平成１８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１７年８月９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、日本語から母語への辞書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４枚、草稿用紙４枚が配布されていることを確認せよ。
５．問題は、A、Bの各科目群について下記のように解答せよ（合計４科目を選択して解答せよ）。
A群：次の３科目から２科目を選択して解答せよ。
(1) 解析・線形代数　(2) 確率・統計　(3) プログラミング
B群：次の３科目から２科目を選択して解答せよ。
(1) 計算機理論　(2) ハードウェア　(3) ソフトウェア
なお、選択した科目名を解答用紙の指定欄に記入せよ。
B群において計算機理論を選択する場合は、計算機理論Iと計算機理論IIのいずれか一方を選択して解答せよ。計算機理論Iと計算機理論IIの両方を選択することはできない。解答用紙の指定欄には計算機理論Iまたは計算機理論IIと記入せよ。
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４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] 微分方程式 $f'''(t)+af''(t)+bf'(t)+cf(t)=0$ を考える。ただし、$f'(t)=\dfrac{df(t)}{dt}$ を表す。

(1)

(a) $\mathbf{x}=(f(t),f'(t),f''(t))^T$ とした時、$\dfrac{d\mathbf{x}}{dt}=\mathbf{A}\mathbf{x}$ を満たす行列 $\mathbf{A}$ を求めよ。ただし、記号 ^{T} は行列の転置を表す。

(b) 行列 $\mathbf{A}$ が対角化可能であるとき、適当な正則行列を $\mathbf{P}$、行列 $\mathbf{A}$ を対角化した行列を $\mathbf{D}$ として $\mathbf{D}=\mathbf{P}^{-1}\mathbf{A}\mathbf{P}$ が成り立つものとする。$\mathbf{y}=\mathbf{P}^{-1}\mathbf{x}$ の変換により $\dfrac{d\mathbf{x}}{dt}=\mathbf{A}\mathbf{x}$ は $\dfrac{d\mathbf{y}}{dt}=\mathbf{D}\mathbf{y}$ と書けることを示せ。

(c) 行列 $\mathbf{A}$ の相異なる固有値を $\lambda_1,\lambda_2,\lambda_3$ とする時、$\dfrac{d\mathbf{y}}{dt}=\mathbf{D}\mathbf{y}$ の一般解を求めよ。

(2)

(a) 微分方程式 $f'''(t)+4f''(t)-f'(t)-4f(t)=0$ の一般解を求めよ。

(b) (a)の微分方程式で初期条件 $f(0)=4$，$f'(0)=-5$，$f''(0)=19$ を満たす解を求めよ。

[2] 次の2重積分を求めることを考える（$a>0,\ b>0$）。

$$\iint_D (x^2+y^2)dxdy \qquad \text{ただし } D:\frac{x^2}{a^2}+\frac{y^2}{b^2}\le 1$$

(1) $x=ar\cos\theta$，$y=br\sin\theta$ として積分式を変換せよ。

(2) 2 重積分を求めよ。

注）

微分方程式：differential equation, 転置：transpose, 対角化：diagonalization
対角化可能：diagonalizable, 固有値：eigen value, 一般解：general solution
初期条件：initial condition, 2重積分：double integral

確率・統計（解の導出過程を書くこと）

[1]あるシステムの故障時間（運用を開始後，故障発生までの時間）を確率変数Tとし，その確率密度関数$f_T(t)$は式(1)で与えられるとする．システムが故障したら，同じ性能の新システムに交換して運用する．故障の発生は互いに独立で，システムの交換に要する時間は無視できるとする．下記の問いに答えよ．

$$f_T(t)=\begin{cases} e^{-t} & t\geq 0 \\ 0 & t<0 \end{cases} \qquad (1)$$

(1) このシステムの平均故障時間$E[T]$を求めよ．

(2) 時刻0でシステムの運用を開始し，時刻tまで故障しない確率$Pr\{T>t\}$を求めよ．

(3) 時刻0でシステムの運用を開始し，2回目の故障までの時間を確率変数Sとする．確率変数Sの確率密度関数$g_S(s)$を求めよ．

(4) 時刻0でシステムの運用を開始し，時刻tまでにn回故障する確率を$P_n(t)$とする．$P_n(t)$と$P_{n-1}(t)$との間に成立する式（積分が残っていても良い）を求めよ．

(5) 設問(4)の$P_n(t)$で，$n=2$の確率$P_2(t)$を求めよ．

[2] 下記の問いに答えよ．

(1) 仮説検定における，(a)帰無仮説，(b)有意水準αを説明せよ．

(2) 中心極限定理を説明せよ．

(3) 確率変数X, Yが互いに独立であるとき，同時確率密度関数$f_{XY}(x, y)$について成立する式を示せ．

[専門用語の英訳]

確率変数 random variable
確率密度関数 probability density function
故障時間 time to failure
独立 independence
平均 mean
仮説検定 hypothesis testing
帰無仮説 null hypothesis
有意水準 level of significance
中心極限定理 central limit theorem
同時確率密度関数 joint probability density function

プログラミング

最後にある図のように定められるプログラミング言語 subsetC でのプログラミングに関する以下の問に答えよ.

[1] 次の関数定義に関する問 1), 2) に答えよ.

```
int f(int a){ a := a + 1;
              { int b; b := 2;
                { int a; a := 3; b := a + b + 4; }
                return a + b;
              }
            }
```

1) 式 `f(5)` を評価したときの値を答えよ.

2) 変数の名前替えだけが許されるという条件の下で, 関数 f の働き (すなわち, 引数 (argument) と戻り値 (return value) の関係) を変えずに, 上記の関数定義の 3 行目

```
{ int a; a := 3; b := a + b + 4; }
```

の下線部分を `int b;` に変更したい. 必要最小限の変数の名前替えを施した関数定義を書け.

[2] subsetC には含まれない構文として, 次のような repeat 文を考える.

repeat ⟨ブロック文⟩ until (⟨論理式⟩)

repeat 文の実行を次のように定める.

> まず⟨ブロック文⟩を 1 回実行し, その後, ⟨論理式⟩が成り立たない間は⟨ブロック文⟩を繰り返し実行し, 成り立ったらこの文の実行を終る.

次のブロック文を subsetC の範囲で書き直せ.

```
{int i; i := 0;
 repeat {i := g(i,y); x := h(x);} until (x <= 0)
 z := i; }
```

ただし, 変数 x, y, z はこのブロックを通用範囲に含む位置で宣言されており, また, 関数 g, h の定義は, subsetC の関数定義として別途与えられているとする.

［3］次の関数定義を考える．

```
int f(int x)
 {if (x > 0) {return x * f(x-1);} else {return 1;} }
```

この関数 f と働き（すなわち，引数と戻り値の関係）が同じで再帰呼出 (recursive call) を使わない関数 g を subsetC で定義せよ．ただし，オーバーフロー (overflow) については考慮しなくてよい．

［4］次の if 文を実行すると，それぞれ，どのような状況になるかを述べよ．

1) `if (0 == 0) { y := 1; } else { y := loop(0); }`

2) `if (1 == 0) { y := 1; } else { y := loop(0); }`

3) `if (loop(0) == 0) { y := 1; } else { y := 0; }`

ただし，関数 loop の定義は次の通りとする．

```
int loop(int x) { while (x == x) { x := -x; }  return 0; }
```

subsetC のプログラムは，以下の変数宣言 (variable declaration) および関数定義 (function definition) を並べたものである．subsetC で使えるデータ型 (data type) は整数型（int 型）だけとする．subsetC は，代入文 (assignment) で= の代わりに:=を使っていることを除いて C 言語のサブセットであり，各構文の意味は C 言語のそれと同じとする．変数宣言の通用範囲（scope）も C 言語のそれと同じで，ブロック文 (block statement) の先頭にある変数宣言の通用範囲はそのブロック文とする．

- ⟨変数宣言⟩の構文は次の通りである．

  構文　`int` ⟨変数⟩ `;`
  例　　`int x;`

- ⟨関数定義⟩の構文は次の通りである．

  構文　`int` ⟨関数名⟩`(` `int` ⟨変数⟩ `, ... , int` ⟨変数⟩ `)` ⟨ブロック文⟩
  例　　
  ```
  int max(int x,int y)
    {int z; if (x > y){z := x;} else {z := y;} return z;}
  ```

- ⟨文⟩は次の 5 つからなる．

  - ⟨ブロック文⟩
    構文　`{`⟨変数宣言⟩`···`⟨変数宣言⟩⟨文⟩`···`⟨文⟩`}`
    例　　`{int i; x := x + i; y := y - i;}`
    例　　`{x := x + i;}`
  - ⟨代入文⟩
    構文　⟨変数⟩ `:=` ⟨式⟩ `;`
    例　　`x := y + z;`
  - ⟨if 文⟩
    構文　`if (` ⟨論理式⟩ `)`⟨ブロック文⟩ `else` ⟨ブロック文⟩
    例　　`if (x > y) {z := x - y;} else {z := y - x;}`
  - ⟨while 文⟩
    構文　`while (`⟨論理式⟩`)`⟨ブロック文⟩
    例　　`while (x > 0){ z := z + y; x := x - 1;}`
  - ⟨return 文⟩
    構文　`return` ⟨式⟩`;`
    例　　`return x * y;`

- ⟨式⟩の構文は次の通りである．

  構文　⟨変数⟩ または ⟨定数⟩ または ⟨関数名⟩`(`⟨式⟩`, ... ,`⟨式⟩`)` または
  ⟨式⟩⟨算術演算子⟩⟨式⟩ または `(`⟨式⟩`)`
  ただし，⟨算術演算子⟩は `+, -, *, /` である．
  例　　`3*x + y - max(x,y)`

- ⟨論理式⟩の構文は次の通りである．

  構文　⟨式⟩⟨関係演算子⟩⟨式⟩ または ⟨論理式⟩`&&`⟨論理式⟩ または
  ⟨論理式⟩`||`⟨論理式⟩ または `!(`⟨論理式⟩`)` または `(`⟨論理式⟩`)`
  ただし，⟨関係演算子⟩は `==, != ,>, <, >=, <=` である．
  例　　`(x > y || x == y) && !(y*y < y)`

図：プログラミング言語 subsetC

計算機理論 I

計算機理論 I を選択する場合には，解答用紙の指定欄に計算機理論 I と記入せよ．なお，計算機理論 I と計算機理論 II の両方を選択することはできない．

記号 (symbol) の有限集合 Σ が与えられるとき，(Σ は $\{|, *, \emptyset, (,)\}$ と互いに素 (disjoint) であるとする) Σ 上の正則表現 (regular expression) は，以下で帰納的に定義される $\Sigma \cup \{|, *, \emptyset, (,)\}$ 上の語 (word) である．正則表現 e の表現する言語 (language) を $L(e)$ と書く．

(1) $\emptyset$, ε および Σ の要素は正則表現である．$L(\emptyset)$ は空集合，$L(\varepsilon) = \{\varepsilon\}$，$a \in \Sigma$ に対して $L(a) = \{a\}$ である．
(2) e_1 および e_2 が正則表現であるとき，$(e_1|e_2)$ および (e_1e_2) は正則表現である．$L((e_1|e_2)) = L(e_1) \cup L(e_2)$，$L((e_1e_2)) = L(e_1) \cdot L(e_2)$ である．
(3) e が正則表現であるとき，(e^*) は正則表現である．$L((e^*)) = (L(e))^*$ である．

ここで，ε は空列 (empty string)，$L_1 \cdot L_2$ は言語 L_1 と言語 L_2 の連接 (concatenation) を表し，L^* は，言語 L の Kleene 閉包 (Kleenean closure) を表す．正則表現の括弧 (parentheses) は表現する言語があいまいでない (unambiguous) 場合は省略 (omit) する．

言語 $L_s = \{a^n b^n | n \geq 0\}$ は正則表現では表現できないことを以下のような事実に基づいて示したい．ここで，正則表現 e と自然数 $n \geq 0$ に対して e^n は以下のように定義される．

$$e^n = \begin{cases} \varepsilon & n = 0 \text{ のとき} \\ ee^{n-1} & n > 0 \text{ のとき} \end{cases}$$

事実 1 正則表現が表す言語は有限オートマトン (finite automaton) で受理 (accept) できる．

事実 2 L_s を受理する有限オートマトン A_s が存在するとすれば，A_s は L_s に含まれない語も受理してしまうことになり矛盾する．

結論 したがって，L_s を表現する正則表現は存在しない．

事実 1 に関して以下の問いに答えよ.

1) 正則表現から ε-動作 (ε-moves) を含む有限オートマトンを系統的 (systematic) に生成する手法を述べよ.
2) 正則表現 $a((ab)^*|(ba)^*)b$ が表す言語を受理する状態数最小 (minimalized) の決定性有限オートマトン (deterministic finite automaton) を求めよ.

次に事実 2 の証明を与える. L_s を受理するオートマトン A_s を $(Q_s, \{a,b\}, \delta, q_0, F_s)$ とする. $a^n b^n \in L_s$ に対して, A_s の状態の系列を含めた系列を受理系列 (acceptance sequence) とよび, 次のように書くことにする.

$$q_0\sigma_1 q_1\sigma_2 q_2 \cdots q_{2n-1}\sigma_{2n} q_{2n}$$

ここで, $\sigma_i \in \{a,b\}$, $q_i \in Q_s$ $(0 \leq i \leq 2n)$, $q_{j+1} = \delta(q_j, \sigma_{j+1})$ $(0 \leq j \leq 2n-1)$, $q_{2n} \in F_s$ とする.

事実 2 に関して以下の問いに答えよ.

3) L_s は次の条件を満たす語 $a^m b^m$ を必ず含むことを示せ.

   $a^m b^m$ に対する受理系列 $q_0\sigma_1 q_1\sigma_2 q_2 \cdots q_{2m-1}\sigma_{2m} q_{2m}$ において,
   適当な k および l が存在して, $q_k = q_l$, $0 \leq k < l \leq 2m$ である.

4) A_s が 3) のような語を受理するとすれば, L_s に属さない語も受理することを示せ.

計算機理論 II

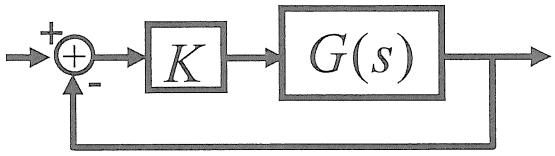
計算機理論 II を選択する場合には，解答用紙の指定欄に計算機理論 II と記入せよ．なお，計算機理論 I と計算機理論 II の両方を選択することはできない．

[1] 下式の微分方程式で表される制御対象について，下記の質問に答えよ．
なお，解答に際して解の導出過程も明記すること．

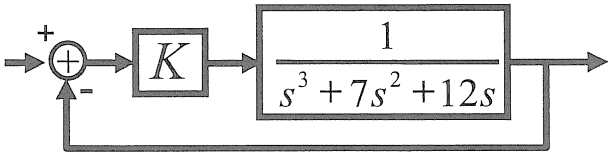
$$\frac{d^3 v_o(t)}{dt^3} + 2\frac{d^2 v_o(t)}{dt^2} + \frac{dv_o(t)}{dt} = v_i(t)$$

1) この制御対象の伝達関数 $G(s) = V_o(s)/V_i(s)$ と，その極を求めよ．但し，$V_i(s)$, $V_o(s)$ は，それぞれ，入力 $v_i(t)$，出力 $v_o(t)$ のラプラス変換である．

2) 下図のように閉ループ系を構成し，$K = 1$ とする．この時の閉ループ系の安定性を，ω を 0 から $+\infty$ に変化させた時の開ループ周波数伝達関数，$G(j\omega)K$ のベクトル軌跡（ナイキスト線図）により判別せよ（$\omega = 0, 1, \sqrt{2}, \infty$ について計算することを推奨する）．もし，この閉ループ系が安定ならば，そのゲイン余裕も求めよ．



[2] 下図の制御系について，下記の質問に答えよ．なお，解答に際して解の導出過程も明記すること．



1) 閉ループ系が丁度安定限界となる K の値と，その時の極を求めよ．

2) 閉ループ系は，３次の伝達関数となるが，そのうち２次遅れ部分の減衰係数 ς が，$\sqrt{2}/2$ となる K の値を求めよ．また，その K の値に対する閉ループ系の極と単位インパルス入力を加えた時の応答を求めよ．

Translations of technical terms

微分方程式:differential equation
制御対象:controlled object
伝達関数:transfer function
極:pole
ラプラス変換:Laplace transformation
閉ループ系:closed loop system
安定性:stability
開ループ:open loop
ゲイン余裕:gain margin
周波数伝達関数:frequency domain transfer function
ベクトル軌跡:vector locus
ナイキスト線図:Nyquist diagram
2次遅れ:2-order delay
減衰係数:damping ratio
単位インパルス:unit impulse

ハードウェア

[1] パイプラインプロセッサ (pipeline processor) におけるハザード (hazard) について，発生要因により分類して説明し，各分類のハザードを回避する方法やその影響を軽減する方法について述べよ．

[2] 以下の問に答えよ．

1) 2 つの完全定義順序機械 (completely specified sequential machines) が与えられたとき，これらの等価性 (equivalence) を判定する手順を説明せよ．ただし，初期状態 (initial state) (リセット状態 (reset state)) は指定されていないものとする．

2) 表 1, 2 の状態遷移表 (state transition table) で示される完全定義順序機械 M_1, M_2 が等価であるかどうかを判定せよ．

等価である場合は M_1 と M_2 の状態の対応関係を示し，等価でない場合はその理由を示せ．

表 1: 順序機械 M_1

現状態 S	次状態 S', 出力 z	
	入力 $x=0$	入力 $x=1$
S_{11}	S_{12},0	S_{14},0
S_{12}	S_{13},0	S_{11},0
S_{13}	S_{14},0	S_{12},1
S_{14}	S_{11},1	S_{15},0
S_{15}	S_{14},0	S_{16},1
S_{16}	S_{15},0	S_{11},0

表 2: 順序機械 M_2

現状態 S	次状態 S', 出力 z	
	入力 $x=0$	入力 $x=1$
S_{21}	S_{24},1	S_{22},0
S_{22}	S_{26},0	S_{23},0
S_{23}	S_{22},0	S_{24},1
S_{24}	S_{23},0	S_{27},0
S_{25}	S_{27},0	S_{23},0
S_{26}	S_{24},1	S_{25},0
S_{27}	S_{24},1	S_{22},0

ソフトウェア

［1］　以下に示すC言語プログラム中の関数quick_sortは，n個の整数型の要素を持つ配列aの各要素を，クイックソートアルゴリズムにより，値の小さい順に並べ換えるプログラムである．このプログラムに関する以下の問に答えよ．ただし，配列の添字は0から始まり，配列aの各要素の値はそれぞれ異なるものとする．また，プログラムの左側の数字は，行の番号を示すもので，プログラムの一部ではない．

```
 1:    #define SWAP(x, y) { int t = x; x = y; y = t; }
 2:
 3:    void qsort(int a[], int left, int right)
 4:    {
 5:        int i, j;
 6:        int p;
 7:
 8:        while (left < right) {
 9:            p = a[(left + right) / 2];
10:            i = left;
11:            j = right;
12:            while (1) {
13:                while (i <= right && a[i] < p) {
14:                    i++;
15:                }
16:                while (left <= j && a[j] > p) {
17:                    j--;
18:                }
19:                if (i >= j) {
20:                    break;
21:                }
22:                SWAP(a[i], a[j]);
23:                i++;
24:                j--;
25:            }
26:            qsort(a, left, i - 1);
27:            left = j + 1;
28:        }
29:    }
30:
31:    void quick_sort(int a[], int n)
32:    {
33:        qsort(a, 0, n - 1);
34:    }
```

1）関数quick_sortの最善の場合（best case）と最悪の場合（worst case）のそれぞれにおける実行時間（execution time）を，nのオーダ（order）で答えよ．また，最善の場合と最悪の場合は，9行目でpがどのような値になった場合に生じるか，それぞれ説明せよ．ただし，最善の場合とは実行時間が最も短くなる場合を，最悪の場合とは実行時間が最も長くなる場合のことをいう．

2）関数 quick_sort のメモリ使用量（memory usage）が最も少なくなる場合と最も多くなる場合のそれぞれにおけるメモリ使用量を，n のオーダで答えよ．また，メモリ使用量が最も少なくなる場合と最も多くなる場合は，9 行目で p がどのような値になった場合に生じるか，それぞれ説明せよ．ただし，ここでいうメモリ使用量には，配列 a を置くための領域は含まないものとする（以下の問でも同様）．

3）関数 quick_sort のメモリ使用量が最も多くなる場合の n に対するオーダを改善するためには，プログラムをどのように修正すればよいか．プログラムの修正箇所が少ない場合には，元のプログラムとの差分のみを示してもよい．また，修正したプログラムで，関数 quick_sort のメモリ使用量が最も多くなる場合のメモリ使用量を，n のオーダで答えよ．

［2］　以下の 3 つの用語の内 2 つを選択し，それぞれ 100 文字以内で説明せよ．

1）アプリケーションプログラミングインタフェース（API, application programming interface）

2）共通部分式除去（common subexpression elimination）

3）モジュール強度（module strength）またはモジュール凝集度（module cohesion）

ただし，3 問とも解答した場合には，1）と 2）について解答したものとして採点し，3）に対する解答は採点しないものとする．