東大 2019 年度創造情報学専門解答例

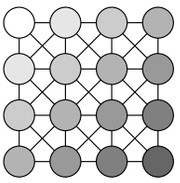
文殊の知恵
高橋那弥

目次

第 1 問

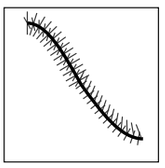
第 1 問　問題文

n × n 点（ピクセル）からなる 2 次元階調グレースケール画像について考える。なお、各点は、縦横斜めの近傍点とつながってるものとする。（右図参照）各ピクセル p は Pixel という型で表現し、その輝度は p.brightness と表現する。画像は、n × n の Pixel の配列 P として与えられる。擬似コード内では、基本的なデータ構造を適宜利用してよい。計算量については、n の関数として示せ。



(1) 黒い背景に白い物体がいくつか写っているとする（右図参照）。そのうちの 1 つの物体の面積を求める方法として、以下のような方法が考えられる。
「ある閾値に対して、それよりも明るい点のみを残し、それ以外の点を考慮から外す。残っている点から 1 つ選び、その点を含む連結領域の大きさ（点の数）を計算する。」
この計算を再帰呼び出しによって行うアルゴリズムを 20 行以内の擬似コードで示し、その計算量を O 記法を用いて答えよ。

(2) 以下のような方法で白い背景の画像に写っている黒い曲線を検出することを考える（右図参照）。自己交差はないものとする。
「両端の 2 点（与えられているものとする。）を連結する点列のうち、点列上の点の明るさの合計値が最小になるものを求める。」
この計算を効率よく行うアルゴリズムを 20 行以内の擬似コードで示し、その計算量を O 記法を用いて答えよ。



(3) 画像を点列で左右に分割する方法として（右図参照）、以下のような方法が考えられる。
「画像の上端と下端を結び、各行につき 1 点を経由するような連結された点列を考える。そのような点列の内、点の明るさの合計が最小になるような点列を求める。」
この計算を効率よく行うアルゴリズムを 20 行以内の擬似コードで示し、その計算量を O 記法を用いて答えよ。

(4) 画像をぼかす方法として、以下のような処理が考えられる。
「各内部点（近傍を 8 つ持つ点）について、その 8 近傍点の輝度の平均値を計算する。全ての内部点についてこの平均値を計算した後、すべての内部点の輝度を対応する平均値へと同時変更する。」
ここで、内部点の元の輝度を並べたベクトルを $\boldsymbol{x}$、変更後の輝度を 1 列にベクトルを $\boldsymbol{x'}$、外部点（画像の中の点の内、内部点以外の点）の輝度を並べたベクトルを $\boldsymbol{b}$ として、$\boldsymbol{x}, \boldsymbol{x'}, \boldsymbol{b}$ の関係を行列を使って表現したい。適切に行列を定義して、$\boldsymbol{x}, \boldsymbol{x'}, \boldsymbol{b}$ の関係式を示せ。

(5) (4) における処理を画像に対して無限回適用すると、画像の輝度 $\boldsymbol{x}$ は $\boldsymbol{x}^{\mathrm{inf}}$ に収束する。$\boldsymbol{x}^{\mathrm{inf}}$ を (4) で定義した行列を用いて解析的な式で表せ。但し、式には極限は含まないものとする。

第 1 問　解答例

(1) 題意を満たす擬似コードは以下のようになる。但し、参照範囲は考慮されているものとする。

```
func: 行数iにおいて明るい点だけを残す関数SearchBright(bool P2[][], int i)
    for j: from 0 to n - 1
        if P[i][j].brightnessが閾値より大きい
            (i, j)点が明るいかどうかを表すP2[i][j]にtrueを代入
        end if
    end for
    return 次の行について再帰呼び出しを行った後の戻り値
end func
func: 面積を計算する関数area(final boolean P2[][], int si, int sj)
    現在訪れている(i, j)点に関して訪れたかどうかを表すvisitid[i][j]を
        trueにする。
    area ← 1
    for i: from si - 1 to si + 1
        for j: from sj - 1 to sj + 1
            if 現在見ている点が訪れていないかつ明るい点である
                その点へ訪問し、その戻り値を現在の面積areaに足す。
            end if
        end for j
    end for i
    面積areaを戻り値として返す。
end func
```

この時の計算量は $\mathrm{O}(n^2)$ となる。

(2) 題意を満たす擬似コードは以下のようになる。但し、参照範囲は考慮されているものとする。

```
for i: from 両端の始点のx座標 to 両端の終点のx座標
    for j: from 両端の始点のy座標 to 両端の終点のy座標
        dp[i][j] ← P[i][j].brightness + min(dp[i - 1][j], dp[i][j - 1], dp[i -
            1][j - 1])
        dproot[i][j] ← dp[i - 1][j], dp[i][j - 1], dp[i - 1][j - 1]の中で一番
            小さい値を持つ点
    end for j
end for i
while rootが両端の終点から両端の始点になるまで
    root ←  dproot[root.x][root.y]
    求めたい点列pにrootを追加
end while
```

この時の計算量は $\mathrm{O}(n^2)$

(3)

第 2 問

第 2 問　問題文

太陽光発電システムについて考えよう。ソーラーパネルの維持管理のため、以下のような適用規則が定められているとする。

(i) n 枚のパネルが一つのグループとして維持管理される。
(ii) パネルはグループ毎に定期的に点検される。
(iii) パネルの状態は各グループ毎に n ビットデータとして報告される。ここで各ビットは対応するパネルに不具合があれば 1、不具合がなければ 0 とする。

不具合のあるパネルの数、すなわち n ビットデータの 1 の個数 k を数える"population count"問題を考えよう。以下の設問に答えよ。

まず、ソフトウェアによる解法を考えよう。ここでは、$0 < n \leq 32, 0 \leq k < \log_2 n$ とする。四則演算、論理演算、シフト演算、および表引きには 1 単位時間かかるとする。単純化のため、インデックスの足し算やループで用いる比較演算の演算時間はゼロとする。

(1) 単純な方式として各ビットの値をチェックし、1 の個数の総和を求める方式が考えられる。この方式の擬似コードを書き、その計算時間を答えよ。
(2) 実際、表引き操作を行うことで上述の方式 (1) を高速化できる。その計算時間を答えよ。
(3) 方式 (1) より高速かつ方式 (2) よりストレージを必要としない方式の擬似コードを示せ。その計算時間を答えよ。

ハードウェアによる解決を考えよう。ここでは、入力はビット列、出力は 2 進数とする。
(4) 入力 3 ビットの population count 論理回路 P_3 の真理値表を書け。AND, OR, NOT ゲートを用いて P_3 を設計せよ。

(5) 入力 6 ビットの population count 論理回路 P_6 を論理回路 P_3 を利用して作成せよ。必要に応じて、追加で AND, OR, NOT ゲートを使ってもよい。
(6) 入力 n ビットの population count 論理回路 P_n を考えるとき、n が増えると遅延が問題になる。この問題を解決する方法を述べよ。

第 2 問　解答例

(1) 題意の擬似コードは以下のようになる。

```
1 k ← 0
2 for i 0 to n - 1
3     mask ← 1 << i
4     tmp ← データd & mask
5     if tmpが0でない
6         k ← k + 1
7     end if
8 end for
```

よってこの計算時間は以下のようになる。

$$3 \times n = n$$

よって、$3n$ 単位時間かかる。

(2) 表引きの操作により、シフト演算と論理演算をしなくて済むため、計算時間は n 単位時間となる。

(3) 題意の擬似コードは以下のようになる。データを d とする。

```
1  //データdを2bit整数16組に分ける
2  d ← d - ((d >> 1) & 0x55555555)
3  //データdを4bit整数8組に分ける
4  d ← (d & 0x33333333) + ((d >> 2) & 0x33333333)
5  //データdを8bit整数4組に分ける
6  d ← (d + (d >> 4)) & 0x0f0f0f0f
7  //データdを16bit整数2組に分ける
8  d ← (d + (d >> 8))
9  //データdを32bit整数1組に分ける
10 d ← (d + (d >> 16))
```

この時の計算時間は 5 単位時間となる

(4) 入力 3 ビットの population count 論理回路 P_3 の真理値表は以下のようになる。

表 2.1　論理回路 P_3 の真理値表

入力 I	出力 X
000	00
001	01
011	10
010	01
100	01
101	10
111	11
110	10

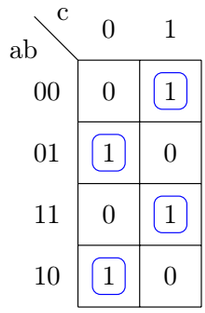


図 2.1　下位 1 ビットカルノー図

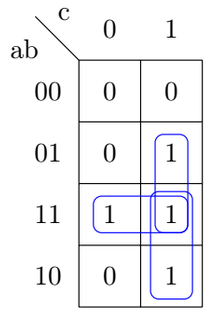


図 2.2　上位 1 ビットカルノー図

$$\text{下位ビットの論理式:}\quad a'b'c + a'bc' + abc + ab'c' = a'(b'c + bc') + a(bc + b'c')$$
$$= a'(bc + b'c')' + a(bc + b'c')$$
$$\text{上位ビットの論理式:}\quad ab + bc + ac = a(b + c) + bc$$
$$= a(b'c')' + bc$$

よってこの時論理回路 P_3 は以下のように設計できる。但し、$F1$ は出力の 2 桁目を表し、$F2$ は出力の 1 桁目を表す。

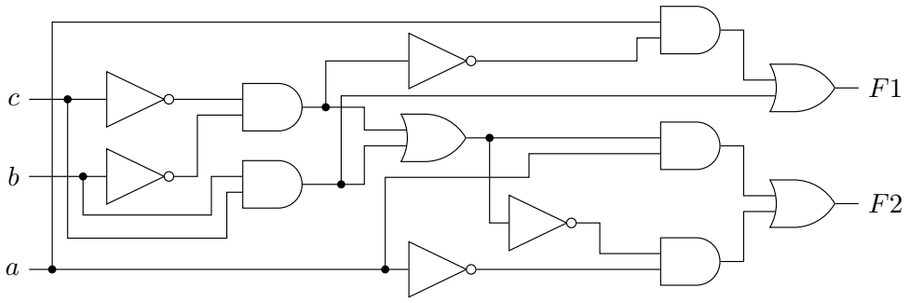


図 2.3　論理回路 P_3

(5) 次に入力 6 ビットの論理回路 P_6 は 2 つの論理回路 P_3 の出力を入力とし、それらの和がこの論理回路の出力となるため、2 ビットの加算器を追加すればよい。よって、その加算器の真理値表は以下のようになる。

表 2.2　下位 1 ビット加算器真理値表

入力 I	出力 X1	繰り上がり C1
00	0	0
01	1	0
11	0	1
10	1	0

表 2.3　上位 1 ビット加算器真理値表

入力 I	前の桁からの繰り上がり C1	出力 X2	繰り上がり C2
00	0	0	0
00	1	1	0
01	0	1	0
01	1	0	1
11	0	0	1
11	1	1	1
10	0	1	0
10	1	0	1

よってカルノー図は以下のようになる。

a \ b	0	1
0	0	[1]
1	[1]	0

図 2.4　下位 1 ビット出力 x カルノー図

a \ b	0	1
0	0	0
1	0	[1]

図 2.5　下位 1 ビット繰り上がり C2 カルノー図

よって、以下のような式になる。

$$X1 = ab' + a'b$$
$$C1 = ab$$

cd \ C1	0	1
00	0	1
01	1	0
11	0	1
10	1	0

図 2.6　上位 1 ビット出力 x カルノー図

cd \ C1	0	1
00	0	0
01	0	1
11	1	1
10	0	1

図 2.7　上位 1 ビット繰り上がり C2 カルノー図

よって、以下のような式になる。

$$X2 = c'd'C1 + c'dC1' + cdC1 + cd'C1' = c'(d'C1 + dC1') + c(d'C1' + dC1)$$
$$C2 = cd + dC1 + cC1 = c(d + C1) + dC1 = c(d'C1')' + dC1$$

ここで、この論理式は論理回路 P_3 と全く同じものであるので、論理回路 P_3 を一つの素子として扱うと、加算器の論理回路は以下のようになる。

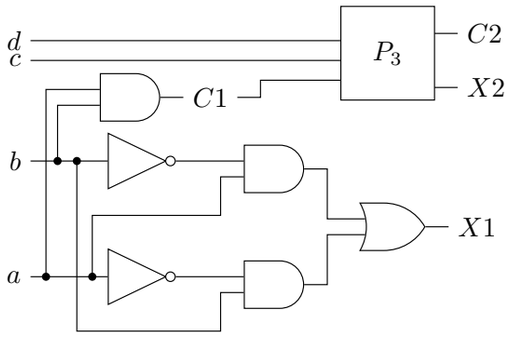


図 2.8　加算器の論理回路

よって、求める論理回路 P_6 は以下のようになる。但し、$F1, F2, F3$ はそれぞれ出力の下 1 桁目, 2 桁目, 3 桁目を表す。

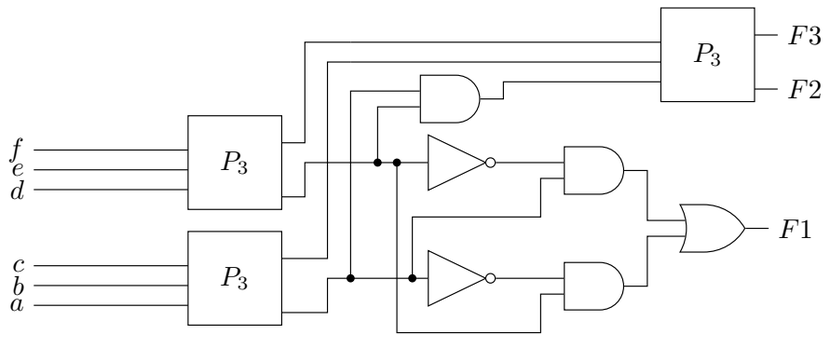


図 2.9　論理回路 P_6

(6)

第 3 問

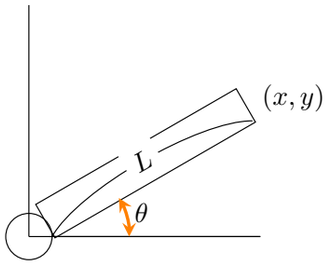
第 3 問　問題文

以下に示す情報システムに関する 8 項目から 4 項目を選択し、各項目を 4〜8 行程度で説明せよ。必要に応じて例や図を用いてよい。

(1) 逆運動学
(2) 隠れマルコフモデル
(3) MinMax 法
(4) NP 完全問題
(5) レイトレーシング
(6) SIMD(Single Instruction Multiple data)
(7) Call by value(値渡し) and call by reference(参照渡し)
(8) 公開鍵暗号

第 3 問　解答例

(1) ロボットのアームの位置座標からロボットのアームの長さとアームの基準面からの角度を求めるもののことである。つまり、以下の図のように位置座標 (x, y) から L, θ を求める学問のことである。



(2) まず N 重マルコフモデルとは前の N 個の出力のみに依存する離散確率過程のことを表すモデルである。$N = 1$ の時は単純マルコフモデルと呼ばれる。つまり、単純マルコフモデルは現在の状態のみで次の状態が確率的に決定されるモデルのことである。但し、単純マルコフモデルは出力が一つしか認められておらず、複雑な出力は表現できないモデルである。そのため、出力を確率的な表現で表すことにより、複数の出力を許すようにしたモデルが隠れマルコフモデルである。

(3) ゲーム理論で使われるアルゴリズムであり、2 人零和ゲームであり、有限の手数で終了することが保証されているのであれば、必ず解くことができるアルゴリズムである。このアルゴリズムは相手の最善手は自分の最悪手であるという考えのもと、自分は最善の手をだし、相手も最善の手を出すと考えた場合に自分の手の場合は最大の評価値を選択し、相手の手の場合は最小の評価値を選択するアルゴリズムである。

		P_2	
		S_1	S_2
P_1	S_1	b_{11} a_{11}	b_{12} a_{12}
	S_2	b_{21} a_{21}	b_{22} a_{22}

よって、上図のような利得表を考えた場合は、2 人零和ゲームであるので $a_{ij} = -b_{ij}$ であり、a_{ij} の値だけ評価すればよく、自分のその戦略を選んだ際の評価値の最小値を考えることにより、相手の最善手を考え、その最小値の中で最大となるものの戦略を選択すれば自分の最善手を選択することになるので、評価対象 P_1 が取るべき戦略は $S_{\max_i \min_j a_{ij}}$ となるアルゴリズムである。

(4)