Kolmogorov記述量に基づく類似度を用いたMIDIファイルの自動分類

谷研究室　　福森　勤

目次

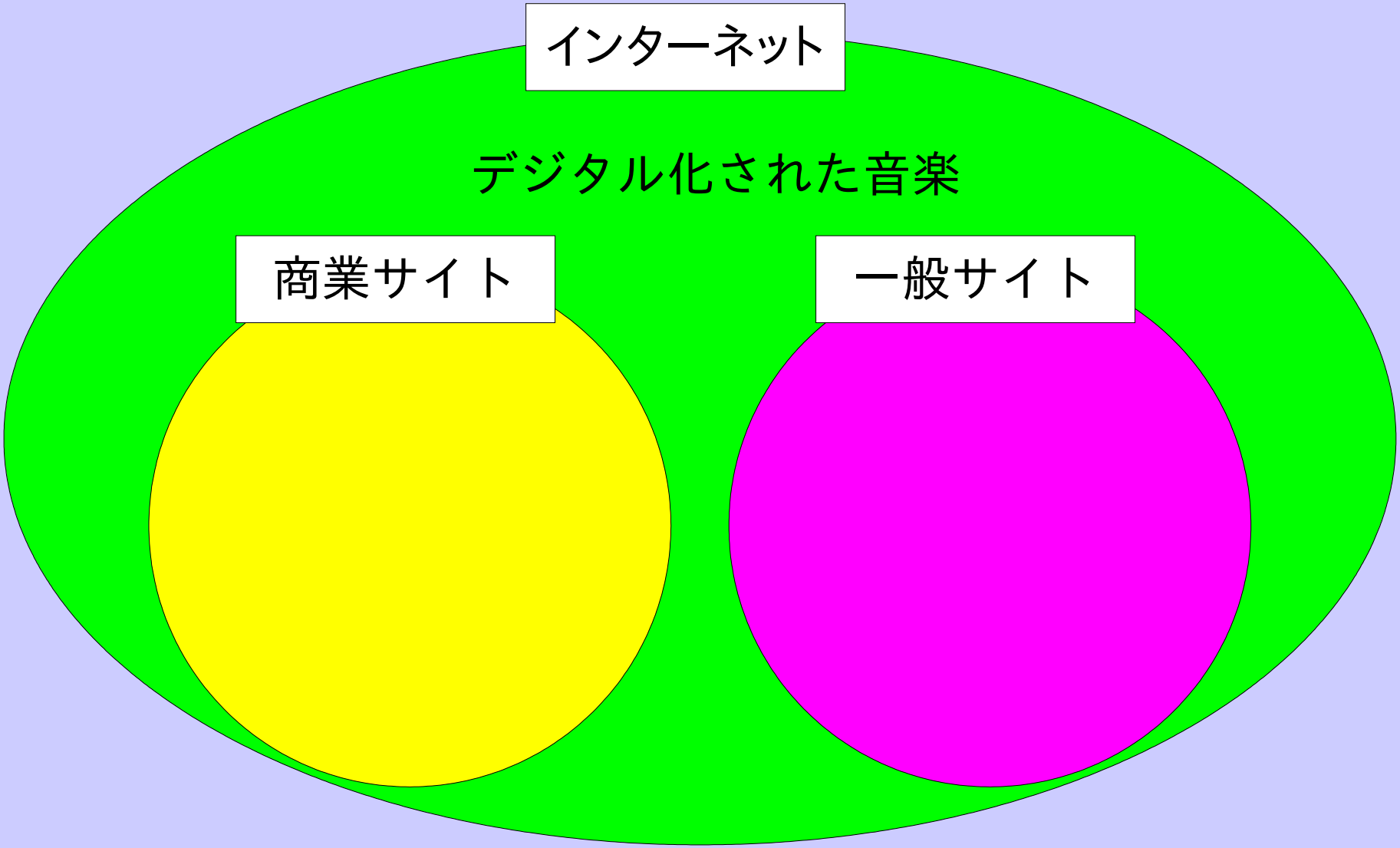
- 研究の概要
- 実験方法
- 実験結果
- 考察
- 今後の課題

目次

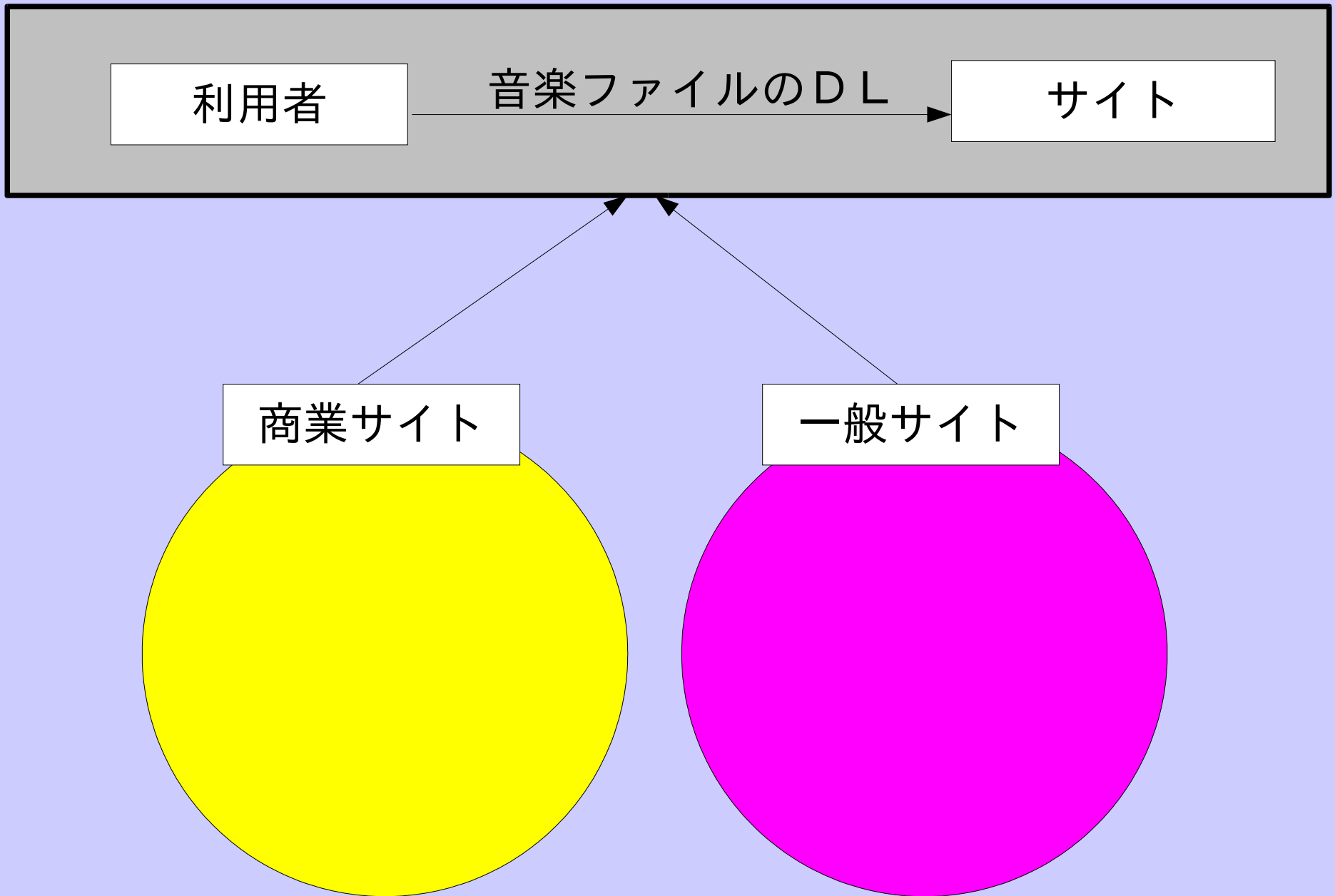
- 研究の概要
- 実験方法
- 実験結果
- 考察
- 今後の課題

研究の概要

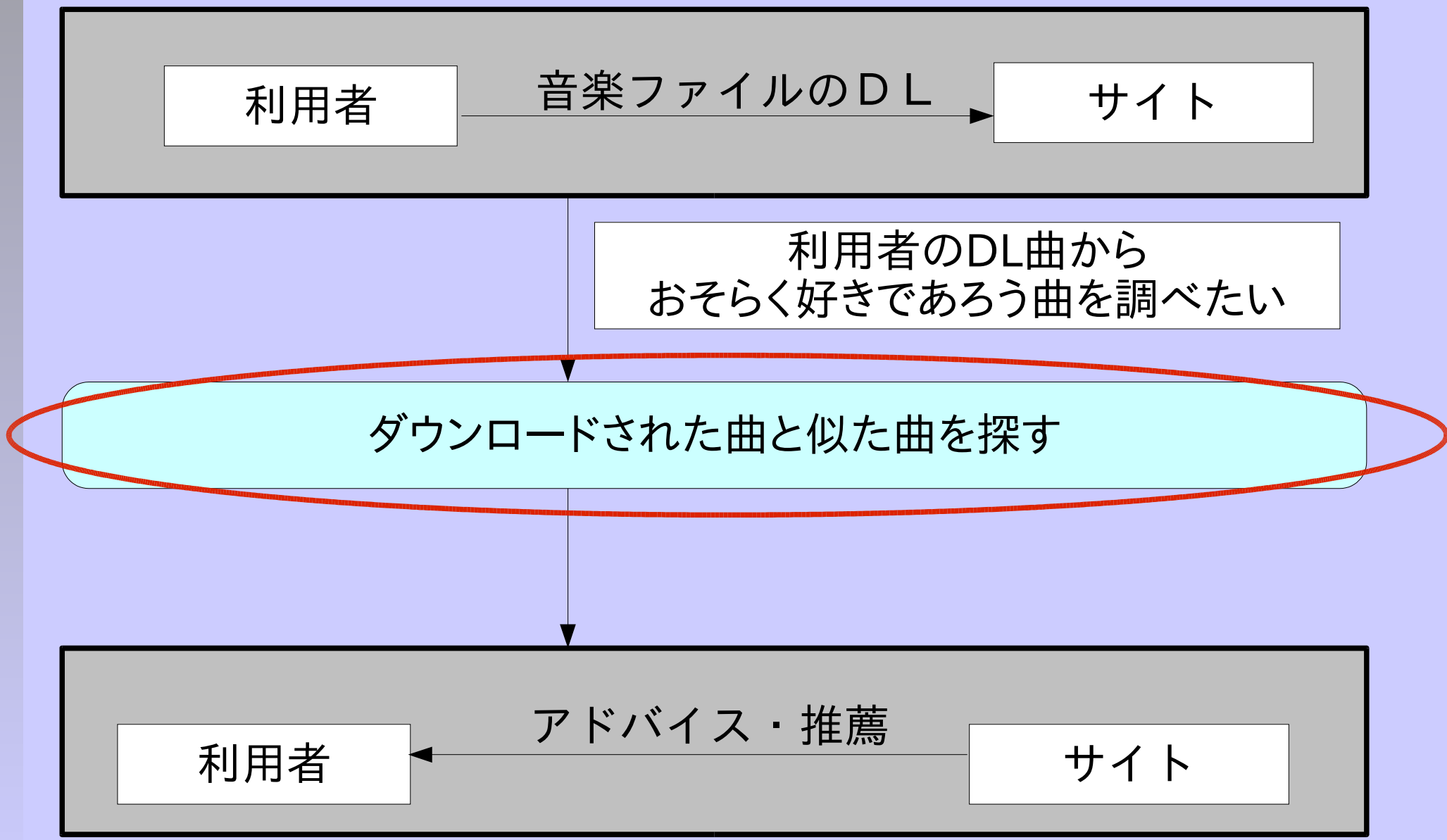
デジタル音楽分布



研究の概要



研究の概要


利用者
音楽ファイルのＤＬ
サイト
利用者のDL曲から
おそらく好きであろう曲を調べたい
ダウンロードされた曲と似た曲を探す
アドバイス・推薦
利用者
サイト

研究の概要

現在：専門家による組織化

↓

理想：自動分類(機械による組織化)

↓

Kolmogorov 記述量に基づいて自動分類する

（研究目的）

研究の概要

先行研究

Rubi氏等

Kolmogorov記述量に基づいた類似度を用いて
電子的に保存された音楽ファイルを自動的に分類

対象：MIDIファイル

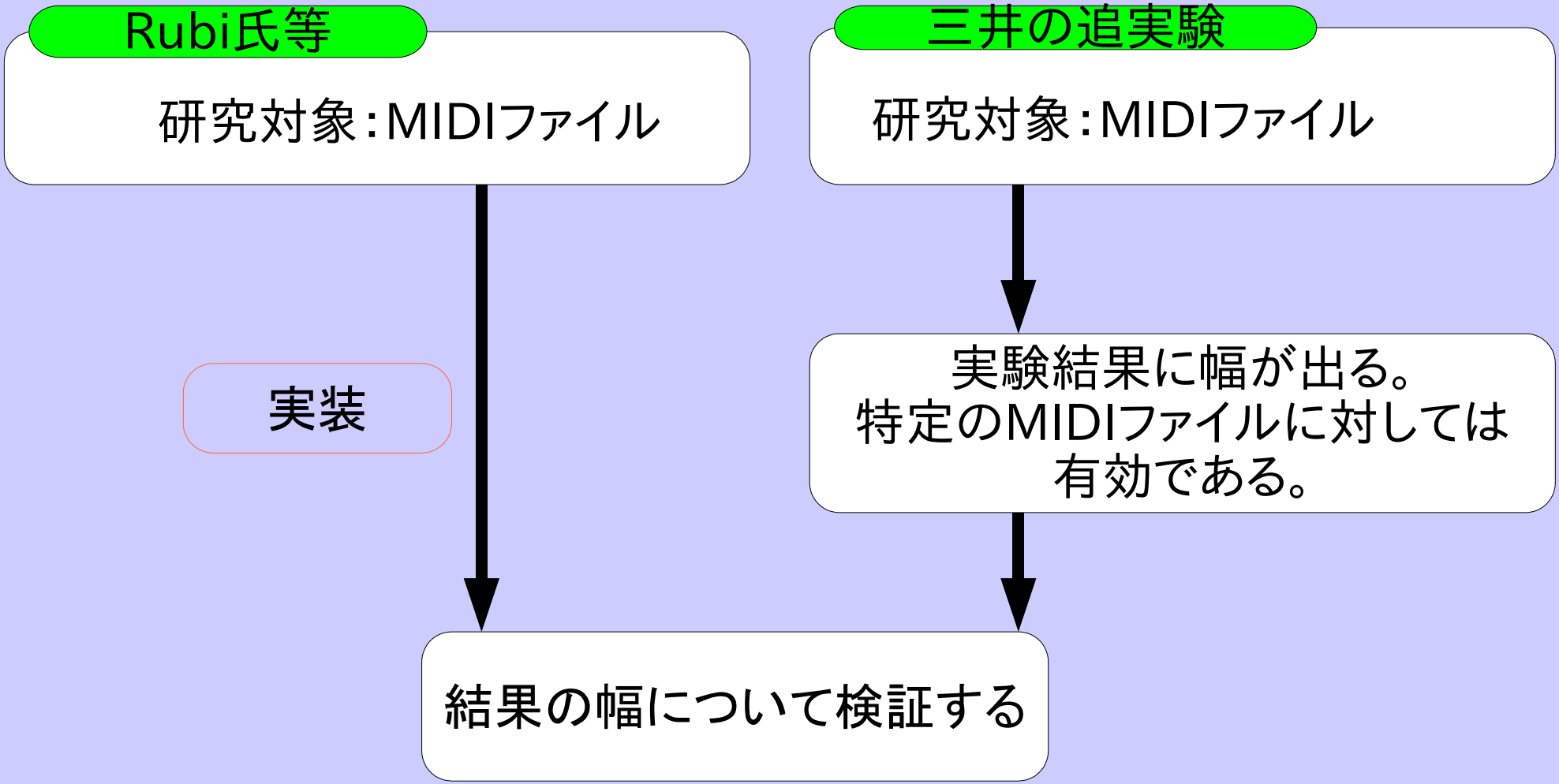
対象楽曲：クラシック，ジャズ，ロックなど

三井による追実験

音楽の分類以外にも，地理的背景や歴史的背景が
出現するかの検証．

対象：MIDIファイル
対象楽曲：クラシック，民謡，など

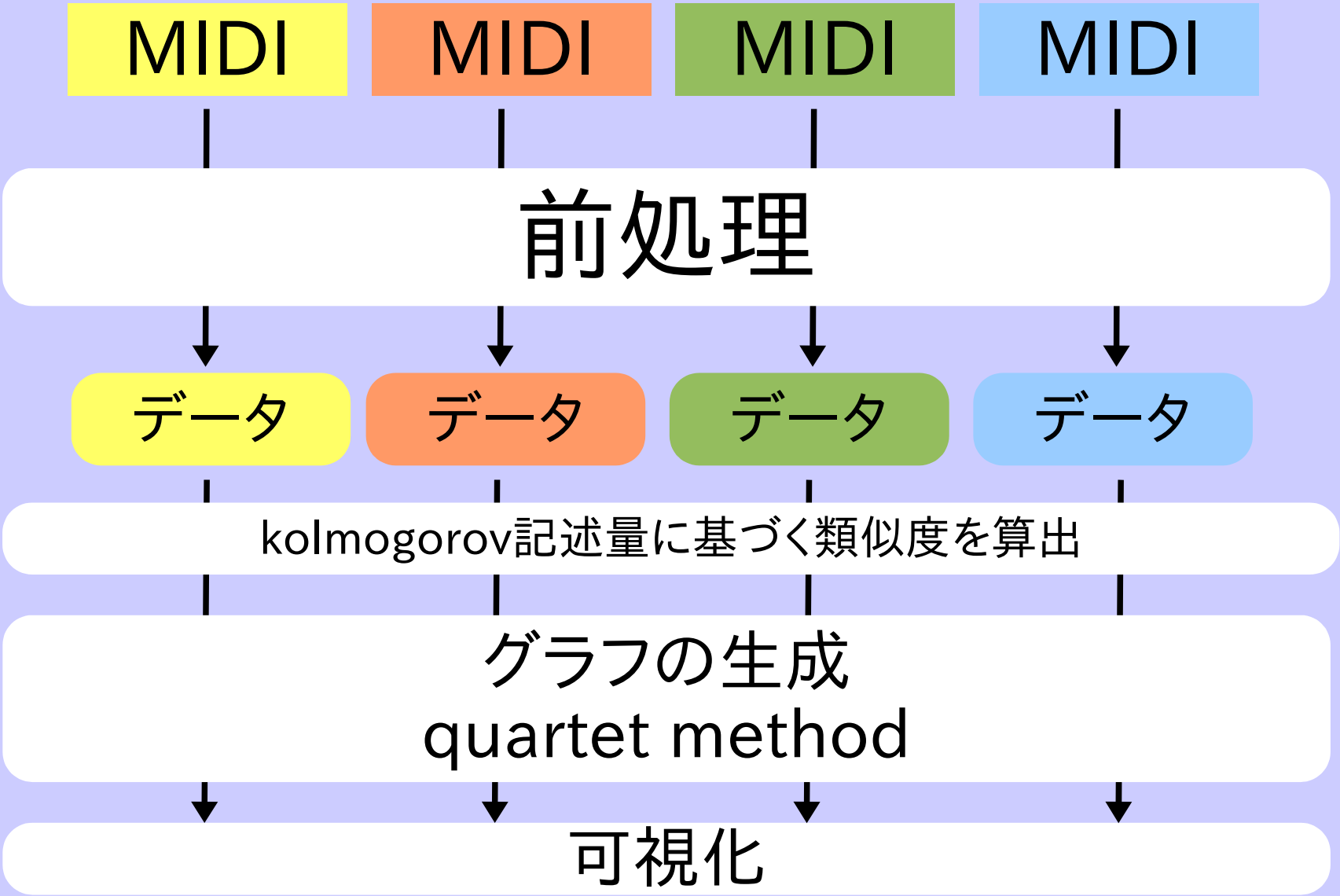
研究の概要


Rubi氏等
研究対象：MIDIファイル
三井の追実験
研究対象：MIDIファイル
実装
実験結果に幅が出る。
特定のMIDIファイルに対しては
有効である。
結果の幅について検証する

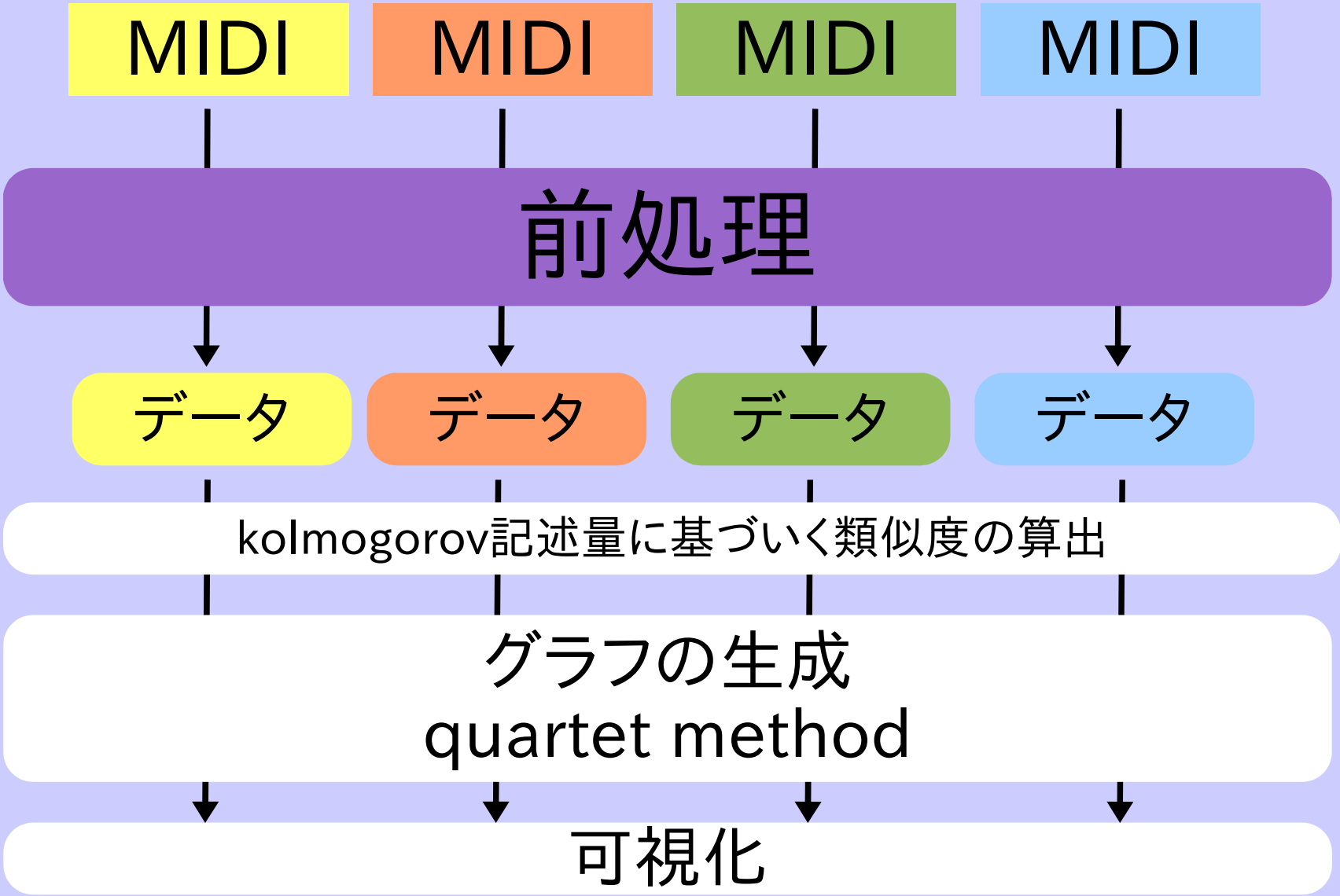
目次

- 研究の概要
- 実験方法
- 実験結果
- 考察
- 今後の課題

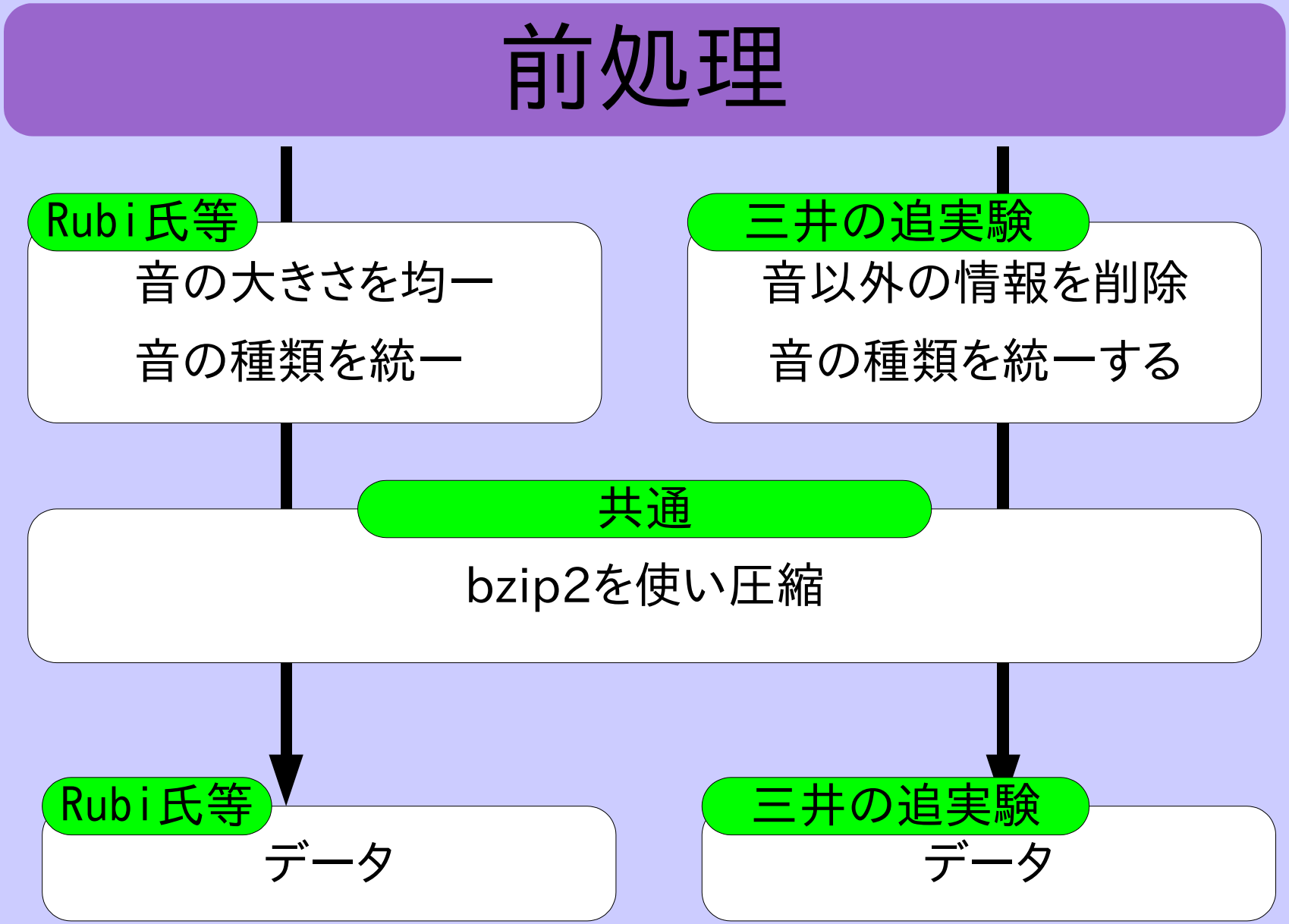
実験方法


MIDI
MIDI
MIDI
MIDI
前処理
データ
データ
データ
データ
kolmogorov記述量に基づく類似度を算出
グラフの生成
quartet method
可視化

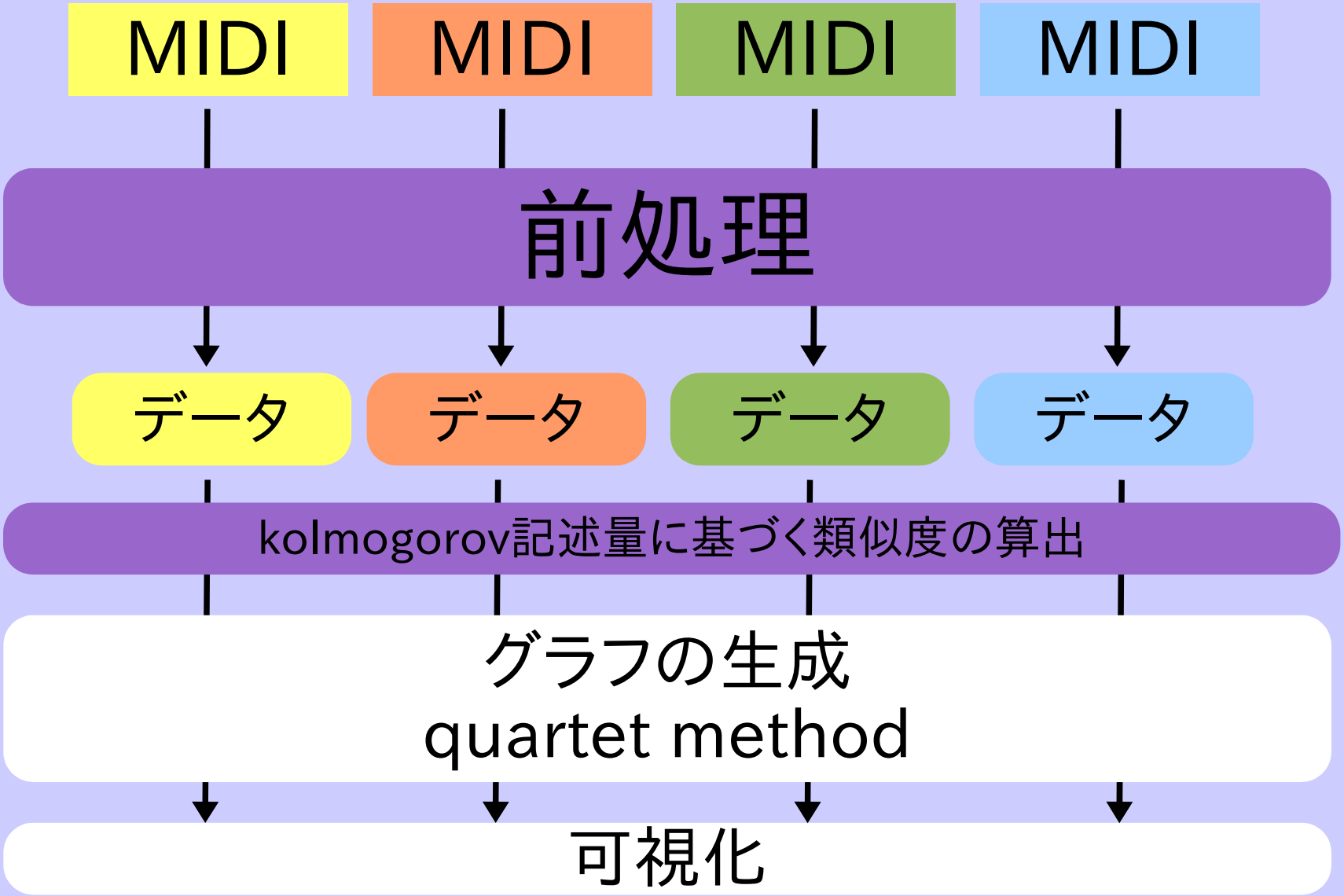
実験方法


MIDI
MIDI
MIDI
MIDI
前処理
データ
データ
データ
データ
kolmogorov記述量に基づいく類似度の算出
グラフの生成
quartet method
可視化

実験方法


前処理
Rubi氏等
音の大きさを均一
音の種類を統一
三井の追実験
音以外の情報を削除
音の種類を統一する
共通
bzip2を使い圧縮
Rubi氏等
データ
三井の追実験
データ

実験方法


MIDI
MIDI
MIDI
MIDI
前処理
データ
データ
データ
データ
kolmogorov記述量に基づく類似度の算出
グラフの生成
quartet method
可視化

kolmogorov記述量に基づく類似度の算出

$$d(x,y) = \frac{max\{K(x|y), K(y|x)\}}{max\{K(x), K(y)\}}$$

$K(y) \geq K(x)$ としたとき,

$$\begin{aligned} d(x,y) &= \frac{K(y|x)}{K(y)} \\ &= \frac{K(y) - I(x:y)}{K(y)} = \frac{K(xy) - K(x)}{K(y)} \end{aligned}$$

kolmogorov記述量に基づく類似度の算出

K : *Kolmogorov* 記述量

$$d(x,y) = \frac{K(xy) - K(x)}{K(y)}$$

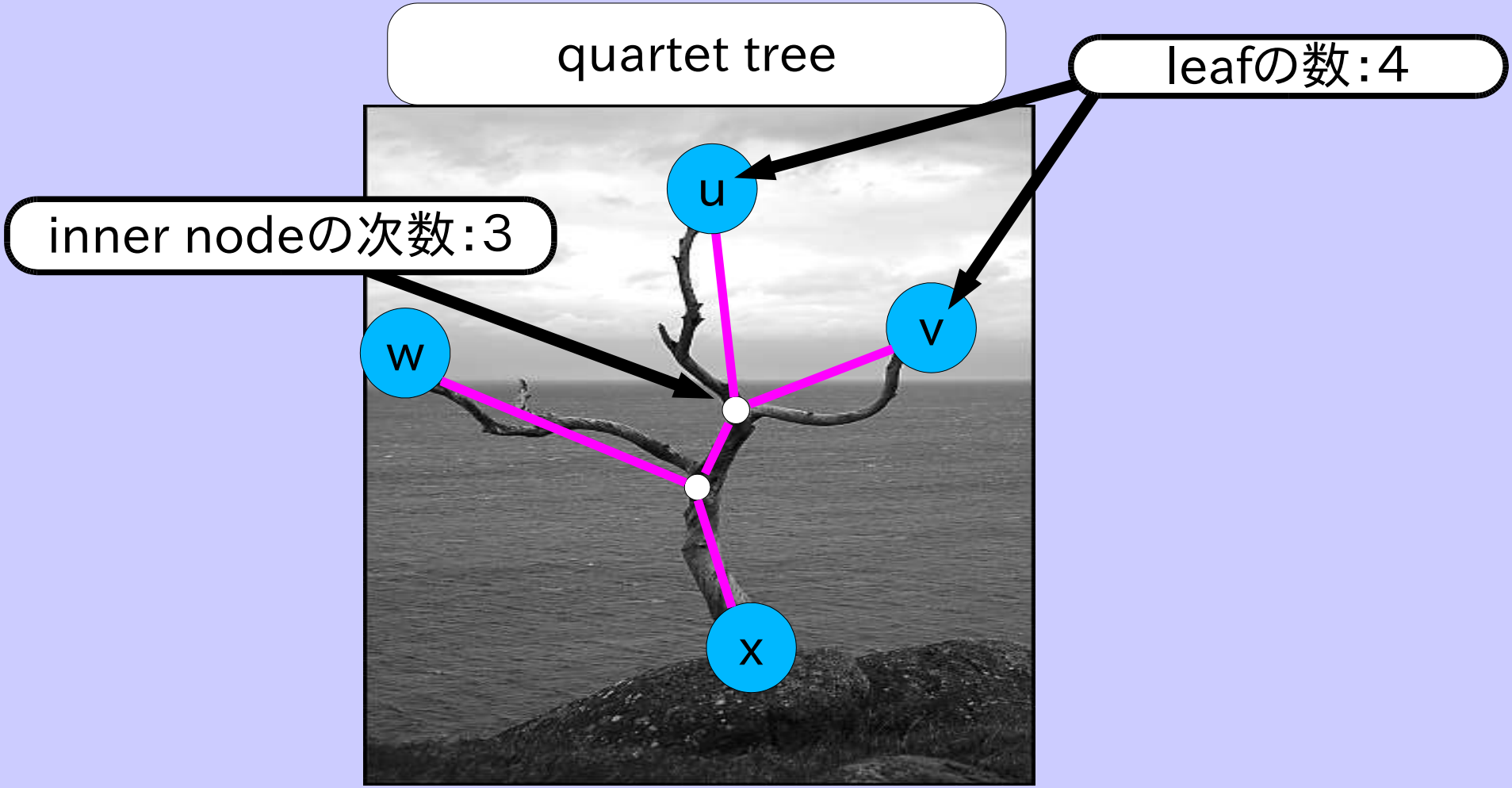
(1) 要求された情報距離 $d(x,y)$ は漠然と長い文字列 x,y によって得られる.

(2) Kolmogorov Complexity は帰納的でないため，計算不可能である.

(3) 実用的な方法で情報距離を近似する際，圧縮方法の一つである“bzip2”を用いる.

$$d(x,y) \fallingdotseq \frac{bzip(xy) - bzip(x)}{bzip(y)}$$

kolmogorov記述量に基づく類似度の算出



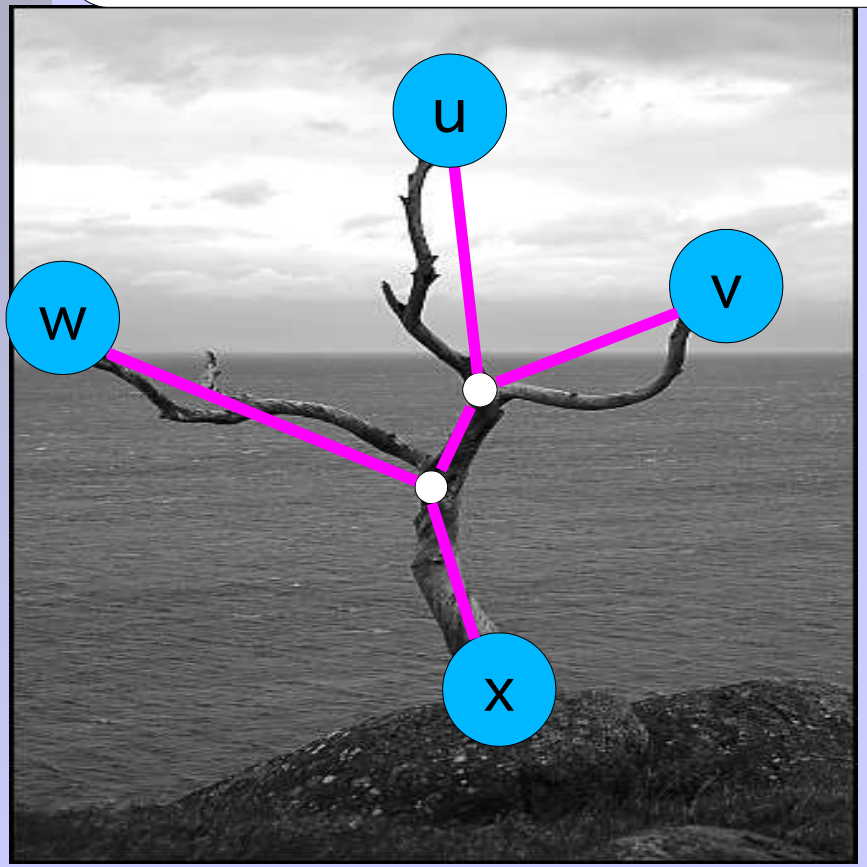
kolmogorov記述量に基づく類似度の算出

quratet treeの評価

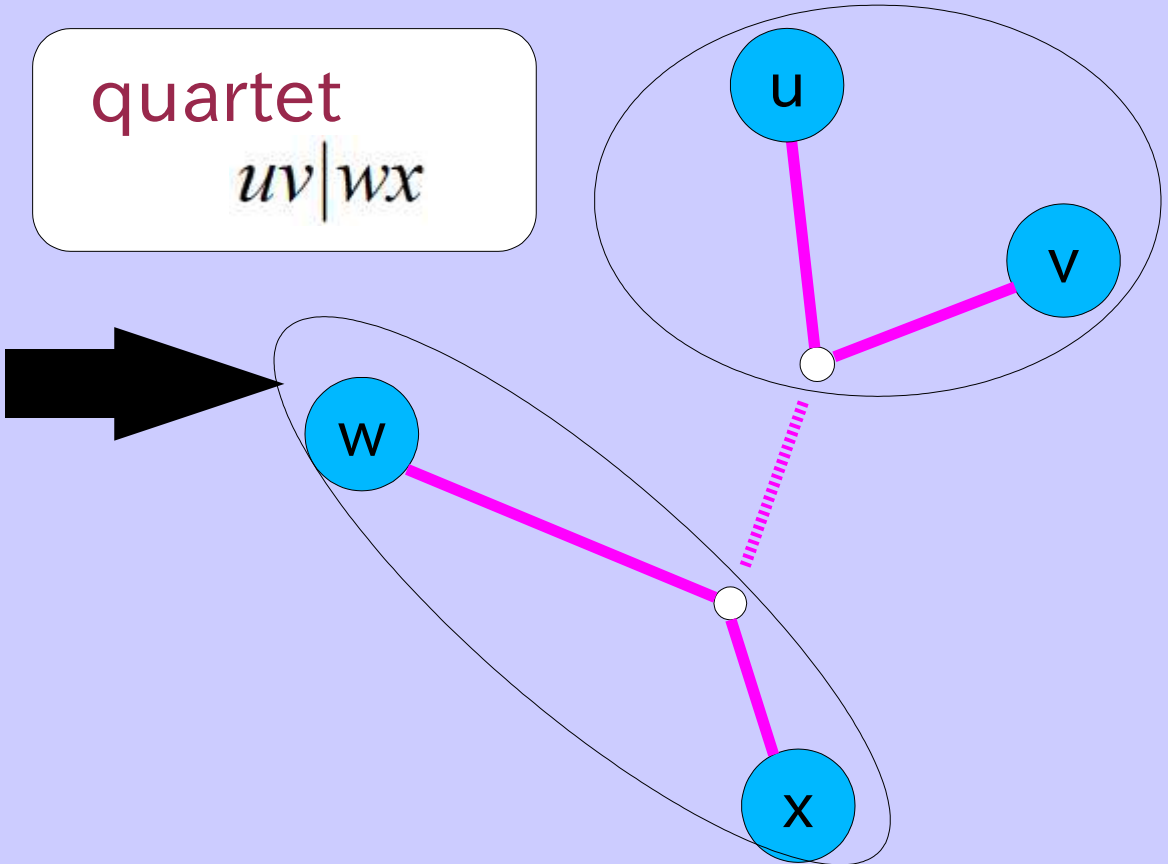
$$C_{uv|wx} = d(u,v) + d(w,x)$$

$$d(x,y) \fallingdotseq \frac{bzip(xy) - bzip(x)}{bzip(y)}$$

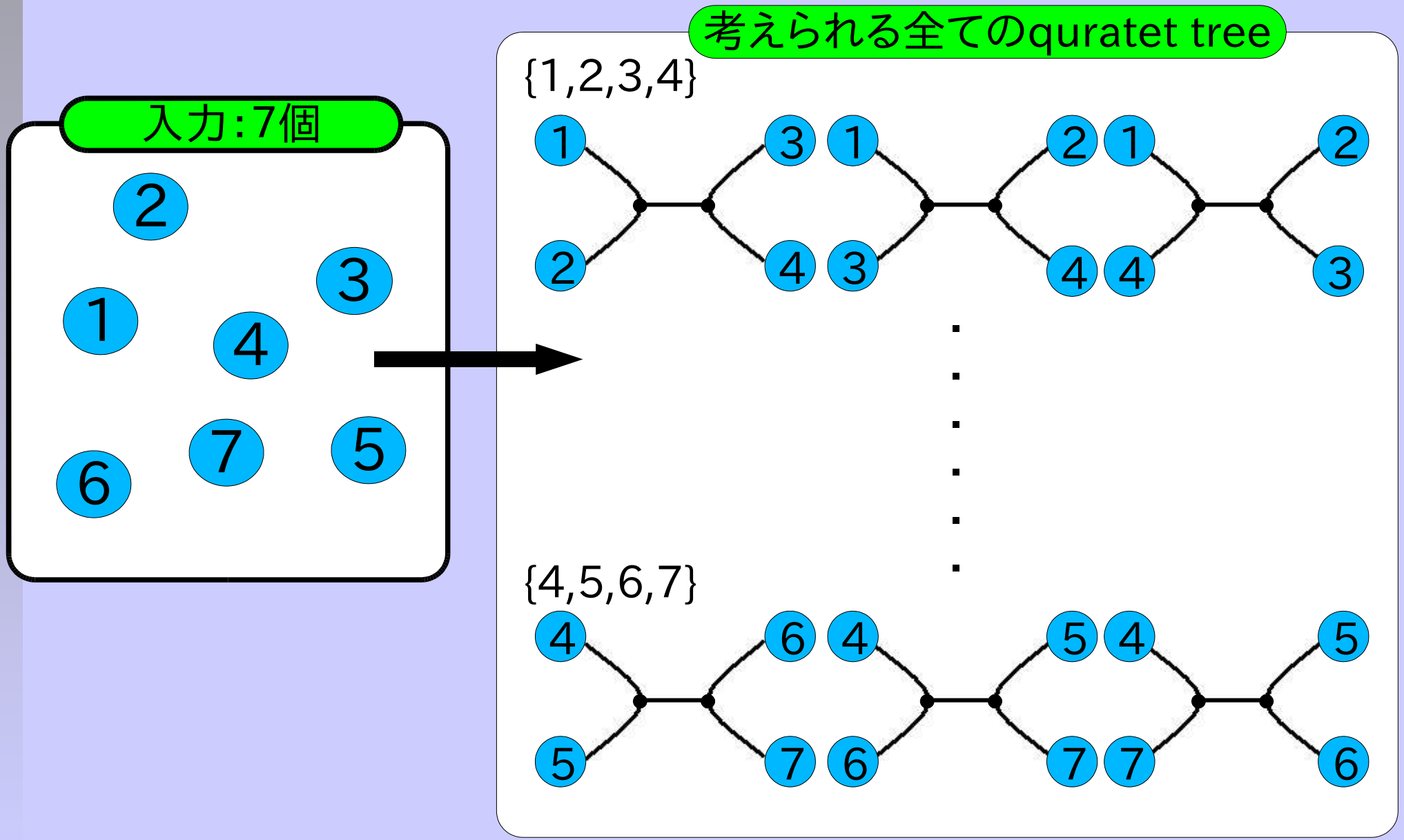
$C_{uv|wx}$ が小さいほど良い木



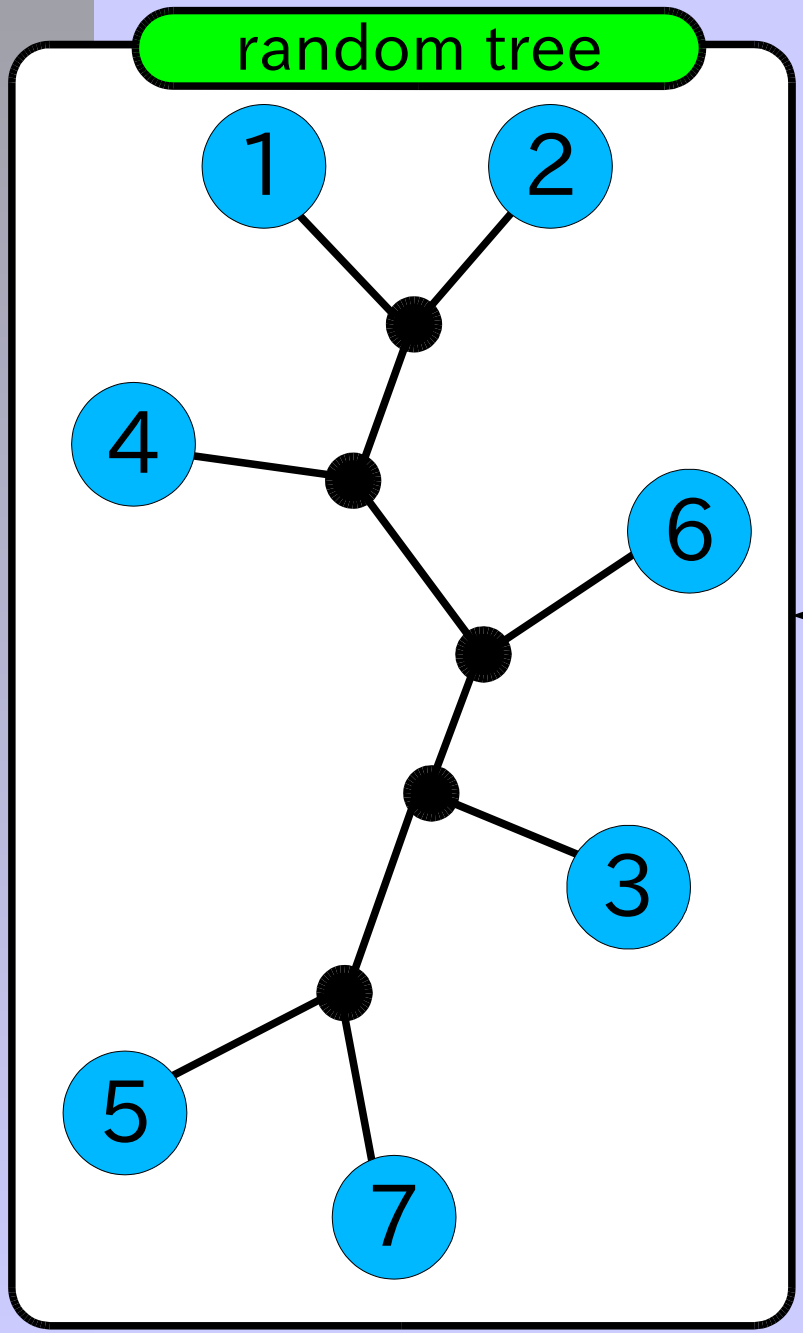
quartet
$uv|wx$

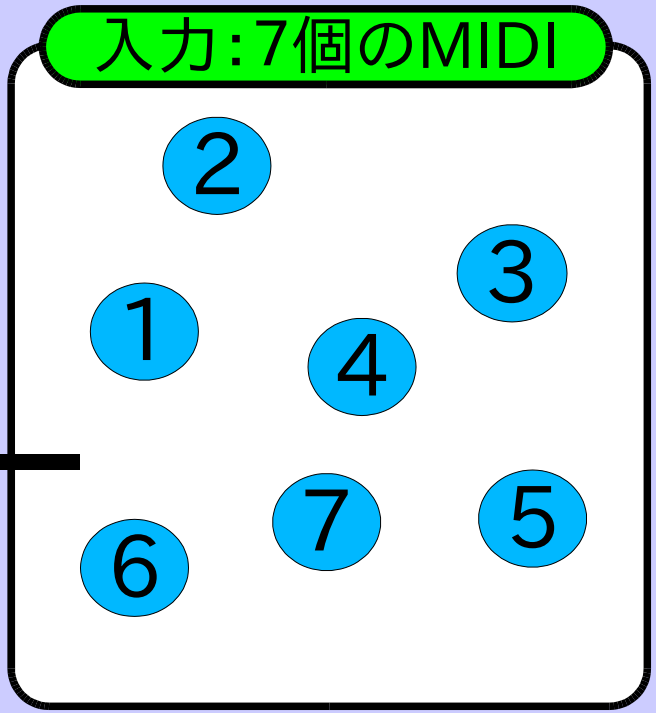


kolmogorov記述量に基づく類似度の算出



kolmogorov記述量に基づく類似度の算出



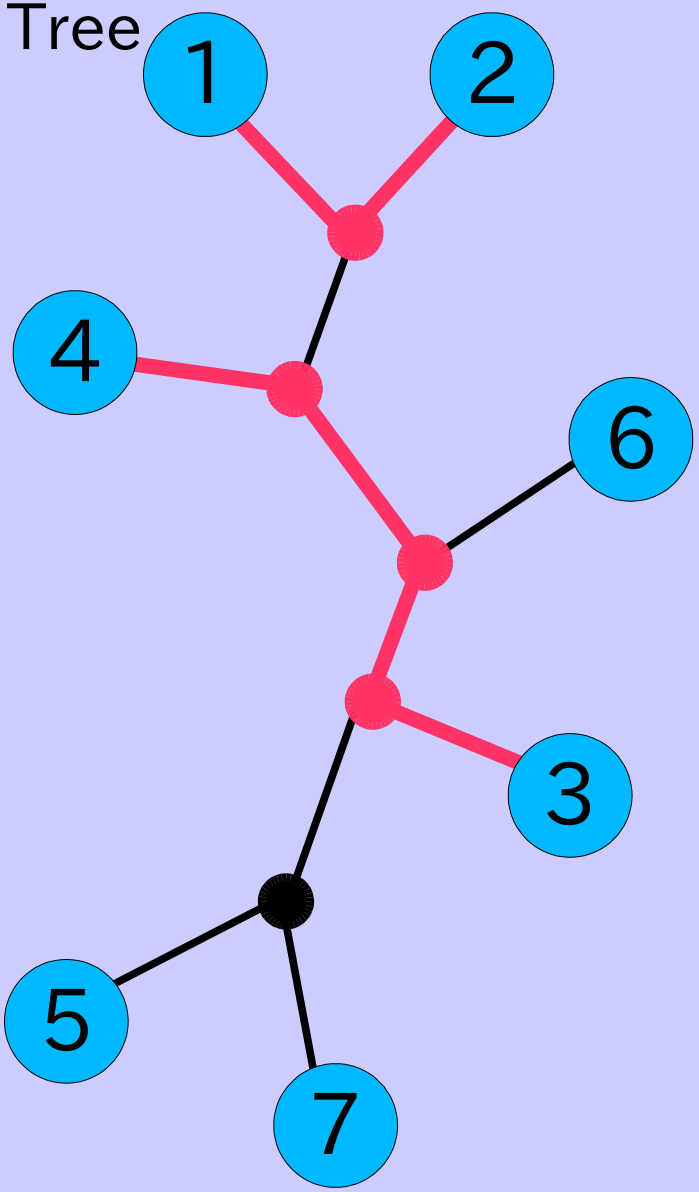


random tree rule

leaf : n個
inner node : n-2個
inner node次数:3

kolmogorov記述量に基づく類似度の算出

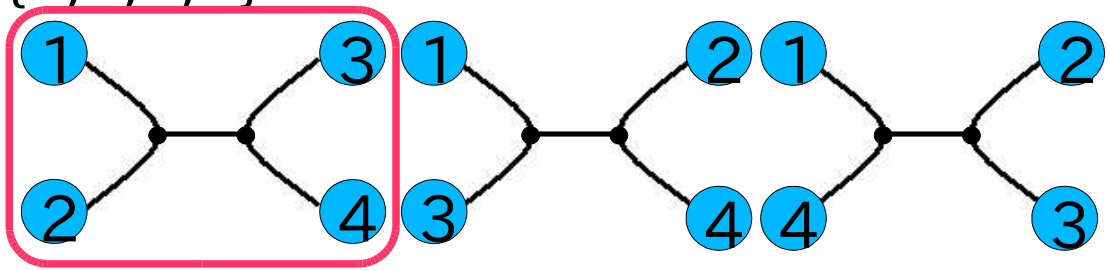
random tree



存在しているquratet treeの組合せを探す

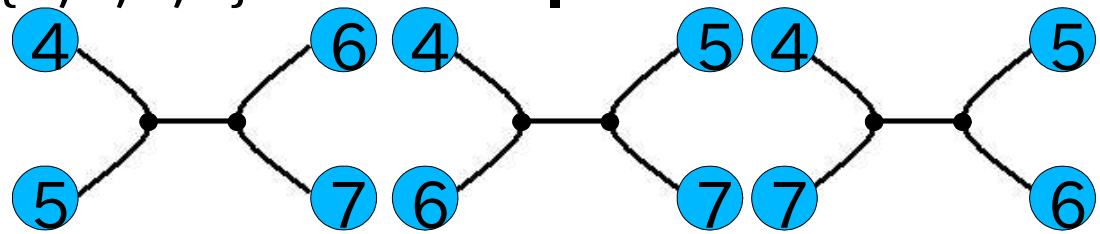
考えられる全てのquratet tree

{1,2,3,4}

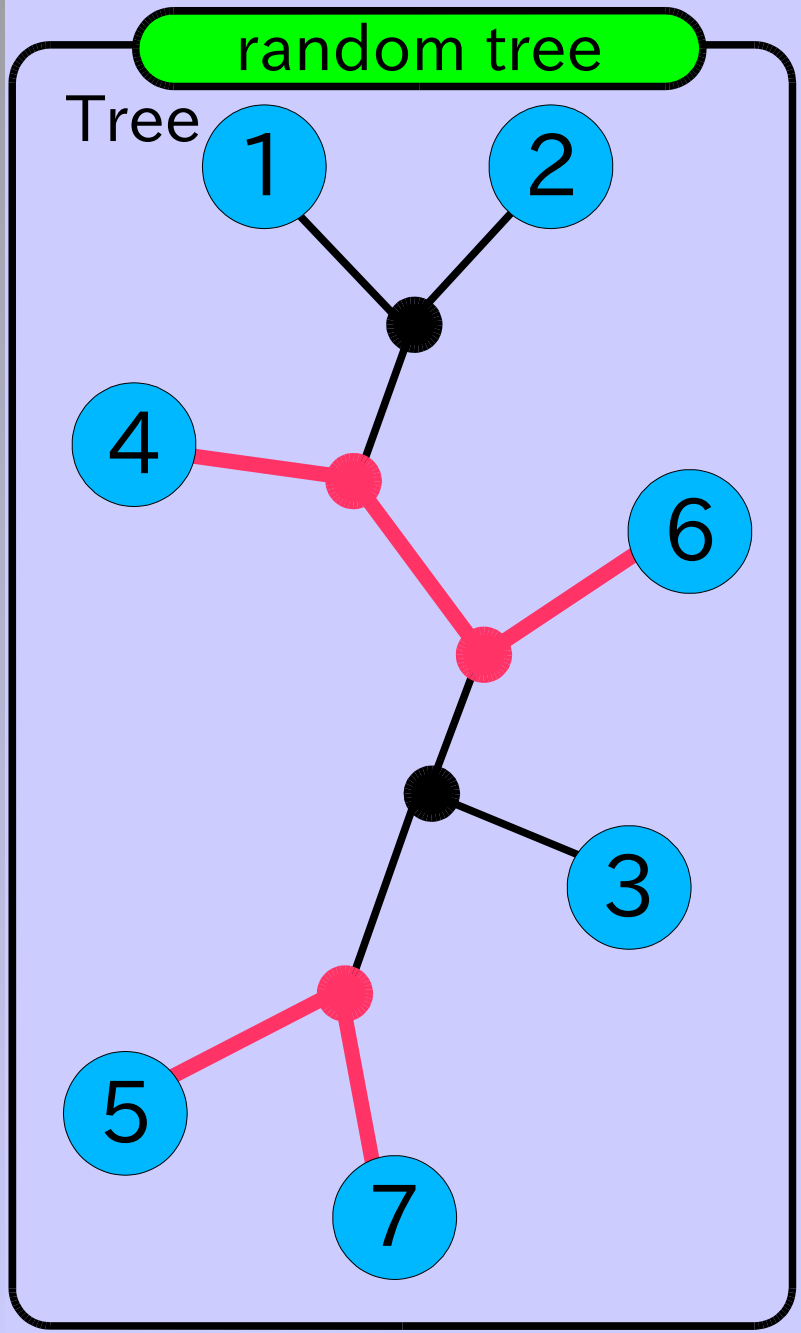


consistent

{4,5,6,7}

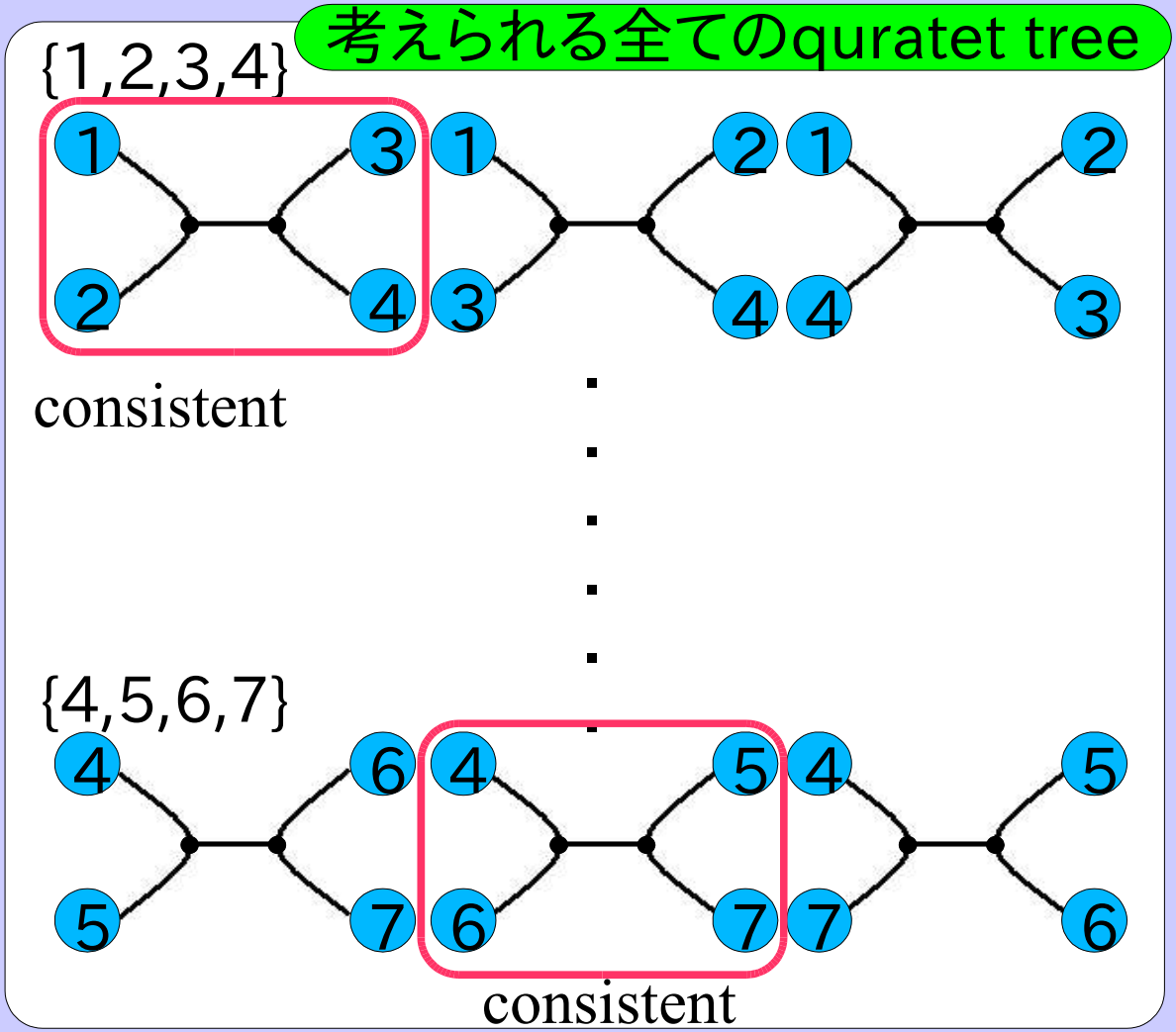


kolmogorov記述量に基づく類似度の算出



存在しているquratet treeの組合せを探す

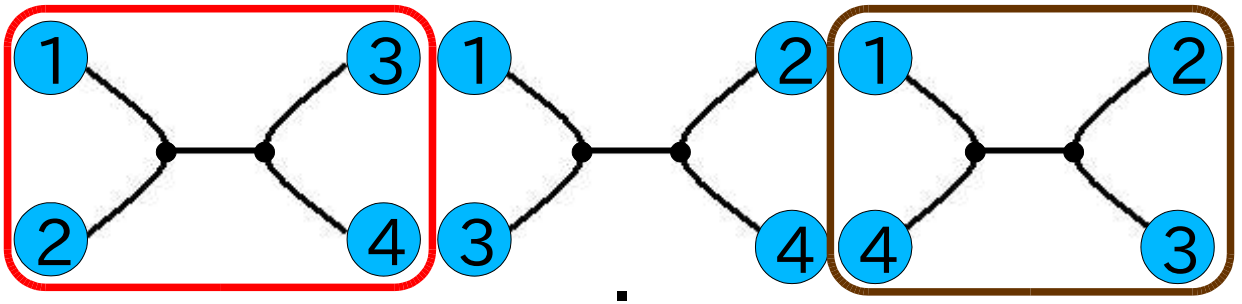
考えられる全てのquratet tree



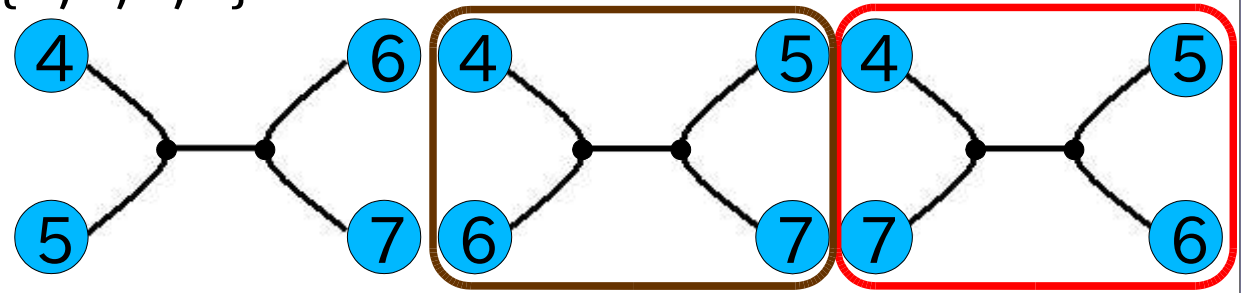
kolmogorov記述量に基づく類似度の算出

考えられる全てのquratet tree

{1,2,3,4}



{4,5,6,7}



$$C_{uv|wx} = d(u,v) + d(w,x)$$

Maximum Cost

3つの組の中で

$C_{uv|wx}$ の値が

一番大きい組合せを選択

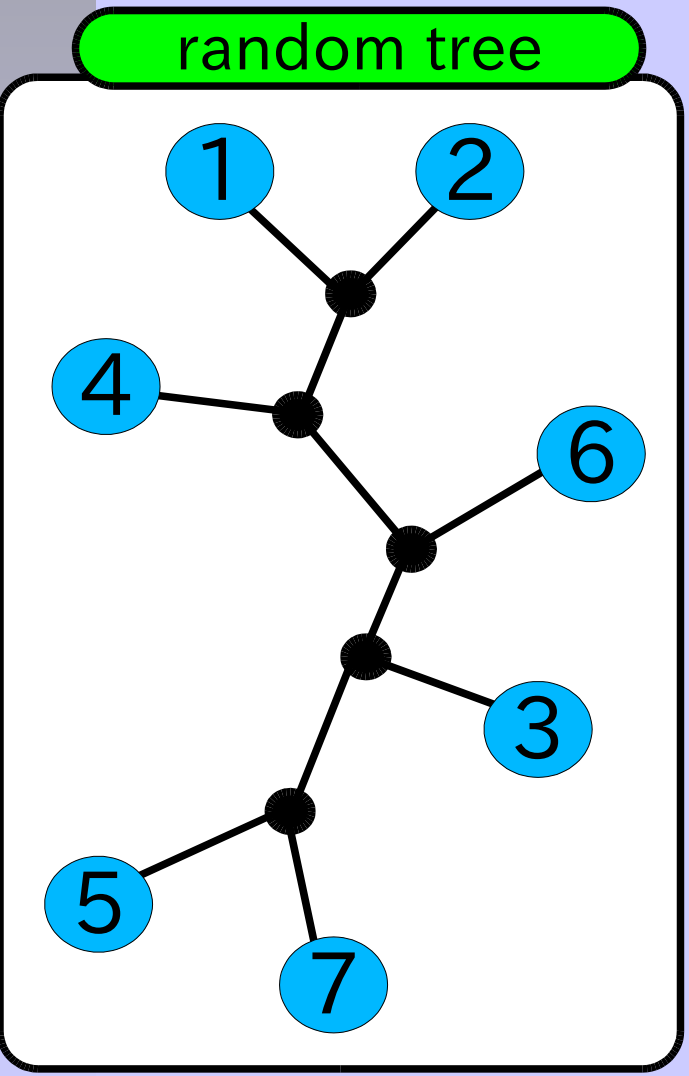
Minimum Cost

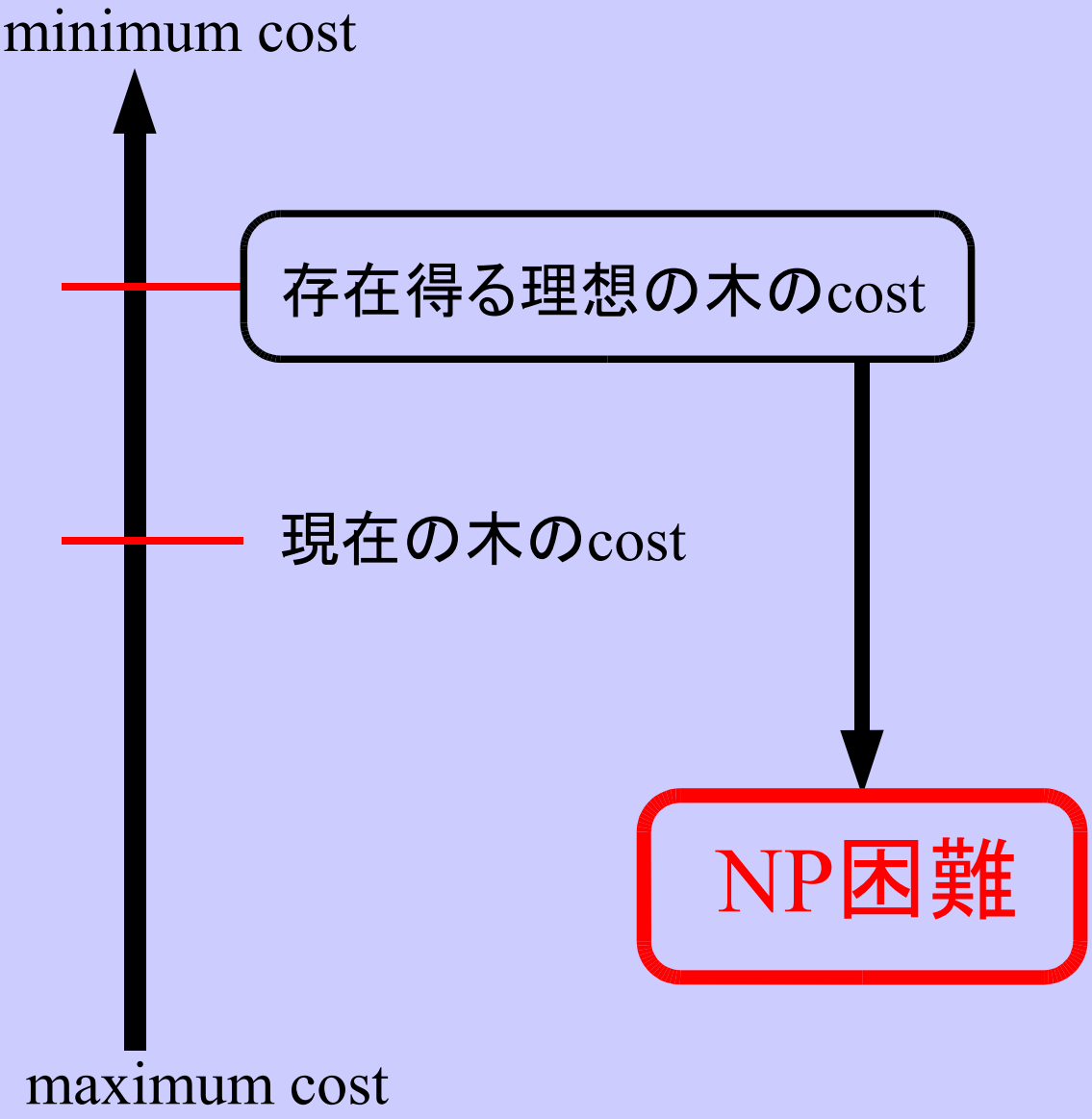
3つの組の中で

$C_{uv|wx}$ の値が

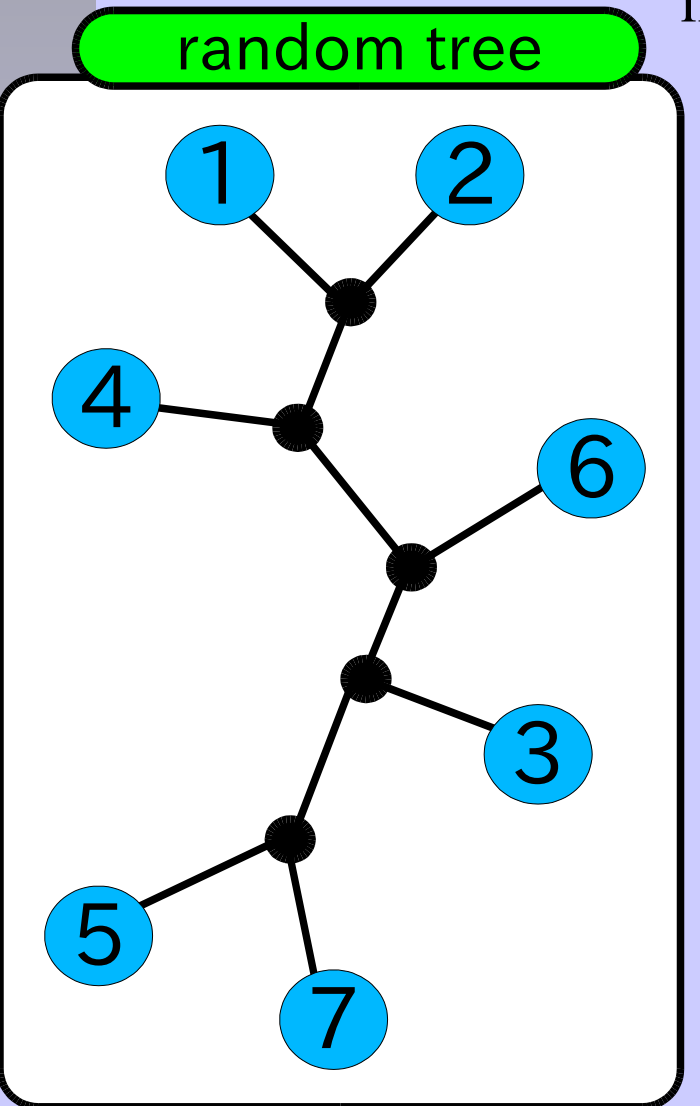
一番小さい組合せを選択

kolmogorov記述量に基づく類似度の算出





kolmogorov記述量に基づく類似度の算出



minimum cost **S(T) = 1**

存在し得る理想の木のcost

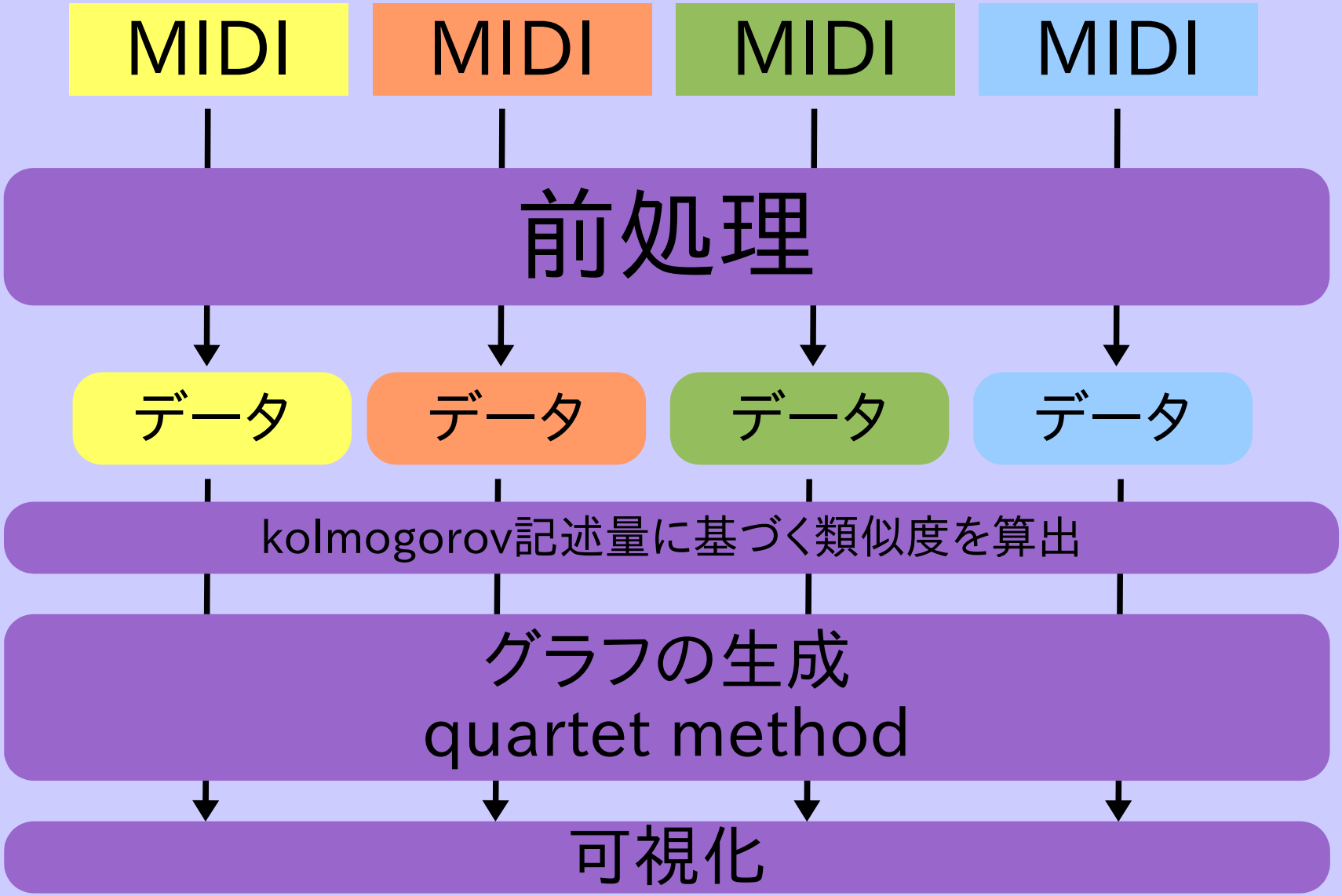
現在の木のcost

S(T) = 0~1

maximum cost **S(T) = 0**

ヒューリスティックを用いてS(T)=1へ近づける

先行研究


MIDI
MIDI
MIDI
MIDI
前処理
データ
データ
データ
データ
kolmogorov記述量に基づく類似度を算出
グラフの生成
quartet method
可視化

グラフの生成
quartet method

ランダムに木を生成

↓

S(T)の値を計算

↓

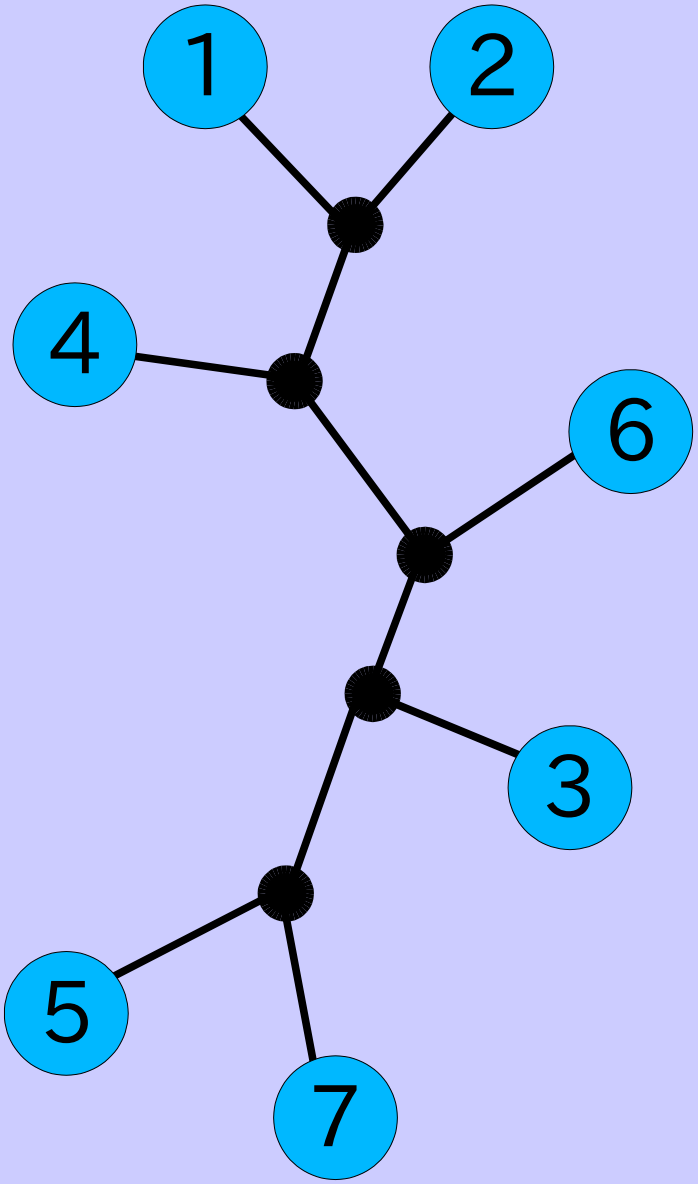
以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

↓

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

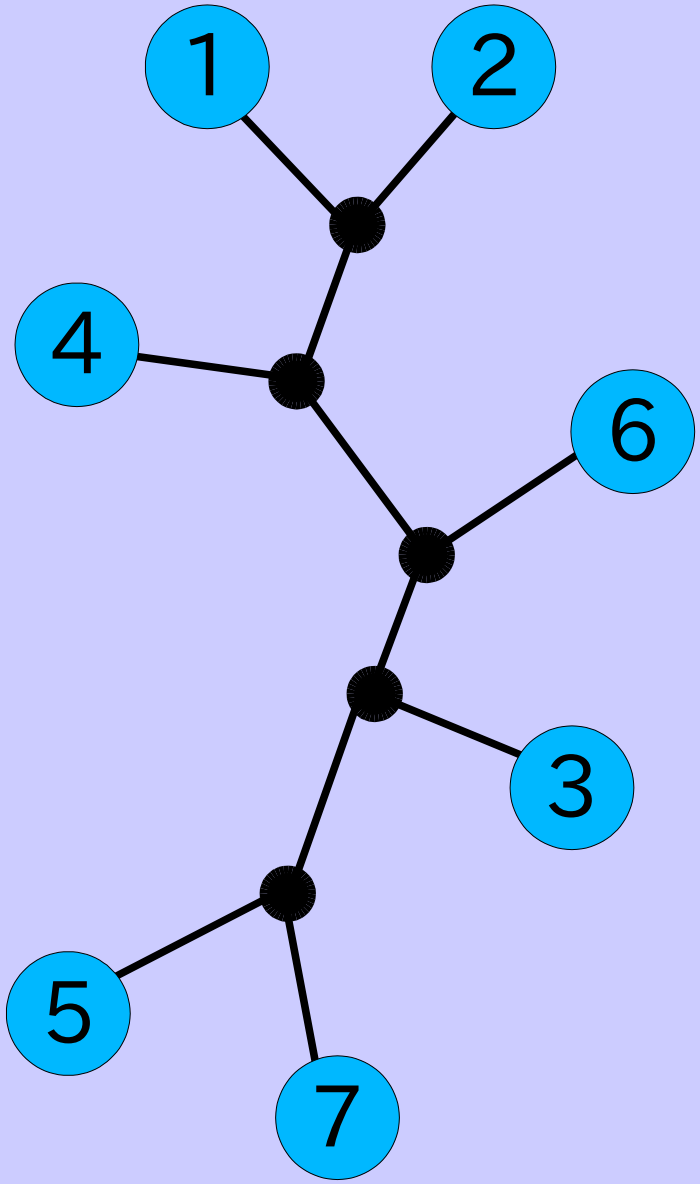
S(T)の値を計算

以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

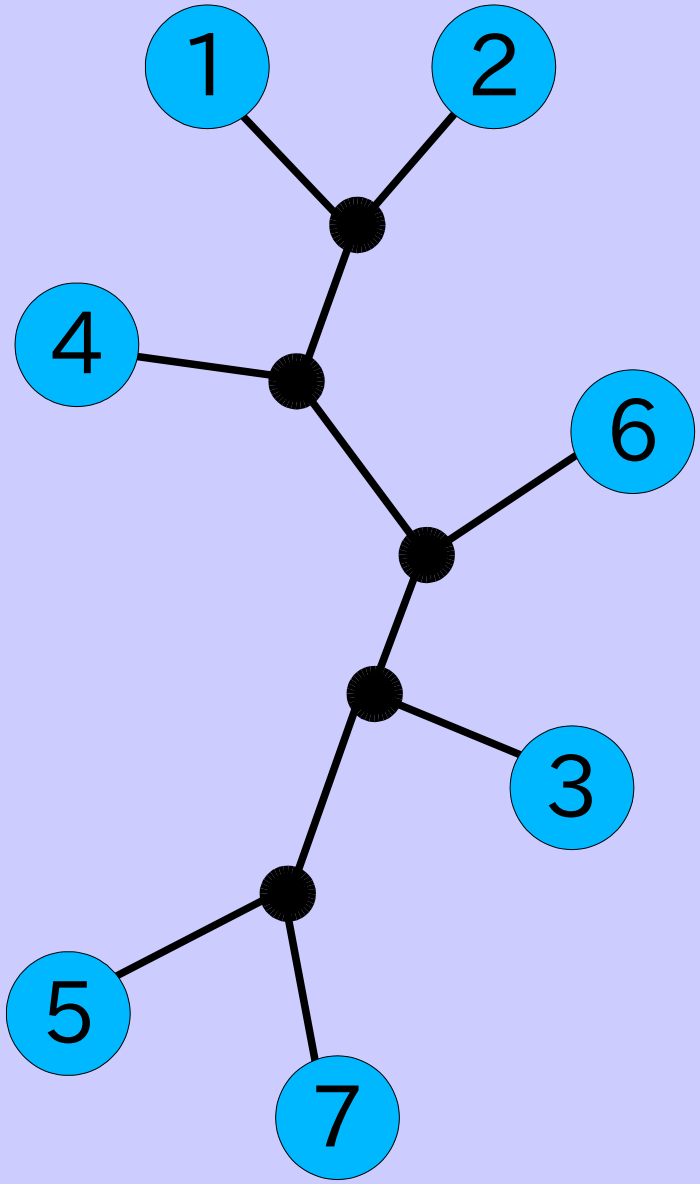
S(T)の値を計算

以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

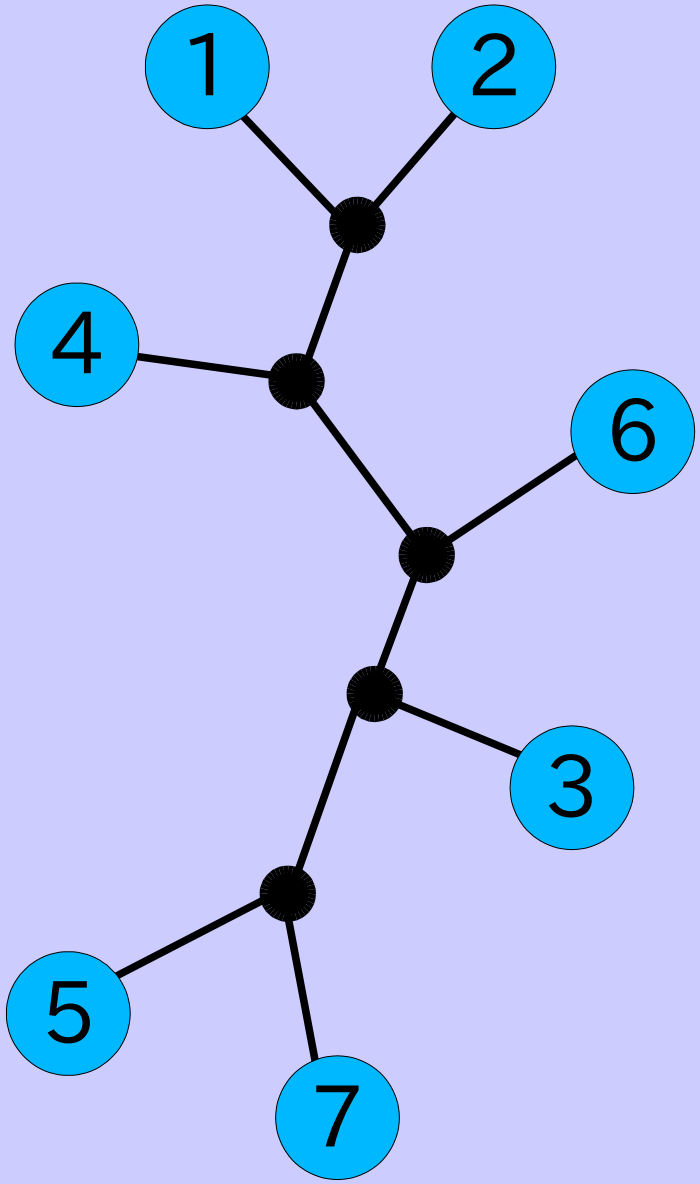
S(T)の値を計算

以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

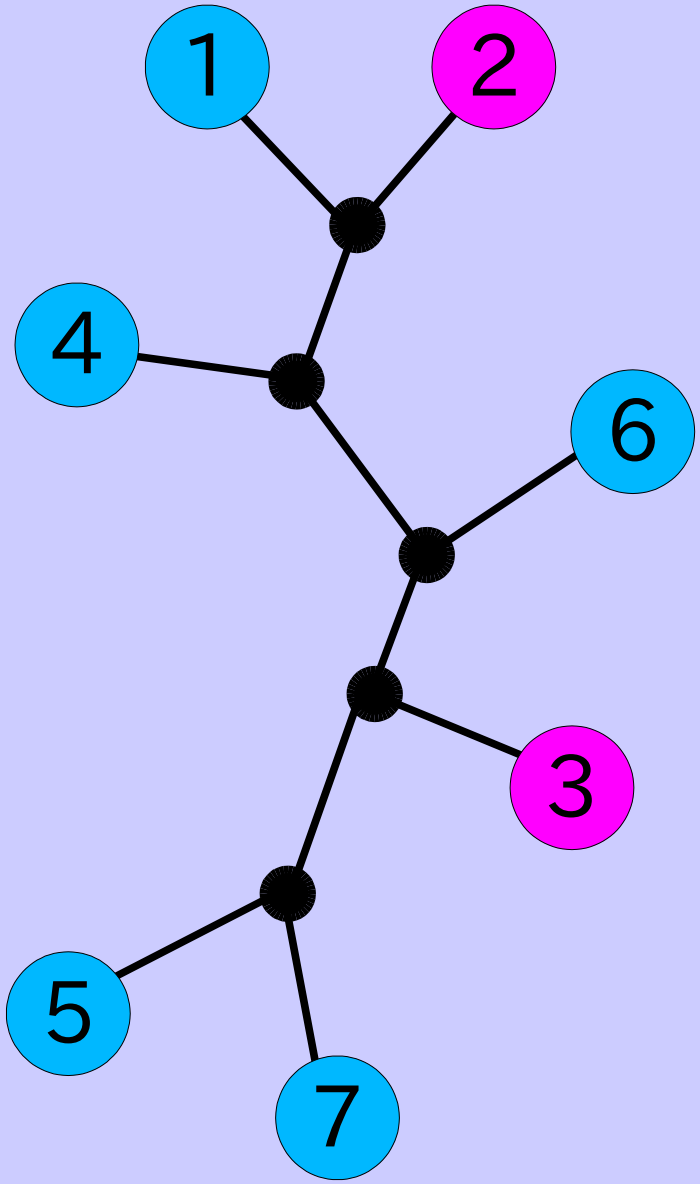
S(T)の値を計算

以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

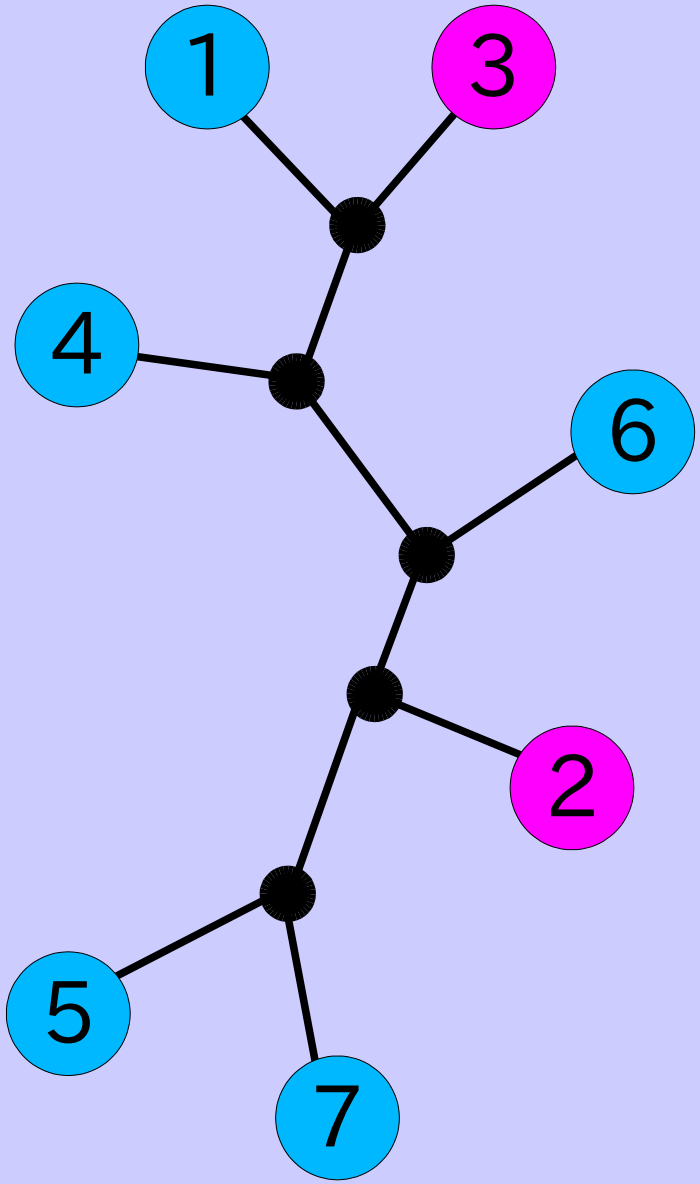
S(T)の値を計算

以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

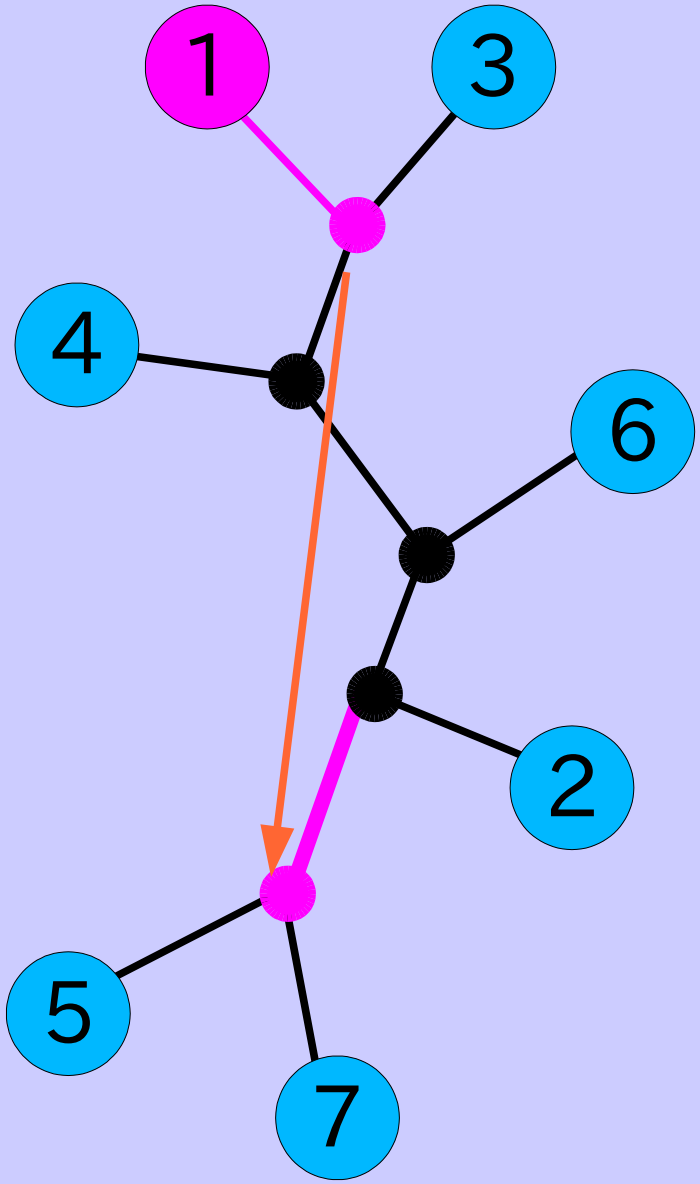
S(T)の値を計算

以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

↓

S(T)の値を計算

↓

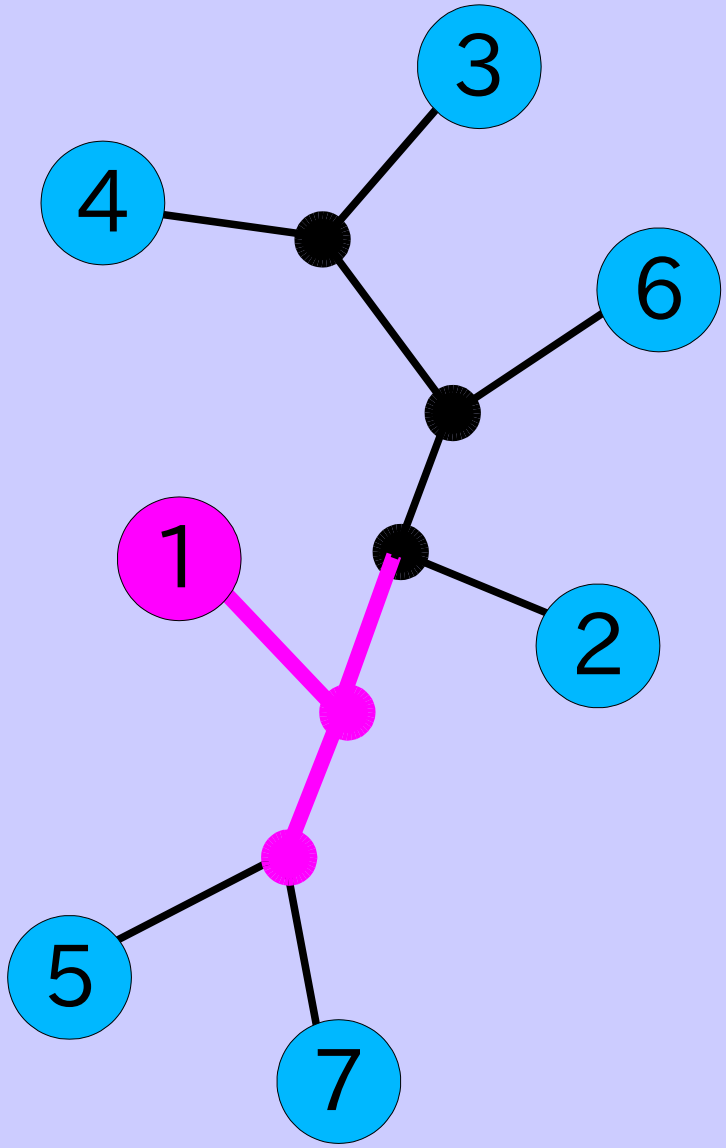
以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

↓

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

↓

S(T)の値を計算

↓

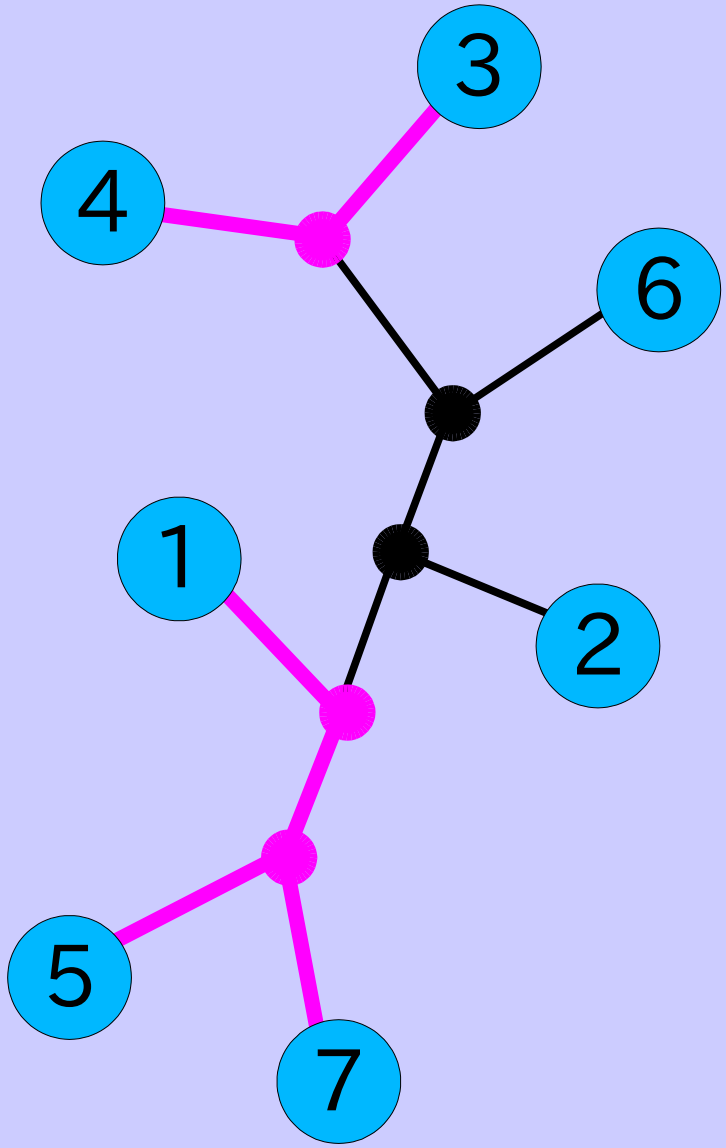
以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

↓

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

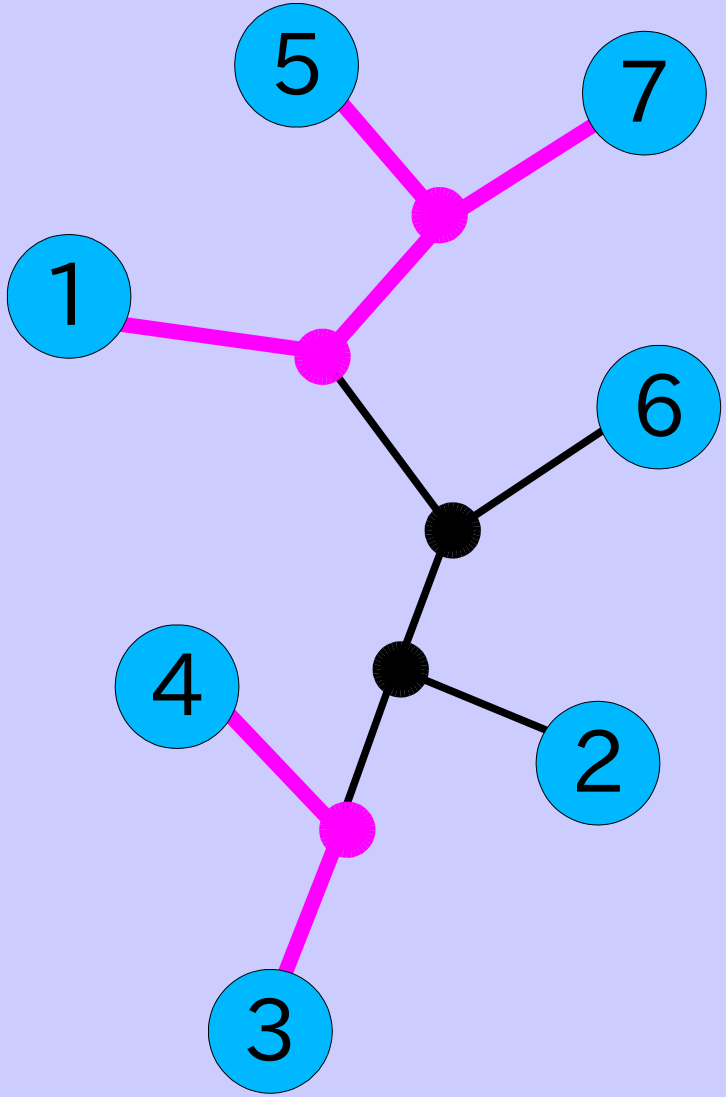
S(T)の値を計算

以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

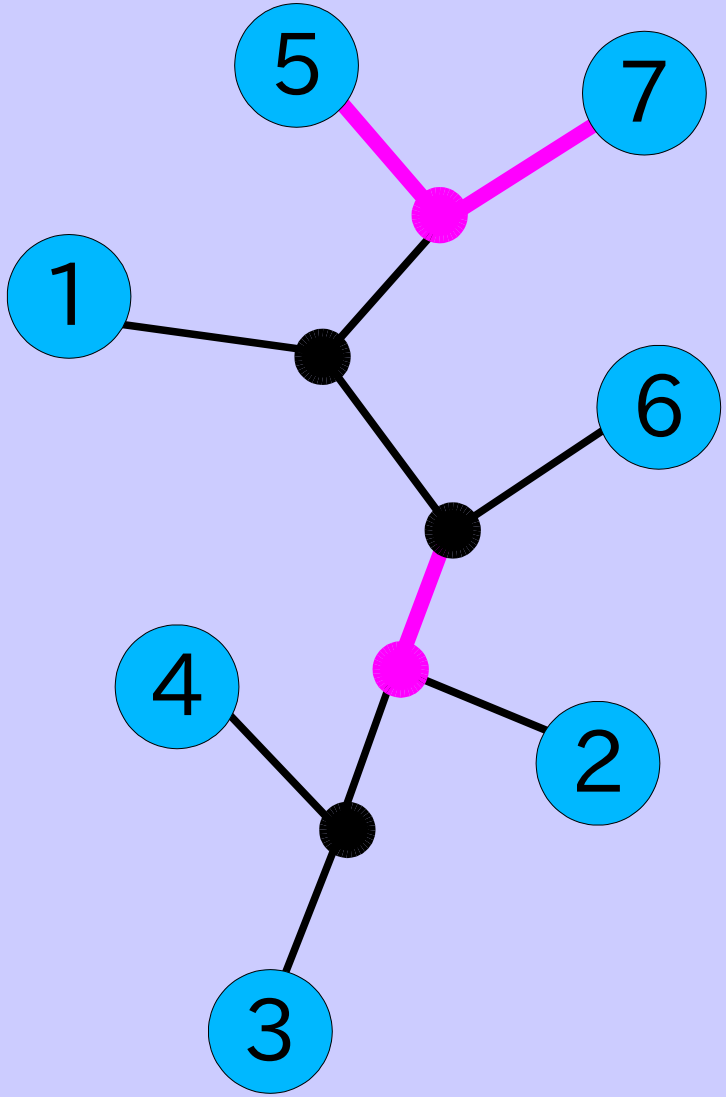
S(T)の値を計算

以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

↓

S(T)の値を計算

↓

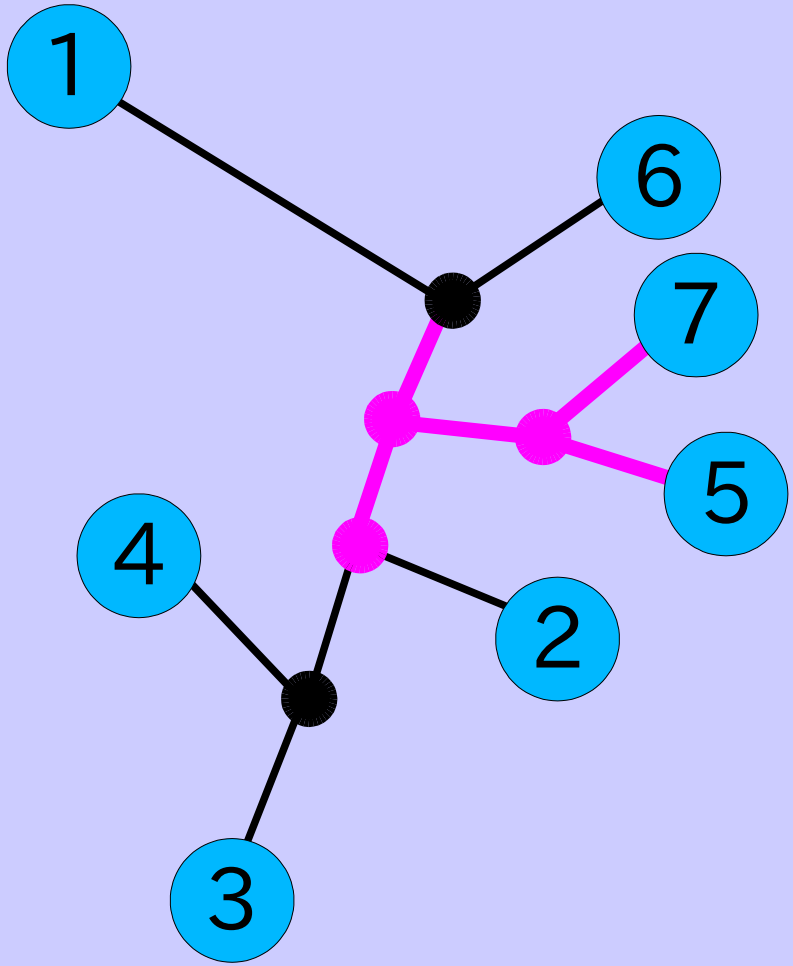
以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

↓

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

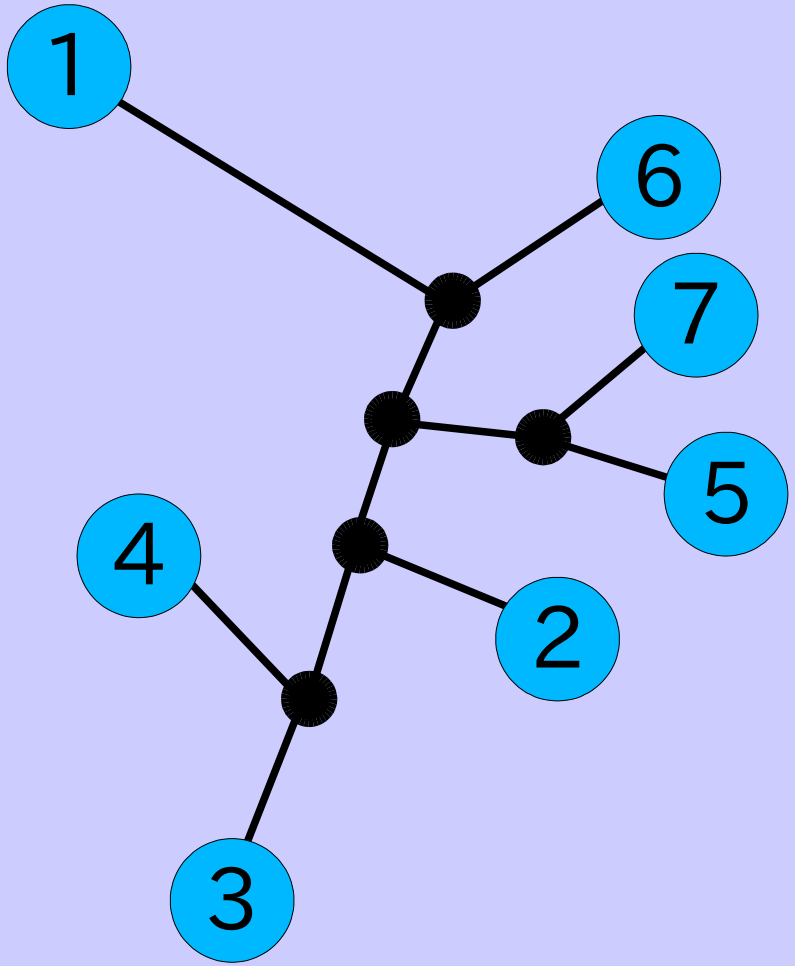
S(T)の値を計算

以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

S(T)の値を計算



グラフの生成
quartet method

ランダムに木を生成

S(T)の値を計算

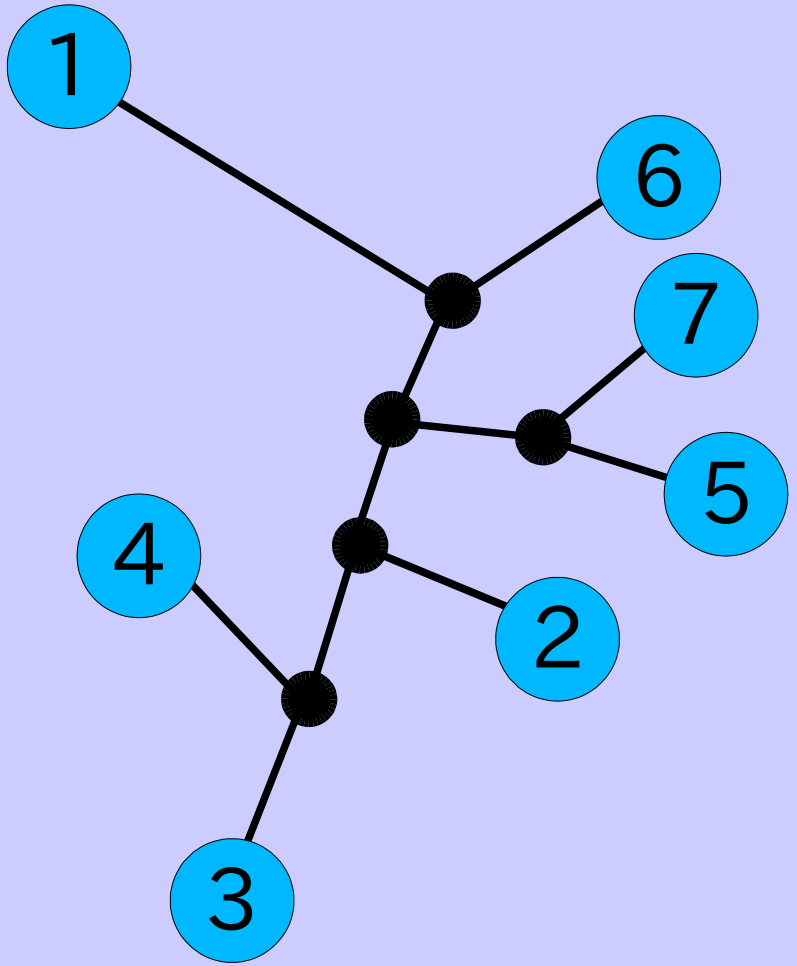
以下の操作からランダムに選ぶ

- leaf swap
- leaf trancefer
- subtree swap
- subtree trancefer

k回繰り返す

S(T)の値を計算

三井のquartet method



目次

- 研究の概要
- 実験方法
- 実験結果
- 考察
- 今後の課題

実験結果

実験1：作曲者で分類されるか(Bach,Debussy,Chopinより
それぞれ4曲)

実験2：地域で分類されるか(世界の２７ヵ国の民謡、
日本のみ2曲)

実験3：地域で分類されるか(scotland 3曲　japan　2曲
america 2曲　france 3曲
spain　2曲)

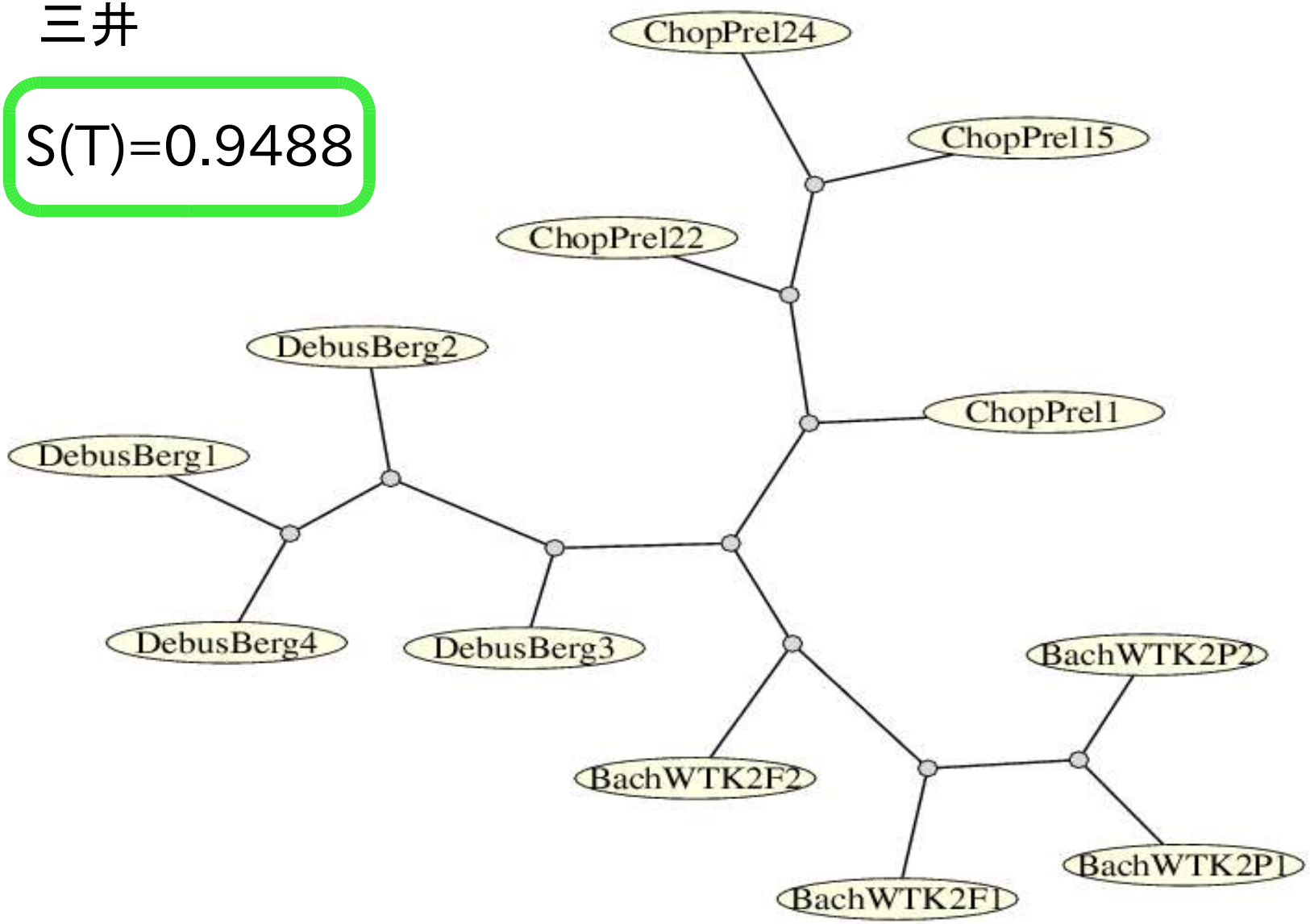
実験4：地域で分類されるか(チャイコフスキー3曲、Erik satie 3曲
Bach 3曲、Mozart 3曲)

実験結果1

Bach,Debussy,Chopinよりそれぞれ4曲

三井

S(T)=0.9488

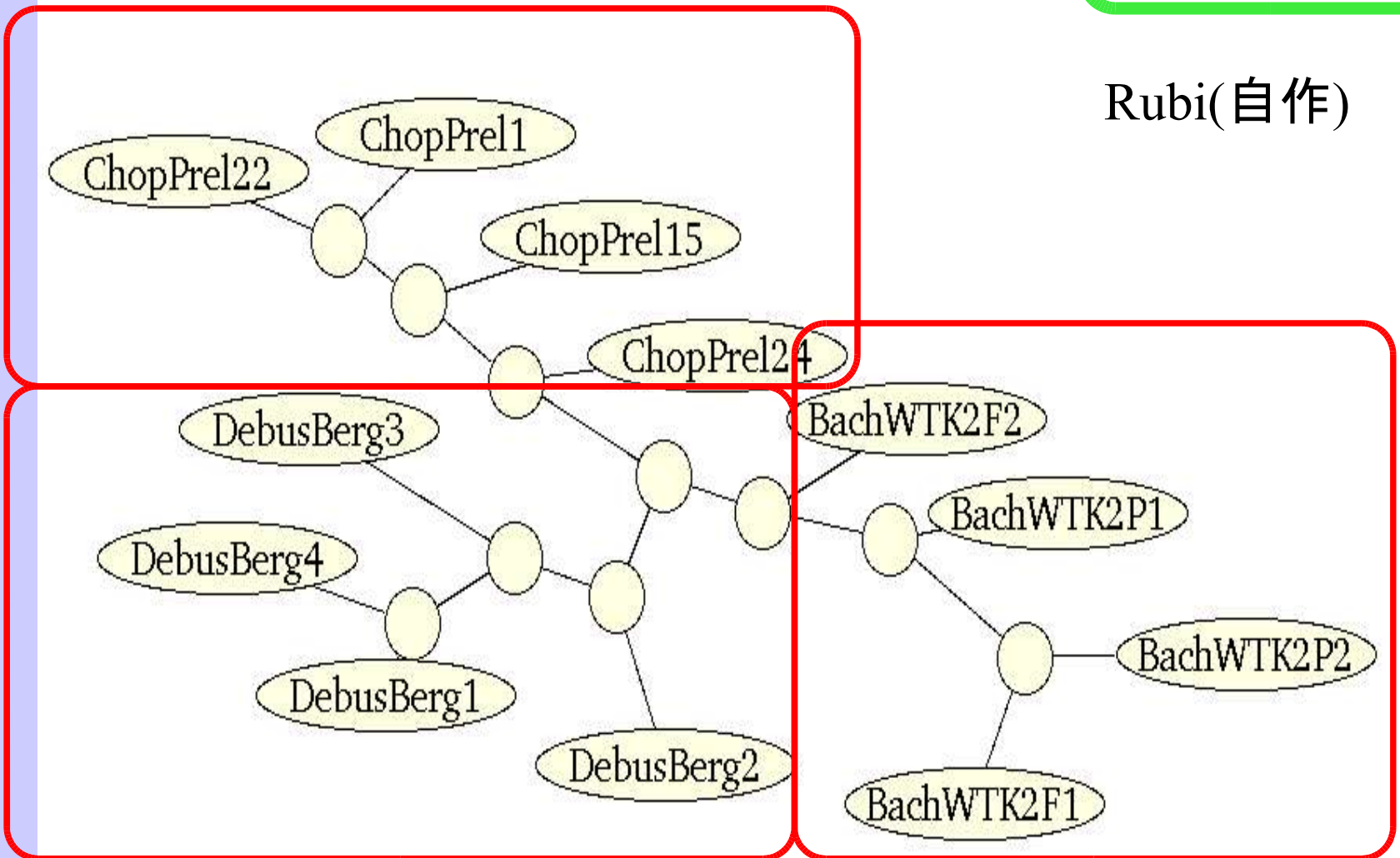


実験結果1

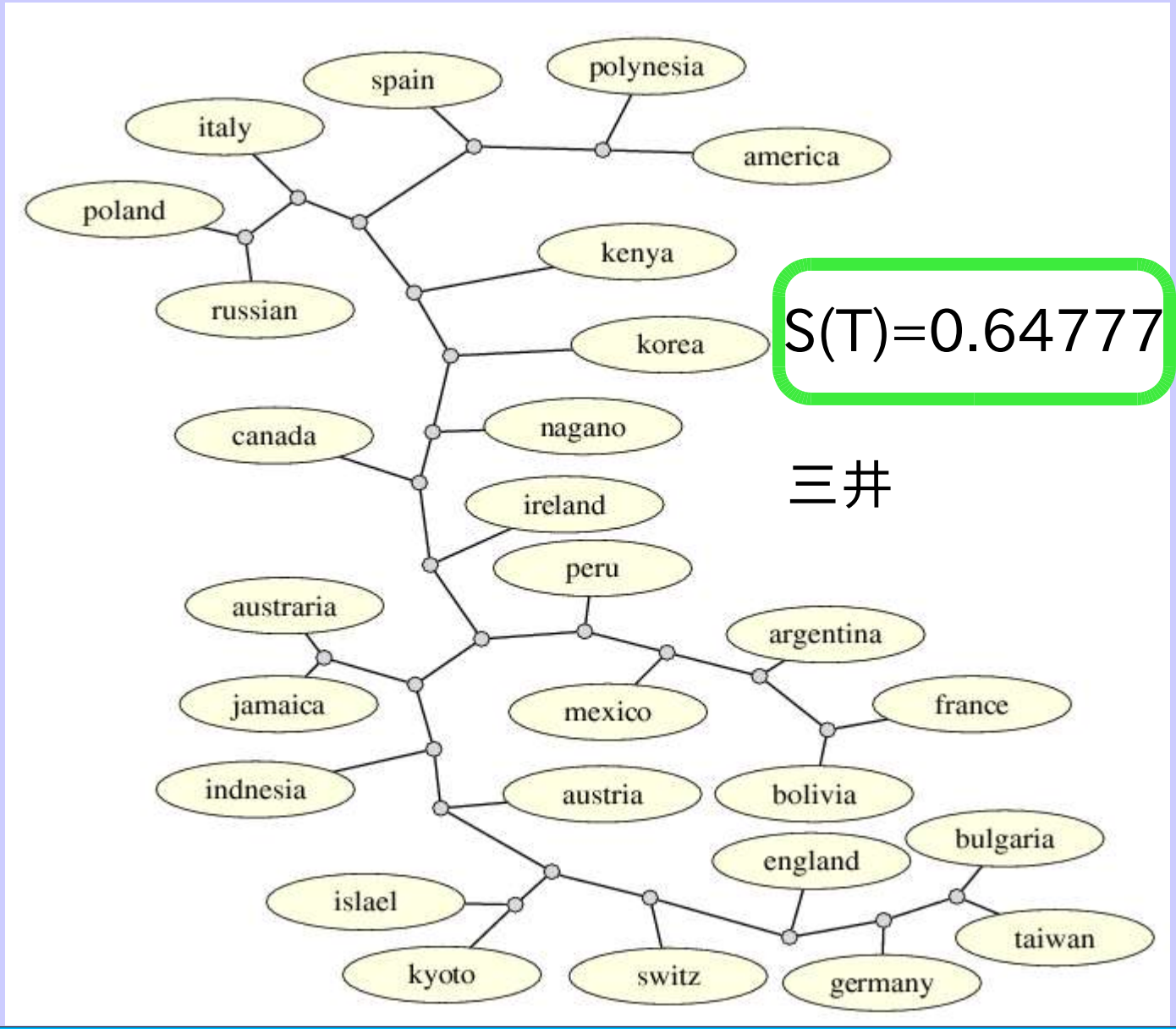
Bach,Debussy,Chopinよりそれぞれ4曲

S(T)=0.9415

Rubi(自作)



実験結果2



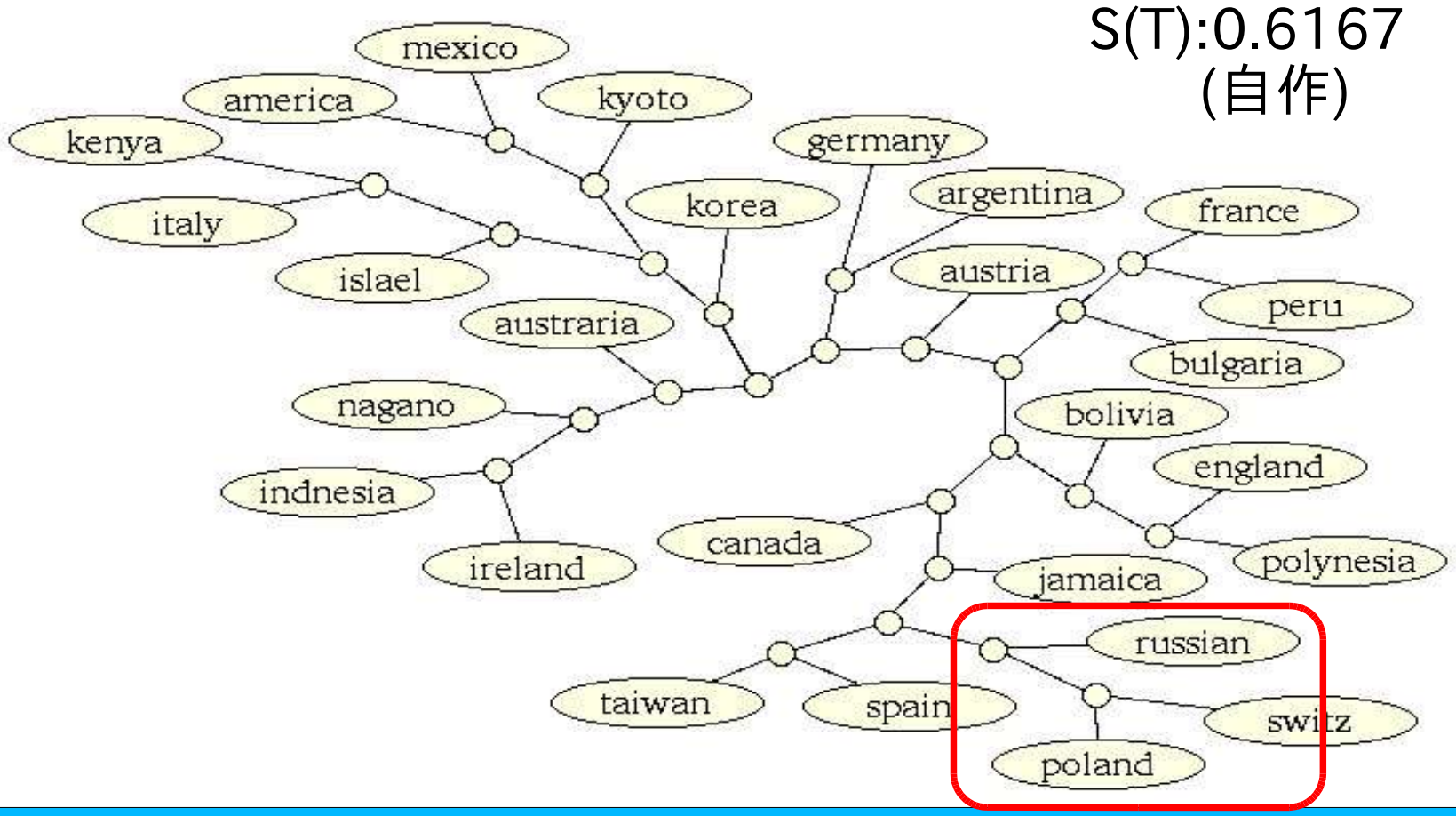
S(T)=0.64777

三井

世界27カ国の民謡, 日本は京都と長野の民謡を使用. 各国一曲,日本のみ2曲

実験結果2

際だった特徴が出ていない



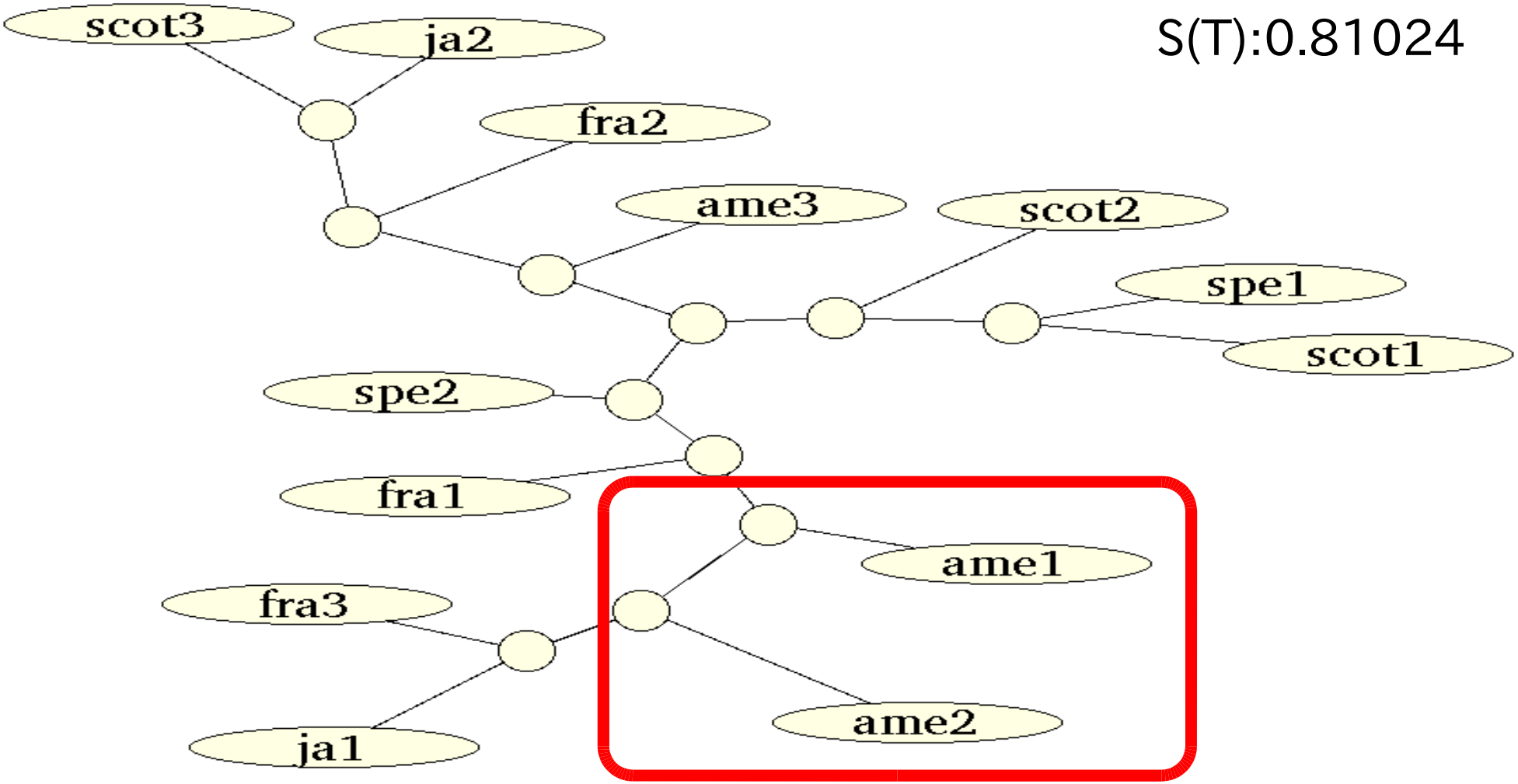
世界27カ国の民謡, 日本は京都と長野の民謡を使用. 各国一曲,日本のみ2曲

scotland 3曲　japan　2曲
america 2曲　france 3曲
spain　2曲

S(T):0.81024

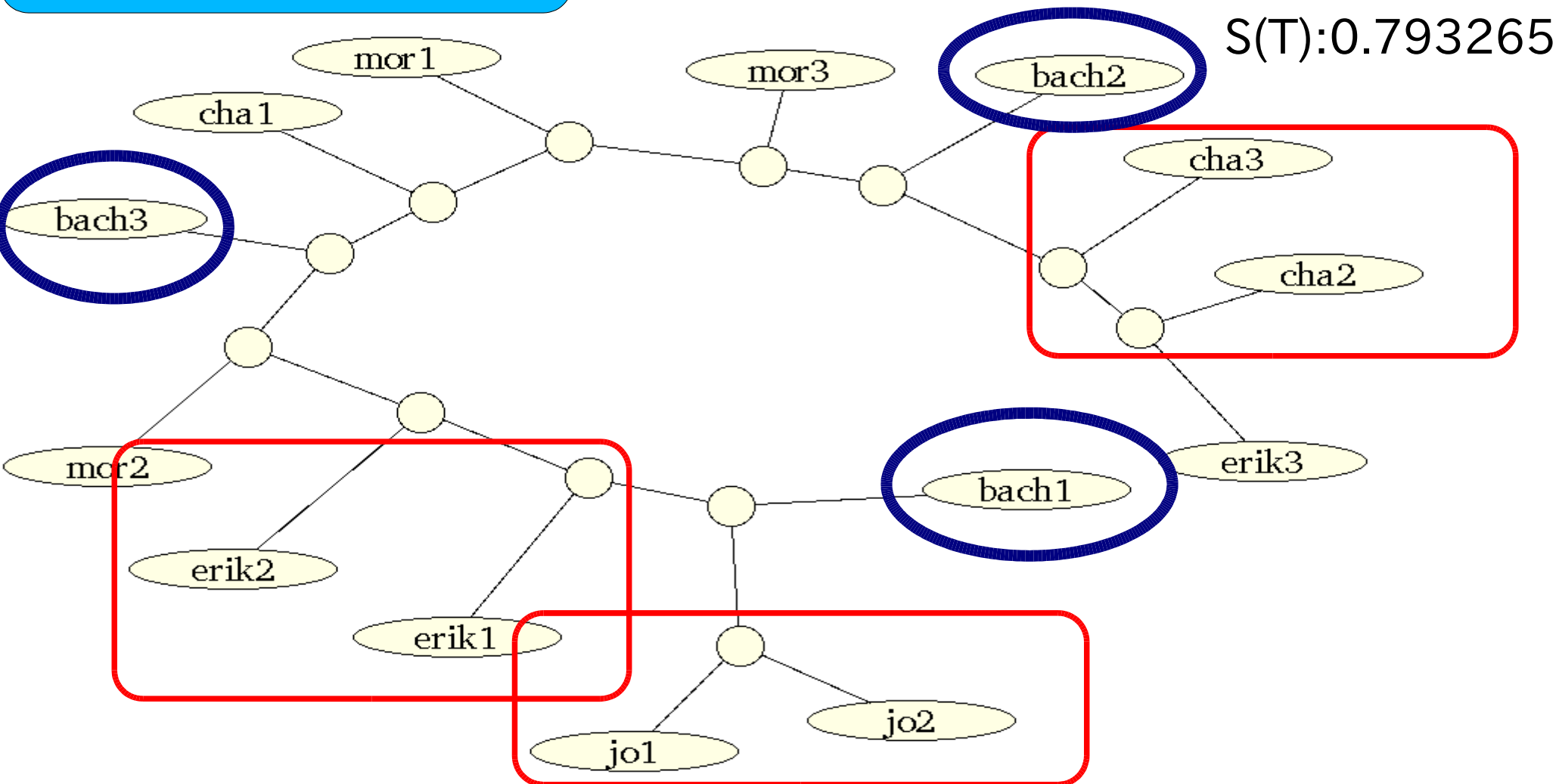
S(T)が高いのに,国で
まとまっていない

S(T):0.81024



チャイコフスキー3曲
Erik satie 3曲
Bach 3曲
Mozart 3曲

S(T):0.793265



目次

- 研究の概要
- 実験方法
- 実験結果
- 考察
- 今後の課題

今後の課題

分類対象や手法に対して、さらに追実験を試みる。

MIDIファイル以外の音楽ファイルで、今回の実験を試してみる。

新しいtranceferの方法を考えてみる。

終了です。