BOOLEAN MATRIX FACTORISATIONS IN DATA MINING (AND ELSEWHERE)

Pauli Miettinen

1 March 2014

max planck institut
informatik

> *In the sleepy days when the provinces of France were still quietly provincial, matrices with Boolean entries were a favored occupation of aging professors at the universities of Bordeaux and Clermont-Ferrand. But one day…*

Gian-Carlo Rota
Foreword to *Boolean matrix theory and applications* by K. H. Kim, 1982

MOTIVATING EXAMPLE

Images by John Tenniel, openclipart.org, and Wikipedia

TABLE OF FEATURES

long-haired	✔	✔	✘
well-known	✔	✔	✔
male	✘	✔	✔

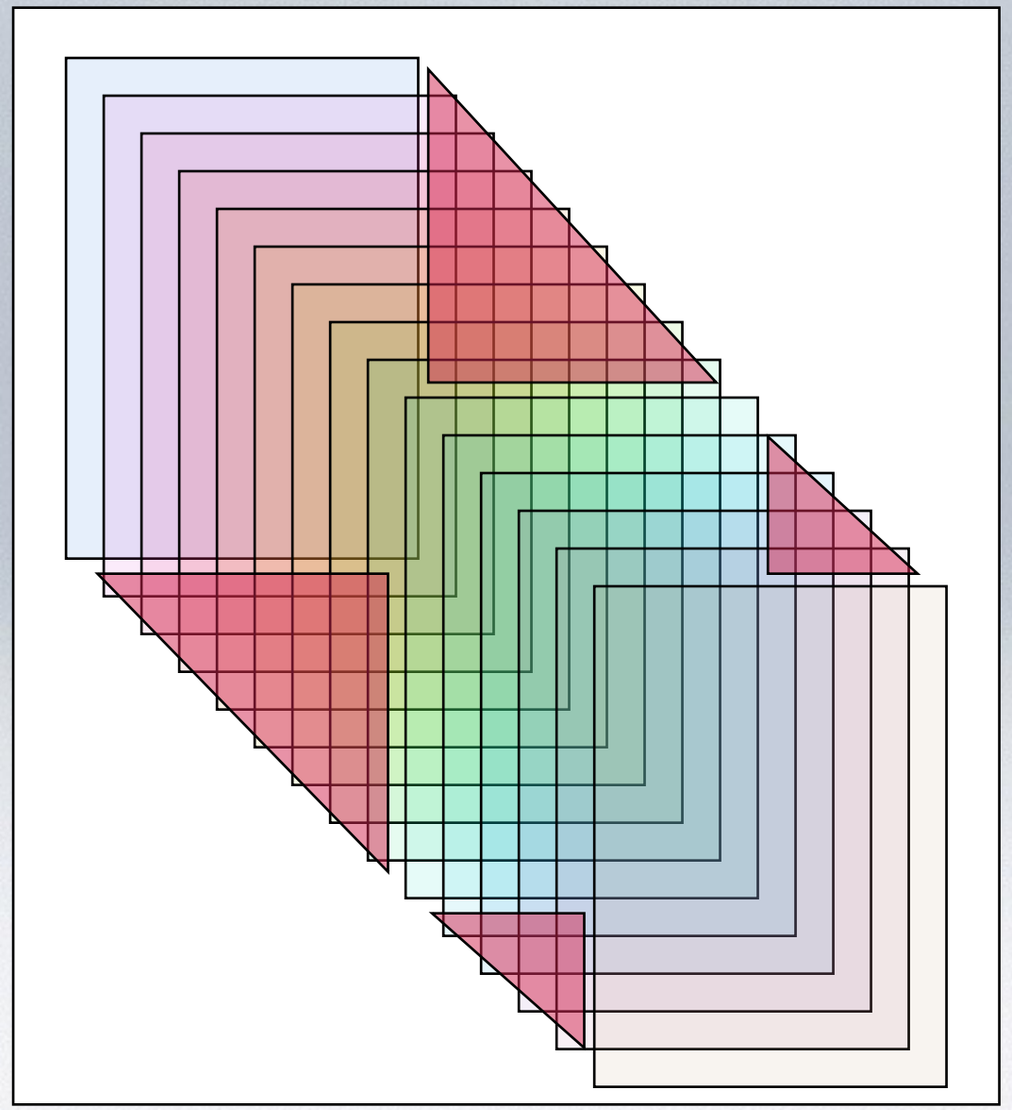
BINARY MATRIX

$$
\begin{array}{r}
\text{long-haired} \\
\text{well-known} \\
\text{male}
\end{array}
\begin{pmatrix}
1 & 1 & 0 \\
1 & 1 & 1 \\
0 & 1 & 1
\end{pmatrix}
$$

FREQUENT ITEMSET MINING

- **Data:** Transactions over items
- **Goal:** Extract all sets of items that appear in many-enough transactions
- **Problem:** Too many frequent itemsets
  - Every subset of a frequent itemset is frequent
- **Solution:** Maximal, closed, and non-derivable itemsets

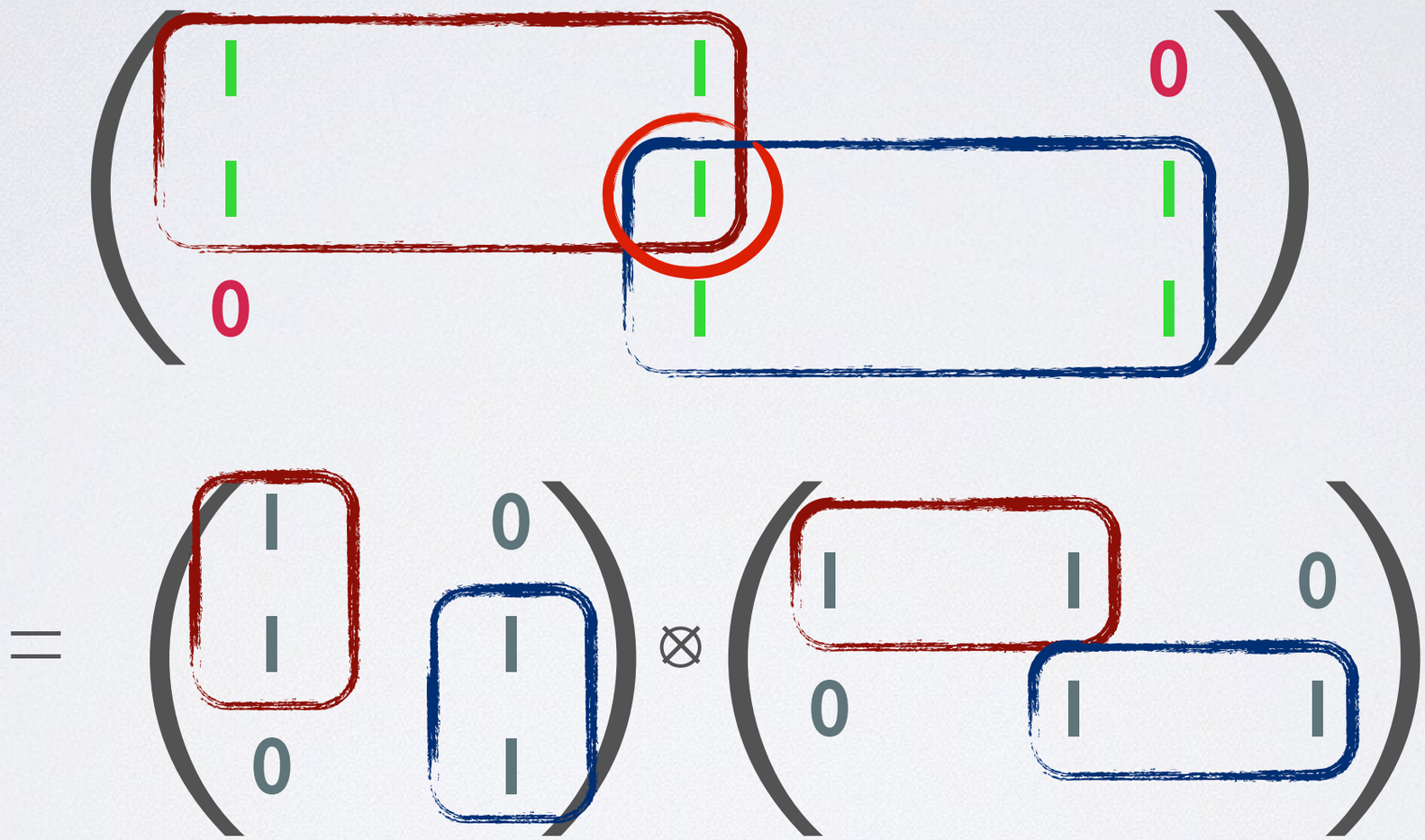
STILL TOO MANY ITEMSETS



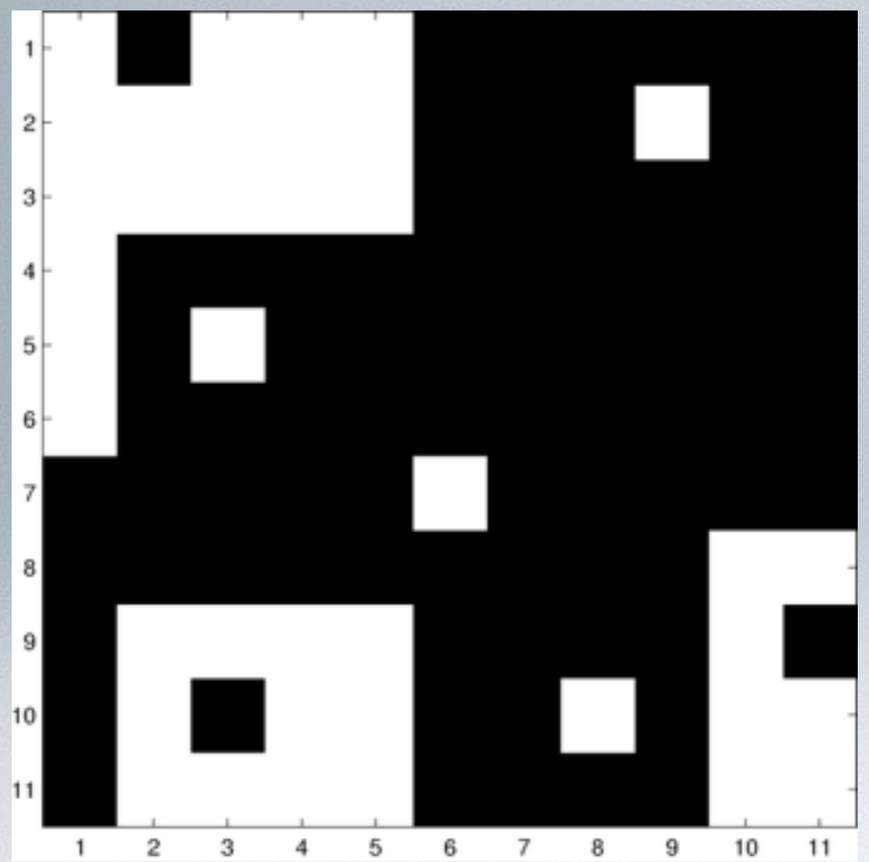
TILING DATABASES

- **Goal:** Find itemsets that *cover* the transaction data
  - Itemset I covers item i in transaction T if $i \in I \subseteq T$
- **Minimum tiling:** Find the smallest number of tiles that cover all items in all transactions
- **Maximum *k*-tiling:** Find k tiles that cover the maximum number of item–transaction pairs
- If you have a set of tiles, these reduce to the Set Cover problem

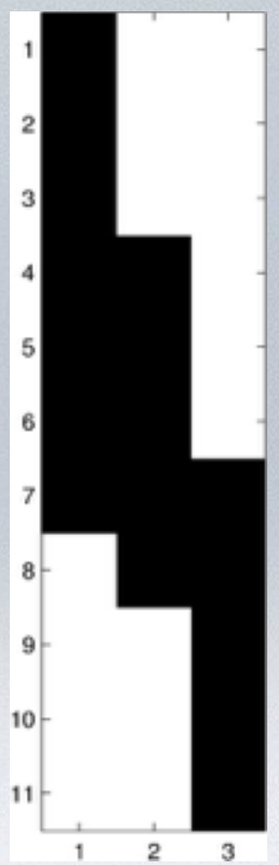
TILING AS A MATRIX FACTORISATION



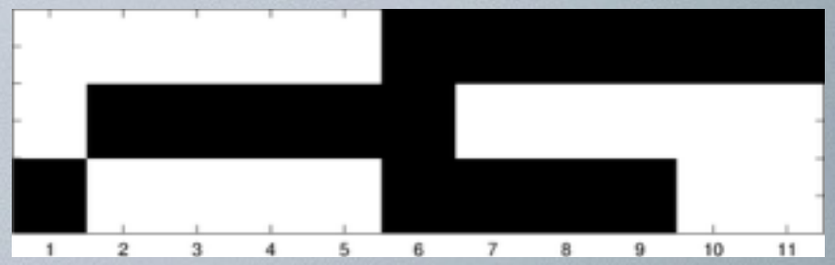
BOOLEAN FACTORISATIONS



≈



∘



max planck institut
informatik

BOOLEAN PRODUCTS AND FACTORISATIONS

- The **Boolean matrix product** of two binary matrices **A** and **B** is their matrix product under Boolean semi-ring

$$(\mathbf{A} \circ \mathbf{B})_{ij} = \bigvee_{i=1}^{k} a_{ik} b_{kj}$$

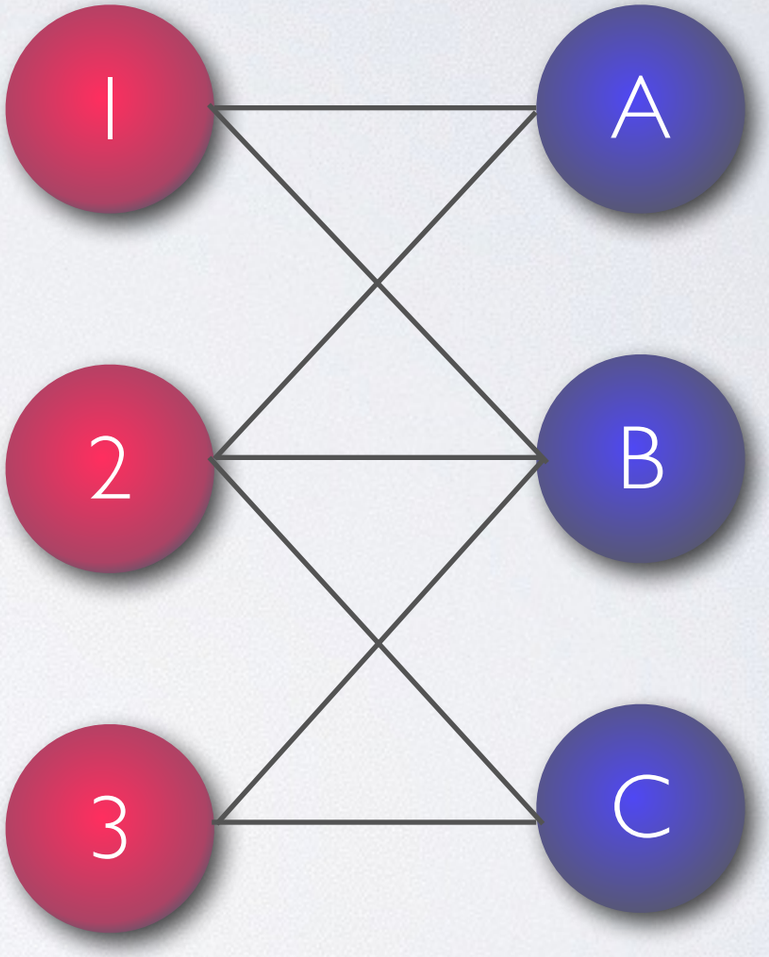
- The **Boolean matrix factorisation** of a binary matrix **A** expresses it as a Boolean product of two binary factor matrices **B** and **C**, that is, **A** = **B**∘**C**

MATRIX RANKS

- The (Schein) **rank** of a matrix **A** is the least number of rank-1 matrices whose sum is **A**
  - $\mathbf{A} = \mathbf{R}_1 + \mathbf{R}_2 + \ldots + \mathbf{R}_k$
  - Matrix is rank-1 if it is an outer product of two vectors
- The **Boolean rank** of binary matrix **A** is the least number of *binary* rank-1 matrices whose element-wise *or* is **A**
  - The least k such that $\mathbf{A} = \mathbf{B} \circ \mathbf{C}$ with **B** having k columns

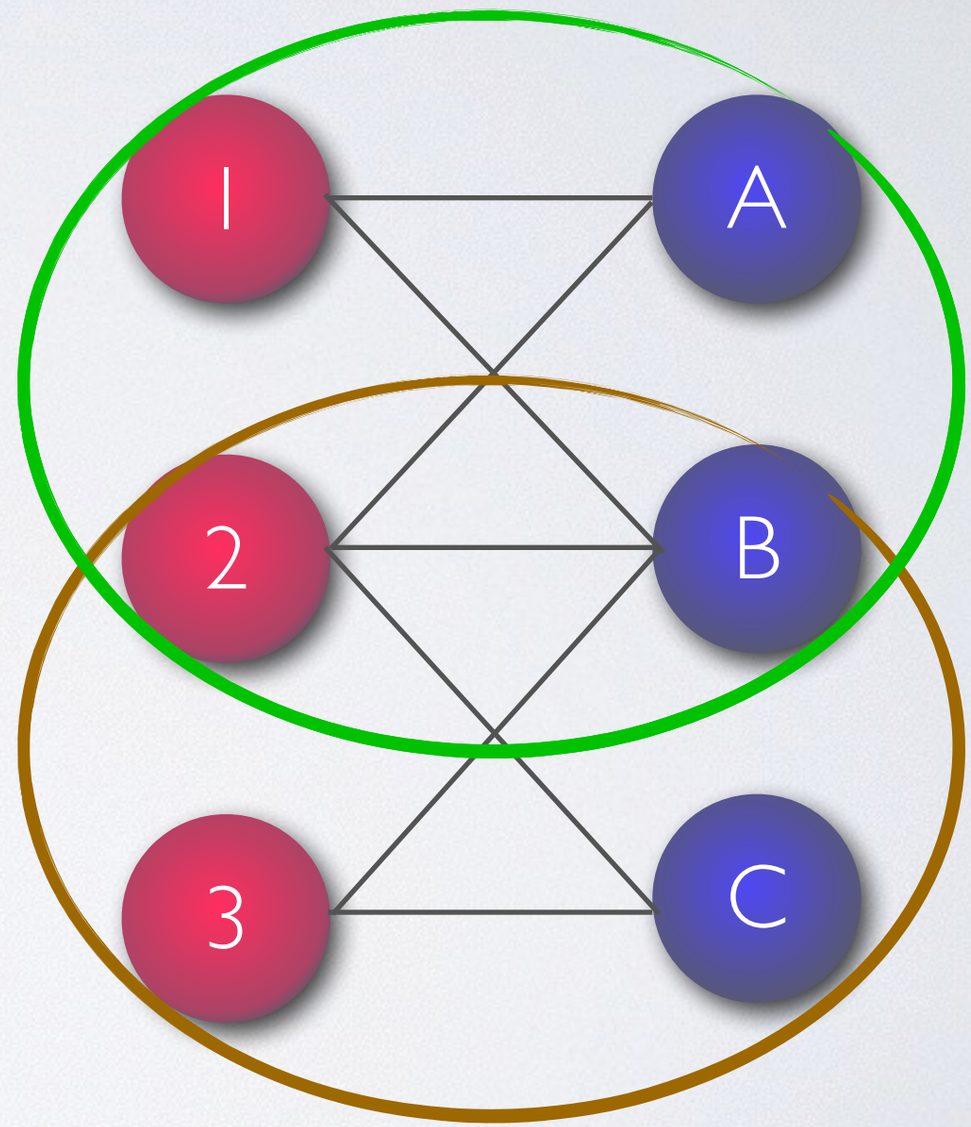
BOOLEAN RANK AND BICLIQUES

- The Boolean rank of a matrix ***A*** is **the least number of complete bipartite subgraphs needed to cover every edge** of the induced bipartite graph $G(\boldsymbol{A})$



BOOLEAN RANK AND BICLIQUES

$$
\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}
\begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} \end{array}
=
\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}
\begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix}
\circ
\begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \end{array}
$$

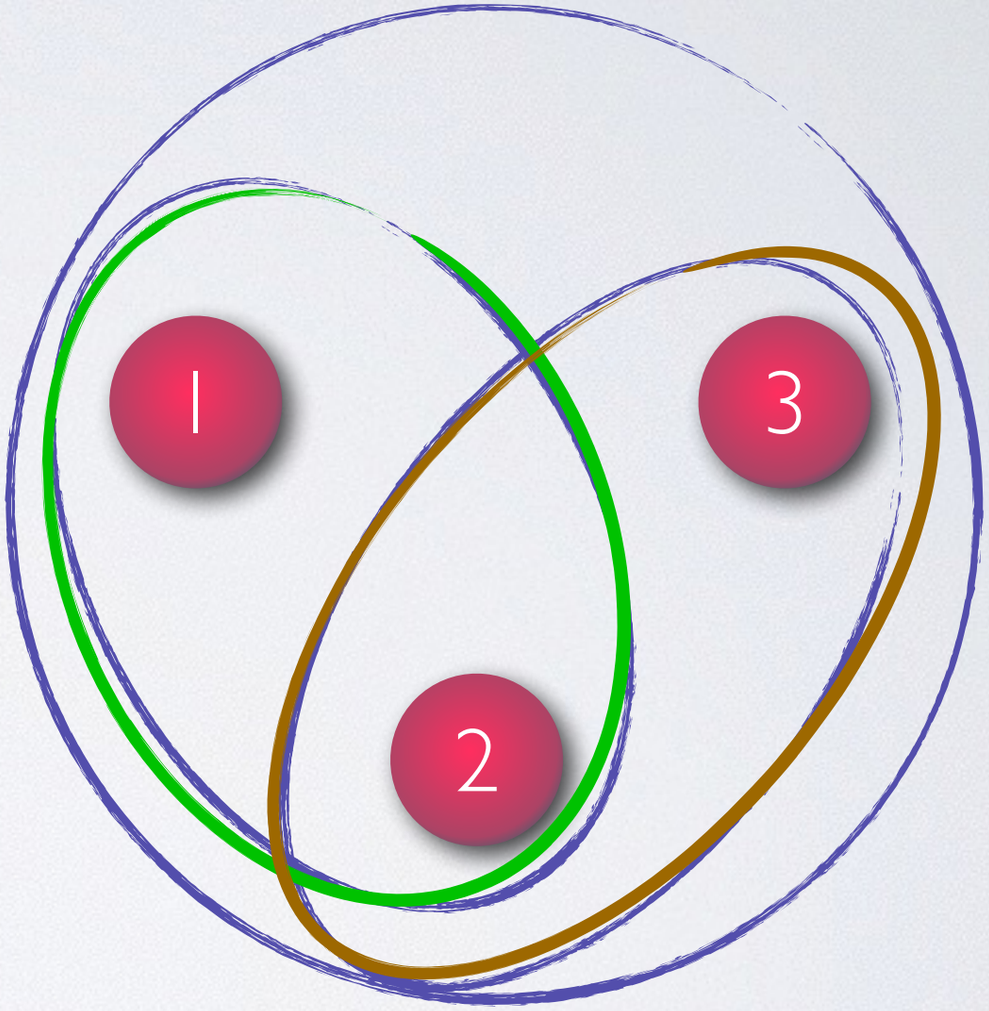


BOOLEAN RANK AND SETS

- The Boolean rank of a matrix ***A*** is **the least number of subsets of** *U*(***A***) **needed to cover every set** of the induced collection *C*(***A***)

- For every *C* in *C*(***A***), if $\mathcal{S}$ is the collection of subsets, have subcollection $\mathcal{S}_C$ such that

$$\bigcup_{S \in \mathcal{S}_C} S = C$$



THE MANY NAMES OF BOOLEAN RANK

- Minimum tiling (data mining)
- Rectangle covering number (communication complexity)
- Minimum bi-clique edge covering number (Garey & Johnson GT18)
- Minimum set basis (Garey & Johnson SP7)
- Optimum key generation (cryptography)
- Minimum set of roles (access control)

COMPARISON OF RANKS

- Boolean rank is NP-hard to compute
  - And as hard to approximate as the minimum clique
- Boolean rank can be less than normal rank
  - $\text{rank}_B(\mathbf{A}) = O(\log_2(\text{rank}(\mathbf{A})))$ for certain $\mathbf{A}$

$$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

- Boolean rank is never more than the non-negative rank

APPROXIMATE FACTORISATIONS

- Noise usually makes real-world matrices (almost) full rank
- We want to find a good low-rank approximation
  - The goodness is measured using the Hamming distance
- Given **A** and *k*, find **B** and **C** such that **B** has *k* columns and |**A** – **B**o**C**| is minimised
  - No easier than finding the Boolean rank

THE BASIS USAGE PROBLEM

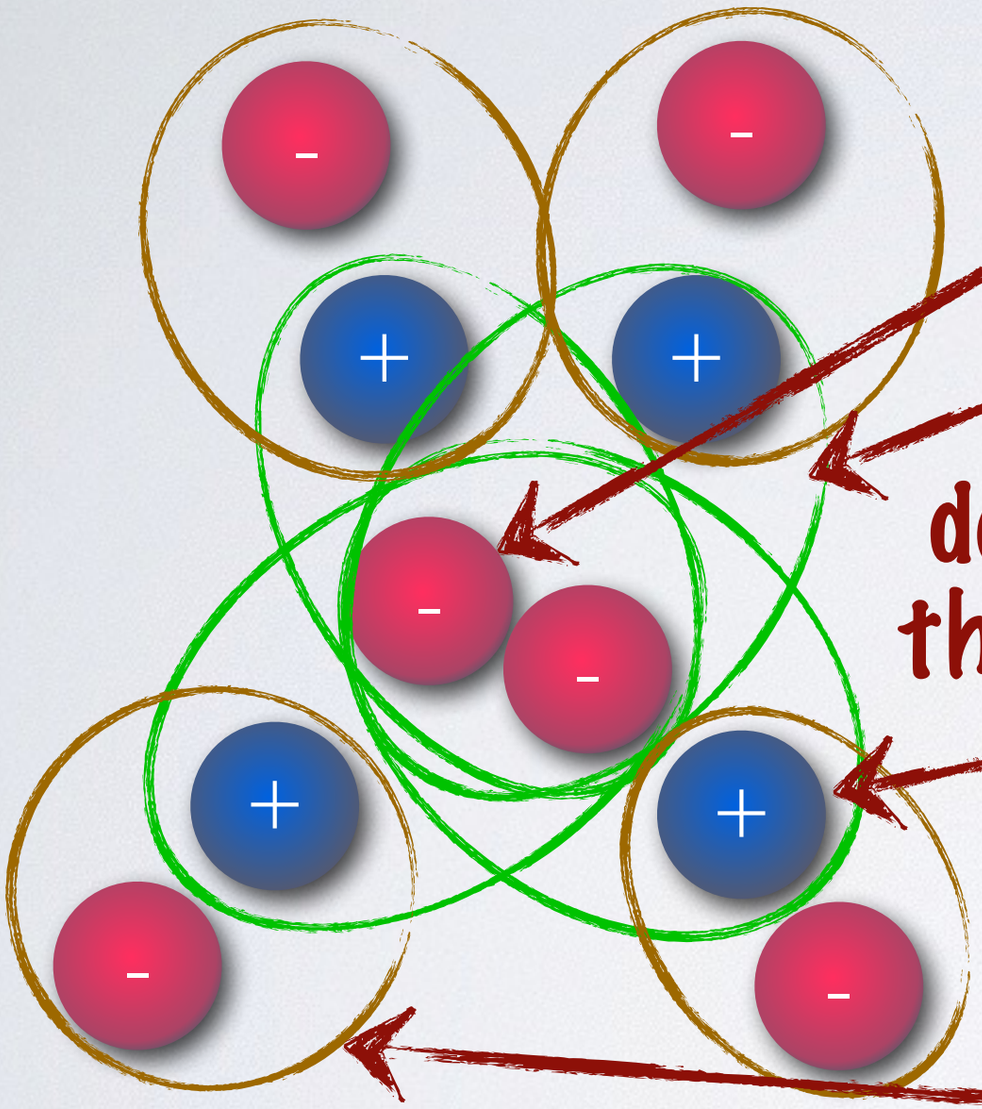
- Finding the factorisation is hard *even if we know one factor matrix*
- **Problem.** Given **B** and **A**, find **X** such that $|\mathbf{A} \circ \mathbf{X} - \mathbf{B}|$ is minimised
  - We can replace **B** and **X** with column vectors
    - $|\mathbf{A} \circ \mathbf{x} - \mathbf{b}|$ versus $||\mathbf{A}\mathbf{x} - \mathbf{b}||$
- Normal algebra: Moore–Penrose pseudo-inverse
- Boolean algebra: no polylogarithmic approximation

BOOLEAN PROJECTION, OR ±PSC

- The minimum-error projection under Boolean algebra is equivalent to the following problem

Positive-Negative Partial Set Cover (**±PSC**).
Given a triple $(P, N, \mathcal{Q})$, where P and N are disjoint sets and $\mathcal{Q} \subseteq 2^{P \cup N}$, find a subcollection $\mathcal{D} \subseteq \mathcal{Q}$ that minimises $|P \setminus (\cup \mathcal{D})| + |N \cap (\cup \mathcal{D})|$.

EXAMPLE

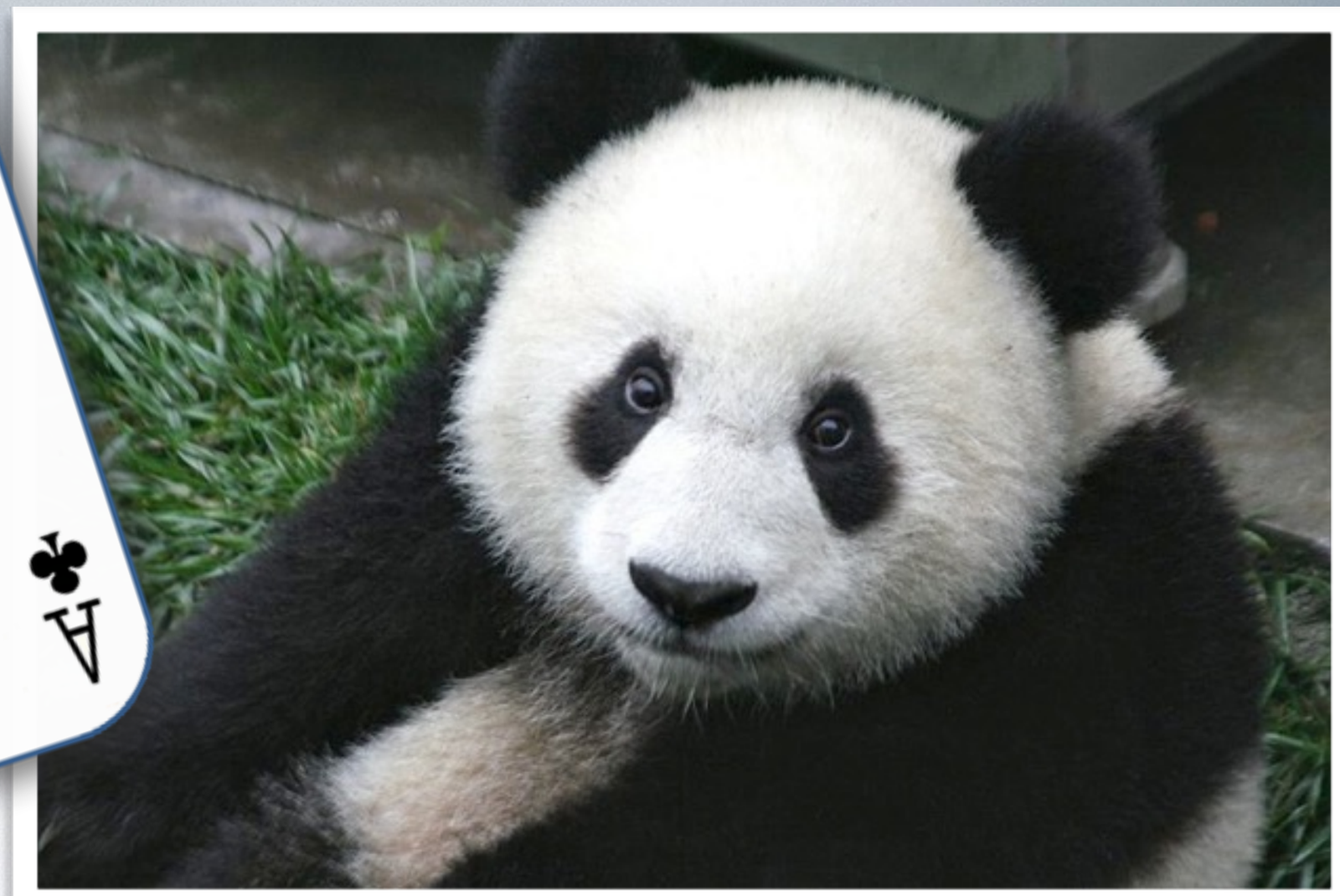


defines the sets

defines the sign

$$a = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

ALGORITHMS



Images by Wikipedia users Arab Ace and Sheilalau

THE BASIS USAGE

- Peleg's algorithm approximates within $2\sqrt{[(k+a)\log a]}$
  - a is the maximum number of 1s in $\boldsymbol{A}$'s columns
- Optimal solution
  - Either an $O(2^k knm)$ exhaustive search, or an integer program
- Greedy algorithm: select each column of $\boldsymbol{B}$ if it improves the residual error

THE ASSO ALGORITHM

- Heuristic – too many hardness results to hope for good provable results in any case
- **Intuition**: If two columns share a factor, they have 1s in same rows
  - Noise makes detecting this harder
  - Pairwise row association rules reveal (some of) the factors
    - $\Pr[a_{ik} = 1 \mid a_{jk} = 1]$

THE PANDA ALGORITHM

- **Intuition**: every good factor has a noise-free core
- Two-phase algorithm:
  1. Find error-free core pattern (maximum area itemset/tile)
  2. Extend the core with noisy rows/columns
- The core patterns are found using a greedy method
  - The 1s already belonging to some factor/tile are removed from the residual data where the cores are mined

SELECTING THE RANK

$$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

PRINCIPLES OF GOOD *K*

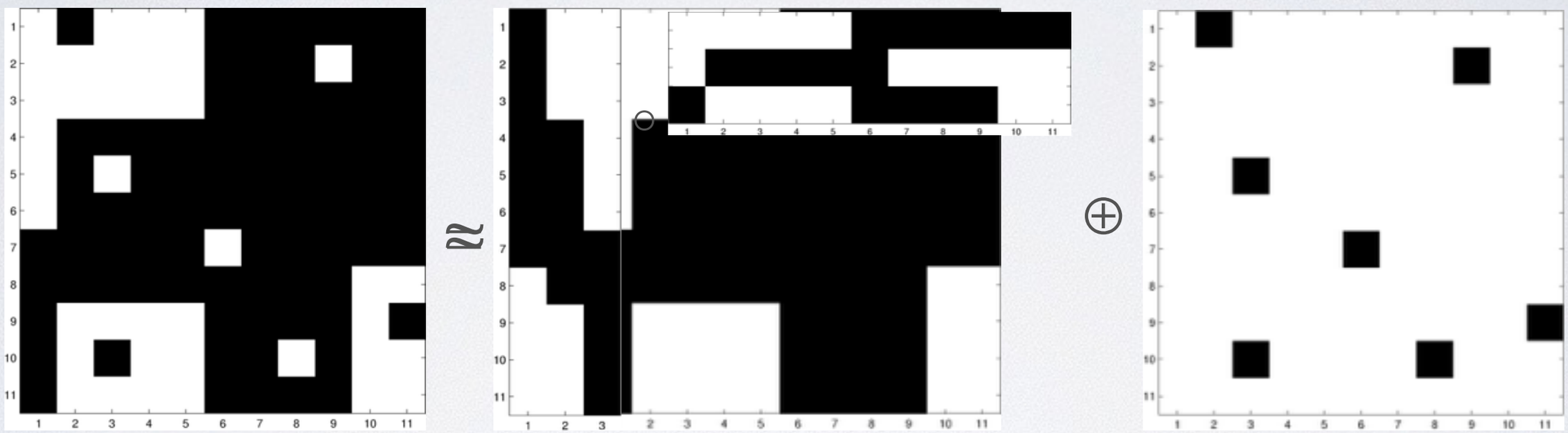
- **Goal:** Separate noise from structure
- We assume data has correct type of structure
  - There are *k* factors explaining the structure
  - Rest of the data does not follow the structure (noise)
- But how to decide where structure ends and noise starts?

MINIMUM DESCRIPTION LENGTH PRINCIPLE

- The best model (order) is the one that allows you to explain your data with least number of bits
  - Two-part (crude) MDL: the cost of model $L(\mathcal{H})$ plus the cost of data given the model $L(D \mid \mathcal{H})$
- Problem: how to do the encoding
  - All involved matrices are binary: well-known encoding schemes

FITTING BMF TO MDL

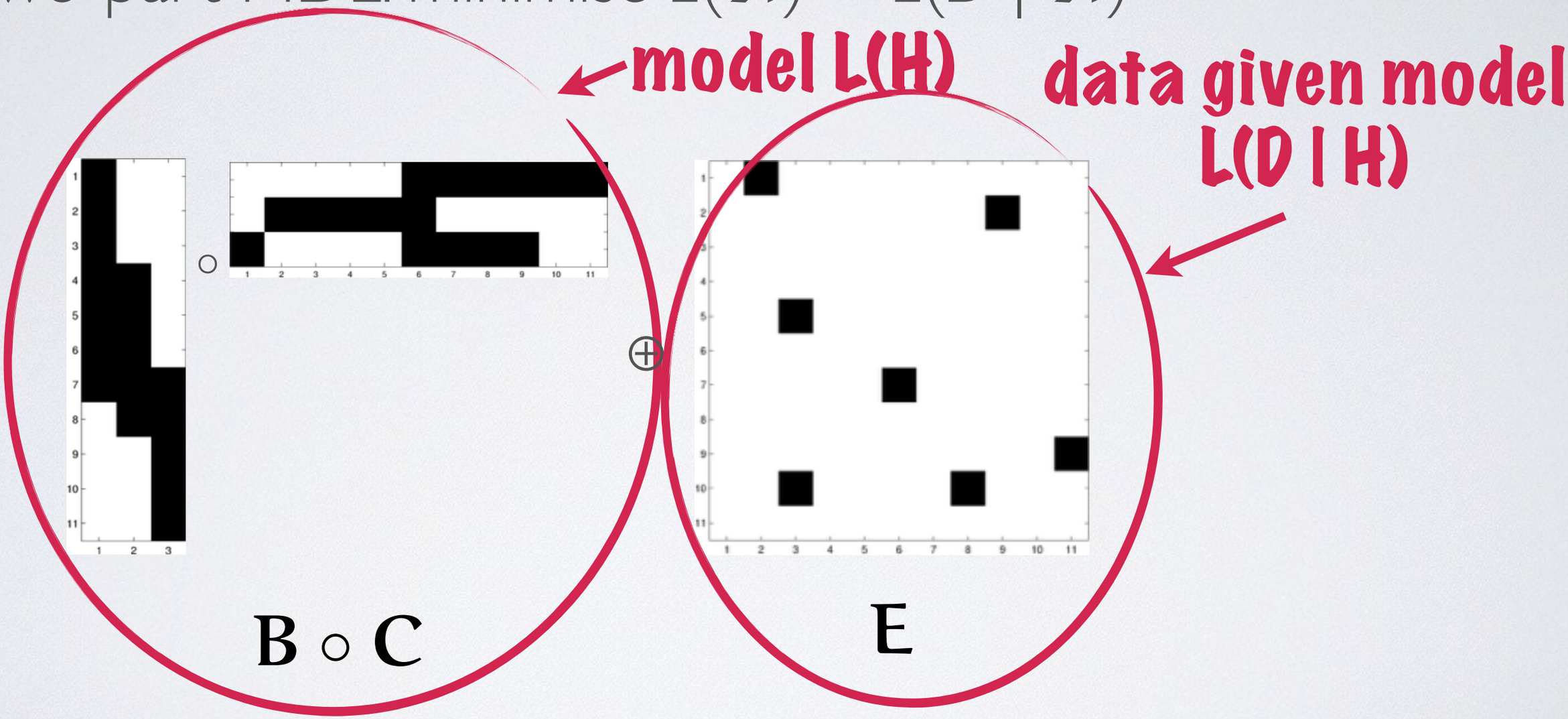
- MDL requires exact representation



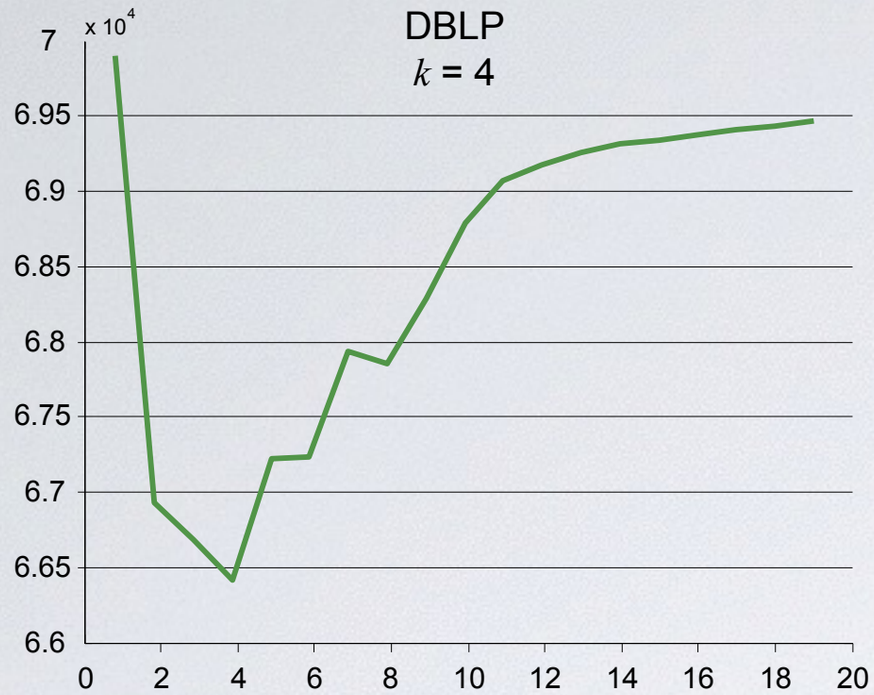
$$\mathbf{A} \approx \mathbf{B} \circ \mathbf{C} \oplus \mathbf{E}$$

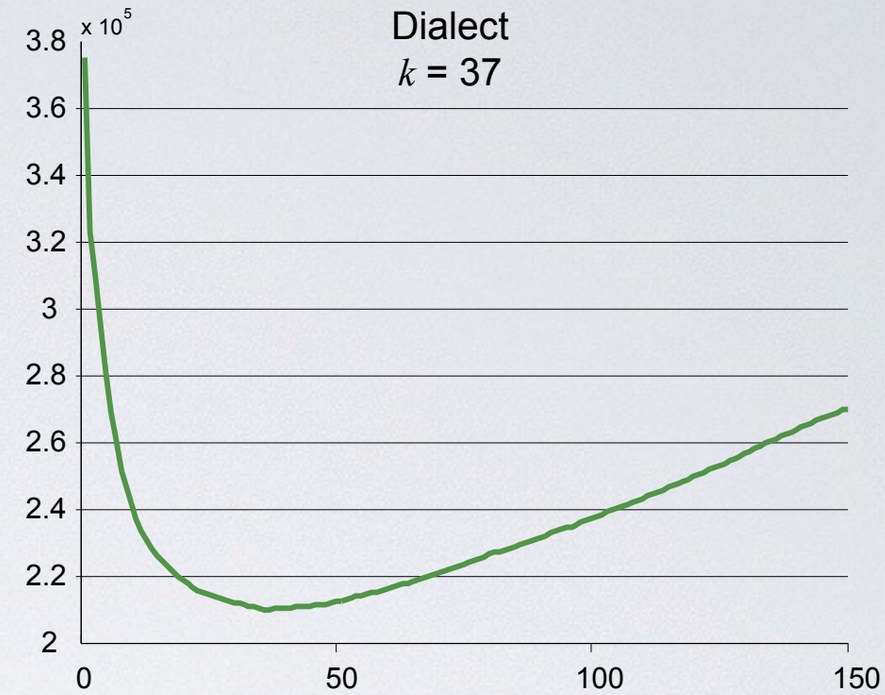
FITTING BMF TO MDL

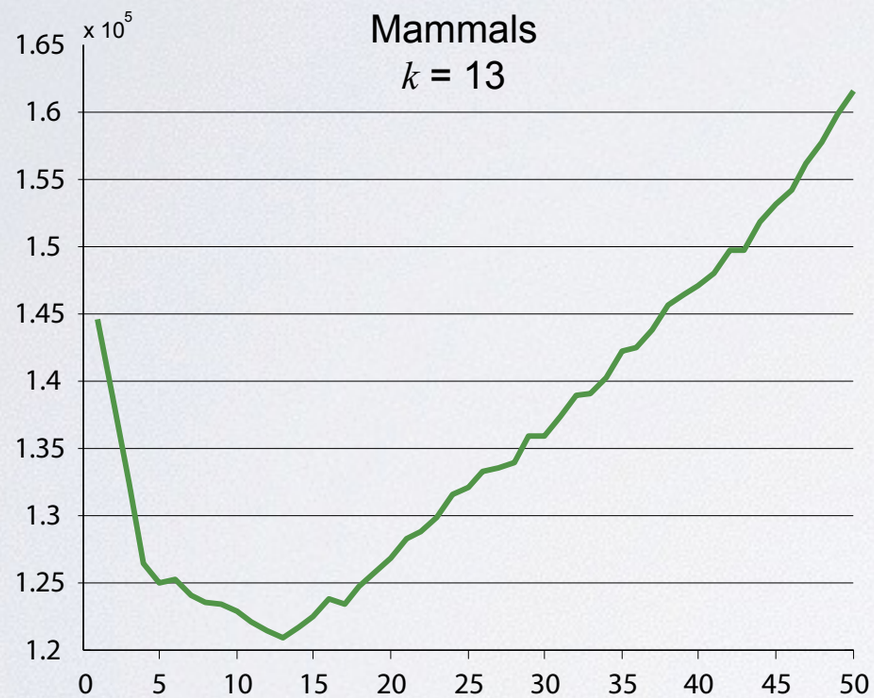
- Two-part MDL: minimise $L(\mathcal{H}) + L(D \mid \mathcal{H})$

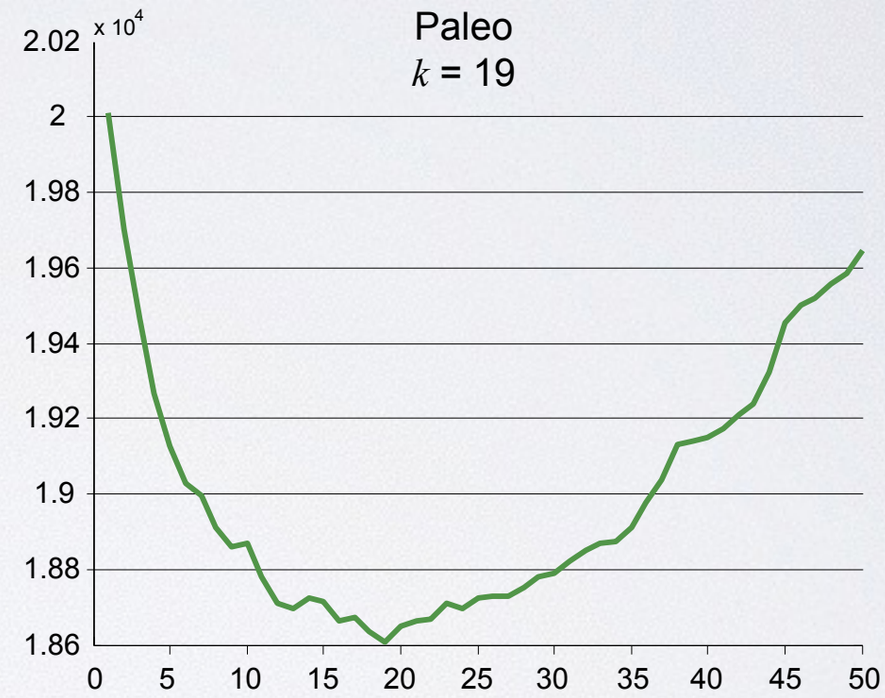


max planck institut
informatik









EXAMPLE: ASSO & MDL

max planck institut
informatik

SPARSE MATRICES

MOTIVATION

- Many real-world binary matrices are sparse
- Representing sparse matrices with sparse factors is desirable
  - Saves space, improves usability, …
- Sparse matrices should be computationally easier

SPARSE FACTORISATIONS #1

- Any binary matrix **A** that admits rank-k BMF has factorisation to matrices **B** and **C** such that $|\mathbf{B}| + |\mathbf{C}| \leq 2|\mathbf{A}|$
  - $|\mathbf{A}|$ is the number of non-zeros in **A**
  - Can be extended to approximate factorisations
  - Tight result (consider a case when **A** has exactly one 1)

SPARSE FACTORISATIONS #2

- Let $s(\mathbf{A}) = \text{zeros}(\mathbf{A})/nm = 1 - |\mathbf{A}|/nm$ be the fraction of zeros in $\mathbf{A}$
- If $\mathbf{B}$ and $\mathbf{C}$ are an **underapproximation** of $\mathbf{A}$ (that is, $\mathbf{B} \circ \mathbf{C} \leq \mathbf{A}$), then $s(\mathbf{B}) + s(\mathbf{C}) \geq s(\mathbf{A})$

CONCLUSIONS

Thank You!

- Boolean matrix factorisations are a topic older than I am
- Applications in many fields of CS and Math
- Approximate factorisations are an interesting tool for data mining
  - Work is not done yet…