



Complex Group Testing and Its Applications

Hung-Lin Fu 傅 恆 霖

Department of Applied Mathematics

Hsin-Chu Chiao Tung University



(4th ITCDM at IIT Madras, Chennai)

- 
- 
- This talk is devoted to my mentor **Frank K. Hwang** for his guidance to enter this wonderful field of search methodology.
 - I also appreciate the joint effort made by Dr. Hong-Bin Chen, Huilan Chang and Chih-Huai Shih.



What is “Group Testing”?



- A problem:

I have a favor number in my mind and the number is in $\{1, 2, 3, \dots, 63\}$. Can you find the number by asking (me) as few *queries* as possible? (A *query* asks if the number is in a certain subset of the given set.)



The answer : **At most 6**

- (1) Is the number larger than 32?
Yes or **Not**?
- (2) If the answer is “Yes”, then ask if the number is larger than **48** or not. On the other hand, if the answer is “**Not**”, then ask if the number is larger than **16** or not.
- Continued ...
- Worst case: **6 (Information bound)**

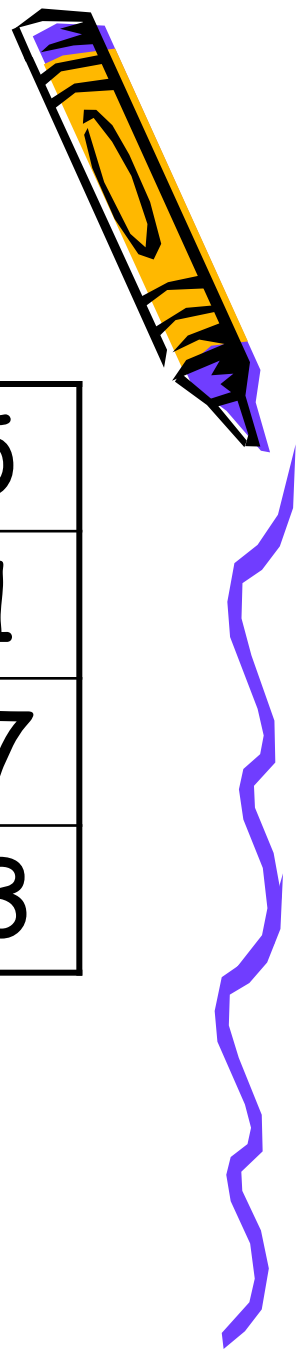


Another Idea



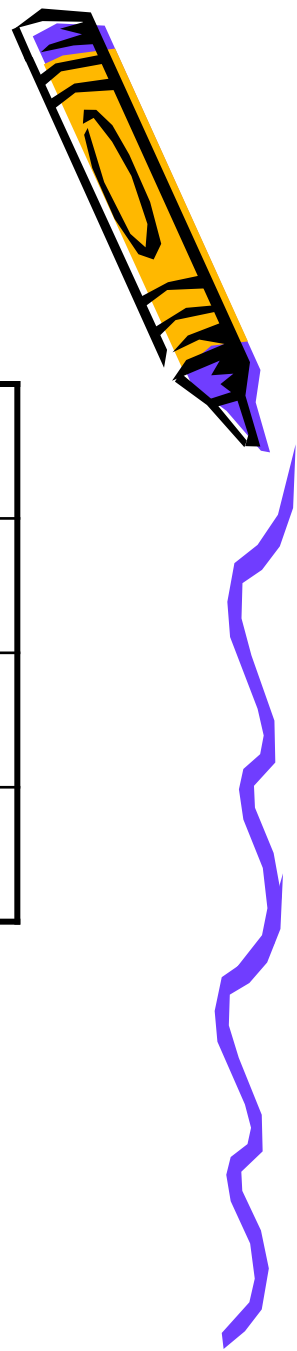
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63





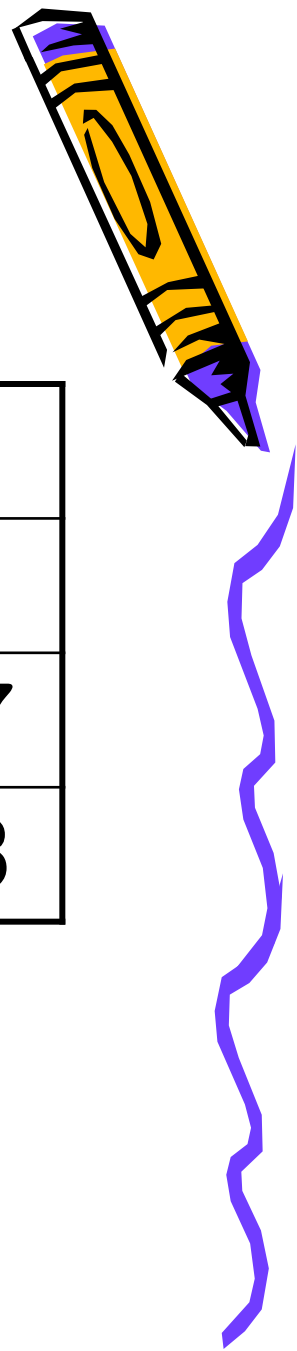
2	3	6	7	10	11	14	15
18	19	22	23	26	27	30	31
34	35	38	39	42	43	46	47
50	51	54	55	58	59	62	63





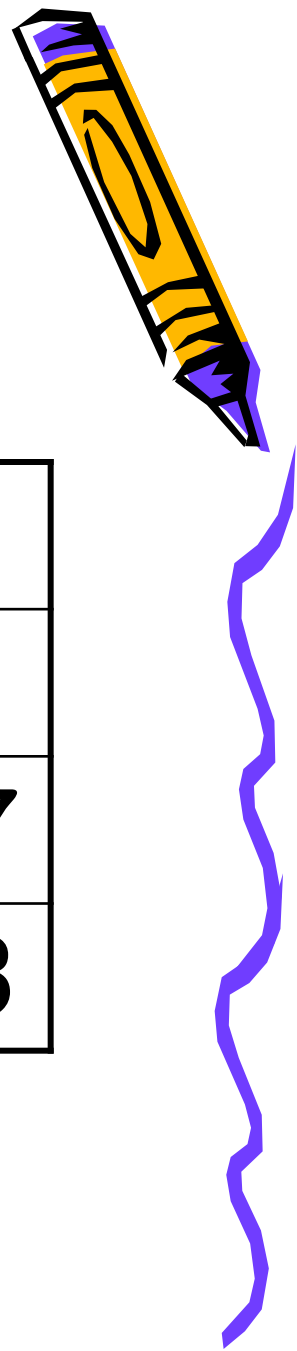
4	5	6	7	12	13	14	15
20	21	22	23	28	29	30	31
36	37	38	39	44	45	46	47
52	53	54	55	60	61	62	63





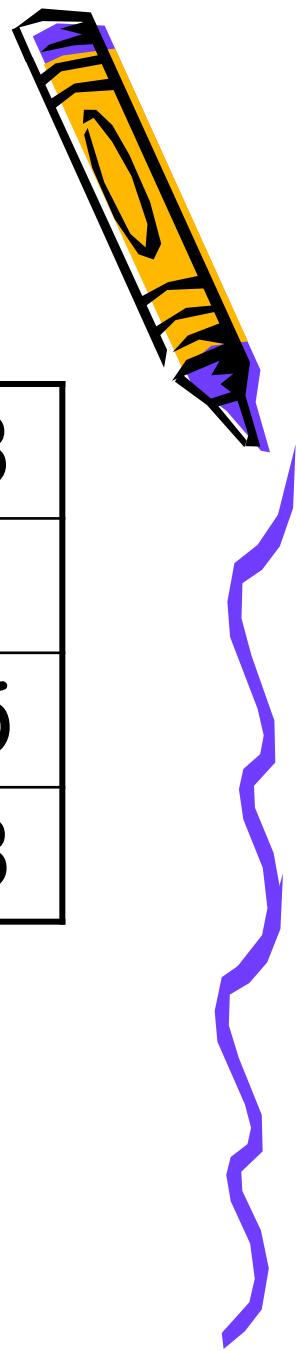
8	9	10	11	12	13	14	15
24	25	26	27	28	29	30	31
40	41	42	43	44	45	46	47
56	57	58	59	60	61	62	63





1	3	5	7	9	11	13	15
17	19	21	23	25	27	29	31
33	35	37	39	41	43	45	47
49	51	53	55	57	59	61	63





16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63



Group Testing

- Consider a set N of n items consisting of at most d *positive* (used to be called defective) items with the others being *negative* (used to be called good) items.
- A group test, sometimes called a *pool*, can be applied to an arbitrary set S of items with two possible outcomes; *negative*: all items in S are negative; *positive*: at least one positive item in S , not knowing which or how many.



Algorithms



- An adaptive (*sequential*) algorithm conducts the tests *one by one* and the outcomes of all previous tests can be used to set up the later test. A *non-adaptive* algorithm specifies a set of tests in advance so that they can be conducted *simultaneously*, thus forbidding using the information of previous tests.



Complex Group Testing



- If the positives are subsets of items instead of just individual items, then we have the complex group testing.
- The subsets are called *complexes*.
- The problems of complex group testing can be formulated as *hidden graphs problems*.



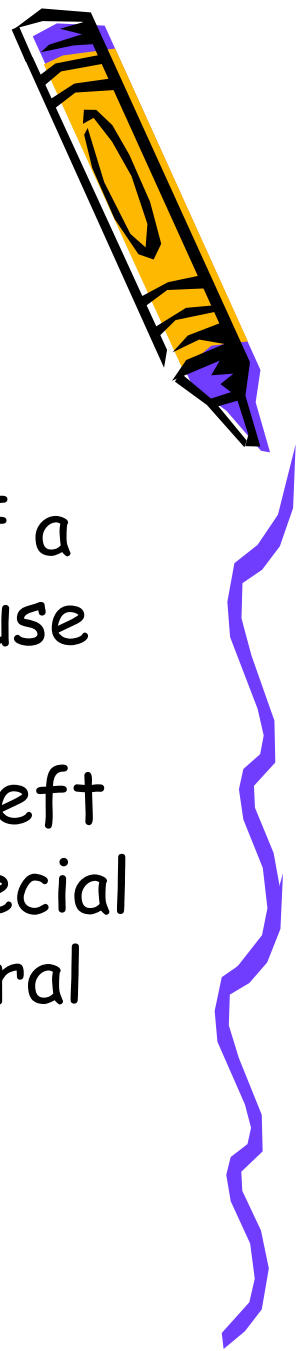


Example of Complex Group Testing

- In screening clone library the goal is to determine which clones in the clone library hybridize with a given probe in an efficient fashion. A clone is said to be **positive** if it hybridize with the **probe**, and **negative** otherwise.



Shotgun Sequencing



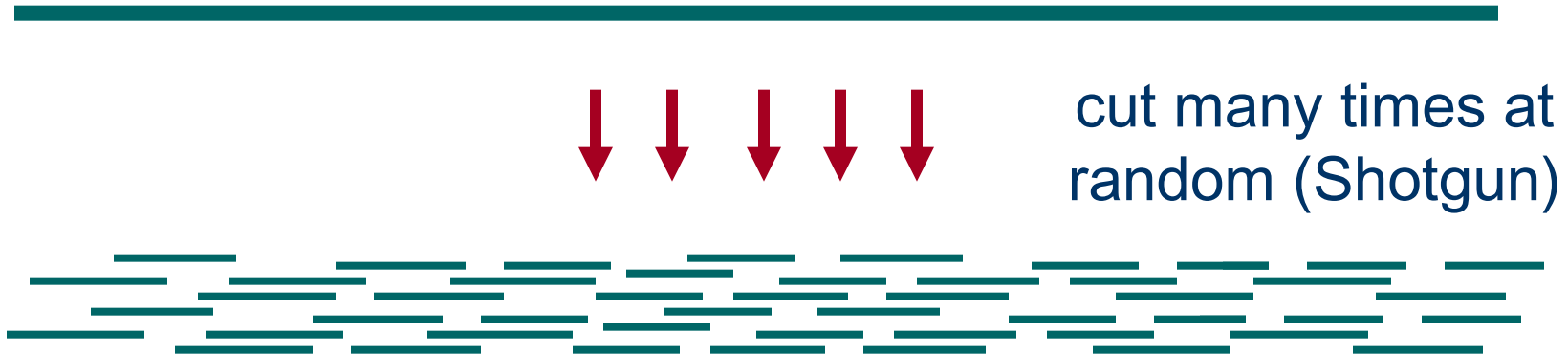
- Shotgun sequencing is a throughput technique resulting in the sequencing of a large number of bacterial genomes, mouse genomes and the celebrated human genomes. In all such projects, we are left with a collection of contigs that for special reasons cannot be assembled with general assembly algorithms. Continued ...



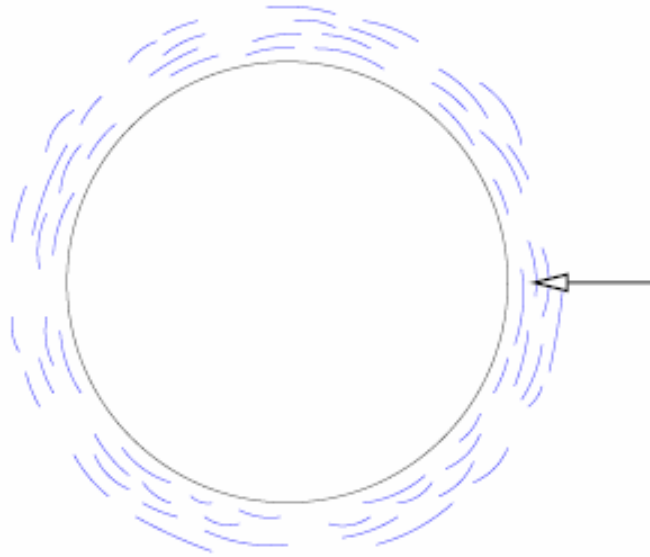
Random shotgun approach



genomic segment

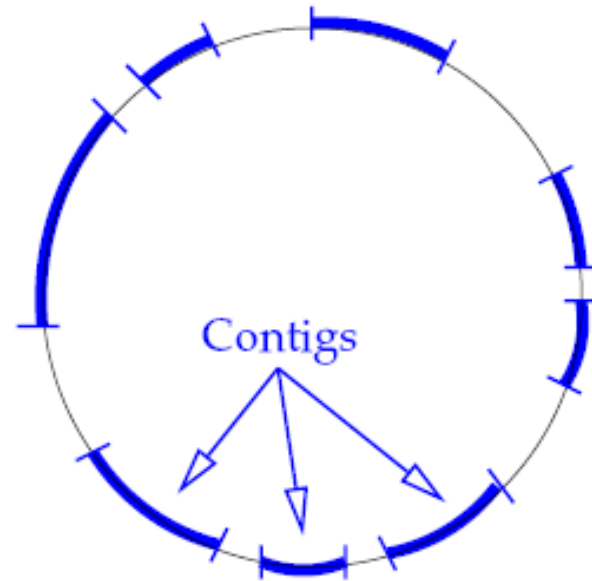


- Whole-genome shotgun sequencing
 - Short reads are obtained and covering the genome with redundancy and possible gaps.

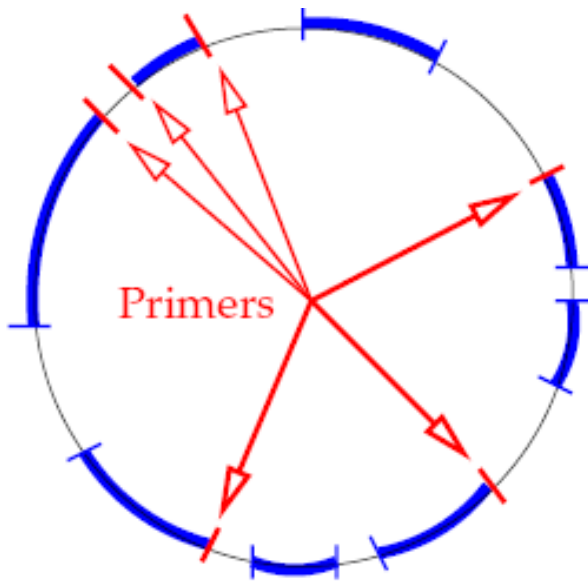


Circular genome

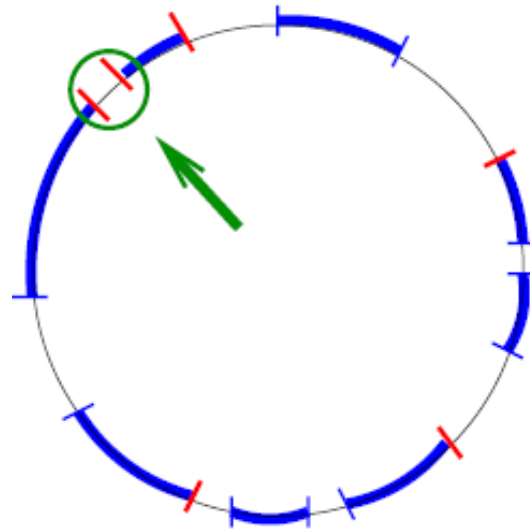
- Reads are assembled into contigs with unknown relative placement.

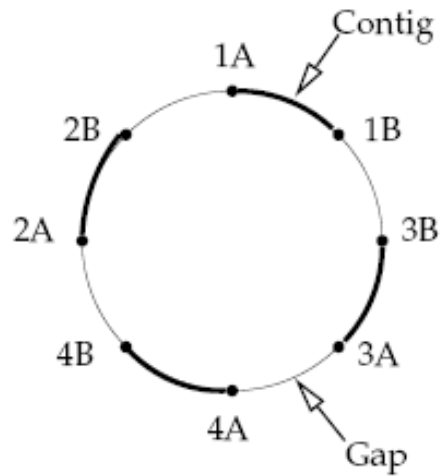


- Primers : (short) fragments of DNA characterizing ends of contigs.



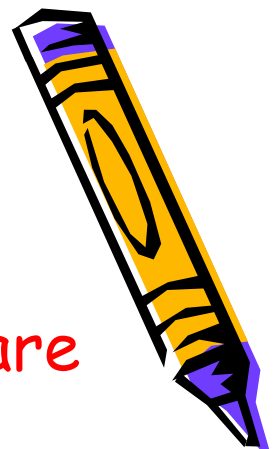
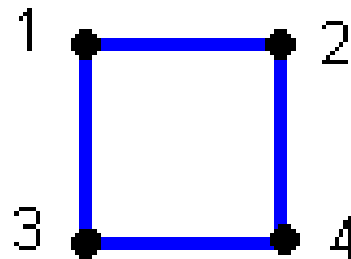
- A **PCR (Polymerase Chain Reaction)** reaction reveals if two primers are proximate (adjacent to the same gap).
- **Multiplex PCR** can treat multiple primers simultaneously and outputs if there is a pair of adjacent primers in the input set and even sometimes the number of such pairs.

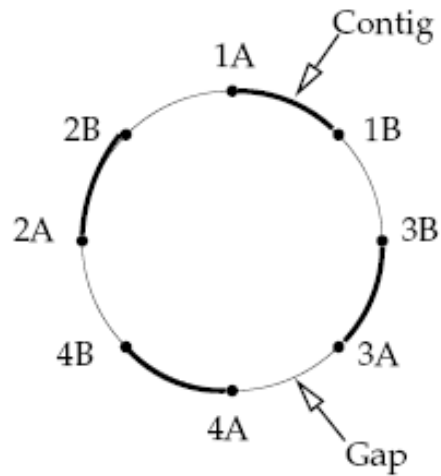




- Two primers of each contig are “mixed together”
 - Find a Hamiltonian cycle by PCRs!

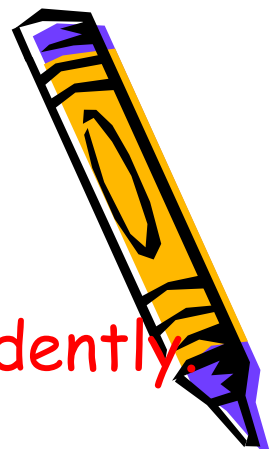
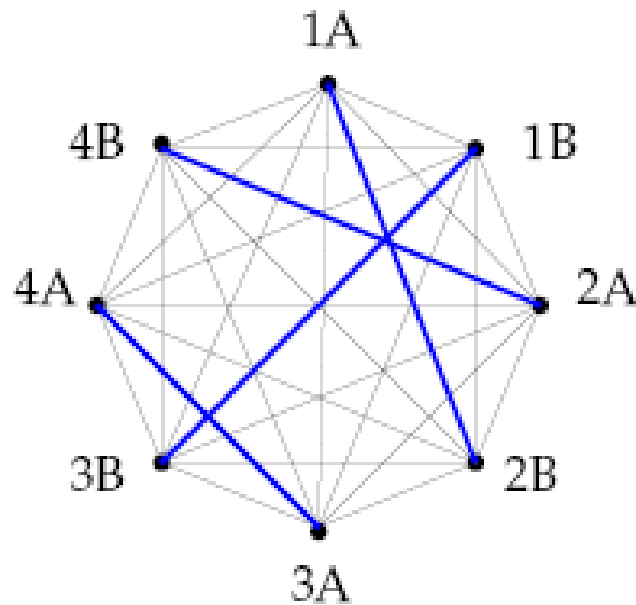
1A ——— 1B
 2A ——— 2B
 3A ——— 3B
 4A ——— 4B





1A ——— 1B
 2A ——— 2B
 3A ——— 3B
 4A ——— 4B

- Primers are treated independently.
 - Find a perfect matching by PCRs.



Goal



- Our goal is to provide an experimental protocol that identifies all pairs of adjacent primers with as few PCRs (queries) (or multiplex PCRs respectively) as possible.



Another Examples

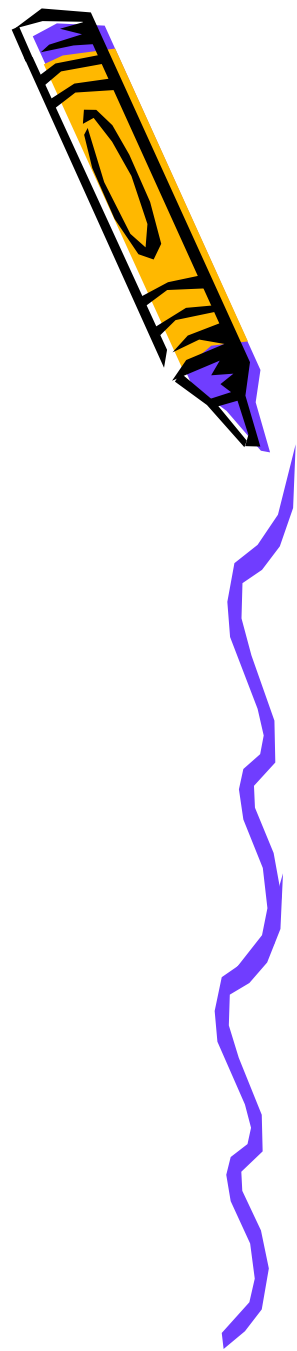


- Eukaryotic DNA transcription and RNA translation (Torney, 1999)
- Identification of protien-to-protien interactions (Lappe and Holm, 2003)
- Superimposed codes and secure key distribution
- Data Pattern



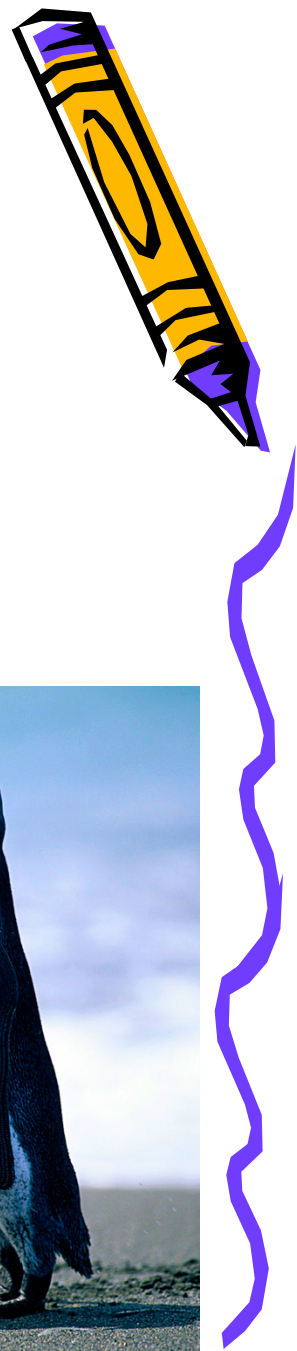
A great reference

- Combinatorial Group Testing problems on Various Models, Ph. D. Thesis (2010), NCTU.
(By Dr. Huilan Chang)



Mathematical Models

Hidden Graphs (Reconstructed)



Idea



- Edge-detecting queries: $Q_G(S)$
- $Q_G(S) = 1$ if the induced subgraph of the vertex subset S of $V(G)$ contains at least one edge of G and $Q_G(S) = 0$ otherwise.
- For convenience, we assume G is defined on $[1, n] = \{1, 2, \dots, n\}$.



Recent Result



- **Theorem** (with Huilan Chand and Chie-Huai Shih, Optimization Letters, 2014)

Let G be a graph with n vertices and m edges. Then we have an adaptive *algorithm* using at most $m(2\log_2 n + 9)$ queries.



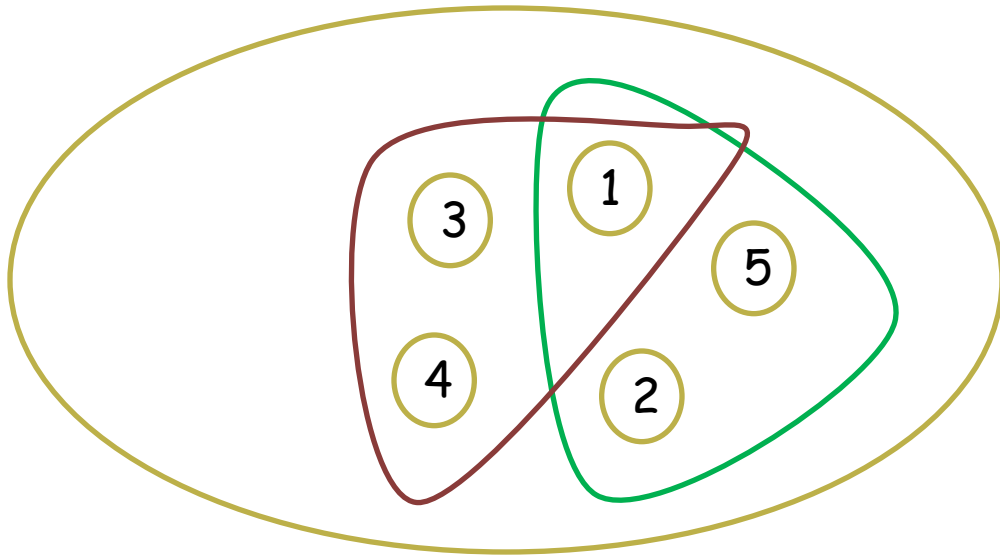
Hidden hypergraphs



- It deals with the case that a complex may contain more than two items.
- The number of positives is also unknown (**Competitive model**).
- The following example shows an idea of finding a hidden 3-uniform hypergraph.



Find a hyperedge



$\{1,2,3,4,5\}$

$Q(\{1,2,3,4,5\})$

$Q(\{1,2,3\})$

$Q(\{1,2,3,4\})$

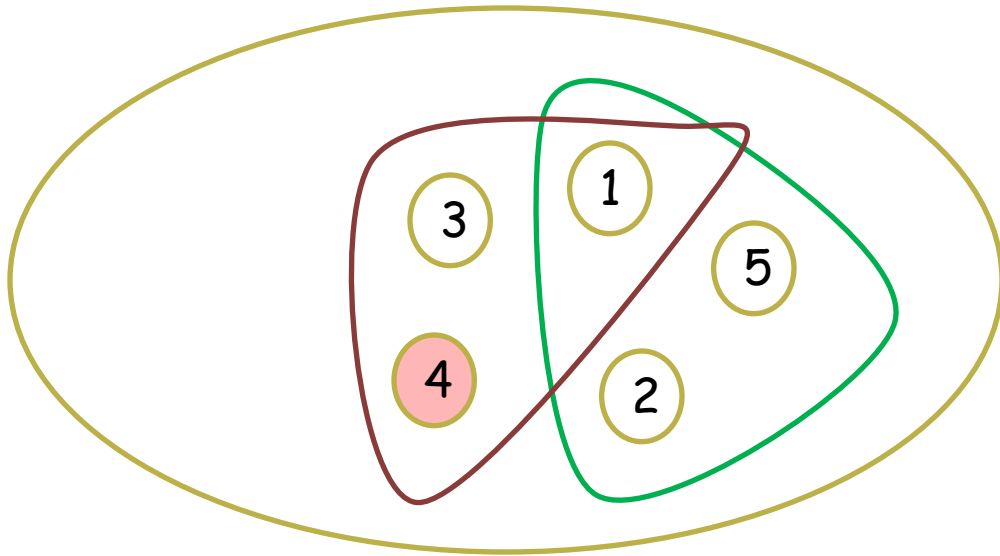
Yes

No

Yes



Find a hyperedge



Algorithm

$\{1, 2, 3, 4, 5\}$

$Q(\{1, 2, 3, 4, 5\})$

Yes

$Q(\{1, 2, 3\})$

No

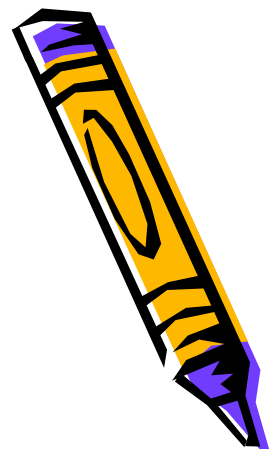
$Q(\{1, 2, 3, 4\})$

Yes

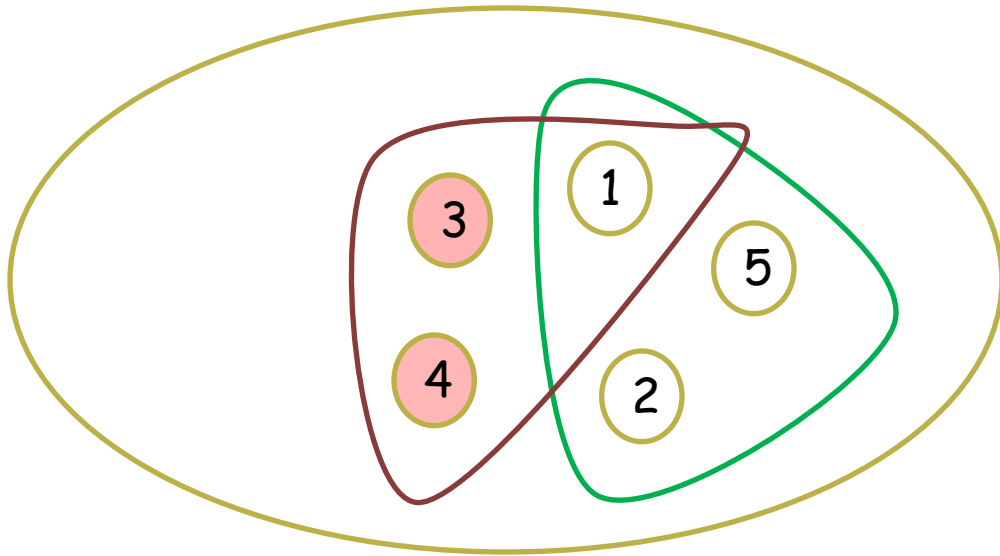
$\{4, 1, 2, 3\}$

$Q(\{4, 1, 2\})$

No



Find a hyperedge



Algorithm

$\{1,2,3,4,5\}$

$Q(\{1,2,3,4,5\})$

Yes

$Q(\{1,2,3\})$

No

$Q(\{1,2,3,4\})$

Yes

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

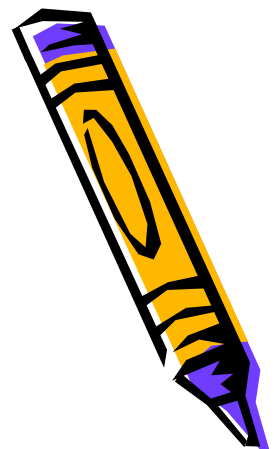
$Q(\{4,1,2\})$

No

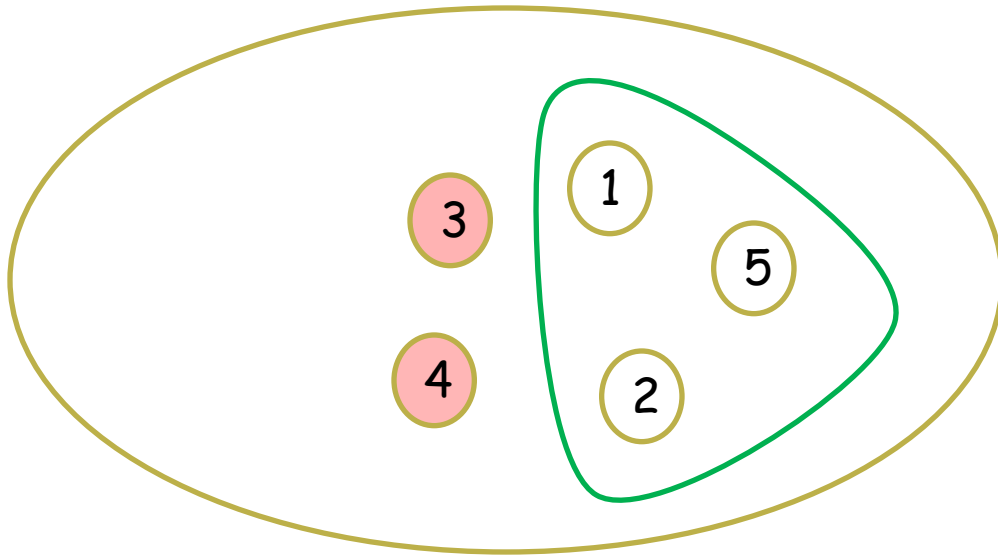
$\{4,3,1,2\}$

$Q(\{4,3,1\})$

Yes



Find another hyperedge



Algorithm

$\{1,2,3,4,5\}$

$Q(\{1,2,3,4,5\})$

Yes

$Q(\{1,2,3\})$

No

$Q(\{1,2,3,4\})$

Yes

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

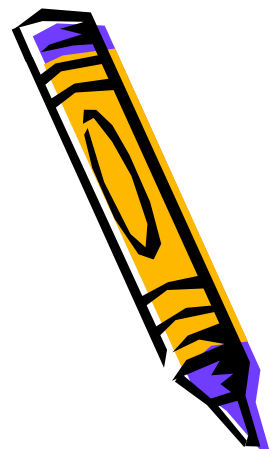
$Q(\{4,1,2\})$

No

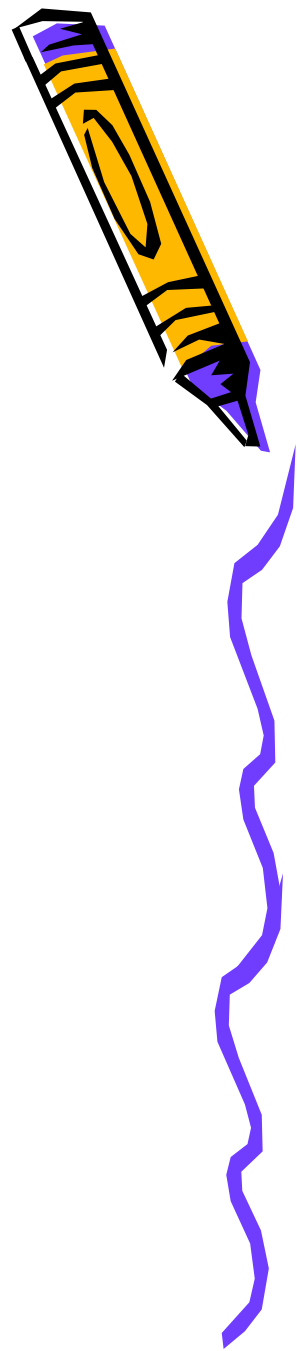
$\{4,3,1,2\}$

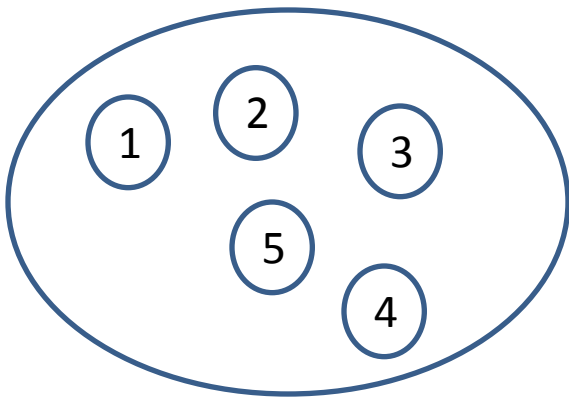
$Q(\{4,3,1\})$

Yes



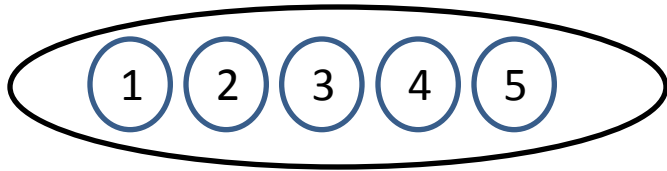
Better idea for more
hyperedges





$V = \{1, 2, 3, 4, 5\}$

$E = \{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{1, 4, 5\}\}$

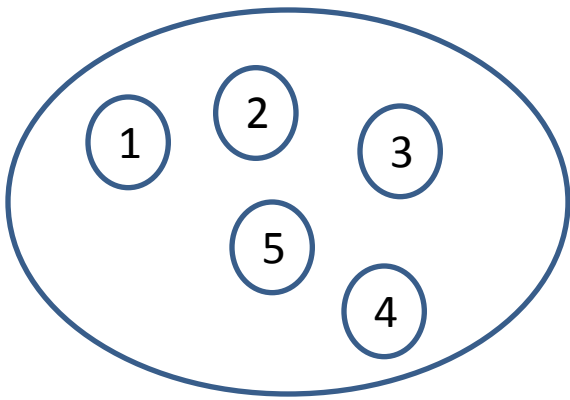


S_1

Initial setting:

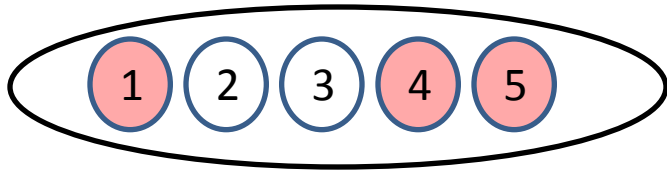
$S_1 = \{1, 2, 3, 4, 5\}$

Find $\{1, 4, 5\}$



$V = \{1, 2, 3, 4, 5\}$

$E = \{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{1, 4, 5\}\}$

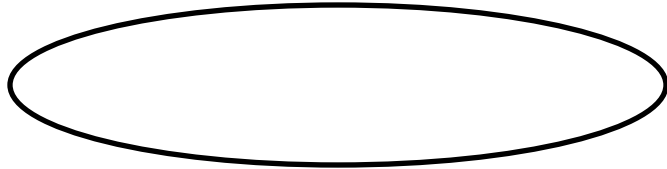
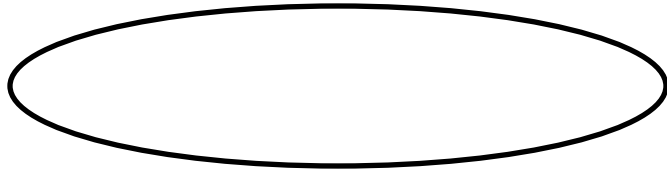


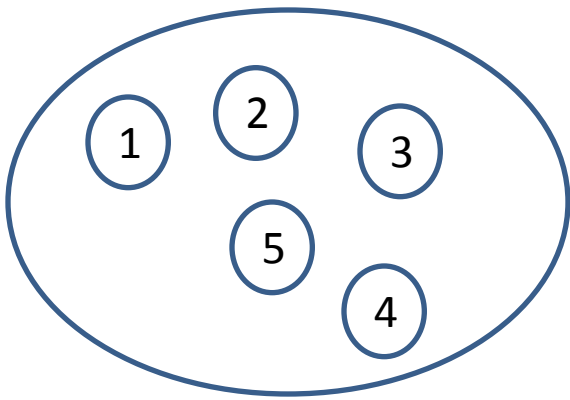
S_0

Initial setting:

$S_0 = \{1, 2, 3, 4, 5\}$

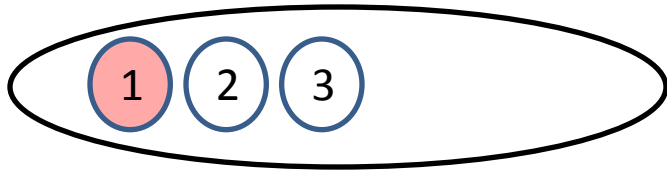
Find $\{1, 4, 5\}$



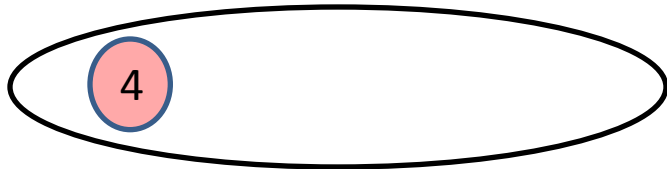


$V = \{1, 2, 3, 4, 5\}$

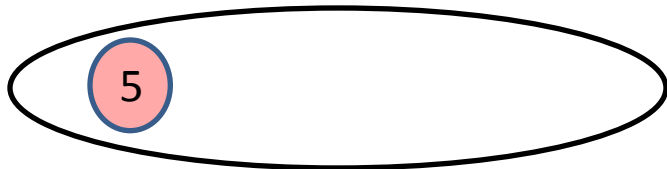
$E = \{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{1, 4, 5\}\}$



S_1



S_2

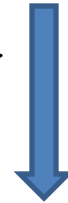


S_3

Initial setting:

$S_1 = \{1, 2, 3, 4, 5\}$

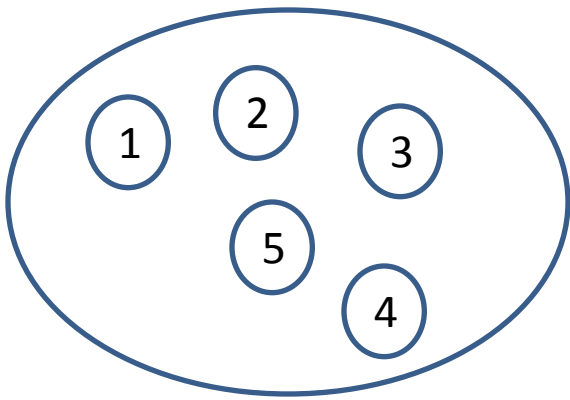
Find $\{1, 4, 5\}$



$S_1 = \{1, 2, 3\}$

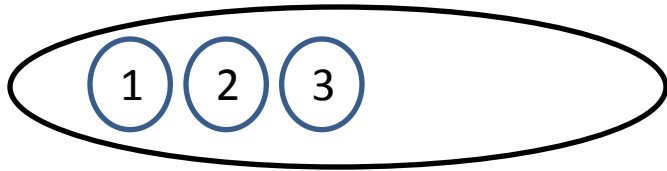
$S_2 = \{4\}$

$S_3 = \{5\}$

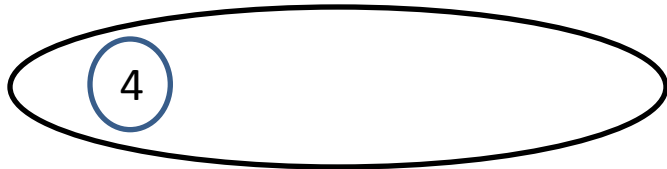


$V = \{1, 2, 3, 4, 5\}$

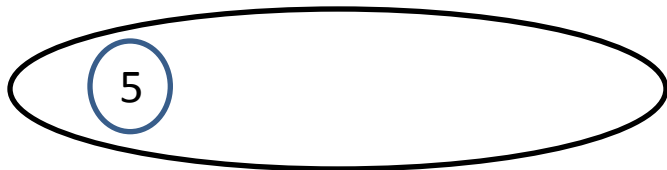
$E = \{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{1, 4, 5\}\}$



S_1



S_2

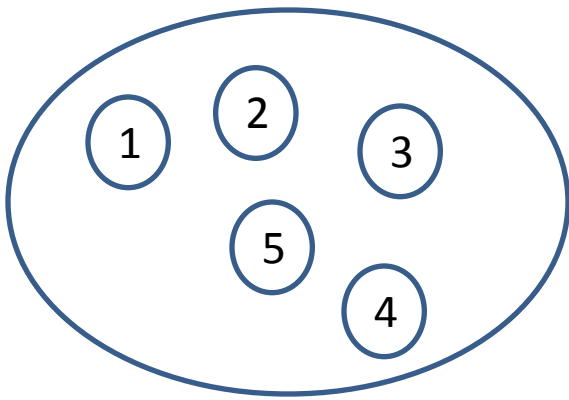


S_3

$S_1 = \{1, 2, 3\}$

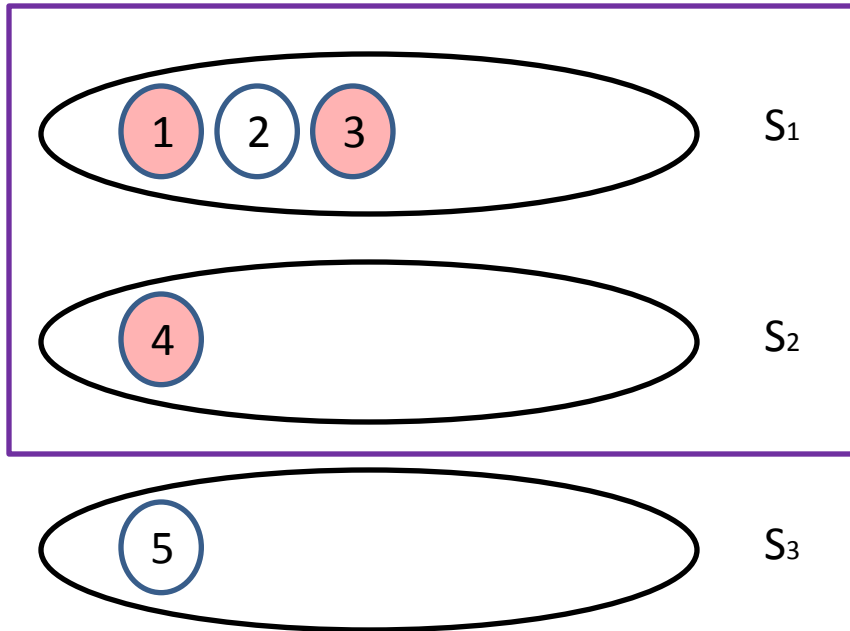
$S_2 = \{4\}$

$S_3 = \{5\}$



$V = \{1, 2, 3, 4, 5\}$

$E = \{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{1, 4, 5\}\}$



$S_1 = \{1, 2, 3\}$

$S_2 = \{4\}$

$S_3 = \{5\}$

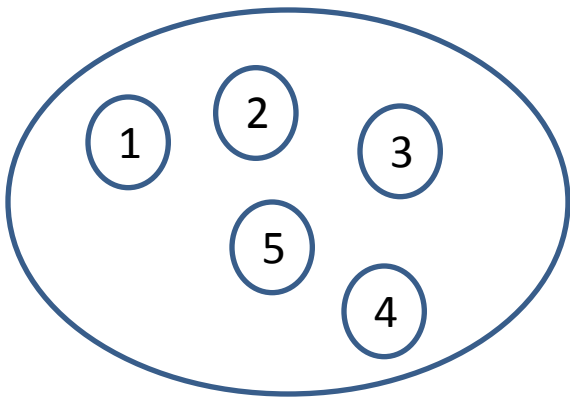
Find $\{1, 3, 4\}$



$S_1 = \{1, 2\}$

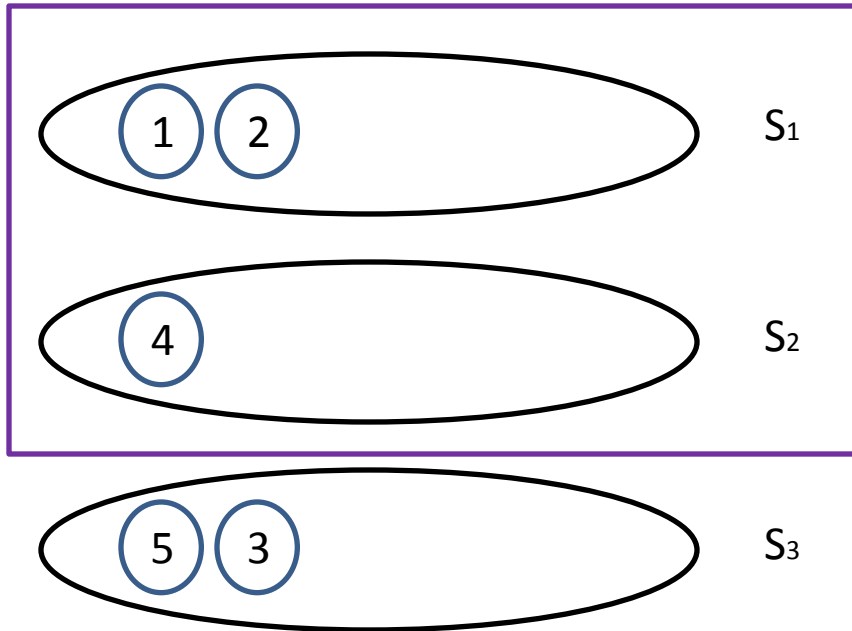
$S_2 = \{4\}$

$S_3 = \{3, 5\}$



$V = \{1, 2, 3, 4, 5\}$

$E = \{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{1, 4, 5\}\}$

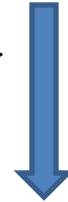


$S_1 = \{1, 2\}$

$S_2 = \{4\}$

$S_3 = \{3, 5\}$

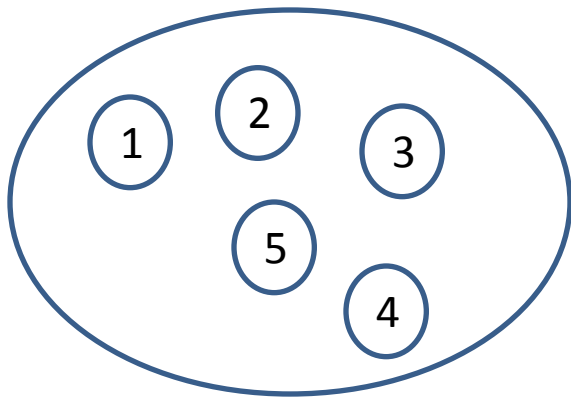
Find $\{1, 2, 4\}$



$S_1 = \{1\}$

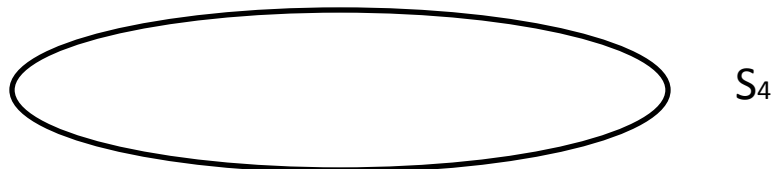
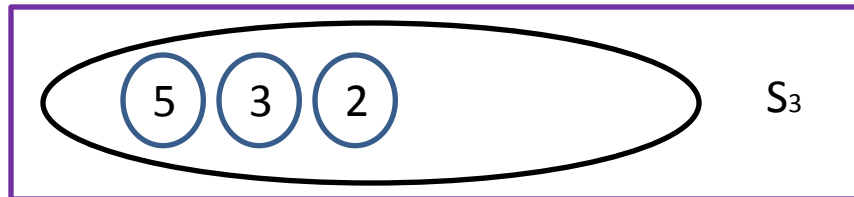
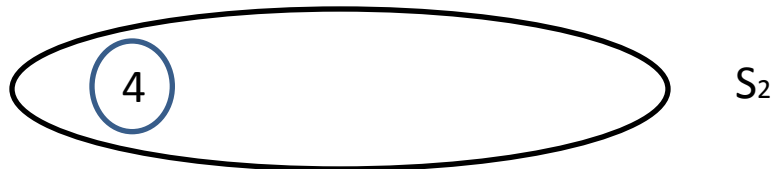
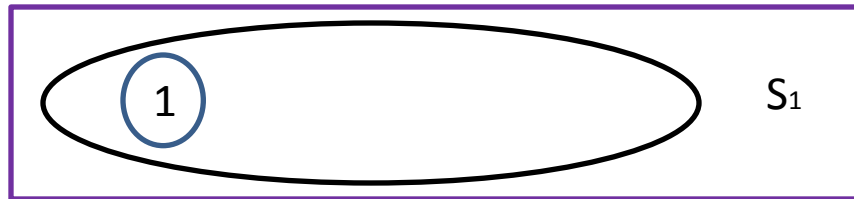
$S_2 = \{4\}$

$S_3 = \{2, 3, 5\}$



$$V = \{1, 2, 3, 4, 5\}$$

$$E = \{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{1, 4, 5\}\}$$



$$S_1 = \{1\}$$

$$S_2 = \{4\}$$

$$S_3 = \{2, 3, 5\}$$

Find $\{1, 2, 3\}$

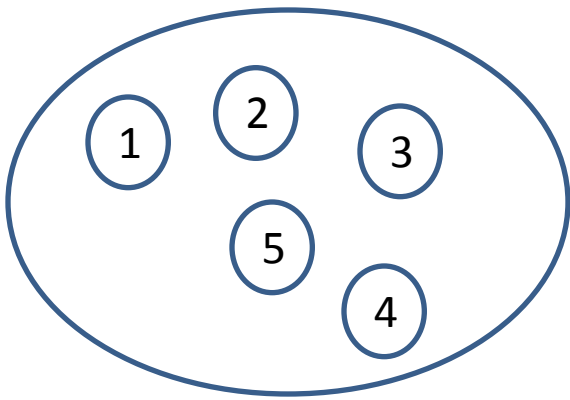


$$S_1 = \{1\}$$

$$S_2 = \{4\}$$

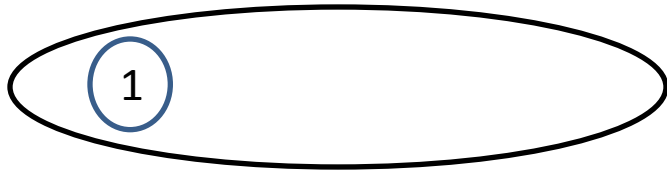
$$S_3 = \{2, 5\}$$

$$S_4 = \{3\}$$

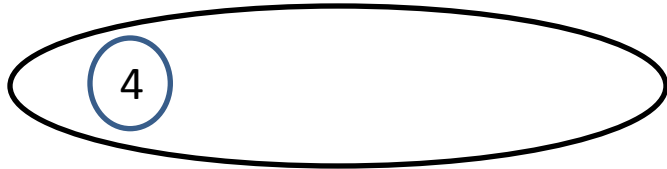


$V = \{1, 2, 3, 4, 5\}$

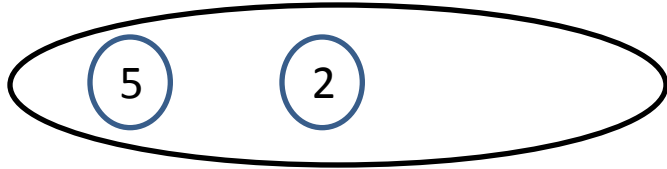
$E = \{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{1, 4, 5\}\}$



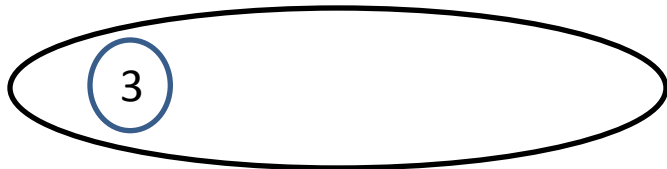
S_1



S_2



S_3



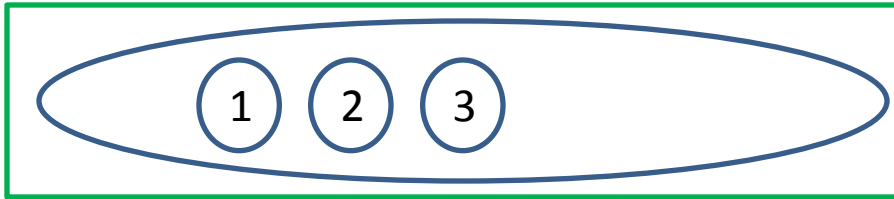
S_4

$S_1 = \{1\}$

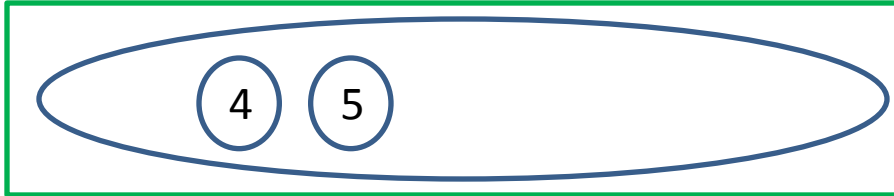
$S_2 = \{4\}$

$S_3 = \{2, 5\}$

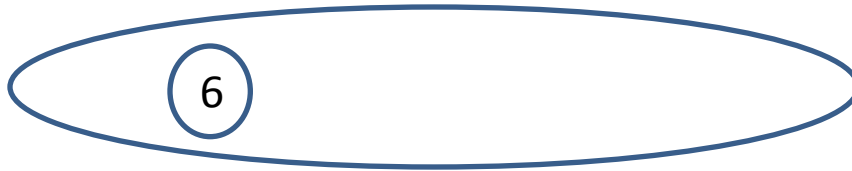
$S_4 = \{3\}$



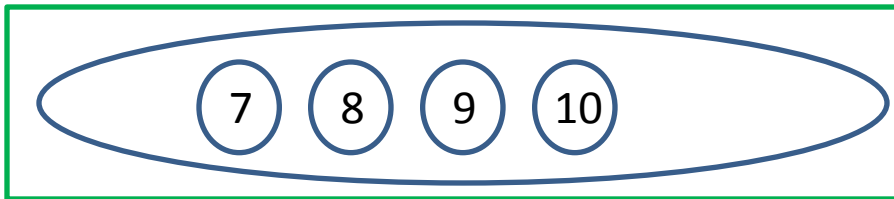
S_1



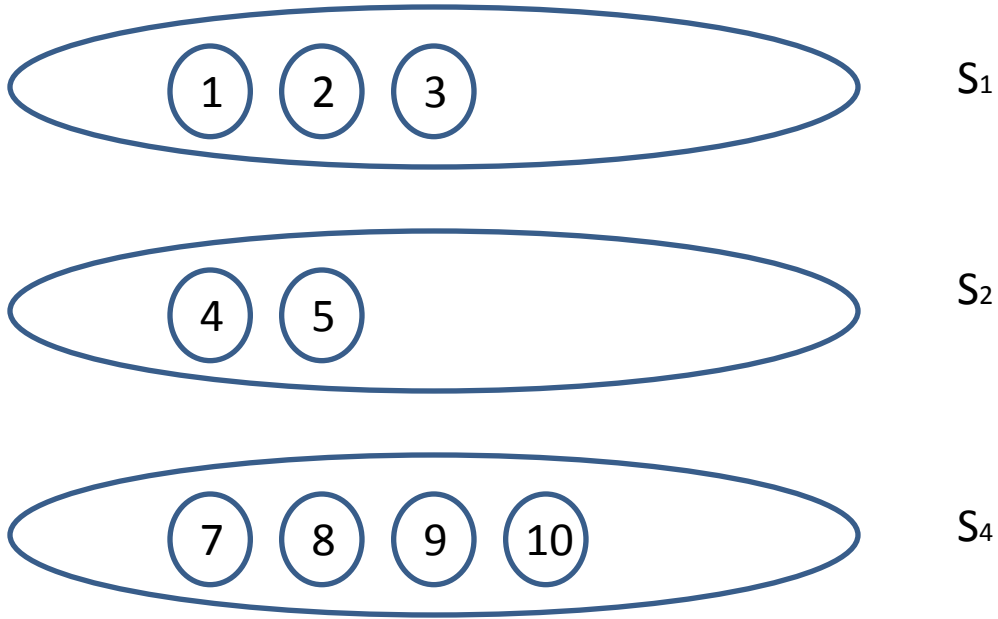
S_2



S_3

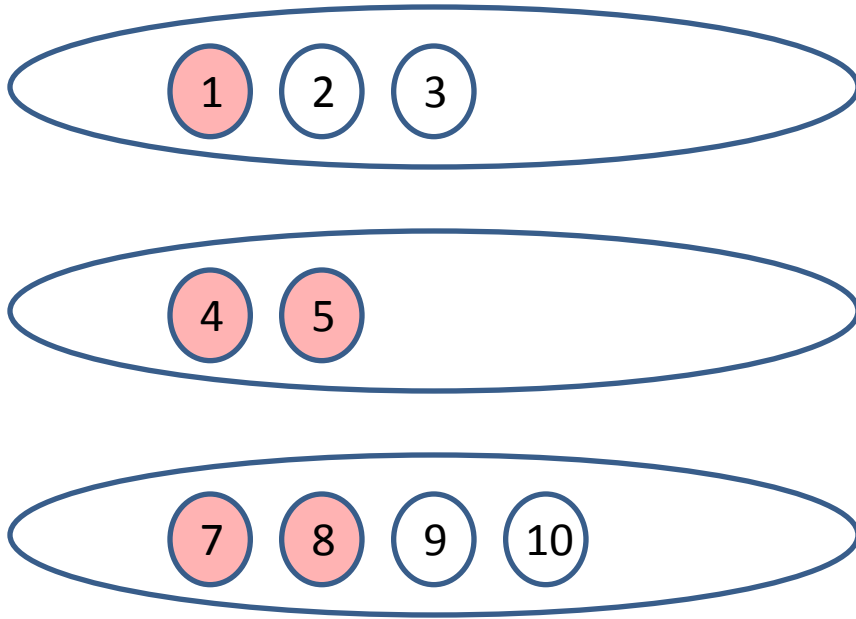


S_4



$$E(G[S_1 \cup S_2 \cup S_4]) = \{\boxed{\{1,4,7\}}, \boxed{\{1,5,8\}}, \{1,4,9\}, \{1,5,7\}\}$$

hidden



S_1

$$Q(S_1 \cup S_2 \cup S_4 - \{1\})$$

NO

S_2

$$Q(S_1 \cup S_2 \cup S_4 - \{4, 5\})$$

NO

S_4

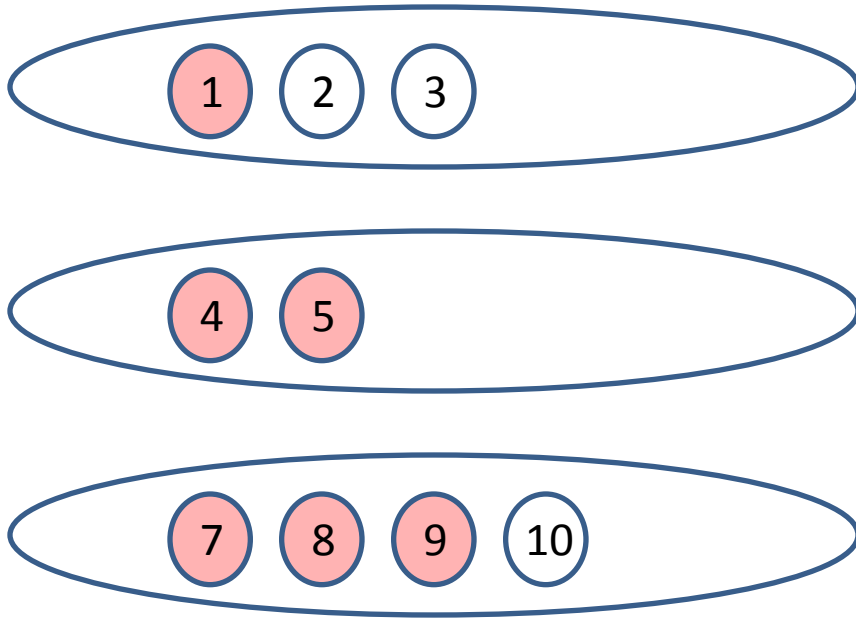
$$Q(S_1 \cup S_2 \cup S_4 - \{7, 8\})$$

Yes

Find $\{1, 4, 9\}$

$$E(G[S_1 \cup S_2 \cup S_4]) = \{\{1, 4, 7\}, \{1, 5, 8\}, \{1, 4, 9\}, \{1, 5, 7\}\}$$

hidden



S_1

$$Q(S_1 \cup S_2 \cup S_4 - \{1\})$$

NO

S_2

$$Q(S_1 \cup S_2 \cup S_4 - \{4, 5\})$$

NO

S_4

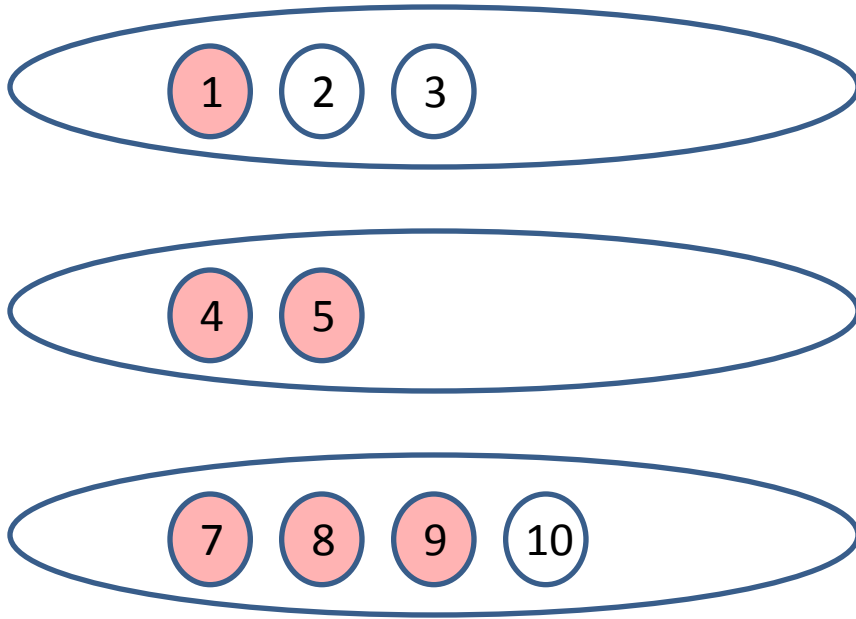
$$Q(S_1 \cup S_2 \cup S_4 - \{7, 8, 9\})$$

NO

$$E(G[S_1 \cup S_2 \cup S_4]) = \{\{1, 4, 7\}, \{1, 5, 8\}, \{1, 4, 9\}, \{1, 5, 7\}\}$$



hidden



S_1

$Q(\{1\} \cup \{4\} \cup \{7,9\})$

NO

S_2

$Q(\{1\} \cup \{5\} \cup \{7,9\})$

Yes

S_4

$Q(\{1\} \cup \{5\} \cup \{7\})$ **Yes**

DONE!

$$E(G[S_1 \cup S_2 \cup S_4]) = \{\{1,4,7\}, \{1,5,8\}, \{1,4,9\}, \{1,5,7\}\}$$



hidden

Conclusion



- We can find a hidden r -uniform hypergraph G with $m \cdot r \cdot \log_2 n + (2^{r+2})^{1/2} r^r m^{r/2}$ edge-detecting queries where m is the number of edges in G .
- It is by *adaptive algorithm*.



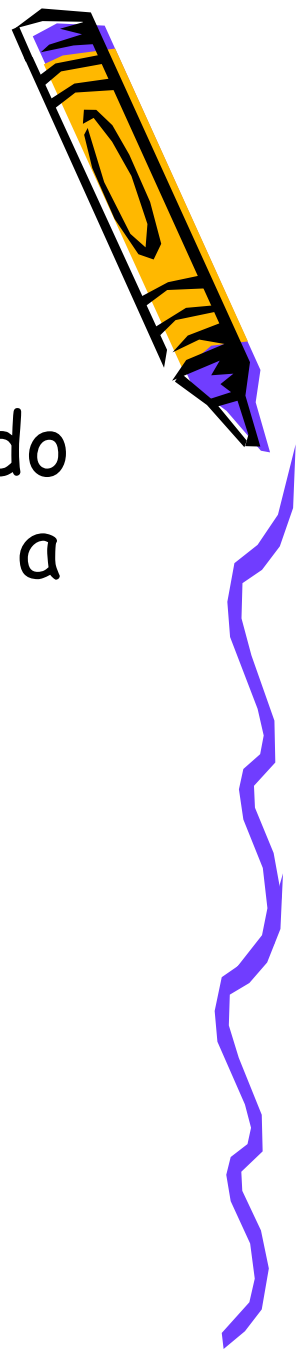
Problems



- We believe that **the best** answer should be around $(m \log_2 n)$ queries, but not able to prove this fact.
- We have no **good** idea about **non-adaptive algorithm** for optimal result at this moment.



More Problems



- If there is an *inhibitor*, then how do we use *adaptive algorithm* to learn a hidden graph?
- What if *errors* occurred when *adaptive algorithm* is utilized?



Reference



- A new construction of 3-bar-*separable* matrices via improved decoding of Macula construction (with F. K. Hwang), **Discrete Optim.**, 5(2008), 700-704.
- An upper bound of the number of tests in pooling designs for the *error-tolerant complex model* (with Hong-Bin. Chen and F. K. Hwang), **Optimization Letters**, 2(2008), no. 3, 425-431.
- The minimum number of e-vertex-cover among *hypergraphs* with e edges of given rank (with F. H. Chang, F. K. Hwang and B. C. Lin), **Discrete Applied Math.**, 157(2009), 164-169.
- Non-adaptive algorithms for *threshold* group testing (with Hong-Bin Chen), **Discrete Applied Math.**, 157(2009), 1581-1585.



Continued



- Identification and classification problem on pooling designs for *inhibitor* models (with Huilan Chang and Hong-Bin Chen), **J. Computational Biology**, 17(2010), No. 7, 927-941.
- Reconstruction of *hidden graphs and threshold* group testing (with Huilan Chang, Hong-Bin Chen and Chi-Huai Shih), **J. Combin. Optimization**, 22(2011), no. 2, 270 - 281.
- Group testing with multiple *mutually-obscuring positives* (with Hong-Bin Chen), **Lecture Notes Computer Science**, 7777(2013), 557 - 568.
- *Threshold* group testing on *inhibitor* model (with Huilan Chang and Chie-Huai Shih), **J. Computational Biology**, Vol. 20, No. 6, 2013, 1 - 7.



Never ever give up!

