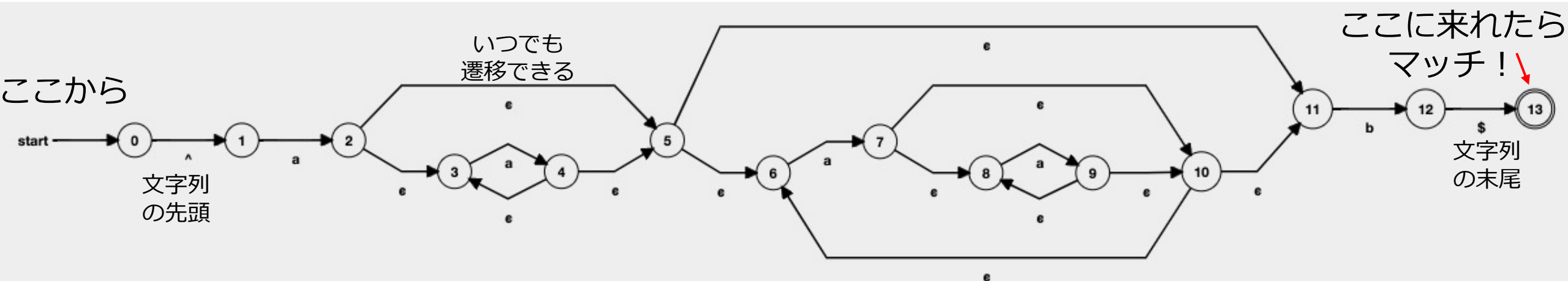


決定性オートマトン(DFA)

<https://cyberzhg.github.io/toolbox/nfa2dfa?regex=XihhKykrYiQ=>
<https://cyberzhg.github.io/toolbox/regex2nfa?regex=XihhKykrYiQ=>

なぜこうなるのか

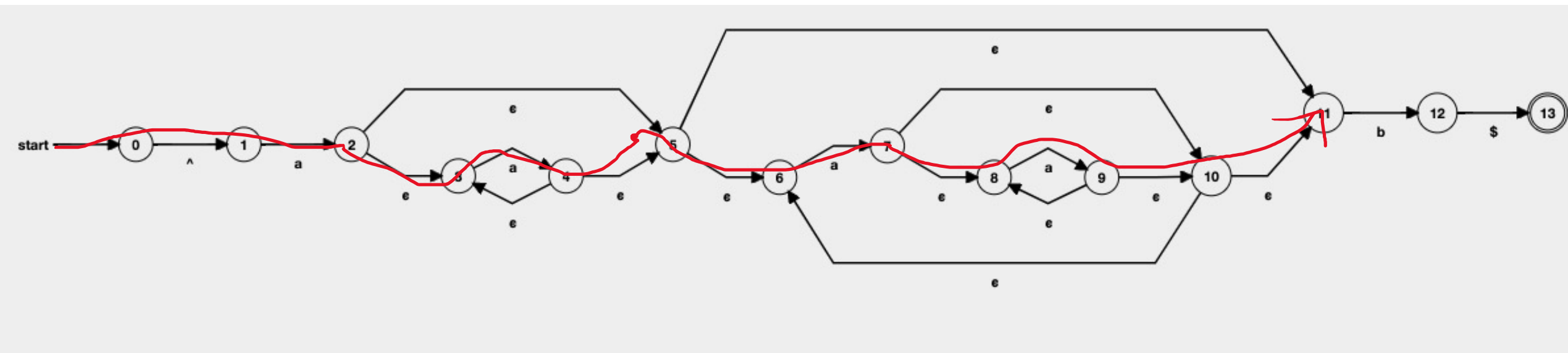
- 正規表現はオートマトンによって文字列を評価しています
- あの正規表現のオートマトンは？



<https://cyberzhg.github.io/toolbox/regex2nfa?regex=XihhKykrYiQ=>

なぜこうなるのか

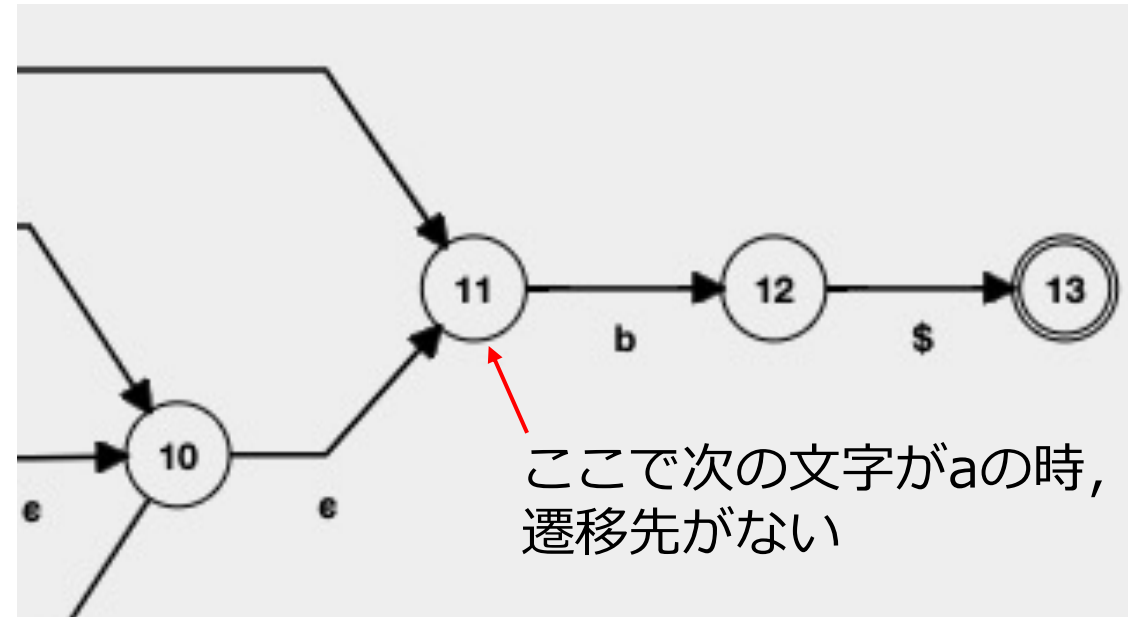
- 正規表現 : $^(a+)+b\$$
- 入力: aaaaaa



<https://cyberzhg.github.io/toolbox/regex2nfa?regex=XihhKykrYiQ=>

なぜこうなるのか

- 正規表現 : $^(a+)+b\$$
- 入力: aaaaaa

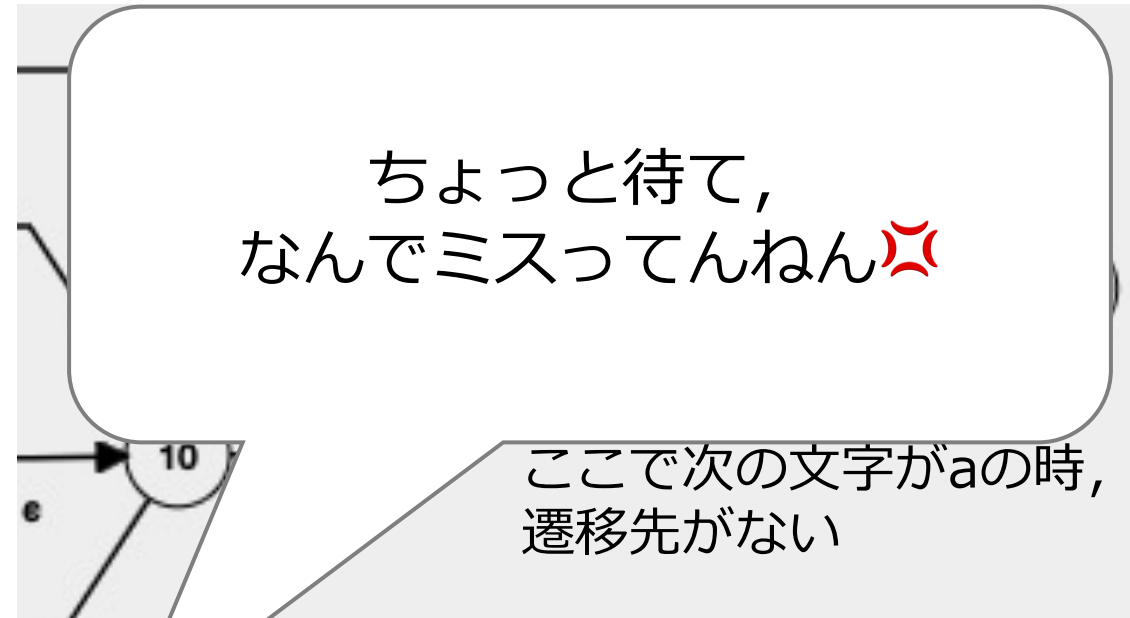


遷移先がないとき, 前にミスったのかなって考える
→ちょっと戻る

<https://cyberzhg.github.io/toolbox/regex2nfa?regex=XihhKykrYiQ=>

なぜこうなるのか

- 正規表現 : $^(a+)+b\$$
- 入力: aaaaaa



遷移先がないとき, 前にミスったのかなって考える
→ちょっと戻る

<https://cyberzhg.github.io/toolbox/regex2nfa?regex=XihhKykrYiQ=>

なぜこうなるのか

- 正規表現： $^(a+)+b\$$
- 入力：aaaaaa

テーマ：+の適用範囲が難しい
問題： $(a+)$ と+はどの部分を指しますか？

a	a	a	a	a	a
_____			_____		
(a+)			+		
_____		_____			
(a+)		+			
_____	_____				
(a+)	+				

→いっぱいあって難しいから、とりあえずいっぱい進んで、
ダメだったら少し戻る方針

MATCH STEPS

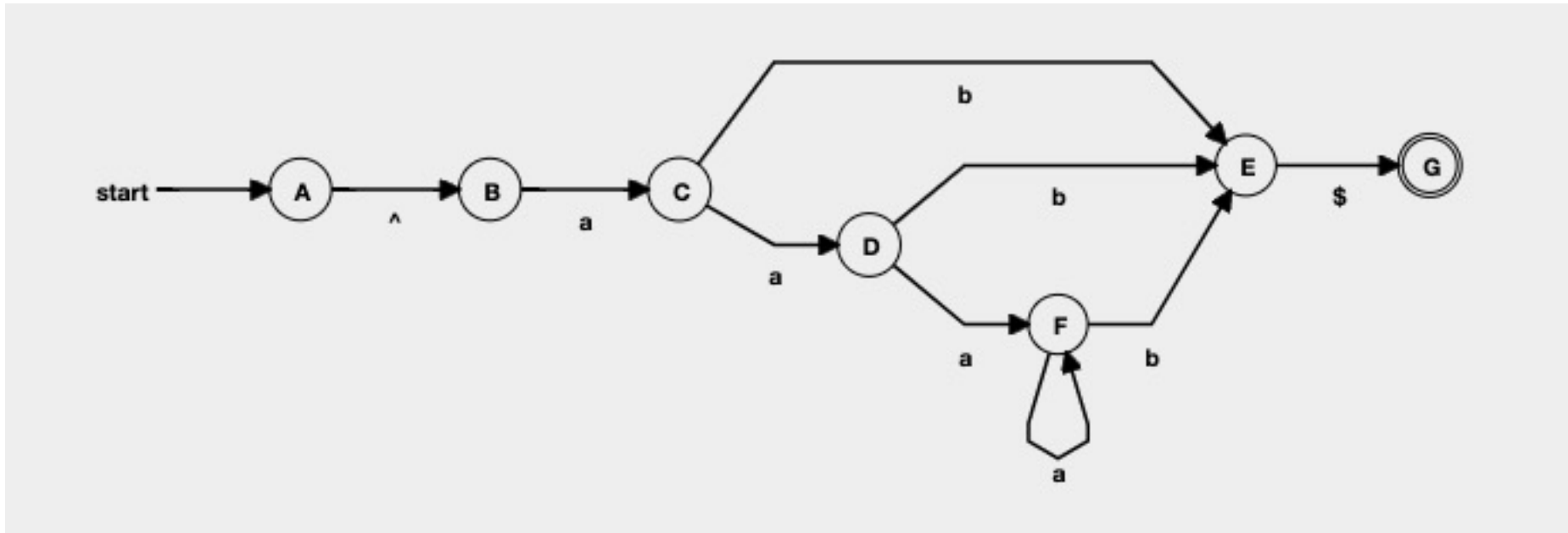
1 1 40 80 120 160 200

Available keyboard shortcuts

MATCH STEP 1

```
/^(a+)+b$  
aaaaaaaaaaaaaaaa
```

→ とりあえずいっぱい進んで、ダメだったら少し戻る方針



決定性オートマトン(DFA)が使われるライブラリを使う

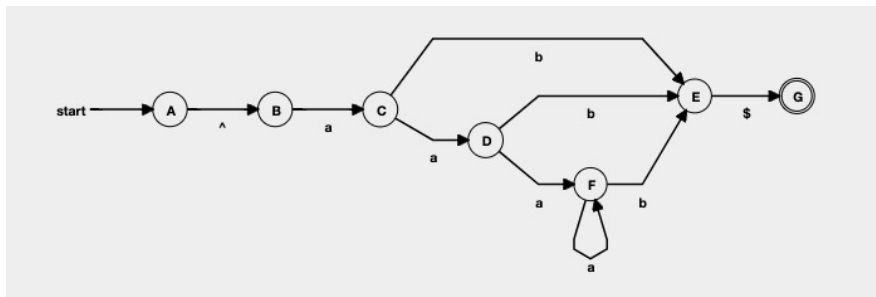
pyre2 0.3.6

`pip install pyre2`



<https://cyberzhg.github.io/toolbox/nfa2dfa?regex=XihhKykrYiQ=>

対策



pyre2 0.3.6

pip install pyre2



決定性オートマトン(DFA)
が使われるライブラリ

予防

- 正規表現はできるだけコピー
 - メアド, 電話番号などは拾える
 - ユーザの入力は信用しない
- 繰り返し表現はできるだけ使わない
 - +や*
- 入力文字数制限
 - 最大30文字までなど
- タイムアウトを設定
 - 実行時間は最大10秒など