「グラフと組み合わせ」課題 7 (解答例)

2013/5/27

1　列挙

n 個の文字の集合

$$S = \{a_0, a_1, \ldots, a_{n-1}\}$$

の全てからなる文字列、つまり同じ文字を含まない、長さ n の文字列を列挙する方法を考える。

1. 何通りの文字列があるかを答えなさい。また、そのことが正しいことを数学的帰納法で示しなさい。
2. 文字列を列挙する再帰的アルゴリズムを構築しなさい。
3. $n = 4$ の場合に、上記のアルゴリズムに従って文字列を列挙しなさい。列挙する途中の過程についても示しなさい。

解答例

1. n 個の文字から構成され、同じ文字を含まない文字列の総数は $n!$ である。このことを数学的帰納法で証明する。
   - $n = 1$ の場合、明らか。
   - ある文字数 n の時、文字列の総数が $n!$ であると仮定する。各文字列に対して、新しい文字 α を加えた文字列を生成することを考える。文字の間が $n-1$ 箇所、それに先頭と末尾を加え、文字 α を加えて長さ $n+1$ の文字列を作る方法は通りである。$n!$ 個のそれぞれに一文字加える方法が $n+1$ 通りあるため、$(n+1) \times n! = (n+1)!$ 通りの文字列が生成される。
2. アルゴリズムを enumString(L, A) とする。ここで、L は、文字列作成に既に使用した文字のリストとする。つまり、最後には生成された文字列となる。その初期値は空である。A は、文字列として未だ使用されていない文字のリストとする。その初期値は S である。

```
enumString(L, A){
    if (|A| == 0) {L を印刷}
    else {
        B は A の複製
        forall (s ∈ A) {
            L ← L ∪ {s}
            B ← B \ {s}
            enumString(L, B)
            L ← L \ {s}
            B ← B ∪ {s}
        }
    }
}
```

3.　$S=\{a,b,c,d\}$ の場合の動作例を示す。再帰関数を f とする。

$f\{[\ ],[a,b,c,d]\}$	$f\{[a],[b,c,d]\}$	$f\{[a,b],[c,d]\}$	$f\{[a,b,c],[d]\}$	$f\{[a,b,c,d],[\ ]\}$
			$f\{[a,b,d],[c]\}$	$f\{[a,b,d,c],[\ ]\}$
		$f\{[a,c],[b,d]\}$	$f\{[a,c,b],[d]\}$	$f\{[a,c,b,d],[\ ]\}$
			$f\{[a,c,d],[c]\}$	$f\{[a,c,d,b],[\ ]\}$
		$f\{[a,d],[b,c]\}$	$f\{[a,d,b],[c]\}$	$f\{[a,d,b,c],[\ ]\}$
			$f\{[a,d,c],[d]\}$	$f\{[a,d,c,b],[\ ]\}$
	$f\{[b],[a,c,d]\}$	$f\{[b,a],[c,d]\}$	$f\{[b,a,c],[d]\}$	$f\{[b,a,c,d],[\ ]\}$
			$f\{[b,a,d],[c]\}$	$f\{[b,a,d,c],[\ ]\}$
		$f\{[b,c],[a,d]\}$	$f\{[b,c,a],[d]\}$	$f\{[b,c,a,d],[\ ]\}$
			$f\{[b,c,d],[a]\}$	$f\{[b,c,d,a],[\ ]\}$

		$f\{[b,d],[a,c]\}$	$f\{[b,d,a],[c]\}$	$f\{[b,d,a,c],[\ ]\}$
			$f\{[b,d,c],[a]\}$	$f\{[b,d,c,a],[\ ]\}$
	$f\{[c],[a,b,d]\}$	$f\{[c,a],[b,d]\}$	$f\{[c,a,b],[d]\}$	$f\{[c,a,b,d],[\ ]\}$
			$f\{[c,a,d],[b]\}$	$f\{[c,a,d,b][\ ]\}$
		$f\{[c,b],[a,d]\}$	$f\{[c,b,a],[d]\}$	$f\{[c,b,a,d],[\ ]\}$
			$f\{[c,b,d],[a]\}$	$f\{[c,b,d,a],[\ ]\}$
		$f\{[c,d],[a,b]\}$	$f\{[c,d,a],[b]\}$	$f\{[c,d,a,b],[\ ]\}$
			$f\{[c,d,b],[a]\}$	$f\{[c,d,b,a],[\ ]\}$
	$f\{[d],[a,b,c]\}$	$f\{[d,a],[b,c]\}$	$f\{[d,a,b],[c]\}$	$f\{[d,a,b,c],[\ ]\}$
			$f\{[d,a,c],[b]\}$	$f\{[d,a,c,b],[\ ]\}$
		$f\{[d,b],[a,c]\}$	$f\{[d,b,a],[c]\}$	$f\{[d,b,a,c],[\ ]\}$
			$f\{[d,b,c],[a]\}$	$f\{[d,b,c,a],[\ ]\}$
		$f\{[d,c],[a,b]\}$	$f\{[d,c,a],[b]\}$	$f\{[d,c,a,b],[\ ]\}$
			$f\{[d,c,b],[a]\}$	$f\{[d,c,b,a],[\ ]\}$

2 実装

前問で作成した再帰的アルゴリズムをプログラムとして実装し、動作を確認しなさい。

解答例

プログラムは別紙に示す。実行結果を以下に示す。各行の最後が列挙すべき長

さ 4 の文字列である。

```
  a  ab  abc  abcd
  abd  abdc
  ac  acb  acbd
  acd  acdb
  ad  adb  adbc
  adc  adcb
  b  ba  bac  bacd
  bad  badc
  bc  bca  bcad
  bcd  bcda
  bd  bda  bdac
  bdc  bdca
  c  ca  cab  cabd
  cad  cadb
  cb  cba  cbad
  cbd  cbda
  cd  cda  cdab
  cdb  cdba
  d  da  dab  dabc
  dac  dacb
  db  dba  dbac
  dbc  dbca
  dc  dca  dcab
  dcb  dcba
24 strings are found.
```

EnumString.java

```java
package EnumerateString;

import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

/**
 *
 * @author tadaki
 */
public class EnumString {

    private List<Character> charList = null;
    private final boolean debug = true;
    private List<String> stringList;

    /**
     * コンストラクタ
     *
     * @param chars 使用する文字の配列
     */
    public EnumString(char chars[]) {
        /**
         * 文字集合への変換
         */
        charList = createList();
        for (int i = 0; i < chars.length; i++) {
            charList.add(Character.valueOf(chars[i]));
        }
    }

    /**
     * 空のリストの生成
     *
     * @return
     */
    static public <T> List<T> createList(){
        return Collections.synchronizedList(new ArrayList<T>());
    }

    /**
     * リストの複製
     *
     * @param orig 複製元
     * @return 複製
     */
```

```java
    static public <T> List<T> copyList(List<T> orig) {
        if (orig == null) {
            return null;
        }
        List<T> nList = createList();
        for (T t : orig) {
            nList.add(t);
        }
        return nList;
    }

    public int enumerateStringMain() {
        List<Character> available = copyList(charList);
        List<Character> str = createList();
        stringList = createList();
        return enumerateString(str, 0, available);
    }

    /**
     * 文字列の列挙（再帰）
     *
     * @param str 構成された文字列
     * @param n これまでに構成した文字列の数
     * @param available 追加できる文字のリスト
     * @return このメソッドによって構成した文字列の数の累積
     */
    private int enumerateString(List<Character> str, int n,
            List<Character> available) {
        if (debug) {
            printString(str, false);
        }
        if (available.isEmpty()) {//使用できる全ての文字を使用
            stringList.add(list2String(str));
            n++;
            if (debug) {
                System.out.println();
            }
        } else {
            List<Character> nAvailable = copyList(available);
            for (Character c : available) {
                str.add(c);
                nAvailable.remove(c);
                n = enumerateString(str, n, nAvailable);
                str.remove(c);
                nAvailable.add(c);
            }
```

```
        }
        return n;
    }

    public String list2String(List<Character> list) {
        StringBuilder b = new StringBuilder();
        for (int i = 0; i < list.size(); i++) {
            b.append(list.get(i));
        }
        return b.toString();
    }

    /**
     * 文字列の印刷
     *
     * @param str 構成された文字列
     * @param n 文字列の番
     * @param b 真ならば最終出力
     */
    private void printString(List<Character> list, boolean b) {
        String str = list2String(list);
        System.out.print("  ");
        System.out.print(str);
        if (b) {
            System.out.println();
        }
    }

    public List<String> getStringList() {
        return stringList;
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        char chars[] = {'a', 'b', 'c', 'd'};
        EnumString eString = new EnumString(chars);
        int n = eString.enumerateStringMain();
        System.out.println(String.valueOf(n) + " strings are found.");
        List<String> list = eString.getStringList();
        for (String s : list) {
            System.out.println(s);
        }
    }
}
```

EnumString.java

main.cpp

```cpp
/*
 * File:   main.cpp
 * Author: tadaki
 *
 * Created on 2013/05/30, 10:35
 */

#include <cstdlib>
#include "EnumString.h"
using namespace std;


/*
 *
 */
int main(int argc, char** argv) {
    char chars[] = {'a', 'b', 'c', 'd'};
    EnumString enumString(chars, 4);
    int n = enumString.enumStringMain();
    cout << n;
    cout << " strings are found¥n";
    list<char*> strList = enumString.getStringList();
    list<char*>::iterator j;
    for (j = strList.begin(); j != strList.end(); j++) {
        cout << *j;
        cout << "¥n";
    }
    return 0;
}
```

EnumString.h

```
/*
 * File:   EnumString.h
 * Author: tadaki
 *
 * Created on 2013/05/30, 13:21
 */
#include <list>
#ifndef EnumString_H
#define EnumString_H
using namespace std;

class EnumString {
public:
    EnumString(const char[], int);
    EnumString(const EnumString& orig);
    virtual ~EnumString();
    int enumStringMain();
    list<char*> getStringList();
private:
    int enumString(char*, int, int,list<char>);
    char* charList;
    list<char*> strList;
    bool debug;
};

#endif  /* EnumString_H */
```

EnumString.cpp

```cpp
/*
 * File:   EnumString.cpp
 * Author: tadaki
 *
 * Created on 2013/05/30, 13:21
 */

#include "EnumString.h"

EnumString::EnumString(const char chars[], int n) {
    charList = new char[n + 1];
    for (int i = 0; i < n; i++)charList[i] = chars[i];
    charList[n] = '¥0';
    debug = true;
}

EnumString::EnumString(const EnumString& orig) {
}

EnumString::~EnumString() {
}

int EnumString::enumStringMain() {
    //最長文字列に対応した文字列領域を確保し、NULLを書き込む
    char* str;
    str = strdup(charList);
    list<char> available;
    for (int i = 0; i < strlen(charList); i++) {
        str[i] = '¥0';
        available.push_back(charList[i]);
    }
    strList.clear(); //文字列一覧の消去
    return enumString(str, 0, 0, available); //列挙開始
}

/**
 * 文字列列挙
 * @param str 生成中の文字列
 * @param n   生成した文字列の総数
 * @param p   文字列中の作業位置
 * @return    生成した文字列総数
 */
int EnumString::enumString(char* str, int n, int p, list<char>
available) {
    if (debug) {
        cout << str;
```

```
        cout << " ";
    }
    if (strlen(str) == strlen(charList)) {
        strList.push_back(strdup(str));
        n++;
        if (debug) {
            cout << "¥n";
        }
    } else {
        list<char> nAvailable;
        for (list<char>::iterator j = available.begin();
                j != available.end(); j++) {
            nAvailable.push_back(*j);
        }

        for (list<char>::iterator j = available.begin();
                j != available.end(); j++) {
            int k = p + 1;
            str[p] = *j;
            nAvailable.remove(*j);
            n = enumString(str, n, k, nAvailable);
            str[p] = '¥0';
            nAvailable.push_back(*j);
        }
    }
    return n;
}

list<char*> EnumString::getStringList() {
    return strList;
}
```