i	t=120 min		t=960 min		t=1440 min	
	前 進 型 k=30 min	C-N-T k=60 min	前 進 型	C-N-T	前 進 型	C-N-T
0	8.0000003	7.9999998	7.8683289	7.8630776	7.3596088	7.3791880
1	8.0000004	7.9999995	7.8374430	7.8347786	7.2963721	7.3192677
2	8.0000002	7.9999975	7.7347437	7.7406038	7.1014241	7.1338230
3	8.0000001	7.9999839	7.5266624	7.5489123	6.7585955	6.8050599
4	8.0000001	7.9998916	7.1610557	7.2082782	6.2473454	6.3094709
5	8.0000002	7.9992421	6.5762624	6.6537069	5.5510775	5.6262304
6	8.0000003	7.9945767	5.7188374	5.8227827	4.6667704	4.7476092
7	7.9554955	7.9605929	4.5678132	4.6819807	3.6133376	3.6888168
8	7.4184675	7.7104602	3.1572561	3.2554683	2.4358401	2.4938571
9	4.9895764	5.8547073	1.5851834	1.6412312	1.2033445	1.2343221
10	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000

```
   J:=J+1;
  if J=5 then OUTPUT;
  TRID;
  if TT < TLthen go to COEFD;
  for J:=J+1 step 1 until 4 do
  for I:=0 step 1 until M do
    T[I, J]:=0.0;
  OUTPUT
end
```

(注1) identifier の意味: M::=m(≦20), To::=to, T|TL::=tend, K::=K, A|AA::=a, KK::=k, Co::=Co, CA::=Ca, A[I]::=Ai, B0|B1|B2::=Bi, C[I]::=Ci, D[I]::=Di,C[I, J]|T[I, J]::=Ci,j である.

(注2) output の形: 5ステップづつまとめて i=0~m について縦にプリントする. ADVANCEPRINTER(0) でページを改める. △は space をコンマとコンマの間は5個分の space を示す.

(注3) input data: それぞれの identifier の type にしたがって card に穿孔する.

(注4) I/0 の statement は standard procedure として扱う. format declaration はない.

(注5) テストは USSC のために作った TORAY ALGOL で行なった.

<計算例> $K=5.56\times10^{-7}\,cm^2/min$, a=0.1 cm, Co=8%, Ca=0%, m=10 の場合である (上表).

6310. 路上駐車のモンテ・カルロ実験

森口 繁一 (東大計数工学科), 戸田 英雄 (電試応用数学研)

1. 問題 ある道路の片側に駐車可能の区間 AB がある (**第1図**). 初めは全体が空いていて, そこへ1台の自動車が来て駐車する場合, A から B までの間に納る限りどこへでも**勝手次第**に駐車する. その位置を PQ とする. 2台目は A から P まで, または Q から B までの間に勝手に駐車する. 以下同様にして次々と駐車していくと, やがて何台か駐車したとき, もうあとはどこにも駐車する余地がない状態が現われる. このときこの駐車場はつまったと呼ぶことにする. 問題は与えられた長さの駐車場は平均何台でつまるか? ということである.

この問題は, 森口[1] がすでに「応用数学夜話②」において考察し, JIS 乱数を用いて駐車の過程のモンテ・カルロ実験の結果 (**第2図**) を与えている. (Dvoretzky の講演と A. Rényi の論文の紹介) その後戸田が JUSE ALGOL でプログラムを試みたものである. 人間は駐車できるかどうかは**一目でわかる!** が機械は計算しないとわからない.

2. 計算機によるモンテ・カルロ実験 駐車の過程がどのように進行するかを見るため次のようにプログラムした.

START: たとえば X=10.0, PQ=1.0 とする.

INIT: X/PQ の整数部を L とし, P[1], P[2], …, P[L], までの L(=10) カ所を考えて, ここが初めは全部空いていることを示すため, 負の数たとえば −1.0 をいれておく. X/PQ の小数部は P[L+1] にいれる.

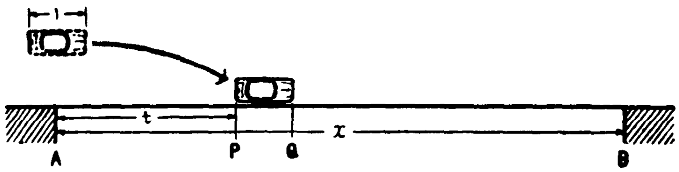
RNDM: 0 から 1.0 までの一様に分布する乱数を発生させて (JUSE ALGOL には RANDOM という real procedure があって, プログラム中には parameter なしの real 型の function designator として一様乱数を発生する.), たとえば 0.62 が得られたとする. R=0.62×L=6.2 として 6.2 の位置から 7.2 の位置までの間に1台駐車せんとするわけだが, その前にできるかどうかを機械は計算しなければならない. (前を見, 後を見て隣りの車や壁とぶつからないかを確める.)

PARK: 駐車可能ならば S=0.2 (R の小数部分) を P[7] に入れて, 6.2 の位置から 7.2 の位置までの区間がつまったことを示す. 同時に駐車数 N を1だけ加える. 駐車不可能のときは **RNDM** にもどって次の乱数を発生させて繰返す. 以下同様にして第2図

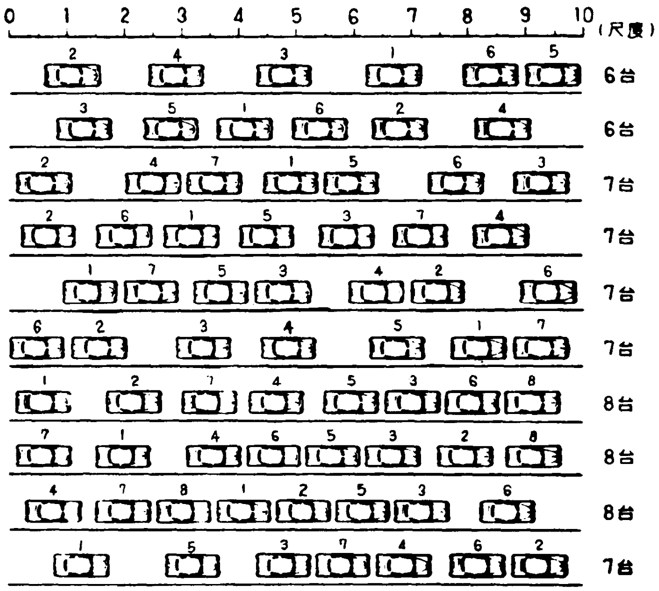
の一番上のような結果が得られる．この試みでは6台で駐車区間が'つまった'ことになる．これを確認するために，**FIND**：でまだ駐車する所があるかを端から探していく．そのため P[1], P[2], …, P[10] まで 10 カ所の位置が空いているか（負であるか）を調べて，最後にもう1台しか入る余地しかないときは，ちょうどそこに入る乱数の発生を待たずに1台駐車させてしまい（機械時間の節約のため） P[10] に至って余地がないとき，'つまったな'と見なす．このようなことを k 回試みて，駐車台数の度数分布を求めるプログラムを次に示す．

3. プログラム

```
    begin integr I, J, K, L, N, JJ;
              real X, PQ, R, S, FL, SUM;
              array FRQ [1 : 20], P [0 : 20];
START:   READINTEGER(K);
              READREAL(X); PQ:=1.0;
              S:=X/PQ;
              L:=ENTIER(S);
              FL:=FLOAT(L);
              for J:=1 step 1 unitl L do
              FRQ(J):=0.0;
              SUM:=0.0;
              P [0]:=0.0;
              P [L+1]:=S−FL;
              for I:=1 step 1 until K do
         begin
INIT:         for J:=1 step 1 until L do
              P (J):=−1.0;
              N:=0;
              JJ:=1;
RNDM:         R:=RANDOM*FL;
              J:=ENTIER(R);
              S:=R−FLOAT(J);
              if P[J+1]>=0.0 then go to RNDM;
              if P[J+2]<0.0 then go to BACK;
```



第1図　自動車のある区間への駐車



第2図　10 台分の区間を満たすいろいろの駐車の仕方

```
        if S-P[J+2]>=0.0 then go to RNDM;
BACK:   if P[J]-S>=0.0 then go to RNDM;
PARK:   P[J+1]:=S;
            N:=N+1;
        CRLF;
        PRINTINTEGER(J+1);
        PRINTREAL(S);
FIND:   if P[JJ]>=0.0 then go to L1;
        if P[JJ+1]>=0.0 then go to L2;
        if P[JJ+2]<0.0 then go to RNDM;
        if P[JJ+2]-P[JJ-1]>=0.0 then go
          to RNDM;
           S:=0.0;  J:=JJ;
           go to PARK;
L2:     if P[JJ+1]-P[JJ-1]>=0.0 then
        being
           S:=P[JJ+1];  J:=JJ-1;
           go to PARK
        end;
L1:     JJ:=JJ+1;
        if JJ<=L then go to FIND;
           FRQ[N]:=FRQ[N]+1.0;
           SUM:=SUM+FLOAT(N);
              CRLF; PRINTINTEGER(I);
              PRINTREAL (SUM/FLOAT(I))
     end;
        for J:=1 step 1 until L do
        begin CRLF;
              PRINTSTRING("FRQ(');
              PRINTINTEGER(J);
              PRINTSTRING(")=');
              PRINTREAL (FRQ[J])
        end
     end
   10 s
   +.1+02.
```

4. 数値例 X=10.0, PQ=1.0, K=10 で試みたが, FRQ (6)=0, FRQ (7)=8, FRQ (8)=2, FRQ (9)=0 となった. 平均台数は (7×8+8×2)/10=7.2(台)で, 駐車効率は 7.2/10=72%である. なお A. Rényi によると駐車台数の期待値は 0.748X−0.252 で X=10 のときは 7.23 となる.

参考文献

1) 森口繁一: 応用数学夜話②, 路上駐車のスペース, 科学朝日, 1962, 2月, pp. 109〜112.