

C.H. Li, A sequential method for screening experimental variables,
J. Amer. Statist. Assoc. 57 (1962), 455-477.

Li can be considered as a pioneer of "Combinatorial G.T."

Li's s-stage Algorithm

- 2-stage algorithm by R. Dorfman (1943)

Partition N into k subsets and test them simultaneously. Then, for the positive groups, test each item individually.

- s -stage algorithm ($s \geq 2$)

Stage 1 : Divide the set N of n items equitably into g_1 groups with each group of size $\lfloor \frac{n}{g_1} \rfloor$ or $\lceil \frac{n}{g_1} \rceil$. Drop the negative groups after tests (simultaneously) and pool the contaminated groups together (at most dk_1 items left).

Stage 2-s : Divide the set of items left from Stage 1 into g_2 groups equitably with size k_2 or k_2-1 . Again, keep the positive groups and pool them together to run the next stage. In the s -stage, each group is of size $k_s=1$. The individual test of each item ends the testing.

e.g. $n=15$, $g_1=4$, $N=\{1, 2, \dots, 15\}$, $D=\{7, 11\}$

Stage 1

$(1, 2, 3, 4)$	-
$(5, 6, 7, 8)$	+
$(9, 10, 11, 12)$	+
$(13, 14, 15)$	-

Stage 2

$(5, 6)$	-
$(7, 8)$	+
$(9, 10)$	-
$(11, 12)$	+

Stage 3

(7)	+
(8)	-
(11)	+
(12)	-

3 tests
4 tests (simultaneously)

- It takes 11 tests to find 7 and 11.
(12)

Definition Let t_s denote the number of tests required by Li's s -stage algorithm.

- In order to minimize t_s , we have to determine the group sizes in each stage, i.e. to determine $k_1, k_2, \dots, k_s$. For convenience, assume that $k_i | n$. (Note that in each stage, the tests are carried simultaneously.)

- $k_1 = n.$

- $t_2 = g_1 + g_2 \leq \frac{n}{k_1} + dk_1$ (positives are in different groups.)

In order to find t_2 , we minimize $f(k_1) = \frac{n}{k_1} + dk_1$.

$$f'(k_1) = -\frac{n}{k_1^2} + d = 0 \Rightarrow k_1^2 = \frac{n}{d} \Rightarrow k_1 = \sqrt{\frac{n}{d}}.$$

$$f(k_1) = 2\sqrt{dn} \text{ where } g_1 = \sqrt{dn}.$$

So, k_1 can be chosen as $\lceil \sqrt{\frac{n}{d}} \rceil$. (For the example above, $\lceil \sqrt{15/2} \rceil = 3$.)

We may check ^{it} again here.

$\begin{array}{l} \textcircled{1, 2, 3} - \\ \textcircled{4, 5, 6} - \\ \textcircled{7, 8, 9} + \Rightarrow ? \\ \textcircled{10, 11, 12} + \\ \textcircled{13, 14, 15} + \end{array}$
 how about k_2

- $t_3 = \sum_{i=1}^3 g_i = g_1 + g_2 + g_3 \leq \frac{n}{k_1} + \frac{dk_1}{k_2} + dk_2.$

Now, minimize $f(k_1, k_2)$. 10 min. Later,

$$k_1 = \left(\frac{n}{d}\right)^{\frac{2}{3}}, \quad k_2 = \left(\frac{n}{d}\right)^{\frac{1}{3}}.$$

- $t_n = \sum_{i=1}^n g_i = g_1 + g_2 + \dots + g_n \leq \frac{n}{k_1} + \frac{dk_1}{k_2} + \frac{dk_2}{k_3} + \dots + \frac{dk_{n-2}}{k_{n-1}} + dk_{n-1}$

Minimize $f(k_1, k_2, \dots, k_{n-1})$ we have $k_i = \left(\frac{n}{d}\right)^{\frac{n-i}{n}}, 1 \leq i \leq n-1.$

$$f(k_1, k_2) = \frac{n}{k_1} + \frac{dk_1}{k_2} + dk_2.$$

$$\frac{\partial f}{\partial k_1} = -\frac{n}{k_1^2} + \frac{d}{k_2} = 0 \quad \text{--- (1)}$$

$$\frac{\partial f}{\partial k_2} = -\frac{dk_1}{k_2^2} + d = 0 \quad \text{--- (2)}$$

From (2) $k_1 = k_2^2$. By (1) $\frac{n}{k_2^4} = \frac{d}{k_2} \Rightarrow k_2^3 = \frac{n}{d}$.

$$k_2 = \left(\frac{n}{d}\right)^{\frac{1}{3}} \text{ and } k_1 = \left(\frac{n}{d}\right)^{\frac{2}{3}}.$$

$$f(k_1, k_2, \dots, k_{s-1}) = \frac{n}{k_1} + \frac{dk_1}{k_2} + \dots + \frac{dk_{s-2}}{k_{s-1}} + dk_{s-1}.$$

$$\frac{\partial f}{\partial k_1} = \frac{\partial f}{\partial k_2} = \dots = \frac{\partial f}{\partial k_{s-1}} = 0$$

$$\Rightarrow k_{s-1} = \left(\frac{n}{d}\right)^{\frac{1}{s}}, \dots, k_1 = \left(\frac{n}{d}\right)^{\frac{s-1}{s}}.$$

Fact (Important)

$$g_i \leq d \left(\frac{n}{d}\right)^{\frac{1}{s}}$$

$$\left(g_i \leq \frac{d \cdot k_{i-1}}{k_i} = \frac{d \cdot \left(\frac{n}{d}\right)^{\frac{s-i+1}{s}}}{\left(\frac{n}{d}\right)^{\frac{s-i}{s}}} \right. \\ \left. = d \cdot \left(\frac{n}{d}\right)^{\frac{1}{s}} \right)$$

$$t_s \leq s \cdot d \cdot \left(\frac{n}{d}\right)^{\frac{1}{s}}.$$

How many stages are the best?

$$\text{Let } f(s) = s \cdot d \left(\frac{n}{d} \right)^{\frac{1}{s}}$$

$$f'(s) = d \left(\frac{n}{d} \right)^{\frac{1}{s}} + s \cdot d \cdot \frac{-\left(\frac{n}{d} \right)^{\frac{1}{s}} \cdot \ln \left(\frac{n}{d} \right)}{s^2}$$

$$= d \cdot \left(\frac{n}{d} \right)^{\frac{1}{s}} \left(1 - \frac{s \cdot \ln \left(\frac{n}{d} \right)}{s^2} \right) = 0$$

$$\Rightarrow s = \ln \left(\frac{n}{d} \right)$$

↑
Unique maximum

$$\text{Hence, } t_s \leq s \cdot d \cdot \left(\frac{n}{d} \right)^{\frac{1}{s}} \leq \ln \left(\frac{n}{d} \right) \cdot d \cdot \left(\frac{n}{d} \right)^{\frac{1}{\ln \left(\frac{n}{d} \right)}} \stackrel{e}{=} \frac{n}{d} = e^s$$

$$= e \cdot d \ln \frac{n}{d}.$$

Theorem (Li) It takes at most $\left\lceil e \cdot d \ln \frac{n}{d} \right\rceil$ tests to determine the set of d positives (out of n items) by using Li's s -stage algorithm.

(*) For convenience to check how good the algorithm is, we may use " $\log_2$ " notation to show the above upper bound.

$$e \cdot d \ln \frac{n}{d} = \frac{e}{\log_2 e} \cdot d \cdot \log_2 \left(\frac{n}{d} \right).$$

(*) Roughly, $d \log_2 n$ tests will be about what we can get.
(the best)

Hwang's Generalized Binary Splitting Algorithm (G)

Binary splitting algorithm shows that $M(1, n) = \lceil \log_2 n \rceil$. But, if $d \geq 2$, then we need about $d \lceil \log_2 n \rceil$ to find d positives. For example, $M(2, 33) \leq 12$. Hwang's algorithm^(G) shows that we can do better by using the following idea.

Algorithm G

Step 1. If $n \leq 2^d - 2$, test the n items individually. If $n \geq 2^d - 1$, set $l = n - d + 1$ and $\alpha = \lceil \log_2(\frac{l}{d}) \rceil$.

Step 2. If $n \geq 2^d - 1$, test a group of size 2^α . If the outcome is negative, the 2^α items are good. Set $n \stackrel{\text{def}}{=} n - 2^\alpha$ and go to Step 1. If the outcome is positive, use binary splitting idea to identify "one" defective and "some" good items, say x . Set $n \stackrel{\text{def}}{=} n - 1 - x$ and $d \stackrel{\text{def}}{=} d - 1$. Go to Step 1.

e.g. $n = 33$, $d = 2$. Set $l = 33 - 2 + 1$ and $\alpha = \lceil \log_2 \frac{32}{2} \rceil = 4$.

Test a group of size 16.

Case 1. The group is negative. Then we need $\wedge$ ^{at most} $1 + \underbrace{M(17, 2)}_{\text{Step 1}}$ tests. 11 tests

Case 2 The group is positive. Then, it takes 4 more tests to find one defective item and possibly some good items, x . So, we need at most $5 + M(1, 32-x)$ to identify two defective items which is at most 10 tests. (The answer will be at most 11.)

(*) For $d=1$, this algorithm may not be better than "Binary splitting algorithm".

Theorem (Algorithm G)

$$M_G(d, n) = \begin{cases} n & \text{for } n-1 \leq 2^d-2, \\ (\alpha+2)d+p-1 & \text{for } n \geq 2^d-1, \end{cases}$$

where $p < d$ is a solution of $l = 2^\alpha d + 2^\alpha p + \theta$, $0 \leq \theta < 2^\alpha$.

Proof. Since $M_G(d, n) = \max\{1 + M_G(d, n-2^\alpha), 1 + \alpha + M_G(d-1, n-1)\}$

by algorithm G, it is left to check which one is larger. First,

we count $1 + M_G(d, n-2^\alpha)$. Let $n' = n - 2^\alpha$ and $d' = d$. Then,

$$l' = n' - d' + 1 = l - 2^\alpha = \begin{cases} 2^\alpha d + 2^\alpha (p-1) + \theta & \text{for } p \geq 1, \\ 2^{\alpha-1} d + 2^{\alpha-1} (d-2) + \theta & \text{for } \boxed{p=0}, \theta < 2^{\alpha-1}, \\ 2^{\alpha-1} d + 2^{\alpha-1} (d-1) + (\theta - 2^{\alpha-1}) & \text{for } \boxed{p=0}, \theta \geq 2^{\alpha-1}. \end{cases}$$

$$\Rightarrow M_G(d, n-2^\alpha) = \begin{cases} (\alpha+2)d-2 & \text{for } p=0, \theta < 2^{\alpha-1}, \\ (\alpha+2)d+p-1 & \text{otherwise} \end{cases}$$

(By induction)

On the other hand, (Similar argument)

$$M_G(d-1, n-1) = \begin{cases} (\alpha+2)(d-1) + (p+1) - 1 & \text{for } p \leq d-3, \\ (\alpha+2)(d-1) - 1 & \text{for } d-2 \leq p \leq d-1. \end{cases}$$

We conclude $M_G(d, n) = (\alpha+2)d + p - 1$. (?) ■

(•) If $2d-1 \leq n \leq 3d-2$, then $\alpha = \left\lfloor \log_2 \frac{l}{d} \right\rfloor$

$$= \left\lfloor \log_2 \frac{n-d+1}{d} \right\rfloor \leq \left\lfloor \log_2 \frac{2d-1}{d} \right\rfloor = 0.$$

This implies that $M_G(d, n) = 2d + p - 1$ where $l = d + p + \theta$, $0 \leq \theta < 1$. Hence $p = l - d = n - 2d + 1$ which shows $M_G(d, n) = n$.

(•) This fact can be further verified later. In fact, it is true for any algorithm using group testing.

(•) $M_G(d, n)$ is not too far away from the information lower bound $\lceil \log \binom{n}{d} \rceil$.

Proposition (Hwang's algorithm)

$$M_G(d, n) - \lceil \log \binom{n}{d} \rceil \leq d-1 \text{ for } d \geq 2.$$

Proof.

$$\begin{aligned}
 \binom{n}{d} &= \binom{l+d-1}{d} > \frac{l^d}{d!} = \frac{(2^\alpha d + 2^\alpha p + \theta)^d}{d!} \\
 &\geq \left[2^\alpha d \left(1 + \frac{p}{d}\right) \right]^d / d! \\
 &= \frac{2^{\alpha d} \cdot d^d \cdot \left(1 + \frac{p}{d}\right)^d}{d!} = 2^{\alpha d} \cdot \left(\frac{d}{d!}\right) \cdot \left(1 + \frac{p}{d}\right)^d \geq 2^p \\
 &\geq 2^{\alpha d} \cdot 2^{d-1} \cdot 2^p = 2^{\alpha d + d + p - 1}
 \end{aligned}$$

$$M_G(d, n) - \lceil \log \binom{n}{d} \rceil$$

$$< \lceil (\alpha + 2)d + p - 1 \rceil - (\alpha d + d + p - 1) = d.$$

- If $d = 2$, then Hwang's algorithm almost solve the problem by leaving a gap "1".

e.g. $n = 15, d = 2$

$$l = 15 - 2 + 1 = 14, \alpha = \lfloor \log_2 7 \rfloor = 2.$$

$$14 = 8 + 4p + \theta, 0 \leq \theta < 4, p = 1, \theta = 2$$

Hwang's algorithm shows that

$$M_G(2, 15) = (2+2) \cdot 2 + 1 - 1 = 8$$

$$\lceil \log_2 \binom{15}{2} \rceil = \lceil \log_2 105 \rceil = 7$$

$$\frac{d^d}{d!} \geq 2^{d-1}$$

By induction on d .

$$\frac{k^k}{k!} \geq 2^{k-1}$$

$$\frac{(k+1)^{k+1}}{(k+1)!} = \frac{(k+1)^k}{k!} \geq \frac{(k+1)^k}{k^k} \cdot \frac{k^k}{k!} \geq 2 \cdot 2^{k-1} = 2^k$$

$$\frac{k^k}{k!} \cdot \frac{(k+1)^k}{k^k}$$

$$\left(1 + \frac{p}{d}\right)^d \geq 2^p$$

$$d = p \Rightarrow \left(1 + \frac{p}{d}\right)^d = 2^p$$

$$x \geq p$$

$$f(x) = \left(1 + \frac{p}{x}\right)^x \nearrow$$

$$(f'(x) > 0)$$

||