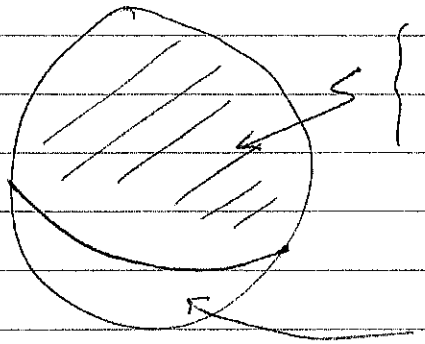
Lecture 2 Oct. 12

NO. 1
DATE

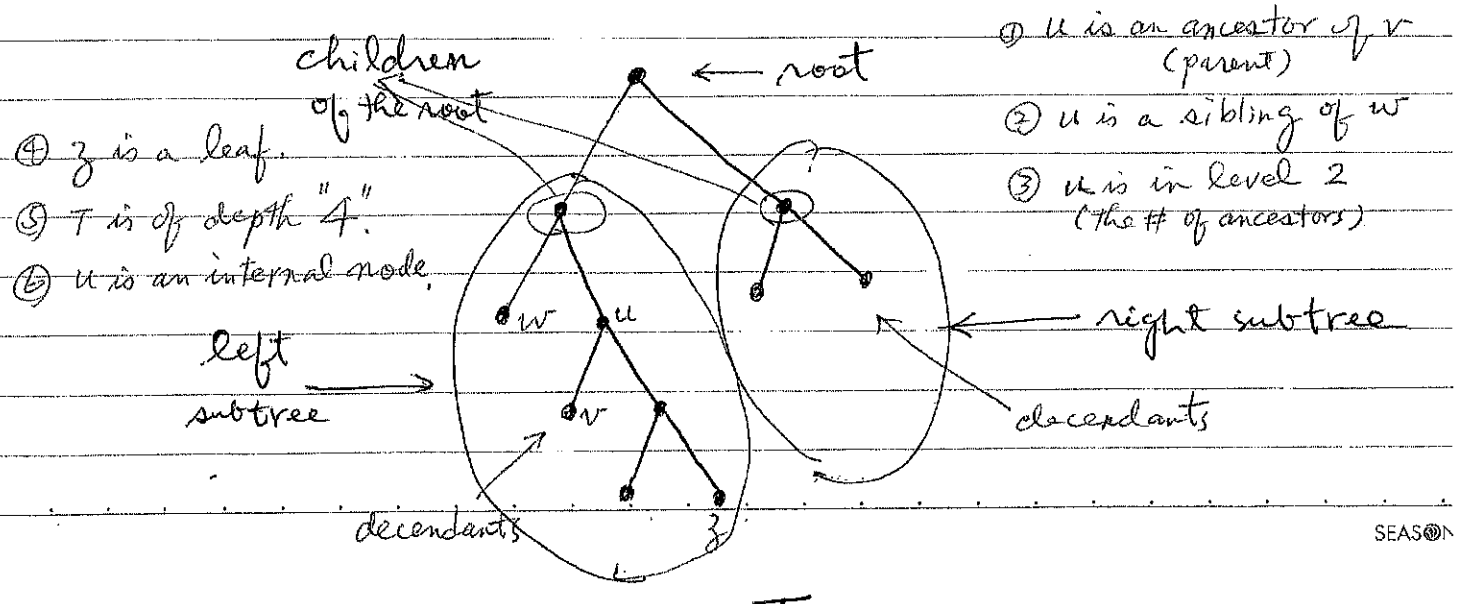
General Sequential Algorithms

- Group testing takes advantage of that by identifying groups containing no positives.
- So, how large a "group" should be?
-



- Keeping a balance between these two conflicting goals is what most algorithms strive for.

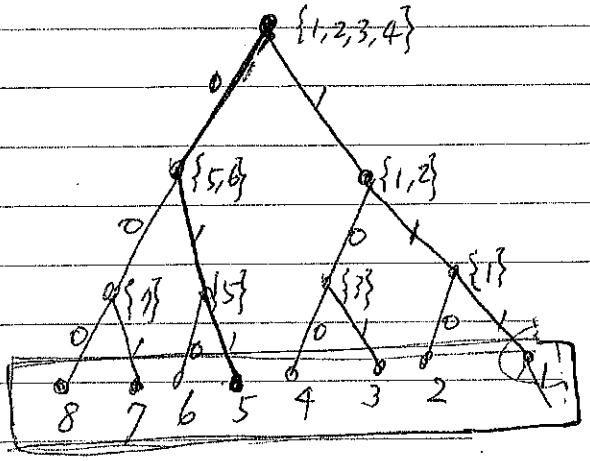
(*) A classical GT can be represented by a binary tree
(Sequencial Algorithm)



SEASON

One positive item in $\{1, 2, \cdots, 8\}$.

Test 1 : $\{1,2,3,4\}$

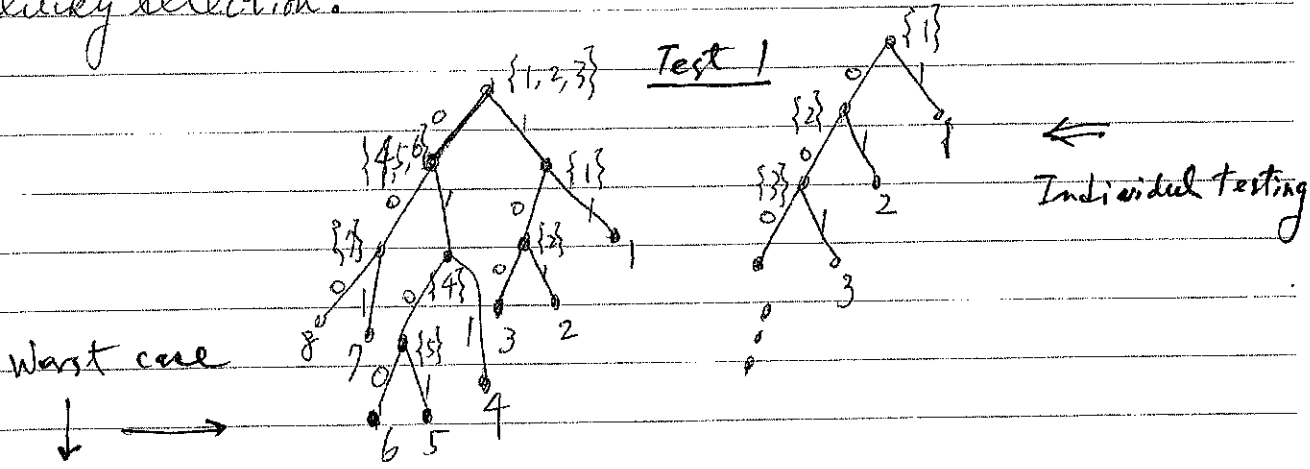


Test 2

Test 3

Two positive items in $\{1, 2, \cdots, 8\}$.

In general, an adaptive algorithm can only be applied to find positives one by one except the test is carried out via a "lucky selection".

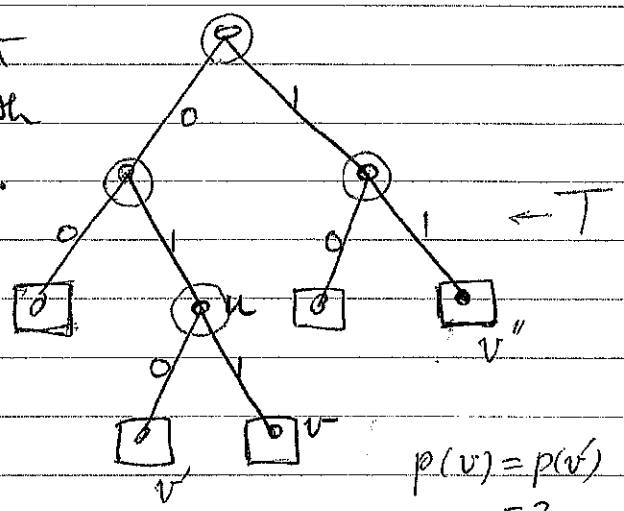


The rules of representation (Sample space S, Algorithm T)

① Each internal node u is associated with a test $t(u)$. Two children are two outcomes of the test and the left one is "negative".

② The test history of u, $H(u)$, is the set of tests and outcomes associated with the nodes and links on the path of u.

③ Each node u is also associated with an event $S(u)$ which consists all the members of S consistent with $H(u)$. $|S(v)| \leq 1$ for each leaf v.



$p(v) = p(v') = 3$,
$p(v'') = 2$

- $t(u)$ splits $S(u)$ into two disjoint subsets.

(*) Let $\{v_1, v_2, \cdots, v_t\}$ be the set of leaves in a binary tree.

Then $\sum_{i=1}^{t} 2^{-p(v_i)} = 1$. ($p(v_i)$ is the length of the path from the root to v_i.)

(•) In the above example, we have $2^{-2} + 2^{-3} + 2^{-3} + 2^{-2} + 2^{-2} = 1$.

Algorithm (Reasonable, admissible)

Fact 7. $M(d,n) \le M(d,n+1)$.

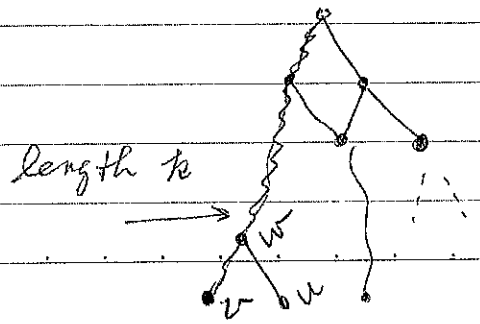
Proof. Adding a good item to any group does not affect the outcome.

(Note) The equality may not hold. $M(1,8) < M(1,9)$.

Fact 8. Let $M(m;d,n)$ denote the min. number of tests to identify d positives among n items when a particular subset of m items is known to be contaminated. Then

$M(m;d,n) \ge 1 + M(d-1,n-1)$ for $m \ge 2$ and $0 < d < n$.

Proof. Since $M(m;d,n) \ge M(2;d,n)$, it suffices to consider $M(2;d,n)$. Let $\{a,b\}$ be a contaminated (group) subset. Let T be an algorithm such that $M_T(2;d,n) = k$, i.e., k is the maximum length of a leaf in T. (Note that T will not test $\{a\}$ or $\{b\}$ as a first test! For otherwise, it takes much more tests than $M(d-1,n-1)+1$.)



Exercise 3 Prove Fact 8.

The details of proving this Fact takes some effort. The main idea is the contaminated set should take part in determining the positives if some of them do need $M_T(z; d, n)$ tests to determine them.

Intutively, even we know a positive subset, it still needs $M(d-1, n-1)+1$ tests, to determine all the d positives.

(We skip the details here.)

Fact 9. $M(d,n) \le n-1$. (By individual test and d is fixed.)

Fact 10. $M(n-1, n) = n-1$. (Determine d, s.t., $M(d,n) \ge n-1$.) problem

Proof. By induction on n. $n=1$ is true. (Using the hypothesis)

$$M(n-1, n) \ge M(n; n-1, n) \ge M(n-2, n-1) + 1 \quad \text{(Fact 8)}$$
$$= n-2+1 = n-1.$$ ∎

Fact 11. $M(d,n) \ge 1 + M(d-1, n-1) \ge M(d-1, n)$ for $0 < d < n$.

Proof. By induction on d, and $d=1$ is true for second inequality. The first one follows from Fact 8 and $M(d,n) = M(n; d, n)$ if $d \ge 1$. We prove Fact 11 in general. (For $d > 1$)

Let, T be an algorithm which tests one item to start. Then (先測一個 Item!) the 2nd inequality of

$M(d-1,n) \le M_T(d-1,n)$

$= 1 + \max\{M(d-1,n-1), M(d-2,n-1)\}$

$= 1 + M(d-1,n-1)$.

Fact 12. Suppose that $n > d+1$. Then $M(d,n) = n-1 \Rightarrow M(d,n-1) = n-2$. (Note that d may be large.) (Individual tests)

Proof. Assume that $M(d,n-1) \neq n-2$, then $M(d,n-1) < n-2$. Let T be the algorithm which tests a single item to start. This implies that $M(d,n) \le M_T(d,n) = 1 + \max\{M(d,n-1), M(d-1,n-1)\}$

By Fact 11, $M(d,n) \le 1 + M(d,n-1) \le 1 + (n-2)$ (assumption)

$= M(d,n)$. $\longrightarrow\!\longleftarrow$ ∎

Fact 13. $M(d,n) = M(d-1,n) \Rightarrow M(d,n) = n-1$.

Proof. By induction on $n-d$ and it is true when $n-d=1$. (Fact 10) without using assumption

By Fact 11, $M(d,n) = M(d-1,n) \Rightarrow M(d,n) = M(d-1,n-1)+1$.

Let T be a minimax algorithm for the (d,n) problem. First, in T, a test of a group of m items is carried out.

Case 1. $m > 1$.

$M_T(d,n) \geq 1 + M(m;d,n) \geq 2 + M(d-1,n-1)$, a contradiction since $M(d,n) = M(d-1,n-1)+1$.

Case 2. $m = 1$

$$M(d,n) \geq M_T(d,n) = 1 + \max\{M(d,n-1), M(d-1,n-1)\}$$

$$= 1 + M(d,n-1).$$

Now, since $d < n-1$ and $M(d,n-1) = M(d-1,n-1)$, we have $M(d-1,n-1) = n-2$ by induction. Therefore,

$$M(d,n) = 1 + M(d-1,n-1) = n-1. \quad \blacksquare$$

Fact 14 If $M(d,n) < n-1$, then $M(d,n) \geq 2l + M(d-l,n-l)$ for $0 < l \leq d < n$.

Proof. By Fact 10, $M(n-1,n) = n-1$, we have $d < n-1$, i.e. $n-d > 1$.

The proof follows by showing $M(d,n) \geq 2 + M(d-1,n-1)$. (?)

($M(d-1,n-1) \geq 2 + M(d-2,n-2)$ inductively)

Let T be a minimax algorithm for the (d,n) problem which starts with a test on m items. First, if $m > 1$, then by

Fact 8, we have $M_T(d,n) \geq 1 + 1 + M(d-1,n-1)$. So, let $m=1$ and suppose that $M_T(d,n) < 2 + M(d-1,n-1)$. This implies that

$$1 + M(d-1,n-1) \geq M_T(d,n) = 1 + \max\{M(d,n-1), M(d-1,n-1)\}$$

$$= 1 + M(d,n-1).$$

By the fact $M(d,n-1) \geq M(d-1,n-1)$, we have $M(d-1,n-1) = M(d,n-1)$. Hence $M(d-1,n-1) = n-2$, (Fact 13) and thus $M(d,n) = n-1$. $\Rightarrow\Leftarrow$

Fact 15 $M(d,n) \geq \min\{n-1, 2\ell + \lceil \log \binom{n-\ell}{d-\ell} \rceil\}$ for $0 < \ell \leq d < n$.

Proof. $M(d-\ell, n-\ell) \geq \lceil \log \binom{n-\ell}{d-\ell} \rceil$.

✓ Fact 16 $M(d,n)+1 \geq M(\bar{d},n) \geq M(d,n+1)$.

Proof.

Claim. For each algorithm T of the (d,n) problem, there exists an algorithm T' for the $(\bar{d},n)$ problem such that $M_T(d,n)+1 \geq M_{T'}(\bar{d},n)$.

If the above claim is true, then $M(d,n)+1 \geq M(\bar{d},n)$. As to the second inequality, it follows by ignoring one item.

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 $\lfloor \frac{n}{g_1} \rfloor$ // k_1 or $\lceil \frac{n}{g_1} \rceil$ // $k_1 - 1$. 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.

SEASON

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

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 (sequential)
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 $k_1, k_2, \cdots, k_s$. For convenience, assume that $k_1 | n$. (Note that in each stage, the tests are carried simultaneously.)

- $t_1 = n$.

- $t_2 = g_1 + g_2 \le \frac{n}{k_1} + dk_1$ (positives are in different groups.) (may be in the same group) ↑ worst case

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

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

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

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

We may check it again here.

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

how about k_2 ($k_2 = 1$ or 2) ⇒ ?

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

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

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

- $t_s = \sum_{i=1}^{s} g_i = g_1 + g_2 + \cdots + g_s \le \frac{n}{k_1} + \frac{dk_1}{k_2} + \frac{dk_2}{k_3} + \cdots + \frac{dk_{s-2}}{k_{s-1}} + dk_{s-1}$

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

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

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

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

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

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

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

$$\frac{\partial f}{\partial k_1} = \frac{\partial f}{\partial k_2} = \cdots = \frac{\partial f}{\partial k_{n-1}} = 0$$

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

=

Fact (Important)

$$q_i \le d\left(\frac{n}{d}\right)^{\frac{1}{n}} \quad \left(q_i \le \frac{d \cdot k_{i-1}}{k_i} = \frac{d \cdot \left(\frac{n}{d}\right)^{\frac{n-i+1}{n}}}{\left(\frac{n}{d}\right)^{\frac{n-i}{n}}} = d \cdot \left(\frac{n}{d}\right)^{\frac{1}{n}}. \right)$$

$$t_n \le n \cdot d \cdot \left(\frac{n}{d}\right)^{\frac{1}{n}}.$$

How many stages are the best ?

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

$$\frac{d(a^{s^{-1}})}{ds} = \frac{d(e^{\ln a \cdot s^{-1}})}{ds}$$

$$= e^{\ln a \cdot s^{-1}} \cdot \ln a \cdot (-s^{-2})$$

$$= -a^{s^{-1}} \cdot \ln a \cdot s^{-2}$$

$$= \frac{-a^{s^{-1}} \cdot \ln a}{s^2}$$

Hence, $t_s \le s \cdot d \cdot \left(\frac{n}{d}\right)^{\frac{1}{s}} \le \ln\left(\frac{n}{d}\right) \cdot d \cdot \boxed{\left(e^{s}\right)^{\frac{1}{s}}}$

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

Theorem (Li) It takes at most $\lceil e \cdot d \ln \frac{n}{d} \rceil$ 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 $n \leq 2d-2$, test the n items individually. If $n \geq 2d-1$, set $l = n-d+1$ and $\alpha = \left\lfloor \log_2 \left(\frac{l}{d}\right) \right\rfloor$.

Step 2. If $n \geq 2d-1$, test a group of size 2^α. If the outcome is negative, the 2^α items are good. Set $n =_{def} n - 2^\alpha$ 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 (In binary splitting history)
$n = 1 - x$ and $d =_{def} d-1$. Go to Step 1.

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

Test a group of size 16.

Case 1. The group is negative. Then we need $\leq 1 + M(2,17)$ tests. (11 tests)
at most
Step 1

SEASON

Case 2 The group is positive. Then, it takes at most 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". ($\alpha = 5$ for the above example.)

Theorem (Algorithm G)

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

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

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,

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

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

(By induction)

On the other hand, (Similar argument)

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

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

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

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

This implies that $M_G(d,n) = 2d+p-1$ where $\ell = d+p+\theta$, $0 \le \theta < 1$. Hence $p = \ell - d = n-2d+1$ 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. (If d is larger, then $M(d,n) = n$.)

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

Proposition (Hwang's algorithm)

$$M_G(d,n) - \lceil \log \tbinom{n}{d} \rceil \le d-1 \text{ for } d \ge 2.$$

Proof.

$$\binom{n}{d} = \binom{l+d-1}{d} > \frac{l^d}{d!} = \frac{(2^\alpha d + 2^\alpha p + \theta)^d}{d!}$$

$$\geq \left[2^\alpha d\left(1+\frac{p}{d}\right)\right]^d / d!$$

$$= \frac{2^{\alpha d} \cdot d^d \cdot \left(1+\frac{p}{d}\right)^d}{d!} = 2^{\alpha d} \cdot \underbrace{\frac{d^d}{d!}}_{\geq 2^{d-1}} \cdot \underbrace{\left(1+\frac{p}{d}\right)^d}_{\geq 2^p}$$

$$\geq 2^{\alpha d} \cdot 2^{d-1} \cdot 2^p = 2^{\alpha d + d + p - 1}.$$

next page

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

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

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

e.g. $n = 15$, $d = 2$ $\quad l = 15 - 2 + 1 = 14$, $\alpha = \lfloor \log_2 7 \rfloor = 2$

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

Hwang's algorithm shows that

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

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

Hence, $7 \leq M(2,15) \leq 8$.

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

By induction on d.

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

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

≥ 2

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

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

$x \geq p$

$f(x) = \left(1+\frac{p}{x}\right)^x \uparrow \qquad (f'(x) > 0)$