テキスト処理 第6回 (2008-05-27)
enumreレポート解説

田中哲
産業技術総合研究所
情報技術研究部門
akr@isc.senshu-u.ac.jp
http://staff.aist.go.jp/tanaka-akira/textprocess-2008/

レポート

- enumre を以下のように拡張せよ
  - 単一文字を配列でなく記述できるようにする
  - :cat で任意個の引数をとれるようにする
- 以下について考察せよ
  - 繰り返しを使っても結果が無限集合にならない場合
- 〆切 2008-05-27 12:00
- RENANDI
- 拡張子が txt なプレインテキストが望ましい
  - docx は読めません (MS Word の XML 形式?)
  - doc も避けてください (MS Word ファイル)

単一文字の配列でない記述

- [:char, "x"] を "x" と書けるようにする
- [:alt, [:cat, [:char, "c"],
[:cat, [:char, "a"],
[:char, "t"]]],
[:cat, [:char, "d"],
[:cat, [:char, "o"],
[:char, "g"]]]] を
[:alt, [:cat, "c", [:cat, "a", "t"]],
[:cat, "d", [:cat, "o", "g"]]] と書けるようにする

ヒント

- 文字列 (だけ) は to_str メソッドを持つ

実装

```
def enumre(r)
  if r.respond_to? :to_str
    [r]
  else
    case r[0]
    ...
    end
  end
end
```

- respond_to? :to_str で文字列かどうか判断する
- 文字列だったらそれを唯一の要素とする集合が答えとなるので、それを表現する配列を返す

多かった間違い

- [r] を返すべきところで r を返す
  enumre([:char, "x"]) は ["x"] を返すので
  enumre("x") も同様に ["x"] を返さなければならない

:cat の任意個引数

- [:cat, "d", [:cat, "o", "g"]] と書くのは面倒
- [:cat, "d", "o", "g"] と書きたい
- なので書けるようにせよ

実装

```
when :cat
  ret = [""]
  1.upto(r.length-1) {|i|
    set1 = enumre(r[i])
    set2 = []
    ret.each {|s1|
      set1.each {|s2|
        set2 << (s1 + s2)
      }
    }
    ret = set2
  }
  ret
```

:cat の任意個引数のしくみ

- 空文字列のみを要素とする集合から始める
- 各正規表現引数を文字列集合に変換する
- それより左の文字列集合と組み合わせて新しい文字列集合を作る

再帰を使った別解

```
when :cat
  if r.length == 1
    [""]
  else
    set1 = enumre(r[1])
    rr = [:cat]
    2.upto(r.length-1) {|i|
      rr << r[i]
    }
    set2 = enumre(rr)
    ret = []
    set1.each {|s1|
      set2.each {|s2|
        ret << (s1 + s2)
      }
    }
    ret
end
```

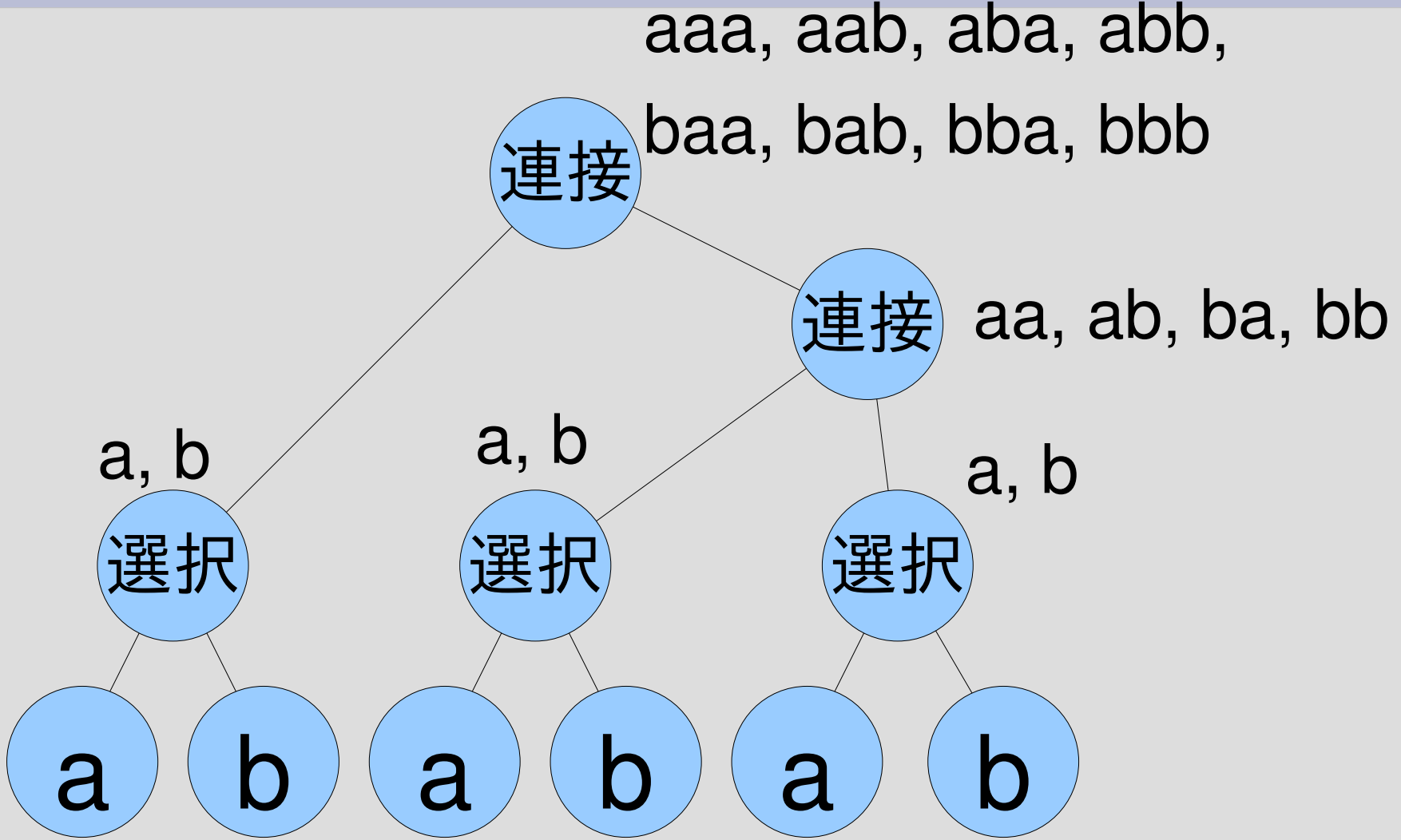
再帰版のしくみ

- [:cat] なら [""] を返す
- [:cat, r1, ...] なら
  enumre(r1) と
  enumre([:cat, ...]) を組み合わせる

多かった間違い

- 組み合わせになっていない
  [:cat, [:alt, "a", "b"],
  [:alt, "a", "b"],
  [:alt, "a", "b"]]
  から
  ["aaa", "aab", "aba", "abb",
  "baa", "bab", "bba", "bbb"]
  を求めなければならないが、そうなっていない

(a|b)(a|b)(a|b) での組み合わせ



繰り返しを使っても結果が無限集合にならない場合

- 繰り返しを使うと正規表現に対応する文字列集合はだいたい無限集合になる
  - a*　　#=> "", a, aa, aaa, aaaa, ...
  - (a|b)* #=> "", a, b, aa, ab, ba, bb, aaa, ...
  - (pine|apple)* #=> "", pine, apple, pinepine, pineapple, applepine, appleapple, ...
- じつは無限集合にならない場合が存在する
- それはどんな場合か考えよ

繰り返しても有限になる場合

- 空集合の繰り返しは空集合
  ϕ^* は ϕ と等しい
- 空文字列の繰り返しは空文字列
  ε^* は ε と等しい

集計

- 空集合だけ　　8
- 空文字列だけ　　4
- 空集合と空文字列　3
- その他　　10
  - 繰り返し回数の上限をつける
  - etc.