平 成 ２３年 度

名古屋大学大学院情報科学研究科
情報システム学専攻
入 学 試 験 問 題

専　　門

平成２２年８月１０日（火）
１２：３０～１５：３０

注 意 事 項

１．試験開始の合図があるまでは、この問題冊子を開いてはならない。

２．試験終了まで退出できない。

３．外国人留学生は、和英辞書などの辞書１冊に限り使用してよい。
電子辞書の持ち込みは認めない。

４．問題冊子、解答用紙３枚、草稿用紙３枚が配布されていることを確認せよ。

５．問題は(１)解析・線形代数、(２)確率・統計、(３)プログラミング、
(４)計算機理論、(５)ハードウェア、(６)ソフトウェアの６科目がある。
このうち３科目を選択して解答せよ。なお、選択した科目名を解答用紙の
指定欄に記入せよ。

６．解答用紙は指定欄に受験番号を必ず記入せよ。解答用紙に受験者の氏名を
記入してはならない。

７．解答用紙は試験終了後に３枚とも提出せよ。

８．問題冊子、草稿用紙は試験終了後に持ち帰ってよい。

解析・線形代数

（解の導出過程も書くこと）

[1] 次の y に関する微分方程式について、以下の問いに答えよ。

$$y\frac{d^2y}{dx^2} - 2\left(\frac{dy}{dx}\right)^2 - y\frac{dy}{dx} = 0 \tag{1}$$

(a) $y = e^z$ とおき、微分方程式 (1) を z に関する微分方程式に書き換えよ。

(b) $\frac{dz}{dx} = v$ とおき、(a) で得られた微分方程式を v について解け。

(c) 微分方程式 (1) の一般解を求めよ。

[2] 次の条件を満たす実平面上の点 (x, y) からなる曲線 $(0 \leq \theta \leq 2\pi)$ について、以下の問いに答えよ。

$$\begin{cases} x = \theta - \sin\theta \\ y = 1 - \cos\theta \end{cases}$$

(a) 曲線の長さを求めよ。

(b) 曲線と x 軸で囲まれた部分の面積を求めよ。

[3] 次の行列 A について、以下の問いに答えよ。

$$A = \begin{pmatrix} 1 & 0 & -2 \\ 0 & 2 & 0 \\ -2 & 0 & 1 \end{pmatrix}$$

(a) 行列 A の全ての固有値を求めよ。また、それに対応する単位固有ベクトルを求めよ。

(b) 行列 $A^6 - 4A^5 + 2A^4 + 3A^3 - 3A^2 + 8A + 7E$ を求めよ。ただし、E は単位行列である。

(c) 行列 A に対し、$P^{-1}AP$ が対角行列であるような正則行列 P、及び、その逆行列 P^{-1} を求めよ。

Translation of technical terms

微分方程式	differential equation	単位固有ベクトル	unit eigenvector
一般解	general solution	単位行列	identity matrix
実平面	real plane	対角行列	diagonal matrix
曲線	curve	正則行列	regular matrix
行列	matrix	逆行列	inverse matrix
固有値	eigenvalue		

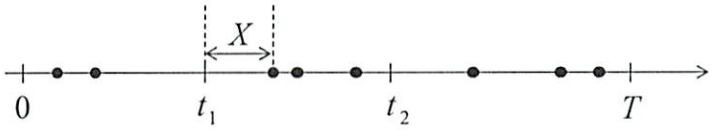
確率・統計（[３]，[４]については，解の導出過程も書くこと.）

［１］ 次の【 ① 】～【 ③ 】に入れる適切な式または語句を解答用紙に書きなさい.

互いに独立な３個の確率変数 X_1, X_2, X_3 のすべてが平均 μ，分散 σ^2 の正規分布に従うとする.

この時，確率変数 $X = X_1 + X_2 + X_3$ は，平均【 ① 】，分散【 ② 】の【 ③ 】分布に従う.

［２］ 区間 $[0,T]$ に n 個の点をランダムに置いたとき，次の【 ④ 】～【 ⑧ 】に入れる適切な式を解答用紙に書きなさい.



(1) 長さ $t_a (= t_2 - t_1 \geq 0)$ の区間 $[t_1, t_2]$ に含まれる点が k 個 $(k \leq n)$ となる事象 A_k の確率を求めたい（図参照）．これは次のような「繰り返し試行の問題」として考えることができる．１回の試行により区間 $[0,T]$ に点を一つ置くことを考えると，「その点が区間 $[t_1, t_2]$ にある」という事象 B の確率は，$P(B) = p =$【 ④ 】となる．この試行を n 回繰り返したとき，「事象 B が k 回生じる」という事象は，n 個の点のうち区間 $[t_1, t_2]$ に k 個あることを意味するので，事象 A_k の確率を p を用いて表すと，$P(A_k) =$【 ⑤ 】となる．ここで $n \gg 1$ および $t_a \ll T$ を仮定すると，

$$P(A_k) \approx e^{-\lambda t_a} \frac{(\lambda t_a)^k}{k!}, \quad \lambda = \frac{n}{T} \quad \text{となる.}$$

(2) 固定点 t_1 から，その右にある最初の点までの距離を確率変数 X で表す（図参照）．ここで $n \gg 1$ を仮定する．確率変数 X の分布関数を $F(x)$ としたとき，$F(x)$ は事象 $\{X \leq x\}$ の確率である．事象 $\{X \leq x\}$ は区間 $[t_1, t_1 + x]$ に少なくとも１つの点があることを意味する．よって区間 $[t_1, t_1 + x]$ に一つも点がない確率 p_0 は $F(x)$ を用いて【 ⑥ 】と表せる．一方，この区間の長さが x なので，p_0 は n と T を用いて【 ⑦ 】となる．そのため，確率密度関数 $f(x)$ は【 ⑧ 】となる.

［３］ e^{λ} のテイラー展開 $e^{\lambda} = \sum_{k=0}^{\infty} \frac{\lambda^k}{k!}$ を用いて，パラメータ λ のポアソン分布

$$f(k) = e^{-\lambda} \frac{\lambda^k}{k!}$$ の平均 μ と分散 σ^2 を求めなさい.

［４］ 確率変数 X と Y は独立で，それぞれ次の分布

$$f(x) = \begin{cases} e^{-x} & (x \geq 0) \\ 0 & (otherwise) \end{cases}, \quad g(y) = \begin{cases} e^{-y} & (y \geq 0) \\ 0 & (otherwise) \end{cases}$$

に従う時，確率変数 $Z = X + 2Y$ の確率密度関数を求めなさい.

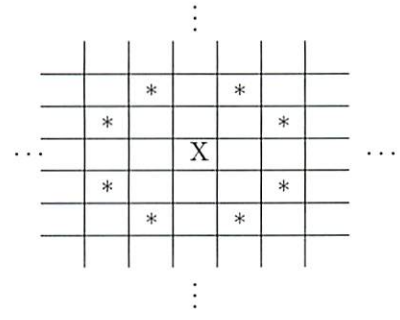
【専門用語の英訳】

独立な independent, 確率変数 random variable, 平均 mean, 分散 variance,
正規分布 normal distribution, 区間 interval, 事象 event, 試行 trial,
分布関数 distribution function, 確率密度関数 probability density function,
テイラー展開 Taylor expansion, ポアソン分布 Poisson distribution

プログラミング

次の問題を騎士巡回問題（knight tour problem）という．

$N \times N$ のチェス盤（N は正整数）が与えられているとする．一つの騎士が，指定された出発点から，どの枡目（square）も 2 度以上訪れることなく，すべての枡目を訪れる移動経路があればその一つを発見せよ．なければないと答えよ．



ただし，右の図において，騎士は現在の枡目 X から $*$ のある枡目だけに移動できる．

アルゴリズム 1 は，騎士巡回問題を解くための再帰（recursive）アルゴリズムである．リスト 1 は，アルゴリズム 1 を実装する C プログラムであり，N と出発点は 2,3,4 行の DEFINE 文で与えられている．ただし，[1-1]～[1-8]，[2]～[4]の部分は未完成である．なお，リスト 1 の行頭の数字は行番号を表し，プログラムには含まれない．次の問いに答えよ．

(1) リスト 1 の 17 行～20 行の [1-1] ～[1-8] に適切な値を入れよ．

(2) リスト 1 の 31 行の if 文の条件 [2] を 80 文字以内で適切に与えよ．

(3) リスト 1 の 33 行の if 文の条件 [3] を 10 文字以内で適切に与えよ．

(4) リスト 1 の 35 行がアルゴリズム 1 の 18 行に対応するように，[4] を 20 文字以内で適切に与えよ．

(5) リスト 1 の 6 行で宣言（declare）されている変数（variable）`board` の役割を述べよ．

(6) リスト 1 の `try` の 3 つの仮引数（formal parameter）の役割を述べよ．

(7) 完成したリスト 1 のプログラムを実行したとき，`try` の最初の 5 回の呼出のそれぞれについて，実引数（actual parameter）の値を示せ．また，5 回目の呼出の直前の `board` の値を 5 行 5 列の表の形で示せ．

(8) 指定された出発点からのすべての移動経路を求めるプログラムは，リスト 1 の 33 行と 34 行のみを適切に変更することで得られる．変更後の 33 行と 34 行をそれぞれ 80 文字以内で与えよ．

アルゴリズム 1 騎士巡回問題アルゴリズム

```
各種の初期化（initialization）をする
出発点の枡目を指定して try を呼び出す
印刷された移動経路がなければ「ない」を出力する
procedure try
    本手続き（procedure）に必要な初期化を行う
    for each 騎士の移動 do
        if 移動後の枡目がチェス盤の上にあり，かつ，まだ訪問していない then
            移動後の枡目をそれまでの移動経路に追加する
            if すべてを訪問した then
                移動経路を印刷する
                成功を返す
            else
                現在位置を移動先に移して，try を再帰呼び出しする
                if 再帰呼び出しの結果が成功 then
                    成功を返す
                end if
            end if
            最後に追加された枡目を移動経路から取り除く
        end if
    do end
    失敗を返す
end procedure
```

リスト 1: find-one-knight-tour.c

```
#include <stdio.h>
#define N 5
#define STARTROW 0
#define STARTCOL 2

int board[N][N], moverow[8], movecol[8], nsols;
int try(int,int,int);
void printout();

main() {
  int i,j;
  nsols = 0;
```

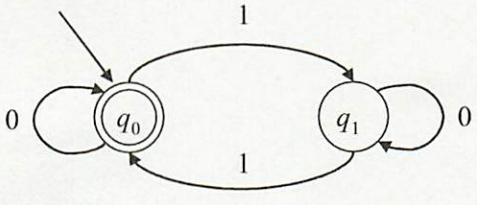
```
13    moverow[0]= 2; movecol[0]= 1;
14    moverow[1]= 1; movecol[1]= 2;
15    moverow[2]=-1; movecol[2]= 2;
16    moverow[3]=-2; movecol[3]= 1;
17    moverow[4]= [1-1] ; movecol[4]= [1-2] ;
18    moverow[5]= [1-3] ; movecol[5]= [1-4] ;
19    moverow[6]= [1-5] ; movecol[6]= [1-6] ;
20    moverow[7]= [1-7] ; movecol[7]= [1-8] ;
21    for (i=0; i<N; i++) for (j=0; j<N; j++) board[i][j] = 0;
22    board[STARTROW][STARTCOL] = 1;
23    try(2,STARTROW,STARTCOL);
24    if (nsols==0) puts("No solution.");
25  }
26
27  int try(int i, int x, int y) {
28    int u,v,k;
29    for (k=0; k<8; k++) {
30      u = x + moverow[k]; v = y + movecol[k];
31      if ( [2] ) {
32          board[u][v] = i;
33          if ( [3] ) { nsols++; printout(); return 1; }
34          else { if (try(i+1,u,v)==1) return 1; }
35          [4] ;
36      }
37    }
38    return 0;
39  }
40
41  void printout() {
42    int i,j;
43    printf("-- result %d --\n", nsols);
44    for (i=0; i<N; i++) {
45      for (j=0; j<N; j++) printf("%2d ", board[i][j]);
46      puts("");
47    }
48  }
```

計算機理論

[1] 入力アルファベット (input alphabet) $\Sigma = \{0, 1\}$ 上の言語 (language) L_1, L_2 を以下のように定める.

L_1 : 0 を含まないか, または, 0 を 3 の倍数個だけ含むすべての文字列からなる集合

L_2 : 次の状態遷移図 (state transition graph) で表される決定性有限オートマトン (deterministic finite automaton) が受理 (accept) する言語



次に, Σ に対する正規表現 (regular expression) E を以下の拡張BNF (extended BNF) で定義する.

$$E ::= \varepsilon \mid \emptyset \mid 0 \mid 1 \mid E \bullet E \mid E + E \mid (E) \mid E^*$$

ただし, ε は空列 (empty string), $\emptyset$ は空集合を表す. また, $\bullet$ は連接 (concatenation), $+$ は和 (set union), * はクリーネ閉包 (Kleene closure) を表す演算子 (operator) であり, 演算子の結合の強さは, 強い方からクリーネ閉包, 連接, 和の順とする. なお, 連接演算子および省略しても演算順序に影響がない括弧 (parenthesis) は省略してよい. 省略した記号は正規表現の長さに含めない.

このとき, 以下の問いに答えよ.

(1) L_1 を受理する決定性有限オートマトンの状態遷移図を示せ. ただし, 状態数は 3 以下とせよ.

(2) L_2 を長さ 9 以内の正規表現で表せ. なお, その長さも書け.

(3) $L_1 \cap L_2$ を受理する決定性有限オートマトンの状態遷移図を示せ.

［2］R，Sを集合A上の2項関係 (binary relation) とし，RとSの合成 (composition) $S \circ R$を

$$S \circ R = \{(x, z) \in A^2 \mid y \in A \text{が存在して}, (x, y) \in R \text{かつ} (y, z) \in S\}$$

と定義する．このとき，A上の2項関係R，S，Tに対して，合成に関する結合則 (associative law)

$$(S \circ R) \circ T = S \circ (R \circ T)$$

が成り立つことが知られている．
さらに，R^n $(n \geq 1)$，R^+ をそれぞれ以下のように定義する．

$$R^1 = R,$$
$$R^n = R^{n-1} \circ R \ (n \geq 2),$$
$$R^+ = \bigcup_{n=1}^{\infty} R^n.$$

このとき，以下の命題を証明せよ．

(1) Rが推移的 (transitive) であるとき，かつそのときに限り，$R^2 \subseteq R$.

(2) 任意の自然数n (≥ 2)とk $(1 \leq k \leq n-1)$に対して，$R^n = R^{n-k} \circ R^k$.

(3) R^+は推移的である．

ハードウェア

[1] 系列 110 および 1010 を検出する系列検出器 (sequence detector) で、下記の条件を満たすものを考える。

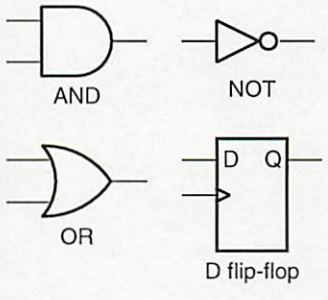
- クロック信号 (clock signal) CLK に同期 (synchronous) して動作する順序回路 (sequential circuit) である。
- 必要最小個数 (minimum number) の D フリップフロップ (D flip-flop) を記憶素子として用いる。
- リセット状態 (reset state) での D フリップフロップの出力値は 0 である。
- 1 ビット (bit) の入力 X および 1 ビットの出力 Z を有する。
- X に 1 クロックサイクル (clock cycle) 毎に入力される論理値が順に $1,1,0$ または $1,0,1,0$ であったとき、次の 1 クロックサイクルの間、Z に 1 を出力する。他の場合は Z に 0 を出力する。
- 重複する系列 (overlapping sequence) も検出する。例えば系列 1101010 が入力された場合、1 クロック目からの 110、2 クロック目からの 1010、4 クロック目からの 1010 の 3 つの系列を検出する。

以下の各問に答えよ。

(1) この系列検出器を ミーリ型順序機械 (Mealy machine) として設計し、最小化した状態遷移図 (state transition graph) を示せ。

(2) (1) で設計した順序機械に状態割当 (state assignment) を行い、状態変数関数 (state variable function) および出力変数関数 (output variable function) を積和形 (sum-of-products form) 論理式で示せ。ただし、これらの論理式のリテラル (literal) 数の総和が最小になるように設計せよ。

(3) (2) で設計した関数を実現する回路の回路図 (schematic diagram) を示せ。ただし、論理ゲートの種類は AND ゲート、OR ゲート、および NOT ゲートのみとし、回路記号は右記のものを使用せよ。



[2] 浮動小数点数 (floating point number) について以下の各問に答えよ。

(1) 基数 (radix) r、符号 (sign) s (正数: 0、負数: 1)、指数 (exponent) e (符号付き整数 (signed integer))、仮数 (mantissa) m (符号無し数 (unsigned number)) の浮動小数点数で表現される値を式で記述せよ。

(2) 浮動小数点数の正規化 (normalization)、浮動小数点演算におけるオーバーフロー (overflow)、アンダーフロー (underflow) について、それぞれ説明せよ。

(3) IEEE754 規格の 2 進単精度 (binary single-precision) 浮動小数点形式は、以下のフィールド (fields) からなる。

- 符号: 1 ビット
- 指数部: 8 ビット、ゲタ (bias) 127 のゲタばき表現 (biased representation)
- 仮数部: 23 ビット、最上位の 1 ビットを除いた固定小数点数 (fixed point number)

この形式による 10 進数値 88.0 の表現を 32 ビット 2 進表現で記述せよ。

ソフトウェア

［1］ソフトウェア設計に関する以下の問いに答えよ。

(1) ソフトウェアのモジュール化設計（modular design）において，インタフェース（interface）と実装（implementaton）を分離することの利点について，具体的なプログラムの例を示して説明せよ。解答は，プログラムの例を除いて400字（英語で記述する場合は130語）程度以下で記述せよ。

(2) ソフトウェアのオブジェクト指向設計（object-oriented design）におけるデータ隠蔽（data hiding）について，具体的なプログラムの例を示して説明せよ。また，その利点について説明せよ。解答は，プログラムの例を除いて400字（英語で記述する場合は130語）程度以下で記述せよ。

［2］複数のプロセス（process）が並行（concurrent）に実行される場合のプロセス間同期（inter-process synchronization）の実現に関する以下の問いに答えよ。

なお，セマフォ（semaphore）とは，プロセス間同期のためにオペレーティングシステムが提供する機能であり，セマフォSに対する操作として，以下に説明する`P(S)`と`V(S)`があるものとする。

`P(S)`

セマフォSの値が0でなければ，1だけ減算する。0の場合には，呼び出したプロセスを，セマフォSに対する待ち状態（waiting state）とする。これらの処理はアトミック（atomic）に実行される。

`V(S)`

セマフォSに対する待ち状態のプロセスがあれば，その内の1つを待ち解除する（release from waiting）。待ち状態のプロセスがない場合には，セマフォSの値を1だけ加算する。これらの処理はアトミックに実行される。

(1) 並行に実行される複数のプロセスが，以下に示す関数（function）`func1`を呼び出す場合に，複数のプロセスが同時に`access_resource`を実行する可能性がある。どのような場合に，複数のプロセスが同時に`access_resource`を実行するか。具体的な実行シーケンスの例を示して説明せよ。なお，変数（variable）`flag`は，すべてのプロセスがアクセスできる`int`型の共有変数（shared variable）であり，0に初期化されているものとする。

```
void func1()
{
    while (flag != 0) {
    }
    flag = 1;
    access_resource();
    flag = 0;
}
```

(2) 並行に実行される複数のプロセスが，以下に示す関数func2を呼び出す場合に，複数のプロセスが同時にaccess_resourceを実行しないように排他制御（exclusive control）したい。初期値1のセマフォSを用いてこの排他制御を実現するには，どのようなセマフォ操作を行えばよいか。[A]および[B]の空欄を埋めてプログラムを完成させよ。

```
void func2()
{
    [ A ]
    access_resource();
    [ B ]
}
```

(3) プロセスAがバッファ（buffer）に文字を書き込むための関数funcW1を呼び出し，それと並行に実行されるプロセスBがバッファから文字を読み出すための関数funcR1を呼び出す場合を考える。バッファが空の場合には，プロセスBがread_from_bufferを呼び出さないようにしたい。初期値0のセマフォS1を用いてこのプロセス間同期を実現するには，どのようなセマフォ操作を行えばよいか。[C]～[F]の空欄を埋めてプログラムを完成させよ。なお，初期状態においては，バッファは空であるものとする。また，空欄に何も入らない場合には，なし（none）と答えよ。

```
void funcW1(char c)
{
    [ C ]
    write_to_buffer(c);
    [ D ]
}

void funcR1(char *c)
{
    [ E ]
    read_from_buffer(c);
    [ F ]
}
```

(4) 前問のプロセス間同期に加えて，バッファに書き込まれる文字が最大10文字になるようにしたい。セマフォを用いてこのプロセス間同期を実現するには，どのようなセマフォ操作を行えばよいか。用いるセマフォの初期値を答え，[C]～[F]の空欄を埋めてプログラムを完成させよ。空欄に何も入らない場合には，なし（none）と答えよ。なお，初期状態においては，バッファは空であるものとする。また，用いるセマフォは1つとは限らないことに留意せよ。

(5) プロセスAが変数shared_varに値を書き込むための関数funcW2を呼び出し，それと並行に実行されるプロセスBが変数shared_varの値を読み出すための関数funcR2を呼び出す場合を考える。プロセスAが変数shared_varに値を書き込んだ後に，プロセスBが変数shared_varの値を読み出すようにしたい。セマフォを用いてこのプロセス間同期を実現するには，どのようなセマフォ操作を行えばよいか。用いるセマフォの初期値を答え，[G]～[J]の空欄を埋めてプログラムを完成させよ。空欄に何も入らない場合には，なし（none）と答えよ。なお，変数shared_varは，両方のプロセスがアクセスできるint型の共有変数であるものとする。また，用いるセマフォは1つとは限らないことに留意せよ。

```
void funcW2(int x)
{
    [ G ]
    shared_var = x;
    [ H ]
}

void funcR2(int *x)
{
    [ I ]
    *x = shared_var;
    [ J ]
}
```