

# ソフトウェア2 第1回

鶴岡 慶雅

# TAオフィスアワー(通称 駆け込み寺)

- 場所
  - 駒場IIキャンパス先端研3号館3階308号室
  - 駒場教養キャンパスから徒歩10分程度
- 日程・時間
  - 12月6日(木)より毎週木曜日(12/6, 12/13, 12/20, 1/10, 1/17, 1/24)
  - 18:30~19:30
- 対象
  - 電気電子数学演習, 電気磁気学I・IIの講義・演習内容について質問があり、学習意欲のある学生であれば誰でも。事前予約など必要なし。
  - 正式には、数学と電磁気の質問を受け付ける場ですが、その他の講義についても、TAが分かる範囲で相談にのってもらえるはずです。
- 詳細

<http://www.ee.t.u-tokyo.ac.jp/~tanemura/lecture/EM/OfficeHour.pdf>

# 自己紹介

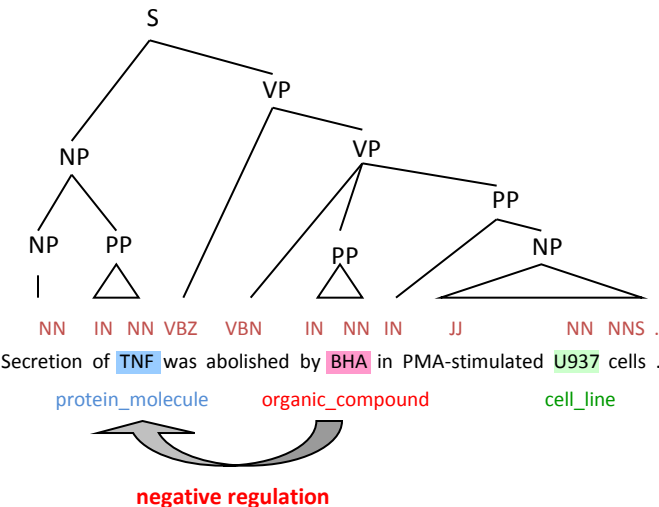
- 所属

- 工学系研究科 電気系工学専攻
- 工学部 電子情報工学科

## ● 専門分野

# 自然言語処理

## 自然言語(日本語・英語など)の文を解析・理解



# テキストマイニング

## 大量の文書から有用な知識を発見

The screenshot shows the Facta+ web application interface. At the top, there's a navigation bar with links like 'diabetes - FACTA Search', 'Home', 'About', 'FAQ', 'Contact', 'Help', and 'Log Out'. Below this is a search bar with the text 'diabetes - FACTA Search' and a search button. The main content area displays search results for 'diabetes'. It includes a list of concepts with their associated counts and a table of concepts ranked by frequency. The table has columns for Human, Disease, Symptom, Drug, and Enzyme, each with a list of related terms and their counts.

diabetes

Find Associated Concepts View Documents

☒ Gene/Protein ☒ Disease ☒ Symptom ☒ Drug ☒ Enzyme ☒ Compound All Clear

Pivot Concepts: Gene/Protein Target Concepts: Disease Find Indirectly Associated Concepts

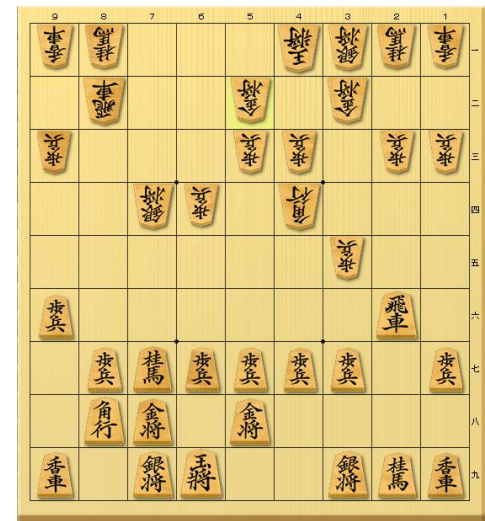
Query **diabetes**  
 293,187 document(s) hit in 18,751,004 MEDLINE articles (0.30 seconds)

Concepts found in the documents ranked by [Frequency](#) | [Printable Mutual Information](#) | [Symmetric Conditional Probability](#)

Human	Disease	Symptom	Drug	Enzyme
insulin 91140	diabetes 277747	overweight 36301	Glucagon 5029	angiotensin converting enzyme 2140
hypertension 37740	hypertension 37740	pain 3448	Nitric Oxide 2707	
hemoglobin 7590	type 2 diabetes 26789	anxiety 2887	Albumin 1723	nerve 2120
glucagon 6704	diabetes 26789	Anesthetics 1382	Angiotensin II 1612	superoxide dismutase 1223
HDL 4500	diabetes 26789	polyuria 1302	Gluathione 1334	aldose reductase 1066
albumin 3702	type 1 diabetes 18803	diarrhea 1040	Nitrosophosphine 1040	protein kinase G 1066
polymerase 3100	peripheral vascular disease 16539	intoxication 916	AD 1192	ATPase 954
MED 3100	diabetes 16539	pediatric 784	Fluorescein 951	carotenes 840
HLA 3100	hyperglycemia 16195	vomiting 620	Fructose 950	nitrogen activated protein kinase 774
LDL 2947	diabetes 16195	abuse 624	Alcaine 847	glucokinase 756
Connective tissue protein 2239	diabetes 16195	abdominal pain 619	Asthma 768	protein kinase 756
collagen 2239	Kidney Disease 11836	chest pain 619	Vitamin E 768	glucose 691
cytokine 2208	Coronary Disease 11012	heartburn 569	Insulin 763	glucose 691
ATP 2140	diabetes 10568	fatigue 591	Fructose 733	glucose 691
growth 2100	diabetes 10568	nausea 591	Amoxicillin 675	glucose 691

## ゲームAI

コンピュータ将棋の思考エンジン etc.



# ソフトウェア2

- C言語入門
  - 構造体
  - ファイル入出力
- 応用(アプリケーション)
  - 数値計算
  - シミュレーション
  - 探索・最適化

# 連絡用ページ

- URL

<http://www.logos.t.u-tokyo.ac.jp/~tsuruoka/lecture/software2/>

ユーザ名: soft2

パスワード: ee2012

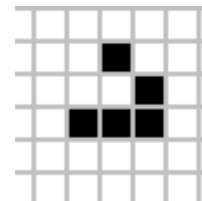
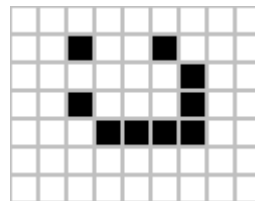
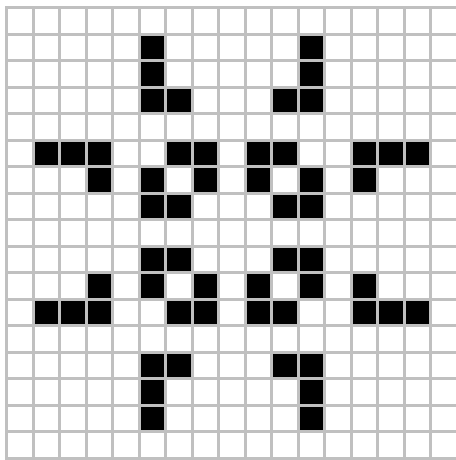
- 資料

- 講義スライド (ppt, pdf)

- サンプルプログラム

# ライフゲーム

- Conway's Game of Life (Conway 1970)
  - イギリスの数学者 John Conway が提案
  - 生命の誕生・衰亡・変化のシミュレーション



# ライフゲーム

- ルール

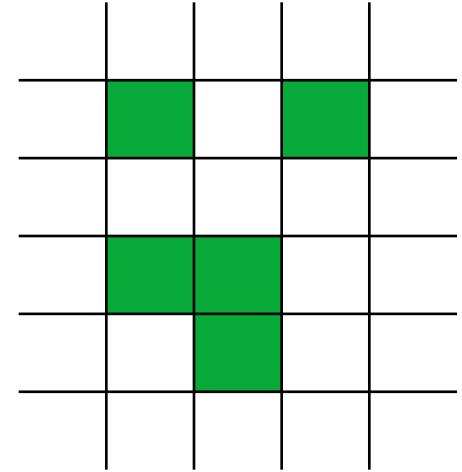
- 2次元グリッド上にセル(cell)を配置
- 次の世代

- セルが有る場所

- 周囲にセルが2個または3個存在するならばそのまま
    - そうでなければセルは消滅

- セルが無い場所

- 周囲にセルが3個存在するならば新しいセルが誕生
    - そうでなければそのまま



# サンプルプログラム life.c

- コンパイル & 実行

```
% gcc life.c
```

```
% ./a.out
```

- ターミナルをもうひとつ開く(結果表示用)

```
% tail -f cells.txt
```

ターミナルのサイズをマウスで調整して ----- が左上にくるように



# 冒頭

```
#include <stdio.h>
#include <unistd.h>
```

← printf 等を使うために必要

← sleep を使うために必要

```
#define HEIGHT 50
#define WIDTH 70
```

```
int cell[HEIGHT][WIDTH];
int cell_next[HEIGHT][WIDTH];
```

← セルの有無を保存するために使う配列

← 次世代を計算するために一時的に使う配列

# メイン

```
int main()  
{
```

```
    int gen = 1;
```

```
    FILE *fp;
```

← ファイル構造体へのポインタを宣言

```
    if ((fp = fopen("cells.txt", "w")) == NULL) {  
        fprintf(stderr, "error: cannot open a file.¥n");  
        return 1;  
    }
```

結果出力用のファイルを開く

```
    init_cells();  
    print_cells(fp);
```

セルの初期配置を決めて表示

```
    while (1) {  
        printf("gen = %d¥n", gen++);  
        update_cells();  
        print_cells(fp);  
    }
```

セルの更新・表示を繰り返す

```
}
```

# update\_cells()

- 次世代のセルの状態を計算

```
void update_cells()
{
    int i, j, k, l;

    for (i = 0; i < HEIGHT; i++) {
        for (j = 0; j < WIDTH; j++) {
            int n = 0;
            for (k = i - 1; k <= i + 1; k++) {
                if (k < 0 || k >= HEIGHT) continue;
                for (l = j - 1; l <= j + 1; l++) {
                    if (k == i && l == j) continue;
                    if (l < 0 || l >= WIDTH) continue;
                    n += cell[k][l];
                }
            }
            if (cell[i][j] == 0) {
                if (n == 3)
                    cell_next[i][j] = 1;
                else
                    cell_next[i][j] = 0;
            } else {
                if (n == 2 || n == 3)
                    cell_next[i][j] = 1;
                else
                    cell_next[i][j] = 0;
            }
        }
    }

    for (i = 0; i < HEIGHT; i++) {
        for (j = 0; j < WIDTH; j++) {
            cell[i][j] = cell_next[i][j];
        }
    }
}
```

個々のマスについて、周囲8マスのセルの数を数える

ルールに従って次世代のセルの状態を決める

すべてのマスを次世代の状態に更新

# print\_cells()

- セルを表示

```
void print_cells(FILE *fp)
{
    int i, j;

    fprintf(fp, "-----¥n");

    for (i = 0; i < HEIGHT; i++) {
        for (j = 0; j < WIDTH; j++) {
            char c = (cell[i][j] == 1) ? '#' : ' ';
            fputc(c, fp);
        }
        fputc('¥n', fp);
    }
    fflush(fp);

    sleep(1);
}
```

fp で指定されているファイルに出力

セルが存在するときは # を出力

バッファされている内容を強制出力  
(tail -f での表示が乱れるので)

1秒停止

# ファイル出力

- ファイルアクセス関数など

`FILE`構造体

ファイルアクセスに関する情報を保持する構造体

```
FILE *fopen(char *name, char *mode);
```

ファイルを開く

```
int fprintf(FILE *fp, const char *format);
```

ファイル`fp`へ書式付で文字列を出力

```
int fputc(int c, FILE *fp);
```

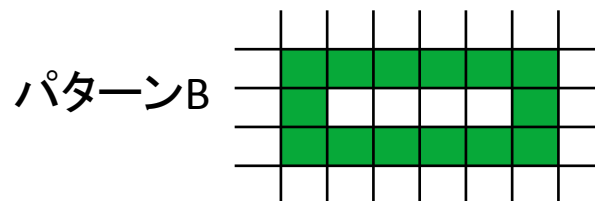
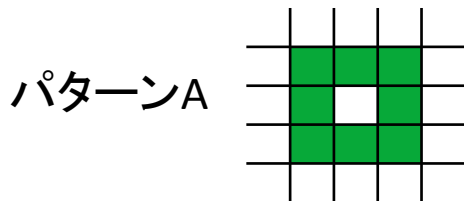
ファイル`fp`へ文字`c`を出力

```
int fclose(FILE *fp);
```

ファイルを閉じる

# 課題(締め切り12/12)

1. 各世代で存在するセルの総数を表示するように“life.c”を拡張せよ
  - プログラムを添付すること(ファイル名は“life1.c”)
  - “life.c”を利用せず自分でゼロからプログラムを作成しても構わない
2. 初期配置パターンをファイルから読み込めるように拡張せよ
  - プログラムを添付すること(ファイル名は“life2.c”)
  - ファイルからの入力に関しては“readfile.c”を参照
3. 中央付近に以下のパターンを配置した場合、最終的にどのように変化するかを確かめ、簡潔に説明せよ



4. [発展課題] 周期性がなく長生きする初期パターン(10セル以内)を自動的に発見できるようにプログラムを拡張せよ
  - プログラムを添付すること(ファイル名は“life3.c”)
  - 発見したパターンとその寿命を簡単に説明すること

# 課題の提出方法

- 宛先
  - [software2@logos.t.u-tokyo.ac.jp](mailto:software2@logos.t.u-tokyo.ac.jp)
- Subject
  - 形式: SOFT-MM-DD-NNNNNNNX
    - MM: 月
    - DD: 日 (授業が行われた日)
    - NNNNNNX: 学籍番号
- 本文
  - 冒頭に学籍番号、氏名を明記

# readfile.c

- ファイル入力のサンプルプログラム
  - ファイルを開く
  - 各行の文字数を表示

- 関数

`fgets(char *s, int size, FILE *fp);`  
ファイルから1行読み込んでバッファに保存

`size_t strlen(const char *s);`  
文字列の長さを取得