A Multi-stage Allocation Process in Dynamic Programming

Shigeru Kushimoto
Takashi Yoshimura

【1】 A multi-stage allocation process was first descussed in R. Bellman "Dynamic Programming" (Princeton 1957) as follows.

Assume that we have a quantity x which we devide into two parts, y and $x-y$, obtaining from the first quantity y a return of g (y) and the second a return of h $(x-y)$. And suppose that as a price for obtaining the return g (y), the original quantity y is reduced to ay, where a is constant, $o \leqq a < 1$, and similary $x-y$ is reduced to $b(x-y)$, $o \leqq b < 1$, as the cost of obtaining $h(x-y)$. If we repeat this operation of allocation, then the total return is

$$R_1(x, y)=g(y)+h(x-y)$$
$$R_2(x, y, y_1)=g(y)+h(x-y)+g(y_1)+h(x_1-y_1)$$
$$\vdots$$
$$R_N(x, y, y_1, \cdots, y_{N-1})=g(y)+h(x-y)+g(y_1)+h(x_1-y_1)+\cdots\cdots$$
$$\cdots\cdots+g(y_{N-1})+h(x_{N-1}-y_{N-1})$$

where

$$x_1=a\,y+b(x-y), \quad o \leqq y \leqq x$$
$$x_2=a\,y_1+b(x_1-y_1), \quad o \leqq y_1 \leqq x_1$$
$$\vdots$$
$$x_{N-1}=a\,y_{N-2}+b(x_{N-2}-y_{N-2}), \quad o \leqq y_{N-2} \leqq x_{N-2}$$
$$o \leqq y_{N-1} \leqq x_{N-1}$$

Let us now define the function

$f_N(x)=$ the maximum return obtained from an $N-$stage process starting with an initial guantity x, for $N=1,2,\cdots\cdots$ and $x \geqq o$

Then we have the functional equation

$$f_N(x)=\max_{\{y, yi\}} R_N(x, y, y_1, \cdots\cdots y_{N-1})$$
$$=\max_{o \leqq y \leqq x}[g(y)+h(x-y)+f_{N-1}(ay+b(x-y))] \cdots\cdots (\text{I})$$

with $f_1(x)=\max_{o \leqq y \leqq x}[g(y)+h(x-y)]$

【2】 We consider the process under the assumption that additional resources are added at each stage, from the conversion of all return $g(y)+h(x-y)$ into resources, as an extension of 【1】.

Similarly, the total return for the first stage is

$$R_1(x, y)=g(y)+h(x-y), \quad o \leqq x \leqq y$$

Let us assume that g(x) and h(x) are continuous functions of x for all $x \geqq o$, so that this maximum will always exist.

Consider now a two-stage process. If the remaining resource from the first stage is

$$ay+b(x-y) \quad , \quad o \leqq a < 1, \quad o \leqq b < 1$$

then the all of resources for the next stage are

$$g(y)+h(x-y)+ay+b(x-y) \quad , \quad o \leqq y \leqq x$$

We set

$$g(y)+h(x-y)+ay+b(x-y)= x_1 = y_1+(x_1-y_1)$$

for $o \leqq y_1 \leqq x_1$, and obtain as a result of this allocation the return $g(y_1)+h(x_1-y_1)$ at the second stage. Then the all of resources are

$$R_2(x,y,y_1)= g(y_1)+h(x_1-y_1)+ay_1+b(x_1-y_1) \quad , \quad o \leqq y_1 \leqq x_1$$

and the maximum is obtained by maximizing this function of y_1.

Let us proceed to the N-stage process where we repeat the above operation of allocation N time in succession. The return from the N-stage process will then be

$$R_N(x,y,y_1 \cdots y_{N-1})= g(y_{N-1})+h(x_{N-1}-y_{N-1}) \quad , \quad o \leqq y_{N-1} \leqq x_{N-1}$$

where the quantities available for subsequent allocation at the end of the first, second, $\cdots$, $(N-1)$ st stage are given by

$$x_1= g(y)+h(x-y)+a y+b(x-y) \quad , \quad o \leqq y \leqq x$$
$$x_2= g(y_1)+h(x_1-y_1)+ay_1+b(x_1-y_1) \quad , \quad o \leqq y_1 \leqq x_1$$
$$\vdots$$
$$x_{N-1}= g(y_{N-2})+h(x_{N-2}-y_{N-2})+a y_{N-2}+b(x_{N-2}-y_{N-2}) \quad , \quad o \leqq y_{N-2} \leqq x_{N-2}$$

The maximum return for the N-stage process will be obtained by maximizing the function $R_N(x,y,y_1,\cdots y_{N-1})$.

In this case,

$$f_N(x)= \max_{\{y,yi\}} R_N(x,y,y_1,\cdots, y_{N-1}) \qquad (N=2,3,\cdots)$$

$$f_1(x)= \max_{o \leqq y \leqq x} [g(y)+h(x-y)]$$

Considering the two-stage process, we see that the return will be the return from the only second stage, at which stage we have an amount $ay+b(x-y)$ and first stage return $g(y)+h(x-y)$ left to allocate. It is clear that whatever the value y chosen initially, this remaining amount

$$g(y)+h(x-y)+a y+b(x-y)$$

must be used in the best possible manner for the remaining stage. It follows that as a result of an initial allocation of y we will obtain a total return of $f_1(g(y)+h(x-y)+ay+b(x-y))$ from the second stage if y_1 is chosen optimally. Then we have the expression

$$R_2(x,y,y_1) = f_1\{g(y)+h(x-y)+ay+b(x-y)\}$$

Since y is to be chosen to yield the maximum of this expression, we derive the recurrence relation

$$f_2(x)= \max_{o \leqq y \leqq x} [f_1\{g(y)+h(x-y)+ay+b(x-y)\}]$$

Generally, using the same operation for N-stage process, we obtain the functional equation

$$f_N(x) = \max_{0 \leqq y \leqq x} [f_{N-1}\{g(y)+h(x-y)+ay+b(x-y)\}] \cdots (\mathrm{II})$$

for $N \geqq 2$, with

$$f_1(x) = \max_{0 \leqq y \leqq x} [g(y)+h(x-y)]$$

Using this equation, at each step of the computation, we obtain, not only $f_k(x)$, but also $y_k(x)$. Then the solution consists of a tabulation of the sequence of functions $\{y_k(x)\}$ and $\{f_k(x)\}$ for $x \geqq 0$, $k=1,2,\cdots$.

【3】 We consider the same problem under the assumption that a part of the return $c\,\{g(y)+h(x-y)\}$, $0 < c < 1$, is added at each stage into resources.

In this case, we have

$$R_1(x, y) = g(y)+h(x-y)$$

$$R_2(x, y, y_1) = (1-c)\{g(y)+h(x-y)\}+g(y_1)+h(x_1-y_1)$$

$$\vdots$$

$$R_N(x, y, y_1, \cdots, y_{N-1}) = (1-c)\{g(y)+h(x-y)+g(y_1)+h(x_1-y_1)+\cdots$$
$$\cdots+g(y_{N-2})+h(x_{N-2}-y_{N-2})\}+g(y_{N-1})+h(x_{N-1}-y_{N-1})$$

where

$$x_1 = ay+b(x-y)+c\{g(y)+h(x-y) \quad , \quad 0 \leqq y \leqq x$$

$$x_2 = ay_1+b(x_1-y_1)+c\{g(y_1)+h(x_1-y_1)\} \quad , \quad 0 \leqq y_1 \leqq x_1$$

$$\vdots$$

$$x_{N-1} = ay_{N-2}+b(x_{N-2}-y_{N-2})+c\{g(y_{N-2})+h(x_{N-2}-y_{N-2})\} \quad , \quad 0 \leqq y_{N-2} \leqq x_{N-2}$$
$$0 \leqq y_{N-1} \leqq x_{N-1}$$

Using the same argumentation for this process, we obtain the functional equation.

$$f_N(x) = \max_{0 \leqq y \leqq x} [(1-c)\{g(y)+h(x-y)\}+f_{N-1}\{ay+b(x-y)+c(g(y)$$
$$+h(x-y))\}] \cdots (\mathrm{III})$$

for $N \geqq 2$ with

$$f_1(x) = \max_{0 \leqq y \leqq x} [g(y)+h(x-y)]$$

This case is a general form for the multi-stage allocation process. If we set $c=0$, we obtain (I), and if we set $c=1$, we obtain (II) , in this equation.

Example.

Find the solution in the case where

$$g(x) = 1-e^{-x}$$

$$h(x) = 1-e^{-2x}$$

$$a = 0.75 \quad , \quad b = 0.3 \quad , \quad c = 0.7$$

$$x = 2 \quad , \quad N = 5$$

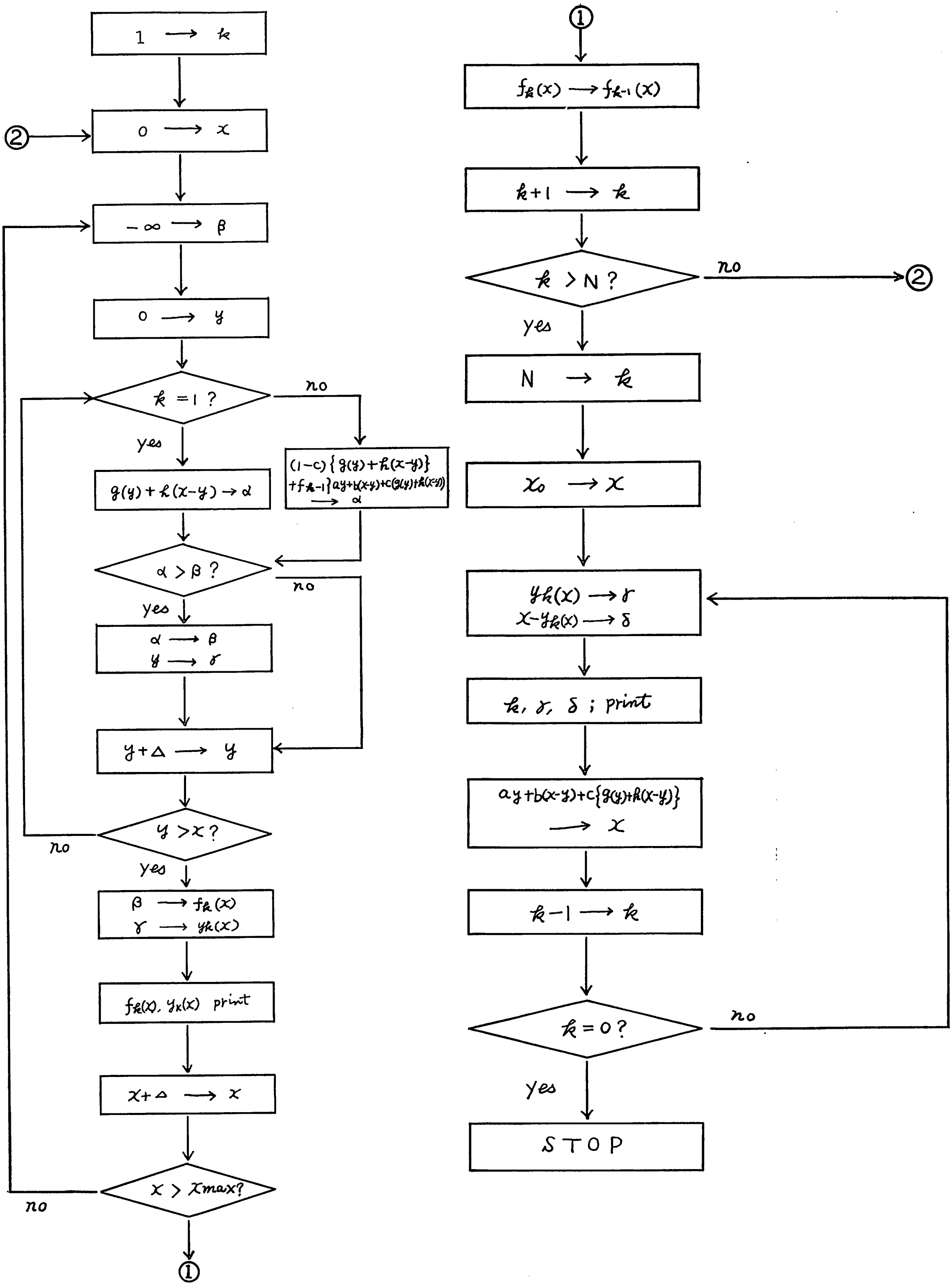
We found the optimal policy for this problem by the following flow chart for digital computer, as follow;

k	y	x−y
1	1.35	0.65
2	1.50	0.74
3	1.60	0.84
4	1.95	0.83
5	1.55	1.16

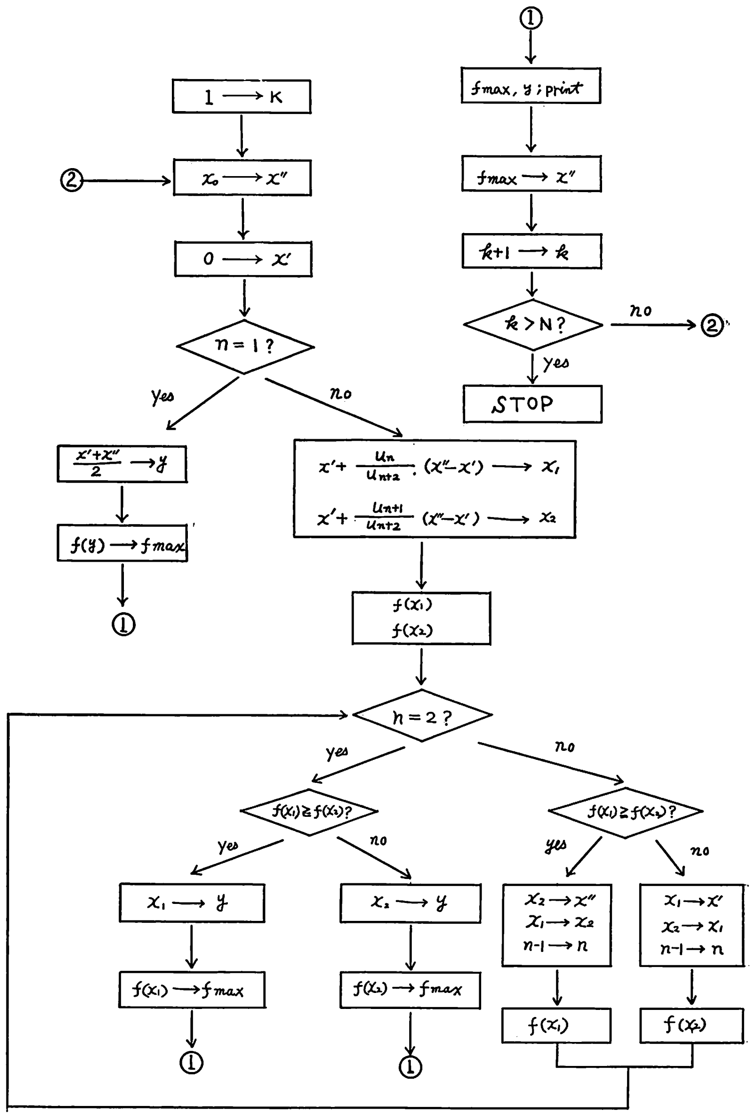
$f_5(2) = 3.57$

In the case 【2】, if g(x) and h(x) are both non decreasing function of x, then $f_N(x)$ is also non decreasing function, and so the optimal policy may be found successively at each stage. Moreover, in the above example, since g(x) and h(x) are concave function, so $f_N(x)$ is a strictly unimodal function. Thus, using the following flow chart 2, we can find the optimal policy more efficiently.

Flow chart for the numerical solution. 1



Flow chart for the numerical solution. 2



{Un} : Fibonacci progression.

$$f(x) = \begin{cases} g(y) + h(x-y) & \text{when } k = N \\ g(y) + h(x-y) + ay + b(x-y) & \text{when } k = 1, 2, \cdots\cdots, N-1 \end{cases}$$