平成21年8月1日（土）
9:00～12:00

大阪大学大学院情報科学研究科

コンピュータサイエンス専攻
情報システム工学専攻
情報ネットワーク学専攻
マルチメディア工学専攻
バイオ情報工学専攻

平成22年度　博士前期課程　入試問題

（A）情報工学

【注意事項】

- 問題数は必須問題3題（問題1～3），選択問題8題（問題4～11），合計11題である．必須問題は3題すべて解答すること．また，選択問題は2題を選択して解答すること．
- 問題用紙は表紙を含めて22枚である．
- 解答用紙は全部で7枚である．
  1枚目（赤色）の解答用紙には問題1（必須問題）の解答を
  2枚目（緑色）の解答用紙には問題2（必須問題）の(1)の解答を
  3枚目（緑色）の解答用紙には問題2（必須問題）の(2)の解答を
  4枚目（黄色）の解答用紙には問題3（必須問題）の(1)の解答を
  5枚目（黄色）の解答用紙には問題3（必須問題）の(2)の解答を
  6枚目（白色）の解答用紙には問題4～11（選択問題）から選択した1題の解答を
  7枚目（白色）の解答用紙には問題4～11（選択問題）から選択したもう1題の解答をそれぞれ記入すること．
  解答用紙は間違えると採点されないことがあるので注意すること．
- 解答用紙は7枚すべてを回収するので，すべての解答用紙に受験番号を記入すること．
- 解答用紙の「試験科目」の欄には解答した問題の科目名（「アルゴリズムとプログラミング」など）を，「問」の欄には対応する問題番号（1～11から一つ）を記入すること．また，選択問題調査票上の，選択した問題の番号（4～11から二つ）に○をつけること．
- 解答欄が不足した場合は裏面を使用すること．その際，表面末尾に「裏面に続く」と明記しておくこと．解答用紙の追加は認めない．
- 解答用紙には，日本語または英語で解答すること．

配点：(1) 15 点，(2) 24 点，(3) 24 点，(4-1) 13 点，(4-2) 12 点，(4-3) 12 点

図 1 の ANSI-C 準拠である C 言語のプログラムは，53 行目の整数型（int 型）配列（array）dat で与えられる入力に対して，昇順（ascending order）に整列（sort）するプログラムである．以下の各問に答えよ．

(1) このプログラムで実現されている整列方式（sorting algorithm）の名称を答えよ．

(2) このプログラムの 4 行目の display 関数を，1 回目に呼び出すときと，3 回目に呼び出すときの，配列 dat の内容を記述せよ．

(3) このプログラムは，入力を指定する 53 行目の配列 dat によって動作が変化する．49 行目の sort 関数の呼び出しにおいて，常に k と b が等しくなり，かつ 25 行目の処理が行われないように，配列 dat を並べ替えよ．

(4) 49 行目の sort 関数の呼び出しにおいて，常に k と b が等しくなる配列 dat が与えられたときについて，以下の各小問に答えよ．

(4-1) 配列 dat の要素数を n としたとき，プログラム実行の全体における sort 関数が呼び出される回数を求めよ．

(4-2) このような状況における，このプログラムが示す整列方式の時間計算量（time complexity）を，整列要素数 n を用いてオーダー表記（order notation）で答えよ．

(4-3) これは，このプログラムが示す整列方式にとってどのような状況であるか，時間計算量の観点から簡潔に述べよ．

```
1   #include <stdio.h>
2   #define MAX 8
3
4   void display(int *arr) {
5       int i;
6
7       for(i=0; i<MAX; i++) printf("%d ", arr[i]);
8       printf("¥n");
9   }
10
11  int arrange(int a, int b, int *arr) {
12      int left, right, bd, temp;
13
14      left=a; right=b; bd=arr[(a+b)/2];
15
16      while(1) {
17          while(arr[left]<bd) {
18              left++; if(left>b) break;
19          }
20          while(arr[right]>=bd) {
21              right--; if(right<a) break;
22          }
23
24          if(right<a) {
25              arr[(a+b)/2]=arr[a]; arr[a]=bd;
26              return a+1;
27          } else if(left<=right) {
28              temp=arr[left];
29              arr[left]=arr[right]; arr[right]=temp;
30
31              left++; right--;
32          } else {
33              break;
34          }
35      }
36
37      return(left);
38  }
39
40  void sort(int a, int b, int *arr) {
41      int k;
42
43      if(b<=a) return;
44
45      k=arrange(a, b, arr);
46      display(arr);
47
48      sort(a, k-1, arr);
49      sort(k, b, arr);
50  }
51
52  int main(void) {
53      int dat[]={30, 50, 70, 40, 20, 80, 60, 10};
54
55      sort(0, MAX-1, dat);
56      display(dat);
57      return 0;
58  }
```

図1　プログラム

配点：(1-1)30 点，(1-2)15 点，(2-1)5 点，(2-2-1)20 点，(2-2-2)10 点，(2-2-3)20 点

(1)

2 ビットの符号なし 2 進数(unsigned binary number) $A(a_1a_0)$，$B(b_1b_0)$ と 1 ビットの制御信号 S を入力とし，$S=0$ であれば二つの値 A,B で最大の値(Max(A,B))を，$S=1$ であれば二つの値 A,B で最小の値(Min(A,B))を出力 $C(c_1c_0)$ とする回路(circuit)を作りたい．ただし，a_0,b_0,c_0 をそれぞれ A,B,C の最下位ビット(Least Significant Bit)とする．以下の各小問に答えよ．

(1-1) 出力 C の各ビット c_1,c_0 を最小積和形（最簡積和形，minimal sum-of-products expression）で表せ．カルノー図（Karnaugh map）も示すこと．

(1-2)(1-1)の出力 c_1 を実現する回路を NOR ゲートのみを用いてゲート数最小で設計せよ．ただし，各ゲートのファンイン(fan in)に制限はないものとし，入力として a_1,a_0,b_1,b_0,S に加えてこれらの否定(not)も利用できるものとする．

(2)

文字の流れる電光掲示板(electrical bulletin board) を作成したい．この電光掲示板は格子状(grid)に並んだ表示素子(display element)で構成されており，図１のように各表示素子の発光(オン, on)／消光(オフ, off) 状態が右から左へと遷移していく．図中，表示素子の発光を黒で，消光を白で表している．電光掲示板への入力は，次時刻において右端の１列に表示される発光パターンである．以下の各小問に答えよ．

(2-1)　各表示素子は図２のような１入力１出力の順序回路によって制御されており，表示素子は論理値(logical value) 1(0)を入力するとオン(オフ)になるとする．表示素子の制御回路は，クロック(*CLK*)に同期して入力が 1(0)の時，表示素子をオン(オフ)にする．この制御回路を図３のように横に接続することで，電光掲示板の１行が構成される．表示素子の制御回路を D-フリップフロップ(D-FF)を用いて実現したい．解答用紙の回路図を完成させよ．

(2-2) 小問(2-1)で設計した回路を用いた電光掲示板に，リセット機能(reset) 及び，表示素子の動作確認機能 (operation check of display elements) を追加したい．リセット機能とは，リセット信号 R が１の時，クロックに同期して全表示素子がオフとなる機能である．表示素子の動作確認機能とは動作確認信号 C が１である間，図４のようにまず全表示素子が同期してオフとなった後，オン→オフ→オン→オフの点滅を全表示素子が同期しつつ繰り返す機能である．なおリセット機能は動作確認機能に優先するものとする．これらの機能を実現する表示素子の制御回路を設計したい．以下の各小問に答えよ．

(2-2-1) 解答用紙に記すこの回路の状態遷移図(state transition diagram)を完成させよ．なお状態遷移図中の S_0～S_2 は状態名である．

(2-2-2) 次のように状態 S_0～S_2 を状態変数(state variable) (Q_1,Q_0)に割り当てる．　　$S_0=(0,0), S_1=(0,1), S_2=(1,0)$

出力 *Out* を Q_1, Q_0, *In*, *R*, *C* の最小積和形（最簡積和形）で表せ．

(2-2-3) 2 個の D-FF を用いてこの回路を設計したい．Q_1, Q_0 に対応する D-FF の入力 D_1,D_0 を Q_1, Q_0, *In*, *R*, *C* の最小積和形（最簡積和形）で表せ．

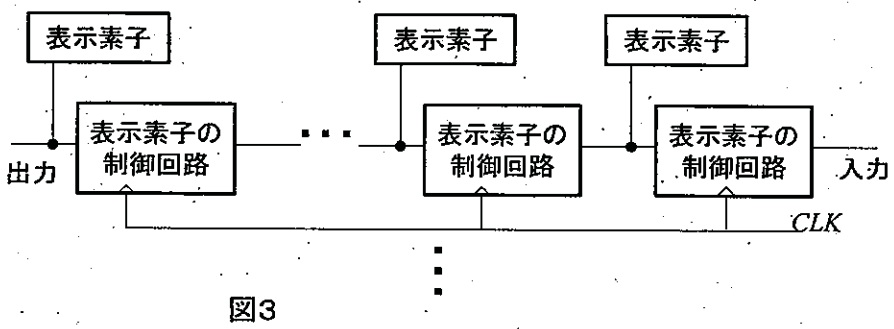


図3

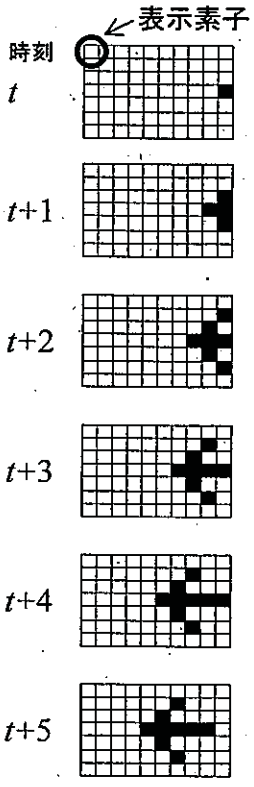


図1

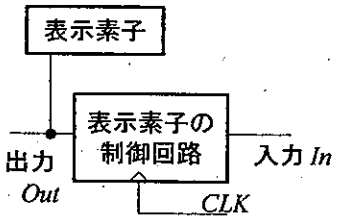


図2

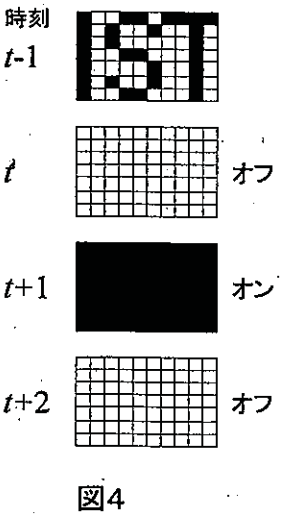


図4

3 （必須問題：計算機システムとシステムプログラム）　（情報工学 5／21）

配点：(1-1) 12 点，(1-2-1) 3 点，(1-2-2) 4 点，(1-2-3) 3 点，(1-2-4) 10 点，(1-3) 18 点，(2-1) 38 点，(2-2) 12 点

(1) 計算機システムにおける数の表現と加算器 (adder) の実現について，以下の小問 (1-1)〜(1-3) に答えよ．以下では，ビット列 (bit sequence) A が 2 進数において表す数を $(A)_2$ とする（例えば $(1000)_2=8$ である）．また，ビット列 A が 1 の補数系 (1's complement system) において表す数を $(A)_{1C}$，2 の補数系 (2's complement system) において表す数を $(A)_{2C}$ とする．

(1-1) 以下の文中の空欄 (X1)〜(X4) にあてはまるビット列ならびに空欄 (Y1)〜(Y4) にあてはまる数を答えよ．

1 の補数系では，負の整数 $-M$ の表現に M の 1 の補数を用いる．例えばビット数 4 のビット列を用いた場合，$5 =$ ((X1))$_{1C}$，$-5 =$ ((X2))$_{1C}$ である．また，ビット数 n のビット列で表現できる数の範囲は，n を用いて (Y1) 〜 (Y2) と表せる．

2 の補数系では，負の整数 $-M$ の表現に M の 2 の補数を用いる．例えばビット数 4 のビット列を用いた場合，$5 =$ ((X3))$_{2C}$，$-5 =$ ((X4))$_{2C}$ である．また，ビット数 n のビット列で表現できる数の範囲は，n を用いて (Y3) 〜 (Y4) と表せる．

(1-2) ビット列 $(a_1\ a_0)$ と $(b_1\ b_0)$ の加算を行う 2 ビットの符号無し加算器，ならびにビット列 $(a_3\ a_2\ a_1\ a_0)$ と $(b_3\ b_2\ b_1\ b_0)$ の加算を行う 4 ビットの符号無し加算器を実現したい．この加算器の実現には AND ゲート，OR ゲート，NOT ゲートを用いるとし，各ゲートの遅延 (delay) は，入力信号の数に関わらず T とする．

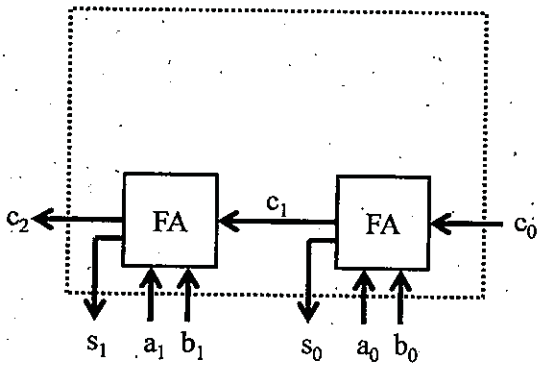


図 1：2 ビット桁上げ伝搬加算器

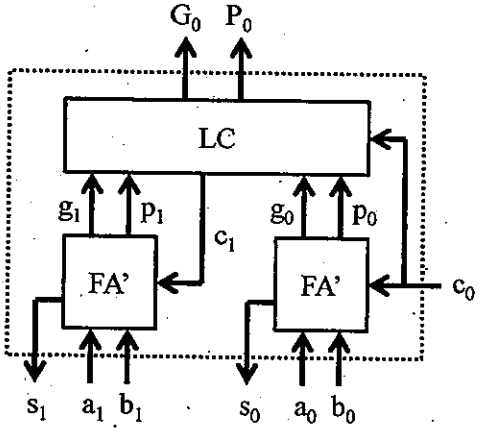


図 2：2 ビット桁上げ先見加算器

(1-2-1) 全加算器 (full adder) FA を用いた，2 ビットの桁上げ伝搬加算器 (ripple-carry adder) を図 1 に示す．c_1 を a_0，b_0 ならびに c_0 を用いて表せ．

(1-2-2) 入力信号 a_i ならびに b_i に対し，$g_i=a_i b_i$ ならびに $p_i=a_i+b_i$ を出力する全加算器 FA' を考える．FA' を用い，2 ビット目の桁上げ (carry) 信号 c_2 の代わりに $c_2=G_0+P_0 c_0$ となる信号 G_0 ならびに P_0 を生成する，2 ビットの桁上げ先見加算器 (carry look-ahead adder) を図 2 に示す．出力信号 G_0 および P_0 を，論理回路 (logic circuit) LC の入力信号で表せ．

(1-2-3) 図 2 の桁上げ先見加算器のすべての入力信号が同時にそろったとする．その時刻から，出力信号 G_0 および P_0 が決定されるまでの時間を，ゲート遅延 T で表せ．

(1-2-4) 3 個の論理回路 LC（これらを LC_1，LC_2，LC_3 で表す）を高さ 2 の 2 分木 (binary tree) 状に接続する．それに 4 個の全加算器 FA' を接続して構成した，4 ビットの桁上げ先見加算器を図 3 に示す．この加算器のすべての入力信号が同時にそろったとする．その時刻から，値が決定されるまでの時間が最も大きい桁上げ信号を，c_1，c_2，c_3 から選んで答えよ（複数ある場合はすべて答えよ）．また，それが決定されるまでの時間をゲート遅延 T で表し，その算出根拠も示せ．

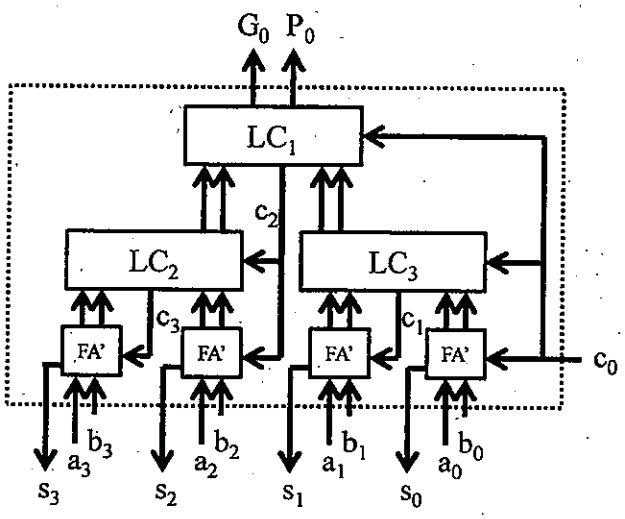


図3： 4ビット桁上げ先見加算器

(1-3) 1または2の補数系による加減算を，n ビットの符号無し加算器で実現したい．以下の文中の空欄 (Z1)～(Z6) にあてはまる語句や式を下の選択肢より選んでその記号で答えよ．同じ選択肢を複数回用いてもよい．なお，ビット列Aに対し，$\bar{A}$ はAのすべてのビットを反転して得られるビット列とする．

まず，1の補数系による加算 $(A)_{1C}+(B)_{1C}$ を考える．簡単のため，加算結果はオーバーフロー (overflow) しないとする（すなわち $|(A)_{1C}+(B)_{1C}| \leqq 2^{n-1}-1$）．

$(A)_{1C}$，$(B)_{1C}$ がともに正数の場合を考える．このとき，$(A)_{1C} = (A)_2$，$(B)_{1C} = (B)_2$ であるため，加算器の出力 $(A)_2+(B)_2$ は

$$(A)_2+(B)_2 = (A)_{1C}+(B)_{1C}$$

となる．したがって，加算器の出力をそのまま加算結果とできる．

$(A)_{1C}$，$(B)_{1C}$ がともに負数の場合を考える．このとき，$(A)_{1C}=$ (Z1) であり，$(B)_{1C}$ についても同様である．また，$\bar{A}$ の定義より，$(A)_2+\{$ (Z2) $\}=2^n-1$ が常に成り立つ（Bについても同様の式が成り立つ）．したがって，加算器の出力は

$$(A)_2+(B)_2 = \{2^n-1\}+\{\text{(Z3)}\}$$

となる． (Z3) は正の値であるため，2^n が最上位ビット (most significant bit) からの桁上げ (carry) に相当する．この桁上げを検出し，それを最下位ビット (least significant bit) に加える処理を行うことで，$(A)_{1C}+(B)_{1C}$ の1の補数である (Z3) が得られる．$(A)_{1C}$，$(B)_{1C}$ のいずれか一方が負数の場合も，同様の処理で加算結果が得られる．この処理は (Z4) とよばれる．

次に，2の補数系による加算 $(A)_{2C}+(B)_{2C}$ を考える．1の補数系の場合と同様，加算結果はオーバーフローしないとする（すなわち $|(A)_{2C}+(B)_{2C}| \leqq 2^{n-1}-1$）．

$(A)_{2C}$，$(B)_{2C}$ がともに負数の場合を考える．このとき，$(A)_{2C}=$ (Z5) であり，$(B)_{2C}$ についても同様である．したがって，加算器の出力は，

$$(A)_2+(B)_2 = 2^n+\{\text{(Z6)}\}$$

となる．この場合，最上位ビットからの桁上げを無視するだけで，$(A)_{2C}+(B)_{2C}$ の2の補数である (Z6) が得られる．したがって，(Z4) は必要ない．

[(Z1)～(Z6) の選択肢]

(ア) $(\bar{A})_2+1$	(イ) $(\bar{A})_2$	(ウ) $-(\bar{A})_2-1$	(エ) $-(\bar{A})_2$
(オ) $2^n-1+(A)_{1C}+(B)_{1C}$	(カ) $2^n-1-(A)_{1C}-(B)_{1C}$	(キ) $2^n+(A)_{1C}+(B)_{1C}$	(ク) $2^n-(A)_{1C}-(B)_{1C}$
(ケ) $2^n-1+(A)_{2C}+(B)_{2C}$	(コ) $2^n-1-(A)_{2C}-(B)_{2C}$	(サ) $2^n+(A)_{2C}+(B)_{2C}$	(シ) $2^n-(A)_{2C}-(B)_{2C}$
(ス) 循環桁上げ (end-around carry)		(セ) 補数変換 (complement conversion)	
(ソ) 先見桁上げ (look-ahead carry)		(タ) 基数変換 (radix conversion)	

(2) プロセスの相互排除 (mutual exclusion) に関する以下の小問 (2-1), (2-2) に答えよ．

(2-1) 次の説明文の空欄 [(a)] ～ [(n)] に当てはまる適当な語句や数字，命令文を選択肢から一つ選び，その番号を答えよ．ただし，選択肢は複数回選んでもよい．

複数のプロセス間で共有されるプログラムやデータなどの資源を共有資源 (shared resource) という．それぞれのプロセスにおいて，同時に一つのプロセスしか利用できない共有資源を使用する領域を危険領域，またはきわどい部分 (critical region or critical section) と呼ぶ．二つ以上のプロセスが同時に危険領域を実行しようとする場合は，プロセス間で同期をとり，一つのプロセスだけが危険領域を実行するようにしなければならない．このようにあるプロセスが危険領域を実行している間，他のプロセスが危険領域を実行しないようにすることを相互排除という．相互排除を実現するには，次の三つの条件が必要である．

条件①　二つ以上のプロセスが同時に危険領域を実行する [(a)] を禁止する．
条件②　二つ以上のプロセスが互いに危険領域の実行を無期限に阻止しあう [(b)] を禁止する．
条件③　危険領域を実行しているプロセスが一つも存在せず，かつ他に要求を出しているプロセスがなければ，危険領域を実行しようとするプロセスにただちに実行許可が与えられる．

ここで二つのプロセスの相互排除を実現することを目指し，手法 1～4 を考える．ただし，条件①～③をすべて満たすとは限らない．まず，図 1 に手法 1 を示す．図中の P1(), P2()がそれぞれプロセス P1, P2 に相当する．手法 1 では，危険領域の実行許可を表す turn という共有変数を用意する．turn が [(c)] の場合はプロセス P1 が危険領域を実行することができる．

```
1: int turn = 1;        /* 共有変数 */
2: void P1() {          /* プロセス P1 */
3:     for ( ; ; ) {
4:         ・・・       /* 危険領域でない */
5:         while (turn != 1)
6:             ;
7:         ・・・       /* 危険領域 */
8:         turn = 2;
9:     }
10: }
```

```
1:
2: void P2() {          /* プロセス P2 */
3:     for ( ; ; ) {
4:         ・・・       /* 危険領域でない */
5:         while (turn != 2)
6:             ;
7:         ・・・       /* 危険領域 */
8:         turn = 1;
9:     }
10: }
```

図 1: 手法 1

図 2 に，手法 2 を示す．プロセス P1 は，c1 を [(d)] にすることにより危険領域を実行しようとしていることを宣言する．c2 が [(e)] の時，プロセス P1 は危険領域を実行することができる．

```
1: int c1 = 1;          /* 共有変数 */
2: int c2 = 1;          /* 共有変数 */
3: void P1() {          /* プロセス P1 */
4:     for ( ; ; ) {
5:         ・・・       /* 危険領域でない */
6:         c1 = 0;
7:         while (c2 == 0)
8:             ;
9:         ・・・       /* 危険領域 */
10:        c1 = 1;
11:    }
12: }
```

```
1:
2:
3: void P2() {          /* プロセス P2 */
4:     for ( ; ; ) {
5:         ・・・       /* 危険領域でない */
6:         c2 = 0;
7:         while (c1 == 0)
8:             ;
9:         ・・・       /* 危険領域 */
10:        c2 = 1;
11:    }
12: }
```

図 2: 手法 2

図 3 に手法 3 を示す．プロセス P1 は c1 を [(d)] にすることにより危険領域を実行しようとしていることを宣言する．プロセス P1 は，c2 が [(e)] の時，危険領域を実行することができる．プロセス P1 は，c2 が [(f)] の時，turn が 1 であれば c2 が [(g)] になるのを待つ．プロセス P1 は，c2 が [(f)] の時，turn が 2 であれば c1 を [(h)] にして turn が 1 になるまで待つ．c1 を [(h)] にするのは，プロセス P2 に危険領域の実行許可を与えることを意味する．

```
1: int turn = 1;        /* 共有変数 */
2: int c1 = 1;          /* 共有変数 */
3: int c2 = 1;          /* 共有変数 */
4: void P1() {          /* プロセス P1 */
5:     for ( ; ; ) {
6:         ・・・       /* 危険領域でない */
7:         c1 = 0;
8:         while (c2 == 0) {
9:             if (turn == 2) {
10:                c1 = 1;
11:                while (turn == 2)
12:                    ;
13:                c1 = 0;
14:            }
15:        }
16:        ・・・       /* 危険領域 */
17:        turn = 2;
18:        c1 = 1;
19:    }
20: }
```

```
1:
2:
3:
4: void P2() {          /* プロセス P2 */
5:     for ( ; ; ) {
6:         ・・・       /* 危険領域でない */
7:         c2 = 0;
8:         while (c1 == 0) {
9:             if (turn == 1) {
10:                c2 = 1;
11:                while (turn == 1)
12:                    ;
13:                c2 = 0;
14:            }
15:        }
16:        ・・・       /* 危険領域 */
17:        turn = 1;
18:        c2 = 1;
19:    }
20: }
```

図 3: 手法 3

複数の共有資源，複数のプロセス間の相互排除を実現する手法として，セマフォ (semaphore) を用いる手法がある．セマフォ S は，複数の共有資源を扱うため，非負の整数値変数であるセマフォ変数 s とプロセスを登録するキュー (queue) Q からなる構造体 (structure) であり，この構造体への操作として次の signal(S)命令，wait(S)命令が定義されている．

```
void signal(S) {
    if （Q が空でない）{ Q からひとつプロセスを取り出し，そのプロセスを [(i)] 状態にする }
    else { s に 1 を加算する }
}
void wait(S) {
    if （s が正である ）{ s から 1 を減算する }
    else { Q にこのプロセスを登録し，この命令を実行したプロセスを [(j)] 状態にする }
}
```

よって，セマフォを用いて，一つの共有資源を有する二つのプロセス間で相互排除を実現する手法は図 4 (手法 4) のように表すことができる．

```
1: セマフォ S を初期化（s を 1，Q を空）
2: void P1() {          /* プロセス P1 */
3:     for ( ; ; ) {
4:         ・・・       /* 危険領域でない */
5:         [(k)]
6:         ・・・       /* 危険領域 */
7:         [(l)]
8:     }
9: }
```

```
1:
2: void P2() {          /* プロセス P2 */
3:     for ( ; ; ) {
4:         ・・・       /* 危険領域でない */
5:         [(m)]
6:         ・・・       /* 危険領域 */
7:         [(n)]
8:     }
9: }
```

図 4: 手法 4

選択肢

（ア）危険領域 (critical region)	（イ）飢餓状態 (starvation)	（ウ）相互排除 (mutual exclusion)
（エ）ビジーウェイト (busy wait)	（オ）ブロック (blocked)	（カ）ライブロック (livelock)
（キ）実行可能 (ready)	（ク）デッドロック (deadlock)	（ケ）協調 (coordination)
（コ）signal(S)；	（サ）wait(S)；	（シ）相互実行 (mutual execution)
（ス）実行 (running)	（セ）並行プログラム (concurrent program)	
（ソ）2	（タ）1	（チ）0

(2-2) 小問 (2-1) に示した手法 1～3 がそれぞれ満たす条件を小問 (2-1) に示した条件①～③からすべて選び，解答用紙の表を埋めよ．

配点：(1-1) 20 点，(1-2) 20 点，(2-1) 15 点，(2-2) 15 点，(2-3) 15 点，(3) 15 点

以下の各問に答えよ．なお，いずれの問も，導出過程を示すこと．また，真空の誘電率 (permittivity) は ε_0 とせよ．

(1) 自由空間 (free space) 中に，半径 (radius) a の導体球 (conducting sphere) があり，電荷 (electric charge) q に帯電している．このとき，以下の各小問に答えよ．適切な座標系 (coordinate system) を定めて答えること．

(1-1) ガウスの法則 (Gauss's law) を用いて，導体外の任意の点における電界 (electric field) $\boldsymbol{E}$ および電束密度 (electric flux density) $\boldsymbol{D}$ を求めよ．

(1-2) 無限遠点を 0 として，導体外の任意の点における電位 (electric potential) ϕ を求めよ．

(2) 図 1 に示すように，自由空間中に，半径 a の導体球が 2 つあり，一方は $-q$，他方は q に帯電している．2 つの導体球間の距離は $2d$ であり，$a \ll d$ である．このとき，以下の各小問に答えよ．

(2-1) 2 つの導体球間の電位差 V を求めよ．

(2-2) $-q$ に帯電している導体球から他方の導体球へ，微小な電荷 Δq を運ぶのに必要な仕事量 ΔW を求めよ．

(2-3) この系に蓄えられている静電エネルギー (electrostatic energy) W を求めよ．

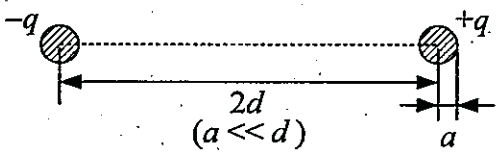


図 1

(3) 図 2 に示すように，接地された (grounded) 十分広い導体板 (conducting plate) があり，その表面から d の距離に半径 a の導体球がある．$a \ll d$ であり，導体球は q に帯電している．この時，導体球に作用する力の大きさ $|\boldsymbol{F}|$ を求めよ．

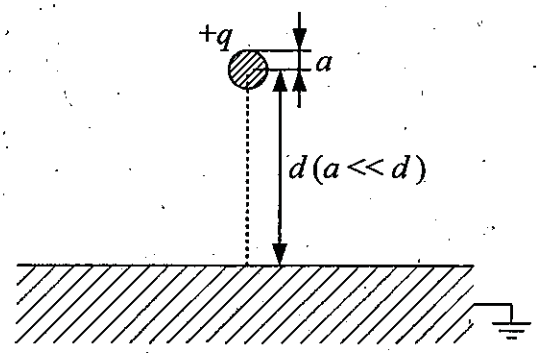


図 2

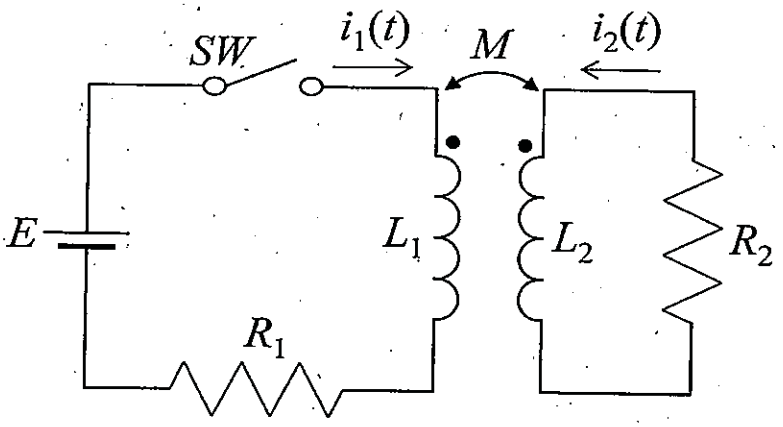
配点：(1)40 点，(2)25 点，(3)35 点

抵抗（resistance）R_1，R_2と，自己インダクタンス（self inductance）L_1，L_2および相互インダクタンス（mutual inductance）Mをもつ結合インダクタ（coupled inductor）からなる下図の回路（circuit）において，時刻$t<0$[s]でスイッチSWは開いており，定常状態（steady state）にあるものとする．以下の各問に答えよ．ただし，計算過程も記すこと．

(1) ラプラス変換領域において，電流$i_1(t)$および$i_2(t)$は，それぞれ$I_1(s)$および$I_2(s)$と表されるものとする．$t=0$ [s]にスイッチSWを閉じたとき，$t>0$[s]におけるラプラス変換領域（s-領域）(Laplace transform domain（s-domain)）上のインピーダンス表現による等価回路を示し，$I_1(s)$および$I_2(s)$に対する閉路方程式（loop equations）をたてよ．

(2) $R_1=R_2=1$[Ω]，$L_1=L_2=2$[H]，$M=1$[H]，$E=1$[V]とする．このとき，(1)でたてた閉路方程式を用いて，$t>0$ [s]における電流$i_1(t)$および$i_2(t)$を求めよ．

(3) (1)の状態から，$t=1$[s]にスイッチSWを開いたものとする．このとき，$t>1$[s]におけるラプラス変換領域上のインピーダンス表現による等価回路を示し，電流$i_2(t)$を求めよ．なお，R_1，R_2，L_1，L_2，M，Eの値は，(2)と同じものを用いよ．ただし，スイッチSWを開く直前の電流$i_1(t)$および$i_2(t)$の値として，定数a_1およびa_2をそれぞれ用いてよい．



配点: (1-1)20 点, (1-2)30 点, (2-1)20 点, (2-2)30 点

以下の各問に答えよ.

(1) 下記の複素関数 (complex function) についての小問に答えよ.

(1-1)

$\int_{|z|=1} (z - \frac{1}{z})^{2n} \frac{dz}{z}$ を計算せよ.

ただし, $\left(a \pm \frac{1}{a}\right)^{2n}$ の定数項 (constant term) が $(\pm 1)^n \frac{(2n)!}{(n!)^2}$ となること を用いてもよい.

(1-2) 上の結果を用いて

$\int_0^{2\pi} \sin^{2n}\theta d\theta$ を計算せよ.

(2) 下記の方程式について, 以下の小問に答えよ.

$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}$ $(0 \leqq x \leqq 1, t \geqq 0)$ の解を $u(x,t)$ とする.

(2-1) 変数分離法 (separation of variables) を用いて, t と x に関する常微分方程式 (ordinary differential equation) を求めよ.

(2-2) 境界条件 (boundary condition) を $u(0,t) = u(1,t) = 0$ とし, (2-1) の結果を用いて一般解 (general solution) を求めよ.

配点: (1-1) 20 点, (1-2-1) 15 点, (1-2-2) 20 点, (2-1) 20 点, (2-2) 25 点

(1) 有向グラフ (directed graph) $G = (V, E)$ を考える. G の頂点 (vertex) の集合 V に対して, 次のように関係 (relation) $R_G \subseteq V \times V$ を定める.

$(u, v) \in R_G$ となることの必要十分条件は, G 上に u から v へ至る経路 (path) が存在し, かつ, v から u へ至る経路が存在することである.

ただし, 経路は同じ頂点を複数回通ってもよいものとし, また, 各頂点 v はそれ自身で v から v への経路をなすと見なす.

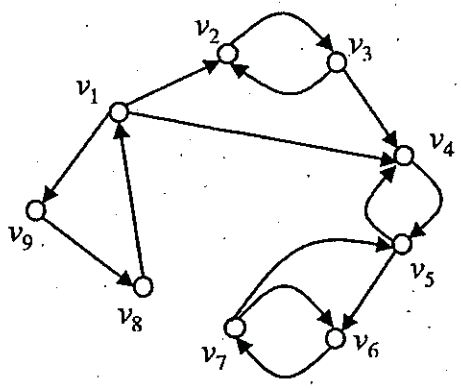
以下の各小問に答えよ.

(1-1) R_G が同値関係 (equivalence relation) であることを証明せよ.

(1-2) R_G の同値類 (equivalence class) の集合を $\overline{C}$ とする. G と $\overline{C}$ から, 頂点の集合 V', 辺 (edge) の集合 $E' \subseteq V' \times V'$ を持つ新しい有向グラフ $G' = (V', E')$ を次のように定める.

- $V' = \{v'_C \mid C \in \overline{C}\}$
  すなわち, G' 中の頂点と, $\overline{C}$ の元が, 1 対 1 に対応するように V' を定める.
- $E' = \{(v'_{C_1}, v'_{C_2}) \mid C_1$ と C_2 が $\overline{C}$ の相異なる元であり, かつ, C_1 中のある頂点から C_2 中のある頂点への辺が G 上に存在する$\}$

(1-2-1) G が下図に示したグラフであるときの G' を図示せよ. また, G' の各頂点と R_G の同値類との対応を示せ.



(1-2-2) 一般にどのような G に対しても, 上記のように定義された G' 上には, 二つ以上の異なる頂点を含む巡回路 (cycle) が存在しないことを証明せよ.

(2) 以下では，N は 1 以上の整数の集合，また，$\mathcal{P}(X)$ は集合 X のべき集合 (power set) をそれぞれ表す．順序対 (x,y) と (u,v) について，「"$(x,y)=(u,v)$" $\Leftrightarrow$ "$x=u$ かつ $y=v$"」と定義する．

このとき，次のように三つの集合 X_1, X_2, X_3 と，それぞれの上で定義された関係 R_1, R_2, R_3 を考える．

$X_1 = \mathcal{P}(N)$, $\quad R_1 = \{(x,y) \mid x \subseteq y\}$
$X_2 = N$, $\quad R_2 = \{(x,y) \mid y = cx$ となるような 2 以上の整数 c が存在する $\}$
$X_3 = N \times N$, $\quad R_3 = \{((x,y),(z,w)) \mid x$ と z の積が y と w の積に等しい $\}$

以下の各小問に答えよ．

(2-1) R_1, R_2, R_3 のうち，推移性 (transitivity) を持たない関係はどれか．すべて挙げよ．また，それぞれについて推移性を持たない証拠を示せ．

(2-2) R_1, R_2, R_3 のうち，半順序関係 (partial order relation) はどれか．すべて挙げよ．また，それぞれについて半順序関係であることを証明せよ．

配点:(1-1)~(1-5) 各 10 点，(2-1)15 点，(2-2)15 点，(2-3)20 点

一階述語論理式 (first-order logic formula) として以下の記号を用いる．$\forall, \exists$はそれぞれ全称作用素 (universal quantifier)，存在作用素 (existential quantifier) であり，併せて限定作用素 (quantifier) と呼ぶ．また，$\rightarrow, \vee, \wedge, \neg$ はそれぞれ，含意 (implication)，選言 (disjunction, or)，連言 (conjunction, and)，否定 (negation, not) の各論理演算子とする．

また，断りのない限り，u, v, w, x, y, z で変数 (variable) 記号，a, b で定数 (constant) 記号，f, g で関数 (function) 記号，p で述語 (predicate) 記号を表わす．

(1) 一階述語論理式の解釈 (interpretation)I は (D, C, F, P) の 4 項組で与えられる．ここで，D は値集合，C は各定数記号への D の要素の割り当て，F は各 n 引数関数記号への $(\mathsf{D}^n \rightarrow \mathsf{D})$ の要素の割り当て，P は各 n 引数述語記号への $(\mathsf{D}^n \rightarrow \{0,1\})$ の要素の割り当てである．ここで 0,1 はそれぞれ偽 (false)，真 (true) を表わす．

例えば一階述語論理式 $\forall x\, p(f(b,x),a)$ に対して，解釈 I_0 として
- D を非負整数 (nonnegative integer) 全体からなる集合とし，
- C として a, b それぞれへ非負整数値 0,1 を割り当て，
- F として 2 引数関数記号 $f(u,w)$ へ非負整数上の加算 $u+w$ を割り当て，
- 2 引数述語記号 $p(u,w)$ へ非負整数上の比較演算 $u > w$ を割り当てたとき (例えば $4 > 3$ の値は 1 である)，

$\forall x\, p(f(b,x),a)$ の，解釈 I_0 のもとでの評価値は 1 となる．

以下の小問の各論理式が (a) 恒真 (valid), (b) 充足可能 (satisfiable) であるが恒真でない，(c) 充足不能 (unsatisfiable) のいずれであるか判断し記号で答えよ．(b) である場合は，さらに評価値を 1 とする解釈，0 とする解釈をそれぞれ一つずつ挙げよ．

なお，解釈として簡単明瞭なもの，例えば D が数個の要素のみからなる集合や，非負整数全体からなる集合など値集合が明確なものを用いること．また F, P は各引数値に対して値が個別に定義されているものや，よく知られた簡単な演算 (四則演算，大小比較等) を考えること．

(1-1) $p(a) \rightarrow \forall x\, p(x)$

(1-2) $(\forall x\, p(x)) \rightarrow \forall x\, p(f(x))$

(1-3) $(\exists x\, p(x)) \rightarrow \forall x\, p(x)$

(1-4) $(\forall x \exists y\, p(x,y)) \rightarrow \exists y \forall x\, p(x,y)$

(1-5) $(\exists y \forall x\, p(x,y)) \rightarrow \forall x \exists y\, p(x,y)$

(2) 以下で与えられる論理式 A が恒真であることを，A の否定のスコーレム化 (Skolemization)(スコーレム連言標準形 (Skolem conjunction normal form) を求めること) を行い，得られた式に対して導出原理 (resolution principle) を用いてその式の充足不能性を示すことによって，示したい．以下の各小問に答えよ．各小問において導出過程も示すこと．

$$A = (p(a,b) \wedge \forall x \forall y \exists z\, (p(x,y) \rightarrow p(g(x,z),y)) \wedge \forall x \forall y\, (p(x,y) \rightarrow p(y,x))) \rightarrow \exists z \exists w \exists v\, p(g(g(z,w),v),a)$$

(2-1) $\neg A$ の冠頭標準形 (prenex normal form) を示せ．
冠頭標準形はすべての限定作用素が先頭にある閉論理式 (closed formula) である．

(2-2) 小問 (2-1) で得た論理式をスコーレム化せよ．

(2-3) 小問 (2-2) で得た論理式 A' をもとに，導出原理を用いて A' が充足不能であることを示せ．

9 （選択問題：計算理論）

配点: (1-1) 20 点, (1-2) 20 点, (1-3) 15 点, (2-1) 15 点, (2-2) 15 点, (2-3) 15 点

(1) オートマトン (automaton) に関する以下の各小問に答えよ．ただし導出の過程は示さなくて良い．なお，オートマトン M が受理する言語を $L(M)$ で表す．

(1-1) 決定性有限オートマトン (deterministic finite automaton) M_1 の初期状態 (initial state) を p，受理状態 (accepting state) を r，入力記号 (input symbol) を 0 と 1 とし，状態遷移表を右の通り定める．$L(M_1) = L(M_2)$ を満たす最小状態数 (minimum number of states) の決定性有限オートマトン M_2 を求め，状態遷移図 (state transition diagram) の形で示せ．ただし，初期状態と受理状態が分かるように描くこと．

M_1

	0	1
p	v	t
q	u	s
r	u	r
s	q	u
t	v	r
u	q	s
v	t	r

(1-2) 決定性有限オートマトン M_3, M_4, M_5 それぞれについて，その初期状態を p，受理状態を p，入力記号を 0 と 1 とし，状態遷移表を下に示す通り定める．M_3, M_4, M_5 それぞれに対し、受理する言語を表す正規表現 (regular expression) を下記の選択肢 (a)〜(i) より選べ．

選択肢:

(a) (0 + 1*0*11)*　(b) (0 + 10*11*0)*　(c) (0 + 11*010)*　(d) (0*10*011*01*)*0*

(e) (0*11*0(11*0)*0)*0*　(f) (0*11*1(01*)*10)*　(g) (0 + 1(00 + (01 + 1)(0*11)*0*101))*

(h) (0 + 1(00 + (01 + 1)(0*11)*0*10))*　(i) (0 + 1(00 + 01(0*1)*0*10))*

M_3

	0	1
p	p	r
q	p	r
r	q	r

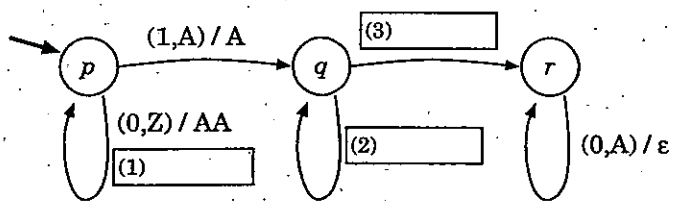
M_4

	0	1
p	p	q
q	q	r
r	p	r

M_5

	0	1
p	p	r
q	p	s
r	q	s
s	s	q

(1-3) 空スタック受理 (acceptance by empty stack) の決定性プッシュダウンオートマトン (deterministic pushdown automaton) M_6 を，$L(M_6) = \{0^i1^j0^{2i} : i, j > 0\}$ となるように構成したい．右の図は M_6 の状態遷移図である．図の空欄 (1)〜(3) それぞれに対し，記号の形で適切な動作一つを埋めよ．



なお，この状態遷移図の見方を以下に説明する．太い矢印が指し示している状態が初期状態である．例えば状態 p から p への矢印に添えられた記号 (0, Z) / AA は，入力テープ上の文字が 0 でスタックトップの記号が Z のとき，スタックトップの記号をポップするとともに入力テープのヘッドを次に進め，次の状態を p にし，スタックに AA (すなわち A 二つ) をプッシュする動作を表している．

Z はスタックの初期 (底) 記号 (initial symbol) であり，実行開始時には Z のみがスタックに入っている．実行は状態遷移図に従って進む．もし合致する動作が無い場合は動作を停止する．もしプッシュする記号が ε の場合はスタックに何もプッシュしない．入力を全て読み終えたときにスタックが空であることが，空スタック受理の必要十分条件である．

(2) 文脈自由文法 (context-free grammar) に関して以下の各小問に答えよ．ただし導出の過程は示さなくて良い．なお，文法 G が生成する言語を $L(G)$ で表す．

(2-1) 文脈自由文法 $G_1(V_1, T_1, P_1, S_1)$ を以下に示す．$L(G_1) = L(G_2)$ であるような ε なし (ε-free) 文脈自由文法 G_2 を示せ．ただし，G_2 の非終端記号 (non-terminal symbol) の集合は V_1 とする．

$G_1(V_1, T_1, P_1, S_1)$

- 非終端記号 (non-terminal symbol) の集合 $V_1 = \{ S, A, B, C, D \}$
- 終端記号 (terminal symbol) の集合 $T_1 = \{ a, b, c, d \}$
- 開始記号 (start symbol) $S_1 = S$
- 生成規則 (generating rule) の集合 $P_1 = \{ \ S \to AB, \quad A \to ABCD, \quad A \to a, \quad B \to C, \quad B \to b, \quad B \to \varepsilon, \quad C \to B, \quad C \to c, \quad C \to \varepsilon, \quad D \to d \ \}$

(2-2) チョムスキー標準形 (Chomsky Normal Form) とは何かを簡潔に説明せよ．

(2-3) 文脈自由文法 $G_3(V_3, T_3, P_3, S_3)$ を以下に示す．$L(G_3) = L(G_4)$ であるようなチョムスキー標準形の文脈自由文法 G_4 を示せ．ただし G_4 で使用する非終端記号は，V_3 に含まれるものに加え，必要に応じて B, C, D, E の四つから適宜使用せよ．

$G_3(V_3, T_3, P_3, S_3)$

- 非終端記号の集合 $V_3 = \{ S, A \}$
- 終端記号の集合 $T_3 = \{ a, b \}$
- 開始記号 $S_3 = S$
- 生成規則の集合 $P_3 = \{ \ S \to AaAb, \quad A \to AA, \quad A \to a \ \}$

配点: (1-1)15 点, (1-2)15 点, (2-1)15 点, (2-2)20 点, (3-1)15 点, (3-2)20 点

(1) 図 1 は通信路行列 (channel transition matrix) が $\begin{pmatrix} 1-p-\delta & \delta & p \\ p & \delta & 1-p-\delta \end{pmatrix}$ となる 2 元対称消失通信路 (binary symmetric erasure channel) である. 送信記号集合は $\{0,1\}$, 受信記号集合は $\{0,E,1\}$ となる. ここで E は消失を表す. この通信路において, 最小距離 (minimum distance) d の 2 元ブロック符号 (binary block code) を用いて通信を行う. 今, 符号語 $\bar{u}$ を送信し, 語 $\bar{v}$ が受信されたとする. 以下の各小問に答えよ.

(1-1) 受信語 $\bar{v}$ の成分に消失 E はないとする. $\bar{v}$ と送信符号語 $\bar{u}$ のハミング距離 (Hamming distance) を t とするとき, $2t < d$ ならば $\bar{v}$ から最も近い符号語は唯一に定まりそれは $\bar{u}$ であることを, 三角不等式 (triangular inequality) を用いて示せ.

(1-2) 受信語 $\bar{v}$ 中には s 個の E があり, 消失ではない成分には誤りがないとする. $s < d$ ならば, 受信語 $\bar{v}$ における s 個の消失 E それぞれに適当に値を割り当てて得られる語のうち, 符号語であるものは送信符号語 $\bar{u}$ だけであることを示せ.

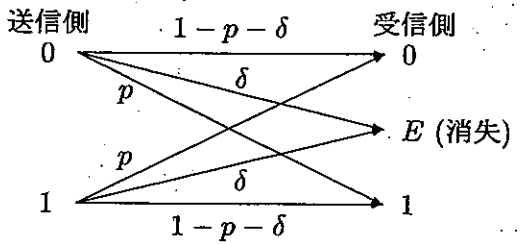


図 1: 2 元対称消失通信路

(2) 図 2 はビット誤り率 p の 2 元対称通信路 (binary symmetric channel) である. この 2 元対称通信路を BSC(p) で表す. 図 3 のように BSC(p) を三つ並列に用いて構成される通信路を考える. ただし, これら三つの BSC(p) は独立であるとする. 通信路の入力が三つの BSC(p) を通して送られ, それらの出力の多数決の結果を全体の通信路の出力とする. 以下の各小問に答えよ.

(2-1) 図 3 の通信路の通信路行列を求めよ.

(2-2) $0 < p < 1/2$ のとき, 図 3 の通信路は通信路 BSC(p) と比べて誤りが少ないという意味で優れていることを, 通信路行列を比較することにより説明せよ.

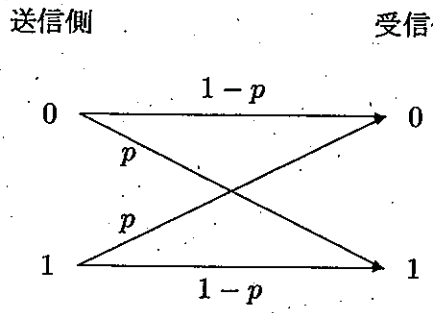


図 2: 2 元対称通信路 BSC(p)

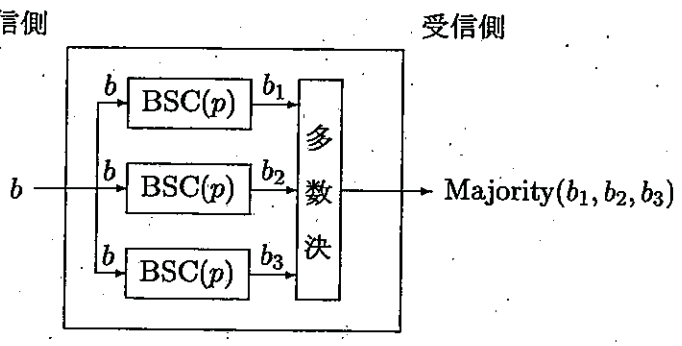


図 3: 2 元対称通信路 BSC(p) を三つ用いた通信路

(3) 図4は, 図2に示した2元対称通信路 BSC(p) を二つ並列に用いて構成される通信路である. ただし, 問 (2) と同様, これら二つの BSC(p) は独立であるとする. 通信路の入力が二つの BSC(p) を通して送られ, それらの出力が一致すればそれが全体の通信路の出力となり, そうでなければ消失となる. 以下の各小問に答えよ.

(3-1) 図4の通信路の通信路行列を求めよ.

(3-2) 通常, 確率 p の値は1と比べてかなり小さい (例えば, 10^{-4} 以下). その場合に図4の通信路と通信路 BSC(p) の優劣について述べた以下の文章の空欄 (ア)〜(ク) を埋めよ. 解答用紙には各空欄の記号と入るべき語句だけを書け. なお, 空欄 (イ), (エ) には適切な整数を, (オ) には用語を入れよ. それ以外の空欄についてはそこに書いてある語句のうち, 適切なものを選べ.

二つの通信路の通信路行列を比較する. 図4の通信路において送信したビットが正しく受信される確率は, 図2の2元対称通信路 BSC(p) におけるその確率に比べて, 少し [(ア) 大きい/小さい]. 図4の通信路において受信結果が消失となる確率は, BSC(p) において受信したビットが誤っている確率の約 [(イ)] 倍である. また, 図4の通信路において受信したビットが誤っている確率は, BSC(p) におけるその確率に比べて, [(ウ) 極めて大きい/少し大きい/少し小さい/極めて小さい]. 問 (1) の議論を発展させると, 1個の誤りは [(エ)] 個の消失に相当することが示せる. しかし, これらのことだけからでは, 図4の通信路と BSC(p) のどちらが優れているかは, 直ちにはわからない. 情報理論的には, どちらの通信路が優れているかを測る尺度として, [(オ)] が知られている. [(オ)] は, 送信側と受信側の [(カ) 相互/自己] 情報量の [(キ) 最大値/最小値] として定義されており, [(ク) 大きい/小さい] 方が優れた通信路である. これら二つの通信路も, [(オ)] を比較することで, どちらが優れているかがわかる.

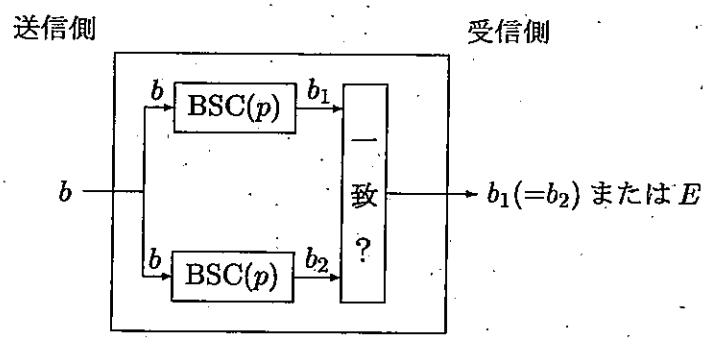


図 4: 2元対称通信路 BSC(p) を二つ用いた通信路

配点：(1) 30 点，(2-1) 10 点，(2-2) 20 点，(2-3) 20 点，(2-4) 20 点

(1) OSI (Open Systems Interconnection) 参照モデルにおけるデータリンク層 (data link layer) について述べた以下の文中の空欄の各記号(a)～(j)に当てはまる最も適切な用語を，選択肢よりそれぞれ選び，その番号を答えよ．なお，異なる記号には異なる選択肢が当てはまる．

データリンク層は，[(a)]に対して，伝送媒体 (transmission media) 上で隣接するノード間の誤りのない伝送路を提供する．データリンク層の主要な機能には，[(b)]，[(c)]，[(d)]，[(e)] などがある．[(b)]は，物理層 (physical layer) によってやりとりされるビット列を，フレーム (frame) と呼ばれるデータリンク層の PDU (Protocol Data Unit) に分割する機能であり，ビット列中のフレームの前後にフラグ (flag) と呼ばれる特殊なビット列を挿入することによって実現される．また，フレームの境界以外の場所でフラグと同じビット列が現れないよう，[(f)]を行う．[(c)]は，誤り検出 (error detection)，誤り訂正 (error correction)，再送制御 (retransmission)，[(g)]からなり，フレームを誤りなく正しい順序で受信側ノードに届ける機能である．[(d)]は，受信側ノードの受信バッファがあふれないように，確認応答 (acknowledgement) を待たずに送信可能なフレームの数を定め，送受信の速度を調整する機能である．[(e)]は，ノード間の伝送媒体の競合を調整する機能であり，[(h)]では搬送波検出 (carrier sense) と[(i)]の機能を有する[(j)]方式が用いられている．

選択肢：

(1) トポロジ制御 (topology control)，(2) 衝突検出 (collision detection)，
(3) 物理層 (physical layer)，(4) ネットワーク層 (network layer)，
(5) カプセル化 (encapsulation)，(6) インターネット (Internet)，
(7) 媒体アクセス制御 (Media Access Control)，(8) 輻輳制御 (congestion control)，
(9) ビット挿入 (bit stuffing)，(10) ビット訂正 (bit correction)，
(11) PPP (Point-to-Point Protocol)，(12) イーサネット (Ethernet)，
(13) 誤り制御 (error control)，(14) フレーム化 (framing)，
(15) トランスポート層 (transport layer)，(16) フロー制御 (flow control)，
(17) 順序制御 (sequence control)，(18) TCP (Transmission Control Protocol)，
(19) HDLC (High-level Data Link Control)，
(20) CSMA/CD (Carrier Sense Multiple Access with Collision Detection)，
(21) HTTP (Hypertext Transfer Protocol)

(2) ネットワーク層（network layer）における経路制御（routing）に関する以下の各小問に答えよ．

(2-1) 宛先アドレス（destination address）が 168.13.171.5 であるパケットが，以下に示す経路表（routing table）をもつルータに到着したとする．このパケットが送られる次のルータは A，B，C，D のいずれか，その理由とともに答えよ．

宛先アドレス	サブネットマスク	次のルータ
168.13.128.0	255.255.240.0	A
168.13.160.0	255.255.240.0	B
168.13.176.0	255.255.240.0	C
168.13.0.0	255.255.0.0	D

(2-2) 経路制御では，送信ホストから受信ホストにいたる経路にコストを設定し，それを最小とする方針がとられる．このような経路コストの例を二つ挙げよ．さらに，それぞれの欠点を一つずつ述べよ．ただし，二つのコストを a，b としたとき，a のコストに対する欠点として「b を最小にできない」は除くこと．また，それぞれ異なる欠点について述べること．

(2-3) 受信したパケットを隣接するすべてのルータに転送するフラッディング（flooding）を用いればパケットは必ず宛先に届くが，パケットがいつまでもネットワーク内を転送され続けないようにする必要がある．そのための方法を二つ挙げ，それぞれについて簡単に説明せよ．

(2-4) インターネットプロトコルで用いられている経路制御プロトコルには，RIP（Routing Information Protocol）などの距離ベクトル（distance vector）型プロトコルと，OSPF（Open Shortest Path First）などのリンク状態（link state）型プロトコルがある．経路計算のためにルータが管理するデータ量，リンク切れやルータ停止が発生した後の経路表の収束（convergence）の速度，それぞれの観点から，距離ベクトル型プロトコルとリンク状態型プロトコルを比較せよ．