On Generation of Graphs with Geometric Representations

Ryuhei Uehara

Japan Advanced Institute of Science and Technology (JAIST)

based on the following papers:

- T. Saitoh, Y. Otachi, K. Yamanaka, and R. Uehara: Random Generation and Enumeration of Bipartite Permutation Graphs, *Journal of Discrete Algorithms*, Vol. 10, pp. 84-97, January, 2012.

- T. Saitoh, K. Yamanaka, M. Kiyomi and R. Uehara: Random Generation and Enumeration of Proper Interval Graphs, *IEICE Transactions*, Vol. E93-D, No. 7, pp. 1816-1823, 2010.

"Random Generation and Enumeration of ??? graphs" (w/o labels)

- proper interval graphs
- bip. permutation graphs

…graphs that have *geometric representations*

- No skip
- No duplicate

… up to isomorphism

…ving papers:

…amanaka, and R. Uehara: Random G…eration and Enumeration of Bipartite Permutatio… …phs, *Journal of*

…Graphs, *IEICE Transactions*, Vol. E93-D, No. 7, pp. 1816-1823, 2010.

Our Algorithms

- Random Generation
  - Input: Natural number *n*
  - Output: Connected ??? graph of *n* vertices
    - Uniformly at random
    - Using a counting algorithm
    - O(*n*+*m*) time (m: #edges)

Focus on this topic

- Enumeration
  - Input: Natural number *n*
  - Output: All the connected ??? graphs of *n* vertices
    - Without duplication
    - Based on reverse search algorithm
    - O(1) time/graph

Skip this topic, sorry!

Known Algorithms

- Generation of a string of parentheses
  - D.B.Arnold and M.R.Sleep, 1980
    - can't generate P.I.G. uniformly at random

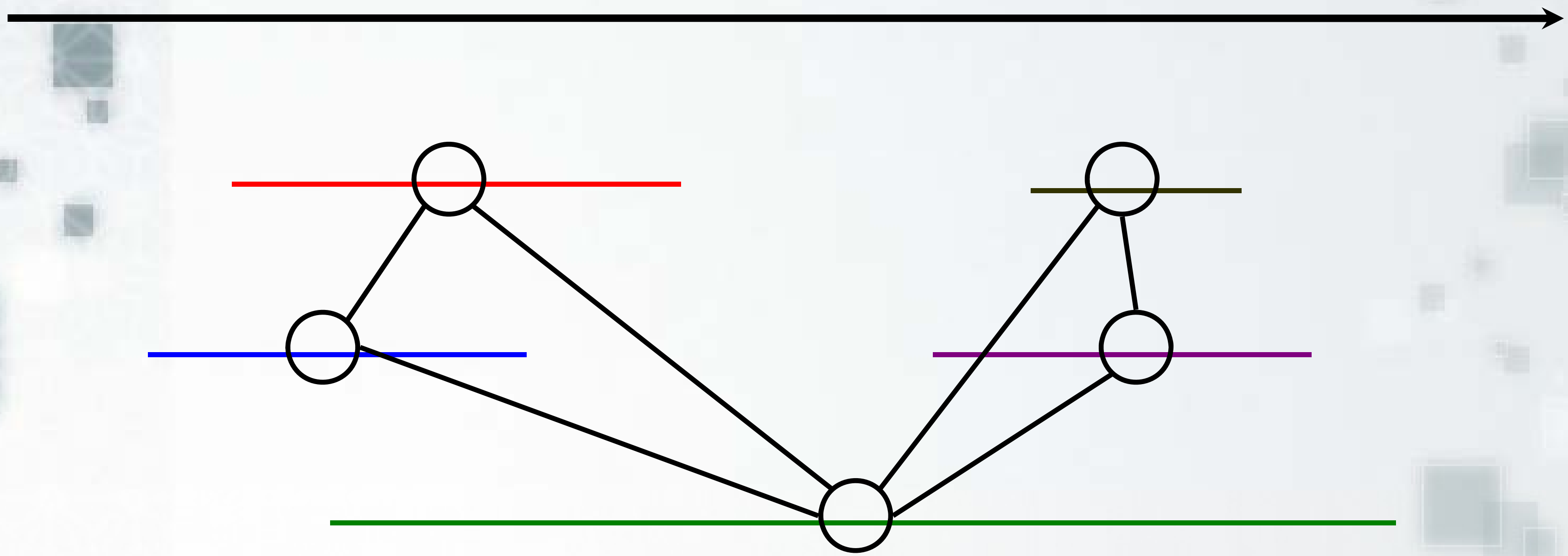
      Not one-to-one correspondence
- Enumeration of strings of parentheses
  - D.E.Knuth, 2005
    - can't enumerate every P.I.G. in O(1) time

      constant size of differences in string

      ⇔ large size of differences in P.I.G.

Interval Graphs

- Have interval representations

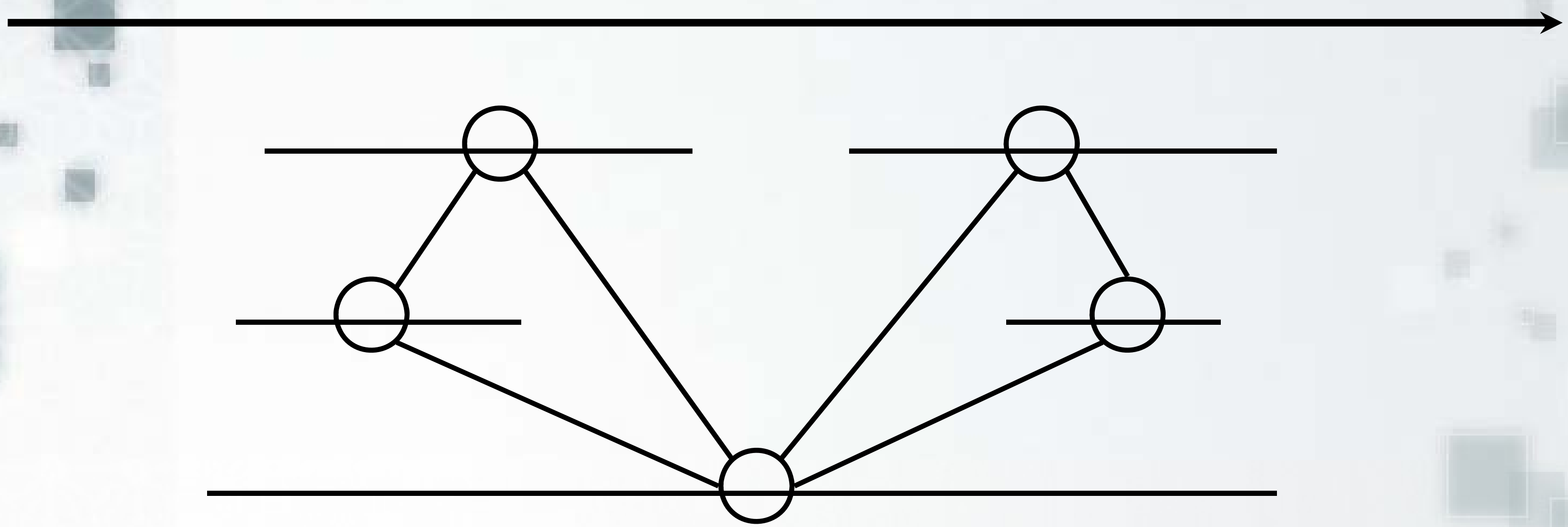


Proper Interval Graphs = Unit interval graphs

Every interval has the same length

- Have unit interval representations

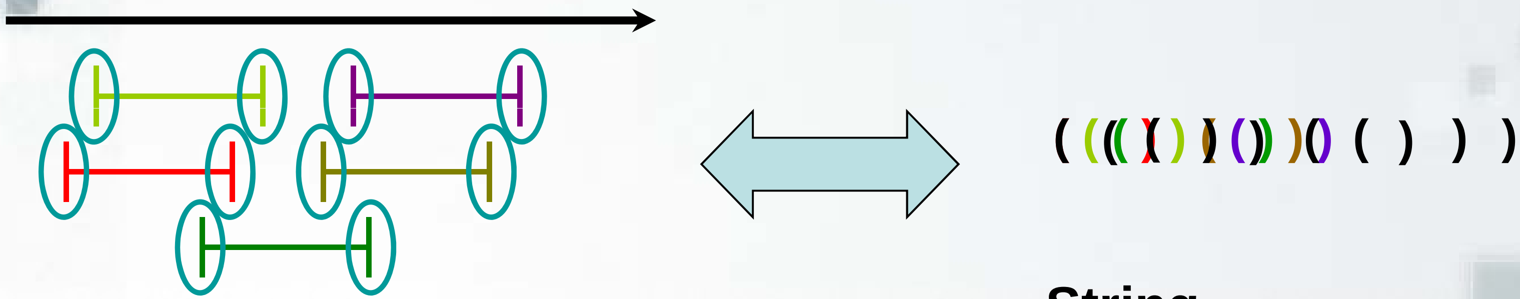
String representations



Definition

- String Representation
  - Encodes a unit interval representation by a string
    - Sweep the unit interval representation from left to right
      - Left endpo
      - Right endpoi → ")" : right parenthesis

Right endpoints appear in order of their left endpoint appearances



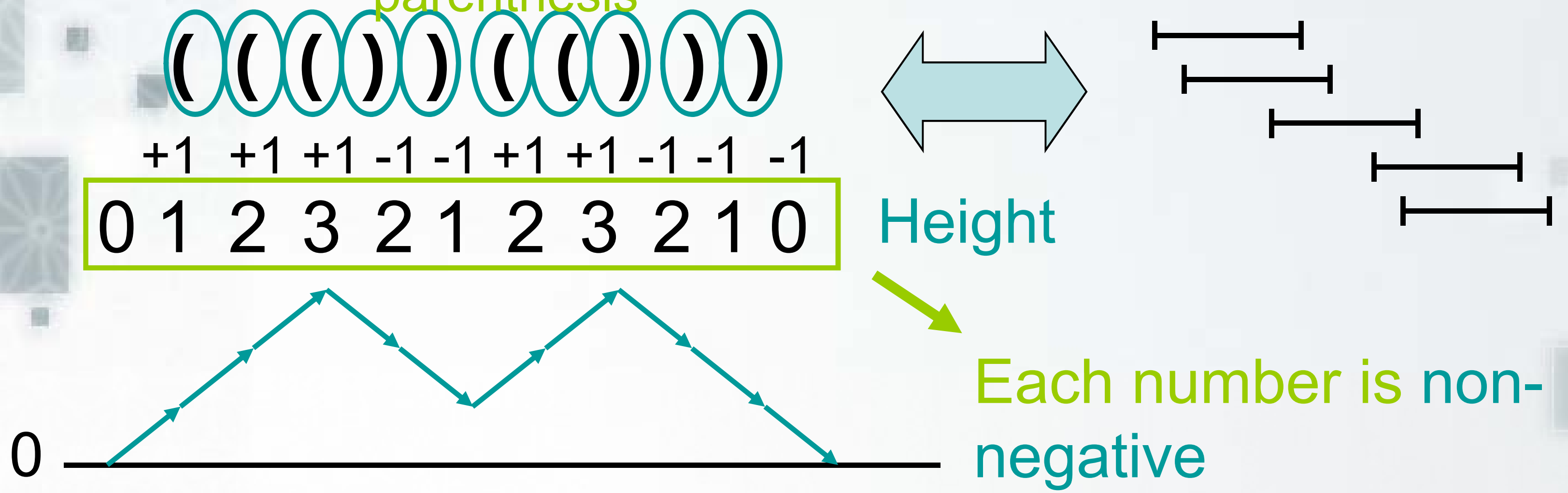
Unit Interval Representation

String Representation

String Representation

Height = # "(" - # ")"

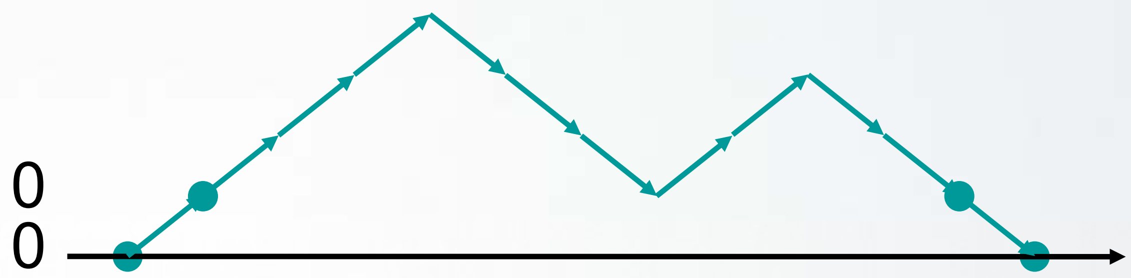
- **Property of string rep. of P. I. G. of *n* vertices**
  - Number of parentheses: 2*n*
    - Number of "(": *n*   Number of ")": *n*
  - Non-negative
    - Each left parenthesis exists in the left side of its right parenthesis



String Representation

- Observation 1
  - String rep. of connected P. I. G.
    - Have exactly 2 places whose heights are 0.
      - The left end and the right end

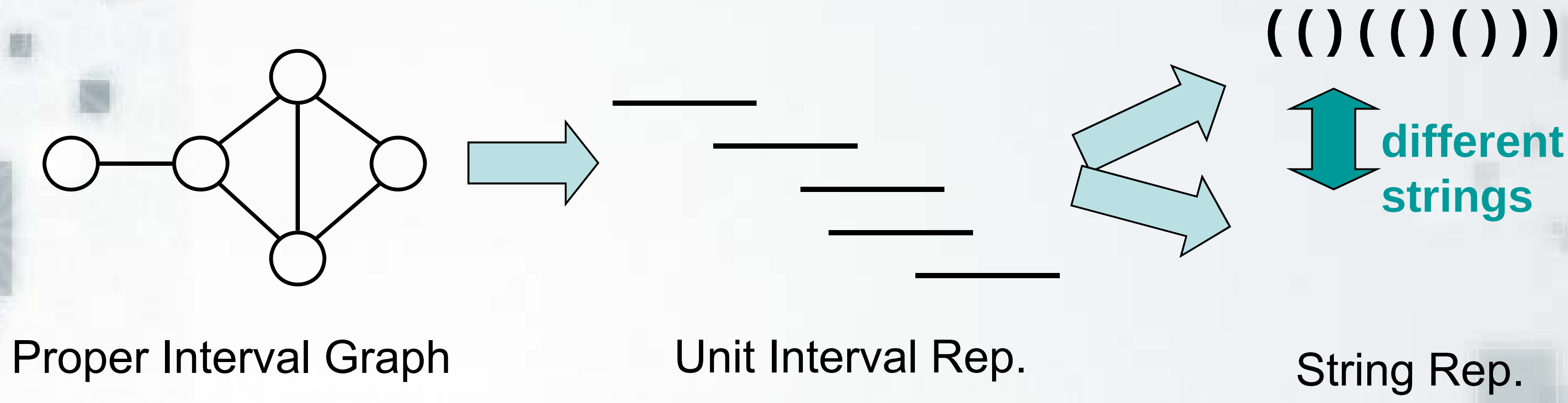
The string removing both ends parentheses is non-negative (((( ))) (( ))))



String Representation

- Lemma 1. (X. Dell, P. Hell, J. Huang, 1996)
  - A connected P. I. G. has only one or two string rep.

This graph has only two string representations.



String Representation

- **Lemma 1.** (X. Dell, P. Hell, J. Huang, 1996)
  - A connected P. I. G. has only one or two string rep.

This graph has only one string representation.

Proper Interval Graph — Unit Interval Rep. — String Rep.

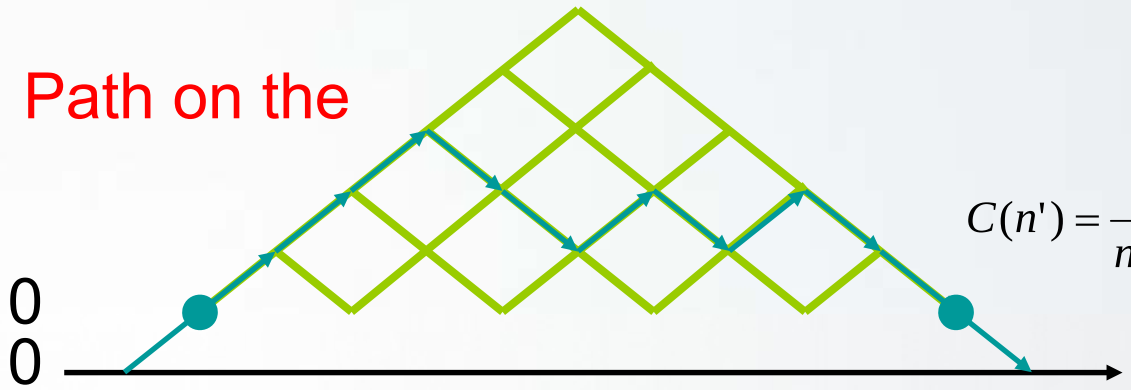
reversible

Random Generation Algorithm

- Generate a string rep. uniformly at random
  - Using a counting algorithm
    - (Generalized) Catalan number

(((( )) ( ) ( )))

String rep. : Path on the area



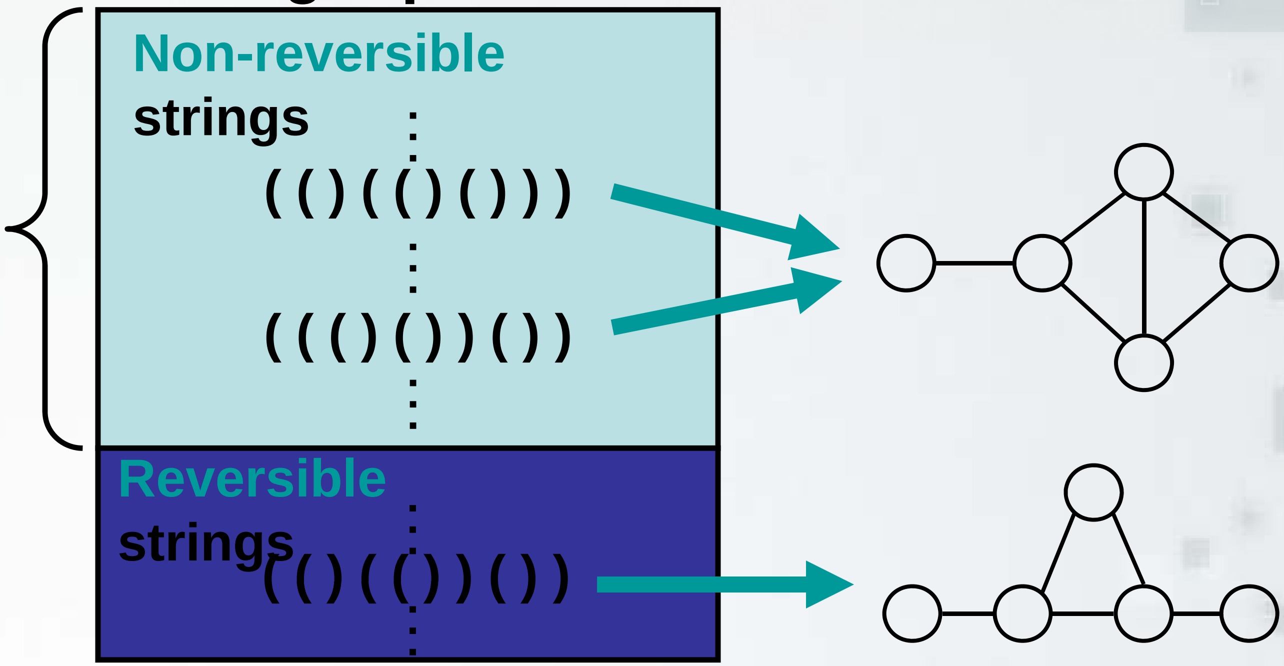
$$C(n') = \frac{1}{n'+1}\binom{2n'}{n'}$$

Adjust the Generation Probability

Not easy

Decrease the generation probability

String rep.



A generation probability of a graph corresponding to non-reversible strings is higher than that of reversible one

Adjust the Generation Probability

S_n: # non-reversible strings

R_n: # reversible strings

$$S_n + R_n = C(n)$$

$$R_n = \binom{n}{\lfloor n/2 \rfloor}$$

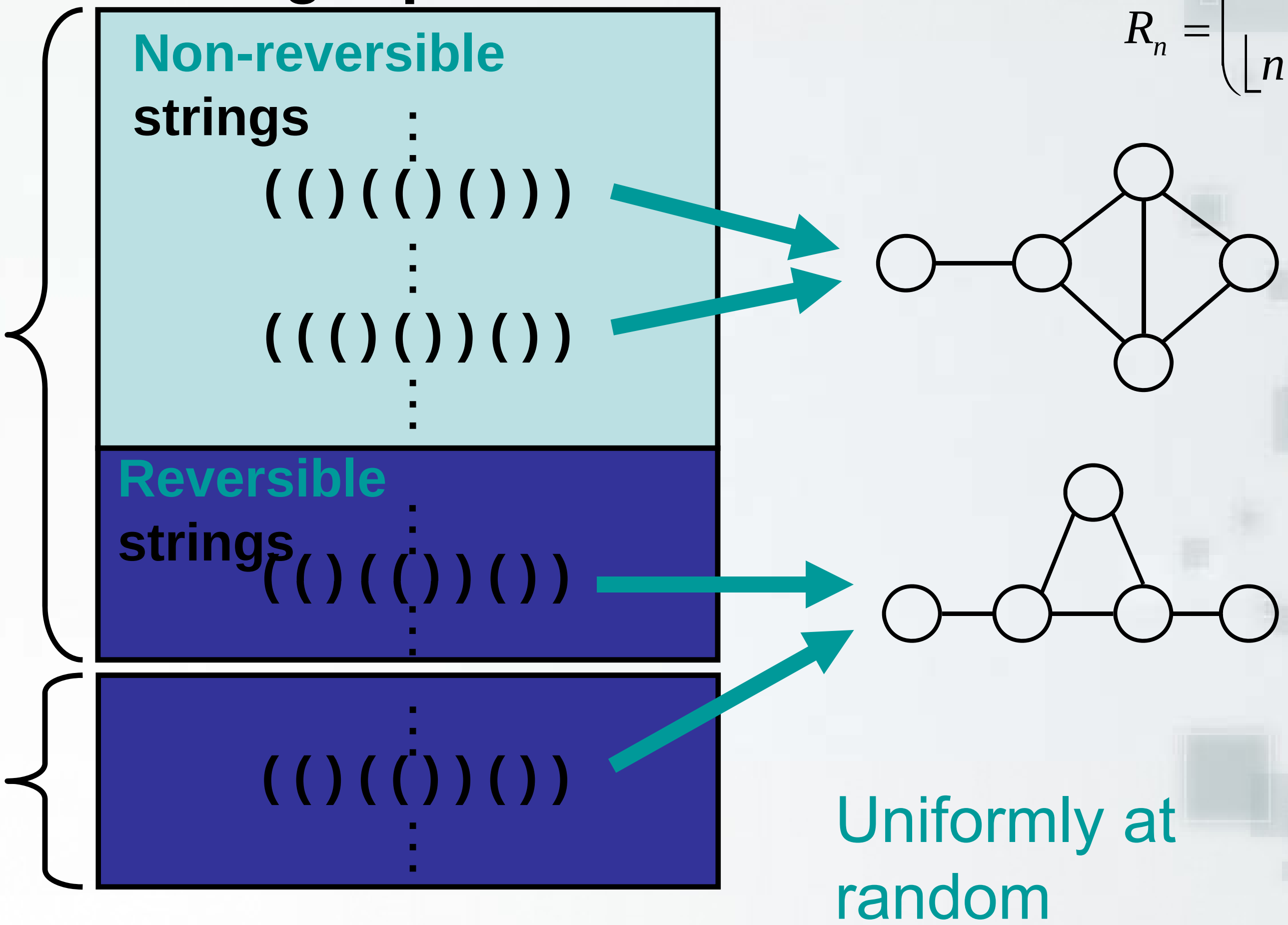
String rep.

Case 1

Prob : $\dfrac{S_n + R_n}{S_n + 2R_n}$

Case 2

Prob : $\dfrac{R_n}{S_n + 2R_n}$



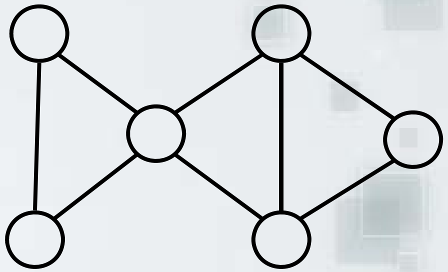
Generalized Catalan Number

$$C(n,i) = \frac{i}{2n+i}\binom{2n+i}{n}$$

Case 1

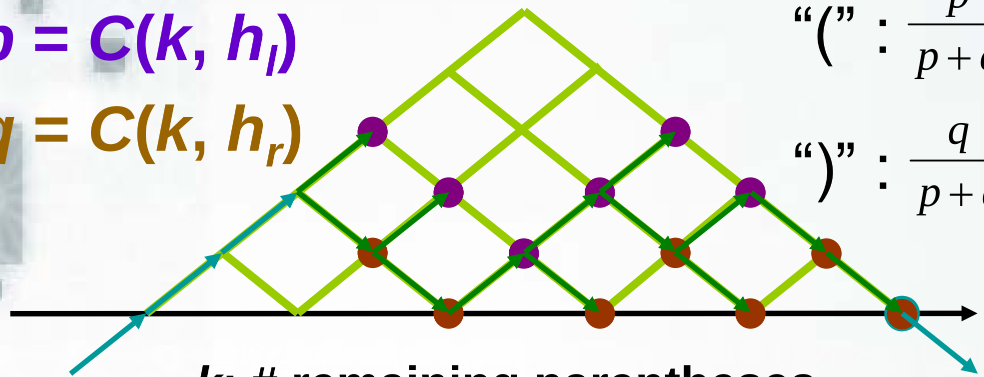
- Generation of a string uniformly at random
  - Generate parentheses from left
    - Select "(" or ")"

(((   )) (( ) ( )))



$p = C(k, h_l)$

$q = C(k, h_r)$



"(" : $\frac{p}{p+q} = \frac{h(k+h+2)}{2k(h+1)}$

")" : $\frac{q}{p+q} = \frac{(k-h)(h+2)}{2k(h+1)}$

k: # remaining parentheses

h: Height

Time complexity

- String Rep. : $O(n)$
- Graph Rep. : $O(n+m)$

m: # edges

Case 2

Generalized Catalan Number

$$C(n,i) = \frac{i}{2n+i}\binom{2n+i}{n}$$

- Generation of reversible string uniformly at random
  - Generate a half of the string from the center to the right end
  1. Choose the height at the center
  2. Generate parentheses from the center
     - Select "(" or ")"

) (())))))))

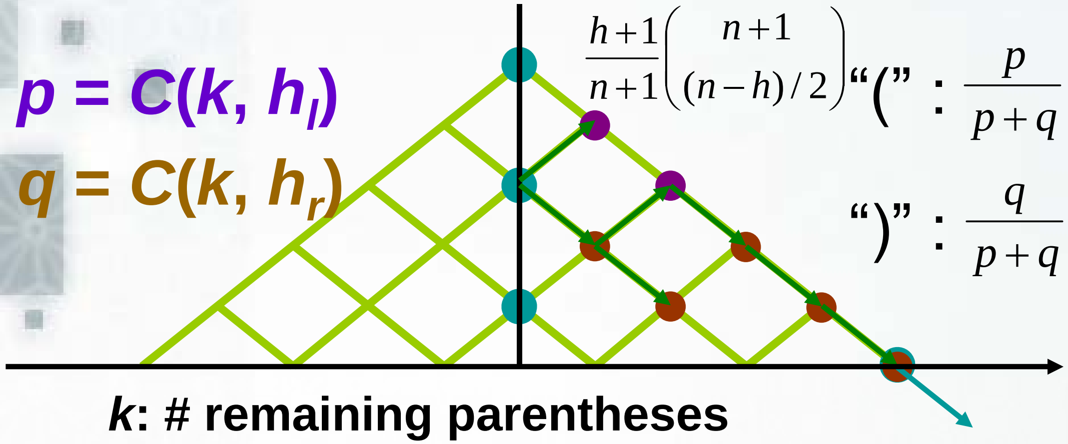
$$\frac{h+1}{n+1}\binom{n+1}{(n-h)/2}$$

"(" : $\frac{p}{p+q} = \frac{h(k+h+2)}{2k(h+1)}$

")" : $\frac{q}{p+q} = \frac{(k-h)(h+2)}{2k(h+1)}$

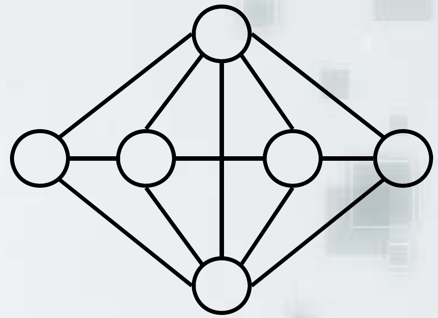
$p = C(k, h_l)$

$q = C(k, h_r)$



k: # remaining parentheses

h: Height



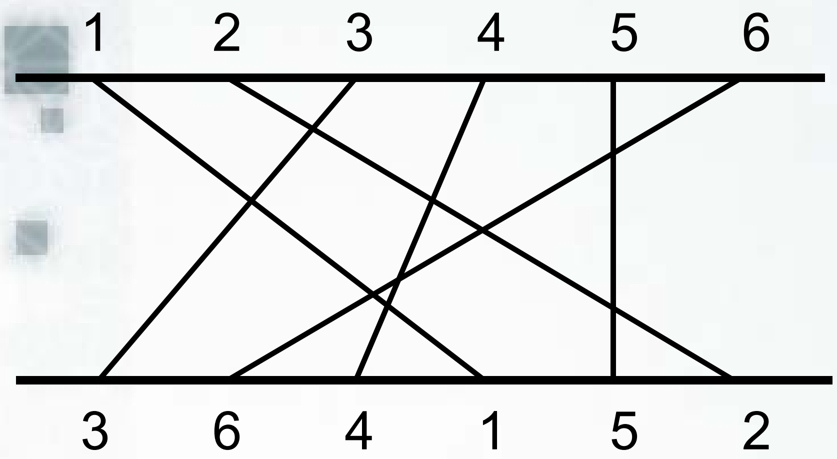
Time complexity

- String Rep. : $O(n)$
- Graph Rep. : $O(n+m)$

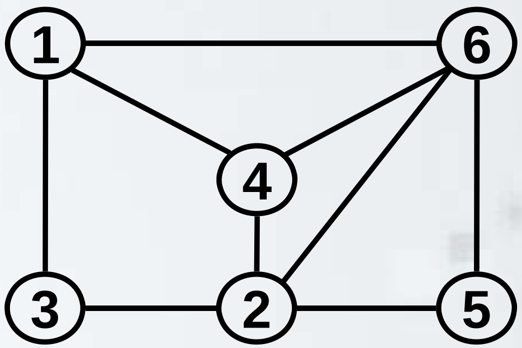
m: # edges

Permutation Graphs

➢ A graph is called a *permutation graph* if the graph has a *line representation*



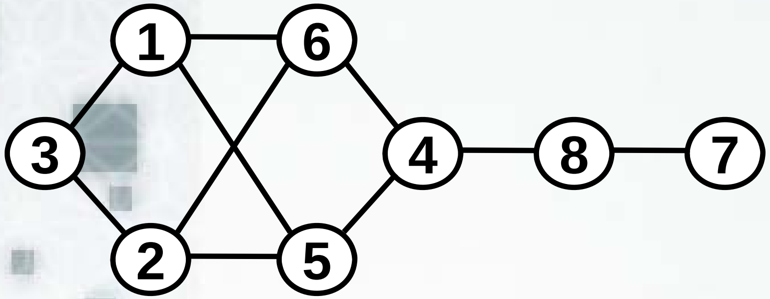
Line representation



Permutation Graph

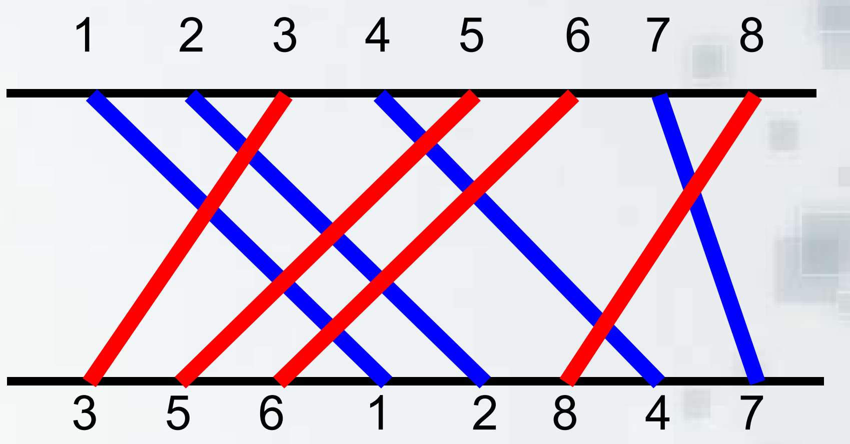
Bipartite Permutation Graphs

- A permutation graph is called a *bipartite permutation graph* if the graph is bipartite



Bipartite permutation graph

(○: vertices in a set X
○: vertices in a set Y)



Line representation

Random generation (and *enumeration*) of bipartite permutation graphs

Useful Property of Connected Bipartite Permutation Graphs

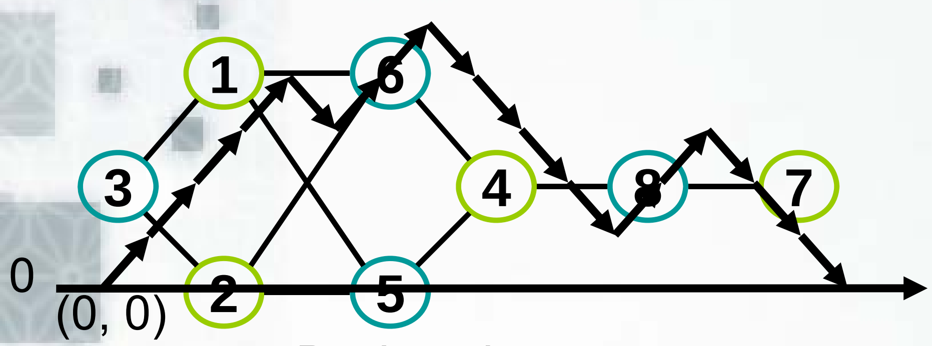
Lemma 1

In a line representation

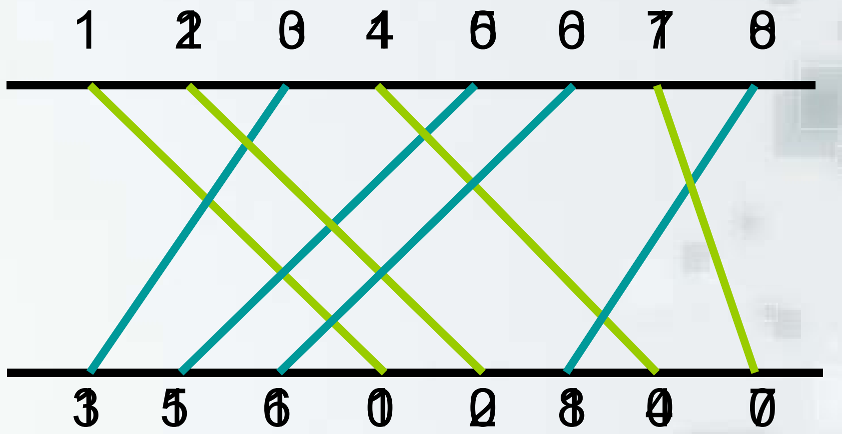
Blue line (corresponds to a vertex in *X*): from upper left to lower right

Red line (corresponds to a vertex in *Y*): from upper right to lower left

Any two lines with the same color have no intersection



Dyck path
Bipartite permutation graph



Line representation

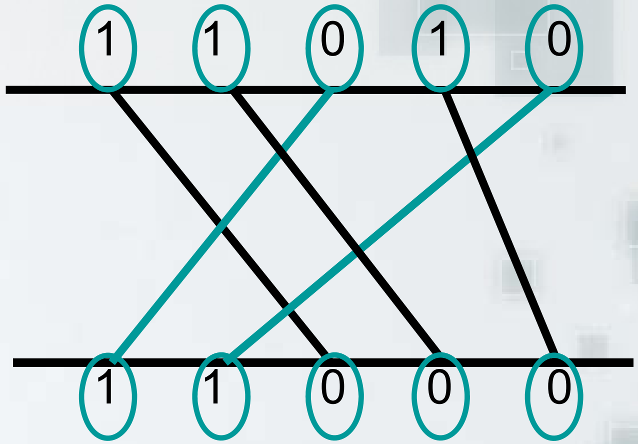
Line representation → 0-1 binary string → Dyck path

How to Construct Dyck Path from 0-1 Binary String

Sweep the two lines alternately

For each character,

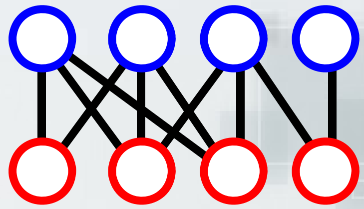
- '1' ⇒ go right and *go up*
- '0' ⇒ go right and *go down*



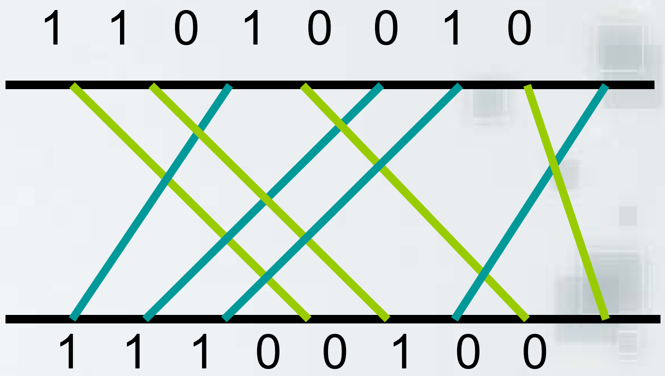
Connected Bipartite Permutation Graph with *n* Vertices

Property of Dyck path

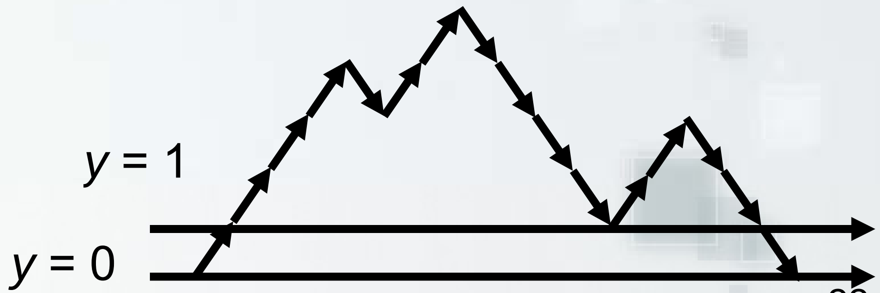
- The last coordinate is (2*n*, 0)
  - The upper / lower lines have *n* '1's and *n* '0's
- Located on *the upper side* of *x*-axis
- For all points but (0,0) and (2*n*,0), *its value of y-coordinate* is equal or greater than 1



A graph

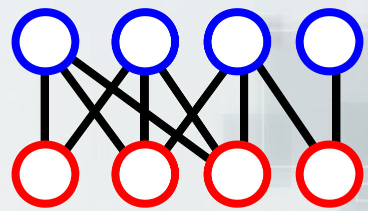


line rep.

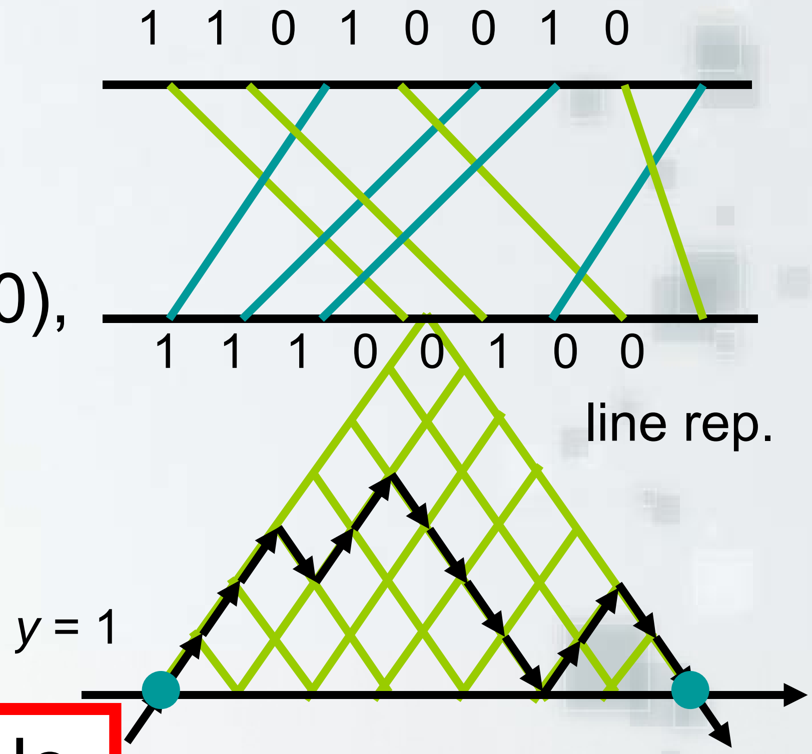


Dyck path

Connected Bipartite Permutation Graph with *n* Vertices



A graph



line rep.

Dyck path

Property of Dyck path

- The last coordinate is (2*n*, 0)
  - The upper / lower lines have *n* '1's and *n* '0's
- Located on *the upper side* of *x*-axis
- For all points but (0,0) and (2*n*,0), *its value of y-coordinate* is equal or greater than 1

For random generation, it is sufficient to *generate a Dyck path* randomly?

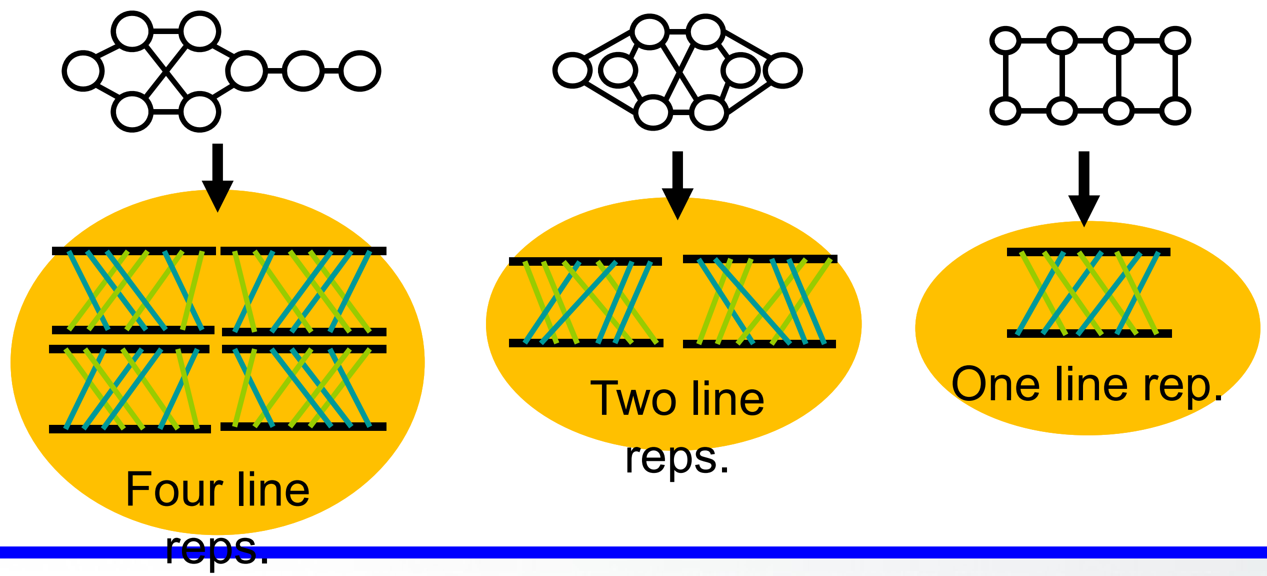
No

The Reason of "No"

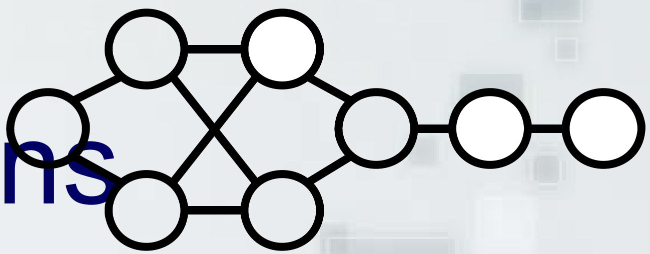
- There is *no* 1 to 1 correspondence between connected bipartite permutation graphs and their line representations

A graph corresponds to *at most four* line representations

Examples

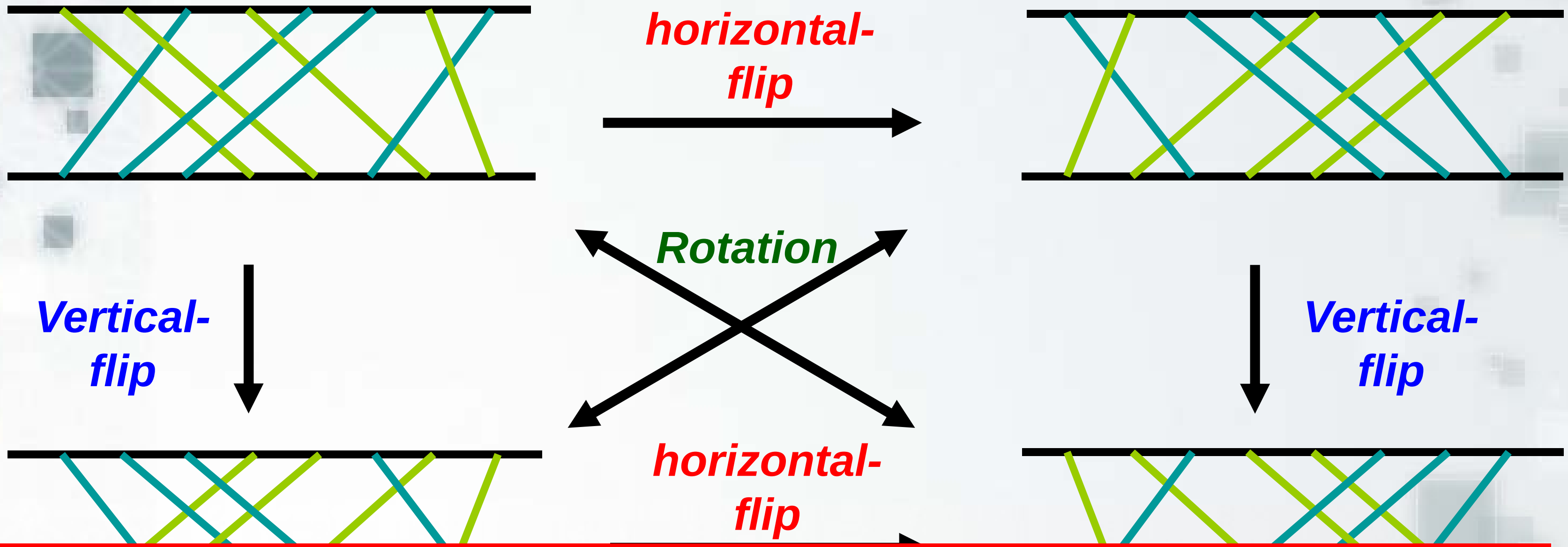


Equivalent Line Representations



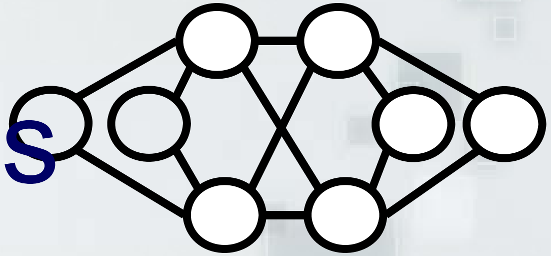
Lemma 2

There exists *at most four* line representations for any connected bipartite permutation graph



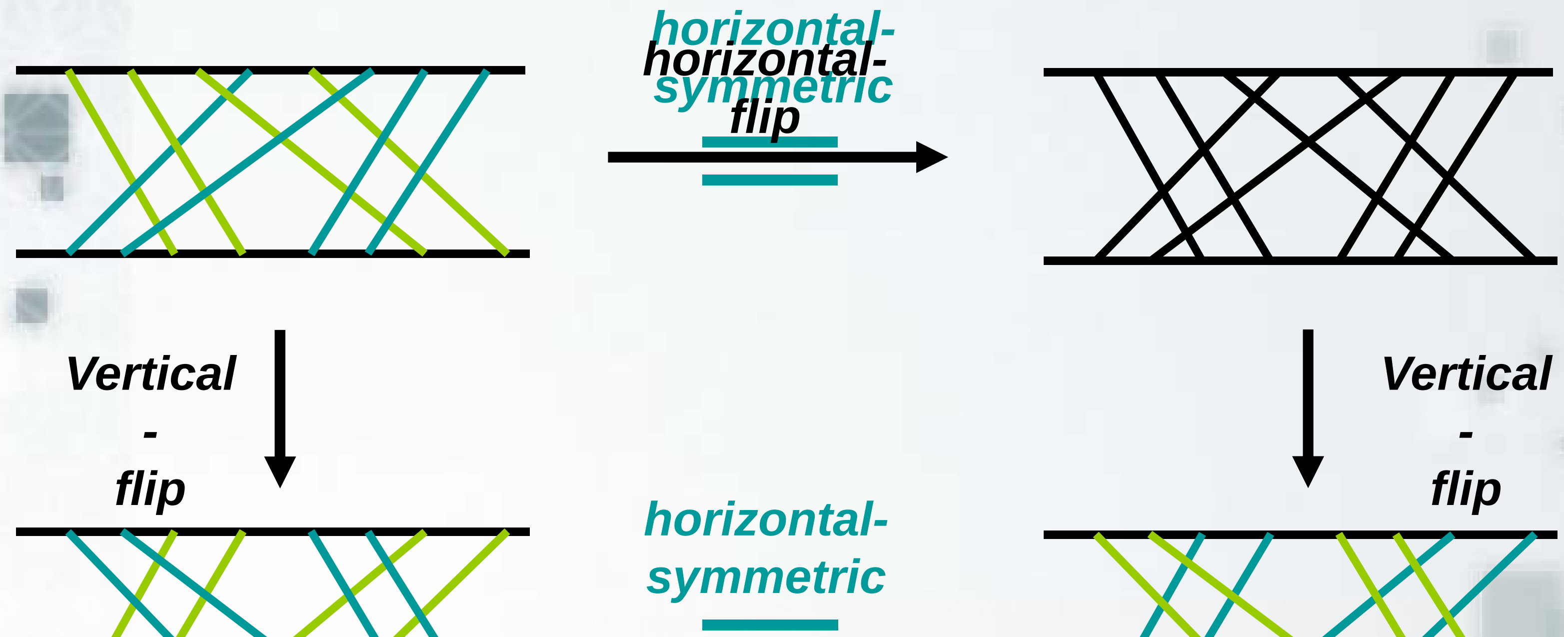
This graph corresponds to *four* line representations

Equivalent Line Representations



Lemma 2

There exists *at most four* line representations for any connected bipartite permutation graph



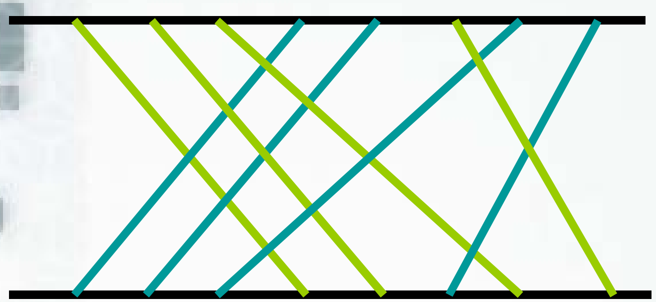
This graph corresponds to *two* line representations

Equivalent Line Representations

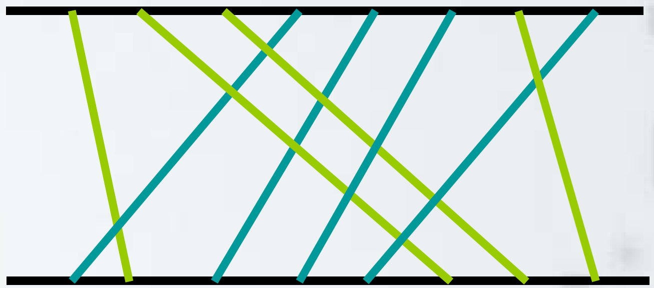
Lemma 2

There exists *at most four* line representations for any connected bipartite permutation graph

Other examples:



Vertical-symmetric
line representation



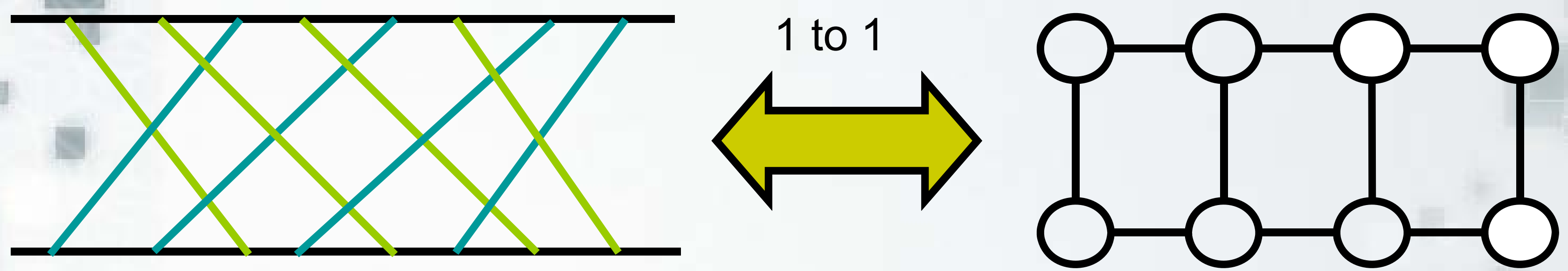
Rotational-symmetric
line representation

Corresponding graphs have *two* line representations resp.

Equivalent Line Representations

Lemma 2

There exists *at most four* line representations for any connected bipartite permutation graph

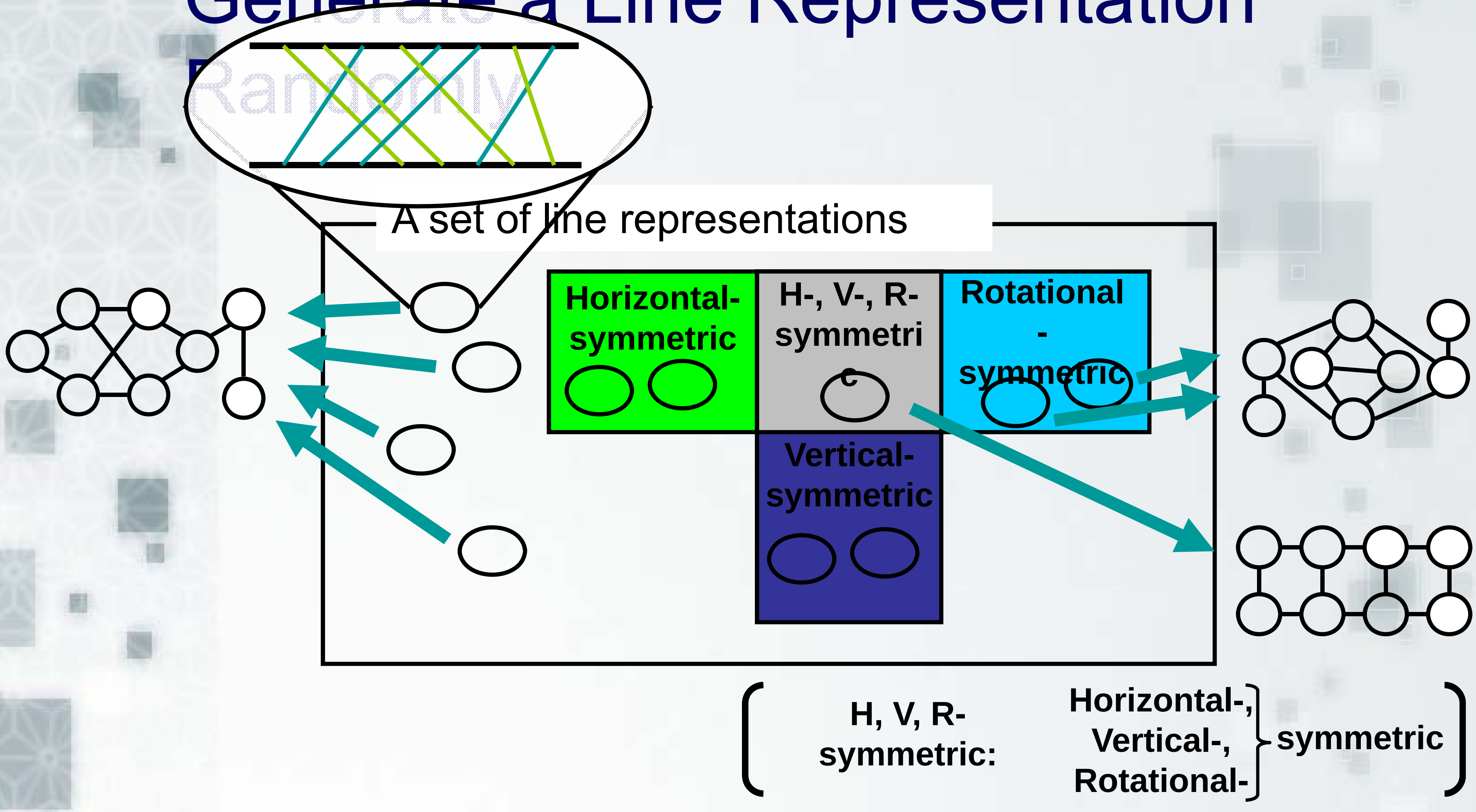


Horizontal, vertical, rotational-symmetric line representation

Bipartite Permutation Graph

This graph have *one* line representation

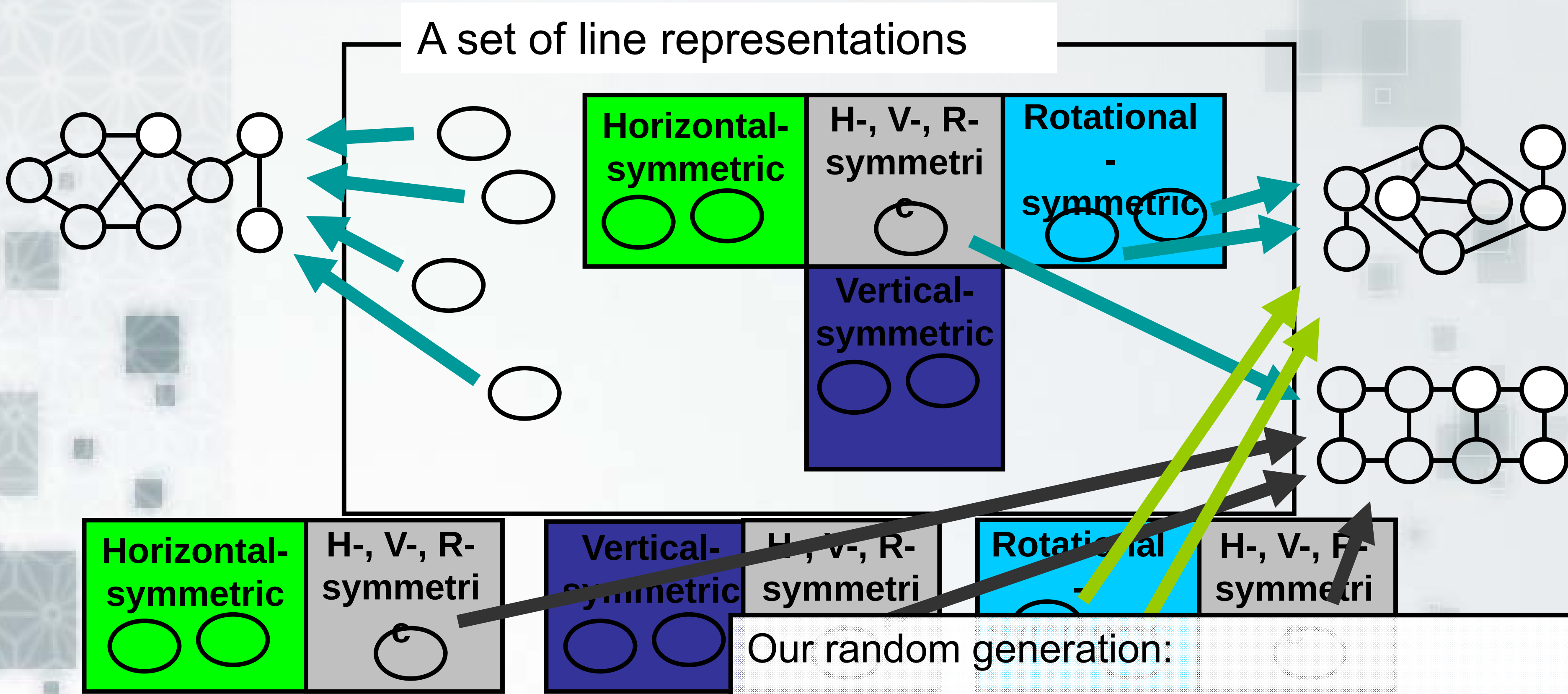
Generate a Line Representation Randomly



If we generate a line representation randomly,

A graph corresponding to *four* line reps. is *frequently* generated

Our Approach: Normalization of Probability



Any graph corresponds to the *four* line representations

Our random generation:

1. Choose one among 4 groups
2. Randomly generate the chosen one

Conclusions and Future Works

Conclusions

For {prop. interval|bipartite permutation} graphs, we have designed the algorithms:

- Random generation algorithms: O(n+m) *time*
- Enumeration algorithms: O(1) *time* / graph

Future works

- Random generation and enumeration of
  - *interval graphs* and *permutation graphs*
  - *graph classes* such that the GI problem is poly-time solvable?