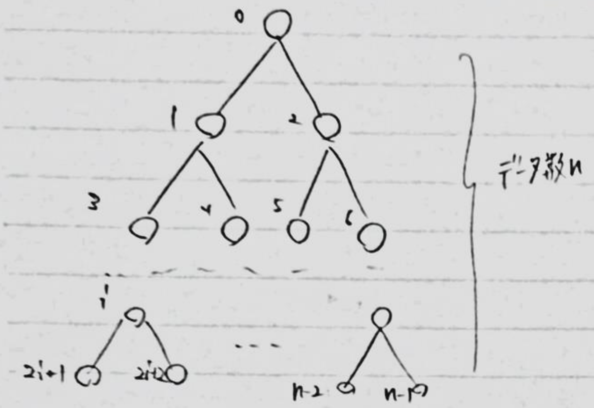
[1] アルゴリズムとプログラミング (文責: 吉見)

No.

Date



(1) 木構造をもちいたソート → ヒープソート

概要: ① n個のデータで最大となるものを ヒープ特性(親は左右の子より大)を 用いて根に持っていく.

② 根(0番)と n個目(n-1番)を入れかえ

③ 0〜n-2 のn-1個でヒープ特性を 用いて最大値を根に持っていく

④ 根(0番)と n-1個目(n-2番)を入れかえ

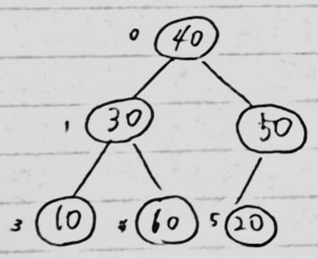
⋮

すべておわると ソーティングできる

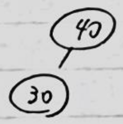
(2) l.28 uph(d,i) : i番目のデータ(仮にx)と親を比べて. 親が小さければ入れかえ

→ 次にまたそのxと親を比べる.

⋮

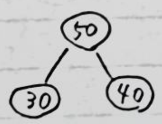


「i=1」 current は30. 親は 40 → 入れかえなし

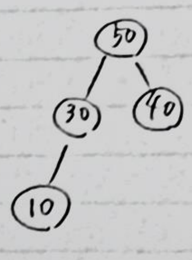


「i=2」 current は50 親は40 → 入れかえ

根に親はなし → 終了

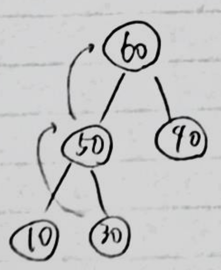


同様に



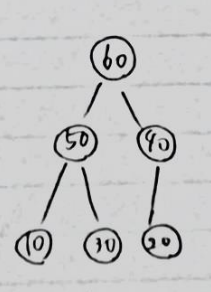
i=3
スワップなし

⇒



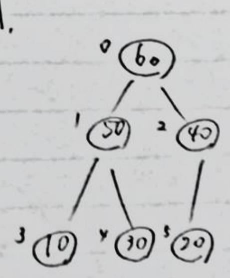
i=4
60が根までスワップ

⇒



i=5
スワップなし

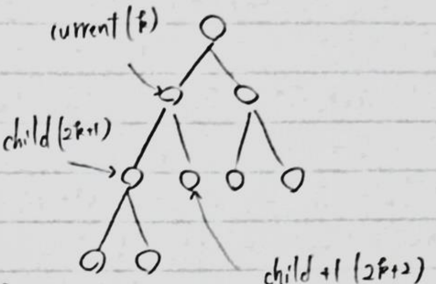
A.



No.
Date . .

(3) current : 対象のデータ

l10 child = current×2+1 = 2k+1 = 左の子



l11 (ア) : child+1 というデータが存在するか
（配列外参照を避けるため）

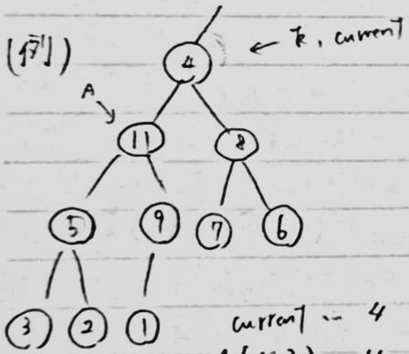
(イ) : 左の子と右の子を比較し大きい方を選択
Aとおく

l12 (ウ) current とその子A とのデータ比較 → 子が大きいとスワップ

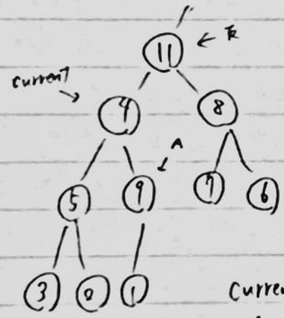
（l14 で current がヒープ特性をみたしながら下にもっていくようになる）

A.
データ current が左右の子のどちらよりも大きいという関係
（これをヒープ特性）

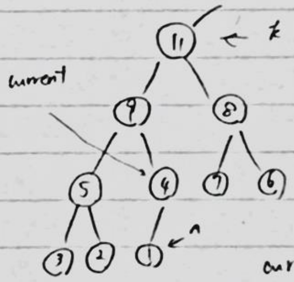
(例)



current … 4
A(左の子) … 11
スワップ



current … 4
A(右の子) … 9
スワップ



current … 4
A(左の子) … 1
スワップなし → downh 終了

※ downh によりできるのは、kに与えた番号以下のヒープ特性をみたすこと
i は、ヒープの内使うノードの数（ソートが終わった部分は含めない）

(4)

uph : 木の高さはデータ n に対して $\lfloor \log_2 n \rfloor$ （$\lfloor\ \rfloor$ は床関数（小数切り捨て））

n つのデータを高々 $\log_2 n$ 回スワップ（スワップは定数時間処理） → $O(n \log n)$

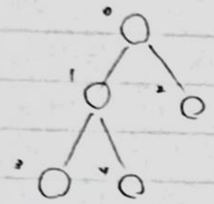
downh : uphと同様に n つのデータをそれぞれ $\log_2 n$ 回スワップ → $O(n \log n)$

∴ $O(n \log n)$

(5) (5-1) ループ回数を減らす.

downhは葉ノードに対して実行しても意味がない(葉には子がいない)

葉ノードを持つノードの番号は「$n/2-1$」でわかる ("/"は商(int)のみ求める)



$n=5$ 子をもつ番号最大のノード: 1

$(n/2)-1 = 2-1 = \underset{\sim}{1}$

$n=6$ 子をもつ番号最大のノード: 2

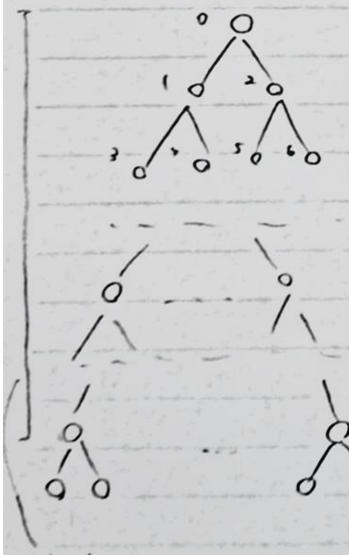
$(n/2)-1 = 3-1 = \underset{\sim}{2}$

よって $\underline{(あ) = n/2-1}$

downhは前述のとおり. $\underline{(い) = d}$

(う) $\underline{n}$ (まだすべてのノードがヒープ特性をみたしていない) (え) $\underline{i}$ (対象ノードは i)

(5-2)



ノード数	swap回数
$2^0=1$	h
$2^1=2$	$h-1$
$2^2=4$	$h-2$
⋮	⋮
2^k	$h-k$
⋮	⋮
2^{h-1}	1
2^h	0

高さh

データ数nの木は左図, 各ノードスワップする最大回数はそのノードの高さと木の高さの差

(最悪はすべてのノードが葉にいくまでよりwhileをループするとき)

よって $T(n) = \sum_{k=0}^{h} 2^k (h-k)$

$h-k=d$とおくと $\begin{pmatrix} k\ 0\to h \\ d\ h\to 0 \end{pmatrix}$

$$T(n) = \sum_{d=0}^{h} 2^{h-d} d = 2^h \sum_{d=0}^{h} \frac{d}{2^d}$$

問題文の式より

$$T(n) = 2^h \left(2 - \frac{2+h}{2^h}\right)$$

No.

Date . .

(5-2) 続き　$T(n) = 2\cdot 2^h - (2+h)$

高さhは (4) でも述べたように $h = \lfloor \log_2 n \rfloor$ なので

$T(n)$ のオーダーは　$O(2^h) = O(2^{\log_2 n}) = \underline{O(n)}$

(6) 降順にする

→ データ比較で 親が左右のどちらの子よりも 小さいように変えるだけ

(ア) データ比較でない (child+1 が配列外参照かどうかみてるだけ)

そのまま　$\underline{child+1 < n}$

(イ) データ比較. 左右の内小さい方をとるので 不等号反転　$\underline{d[child] > d[child+1]}$

(ウ) 同じくデータ比較 不等号反転　$\underline{d[current] > d[child]}$

(エ) 同じくデータ比較 〃　$\underline{d[parent] > d[current]}$

「令和2年度のアルゴリズムとプログラミングについて.」

過去をさかのぼればわかるように、10年間マージソートとヒープソートが出ていなかったため予想はできた. (二分ヒープは出たが、ソーティングでなかった (平成23年度)) 次はマージかも…??

問題の難易度的にはやや難と思われる. uph.と downh の動作や木構造などすこしややこしい、落ちつけば解けるが、本番は緊張するので焦らないこと。

2. 計算機システムとシステムプログラム <文責：~~寺井~~>
近藤

(1)
(1-1)
a: エ (表現できる数の範囲が広い)

b: イ (浮動小数点)

c: ウ (演算回路が簡素になる)

d: ア (固定小数点)

e: ク (0.1)

(1-2)
$e = 15, 0$はそれぞれ∞扱いになるので、そこは注意して浮動小数点について考えていく

(1-2-1)
正の整数より $s = 0$

最大値は $e = 14 \cdots 2^7$

したがって、最大の正の整数は (0 1110 111111111) で2ビット表現できる

$1.111111111 * 2^7 = 11111111.11 = 255.75$

(1-2-2)
正の整数より $s = 0$

最小値は $e = 1 \cdots 2^{(-6)}$

したがって、最小の正の整数は (0 0001 000000000) で2ビット表現できる

$1.000000000 * 2^{(-6)} = 1/2^6 = 1/64 = 0.015625$

(1-2-3)
負の整数より $s = 1$

$36 = 100100= 1.00100 * 2^5$

$0.66 = 0.10101000..$

$e - 7 = 5 \rightarrow e = 12$

したがって (1 1100 001001010)

(2)
(2-1)
a: キ (アドレス空間)

b: カ (仮想アドレス)

c: オ (実アドレス)

d: ア (主記憶)

e: コ (ページング)

f: サ (セグメント)

g: シ (外部断片化)
h: ウ (内部断片化)
i: イ (ページ)

(2-2)
<LRU>

○	○	○		○	○	○		○		○		○
0	1	2	0	3	1	4	3	2	3	1	2	4
0	0	0	0	0	0	4	4	4	4	1	1	1
	1	1	1	3	3	3	3	3	3	3	3	4
		2	2	2	1	1	1	2	2	2	2	2

<FIFO>

○	○	○		○		○				○	○	
0	1	2	0	3	1	4	3	2	3	1	2	4
0	0	0	0	3	3	3	3	3	3	3	2	2
	1	1	1	1	1	4	4	4	4	4	4	4
		2	2	2	2	2	2	2	2	1	1	1

(2-3)
性能低下を10%以内に抑える
→アクセス時間がページフォールトがない時と比較して最大でも10%以下の増加に抑える
1命令で主記憶に2回アクセスするため、最大で2回のページフォールトが発生する
ページフォールトが起きる確率をPとすると、起きない確率は(1-P)となる
したがって
2*2*(1-P)^2 + 2*(P*(1-P))*(2+(2+8*10^3)) + 2*(2+ 8*10^3)*P^2 = (2+2)*1.1
答: 2.5*10^(-5) = 2.5*10^(-3) %

(2-4)
ページ枠を増やすことで時間的参照局所性を多く持たせることで、ページフォールトを削減する

<所感>

今年度は計算機システムの問題が他の大問と比べて比較的に解きやすいものだったと思われる。
(アルゴリズムや電子論理回路の問題が難易度が高め…)
問われたこともそこまで突飛なものは少ないと思われるため、ここを落ち着いて解いて得点源にしていきたい。
これは違う研究室の人が言っていたが、もしかしたら来年はパイプラインが出るかもしれない。。。？？？

大問４　計算理論

文責：寺井

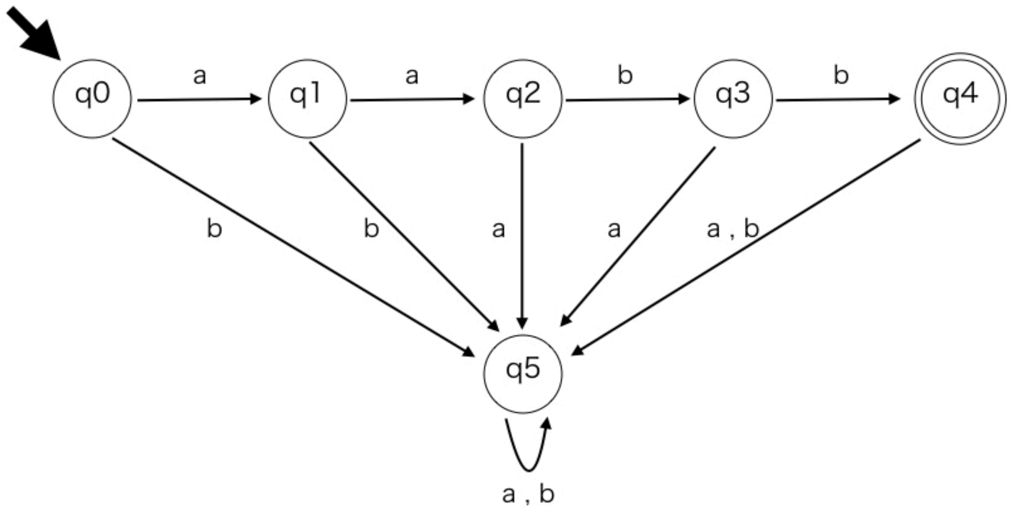
(1)

(1-1)　(A) (a,0)/00　　　　(B) (ε,0)/0 もしくは (b,0)/ε　　　(C) (b,0)/ε

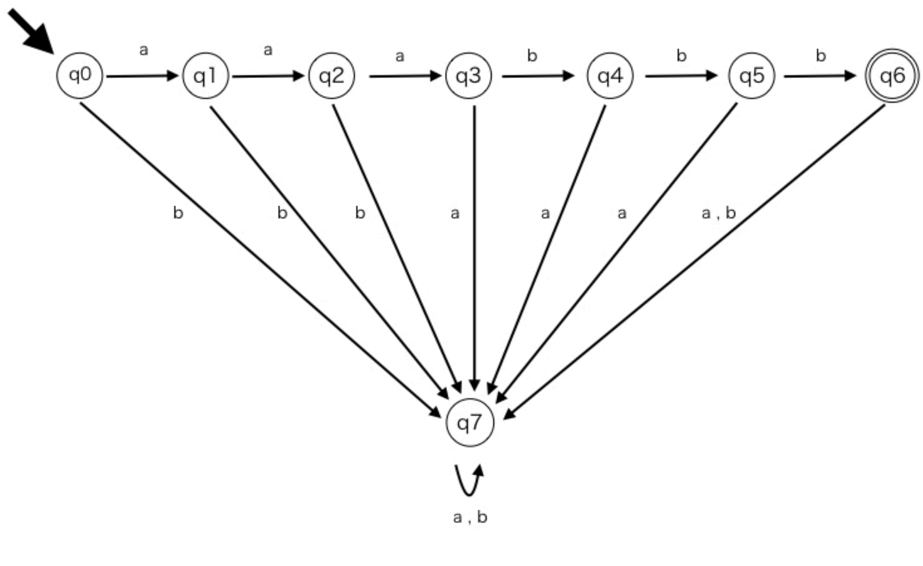
(1-2)

(1-2-1)

L2



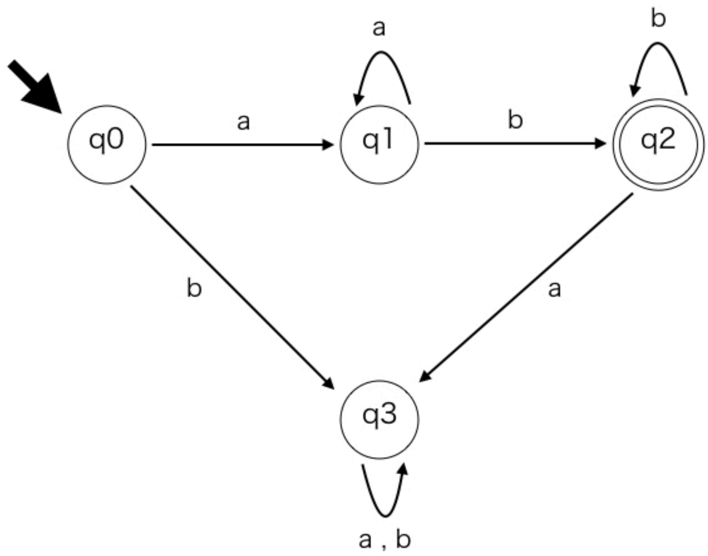
L3



(1-2-2) あまり自信ない・・・

aをi個読み終えた時点での状態数はk-i個である。しかし残っている文字列は2k-i個であり、一つの文字数に対して1つの状態が必要である。よって状態数が不足しているため、このオートマトンはL_{ab}を認識できない。

(1-3)



(2)　　いわゆるCYKアルゴリズムについての問題

(2-1)

表を埋める前に、G1から$\omega 1$ = aaab が導出できるか自分で試してみると良いかもしれない

初期条件より、最初の表は下のようになる

M[i , j]	j = 1	j = 2	j = 3	j = 4
i = 1	{A,C}			
i = 2	-	{A,C}		
i = 3	-	-	{A,C}	
i = 4	-	-	-	{B,D}

l = 2の時を例に考える
n-l+1=3 より、６行目のfor文はi = 1からi = 3 までループする

・i = 1 の時 j = i+l-1 = 2 よって８行目のfor文はk = 1の時のみ動作する
この時、 M[1,1] とM[2,2]を参照し、X→AA、X→AC、X→CCを満たす全てのXをM[1,2]に追加する。すなわちM[1,2]にA→AAを満たすAを追加し、M[1,2] = {A}となる
以上の操作を繰り返すと、l = 2に対して以下の表が得られる

M[i , j]	j = 1	j = 2	j = 3	j = 4
i = 1	{A,C}	{A}		
i = 2	-	{A,C}	{A}	
i = 3	-	-	{A,C}	{S}
i = 4	-	-	-	{B,D}

よって最終的な表は下のようになる

M[i , j]	j = 1	j = 2	j = 3	j = 4
i = 1	{A,C}	{A}	{A}	{S}
i = 2	-	{A,C}	{A}	{S}
i = 3	-	-	{A,C}	{S}
i = 4	-	-	-	{B,D}

(2-2)
文字列 aa は以下のようにするとG2から生成される

S→SA→SAA→AA→aA→~~AA~~
aa

しかし、文字列 aa に対して上記アルゴリズムを実行すると、下の表が作成される

M[i , j]	j = 1	j = 2
i = 1	{A}	∅
i = 2	-	{A}

これは間違った判定をしている

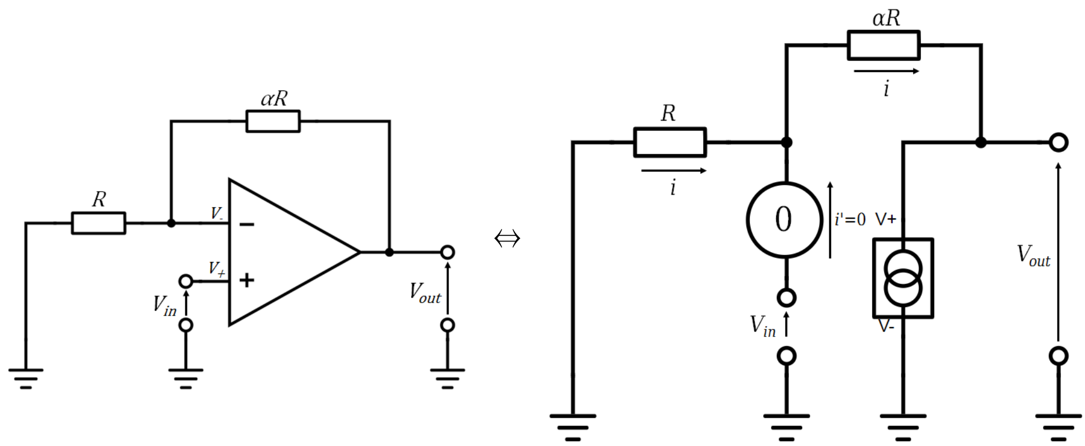
（所感）
難しかった大問１や大問６に比べて簡単だったので得点したいところ。
あまり勉強してなかったCYKアルゴリズムが出て驚いたが、問題文を読んでソースコードをひとつずつ辿れば正解できるので焦らなければ大丈夫。

大問６　電子回路と論理設計

(1)

(1-1)

図１を等価回路に変換して考える.



電流iについて,

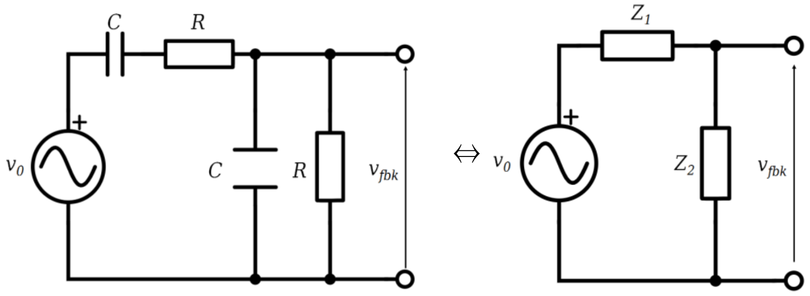
$0 - Ri - \alpha Ri = V_{out}$ かつ $0 + V_{in} - \alpha Ri = V_{out}$ より

$A = V_{out}/V_{in} = \mathbf{1 + \alpha}$　となる.

(1-2)

直列および並列に繋がれたキャパシタと抵抗をそれぞれまとめ,

それぞれのインピーダンスを考えると比較的楽に進む.



$Z_1 = \frac{1}{j\omega c} + R,\ \frac{1}{Z_2} = \frac{1}{1/j\omega C} + \frac{1}{R}$ より, $Z_1 = \frac{1+j\omega CR}{j\omega C},\ Z_2 = \frac{R}{1+j\omega CR}$ となる.

$v_0 - Z_1 i - Z_2 i = 0,\ v_0 - Z_1 i = v_{fbk}$ より,

$$\beta = {v_{fbk}}/{v_0}$$

$$= 1 - {Z_1 i}/{v_0}$$

$$= \frac{Z_2}{Z_1 + Z_2}$$

$$\beta = \frac{Z_2}{Z_1 + Z_2} = \frac{\dfrac{R}{1 + j\omega CR}}{\dfrac{1 + j\omega CR}{j\omega C} + \dfrac{R}{1 + j\omega CR}} = \frac{j\omega CR}{(1 + j\omega CR)^2 + j\omega CR}$$

$$= \frac{j\omega CR}{\{1 - (\omega CR)^2\} + 3j\omega CR}$$

ここで，$|1/\beta|$ を考える．$\omega CR = x$とおくと，

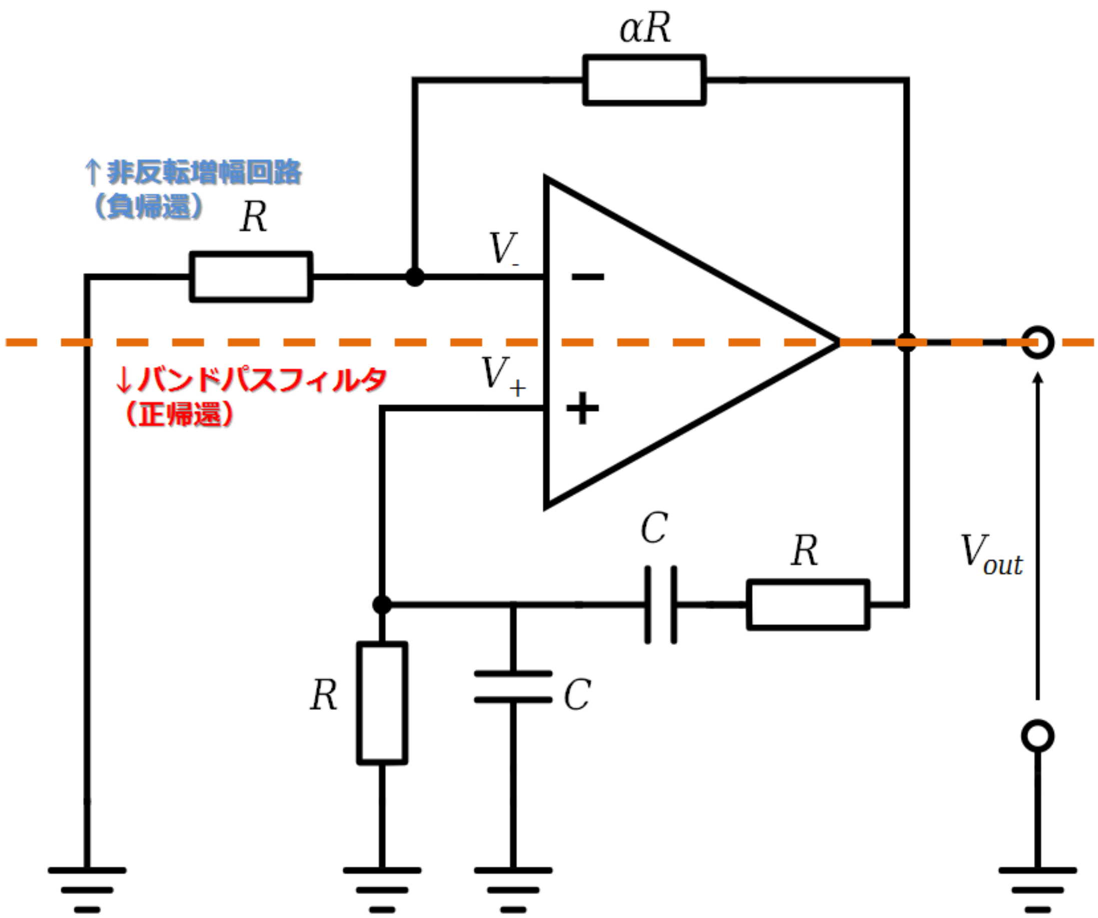
$$|1/\beta|^2 = \left|\frac{(1-x^2)+3xj}{xj}\right|^2 = 9 + \frac{(1-x^2)^2}{x^2} = \frac{1+7x^2+x^4}{x^2}$$

$(|1/\beta|^2)' = 2x - \frac{2}{x^3}$ より，$|1/\beta|$ は$x = 1$ で最小値を取る．

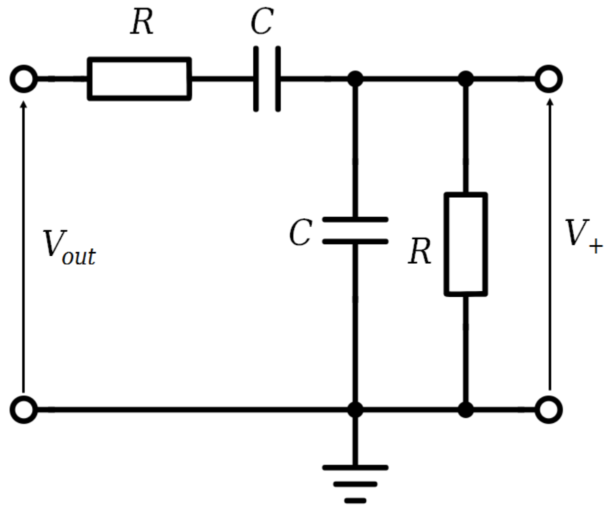
すなわち，$|\beta|^2$ は $x = 1$ で最大化され，その値は1/9となる．このとき，$x = \omega_c CR$より $\boldsymbol{\omega_c = \frac{1}{CR}}$ ．これをβに代入すると$\boldsymbol{\beta = {}^{1}/_{3}}$となり，位相差は**0** となる．

(1-3)

図 3 の回路を分割して考える(図 3 の回路は図 1 の回路に図 2 の回路を印加したものであることから).



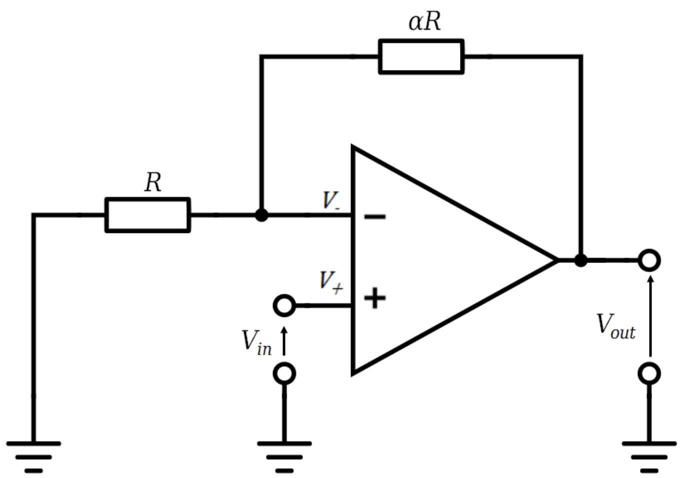
正帰還(バンドパスフィルタ)側



この回路は, (1-2)より，角周波数ω_cのもとで(かつ位相差 0 となるため), $\dfrac{V_+}{V_{out}} = \dfrac{1}{3}$ が成り立つ.

すなわち，オペアンプの入力V_+には最大で$\frac{1}{3}V_{out}$が入力される． $V_+ = \frac{1}{3}V_{out}$

負帰還(非反転増幅回路)側



この回路では, (1-1)より, $\dfrac{V_{out}}{V_+} = 1 + \alpha$ が成り立つ．負帰還側の回路における出力

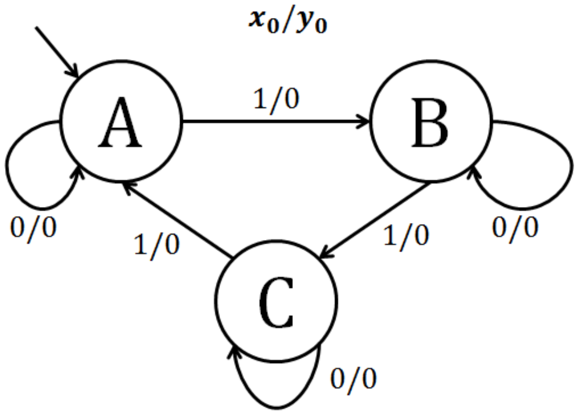
$V_+ = \frac{1}{3}V_{out}$と併せて，$1 + \alpha = 3$ すなわち $\boldsymbol{\alpha = 2}$ が導かれる.

$\omega_c = \frac{1}{CR} = 2\pi f$ より，この時の発振周波数fは $\boldsymbol{f = \dfrac{1}{2\pi CR}}$ となる.

(2)

(2-1)

初期状態の明示を忘れないように.



(2-2)

状態遷移表からカルノー図を作成する.

k_0	k_1	x_0	k_0^+	k_1^+	y_0
0	0	0	0	0	0
0	1	0	0	1	0
1	0	0	1	0	0
1	1	0	d	d	d
0	0	1	0	1	0
0	1	1	1	0	0
1	0	1	0	0	0
1	1	1	d	d	1

x_0 \ k_0k_1	00	01	11	10
0	0	0	d	0
1	0	0	d	1

$y_0 = x_0k_0$

x_0 \ k_0k_1	00	01	11	10
0	0	0	d	1
1	0	1	d	0

$k_0^+ = x_0k_1 \wedge \overline{x_0}k_0$

x_0 \ k_0k_1	00	01	11	10
0	0	1	d	0
1	1	0	d	0

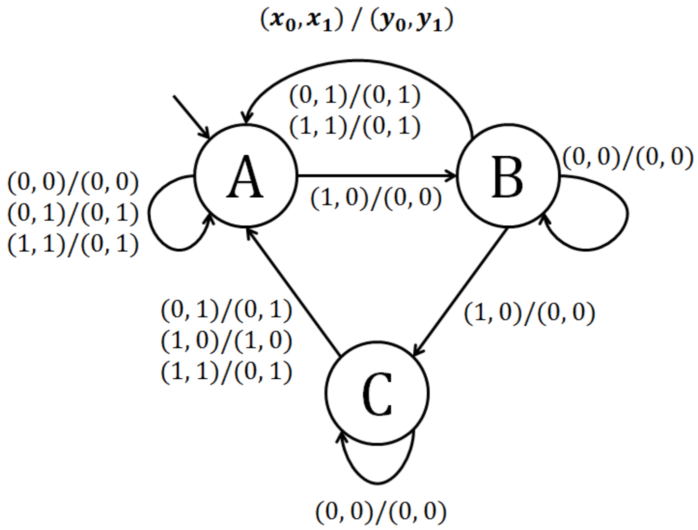
$k_1^+ = x_0\overline{k_0}\,\overline{k_1} \wedge \overline{x_0}k_1$

(2-3)

図 4 の回路図より，入力x_0，x_1, k_0, k_1をリテラルとする出力y_0，y_1, k_0^+, k_1^+ の最簡積和形がわかる．

最簡積和形→カルノー図→状態遷移表の順で変換することにより，以下の状態遷移図が得られる．

なお，問題文より，未定義の状態$D = (k_0, k_1) = (1, 1)$から，あるいはD への遷移を考える必要はない．



(2-4)

状態遷移表を参照すると，入力信号$x_1 = 0$のときは(2-1)の制御回路と同じ挙動を示すこと，また，$x_1 = 1$のとき，出力y_0は常に 0, y_1は常に 1となり，状態は常に A となることがわかる．

したがって, (2-3)の制御回路は(2-1)の回路の持つ機能に加え，入力信号$x_1 = 1$のときにチケットを発券せず($y_0 = 0$であるため)，硬貨の投入を受け付けない/(受け付けた硬貨を返却するしないにかかわらず)投入金額を 0 円とする(状態が常に A であるため)機能をもつ制御回路であると考えられる．

(参考) : (2-3)で求めた最簡積和形，カルノー図，状態遷移表は以下の通り．

最簡積和形:

$k_0^+ = x_0\overline{x_1}k_1 \wedge \overline{x_0}\overline{x_1}k_0$,　$k_1^+ = x_0\overline{x_1}\overline{k_0}\,\overline{k_1} \wedge \overline{x_0}\overline{x_1}k_1$,　$y_0 = x_0\overline{x_1}k_0$,　$y_1 = x_1$

カルノー図:

x_0x_1 \ k_0k_1	00	01	11	10
00	0	0	d	1
01	0	0	d	0
11	0	0	d	0
10	0	1	d	0

$k_0^+ = x_0\overline{x_1}k_1 \wedge \overline{x_0}\overline{x_1}k_0$

x_0x_1 \ k_0k_1	00	01	11	10
00	0	1	d	0
01	0	0	d	0
11	0	0	d	0
10	1	0	d	0

$k_1^+ = x_0\overline{x_1}\overline{k_0}\,\overline{k_1} \wedge \overline{x_0}\overline{x_1}k_1$

x_0x_1 \ k_0k_1	00	01	11	10
00	0	0	d	0
01	0	0	d	0
11	0	0	d	0
10	0	0	d	1

$y_0 = x_0\overline{x_1}k_0$

x_0x_1 \ k_0k_1	00	01	11	10
00	0	0	d	0
01	1	1	d	1
11	1	1	d	1
10	0	0	d	0

$y_1 = x_1$

状態遷移表:

k_0k_1		x_0	x_1	$k_0^+k_1^+$		y_0	y_1
0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0
1	0	0	0	1	0	0	0
0	0	1	0	0	1	0	0
0	1	1	0	1	0	0	0
1	0	1	0	0	0	1	0

k_0k_1		x_0	x_1	$k_0^+k_1^+$		y_0	y_1
0	0	0	1	0	0	0	1
0	1	0	1	0	0	0	1
1	0	0	1	0	0	0	1
0	0	1	1	0	0	0	1
0	1	1	1	0	0	0	1
1	0	1	1	0	0	0	1

所感:

MOSFET や NAND だいすき！から脱却したと思ったらなんと(形式の変わった H24 以降今まで出題されていなかった)交流回路の解析が出題. (僕もそうでしたが)全くのノーマークだった人が多かったそうで, 試験前に「オペアンプも試験範囲入ってる」って言ってた弊研の H 君には頭が上がりません. H 君は「来年はマルチバイブレータが出るとみた」と言っていたのでヤマを張ってもいいかもしれません(責任は取りません).

論理設計は, いつもどおりの流れの後にその逆手順の作業をさせるという変わり種な感じの問題でしたが, 愚直にやっても解けるめんどくさいだけの問題だと思います. うまくやればもっと楽に解けるかもしれませんね.

(文責:塚越)