平成 20 年度

東京大学大学院情報理工学系研究科

コンピュータ科学専攻

入学試験問題

# 専門科目 I

解答

東京大学理学部情報科学科 2008

2012 年 8 月 7 日

## 問題 1

### (1)

図を描いて比を計算すれば一発

$P_1$,$P_2$, $P_3$ 全てにおいて、

$$(x', y') = (xz_0/z, yz_0/z)$$

となる。($x$, $y$, $z$ にはそれぞれの点の値を入れる)

### (2)

ベクトルを用いる。平面上において、ある三本のベクトルがあって、$a$, $b$, $p$ としたとき、($a$ は $\overrightarrow{OA}$、b は $\overrightarrow{OB}$、p は $\overrightarrow{OP}$ とする。) 点 $P$ が $\triangle OAB$ の内側にあるとは、ある実数 $s$,$t$ を用いて

$$p = sa + tb$$

と表せ、$s > 0$, $t > 0$, $s + t < 1$ である。よって、三本のベクトル $\overrightarrow{P_1'P_2'}$、$\overrightarrow{P_1'P_3'}$、$\overrightarrow{P_1'}$, $P'$ で同様のことを確かめればよい。

### (3)

法線ベクトルは二本のベクトル

$$\vec{a} = (ax, ay, az), \vec{b} = (bx, by, bz)$$

の外積によって求める。具体的には

$$\vec{a} \times \vec{b} = (aybz - azby, azbx - axbz, axby - aybx)$$

例えば P1 における法線ベクトルを求めたければ、$\overrightarrow{P_1P_2}$、$\overrightarrow{P_1P_3}$ に関して同様の操作を行えばよい。その法線ベクトルを $\vec{n}$ とし、

$$\vec{n} = (u, v, w)$$

とすると、三角形 $P_1P_2P_3$ を含む平面の式は

$$u(x - x_1) + v(y - y_1) + w(z - z_1) = 0$$

で表される。

### (4)

これもベクトルで考えればよい。以下のベクトルは $\overrightarrow{P'}$、$\overrightarrow{P_1'}$、$\overrightarrow{P_2'}$ のように一点しか書かれていない場合は原点を根とするものとする。投影面上の点 $P'$ がある実数 $s$ と $t$ を用いて、

$$\overrightarrow{P_1'P'} = s\overrightarrow{P_1'P_2'} + t\overrightarrow{P_1'P_3'}$$

$$(\overrightarrow{P'} - \overrightarrow{P_1'}) = s(\overrightarrow{P_2'} - \overrightarrow{P_1'}) + t(\overrightarrow{P_3'} - \overrightarrow{P_1'})$$

$$\overrightarrow{P'} = (1-s-t)\overrightarrow{P'_1} + s\overrightarrow{P'_2} + t\overrightarrow{P'_3}$$

と書くことが出来る。ここで求めた $s$,$t$ を用いて、点 $P$ の座標は

$$\overrightarrow{P} = (1-s-t)\overrightarrow{P_1} + s\overrightarrow{P_2} + t\overrightarrow{P_3}$$

と書くことが出来る。式を求めよとあるのでこれでは不十分かもしれない。もっと良い方法、計算した結果があれば追記をお願いしたい。

## 0.1 (5)

・点 $P$ が $P_1$と $P_2$ の中点かつ点 $P'$ が $P'_1$と $P'_2$ の中点となる条件。点 $P'$ の式・・・

$$\begin{aligned}(x', y') &= 1/2(x'_1, y'_1) + 1/2(x'_2 + y'_2) \\ &= 1/2(x_1 z_0/z_1, y_1 z_0/z_1) + 1/2(x_2 z_0/z_2, y_2 z_0/z_2) \\ &= ((x_1 z_2 + x_2 z_1) z_0/2 z_1 z_2, (y_1 z_2 + y_2 z_1) z_0/2 z_1 z_2)\end{aligned}$$

点 P の式・・・$(x, y, z) = 1/2(x_1, y_1, z_1) + 1/2(x_2, y_2, z_3) = ((x_1 + x_2)/2, (y_1 + y_2)/2, (z_1 + z_2)/2)$ これが投影面に落とされたときの式・・・

$$(x', y', z_0) = ((x_1 + x_2) z_0/(z_1 + z_2), (y_1 + y_2) z_0/(z_1 + z_2), z_0)$$

この二つの表現が同じ値を持つことが題中の条件なので、

$$(x_1 z_2 + x_2 z_1)/2 z_1 z_2 = (x_1 + x_2)/(z_1 + z_2)$$

$$(y_1 z_2 + y_2 z_1)/2 z_1 z_2 = (y_1 + y_2)/(z_1 + z_2)$$

これは幾何学的には、辺 $P_1P_2$ が線分 $OP$ と直行するということを意味する。

点 $P$ が $\triangle P_1P_2P_3$ の重心かつ点 $P'$ が $\triangle P'_1P'_2P'_3$ の重心となる条件。同様に点 $P'$ の式・・・

$$\begin{aligned}(x', y') &= 1/3(x'_1, y'_1) + 1/3(x'_2, y'_2) + 1/3(x'_3, y'_3) \\ &= ((x'_1 + x'_2 + x'_3)/3, (y'_1 + y'_2 + y'_3)/3) \\ &= ((x_1 z_2 z_3 + z_1 x_2 z_3 + z_1 z_2 x_3) z_0/3 z_1 z_2 z_3, (y_1 z_2 z_3 + z_1 y_2 z_3 + z_1 z_2 y_3) z_0/3 z_1 z_2 z_3)\end{aligned}$$

点 $P$ の式・・・

$$(x, y, z) = ((x_1 + x_2 + x_3)/3, (y_1 + y_2 + y_3)/3, (z_1 + z_2 + z_3)/3)$$

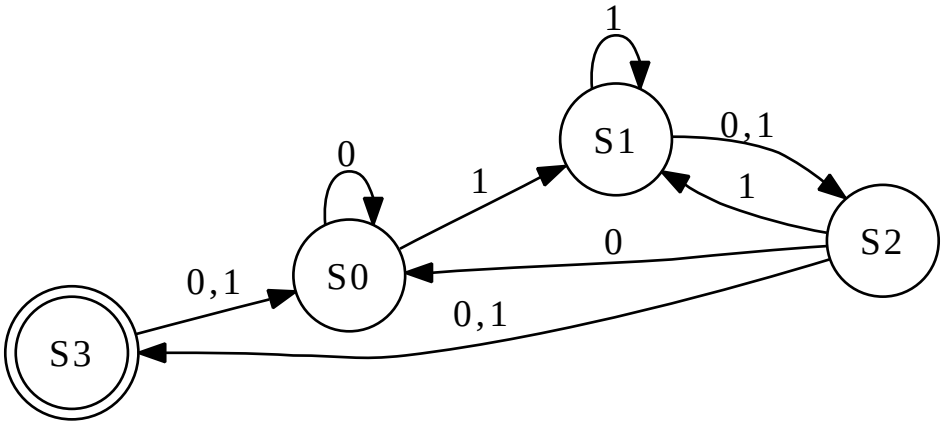
よってここから条件を求めると、

$$(x_1 z_2 z_3 + z_1 x_2 z_3 + z_1 z_2 x_3)/3 z_1 z_2 z_3 = (x_1 + x_2 + x_3)/(z_1 + z_2 + z_3)$$

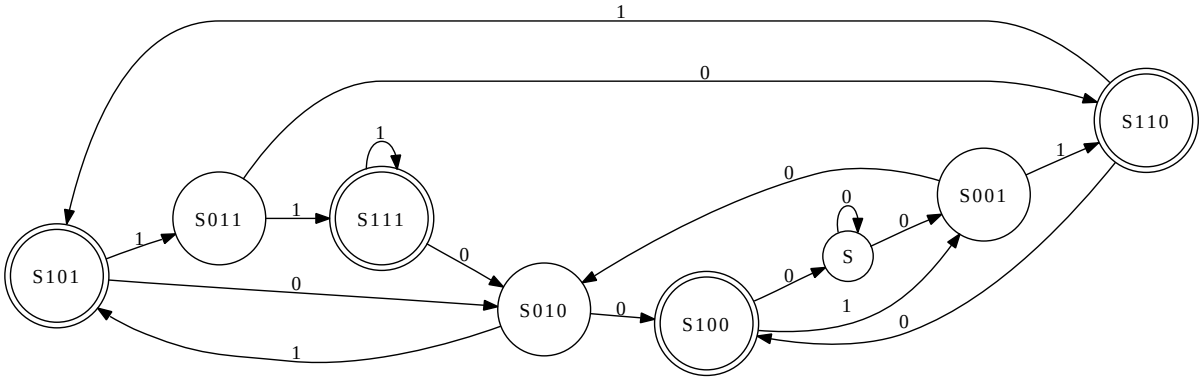
$$(y_1 z_2 z_3 + z_1 y_2 z_3 + z_1 z_2 y_3)/3 z_1 z_2 z_3 = (y_1 + y_2 + y_3)/(z_1 + z_2 + z_3)$$

## 問題 2

(1)




(2)




(3)

(2) でわかったとおり、決定性オートマトンにおいて現在受け付けた文字列の後ろから $N$ 番目の文字が”1”であるかどうかを保障するためには、後ろから $N$ 文字全ての状態が少なくとも必要であるため、$L_n$ を受理する決定性オートマトンは、$0,1$ どちらかを表す文字、$N$ 文字分である $2^N$ 個の状態数を最低限でも持つことになる。

## 問題 3

### (1)

ヒープとは，各ノードが 1 つの数字を持つ二分木で，親ノードは子ノードより必ず小さい数字を持つようにしたものである．

$n$ 個の数字がヒープに含まれる時，数字の追加，最小の数字の取得・削除などが $O(\log n)$ 時間で行える．

**解説**

非常に基礎的かつシンプルなデータ構造なので，要チェックだと思います．

### (2)

**$\frac{1}{4} \leq |X_{small}|$ であること**

$y_i \leq a$ なる $i$ の個数は $\lceil \frac{m}{2} \rceil$ である．従って，$y_i \leq a$ かつ $|X_i| = 5$ を満たす $i$ の個数は少なくとも $\lceil \frac{m}{2} \rceil - 1$ 個である．そのような各 $i$ に関して， $y_i$ は $X_i$ の中間値かつ $y_i \leq a$ より，$|X_i|$ の少なくとも 3 つの要素が $|X_{small}|$ に含まれる．よって，$\lceil x \rceil \geq x$ より，以下が成立する．

$$
\begin{aligned}
|X_{small}| &\geq 3\left(\left\lceil \frac{m}{2} \right\rceil - 1\right) \\
&\geq 3\left(\frac{m}{2} - 1\right) = \frac{3}{2}m - 3 = \frac{3}{2}\left\lceil \frac{|X|}{5} \right\rceil - 3 \\
&\geq \frac{3}{2}\frac{|X|}{5} - 3 = \frac{3}{10}|X| - 3
\end{aligned}
$$

ここで，$N \leq 60$ とすると，$|X_{small}| \geq \frac{1}{4}$ となる．

**$|X_{small}| \leq \frac{3}{4}$ であること**

同様に，$y_i \leq a$ を満たす $i$ の個数は高々 $\lceil \frac{m}{2} \rceil$ 個なので，$\lceil x \rceil < x + 1$ より，以下が成立する．

$$
\begin{aligned}
|X_{small}| &\leq 3\left\lceil \frac{m}{2} \right\rceil \\
&< 3\left(\frac{m}{2} + 1\right) = \frac{3}{2}m + 3 = \frac{3}{2}\left\lceil \frac{|X|}{5} \right\rceil + 3 \\
&< \frac{3}{2}\left(\frac{|X|}{5} + 1\right) + 3 = \frac{3}{10}|X| + \frac{9}{2}
\end{aligned}
$$

ここで，$N \leq 10$ とすると，$|X_{small}| \leq \frac{3}{4}$ となる．

**解説**

繰り上げの記号に関する不等式と，中間値をとる操作に関する不等式を，いっぱい組み合わせる．面倒．

## (3)

8 行目の再帰呼び出しに関しては，高々 5 個のため，一度の呼び出しを定数時間とでき，再帰呼び出しとして考える必要がない．10 行目の再帰呼び出しの引数のサイズは $\left\lceil \frac{n}{5} \right\rceil$ である．17 行目あるいは 20 行目の再帰呼び出しの引数のサイズは，(2) より，高々 $\left\lceil \frac{3}{4}n \right\rceil$ と考えられる．以上より，

$$T(n) = O(n) + T\left(\left\lceil \frac{n}{5} \right\rceil\right) + T\left(\left\lceil \frac{3}{4}n \right\rceil\right)$$

## (4)

十分大きい定数 $\alpha > 0$ が存在して，(3) の式は，以下のように書ける．

$$T(n) = \alpha n + T\left(\left\lceil \frac{n}{5} \right\rceil\right) + T\left(\left\lceil \frac{3}{4}n \right\rceil\right)$$

十分大きい $N$ に対し，$T(n) \leq N\alpha n$ とでき，従って $T(n) = O(n)$ と言えることを数学的帰納法により示す．

**$n \leq N$ の時**

最悪 $O(N \log N)$ 時間かかるが，$N$ は定数なので，これも定数時間と考えることができる．$\alpha$ が十分大きければ，条件を満たす．

**$n > N$ かつ $n$ 未満について条件が満たされていることが分かっている時**

$$\begin{aligned} T(n) &\leq \alpha n + N\alpha \left\lceil \frac{n}{5} \right\rceil + N\alpha \left\lceil \frac{3}{4}n \right\rceil \\ &\leq \alpha n + N\alpha \left(\frac{n}{5} + 1\right) + N\alpha \left(\frac{3}{4}n + 1\right) \\ &= N\alpha n - \left(\left(\frac{N}{20} - 1\right)\alpha n - 2N\alpha\right) \end{aligned}$$

となり，$n > N$ と $N$ が十分大きいことより

$$\begin{aligned} \left(\frac{N}{20} - 1\right)\alpha n - 2N\alpha &> \left(\frac{N}{20} - 1\right)\alpha N - 2N\alpha \\ &= \left(\frac{N}{20} - 3\right)\alpha N > 0 \end{aligned}$$

と分かるので，条件を満たす．

**解説**

ちょっと強引な感じで荒く抑えた気がする．

## 問題 3

### (1)

ヒープ構造とは、親ノードの保持する値が子ノードのそれよりも小さくなっている二分木である。N 個から k 番目に大きいデータを取り出すのに、$O(N + k \log N)$ の計算量で済むという利点をもつ (ソートしてから k 番目を参照すると $O(N \log N)$ かかる。しかし、本問はそれを $O(N)$ で行うアルゴリズムであるためありがたみがない)。

### (2)

$\frac{1}{4}$ と $\frac{3}{4}$ は本質的な値ではなく、$\frac{4}{5}$ より小さい値で上からおさえられればよい。

$y_i$ の中で a より小さい数からなる列 $Y_{small}$ を考えると、$|Y_{small}| \geq \lceil \frac{m}{2} \rceil - 1$。 $y_i$ の性質から $|X_{small}| \geq 3|Y_{small}| - 2$。$\lceil \frac{m}{2} \rceil \geq \frac{m+1}{2}, m \geq \frac{|X|+4}{5}$ を代入して $|X_{small}| \geq \frac{3}{10}|X| - 19/5$。N を十分大きくとると $\frac{1}{20}|X| > 19/5$ であるため $|X_{small}| \geq \frac{1}{4}|X|$、右の不等式も同様である。

### (3)

同様に、$\frac{1}{4}|X| \leq |X_{large}| \leq \frac{3}{4}|X|$ がいえる。各ステップでの計算量を考えると、以下の式を得る。

$T(n) = C + \frac{n}{5} + T(\frac{n}{5}) + T(\frac{3}{4})$ (C は定数)

上式は $T(n) = 4n + 20C$ で成立するので、O(n) である。

## 問題 4

### (1)

Q = DFF(D,clk)
out = (Q and (not clk)) or ((not Q) and clk)

### (2)

- 同期信号が含まれている。別の方法で同期を取る必要がない。
- 信号の値として 0 と 1 しか取らない。複数の段階を取る方式に比べてハードウエアが単純になり、品質の高くないケーブルでもエラーが発生しにくい。

この他にもシリアル通信一般の特徴やエンコーダ・デコーダが比較的単純などの特徴が挙げられますが、(3) との兼ね合いで上記のものが良いと思います。

### (3)

データ 1 の時信号 1 データ 0 の時信号 0 とする。このままでは同じデータが続いたときに同期が取れなくなってしまう。そこで、データを一定のまとまりごとにより長く送信に都合の良い bit 列に変換する。例えば 4bit を 5bit に変換して伝送すれば、5bit 中に信号の変化が 2 回以上現れるようにすることができる。この場合要求する帯域はデータの 125% となり、また (2) で挙げた特徴は維持している。一方で、変換テーブルが必要。まとまった bit 単位でないとデコードできない。デコーダが変換の区切りを認識する必要がある。などエンコーダ・デコーダの構成は複雑になってしまう。

# 専門科目 II

# 問題 2

## 解答

(1)(2)(3) 全部まとめて回答したほうがいいでしょう。

閉包とか難しい言葉を使ってきますが、単に何回か繰り返したらの意味です。$\overset{*}{\to}$ の記号自体はおなじみですね。あとはスタックの書き方で文字をいきなり並べるのが気持ち悪いですがまあ頑張りましょうw

さて、帰納法ですが、書き換え規則に関する帰納法を使う人もいるかも知れませんが、E(あるいは後置記法) の構造に関する帰納法でゴリゴリ解く方が確実に解けるのでいいでしょう。

で、それを決めると今度は証明の中核に、

「$E_1 + E_2$ のとき、$E_1 + E_2 \Rightarrow w_1 w_2+$ となって、$E_1$,$E_2$ のときに○○が成り立つから、$(w_1 w_2 +$残り$,$残り$)$ が $(+$ 残り$, v_2 v_1$残り$)$ の形になって…」

というのを持って来たくなるので、これに合わせて形を変形します。

あとは数学的帰納法を使うときに、「○○を、○○ (命題のどの部分) の、○○に関する数学的帰納法によって示す」を明示するようにすると採点官の印象が良くなると思います。

ということで、

以下の一般化された命題を考え、それを E の式の構造に関する数学的帰納法を使って証明する。

#### 補題 (一般化された命題)

eval(E)=v かつ $E \Rightarrow w$ であるならば、$(w\ w_r, s) \overset{*}{\to} (w_r, v\ s)$ となる。

#### 証明

補題を E の式の構造に関する数学的帰納法を使って証明する。

■(i)E=c **のとき**　c の値を $v_c$ とおく。eval(c)=v=$v_c$ かつ $c \Rightarrow w = c$ のとき、書換え規則より、

$(c\ w_r, s) \to (w_r, v_c\ s)$

となるため、

$(w\ w_r, s) \overset{*}{\to} (w_r, v\ s)$ となる。

よって、E=c のとき補題は成立

■(ii)E=$E_1 + E_2$ **のとき**　式 $E_1\ E_2$ に対して、補題が成立すると仮定する。E=$E_1 + E_2$ のときも補題が成立することを示す。

このとき、$E_1 \Rightarrow w_1$,$E_2 \Rightarrow w_2$,eval$(E_1)$=$v_1$,eval$(E_2)$=$v_2$ とおく。

v=eval(E)=eval$(E_1 + E_2)$=$v_1 + v_2$=eval$(v_1 + v_2)$

帰納法の仮定より、$(w_1\ w_{r1}, s_1) \overset{*}{\to} (w_{r1}, v_1\ s_1) \cdots$(*1),$(w_2\ w_{r2}, s_2) \overset{*}{\to} (w_{r2}, v_2\ s_2) \cdots$(*2) が成立する。

このとき、$E \Rightarrow w = w_1 w_2+$ となる。

よって、$(w\ w_r, s) = (w_1 w_2 +\ w_r, s)$ となり、まず (*1) を適用して、

$(w_1 w_2 +\ w_r, s) \overset{*}{\to} (w_2 +\ w_r, v_1\ s)$

次に (*2) を適用して、

$(w_2 +\ w_r, v_1\ s) \overset{*}{\to} (+\ w_r, v_2\ v_1\ s)$

そして、書換え規則より、
$(+\ w_r, v_2\ v_1\ s) \to (w_r, v\ s)$
よって、($\overset{*}{\to}$ が推移閉包であることから ← この一文は余計かな。)
$(w\ w_r, s) \overset{*}{\to} (w_r, v\ s)$
となり、E=$E_1 + E_2$ のときも補題が成立することが示された。

■(iii)E=$E_1 \times E_2$ **のとき**　式 $E_1$ $E_2$ に対して、補題が成立すると仮定したとき、E=$E_1 \times E_2$ のときも補題が成立することも、(ii) の + を × にして、全く同様に示すことができる。

■**(結論)**　よって、(i) から (iii) より、E の式の構造に関する数学的帰納法から、補題が示された。

**与題**

補題を使って与題を示す。
補題の、$(w\ w_r, s) \overset{*}{\to} (w_r, v\ s)$ の中の $w_r$,$s$ をそれぞれ $\epsilon$ とすれば与題が得られる。
よって、与題も示された。
Q.E.D.

## 問題 3

### (1)

$U'_{min} = (U[2], U[3], U[4], U[7])$ と $V'_{min} = (V[1], V[2], V[5], V[6])$ のときに最小値 $min(U, V) = -0.8$

**コメント**

同じ座標がなくて、距離 1 以外のは使う必要ないことを考えれば手計算でも結構簡単。
距離 1 の点のペアはこの４つしかなく、全部同時に使えるので最小。

### (2)

ある $l$ に対して、

$$\begin{aligned} min(U[1 \dots i_l], V[1 \dots j_l]) &= f((U[i'_1], U[i'_2], \dots, U[i'_{l'}]), (V[j'_1], V[j'_2], \dots, V[j'_{l'}])) \\ &< f((U[i_1], U[i_2], \dots, U[i_l]), (V[j_1], V[j_2], \dots, V[j_l])) \end{aligned}$$

であったと仮定すると、

$$f((U[i'_1], \dots, U[i'_{l'}], U[i_{l+1}], \dots, U[i_k]), (V[j'_1], \dots, V[j'_{l'}], V[j_{l+1}], \dots, V[j_k])) < f(U'_{min}, V'_{min})$$

となり、$(U'_{min}, V'_{min})$ が最小であったことに反する。
よって、任意の $l$ に対して等号が成立する。

### (3)

(2) の性質より、$a[i][j] = min(U[1 \dots i], V[1 \dots j])$ と定義すると、a に関して次の式が成り立つ。

$$\begin{aligned} &a[i][j] = min(a[i-1][j-1] + dist(U[i], V[j]) - c, a[i][j-1], a[i-1][j])(i \geq 1, j \geq 1) \\ &a[i][0] = 0 \\ &a[0][j] = 0 \end{aligned}$$

この式に従って動的計画法を行うことにより、$O(|U||V|)$ で $min(U, V) = a[|U|][|V|]$ の値を求めることが出来る。

## 問題 4

### (1)

スループットとはプロトコルのオーバーヘッドなどを覗いた実質的な通信速度のことをいう。

パケットのサイズは 1000 バイト固定でそのうち 6 バイトがヘッダ情報として使用されるため、1 パケットで転送できるデータサイズは 994 バイト固定。

送信側→受信側の DATA パケット転送にかかる時間が $1000/10^6 + 10 \times 10^{-3} = 0.011$sec

受信側→送信側の ACK パケットも 1000 バイトの固定長[*1]なので同じく 0.011sec

以上より[*2]

$$994/(0.011 + 0.011) = 3.6 \times 10^5 \text{byte/sec}$$

### (2)

**受信側の処理**

DATA パケットを受け取ったら、ACK パケットを送信側に送り返す。ACK パケットの Sequence Number は受信したパケットの Sequence Number とする。**ただし受信済みの DATA パケットと同じ Sequence Number を持っていた場合は ACK パケットを送り返すのみでデータの中身は破棄する。**

**送信側の処理**

Sequence Number の初期値は乱数で決める。DATA パケットを送信する度に、当該パケットの ACK パケットを受信するまで、次のパケットの送信を待つ。DATA パケットを送信する毎に Sequence Number に (modulo $2^{24}$ で)1 を足す。**ただし一定時間待機しても ACK パケットが返ってこない場合は同じ内容の DATA パケットを再送するものとし、このときは Sequence Number に 1 を足す前の値を設定する。**

### (3)

**受信側の処理**

DATA パケットを受け取ったら、ACK パケットを送信側に送り返す。ACK パケットの Sequence Number は受信したパケットの Sequence Number とする。**ただし受信済みの DATA パケットと同じ Sequence Number を持っていた場合は ACK パケットを送り返すのみでデータの中身は破棄する。**

**送信側の処理**

Sequence Number の初期値は乱数で決める。DATA パケットを送信する度に、当該パケットの ACK パケットを受信するまで、次のパケットの送信を**待たず、Sequence Number に (modulo $2^{24}$ で)1 を足して次のパケットを送信する。それぞれの DATA パケットについて、送信から一定時間の間にその Sequence Number を持った ACK パケットが返ってこなかった場合は同じ内容の DATA パケットを再送する。**

---

[*1] 本当か？本当だよね？

[*2] 有効数字は 2 桁でいいのかどうか。通信遅延の 10msec が 2 桁なので 2 桁にしておいた。

なお受信側へのパケットの到着順序は保証されない。[*3]

## (4)

### (a)

```
+-----------------+-------------+-----------+-------------+--------------+
| Sequence Number | Packet Type | Data Size | Destination | data         |
|    (3 Bytes)    |   (1 Byte)  | (2 Bytes) |   (1 Byte)  |              |
+-----------------+-------------+-----------+-------------+--------------+
Packet Type: 1 DATA
             0 ACK
Destination:     +----------+----------+----------+
                 | Not used |    X     |    Y     |
             MSB | (2 Bits) | (3 Bits) | (3 Bits) | LSB
                 +----------+----------+----------+
```

X 座標と Y 座標で送信先の計算機を指定する。

### (b)

**受信側の処理**

**自分宛の**DATA パケットを受け取ったら、ACK パケットを送信側に送り返す。ACK パケットの Sequence Number は受信したパケットの Sequence Number とする。**ただし受信済みの DATA パケットと同じ Sequence Number を持っていた場合は ACK パケットを送り返すのみでデータの中身は破棄する。**

**送信側の処理**

Sequence Number の初期値は乱数で決める。DATA パケットを送信する度に、当該パケットの ACK パケットを受信するまで、次のパケットの送信を**待たず、Sequence Number に (modulo $2^{24}$ で)1 を足して次のパケットを送信する。それぞれの DATA パケットについて、送信から一定時間の間にその Sequence Number を持った自分宛の ACK パケットが返ってこなかった場合は同じ内容の DATA パケットを再送する。**

**ルーティング処理**

全ての計算機は自分自身の X 座標と Y 座標を認識しており、

- X 座標の小さくなる方向を「左」、大きくなる方向を「右」
- Y 座標の小さくなる方向を「上」、大きくなる方向を「下」

と呼ぶものとする。

パケットを受け取ったら以下の条件分岐のうち最初に合致したもの 1 つを実行する。

- 受け取ったパケットの宛先座標が自分のものだった場合、受信側として処理を行う。
- Y 座標が自分より小さかった場合、上方に接続している計算機へパケットを流す。

[*3] 順序を保証せよとはどこにも書かれていないので都合のいい方に解釈した

- Y 座標が自分より大きかった場合、下方に接続している計算機へパケットを流す。
- X 座標が自分より小さかった場合、左方に接続している計算機へパケットを流す。
- X 座標が自分より大きかった場合、右方に接続している計算機へパケットを流す。