Group theoretical methods in Machine Learning

Risi Kondor
Columbia University

Tutorial at ICML 2007

Part 2

The symmetric group

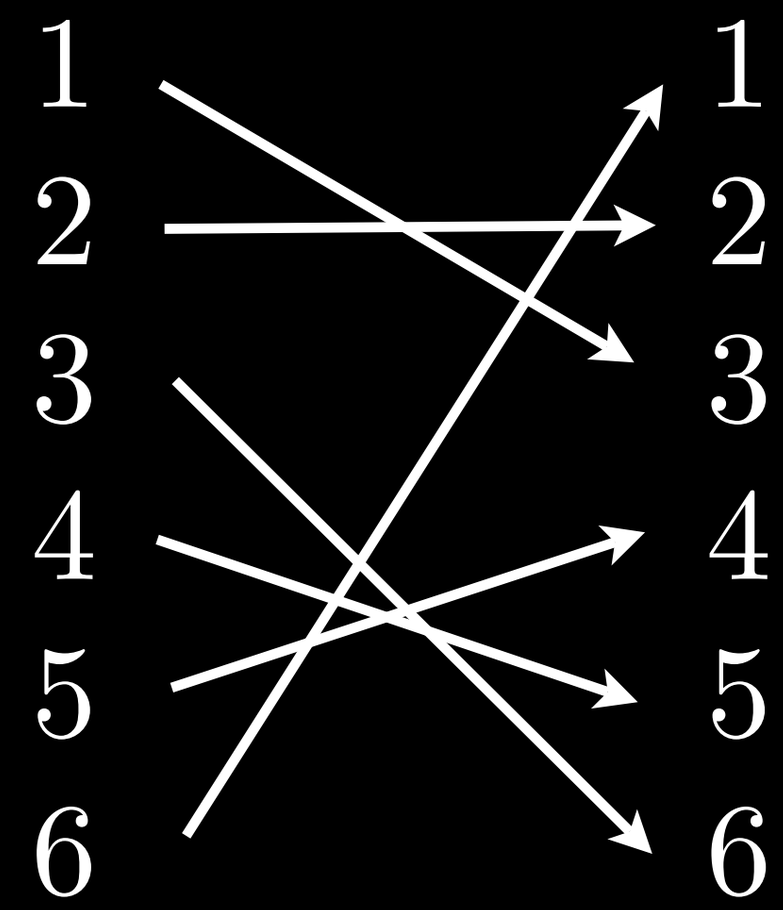
$\mathbb{S}_n$ is the group of bijections

$$\sigma\colon\ \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$$

under composition of maps.

Clearly, $|\,\mathbb{S}_n\,| = n!$.

$$\sigma(1) = 3$$
$$\sigma(2) = 2$$
$$\sigma(3) = 6$$
$$\sigma(4) = 5$$
$$\sigma(5) = 4$$
$$\sigma(6) = 1$$



$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 2 & 6 & 5 & 4 & 1 \end{pmatrix}$$

Cycle notation

Example:

$$\sigma = (163)(45)(2)$$

Cycle type:

$$\sigma = (3, 2, 1)$$

Generators

Transpositions (i, j) generate the whole group.

In fact, adjacent transpositions are sufficient,
since (assuming $i < j$)

$$(i,j) = (i,i+1)\dots(j-2,-1)(j-1,j)\dots(i+1,i+2)(i,i+1)$$

Subgroups

Cayley's theorem:

Any finite group G is a subgroup of $\mathbb{S}_{|G|}$.

Subgroups

$\mathbb{S}_k < \mathbb{S}_n$ permutes $\{1, 2, \ldots, k\}$

$\mathbb{S}_\lambda = \mathbb{S}_{\lambda_1} \times \mathbb{S}_{\lambda_2} \times \ldots \times \mathbb{S}_{\lambda_k} < \mathbb{S}_n$ permutes

$\{1, 2, \ldots, \lambda_1\}, \{\lambda_1 + 1, \ldots, \lambda_1 + \lambda_2\}, \ldots, \{n - \lambda_k, \ldots, n\}$

Normal subgroups

$$A_n = \{\, \sigma \in \mathbb{S}_n \mid \operatorname{sgn}(\sigma) = 1 \,\}$$

For $n \geq 5$ this is the only one!

Representations

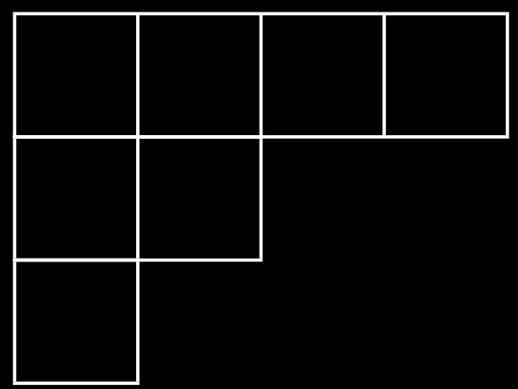
1. The trivial representation $\rho_{\mathrm{triv}}(\sigma) = (1)$

2. The alternating representation $\rho_{\mathrm{triv}}(\sigma) = \mathrm{sgn}(\sigma)$

3. The defining representation $[\rho_{\mathrm{def}}(\sigma)]_{i,j} = \delta_{\sigma(i),j}$

reducible!!!

Young diagrams



$\longleftrightarrow$ integer partitions $\lambda \vdash n$

$\longleftrightarrow$ irreducible representations $\rho \in \mathcal{R}$

Young Tableaux

1	3	6	7
2	5		
4			

standard tableau if numbers increase left to right and top to bottom

standard tableaux of shape λ $\longleftrightarrow$ dimensions of ρ_λ

Young's Orthogonal Representation (YOR)

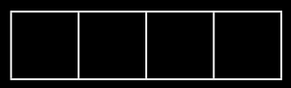
$$[\rho_\lambda(\tau_k)]_{t,t} = 1/d_t(k,k+1)$$

$$[\rho_\lambda(\tau_k)]_{\tau_k(t),t} = \sqrt{1-1/d_t(k,k+1)^2}$$

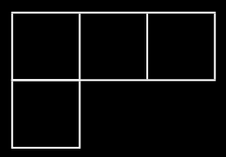
if $\tau_k(t)$ standard

All $\rho_\lambda(\sigma)$ are real!

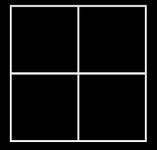
$\mathbb{S}_4$



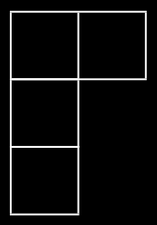
$d = 1$ $\rho_{(4)}(\sigma) = (1)$



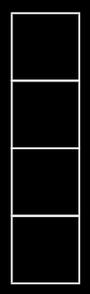
$d = 3$



$d = 2$



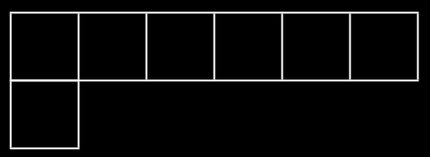
$d = 3$



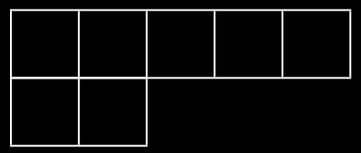
$d = 1$ $\rho_{(1,1,1,1)}(\sigma) = (\mathrm{sgn}(\sigma))$

$\mathbb{S}_n$

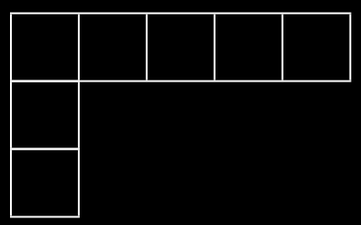
$d = 1$



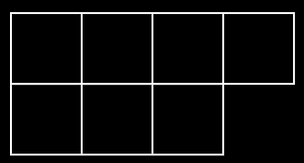
$d = n - 1$



$d = n(n-3)/2$



$d = (n-1)(n-2)/2$



$d = n(n-1)(n-5)/6$

Hook rule

$$d_\lambda = \frac{n!}{\prod_{i=1}^{n} l_i}$$

$$l_i = \text{length of hook } i$$

Restricted representations

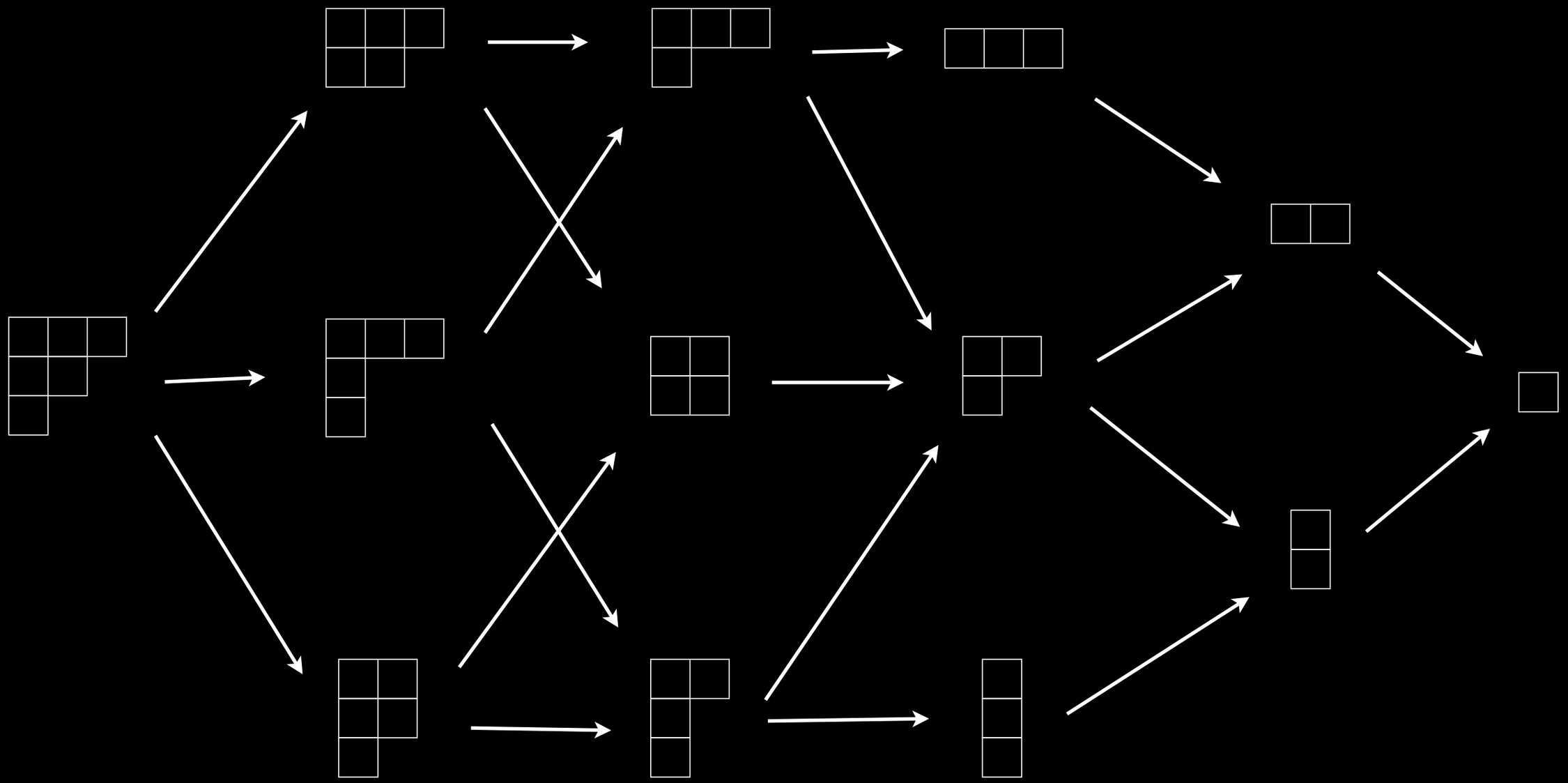
If ρ is a representation of $\mathbb{S}_n$
$\rho\downarrow_{\mathbb{S}_k}$ is a representation of $\mathbb{S}_k$ given by

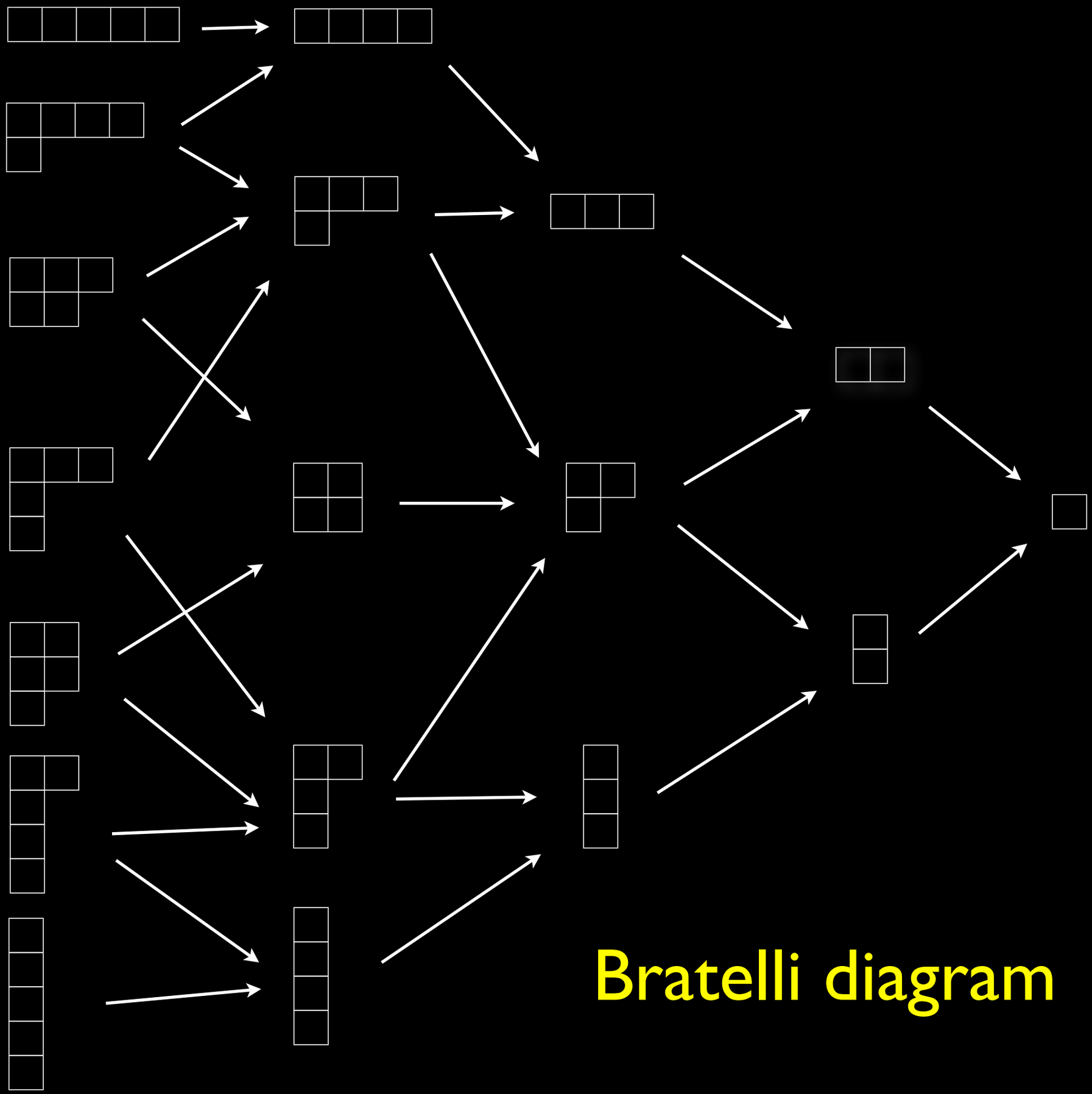
$$\rho\downarrow_{\mathbb{S}_k}(\sigma) = \rho(\sigma) \qquad\qquad \sigma \in \mathbb{S}_k.$$

Young's Rule

In YOR

$$\rho_\lambda \downarrow_{\mathbb{S}_{n-1}} (\sigma) = \bigoplus_{\substack{\lambda^- \vdash n-1 \\ \lambda^- < \lambda}} \rho_{\lambda^-}(\sigma)$$





Bratelli diagram

Clausen's FFT for $\mathbb{S}_n$

Define the contiguous cycle

$$[\![i, j]\!] = (i, i+1, i+2, \ldots, j)$$

The cosets $[\![1, n]\!]\mathbb{S}_{n-1},\ [\![2, n]\!]\mathbb{S}_{n-1}, \ldots, [\![n, n]\!]\mathbb{S}_{n-1}$ form a partition of $\mathbb{S}_n$

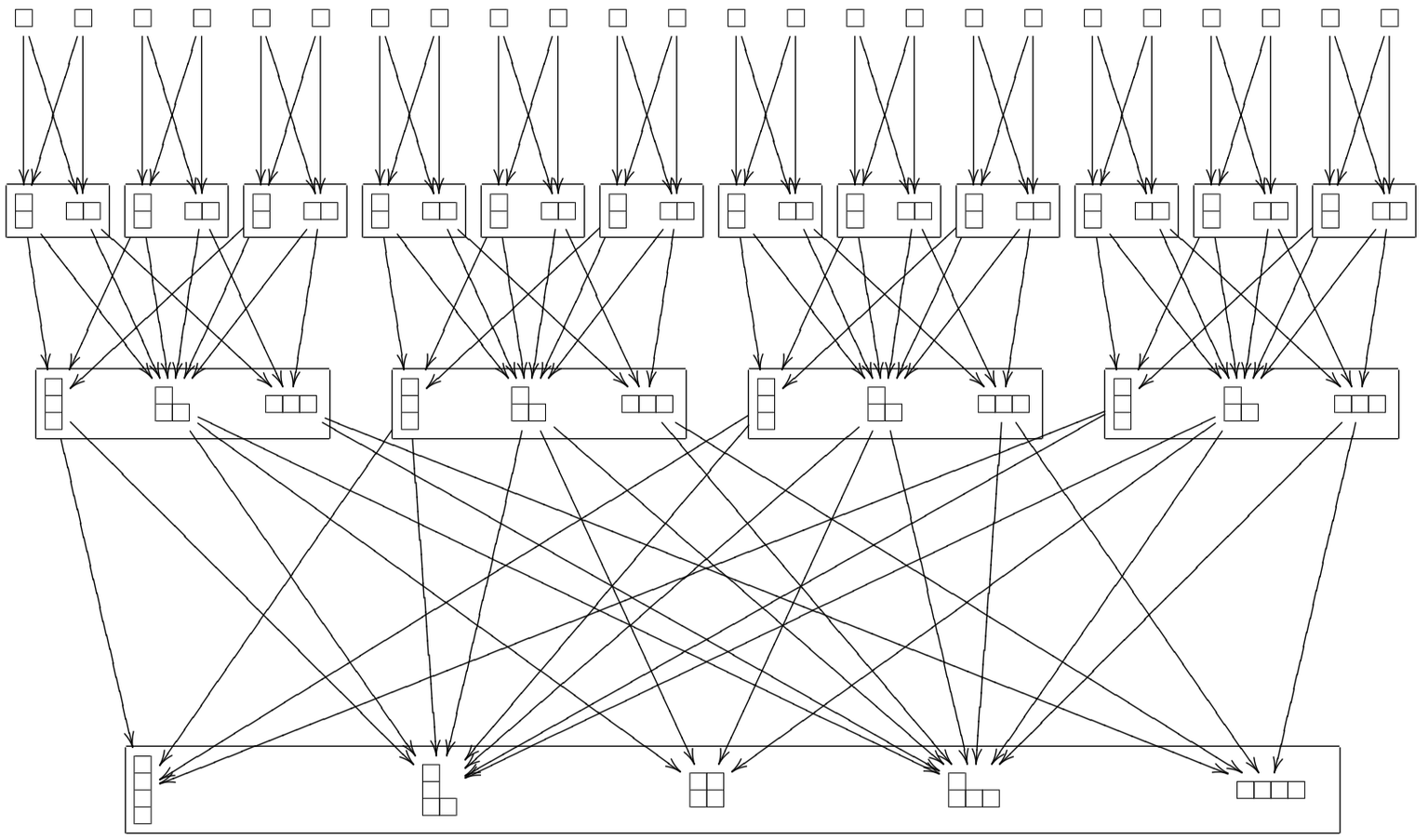
Idea: decompose FT over $\mathbb{S}_n$ into n smaller FTs over the $[\![i, n]\!]\mathbb{S}_{n-1}$ cosets.

$$\widehat{f}(\lambda) = \sum_{\sigma \in \mathbb{S}_n} \rho_\lambda(\sigma)\, f(\sigma)$$

$$= \sum_{i=1}^{n} \sum_{\sigma' \in \mathbb{S}_{n-1}} \rho_\lambda([\![i,n]\!]\,\sigma')\; f([\![i,n]\!]\,\sigma')$$

$$= \sum_{i=1}^{n} \rho_\lambda([\![i,n]\!]) \sum_{\sigma' \in \mathbb{S}_{n-1}} \rho_\lambda(\sigma')\; f_{[\![i,n]\!]\mathbb{S}_{n-1}}(\sigma')$$

$$= \sum_{i=1}^{n} \rho_\lambda([\![i,n]\!]) \bigoplus_{\substack{\lambda^- \vdash n-1 \\ \lambda^- < \lambda}} \widehat{f}_{[\![i,n]\!]\mathbb{S}_{n-1}}(\lambda^-)$$



Op count

- $[\![i,k]\!]$ is a product of $k-i$ adjacent transpositions
- Computing $\rho_\lambda([\![i,j]\!]) \cdot M$ takes $2\,d_\lambda^2$ time
- For $\lambda \vdash k \quad d_\lambda^2 = k!$
- Layer k has $n!/k!$ Fourier transforms.

$$\sum_{k=1}^{n} \sum_{i=1}^{k} 2\,(k-i)\,n! \;= n!\,\frac{(n+1)\,n\,(n-1)}{3}$$

$\mathbb{S}_n\mathtt{ob}$

SnOB, FFT for the symmetric group

http://www1.cs.columbia.edu/~risi/SnOB/

Google

$S_n$ob

A C++ library for fast Fourier transforms on the symmetric group.

author: Risi Kondor, Columbia University (risi@cs.columbia.edu)

Development version as of August 23, 2006 (unstable!):

Documentation: [ps][pdf]
C++ source code: [directory]
BiBTeX entry: [bib]
Entire package: [tar.gz]

References:

1. Michael Clausen: **Fast generalized Fourier transforms.** Theoretical Computer Science **67(1)**: 55-63, 1989.
2. David K. Maslen and Daniel N. Rockmore: **Generalized FFTs --- a survey of some recent results.** Proceedings of the DIMACS Workshop on Groups and Computation, 1997. [ps]
3. K.-J. Kueh, T. Olson, D. Rockmore and K.-S. Tan: **Nonlinear approximation theory on finite**

http://www.cs.columbia.edu/~risi/SnOB

SnFourierTransform.hpp

SnFourierTransform.hpp:1 | <No selected symbol>

```
#include <vector>

#include "base.h"
#include "Matrix.hpp"
#include <sstream>
#include "Sn.hpp"
#include "SnFunction.hpp"
#include "StandardTableau.hpp"

using namespace std;

class Sn::FourierTransform: FiniteGroup::FourierTransform{

public:

  friend class Sn::Function;
  friend class Sn::Ftree;

  FourierTransform(const Sn& _group);
  FourierTransform(const Sn& _group, int dummy):group(&_group),n(_group.n){};
  FourierTransform(const Sn& _group, const vector<Matrix<FIELD >*> matrices);
  FourierTransform(const Function& f);
  ~FourierTransform();

  Function* iFFT() const;

  FIELD operator()(const StandardTableau& t1, const StandardTableau& t2) const;

  double norm2() const {double result; for(int i=0; i<matrix.size(); i++) result+=1; return result;}

  string str() const;

  vector<Matrix<FIELD >*> matrix;


private:

  void fft(const Sn::Function& f, const int offset);
  void ifft(Sn::Function* target, const int _offset) const;

  const int n;
  const Sn* group;

};
```

Sn::Irreducible

Represents an irreducible representation ρ_λ of S_n.

Parent class: `FiniteGroup::Irreducible`

CONSTRUCTORS

`Irreducible(Sn* G, Partition& lambda)`
Construct the irreducible representation of the symmetric group `G` corresponding to the partition `lambda`.

MEMBER FUNCTIONS

`Matrix<FIELD >* rho(const Sn::Element& sigma)`
Returns $\rho(\sigma)$, the representation matrix of permutation `sigma` in Young's orthogonal representation.

`FIELD character(const Partition& mu)`
Returns $\chi(\mu)$, the character of this representation at permutations of cycle type μ.

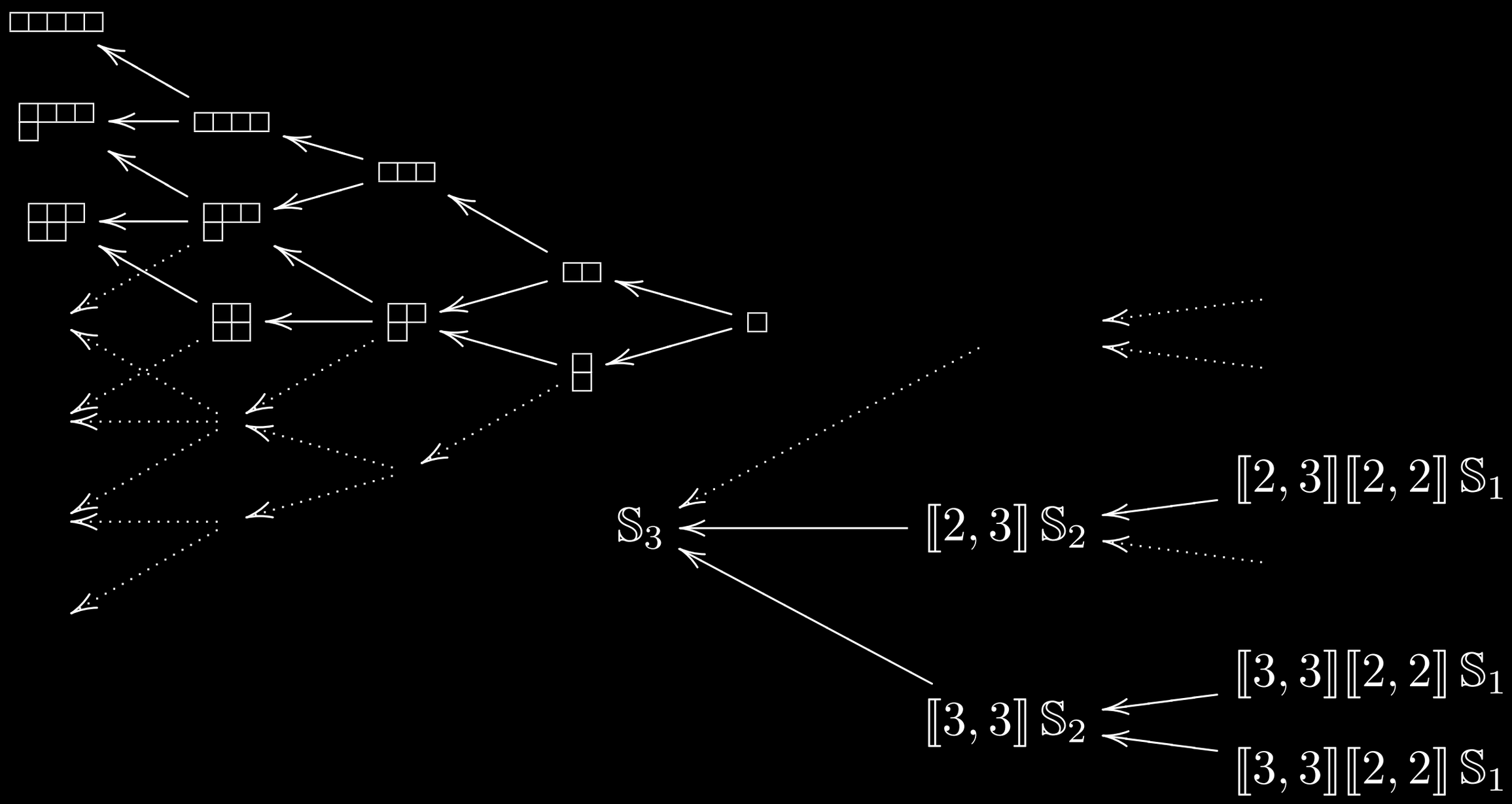
`void computeTableaux()`
Compute the standard tableaux of this irreducible if they have not already been computed. Because this is an expensive operation, it is postponed until some function is called (such as `rho` or `character`) which requires the tableaux of this particular irreducible. `computeTableaux()` is called automatically by these functions, and once the tableaux have been computed they are stored for the lifetime of the `Irreducible`.

`StandardTableau* tableau(const int t)`
Return a new standard tableaux of index `t`. This works even if `tableauV` has not been computed.

`void computeYOR()`
Compute and store the coefficients (2.5) and (2.6) in Young's orthogonal representation for all adjacent transpositions τ_k and all tableau t of shape λ. Because this is an expensive operation, these coefficients are not normally computed until they are demanded by functions such as `rho` or `character`. `computeYOR()` is called automatically by these functions, and once the tableaux have been computed they are stored for the lifetime of the `Irreducible`. `computeYOR()` also requires the tableaux, so it calls `computeTableaux()` if those have not been computed yet.

`void applyCycle(const int j, Matrix<FIELD>& M[, int m])`
`void applyCycle(const int j, Matrix<FIELD>& M[, int m])`

Sparse transforms



Twisted transforms

Restrict to two sided cosets $\sigma_L \mathbb{S}_k \sigma_R$

The isotypal components of $\mathbb{S}_n$

Persi Diaconis

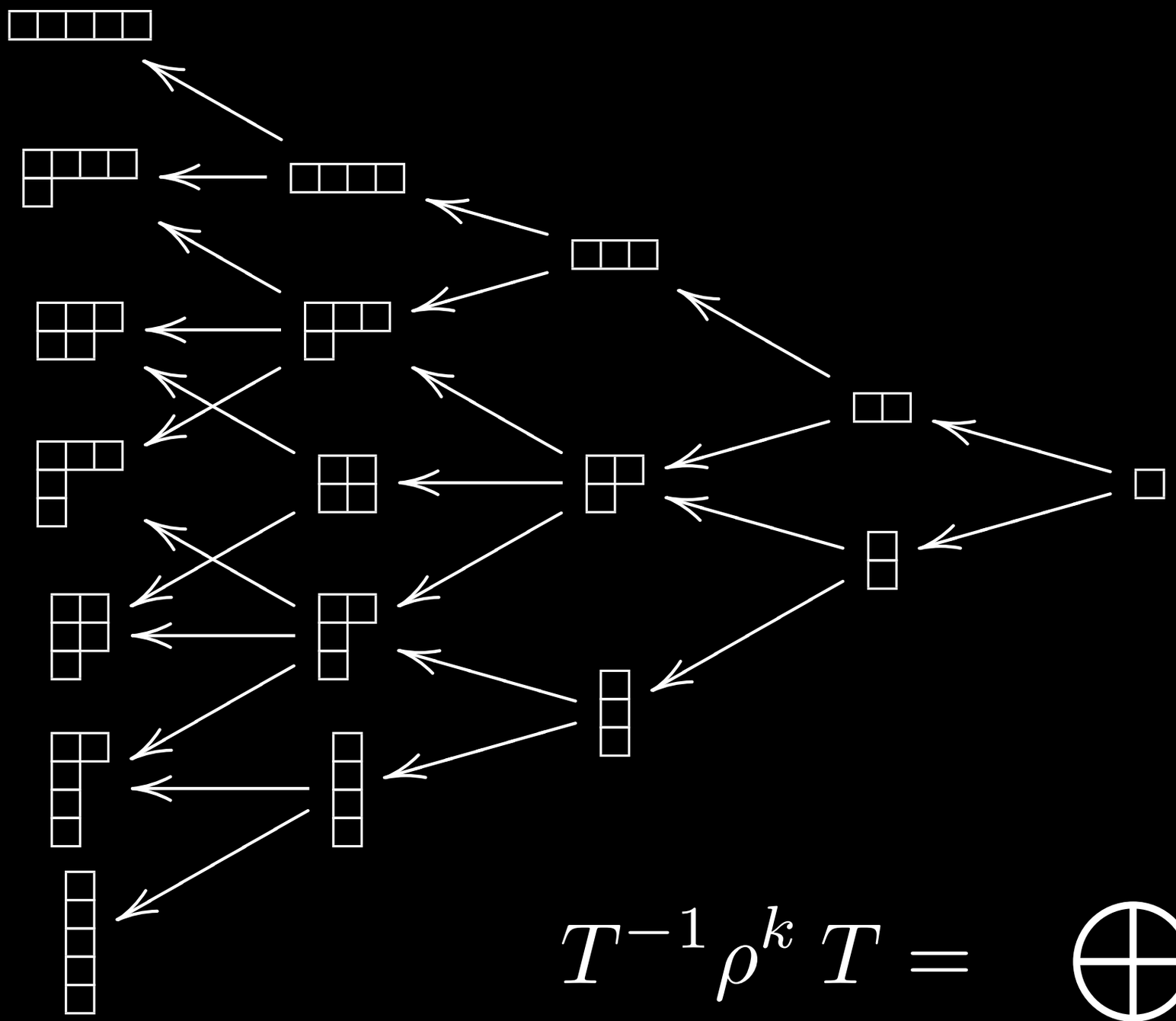
"Group Representations in Probability and Statistics"
IMS, 1988

Interpretation of σ : rank of j is $i = \sigma^{-1}(j)$

Partial rankings: $\sigma(\mathbb{S}_{\lambda_1} \times \ldots \times \mathbb{S}_{\lambda_k})$ cosets

Permutation representations:

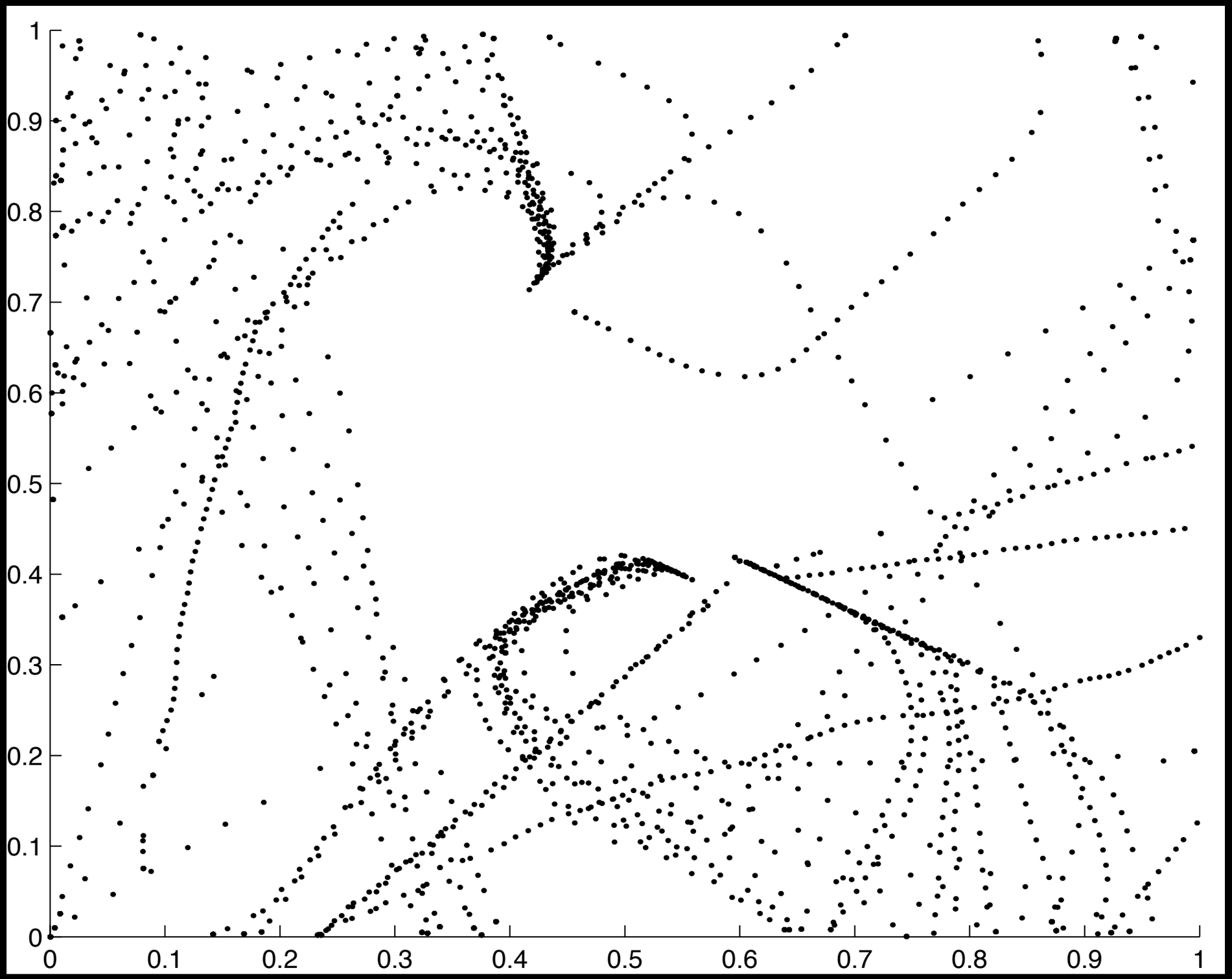
$$\left[\rho^k(\sigma)\right]_{(j_1,\ldots,j_k),(i_1,\ldots,i_k)} = \begin{cases} 1 & \text{if } \sigma(i_a) = j_a, \quad a = 1, 2, \ldots, k \\ 0 & \text{otherwise} \end{cases}$$



$$T^{-1}\rho^k\,T = \bigoplus_{\substack{\lambda\vdash n-k\\ \lambda\geq(n-k)}} \rho_\lambda$$

Application: multi-object tracking

R. Kondor, A. Howard, T. Jebara (AISTATS 2007)



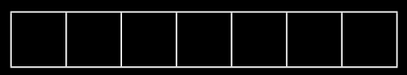
http://www4.passur.com/jfk.html

Band limited approximation to $p(\sigma)$

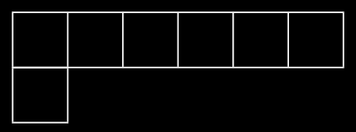
n	$n!$
2	2
3	6
4	24
5	120
6	720
7	5040
8	40320
9	362880
10	3628800
11	39916800
12	$4.8 \cdot 10^8$
$\vdots$	$\vdots$
15	$1.3 \cdot 10^{12}$
$\vdots$	$\vdots$
20	$2.4 \cdot 10^{18}$

$$\Delta_{\sigma_1,\sigma_2} = \begin{cases} 1 & \text{if } \sigma_1 = (i,j) \cdot \sigma_2 \\ -n(n-1)/2 & \text{if } \sigma_1 = \sigma_2, \\ 0 & \text{otherwise.} \end{cases}$$

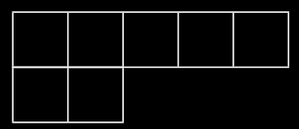
$$\alpha_\lambda = -\binom{n}{2}\left(1 - \frac{\mathrm{tr}\left[\rho_\lambda((1,2)\right]}{d_\lambda}\right)$$



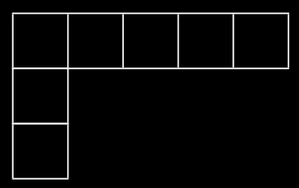
$d = 1$



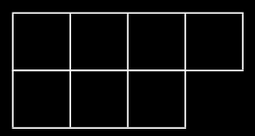
$d = n - 1$



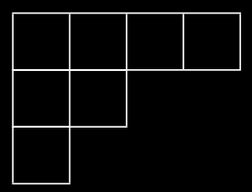
$d = n(n-3)/2$



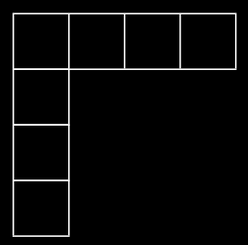
$d = (n-1)(n-2)/2$



$d = n(n-1)(n-5)/6$



$d = n(n-2)(n-4)/3$



$d = (n-1)(n-2)(n-3)/6$

1. Noise: diffusion over transpositions

2. Observations: $O_{i \to j}$ with probability π

3. Inference: $p[\sigma(i) = j]$

1. Noise model

$$\boldsymbol{p}_{t'} = \left(I + \frac{\beta(t'-t)}{m} \Delta \right)^m \boldsymbol{p}_t = e^{\beta(t'-t)\Delta} \qquad m \to \infty$$

$$\widehat{p}_{t'}(\rho_\lambda) = e^{\beta \alpha_\lambda (t'-t)} \, \widehat{p}_t(\rho_\lambda)$$

cost: $O(d_{\max}^2)$

2. Observations

$$p\,(\,O_{i\to j}\,|\,\sigma\,) = \begin{cases} \pi & \text{if } \ \sigma(i) = j, \\ (1-\pi)/(n-1) & \text{if } \ \sigma(i) \neq j. \end{cases}$$

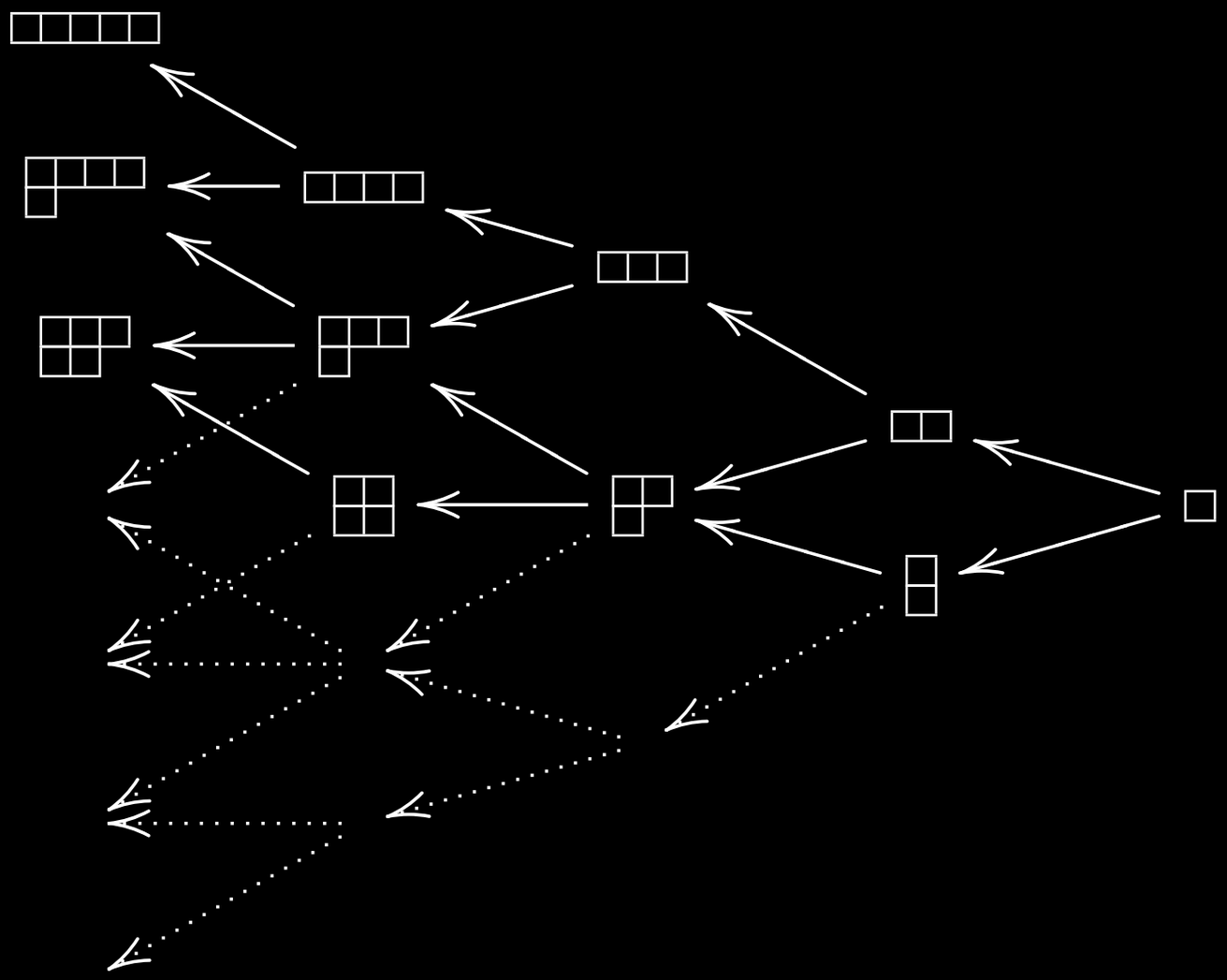
Compute the FT of $\quad f_{i\to j}(\sigma') = f([\![j,n]\!]\,\sigma'\,[\![i,n]\!]^{-1})$

$$\widehat{f}(\lambda) = \sum_{j=1}^{n} \rho([\![j,n]\!]) \left[\bigoplus_{\lambda^-} \widehat{f}_{i\to j}(\rho_{\lambda^-})\right] \rho([\![i,n]\!]^{-1})$$

cost: $\quad O(d_{\max}^2 n)$

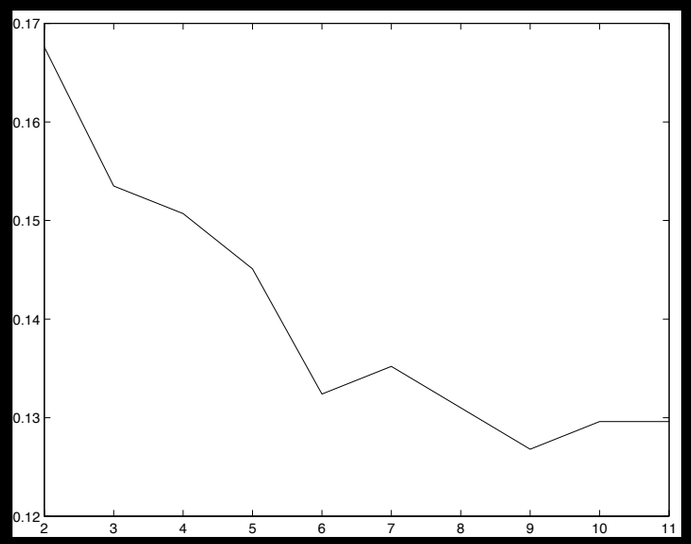
3. Inference

$$p[\sigma(i) = j] = \widehat{p}_{i \to j}((n-1))$$

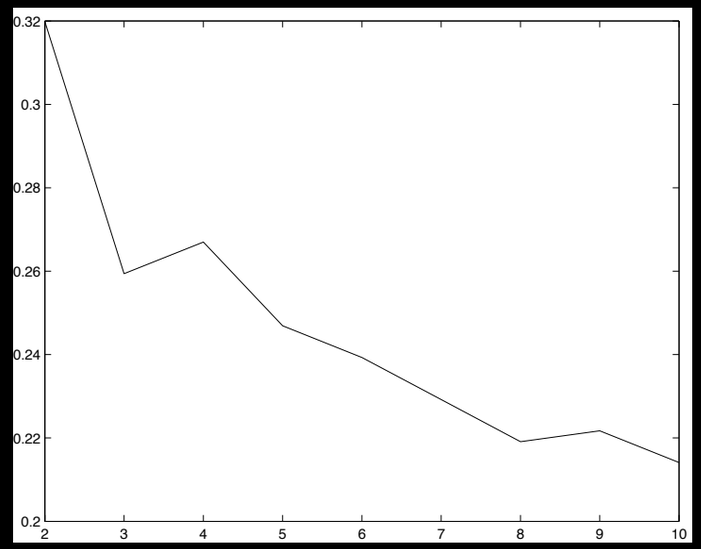


cost: $O(n^3)$

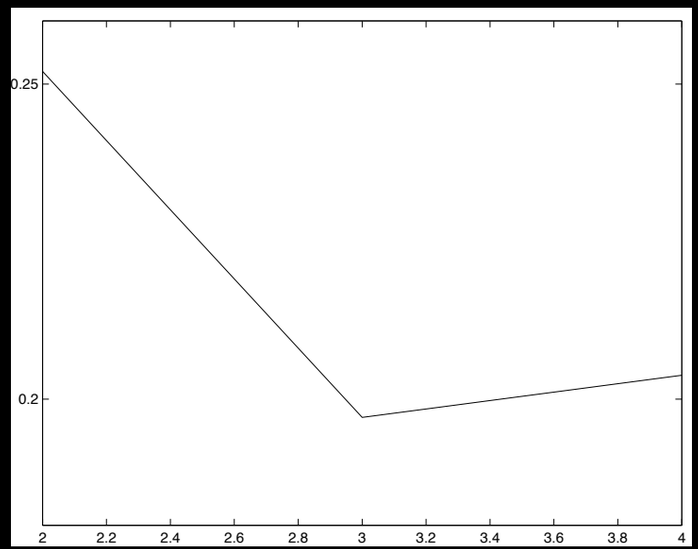
Results



$n = 6$
$n! = 720$



$n = 10$
$n! = 6.6 \cdot 10^6$



$n = 15$
$n! = 1.3 \cdot 10^{12}$

using just the components
$\rho_{(4)}, \rho_{(n-1,1)}, \rho_{(n-2,2)}, \rho_{(n-2,1,1)}$

$n = 15 \qquad t \approx 59\text{ms}$
$n = 30 \qquad t \approx 3\text{s}$