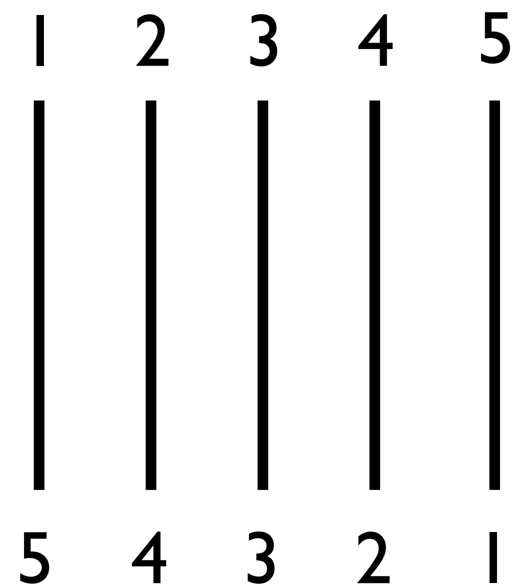
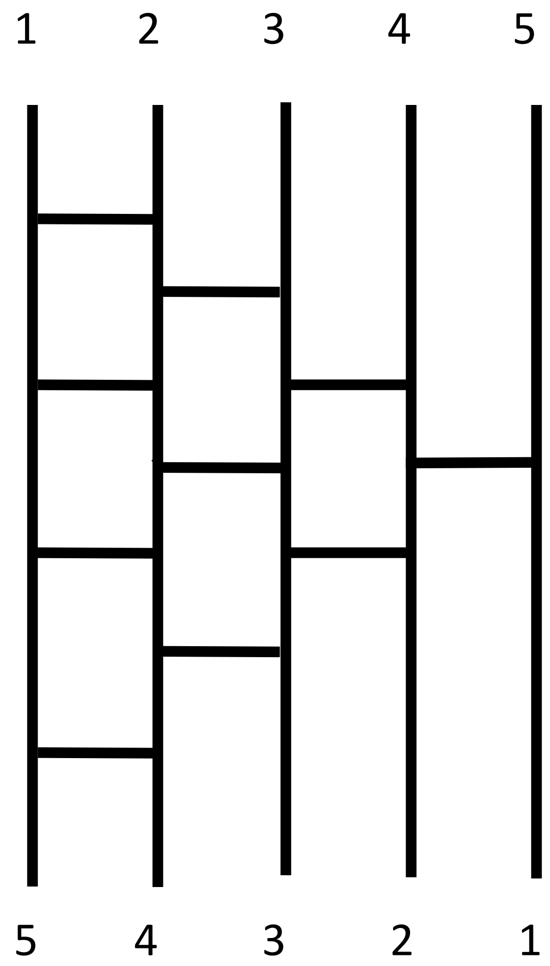
πDDを用いた プリミティブソーティングネットワークの 数え上げ

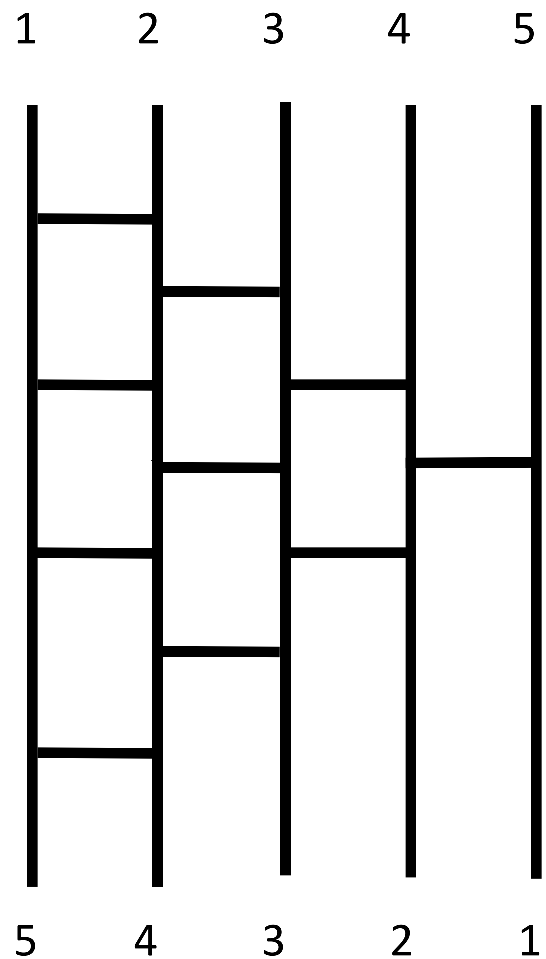
川原 純　斎藤 寿樹　吉仲 亮　湊 真一

問題

横線を引いて
あみだを完成させよ

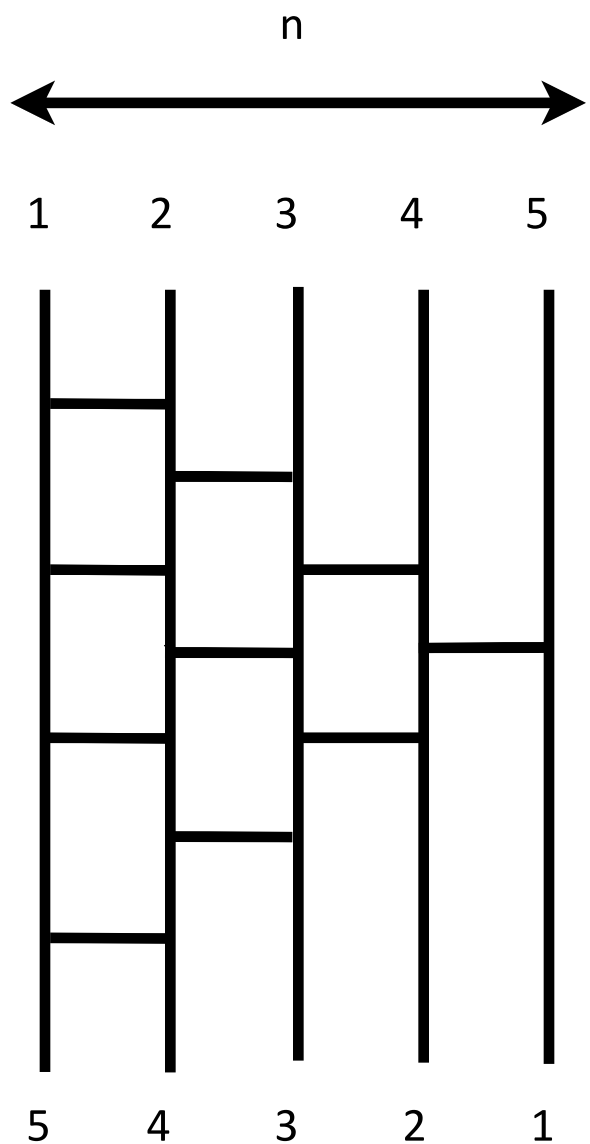



1 2 3 4 5
5 4 3 2 1



疑問

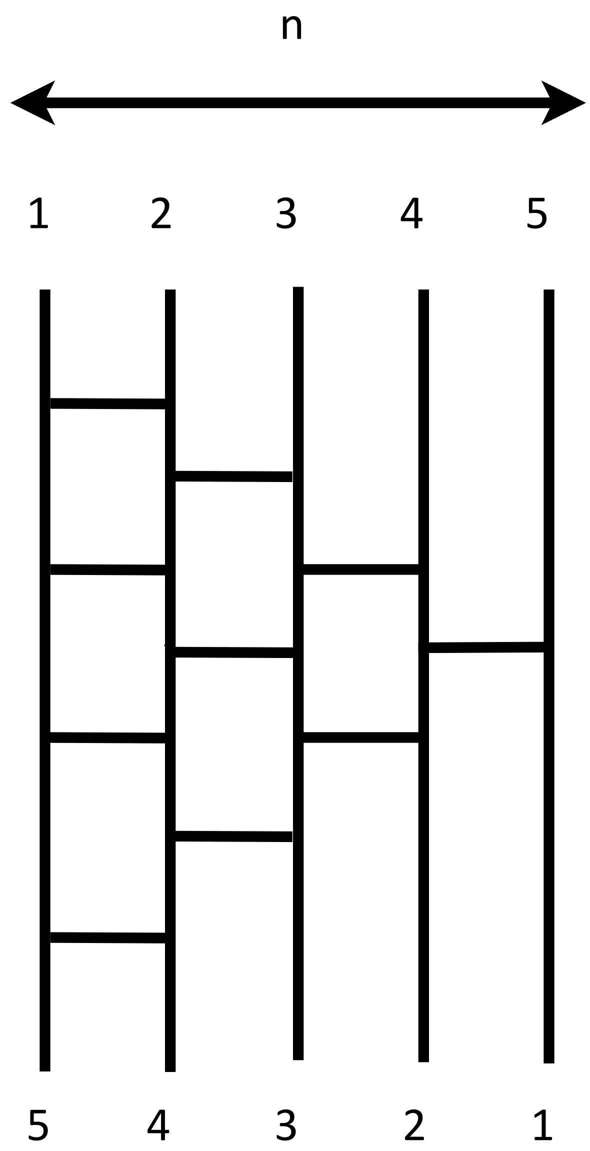
最小本数は？



疑問

最小本数は？

$n(n-1)/2$

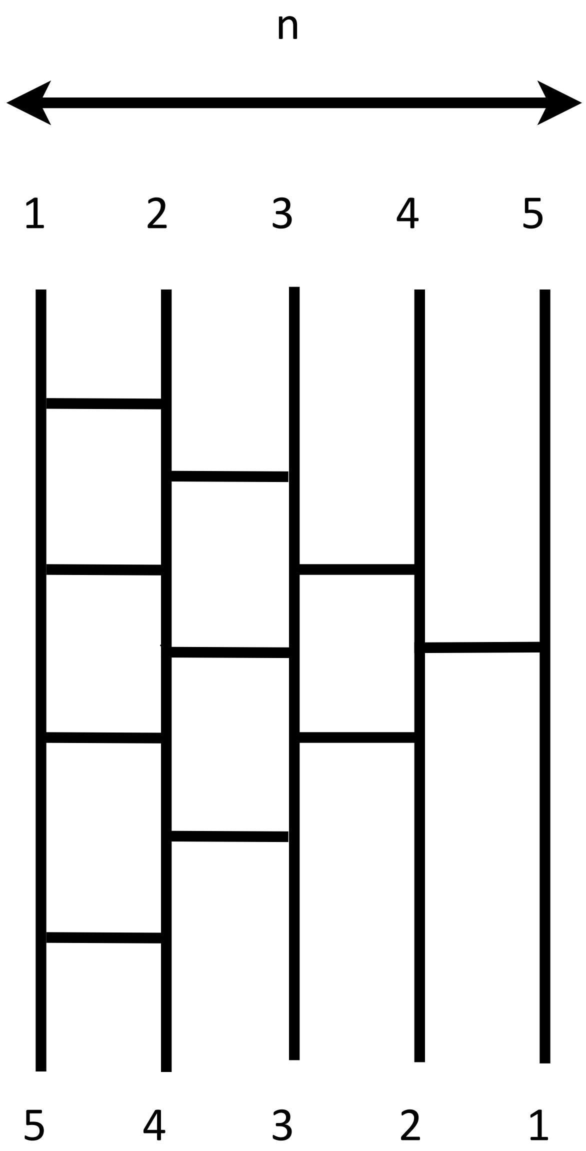


疑問

最小本数は？

n(n-1)/2

引き方は何通り？



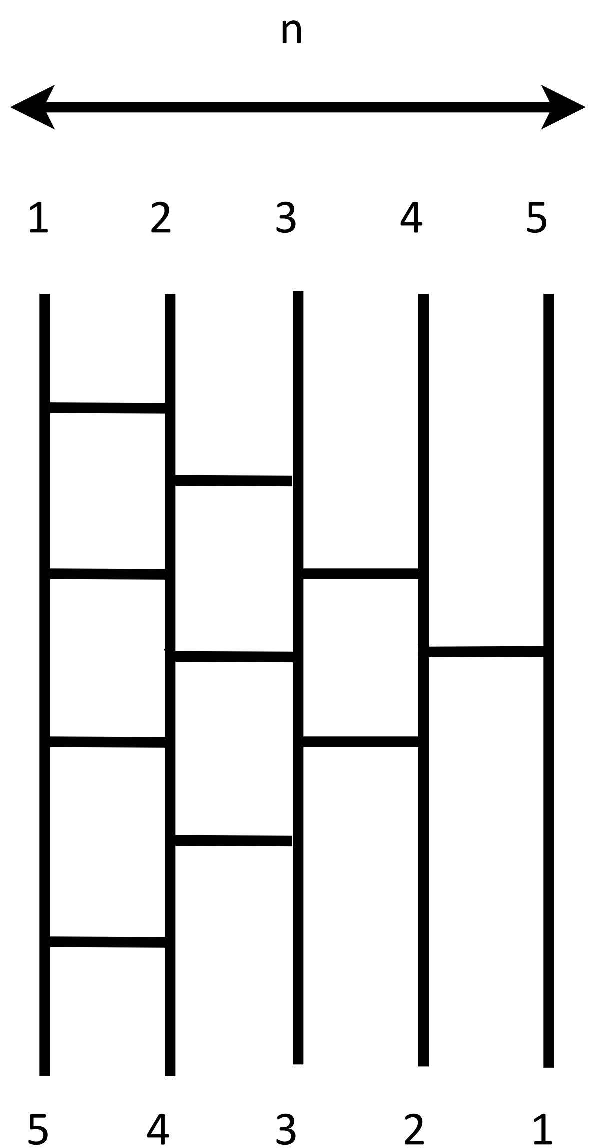
疑問

最小本数は？

n(n-1)/2

引き方は何通り？

たくさん



疑問

最小本数は？

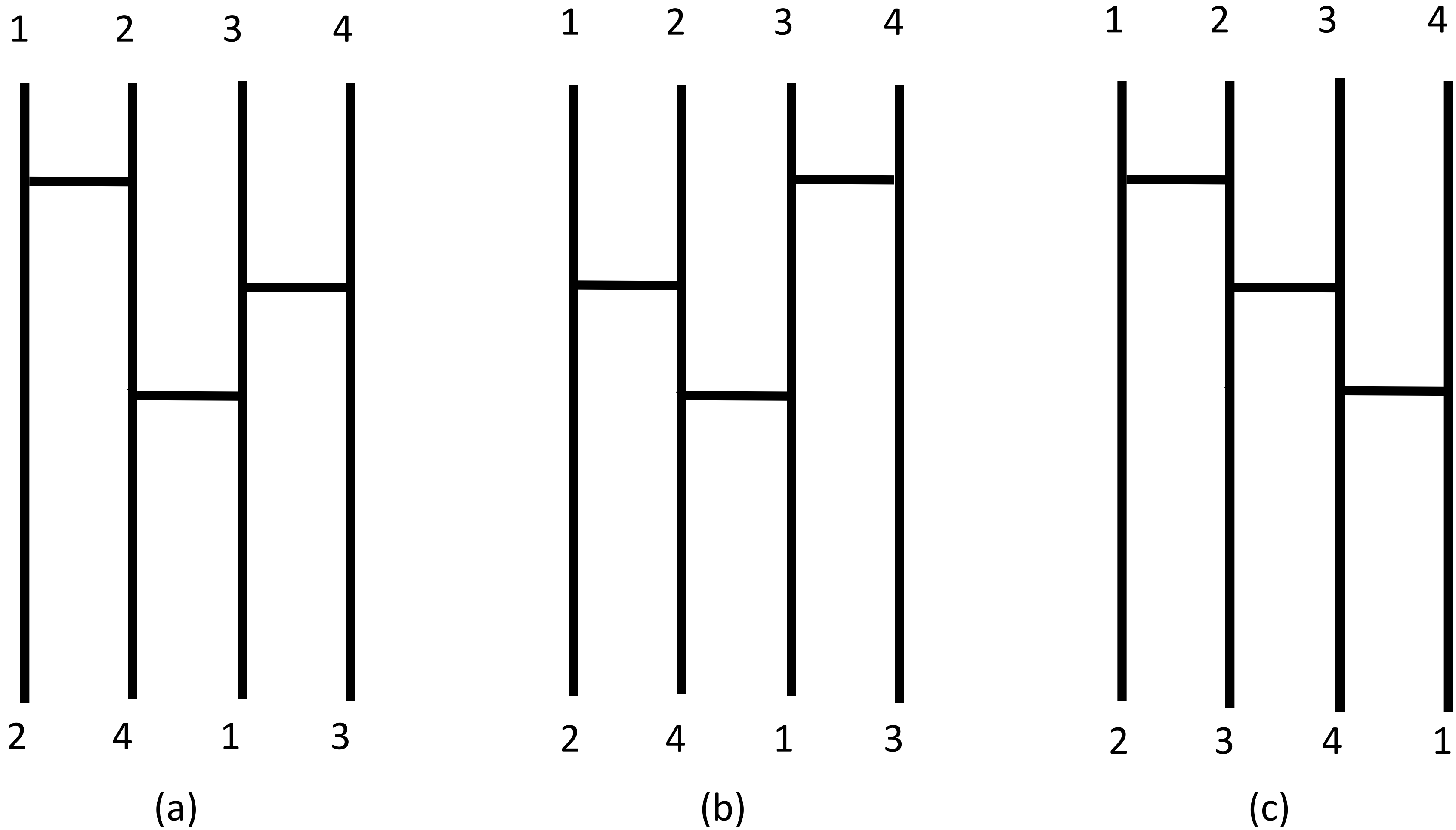
n(n-1)/2

引き方は何通り？

たくさん

n(n-1)/2 本使った

引き方は何通り？



(a) と (b) は同じあみだ、

(a) と (c) は異なるあみだとみなす

(a) を**標準形あみだ**という

過去の結果と今回の結果

n	bar	あみだの数	備考
2	1	1	
3	3	2	
4	6	8	
5	10	62	
6	15	908	
7	21	24698	
8	28	1232944	
9	36	112018190	Knuth 1992の文献に掲載
10	45	18410581880	
11	55	5449192389984	[Yamanaka et al. 2009]
12	66	2894710651370540	[Matthew 2011]
13	78	2752596959306389652	今回

OEIS (On-Line Encyclopedia of Integer Sequences) に掲載

http://oeis.org/A006245/　本発表の論文もここから取れます

なぜ あみだくじ？

- 研究背景
  - あみだくじの数え上げにπDDを用いることを思いついた
  - 実装してみたら世界タイ記録の結果が得られた

アルゴリズム的に改良して、世界新記録

ソースコードは（πDDライブラリを除いて）わずか150行ほど

πDDの代数演算を直接記述

πDD の威力を示す例

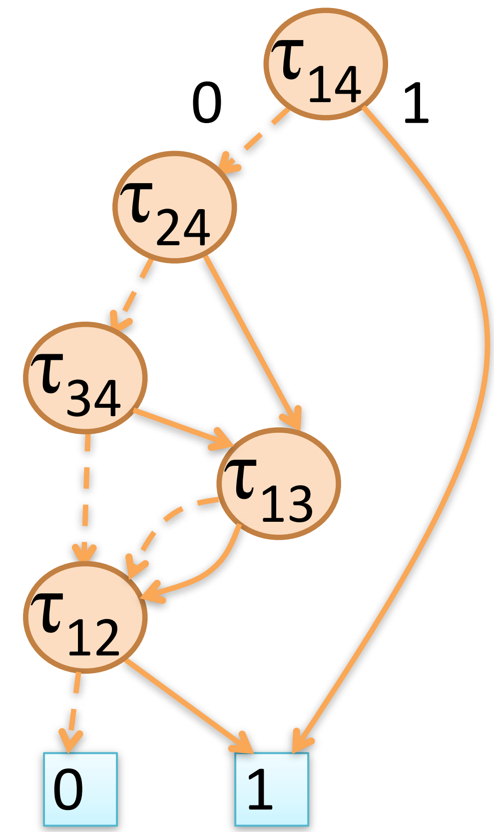
πDDとは？

置換の集合を表すデータ構造

$$P = \{(4, 2, 3, 1), (4, 1, 3, 2), (4, 3, 1, 2), (2, 4, 1, 3), (2, 1, 4, 3), (2, 1, 3, 4)\}$$

$$= \{ \tau_{14}, \tau_{12}\tau_{24}, \tau_{12}\tau_{13}\tau_{24}, \tau_{12}\tau_{13}\tau_{34}, \tau_{12}\tau_{34}, \tau_{12} \}$$

πDD



置換の表し方

(4, 2, 1, 3)

$$\begin{array}{cccc} 1 & 2 & 3 & 4 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 4 & 2 & 1 & 3 \end{array}$$

互換

2つの数字だけを交換する置換

(1, 2, 6, 4, 5, 3, 7)

$$\begin{array}{cc} 3 & 6 \\ \downarrow & \downarrow \\ 6 & 3 \end{array}$$

τ_{36} と表記

置換の積

(1, 3, 2)× (3, 1, 2) = (3, 2, 1)

$$\begin{array}{ccc} 1 & 2 & 3 \\ \downarrow & \downarrow & \downarrow \\ 1 & 3 & 2 \\ \downarrow & \downarrow & \downarrow \\ 3 & 2 & 1 \end{array}$$

(1, 3, 2)

(3, 1, 2)

置換と互換

任意の置換は互換の積で表される

$$(3, 4, 2, 1) = \tau_{23}\tau_{13}\tau_{14}$$
$$= \tau_{12}\tau_{13}\tau_{23}\tau_{14}\tau_{24}$$

一意ではない

以下の形への分解は一意

$$(3, 4, 2, 1) = \tau_{12}\tau_{23}\tau_{14}$$

$<\quad<$

各 τ_{xy} : $x < y$

標準分解と呼ぶ

$$(3, 4, 2, 1) = (3, 1, 2, 4)\ \tau_{14}$$
$$= (2, 1, 3, 4)\ \tau_{23}\tau_{14}$$
$$= \tau_{12}\tau_{23}\tau_{14}$$

置換の集合

P = {(4, 2, 3, 1), (4, 1, 3, 2), (4, 3, 1, 2),
(2, 4, 1, 3), (2, 1, 4, 3), (2, 1, 3, 4)}

を効率の良いデータ構造で記憶したい

各要素を標準分解する

$P = \{ \tau_{14}, \tau_{12}\tau_{24}, \tau_{12}\tau_{13}\tau_{24},$
$\tau_{12}\tau_{13}\tau_{34}, \tau_{12}\tau_{34}, \tau_{12} \}$

τ の集合の集合と考えて、ZDDで表す

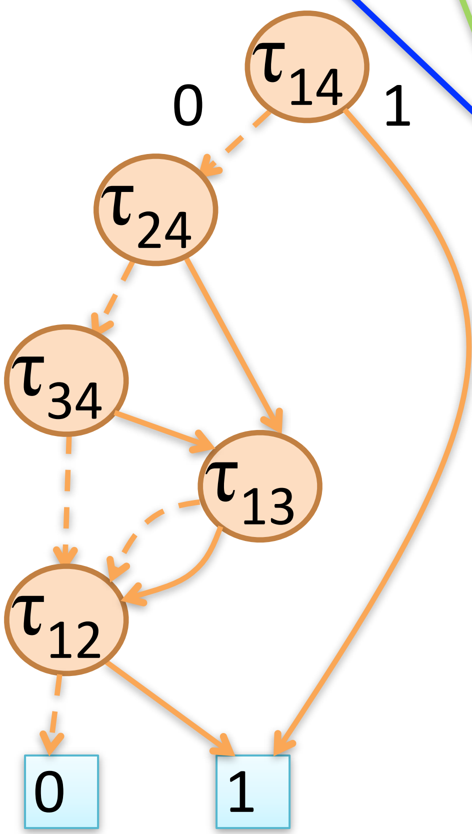
$P = \{(4, 2, 3, 1), (4, 1, 3, 2), (4, 3, 1, 2),$
$(2, 4, 1, 3), (2, 1, 4, 3), (2, 1, 3, 4)\}$

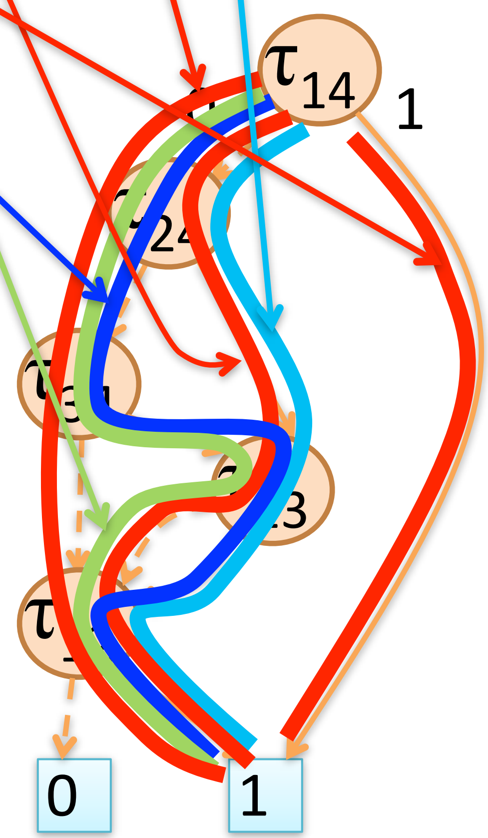
$= \{ \tau_{14}, \tau_{12}\tau_{24}, \tau_{12}\tau_{13}\tau_{24}, \tau_{12}\tau_{13}\tau_{34}, \tau_{12}\tau_{34}, \tau_{12} \}$

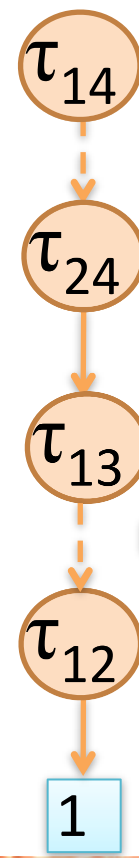
$\boxed{1}$ へのパス1本が

1つの要素に対応

1枝 τ を通るとき、τを掛ける







$\Rightarrow \tau_{24} \times \tau_{12}$

πDD の特徴

- ZDD と同じく、集合演算が効率良く実行可能

P, Q を πDD （置換の集合）とする

$$P \cup Q = \{ \pi \mid \pi \in P \text{ or } \pi \in Q \}$$

$$P \cap Q = \{ \pi \mid \pi \in P \text{ and } \pi \in Q \}$$

$$P \setminus Q = \{ \pi \mid \pi \in P \text{ and } \pi \notin Q \}$$

$$P * \tau = \{ \pi\tau \mid \pi \in P \}$$

$$P * Q = \{ \pi\pi' \mid \pi \in P \text{ and } \pi' \in Q \}$$

等の演算ができる

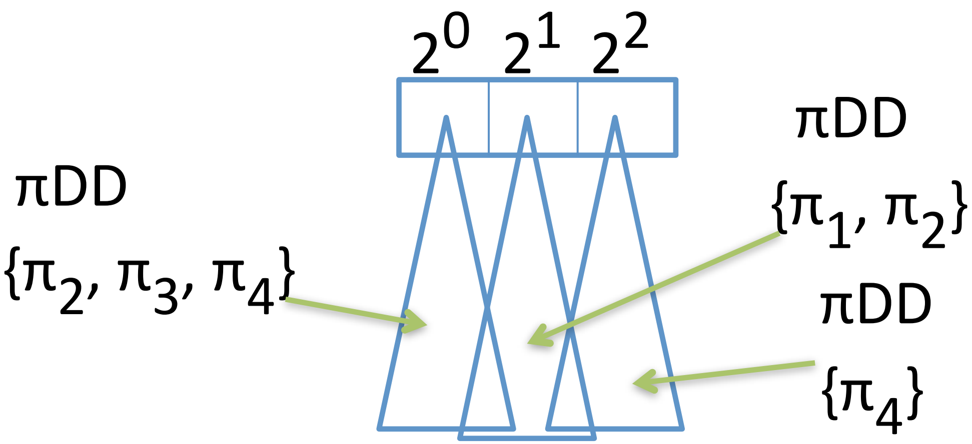
置換の多重集合

- 置換の個数の情報も保持する（多重集合）

P = { 2 (4, 2, 3, 1), 3 (4, 1, 3, 2), 1 (4, 3, 1, 2), 5 (1, 2, 4, 3) }

= { 2 π_1, 3 π_2, 1 π_3, 5 π_4 }

以下のように保持する



各 2^i のマスは πDD を1つもつ

π_j が含まれるすべてのマスについて 2^i を足した値が π_j の個数

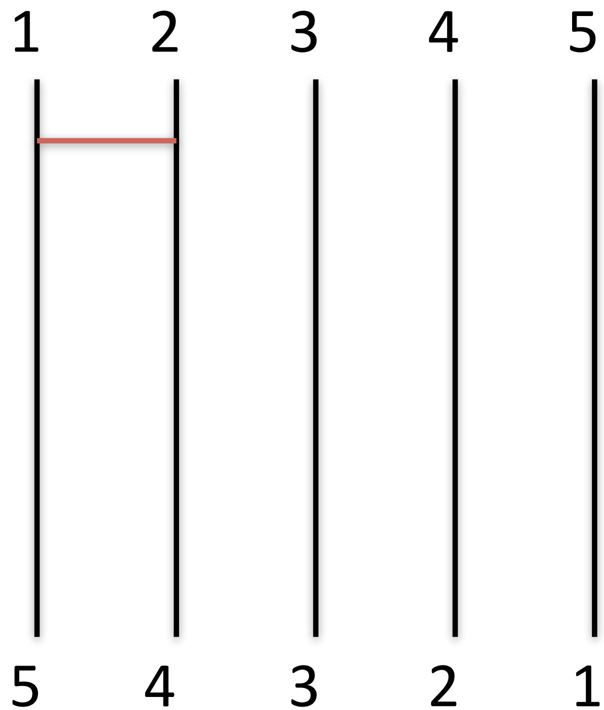
置換の多重集合の演算

$P = \{ 2\,\pi_1, 3\,\pi_2, 1\,\pi_3, \quad 5\,\pi_4 \}$

$Q = \{ 6\,\pi_1, 2\,\pi_2, \qquad 9\,\pi_4, \ 7\,\pi_5 \}$

$P + Q = \{ 8\,\pi_1, 5\,\pi_2, 1\,\pi_3, 14\,\pi_4, \ 7\,\pi_5 \}$

他にもいろいろ考えられるが、省略



1本の線で実現できる置換は

τ_{12}　τ_{23}　τ_{34}　τ_{45}

2本の線で実現できる置換は

$\tau_{12}\tau_{12}$　$\tau_{12}\tau_{23}$　$\tau_{12}\tau_{34}$　$\tau_{12}\tau_{45}$

$\tau_{23}\tau_{12}$　…

（標準形を考慮しない場合）

i – 1 本の線で実現できる置換の集合を P とすると、

i 本の線で実現できる置換は

$$P * \tau_{12} + P * \tau_{23} + P * \tau_{34} + P * \tau_{45}$$

$$P * \tau = \{ \pi\tau \mid \pi \in P \}$$

アルゴリズム

置換の多重集合　$P_1, P_2, \ldots, P_{n-1}$　（P_i は最後に引かれた線が (i, i+1) である置換の多重集合）

$P_i := \{\tau_{i,\, i+1}\}$　　$(i = 1, \ldots, n-1)$　　（線が1本引かれた状態からスタート）

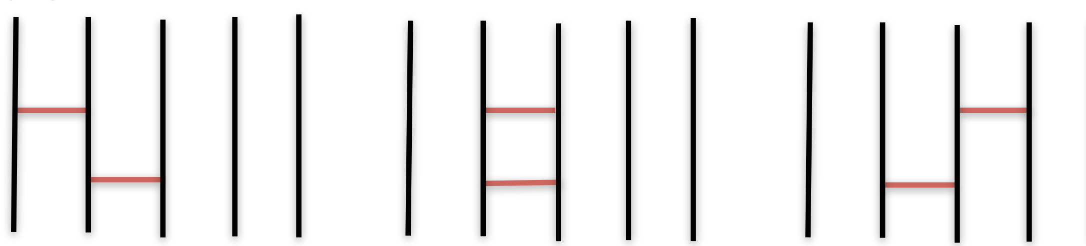
以下を $n(n-1)/2 - 1$ 回繰り返す

for (i = 1 to n – 1) do

$P_i^{New} := (P_1 + \cdots + P_i + P_{i+1}) * \tau_{i,\, i+1}$　　（$P_n = \phi$ とする）

for (i = 1 to n – 1) do

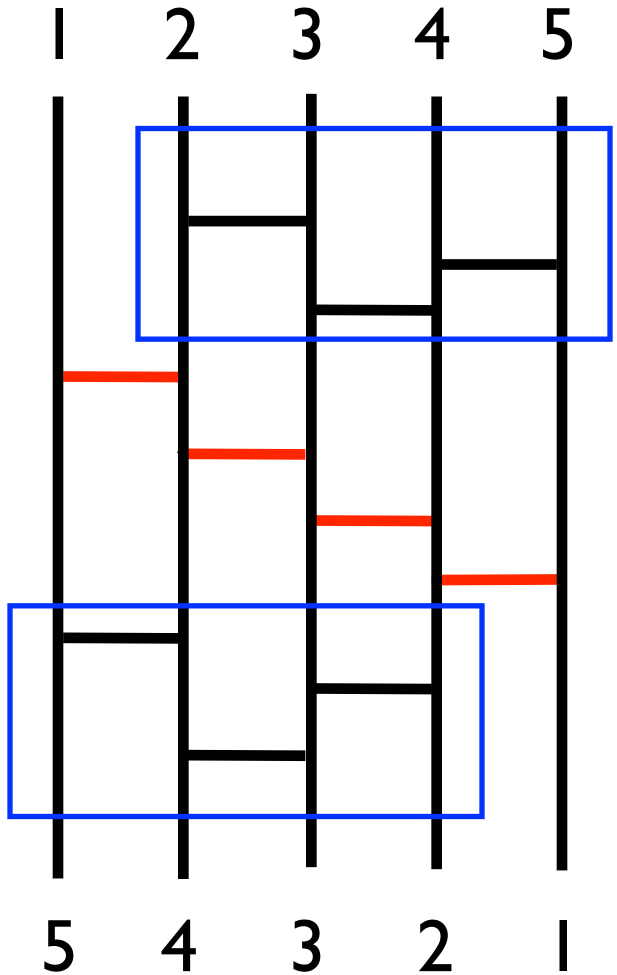
$P_i := P_i^{New}$



$P = P_1 + \cdots + P_{n-1}$

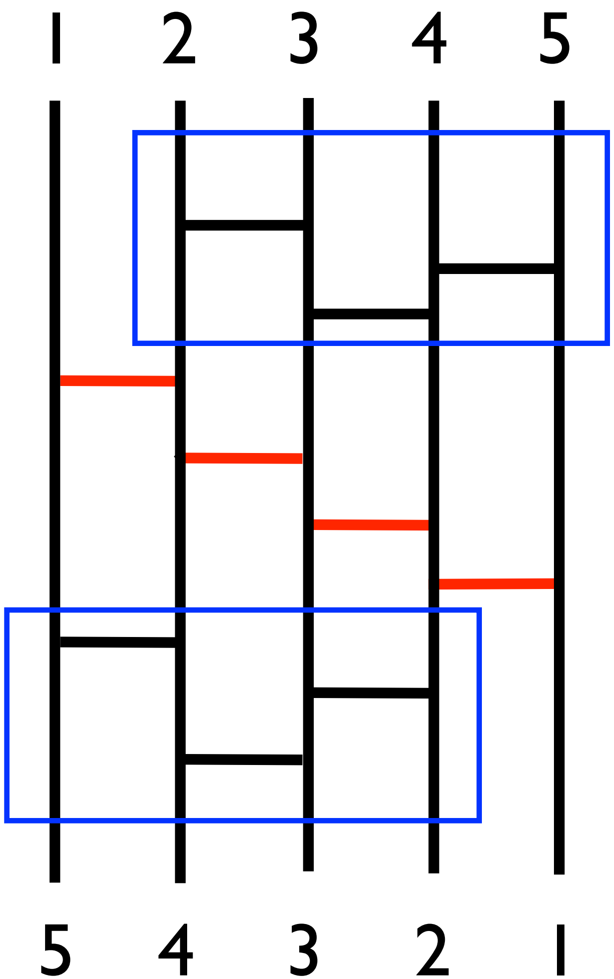
P の中の (n, n – 1,..., 1) の数が、求めたいものになる

改良アルゴリズム



横棒数最小の
逆順あみだは
左のような
構造をもつ

改良アルゴリズム



$\boxed{\tau_2\ \tau_4\ \tau_3}\ \boxed{\tau_1\ \tau_2\ \tau_3\ \tau_4}\ \boxed{\tau_1\ \tau_3\ \tau_2}$

標準形あみだ　　標準形あみだ

改良アルゴリズム

$n := n - 1$

置換の多重集合 $P_1, \ldots, P_{n-1}, Q_1, \ldots, Q_{n-1}$

$P_i := \{\tau_{i,\,i+1}\}$　　$(i = 1, \ldots, n-1)$

以下を $n(n-1)/2 - 1$ 回繰り返す

for $(i = 1$ to $n - 1)$ do

$$P_i^{New} := (P_1 + \cdots + P_i + P_{i+1}) * \tau_{i,\,i+1}$$

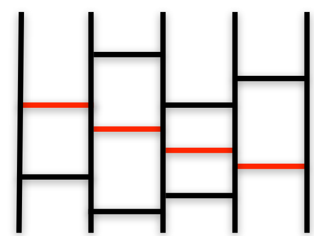
$$Q_i^{New} := (P_1 + \cdots + P_{n-1} + Q_1 + \cdots + Q_i + Q_{i+1}) * \tau_{i,\,i+1}$$

for $(i = 1$ to $n - 1)$ do

$$P_i := P_i^{New} \qquad Q_i := Q_i^{New}$$

（P_i は最後に引かれた線が $(i, i+1)$ である置換の多重集合 対角barを超える前）

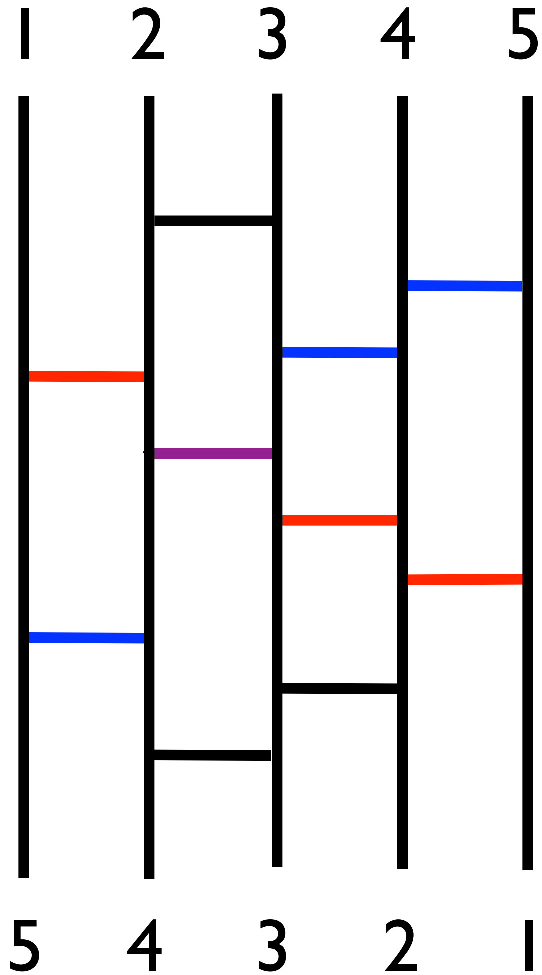
（Q_i は最後に引かれた線が $(i, i+1)$ である置換の多重集合 対角barを超えた後）



$$P = P_1 + \cdots + P_{n-1} + Q_1 + \cdots + Q_{n-1}$$

P の中の $(n, n-1, \ldots, 1)$ の数が、求めたいものになる

さらに改良アルゴリズム



横棒数最小の
逆順あみだは
左のような
構造をもつ

実験結果

n	bar	あみだの数	ALG1 time	ALG2 time	ALG3 time
2	1	1	0.0	0.0	0.0
3	3	2	0.0	0.0	0.0
4	6	8	0.0	0.0	0.0
5	10	62	0.0	0.0	0.0
6	15	908	0.0	0.0	0.0
7	21	24698	0.0	0.1	0.0
8	28	1232944	1.0	0.3	0.2
9	36	112018190	14.6	4.1	3.1
10	45	18410581880	232.1	62.0	48.29
11	55	5449192389984	4369.6	964.1	800.6
12	66	2894710651370540	109387.0	20172.6	18316.4
13	78	2752596959306389652	--	473314.0	443239.0

sec.

まとめ

πDDを用いたシンプルなアルゴリズムALG1

→ 150行くらいのソースコードで世界タイ記録

あみだの構造を利用したアルゴリズムALG2, ALG3

→ n=13 世界新記録

今後の課題

多重数え上げのアルゴリズムとして汎用化