# Machine Learning I

Bjoern Andres
bjoern.andres@tu-dresden.de

Machine Learning for Computer Vision
Faculty of Computer Science
TU Dresden

Version 1.0

# Contents

# Chapter 1

# Introduction

## 1.1 Notation

We shall use the following notation:

- We write "iff" as shorthand for "if and only if".
- For any finite set $A$, we denote by $|A|$ the number of elements of $A$.
- For any set $A$, we denote by $2^A$ the power set of $A$.
- For any set $A$ and any $m \in \mathbb{N}$, we denote by $\binom{A}{m}$ the set of all $m$-elementary subsets of $A$, i.e. $\binom{A}{m} = \{B \in 2^A \mid |B| = m\}$.
- For any sets $A, B$, we denote by $B^A$ the set of all maps from $A$ to $B$
- For any map $f \in B^A$, any $a \in A$ and any $b \in B$, we may write $b = f(a)$ or $b = f_a$ instead of $(a, b) \in f$
- Given any set $J$ and, for any $j \in J$, a set $S_j$, we denote by $\prod_{j \in J} S_j$ the Cartesian product of the family $\{S_j\}_{j \in J}$, i.e.

$$\prod_{j \in J} S_j = \left\{ f \colon J \to \bigcup_{j \in J} S_j \;\middle|\; \forall j \in J \colon f(j) \in S_j \right\} \tag{1.1}$$

- We denote by $\langle \cdot, \cdot \rangle$ the standard inner product, and by $\| \cdot \|$ the Euclidean norm.
- For any $m \in \mathbb{N}$, we define $[m] = \{0, \dots, m - 1\}$.

# Chapter 2

# Supervised learning

## 2.1 Intuition

Informally, supervised learning is the problem of finding in a family $g : \Theta \to Y^X$ of functions, one $g_\theta : X \to Y$ that minimizes a weighted sum of two objectives:

1. $g$ deviates little from a finite set $\{(x_s, y_s)\}_{s \in S}$ of input-output-pairs
2. $g$ has low complexity, as quantified by a function $R : \Theta \to \mathbb{R}_0^+$

We note that the family $g$ can have meaning beyond a mere parameterization of functions from $X$ to $Y$. For instance, $\Theta$ can be a set of forms, $g$ the functions defined by these forms, and $R$ the length of forms. In that case, supervised learning is really an optimization problem over forms of functions, and $R$ penalizes the complexity of these forms. Moreover, $g$ can be chosen so as to constrain the set of functions from $X$ to $Y$ in the first place.

We concentrate exclusively on the special case where $Y$ is finite. In fact, we concentrate on the case where $Y = \{0, 1\}$ in this chapter and reduce more general cases to this case in Chapter 4.

Moreover, we allow ourselves to take a detour by not optimizing over a family $g : \Theta \to \{0,1\}^X$ directly but instead optimizing over a family $f : \Theta \to \mathbb{R}^X$ and defining $g$ w.r.t. $f$ via a function $L : \mathbb{R} \times \{0, 1\} \to \mathbb{R}_0^+$, called a *loss function*, such that

$$\forall \theta \in \Theta \; \forall x \in X : \quad g_\theta(x) = \underset{\hat{y} \in \{0,1\}}{\operatorname{argmin}} \; L(f_\theta(x), \hat{y}) \ . \tag{2.1}$$

## 2.2 Definition

**Definition 1** For any $S \neq \varnothing$ finite, called a set of *samples*, any $X \neq \varnothing$, called an *attribute space* and any $x : S \to X$, the tuple $(S, X, x)$ is called *unlabeled data*.

For any $y : S \to \{0, 1\}$, given in addition and called a *labeling*, the tuple $(S, X, x, y)$ is called *labeled data*.

**Definition 2** For any labeled data $T = (S, X, x, y)$, any $\Theta \neq \varnothing$ and family of functions $f : \Theta \to \mathbb{R}^X$, any $R : \Theta \to \mathbb{R}_0^+$, called a *regularizer*, any $L : \mathbb{R} \times \{0, 1\} \to \mathbb{R}_0^+$, called a *loss function*, and any $\lambda \in \mathbb{R}_0^+$, called a *regularization parameter*, we define the following optimization problems:

- The instance of the *supervised learning problem* w.r.t. $T, \Theta, f, R, L$ and $\lambda$ is defined as

$$\inf_{\theta \in \Theta} \quad \lambda R(\theta) + \frac{1}{|S|} \sum_{s \in S} L(f_\theta(x_s), y_s) \tag{2.2}$$

- The instance of the *exact supervised learning problem* w.r.t. $T, \Theta, f$ and $R$ is defined as

$$\inf_{\theta \in \Theta} \quad R(\theta) \tag{2.3}$$

$$\text{subject to} \quad \forall s \in S : \quad f_\theta(x_s) = y_s \tag{2.4}$$

- The instance of the *bounded regularity problem* w.r.t. $T, \Theta, f, R$ and $m$ is to decide whether there exists a $\theta \in \Theta$ such that

$$R(\theta) \leq m \tag{2.5}$$

$$\forall s \in S: \quad f_\theta(x_s) = y_s \tag{2.6}$$

**Definition 3** For any unlabeled data $T = (S, X, x)$, any $\hat{f} : X \to \mathbb{R}$ and any $L : \mathbb{R} \times \{0,1\} \to \mathbb{R}_0^+$, the instance of the *inference problem* w.r.t. $T, f$ and $L$ is defined as

$$\min_{y' \in \{0,1\}^S} \sum_{s \in S} L(\hat{f}(x_s), y'_s) \tag{2.7}$$

**Lemma 1** *The solutions to the inference problem are the* $y : S \to \{0,1\}$ *such that*

$$\forall s \in S\colon \quad y_s \in \operatorname*{argmin}_{\hat{y} \in \{0,1\}} L(\hat{f}(x_s), \hat{y}) \ . \tag{2.8}$$

*Moreover, if*

$$\hat{f}(X) \subseteq \{0,1\} \tag{2.9}$$

*and*

$$\forall r \in \mathbb{R}\ \forall \hat{y} \in \{0,1\}\colon \quad L(r, \hat{y}) = \begin{cases} 0 & \text{if } r = \hat{y} \\ 1 & \text{otherwise} \end{cases} \tag{2.10}$$

*then*

$$\forall s \in S\colon \quad y'_s = \hat{f}(x_s) \ . \tag{2.11}$$

Proof Generally, we have

$$\min_{y \in \{0,1\}^S} \sum_{s \in S} L(\hat{f}(x_s), y_s) = \sum_{s \in S} \min_{y_s \in \{0,1\}} L(\hat{f}(x_s), y_s) \tag{2.12}$$

By (2.9), $L(\hat{f}(x_s), \hat{f}(x_s))$ is well-defined for any $s \in S$. By (2.10) and non-negativity of $L$, we have

$$\forall y_s \in \{0,1\}\colon \quad L(\hat{f}(x_s), \hat{f}(x_s)) = 0 \leq L(\hat{f}(x_s), y_s) \ . \tag{2.13}$$

Thus, $y_s = \hat{f}(x_s)$ is optimal for any $s \in S$.

We note that the exact supervised learning problem formalizes a philosophical principle known as Ockham's razor.

# Chapter 3

# Deciding

## 3.1 Disjunctive normal forms

### 3.1.1 Data

Throughout Section 3.1, we consider binary attributes. More specifically, we consider some finite set $V \neq \varnothing$ and labeled data $T = (S, X, x, y)$ such that $X = \{0,1\}^V$. Hence, $x\colon S \to \{0,1\}^V$ and $y\colon S \to \{0,1\}$.

### 3.1.2 Familiy of functions

Throughout Section 3.1, we identify $\Theta$ with a set of disjunctive normal forms. More specifically, we consider $\Gamma = \{(V_0, V_1) \in 2^V \times 2^V \mid V_0 \cap V_1 = \varnothing\}$ and $\Theta = 2^\Gamma$ and the following definition.

**Definition 4** For any $\theta \in \Theta$ and the $f_\theta\colon \{0,1\}^V \to \{0,1\}$ such that

$$\forall x \in \{0,1\}^V: \quad f_\theta(x) = \bigvee_{(V_0,V_1)\in\theta} \prod_{v\in V_0} (1 - x_v) \prod_{v \in V_1} x_v \enspace , \tag{3.1}$$

the form on the r.h.s. of (3.1) is called the *disjunctive normal form (DNF)* defined by $V$ and $\theta$. The function $f_\theta$ is said to be defined by the DNF. If there exists a $k \in \mathbb{N}$ such that $\forall (V_0, V_1) \in \theta\colon |V_0 \cup V_1| \leq k$, the DNF is also called a *k-DNF*.

For $\theta \in \Theta$ and the $g_\theta\colon \{0,1\}^V \to \{0,1\}$ such that

$$\forall x \in \{0,1\}^V: \quad g_\theta(x) = \prod_{(V_0,V_1)\in\theta} \left( \bigvee_{v\in V_0} (1 - x_v) \vee \bigvee_{v\in V_1} x_v \right) \tag{3.2}$$

the form on the r.h.s. of (3.2) is called the *conjunctive normal form (CNF)* defined by $V$ and $\theta$. The function $g_\theta$ is said to be defined by this CNF. If there exists a $k \in \mathbb{N}$ such that $\forall (V_0, V_1) \in \theta\colon |V_0 \cup V_1| \leq k$, the CNF is also called a *k-CNF*.

### 3.1.3 Regularization

**Definition 5** The functions $R_d, R_l : \Theta \to \mathbb{N}_0$ whose values are defined below for any $\theta \in \Theta$ are called the *depth* and *length*, resp., of the forms defined by $\theta$.

$$R_d(\theta) = \max_{(V_0,V_1)\in\theta} (|V_0| + |V_1|) \tag{3.3}$$

$$R_l(\theta) = \sum_{(V_0,V_1)\in\theta} (|V_0| + |V_1|) \tag{3.4}$$

### 3.1.4 Loss function

We consider the loss function $L$ such that

$$\forall r \in \mathbb{R}\ \forall \hat{y} \in \{0,1\}\colon \quad L(r,\hat{y}) = \begin{cases} 0 & r = \hat{y} \\ 1 & \text{otherwise} \end{cases} \quad . \tag{3.5}$$

### 3.1.5 Learning problem

**Definition 6** For any $R \in \{R_l, R_d\}$ and any $\lambda \in [0,1)$, the instance of the *supervised learning problem of DNFs* with respect to $T, L, R$ and $\lambda$ has the form

$$\min_{\theta \in \Theta} \quad \lambda R(\theta) + \frac{1-\lambda}{|S|} \sum_{s \in S} L(f_\theta(x_s), y_s) \tag{3.6}$$

In order to examine its computational complexity, we consider the related decision problems:

**Definition 7** Let $m \in \mathbb{N}$. The instance of the *bounded depth DNF problem (*DEPTH-$m$-DNF*)* w.r.t. $T$ and $m$ is to decide whether there exists a $\theta \in \Theta$ such that

$$R_d(\theta) \leq m \tag{3.7}$$

$$\forall s \in S\colon \quad f_\theta(x_s) = y_s \quad . \tag{3.8}$$

The instance of the *bounded length DNF problem (*LENGTH-$m$-DNF*)* w.r.t. $T$ and $m$ is to decide whether there exists a $\theta \in \Theta$ such that

$$R_l(\theta) \leq m \tag{3.9}$$

$$\forall s \in S\colon \quad f_\theta(x_s) = y_s \quad . \tag{3.10}$$

We relate these problems to SET-COVER.

**Definition 8 (Haussler (1988))** For any instance $(S', \Sigma, m)$ of SET-COVER, the *Haussler data* $T = (S, X, x, y)$ induced by $(S', \Sigma, m)$ is the labeled data such that

- $S = S' \cup \{1\}$
- $X = \{0,1\}^\Sigma$
- $x_1 = 1^\Sigma$ and

$$\forall s \in S'\ \forall \sigma \in \Sigma\colon \quad x_s(\sigma) = \begin{cases} 0 & \text{if } s \in \sigma \\ 1 & \text{otherwise} \end{cases} \tag{3.11}$$

- $y_1 = 1$ and $\forall s \in S'\colon y_s = 0$

**Lemma 2 (Haussler (1988))** *For any instance $(S', \Sigma, m)$ of* SET-COVER*, the Haussler data $T = (S, X, x, y)$ induced by $(S', \Sigma, m)$, and any $\Sigma' \in \binom{\Sigma}{m}$:*

$$\bigcup_{\sigma \in \Sigma'} \sigma = S' \quad \Leftrightarrow \quad \forall s \in S'\colon \prod_{\sigma \in \Sigma'} x_s(\sigma) = 0$$

PROOF

$$\bigcup_{\sigma \in \Sigma'} \sigma = S'$$

$$\Leftrightarrow \quad \forall s \in S'\ \exists \sigma \in \Sigma'\colon \quad s \in \sigma \tag{3.12}$$

$$\Leftrightarrow \quad \forall s \in S'\ \exists \sigma \in \Sigma'\colon \quad x_s(\sigma) = 0 \tag{3.13}$$

$$\Leftrightarrow \quad \forall s \in S'\colon \quad \prod_{\sigma \in \Sigma'} x_s(\sigma) = 0 \tag{3.14}$$

**Theorem 1** *a)* SET-COVER $\leq$ DEPTH-$m$-DNF
*b)* SET-COVER $\leq$ LENGTH-$m$-DNF

PROOF The proof is for any $R \in \{R_d, R_l\}$.
Let $(S', \Sigma, m)$ any instance of SET-COVER.
Let $T = (S, X, x, y)$ the Haussler data induced by $(S', \Sigma, m)$.
We show: There exists a cover $\Sigma' \subseteq \Sigma$ of $S'$ with $|\Sigma'| \leq m$ iff there exists a $\theta \in \Theta$ such that $R(\theta) \leq m$ and $\forall s \in S\colon f_\theta(x_s) = y_s$.
($\Rightarrow$) Let $\Sigma' \subseteq \Sigma$ a cover of $S$ and $|\Sigma'| \leq m$.
Let $V_0 = \varnothing$ and $V_1 = \Sigma'$ and $\theta = \{(V_0, V_1)\}$. Thus,

$$\forall x' \in X\colon \quad f_\theta(x') = \prod_{\sigma \in \Sigma'} x'(\sigma) \tag{3.15}$$

On the one hand, $f(S') = 0$, by Lemma 2, and $f(1^\Sigma) = 1$, by definition of $f_\theta$. Thus, $\forall s \in S\colon f(x_s) = y_s$.
On the other hand, $R(\theta) = m$.
($\Leftarrow$) Let $\theta \in \Theta$ such that $R(\theta) \leq m$ and $\forall s \in S\colon f_\theta(x_s) = y_s$.
There exists a $(\Sigma_0, \Sigma_1) \in \theta$ such that $\Sigma_0 = \varnothing$, because $1 = y_1 = f_\theta(x_1) = f_\theta(1^\Sigma)$.
Moreover

$$\begin{aligned}
& \forall s \in S'\colon \quad f(x_s) = 0 \\
\Rightarrow \quad & \forall s \in S'\colon \quad \bigvee_{(V_0,V_1)\in\theta} \prod_{v \in V_0} (1 - x_s(v)) \prod_{v \in V_1} x_s(v) = 0 && (3.16)\\
\Rightarrow \quad & \forall s \in S' \ \forall (V_0, V_1) \in \theta\colon \quad \prod_{v \in V_0} (1 - x_s(v)) \prod_{v \in V_1} x_s(v) = 0 && (3.17)
\end{aligned}$$

Thus, for $(\varnothing, \Sigma_1) \in \theta$ in particular:

$$\forall s \in S'\colon \quad \prod_{\sigma \in \Sigma_1} x_s(\sigma) = 0 \tag{3.18}$$

And by virtue of Lemma 2:

$$\bigcup_{\sigma \in \Sigma_1} \sigma = S' \tag{3.19}$$

Furthermore, $|\Sigma_1| \leq R(\theta) = m$.

### 3.1.6 Inference problem

For any $\theta \in \Theta$, the inference problem w.r.t. $f_\theta, L$ and any suitable $S', X', x'$ is solved by computing $f_\theta(x'_s)$ for any $s \in S'$, by Lemma 1.

### 3.1.7 Inference algorithm

Computing $f_\theta(x'_s)$ requires evaluating the form (3.1). The number of operations is bounded by $R_l(\theta) = O(|\theta| R_d(\theta)) = O(|\theta||V|)$.

## 3.2 Binary decision trees

### 3.2.1 Data

Throughout Section 3.2, we again consider some finite set $V \neq \varnothing$ and labeled data $T = (S, X, x, y)$ such that $X = \{0,1\}^V$. Hence, $x\colon S \to \{0,1\}^V$ and $y\colon S \to \{0,1\}$.

### 3.2.2 Familiy of functions

**Definition 9** A tuple $(V, Y, D, D', d^*, E, \delta, v, y)$ is called a $V$-variate $Y$-valued *binary decision tree (BDT)* iff the following conditions hold:

- $V$ is finite and non-empty, called the set of *variables*
- $Y$ is finite, called the set of *values*
- $(D \cup D', E)$ is a finite, non-empty, directed binary tree
- $d^* \in D \cup D'$ is the unique root of this tree
- $\delta : E \to \{0, 1\}$
- Every $d \in D'$ is a leaf and every $d \in D$ has precisely two out-edges $e = (d, d'), e' = (d, d'')$ such that $\delta(e) = 0$ and $\delta(e) = 1$
- $v : D \to V$ assigning to each interior node a variable
- $y : D' \to Y$ assigning to each leaf a value

For any BDT $(V, Y, D, D', d^*, E, \delta, v, y)$, any $d \in D$ and any $j \in \{0, 1\}$, let $d_{\downarrow j} \in D \cup D'$ the unique node such that $e = (d, d_{\downarrow j}) \in E$ and $\delta(e) = j$.

Throughout Section 3.2, we consider $Y = \{0, 1\}$.

**Definition 10** For any BDT $\theta = (V, Y, D, D', d^*, E, \delta, v, y)$ and any $d \in D \cup D'$, the tuple $\theta[d] = (V, Y, D_2, D_2', d, E', \delta', v', y')$ with $(D_2 \cup D_2', E')$ the subtree of $(D \cup D', E)$ rooted at $d$ and with $\delta'$, $v'$ and $y'$ the restrictions of $\delta$, $v$ and $y$ to the subsets $D_2$, $D_2'$ and $E'$ is called the *binary decision subtree* of $\theta$ rooted at $d$.

**Lemma 3** *For any BDT $\theta = (V, Y, D, D', d^*, E, \delta, v, y)$ and any $d \in D \cup D'$, the binary decision subtree $\theta[d]$ is itself a $V$-variate $Y$-valued BDT.*

**Definition 11** For any BDT $\theta = (V, Y, D, D', d^*, E, \delta, v, y)$, the function defined by $\theta$ is the $f_\theta : \{0, 1\}^V \to Y$ such that $\forall x \in \{0, 1\}^V$:

$$f_\theta(x) = \begin{cases} y(d^*) & \text{if } D = \varnothing \\ (1 - x_{v(d^*)}) f_{\theta[d^*_{\downarrow 0}]}(x) + x_{v(d^*)} f_{\theta[d^*_{\downarrow 1}]}(x) & \text{otherwise} \end{cases} . \tag{3.20}$$

### 3.2.3 Regularization

**Definition 12** For any BDT $\theta = (V, Y, D, D', d^*, E, \delta, v, y)$, the *depth* of $\theta$ is the natural number $R(\theta) \in \mathbb{N}$ such that

$$R(\theta) = \begin{cases} 0 & \text{if } D = \varnothing \\ 1 + \max\{R(\theta[d^*_{\downarrow 0}]), R(\theta[d^*_{\downarrow 1}])\} & \text{otherwise} \end{cases} . \tag{3.21}$$

### 3.2.4 Loss function

We consider the loss function $L$ such that

$$\forall r \in \mathbb{R}\ \forall \hat{y} \in \{0, 1\} \colon \quad L(r, \hat{y}) = \begin{cases} 0 & r = \hat{y} \\ 1 & \text{otherwise} \end{cases} . \tag{3.22}$$

### 3.2.5 Learning problem

**Definition 13** For any $m \in \mathbb{N}$, the *bounded depth BDT problem (*DEPTH-$m$-BDT*)* w.r.t. $T$ and $m$ is to decide whether there exists a BDT $\theta = (V, Y, D, D', d^*, E, \delta, v, y')$ such that

$$R(\theta) \leq m \tag{3.23}$$
$$\forall s \in S\colon \quad f_\theta(x_s) = y_s \ . \tag{3.24}$$

**Theorem 2** EC-3 $\leq_p$ DEPTH-$m$-BDT

PROOF For any instance $(S', \Sigma)$ of EC-3 and the $n \in \mathbb{N}$ such that $|S'| = 3n$, we construct the instance of DEPTH-$m$-BDT such that

- $V = \Sigma$
- $S = S' \cup \{0\}$
- $x : S \to \{0,1\}^\Sigma$ such that $x_0 = 0$ and

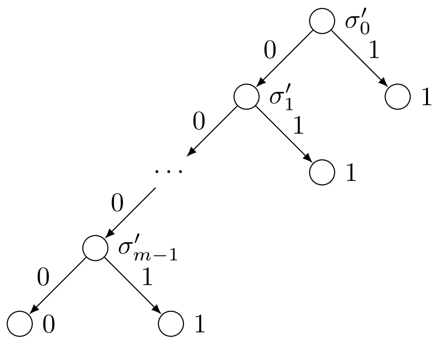
$$\forall s \in S' \ \forall \sigma \in \Sigma\colon \quad x_s(\sigma) = \begin{cases} 1 & \text{if } s \in \sigma \\ 0 & \text{otherwise} \end{cases} \tag{3.25}$$

- $y : S \to \{0,1\}$ such that $y_0 = 0$ and $\forall s \in S'\colon y_s = 1$.
- $m = n$

Next, we show that the instance EC-3 has a solution iff the instance of DEPTH-$m$-BDT has a solution.

($\Rightarrow$) Let $\Sigma' \subseteq \Sigma$ a solution to the instance of EC-3.

Consider any order on $\Sigma'$ and the bijection $\sigma' : [n] \to \Sigma'$ induced by this order. We show that the BDT $\theta$ depicted below solves the instance of DEPTH-$m$-BDT.

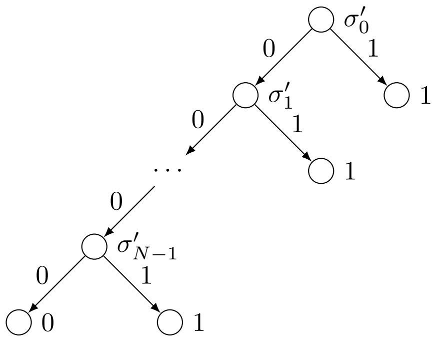


The BDT satisfies (3.23) as $R(\theta) = m$.

The BDT satisfies (3.24) because (i) $f_\theta(x_0) = 0 = y_0$ and (ii) at each of the $m$ interior nodes, three additional elements of $S'$ are mapped to 1. Thus, all $3m$ many elements $s \in S'$ are mapped to 1. That is $\forall s \in S'\colon f_\theta(x_s) = 1 = y_s$.

($\Leftarrow$) Let $\theta = (V, Y, D, D', d^*, E, \delta, \sigma, y')$ a $V$-variate BDT that solves the instance of DEPTH-$m$-BDT.

W.l.o.g., we assume, for any interior node $d \in D$, that $d_{\downarrow 1}$ is a leaf and $y'(d_{\downarrow 1}) = 1$. Hence, $\theta$ is of the form depicted below.



Therefore,

$$\forall x \in X: \quad f_\theta(x) = \begin{cases} 1 & \text{if } \exists j \in [N]\colon x(\sigma_j) = 1 \\ 0 & \text{otherwise} \end{cases} \quad , \tag{3.26}$$

and thus,

$$\forall s \in S: \quad f_\theta(x_s) = \begin{cases} 1 & \text{if } \exists j \in [N]\colon s \in \sigma_j \\ 0 & \text{otherwise} \end{cases} \quad , \tag{3.27}$$

by definition of $x$ in (3.25). Consequently,

$$\sigma(D) = \bigcup_{j=0}^{N-1} \sigma_j = S' \ , \tag{3.28}$$

by definition of $y$ such that $y(S') = 1$.

Moreover,

$$3m = |S'| \overset{(3.28)}{=} \left| \bigcup_{j=0}^{N-1} \sigma_j \right| \leq \sum_{j=0}^{N-1} |\sigma_j| = \sum_{j=0}^{N-1} 3 = 3N \overset{(3.23)}{\leq} 3m \ .$$

Therefore,

$$\forall \{j,l\} \in \tbinom{[N]}{2}\colon \quad \sigma_k \cap \sigma_l = \emptyset \ , \tag{3.29}$$

by Lemma 24. Hence,

$$\sigma(D) = \cup_{j=1}^{N-1} \sigma_j$$

is a solution to the instance of EC-3 defined by $(S', \Sigma)$, by (3.28) and (3.29).

**Lemma 4** DEPTH-$m$-BDT *is* NP*-complete.*

PROOF DEPTH-$m$-BDT $\in$ NP, as solutions can be verified efficiently.

DEPTH-$m$-BDT is NP-hard, by Theorem 2 and Lemma 22.

**Corollary 1** *Discriminative learning of BDTs is* NP*-hard.*

### 3.2.6 Learning algorithm

We introduce a popular heuristic for constructing BDTs.

**Definition 14** For any finite set $S$, any functions $y, y' : S \to Y$ and the set

$$G(y,y') = \Big\{ \{s,t\} \in \tbinom{S}{2} \;\Big|\; (y(s) = y(t) \wedge y'(s) \neq y'(t)) \vee (y(s) \neq y(t) \wedge y'(s) = y'(t)) \Big\} \tag{3.30}$$

the function $\Delta : Y^S \times Y^S \to [0,1]$ such that

$$\Delta(y,y') = \frac{|G(y,y')|}{\binom{|S|}{2}} \tag{3.31}$$

is called the *impurity pseudometric* of functions $Y^S$.

**Exercise 1** *(i) Show that the impurity pseudometric is indeed a pseudometric.*
*(ii) Show that the impurity pseudometric is not necessarily a metric.*

**Definition 15** For any of the given labeled data $T = (S, X, x, y)$, the *greedy impurity minimizing algorithm* constructs a BDT $\theta = (V, Y = \{0,1\}, D, D', d^*, E, \delta, v, y')$ by initializing $D = D' = E = \varnothing$ and executing the procedure grow$(d^*, S)$.

```
grow($d, S'$)
if $\forall s, s' \in S' : y_s = y_{s'}$
    $D' := D' \cup \{d\}$                                   add leaf $d$
    choose $s \in S'$
    $y'_d := y_s$
else
    $D := D \cup \{d\}$                                     add interior node $d$
    choose $\hat{v} \in \min_{v \in V} \Delta(y, x_v)$
    for $j \in \{0,1\}$
        let $d_j \notin D \cup D'$
        grow($d_j, \{s \in S' \mid x_s(\hat{v}) = j\}$)          recurse
        $e := (d, d_j)$
        $E := E \cup \{e\}$
        $\delta_e := j$
```

### 3.2.7 Inference problem

For any BDT $\theta = (V, Y, D, D', d^*, E, \delta, v, y')$, the inference problem w.r.t. $f_\theta, L$ and any suitable $S', X', x'$ is solved by computing $f_\theta(x'_s)$ for any $s \in S'$, by Lemma 1.

### 3.2.8 Inference algorithm

The inference problem is solved by evaluating the form (3.20). The time complexity is $O(R(\theta))$.

## 3.3 Linear functions

### 3.3.1 Data

Throughout Section 3.3, we consider real attributes. More specifically, we consider some finite set $V \neq \varnothing$ and labeled data $T = (S, X, x, y)$ with $X = \mathbb{R}^V$. Hence, $x : S \to \mathbb{R}^V$ and $y : S \to \{0,1\}$.

### 3.3.2 Familiy of functions

Throughout Section 3.3, we consider linear functions. More specifically, we consider $\Theta = \mathbb{R}^V$ and $f : \Theta \to \mathbb{R}^X$ such that

$$\forall \theta \in \Theta \; \forall \hat{x} \in X : \quad f_\theta(\hat{x}) = \langle \theta, \hat{x} \rangle \; . \tag{3.32}$$

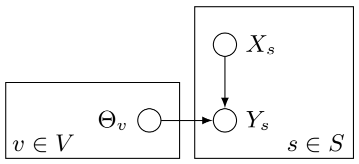
### 3.3.3 Probabilistic model

#### Random variables

- For any $s \in S$, let $X_s$ be a random variable whose realization is a vector $x_s \in \mathbb{R}^V$, called the *attribute vector* of $s$
- For any $s \in S$, let $Y_s$ be a random variable whose realization is a binary number $y_s \in \{0, 1\}$, called the *label* of $s$
- For any $v \in V$, let $\Theta_v$ be a random variable whose realization is a real number $\theta_v \in \mathbb{R}$, called a *parameter*

#### Conditional independence assumptions

We assume a probability distribution that factorizes according to the Bayesian net depicted below.



#### Factorization

- Firstly:

$$P(X, Y, \Theta) = \prod_{s \in S} P(Y_s \mid X_s, \Theta) P(X_s) \prod_{v \in V} P(\Theta_v) \tag{3.33}$$

- Secondly:

$$\begin{aligned} P(\Theta \mid X, Y) &= \frac{P(X, Y, \Theta)}{P(X, Y)} \\ &= \frac{P(Y \mid X, \Theta)\, P(X)\, P(\Theta)}{P(X, Y)} \\ &\propto P(Y \mid X, \Theta)\, P(\Theta) \\ &= \prod_{s \in S} P(Y_s \mid X_s, \Theta) \prod_{v \in V} P(\Theta_v) \end{aligned} \tag{3.34}$$

#### Forms

We consider:

- The *logistic distribution*

$$\forall s \in S : \qquad p_{Y_s \mid X_s, \Theta}(1) = \frac{1}{1 + 2^{-f_\theta(x_s)}} \tag{3.35}$$

- A $\sigma \in \mathbb{R}^+$ and the *normal distribution*:

$$\forall v \in V : \qquad p_{\Theta_v}(\theta_v) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\theta_v^2 / 2\sigma^2} \tag{3.36}$$

### 3.3.4 Learning problem

**Lemma 5 (Logistic regression)** *Estimating maximally probable parameters $\theta$, given attributes $x$ and labels $y$, i.e.,*

$$\operatorname*{argmax}_{\theta \in \mathbb{R}^m} \quad p_{\Theta|X,Y}(\theta, x, y)$$

*is identical to the supervised learning problem w.r.t. $L$, $R$ and $\lambda$ such that*

$$\forall r \in \mathbb{R}\ \forall \hat{y} \in \{0,1\}\colon \quad L(r, \hat{y}) = -\hat{y}r + \log\left(1 + 2^r\right) \tag{3.37}$$

$$\forall \theta \in \Theta\colon \quad R(\theta) = \|\theta\|_2^2 \tag{3.38}$$

$$\lambda = \frac{\log e}{2\sigma^2} \tag{3.39}$$

Proof Firstly,

$$\begin{aligned}
&\operatorname*{argmax}_{\theta \in \mathbb{R}^m} \quad p_{\Theta|X,Y}(\theta, x, y) \\
\overset{(3.34)}{=}\ &\operatorname*{argmax}_{\theta \in \mathbb{R}^m} \quad \prod_{s \in S} p_{Y_s|X_s,\Theta}(y_s, x_s, \theta) \prod_{v \in V} p_{\Theta_v}(\theta_v) \\
=\ &\operatorname*{argmax}_{\theta \in \mathbb{R}^m} \quad \sum_{s \in S} \log p_{Y_s|X_s,\Theta}(y_s, x_s, \theta) + \sum_{v \in V} \log p_{\Theta_v}(\theta_v)
\end{aligned} \tag{3.40}$$

Substituting in (3.40) the linearization

$$\begin{aligned}
&\log p_{Y_s|X_s,\Theta}(y_s, x_s, \theta) \\
= \quad & y_s \log p_{Y_s|X_s,\Theta}(1, x_s, \theta) + (1 - y_s) \log p_{Y_s|X_s,\Theta}(0, x_s, \theta) \\
= \quad & y_s \log \frac{p_{Y_s|X_s,\Theta}(1, x_s, \theta)}{p_{Y_s|X_s,\Theta}(0, x_s, \theta)} + \log p_{Y_s|X_s,\Theta}(0, x_s, \theta)
\end{aligned} \tag{3.41}$$

as well as (3.35) and (3.36) yields the form (3.42) below that is called the instance of the $l_2$-regularized *logistic regression problem* with respect to $x$, $y$ and $\sigma$.

$$\operatorname*{argmin}_{\theta \in \mathbb{R}^m} \quad \sum_{s \in S} \left(-y_s \langle \theta, x_s \rangle + \log\left(1 + 2^{\langle \theta, x_s \rangle}\right)\right) + \frac{\log e}{2\sigma^2} \|\theta\|_2^2 \tag{3.42}$$

**Exercise 2** *a) Derive* (3.42) *from* (3.40) *using* (3.41), (3.35) *and* (3.36)
*b) Is the objective function of* (3.42) *convex?*

### 3.3.5 Inference problem

**Lemma 6** *Estimating maximally probable labels $y$, given attributes $x'$ and parameters $\theta$, i.e.,*

$$\operatorname*{argmax}_{y \in \{0,1\}^S} \quad p_{Y|X,\Theta}(y, x', \theta) \tag{3.43}$$

*is identical to the inference problem w.r.t. $f$ and $L$. It has the solution*

$$\forall s \in S' : \quad y_s = \begin{cases} 1 & \text{if } f_\theta(x'_s) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.44}$$

Proof Firstly,

$$\begin{aligned}
& \operatorname*{argmax}_{y \in \{0,1\}^{S'}} \quad p_{Y|X,\Theta}(y, x', \theta) \\
= & \operatorname*{argmax}_{y \in \{0,1\}^{S'}} \quad \prod_{s \in S'} p_{Y_s|X_s,\Theta}(y_s, x'_s, \theta) \\
= & \operatorname*{argmax}_{y \in \{0,1\}^{S'}} \quad \sum_{s \in S'} \log p_{Y_s|X_s,\Theta}(y_s, x'_s, \theta) \\
= & \operatorname*{argmax}_{y \in \{0,1\}^{S'}} \quad \sum_{s \in S'} \left( y_s \log \frac{p_{Y_s|X_s,\Theta}(1, x'_s, \theta)}{p_{Y_s|X_s,\Theta}(0, x'_s, \theta)} + \log p_{Y_s|X_s,\Theta}(0, x'_s, \theta) \right) \\
= & \operatorname*{argmin}_{y \in \{0,1\}^{S'}} \quad \sum_{s \in S'} \left( -y_s f_\theta(x'_s) + \log \left( 1 + 2^{f_\theta(x'_s)} \right) \right) \\
= & \operatorname*{argmin}_{y \in \{0,1\}^{S'}} \quad \sum_{s \in S'} L(f_\theta(x'_s), y_s) \ .
\end{aligned}$$

Secondly,

$$\min_{y \in \{0,1\}^{S'}} \sum_{s \in S'} \left( -y_s f_\theta(x'_s) + \log \left( 1 + 2^{f_\theta(x'_s)} \right) \right) = \sum_{s \in S'} \max_{y_s \in \{0,1\}} y_s f_\theta(x'_s) \ .$$

### 3.3.6 Inference algorithm

The inference problem is solved by computing independently for each $s \in S'$ the label

$$y_s = \begin{cases} 1 & \text{if } \langle \theta, x'_s \rangle > 0 \\ 0 & \text{otherwise} \end{cases} \ . \tag{3.45}$$

The time complexity is $O(|V||S'|)$.

# Chapter 4

# Semi-supervised and unsupervised learning

## 4.1 Intuition

So far, we have considered learning problems w.r.t. labeled data $(S, X, x, y)$ where, for every $s \in S$, a label $y_s \in \{0, 1\}$ is given, and inference problems w.r.t. unlabeled data $(S', X', x)$ where no label is given and every combination of labels $y' : S \to \{0, 1\}$ is a feasible solution.

Next, we consider learning problems where not every label is given and inference problems where not every combination of labels is feasible. Unlike before, the data we look at in both problems coincides, consisting of tuples $(S, X, x, \mathcal{Y})$ where $\mathcal{Y} \subseteq \{0, 1\}^S$ is a set of feasible labelings. In particular, $\mathcal{Y} = \{0, 1\}^S$ is the special case of unlabeled data, and $|\mathcal{Y}| = 1$ is the special case of labeled data. Non-trivial choices of $\mathcal{Y}$ allow us to express problems of learning and inferring finite structures such as maps (Chapter 5), equivalence relations (Chapter 6) and orders (Chapter 8).

## 4.2 Definition

**Definition 16** For any $S \neq \emptyset$ finite, called a set of *samples*, any $X \neq \emptyset$, called an *attribute space*, any $x : S \to X$ and any $\emptyset \neq \mathcal{Y} \subseteq \{0, 1\}^S$, called a set of *feasible labelings*, the tuple $T = (S, X, x, \mathcal{Y})$ is called *constrained data*.

**Definition 17** For any constrained data $T = (S, X, x, \mathcal{Y})$, any $\Theta \neq \emptyset$ and family of functions $f : \Theta \to \mathbb{R}^X$, any $R : \Theta \to \mathbb{R}_0^+$, called a *regularizer*, any $L : \mathbb{R} \times \{0, 1\} \to \mathbb{R}_0^+$, called a *loss function* and any $\lambda \in \mathbb{R}_0^+$, called a *regularization parameter*, the instance of the *learning and inference problem* w.r.t. $T, \Theta, f, R, L$ and $\lambda$ is defined as

$$\min_{y \in \mathcal{Y}} \inf_{\theta \in \Theta} \quad \lambda R(\theta) + \frac{1}{|S|} \sum_{s \in S} L(f_\theta(x_s), y_s) \tag{4.1}$$

The special case of one-elementary $\mathcal{Y} = \{y\}$ is called the *supervised learning problem*.

The special case of one-elementary $\Theta = \{\hat{\theta}\}$ written below is called the *inference problem*.

$$\min_{y \in \mathcal{Y}} \quad \sum_{s \in S} L(f_\theta(x_s), y_s) \tag{4.2}$$

The mixed integer optimization problem (4.1) is an topic of machine learning research beyond the scope of this lecture. Special cases of the binary optimization problem (4.2) have been studied intensively and are discussed in the following chapters.

# Chapter 5

# Classifying

## 5.1 Maps

For any finite set $A \neq \varnothing$ whose elements we seek to classify and any finite set $B \neq \varnothing$ of class labels, we are interested in *maps* $\varphi : A \to B$ that assign to every element $a \in A$ precisely one class label $\varphi(a) \in B$. Maps are precisely those subsets of $\varphi \subseteq A \times B$ that satisfy

$$\forall a \in A \; \exists b \in B : \quad (a, b) \in \varphi \tag{5.1}$$
$$\forall a \in A \; \forall b, b' \in B : \quad (a, b) \in \varphi \land (a, b') \in \varphi \Rightarrow b = b' \; . \tag{5.2}$$

They are characterized by those functions $y : A \times B \to \{0, 1\}$ that satisfy

$$\forall a \in A\colon \quad \sum_{b \in B} y_{ab} = 1 \; . \tag{5.3}$$

We reduce the problem of learning and inferring maps to the problem of learning and inferring decisions, by choosing constrained data with

$$S = A \times B \tag{5.4}$$
$$\mathcal{Y} = \left\{ y\colon A \times B \to \{0,1\} \;\middle|\; \forall a \in A\colon \sum_{b \in B} y_{ab} = 1 \right\} \; . \tag{5.5}$$

## 5.2 Linear functions

### 5.2.1 Data

Throughout Section 5.2, we consider some finite set $V \neq \varnothing$ and constrained data $(S, X, x, \mathcal{Y})$ with $S = A \times B$ as in (5.4), $X = B \times \mathbb{R}^V$, and $\mathcal{Y}$ as in (5.5). More specifically, we assume that, for any $(a, b) \in A \times B$, the class label $b$ is the first attribute of $(a, b)$, i.e.,

$$\forall a \in A \; \forall b \in B \; \exists \hat{x} \in \mathbb{R}^V : \quad x_{ab} = (b, \hat{x}) \tag{5.6}$$

As a special case, we consider labeled data where we are given just one $\mathcal{Y} = \{y\}$ with $y$ satisfying the constraints (5.3).

### 5.2.2 Familiy of functions

Throughout Section 5.2, we consider linear functions. More specifically, we consider $\Theta = \mathbb{R}^{B \times V}$ and $f : \Theta \to \mathbb{R}^X$ such that

$$\forall \theta \in \Theta \; \forall b \in B \; \forall \hat{x} \in \mathbb{R}^V : \quad f_\theta((b, \hat{x})) = \sum_{v \in V} \theta_{bv} \, \hat{x}_v = \langle \theta_{b \cdot}, \hat{x} \rangle \; . \tag{5.7}$$

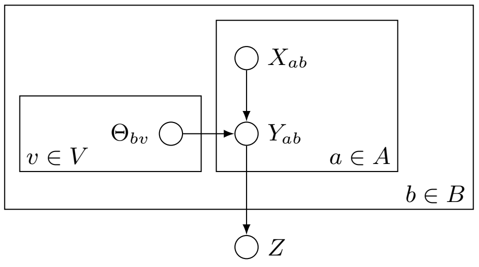
### 5.2.3 Probabilistic model

#### Random variables

- For any $(a, b) \in A \times B$, let $X_{ab}$ be a random variable whose realization is a vector $x_{ab} \in B \times \mathbb{R}^V$, called the *attribute vector* of $(a, b)$.
- For any $(a, b) \in A \times B$, let $Y_{ab}$ be a random variable whose realization is a binary number $y_{ab} \in \{0, 1\}$, called the *decision* of classifying $a$ as $b$
- For any $b \in B$ and any $v \in V$, let $\Theta_{bv}$ be a random variable whose realization is a real number $\theta_{bv} \in \mathbb{R}$, called a *parameter*
- Let $Z$ be a random variable whose realization is a subset $z \subseteq \{0, 1\}^{A \times B}$. For multiple label classification, we are interested in $z = \mathcal{Y}$, the set of the characteristic functions of all maps from $A$ to $B$.

#### Conditional independence assumptions

We assume a probability distribution that factorizes according to Bayesian net depicted below.



#### Factorization

These conditional independence assumptions imply the following factorizations:

- Firstly:

$$P(X, Y, Z, \Theta) = P(Z \mid Y) \prod_{(a,b) \in A \times B} P(Y_{ab} \mid X_{ab}, \Theta) \prod_{(b,v) \in B \times V} P(\Theta_{bv}) \prod_{(a,b) \in A \times B} P(X_{ab}) \tag{5.8}$$

- Secondly:

$$\begin{aligned} P(\Theta \mid X, Y, Z) &= \frac{P(X, Y, Z, \Theta)}{P(X, Y, Z)} \\ &= \frac{P(Z \mid Y)\, P(Y \mid X, \Theta)\, P(X)\, P(\Theta)}{P(Z \mid X, Y)\, P(X, Y)} \\ &= \frac{P(Z \mid Y)\, P(Y \mid X, \Theta)\, P(X)\, P(\Theta)}{P(Z \mid Y)\, P(X, Y)} \\ &= \frac{P(Y \mid X, \Theta)\, P(X)\, P(\Theta)}{P(X, Y)} \\ &\propto P(Y \mid X, \Theta)\, P(\Theta) \\ &= \prod_{(a,b) \in A \times B} P(Y_{ab} \mid X_{ab}, \Theta) \prod_{(b,v) \in B \times V} P(\Theta_{bv}) \end{aligned} \tag{5.9}$$

- Thirdly,

$$\begin{aligned}
P(Y \mid X, Z, \theta) &= \frac{P(X, Y, Z, \Theta)}{P(X, Z, \Theta)} \\
&= \frac{P(Z \mid Y)\, P(Y \mid X, \Theta)\, P(X)\, P(\Theta)}{P(X, Z, \Theta)} \\
&\propto P(Z \mid Y)\, P(Y \mid X, \Theta) \\
&= P(Z \mid Y) \prod_{(a,b) \in A \times B} P(Y_{ab} \mid X_{ab}, \Theta) \qquad (5.10)
\end{aligned}$$

**Forms**

Here, we consider:

- The *logistic distribution*

$$\forall (a,b) \in A \times B: \qquad p_{Y_{ab} \mid X_{ab}, \Theta}(1) = \frac{1}{1 + 2^{-f_\theta(x_{ab})}} \qquad (5.11)$$

- A $\sigma \in \mathbb{R}^+$ and the *normal distribution*:

$$\forall (b,v) \in B \times V: \qquad p_{\Theta_{bv}}(\theta_{bv}) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\theta_{bv}^2 / 2\sigma^2} \qquad (5.12)$$

- A uniform distribution on a subset:

$$\forall z \subseteq \{0,1\}^{A \times B}: \quad p_{Z \mid Y}(z) \propto \begin{cases} 1 & \text{if } y \in z \\ 0 & \text{otherwise} \end{cases} \qquad (5.13)$$

  Note that $p_{Z|Y}(\mathcal{Y})$ is non-zero iff the relation $y^{-1}(1) \subseteq A \times B$ is a map.

### 5.2.4 Learning problem

**Lemma 7** *Estimating maximally probable parameters $\theta$, given attributes $x$ and decisions $y$, i.e.,*

$$\underset{\theta \in \mathbb{R}^{B \times V}}{\operatorname{argmax}} \quad p_{\Theta \mid X, Y}(\theta, x, y)$$

*is identical to the supervised learning problem w.r.t. $L$, $R$ and $\lambda$ such that*

$$\begin{aligned}
\forall r \in \mathbb{R}\ \forall \hat{y} \in \{0,1\}: \quad & L(r, \hat{y}) = -\hat{y} r + \log\left(1 + 2^r\right) && (5.14) \\
\forall \theta \in \Theta: \quad & R(\theta) = \|\theta\|_2^2 && (5.15) \\
& \lambda = \frac{\log e}{2\sigma^2} && (5.16)
\end{aligned}$$

*Moreover, this problem separates into $|B|$ independent supervised learning problems, each w.r.t. parameters in $\mathbb{R}^V$, with $L$ and $\lambda$ as above, and with*

$$\forall \theta' \in \mathbb{R}^V: \qquad R'(\theta') = \|\theta'\|_2^2 \qquad (5.17)$$

Proof Analogous to the case of binary classification from Section 3.3, we now obtain:

$$\begin{aligned}
& \underset{\theta \in \mathbb{R}^{B \times V}}{\operatorname{argmax}} \quad p_{\Theta \mid X, Y}(\theta, x, y) \\
= & \underset{\theta \in \mathbb{R}^{B \times V}}{\operatorname{argmin}} \quad \sum_{(a,b) \in A \times B} \Big( -y_{ab} f_\theta(x_{ab}) + \log\left(1 + 2^{f_\theta(x_{ab})}\right) \Big) + \frac{\log e}{2\sigma^2} \|\theta\|_2^2 \ . \qquad (5.18)
\end{aligned}$$

Consider the unique $x' : A \times B \to \mathbb{R}^V$ such that, for any $(a,b) \in A \times B$, we have $x_{ab} = (b, x'_{ab})$.

Problem (5.18) separates into $|B|$ many $l_2$-regularized logistic regression problems, one for each $b \in B$, because

$$\begin{aligned}
&\min_{\theta \in \mathbb{R}^{B \times V}} \sum_{(a,b) \in A \times B} \left( -y_{ab} \langle \theta_{b\cdot}, x'_{ab} \rangle + \log \left( 1 + 2^{\langle \theta_{b\cdot}, x'_{ab} \rangle} \right) \right) + \frac{\log e}{2\sigma^2} \|\theta\|_2^2 \\
=& \min_{\theta \in \mathbb{R}^{B \times V}} \sum_{b \in B} \left( \sum_{a \in A} \left( -y_{ab} \langle \theta_{b\cdot}, x'_{ab} \rangle + \log \left( 1 + 2^{\langle \theta_{b\cdot}, x'_{ab} \rangle} \right) \right) + \frac{\log e}{2\sigma^2} \|\theta_{b\cdot}\|_2^2 \right) \\
=& \sum_{b \in B} \min_{\theta_{b\cdot} \in \mathbb{R}^V} \left( \sum_{a \in A} \left( -y_{ab} \langle \theta_{b\cdot}, x'_{ab} \rangle + \log \left( 1 + 2^{\langle \theta_{b\cdot}, x'_{ab} \rangle} \right) \right) + \frac{\log e}{2\sigma^2} \|\theta_{b\cdot}\|_2^2 \right) \quad .
\end{aligned}$$

### 5.2.5 Inference problem

**Lemma 8** *For any constrained data as defined above, any $\theta \in \mathbb{R}^{B \times V}$ and any $\hat{y} : A \times B \to \{0,1\}$, $\hat{y}$ is a solution to the inference problem*

$$\min_{y \in \mathcal{Y}} \sum_{(a,b) \in A \times B} L(f_\theta(x_{ab}), y_{ab}) \tag{5.19}$$

*iff there exists an $\varphi : A \to B$ such that*

$$\forall a \in A: \quad \varphi(a) \in \max_{b \in B} \langle \theta_{b\cdot}, x'_{ab} \rangle \tag{5.20}$$

*and*

$$\forall (a,b) \in A \times B: \quad \hat{y}_{ab} = 1 \Leftrightarrow \varphi(a) = b \quad . \tag{5.21}$$

Proof

$$\begin{aligned}
& \sum_{(a,b) \in A \times B} L(f_\theta(x_{ab}), y_{ab}) \\
=& \sum_{(a,b) \in A \times B} \left( L(f_\theta(x_{ab}), 1)\, y_{ab} + L(f_\theta(x_{ab}), 0)\, (1 - y_{ab}) \right) \\
=& \sum_{(a,b) \in A \times B} \left( L(f_\theta(x_{ab}), 1) - L(f_\theta(x_{ab}), 0) \right) y_{ab} + \text{const.} \\
=& \sum_{(a,b) \in A \times B} \left( -f_\theta(x_{ab}) \right) y_{ab} && \text{by (5.14)} \\
=& \sum_{(a,b) \in A \times B} \left( -\langle \theta_{b\cdot}, x'_{ab} \rangle \right) y_{ab} && x_{ab} = (b, x'_{ab}) \\
=& \sum_{a \in A} \sum_{b \in B} \left( -\langle \theta_{b\cdot}, x'_{ab} \rangle \right) y_{ab}
\end{aligned}$$

### 5.2.6 Inference algorithm

The inference problem is solved by solving (5.20) independently for each $a \in A$. The time complexity is $O(|A||B||V|)$.

# Chapter 6

# Partitioning

## 6.1 Partitions and equivalence relations

Throughout this chapter, we consider some finite set $A \neq \varnothing$ that we seek to partition into subsets. Hence, our feasible solutions are the *partitions* of $A$. A partition $\Pi$ of $A$ is a collection $\Pi \subseteq 2^A$ of non-empty and pairwise disjoint subsets of $A$ whose union is $A$.

The partitions of $A$ are characterized by the equivalence relations on $A$. An equivalence relation $\equiv$ on $A$ is a binary relation $\equiv \subseteq A \times A$ that is reflexive, symmetric and transitive. For any partition $\Pi$ of $A$, the equivalence relation $\equiv_\Pi$ induced by $\Pi$ is such that

$$\forall a, a' \in A\colon \quad a \equiv_\Pi a' \;\Leftrightarrow\; \exists U \in \Pi\colon a \in U \wedge a' \in U \ . \tag{6.1}$$

In turn, the equivalence relations on $A$ are characterized by those $y : \binom{A}{2} \to \{0,1\}$ that satisfy

$$\forall \{a,b,c\} \in \tbinom{A}{3}\colon \quad y_{\{a,b\}} + y_{\{b,c\}} - 1 \leq y_{\{a,c\}} \ . \tag{6.2}$$

For any partition $\Pi$ of $A$, consider the $y^\Pi : \binom{A}{2} \to \{0,1\}$ such that

$$\forall \{a,a'\} \in \tbinom{A}{2}\colon \quad y^\Pi_{\{a,a'\}} = 1 \;\Leftrightarrow\; \exists U \in \Pi\colon a \in U \wedge a' \in U \ . \tag{6.3}$$

Herein, for any $\{a,b\} \in \binom{A}{2}$, the decision $y_{\{a,b\}} = 1$ means that $a$ and $b$ are in the same subset, whereas the decision $y_{\{a,b\}} = 0$ means that $a$ and $b$ in distinct subsets.

We reduce the problem of learning and inferring partitions to the problem of learning and inferring decisions. This is done by choosing constrained data with

$$S = \tbinom{A}{2} \tag{6.4}$$

$$\mathcal{Y} = \left\{ y : \tbinom{A}{2} \to \{0,1\} \;\middle|\; (6.2) \right\} \ . \tag{6.5}$$

## 6.2 Linear functions

### 6.2.1 Data

Throughout Section 6.2, we consider some finite set $V \neq \varnothing$ and constrained data $(S, X, x, \mathcal{Y})$ with $S = \binom{A}{2}$ as in (6.4), $X = \mathbb{R}^V$ and $\mathcal{Y}$ as in (6.5). As a special case, we consider labeled data, i.e., just one $\mathcal{Y} = \{y\}$ with $y$ satisfying the constraints (6.2).

### 6.2.2 Familiy of functions

Throughout Section 6.2, we consider linear functions. More specifically, we consider $\Theta = \mathbb{R}^V$ and $f : \Theta \to \mathbb{R}^X$ such that

$$\forall \theta \in \Theta \; \forall \hat{x} \in \mathbb{R}^V\colon \quad f_\theta(\hat{x}) = \langle \theta, \hat{x} \rangle \ . \tag{6.6}$$

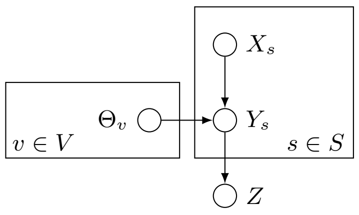
### 6.2.3 Probabilistic model

#### Random variables

- For any $\{a,a'\} \in S$, let $X_{\{a,a'\}}$ be a random variable whose realization is a vector $x_{\{a,a'\}} \in \mathbb{R}^V$, called the *attribute vector* of the pair $\{a,a'\}$.
- For any $\{a,a'\} \in S$, let $Y_{\{a,a'\}}$ be a random variable whose realization is a binary number $y_{\{a,a'\}} \in \{0,1\}$, called the *decision* of joining $a$ and $a'$ in the same subset.
- For any $v \in V$, let $\Theta_v$ be a random variable whose realization is a real number $\theta_v \in \mathbb{R}$, called a *parameter*
- Let $Z$ be a random variable whose realization is a subset $z \subseteq \{0,1\}^S$. We are interested in $z = \mathcal{Y}$, a characterization of all partitions of $A$.

#### Conditional independence assumptions

We assume a probability distribution that factorizes according to the Bayesian net depicted below.



#### Factorization

These conditional independence assumptions imply the following factorizations:

- Firstly:

$$P(X,Y,Z,\Theta) = P(Z \mid Y) \prod_{s \in S} P(Y_s \mid X_s, \Theta) \prod_{s \in S} P(X_s) \prod_{v \in V} P(\Theta_v) \tag{6.7}$$

- Secondly:

$$\begin{aligned}
P(\Theta \mid X,Y,Z) &= \frac{P(X,Y,Z,\Theta)}{P(X,Y,Z)} \\
&= \frac{P(Z \mid Y)\, P(Y \mid X,\Theta)\, P(X)\, P(\Theta)}{P(Z \mid X,Y)\, P(X,Y)} \\
&= \frac{P(Z \mid Y)\, P(Y \mid X,\Theta)\, P(X)\, P(\Theta)}{P(Z \mid Y)\, P(X,Y)} \\
&= \frac{P(Y \mid X,\Theta)\, P(X)\, P(\Theta)}{P(X,Y)} \\
&\propto P(Y \mid X,\Theta)\, P(\Theta) \\
&= \prod_{s \in S} P(Y_s \mid X_s, \Theta) \prod_{v \in V} P(\Theta_v)
\end{aligned} \tag{6.8}$$

- Thirdly,

$$\begin{aligned}
P(Y \mid X, Z, \theta) &= \frac{P(X, Y, Z, \Theta)}{P(X, Z, \Theta)} \\
&= \frac{P(Z \mid Y)\, P(Y \mid X, \Theta)\, P(X)\, P(\Theta)}{P(X, Z, \Theta)} \\
&\propto P(Z \mid Y)\, P(Y \mid X, \Theta) \\
&= P(Z \mid Y) \prod_{s \in S} P(Y_s \mid X_s, \Theta) \qquad (6.9)
\end{aligned}$$

**Forms**

Here, we consider:

- The *logistic distribution*

$$\forall s \in S: \qquad p_{Y_s|X_s,\Theta}(1) = \frac{1}{1 + 2^{-f_\theta(x_s)}} \qquad (6.10)$$

- A $\sigma \in \mathbb{R}^+$ and the *normal distribution*:

$$\forall v \in V: \qquad p_{\Theta_v}(\theta_v) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\theta_v^2/2\sigma^2} \qquad (6.11)$$

- A uniform distribution on a subset:

$$\forall z \subseteq \{0,1\}^S: \quad p_{Z|Y}(z) \propto \begin{cases} 1 & \text{if } y \in z \\ 0 & \text{otherwise} \end{cases} \qquad (6.12)$$

  Note that $p_{Z|Y}(\mathcal{Y})$ is non-zero iff $y$ induces an equivalence relations and hence a partition of $A$.

### 6.2.4 Learning problem

**Corollary 2** *Estimating maximally probable parameters $\theta$, given attributes $x$ and labels $y$, i.e.,*

$$\operatorname*{argmax}_{\theta \in \mathbb{R}^m} \quad p_{\Theta|X,Y}(\theta, x, y)$$

*is identical to the supervised learning problem w.r.t. $L$, $R$ and $\lambda$ such that*

$$\begin{aligned}
\forall r \in \mathbb{R}\ \forall \hat{y} \in \{0,1\}: \quad L(r, \hat{y}) &= -\hat{y}r + \log(1 + 2^r) && (6.13) \\
\forall \theta \in \Theta: \quad R(\theta) &= \|\theta\|_2^2 && (6.14) \\
\lambda &= \frac{\log e}{2\sigma^2} && (6.15)
\end{aligned}$$

### 6.2.5 Inference problem

**Corollary 3** *For any constrained data as defined above and any $\theta \in \mathbb{R}^V$, the inference problem has the form of* SET-PARTITION, *i.e.*

$$\min_{y \colon \binom{A}{2} \to \{0,1\}} \quad \sum_{\{a,a'\} \in S} \left(-\langle \theta, x_{\{a,a'\}} \rangle\right) y_{\{a,a'\}} \qquad (6.16)$$

$$\text{subject to} \quad \forall \{a,b,c\} \in \tbinom{A}{3}: \quad y_{\{a,b\}} + y_{\{b,c\}} - 1 \le y_{\{a,c\}} \ . \qquad (6.17)$$

SET-PARTITION has been studied intensively, notably by Chopra and Rao (1993), Bansal et al. (2004) and Demaine et al. (2006).

**Lemma 9** SET-PARTITION *is* NP-*hard.*

### 6.2.6 Inference algorithm

Below, we discuss three local search algorithms for SET-PARTITION. For simplicity, we define $c : S \to \mathbb{R}$ such that

$$\forall \{a, a'\} \in S \colon \quad c_{\{aa'\}} = -\langle \theta, x_{\{a,a'\}} \rangle \ , \tag{6.18}$$

and write the objective function $\varphi : \{0,1\}^S \to \mathbb{R}$ such that

$$\forall y \in \{0,1\}^S \colon \quad \varphi(y) = \sum_{\{a,a'\} \in S} c_{\{a,a'\}} \, y_{\{a,a'\}} \tag{6.19}$$

#### Greedy joining

The greedy joining algorithm starts from any initial partition and searches for partitions with lower objective value by joining pairs of subsets recursively. As subsets can only grow and the number of subsets decreases by precisely one in every step, one typically starts from the finest partition $\Pi_0$ of $A$ into one-elementary subsets.

**Definition 18** For any partition $\Pi$ of $A$, and any $B, C \in \Pi$, let $\text{join}_{BC}[\Pi]$ be the partition of $A$ obtained by joining the sets $B$ and $C$ in $\Pi$, i.e.

$$\text{join}_{BC}[\Pi] = (\Pi \setminus \{B, C\}) \cup \{B \cup C\} \ . \tag{6.20}$$

**Algorithm 1** The greedy joining algorithm is defined by the recursion below.

```
$\Pi' = \text{greedy-joining}(\Pi)$
choose $\{B, C\} \in \operatorname*{argmin}_{\{B',C'\} \in \binom{\Pi}{2}} \varphi(y^{\text{join}_{B'C'}[\Pi]}) - \varphi(y^{\Pi})$
if $\varphi(y^{\text{join}_{BC}[\Pi]}) - \varphi(y^{\Pi}) < 0$
    $\Pi' := \text{greedy-joining}(\text{join}_{BC}[\Pi])$
else
    $\Pi' := \Pi$
```

**Exercise 3** *a) Write the difference $\varphi(y^{\text{join}_{B'C'}[\Pi]}) - \varphi(y^{\Pi})$ in terms of the $c$ defined in* (7.16).
*b) Implement greedy joining efficiently.*
*c) Establish a bound on the time complexity of your implementation.*

#### Greedy moving

The greedy moving algorithm starts from any initial partition, e.g., the fixed point of greedy joining. It seeks to lower the objective function by recursively moving individual elements from one subset to another, or to a new subset. When an element is moved to a new subset, the number of subsets in the partition increases. When the last element is moved from a subset, the number of subsets in the partition decreases.

**Definition 19** For any partition $\Pi$ of $A$, any $a \in A$ and any $U \in \Pi \cup \{\emptyset\}$, let $\text{move}_{aU}[\Pi]$ the partition of $A$ obtained by moving the element $a$ to a subset $U \cup \{a\}$ in $\Pi$.

$$\text{move}_{aU}[\Pi] = \Pi \setminus \{U\} \setminus \{W \in \Pi \mid a \in W\} \cup \{U \cup \{a\}\} \cup \bigcup_{\{W \in \Pi \mid a \in W \wedge \{a\} \neq W\}} \{W \setminus \{a\}\} \ . \tag{6.21}$$

**Algorithm 2** The greedy moving algorithm is defined by the recursion below.

```
$\Pi' = \text{greedy-moving}(\Pi)$

choose $(a, U) \in \operatorname*{argmin}_{(a',U') \in A \times (\Pi \cup \{\varnothing\})} \varphi(y^{\text{move}_{a'U'}[\Pi]}) - \varphi(y^{\Pi})$
if $\varphi(y^{\text{move}_{aU}[\Pi]}) - \varphi(y^{\Pi}) < 0$
    $\Pi' := \text{greedy-moving}(\text{move}_{aU}[\Pi])$
else
    $\Pi' := \Pi$
```

**Exercise 4** *a) Write the difference $\varphi(y^{\text{move}_{aU}[\Pi_t]}) - \varphi(y^{\Pi_t})$ in terms of the $c$ defined in* (7.16).
*b) Implement greedy moving.*

**Greedy moving using the technique of Kernighan and Lin (1970)**

Both algorithms discussed above terminate as soon as no transformation (join and move, resp.) leads to a partition with strictly lower objective value. This can be sub-optimal in case transformations that increase the objective value at one point in the recursion can lead to transformations that decrease the objective value at later points in the recursion and the decrease overcompensates the increase. A generalization of greedy local search introduced by Kernighan and Lin (1970) can escape such sub-optimal fixed points. It is applied to greedy moving below.

**Algorithm 3** The greedy moving algorithm generalized by the technique of Kernighan and Lin (1970) is defined by the recursion below.

```
$\Pi' = \text{greedy-moving-kl}(\Pi)$

$\Pi_0 := \Pi$
$\delta_0 := 0$
$A_0 := A$
$t := 0$
repeat                                                        (build sequence of moves)
    choose $(a_t, U_t) \in \operatorname*{argmin}_{(a,U) \in A_t \times (\Pi \cup \{\varnothing\})} \varphi(y^{\text{move}_{aU}[\Pi_t]}) - \varphi(y^{\Pi_t})$
    $\Pi_{t+1} := \text{move}_{a_t U_t}[\Pi_t]$
    $\delta_{t+1} := \varphi(y^{\Pi_{t+1}}) - \varphi(y^{\Pi_t}) < 0$
    $A_{t+1} := A_t \setminus \{a_t\}$                                    (move $a_t$ only once)
    $t := t + 1$
until $A_t = \varnothing$
$\hat{t} := \min \operatorname*{argmin}_{t' \in \{0,\ldots,|A|\}} \sum_{\tau=0}^{t'} \delta_\tau$                          (choose sub-sequence)
if $\sum_{\tau=0}^{\hat{t}} \delta_\tau < 0$
    $\Pi' := \text{greedy-moving-kl}(\Pi_{\hat{t}})$                                      (recurse)
else
    $\Pi' := \Pi$                                                         (terminate)
```

**Exercise 5** *a) Implement greedy moving using the technique of Kernighan and Lin (1970).*
*b) Generalize the greedy joining algorithm using the technique of Kernighan and Lin (1970).*
*c) Implement greedy joining using the technique of Kernighan and Lin (1970).*

# Chapter 7

# Clustering

## 7.1 Decompositions and multicuts

This section is concerned with learning and inferring decompositions (clusterings) of a graph. We introduce some terminology of Horňáková et al. (2017):

**Definition 20** Let $G = (A, E)$ be any graph. A subgraph $G' = (A', E')$ of $G$ is called a *component* of $G$ iff $G'$ is non-empty, node-induced[1] and connected[2]. A partition $\Pi$ of the node set $A$ is called a *decomposition* of $G$ iff, for every $U \in \Pi$, the subgraph $(U, E \cap \binom{U}{2})$ of $G$ induced by $U$ is connected (and thus a component of $G$).

For any graph $G$, we denote by $D_G$ the set of all decompositions of $G$. Useful in the study of decompositions are the multicuts of a graph:

**Definition 21** For any graph $G = (A, E)$, a subset $M \subseteq E$ of edges is called a *multicut* of $G$ iff, for every cycle $C \subseteq E$ of $G$, we have $|C \cap M| \neq 1$.

For any graph $G$, we denote by $M_G$ the set of all multicuts of $G$. For any decomposition of a graph $G$, the set of those edges that straddle distinct components is a multicut of $G$. This multicut is said to be induced by the decomposition. In fact, the map from decompositions to induced multicuts is a bijection from $D_G$ to $M_G$ (Horňáková et al., 2017, Lemma 2). This bijection allows us to state the problem of learning and inferring decompositions as one of learning and inferring multicuts.

The characteristic function $y \colon E \to \{0, 1\}$ of a multicut $y^{-1}(1)$ decides, for every edge $\{a, a'\} = e \in E$, whether the incident nodes belong to the same component ($y_e = 0$) or distinct components ($y_e = 1$). By the definition of a multicut, these decisions are not necessarily independent. More specifically:

**Lemma 10** *For any graph $G = (V, E)$ and any $y \colon E \to \{0, 1\}$, the set $y^{-1}(1)$ of those edges that are mapped to 1 is a multicut of $G$ iff the following inequalities are satisfied:*

$$\forall C \in \text{cycles}(G)\ \forall e \in C\colon \quad y_e \leq \sum_{e' \in C \setminus \{e\}} y_{e'} \tag{7.1}$$

**Exercise 6** *a) Prove Lemma 10.*

*b) Show that it is sufficient in* (7.1) *to consider only chordless cycles.*

[1] I.e. $E' = E \cap \binom{A'}{2}$

[2] A component is not necessarily maximal w.r.t. the subgraph relation.

Now that we have a finite set $E$, decisions $y\colon E \to \{0,1\}$ and constraints (7.1), we can state the problem of learning and inferring multicuts as a learning and inference problem (4.1) with

$$S = E \tag{7.2}$$

$$\mathcal{Y} = \left\{ y\colon S \to \{0,1\} \;\middle|\; \forall C \in \text{cycles}(G)\ \forall e \in C\colon y_e \leq \sum_{e' \in C \setminus \{e\}} y_{e'} \right\} \tag{7.3}$$

## 7.2 Linear functions

### 7.2.1 Data

Throughout Section 7.2, we consider some graph $G = (A, E)$ and constrained data $(S, X, x, \mathcal{Y})$ with $S = E$, as in (7.2), $\mathcal{Y}$ defined as in (7.3), and $X = \mathbb{R}^V$ with some finite, non-empty set $V$. As a special case, we consider labeled data, i.e., $\mathcal{Y} = \{y\}$ with $y$ satisfying the constraints (7.1).

### 7.2.2 Familiy of functions

Throughout Section 7.2, we consider linear functions. More specifically, we consider $\Theta = \mathbb{R}^V$ and $f : \Theta \to \mathbb{R}^X$ such that

$$\forall \theta \in \Theta\ \forall \hat{x} \in \mathbb{R}^V\colon \quad f_\theta(\hat{x}) = \langle \theta, \hat{x} \rangle\ . \tag{7.4}$$

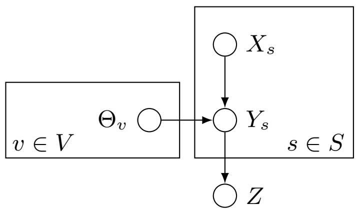
### 7.2.3 Probabilistic model

**Random variables**

- For any $\{a, a'\} \in S$, let $X_{\{a,a'\}}$ be a random variable whose realization is a vector $x_{\{a,a'\}} \in \mathbb{R}^V$, called the *attribute vector* of the pair $\{a, a'\}$.
- For any $\{a, a'\} \in S$, let $Y_{\{a,a'\}}$ be a random variable whose realization is a binary number $y_{\{a,a'\}} \in \{0,1\}$, called the *decision* of assigning $a$ and $a'$ to distinct components
- For any $v \in V$, let $\Theta_v$ be a random variable whose realization is a real number $\theta_v \in \mathbb{R}$, called a *parameter*
- Let $Z$ be a random variable whose realization is a subset $z \subseteq \{0,1\}^S$. We are interested in $z = \mathcal{Y}$, a characterization of all multicuts (and hence, decompositions) of $G$

**Conditional independence assumptions**

We assume a probability distribution that factorizes according to the Bayesian net depicted below.



**Factorization**

These conditional independence assumptions imply the following factorizations:

- Firstly:

$$P(X, Y, Z, \Theta) = P(Z \mid Y) \prod_{s \in S} P(Y_s \mid X_s, \Theta) \prod_{s \in S} P(X_s) \prod_{v \in V} P(\Theta_v) \tag{7.5}$$

- Secondly:

$$\begin{aligned}
P(\Theta \mid X, Y, Z) &= \frac{P(X, Y, Z, \Theta)}{P(X, Y, Z)} \\
&= \frac{P(Z \mid Y)\, P(Y \mid X, \Theta)\, P(X)\, P(\Theta)}{P(Z \mid X, Y)\, P(X, Y)} \\
&= \frac{P(Z \mid Y)\, P(Y \mid X, \Theta)\, P(X)\, P(\Theta)}{P(Z \mid Y)\, P(X, Y)} \\
&= \frac{P(Y \mid X, \Theta)\, P(X)\, P(\Theta)}{P(X, Y)} \\
&\propto P(Y \mid X, \Theta)\, P(\Theta) \\
&= \prod_{s \in S} P(Y_s \mid X_s, \Theta) \prod_{v \in V} P(\Theta_v) \qquad (7.6)
\end{aligned}$$

- Thirdly,

$$\begin{aligned}
P(Y \mid X, Z, \theta) &= \frac{P(X, Y, Z, \Theta)}{P(X, Z, \Theta)} \\
&= \frac{P(Z \mid Y)\, P(Y \mid X, \Theta)\, P(X)\, P(\Theta)}{P(X, Z, \Theta)} \\
&\propto P(Z \mid Y)\, P(Y \mid X, \Theta) \\
&= P(Z \mid Y) \prod_{s \in S} P(Y_s \mid X_s, \Theta) \qquad (7.7)
\end{aligned}$$

**Forms**

Here, we consider:

- The *logistic distribution*

$$\forall s \in S: \qquad p_{Y_s \mid X_s, \Theta}(1) = \frac{1}{1 + 2^{-f_\theta(x_s)}} \qquad (7.8)$$

- A $\sigma \in \mathbb{R}^+$ and the *normal distribution*:

$$\forall v \in V: \qquad p_{\Theta_v}(\theta_v) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\theta_v^2/2\sigma^2} \qquad (7.9)$$

- A uniform distribution on a subset:

$$\forall z \subseteq \{0,1\}^S: \quad p_{Z \mid Y}(z) \propto \begin{cases} 1 & \text{if } y \in z \\ 0 & \text{otherwise} \end{cases} \qquad (7.10)$$

  Note that $p_{Z|Y}(\mathcal{Y})$ is non-zero iff $y^{-1}(1)$ is a multicut and hence defines a decomposition of $G$.

### 7.2.4 Learning problem

**Corollary 4** *Estimating maximally probable parameters $\theta$, given attributes $x$ and labels $y$, i.e.,*

$$\operatorname*{argmax}_{\theta \in \mathbb{R}^m} \quad p_{\Theta \mid X, Y}(\theta, x, y)$$

*is identical to the supervised learning problem w.r.t. $L$, $R$ and $\lambda$ such that*

$$\begin{aligned}
\forall r \in \mathbb{R}\ \forall \hat{y} \in \{0,1\}: &\quad L(r, \hat{y}) = -\hat{y} r + \log\left(1 + 2^r\right) \qquad (7.11) \\
\forall \theta \in \Theta: &\quad R(\theta) = \|\theta\|_2^2 \qquad (7.12) \\
&\quad \lambda = \frac{\log e}{2\sigma^2} \qquad (7.13)
\end{aligned}$$

### 7.2.5 Inference problem

**Corollary 5** *For any constrained data as defined above and any $\theta \in \mathbb{R}^V$, the inference problem has the form of* CORRELATION-CLUSTERING, *i.e.*

$$\min_{y\colon S \to \{0,1\}} \sum_{\{a,a'\} \in S} (-\langle \theta, x_{\{a,a'\}} \rangle)\, y_{\{a,a'\}} \tag{7.14}$$

$$\text{subject to} \quad \forall C \in \text{cycles}(G)\ \forall e \in C\colon \quad y_e \leq \sum_{e' \in C \setminus \{e\}} y_{e'} \ . \tag{7.15}$$

CORRELATION-CLUSTERING has been studied intensively, notably by Chopra and Rao (1993), Bansal et al. (2004) and Demaine et al. (2006).

**Lemma 11 (Bansal et al. (2004))** CORRELATION-CLUSTERING *is* NP-*hard.*

Bansal et al. (2004) establish NP-hardness of CORRELATION-CLUSTERING by a reduction of $k$-TERMINAL-CUT whose NP-hardness is an important result of Dahlhaus et al. (1994).

### 7.2.6 Inference algorithm

Below, we discuss three local search algorithms for CORRELATION-CLUSTERING. For simplicity, we define $c : S \to \mathbb{R}$ such that

$$\forall \{a,a'\} \in S\colon \quad c_{\{aa'\}} = -\langle \theta, x_{\{a,a'\}} \rangle \tag{7.16}$$

and write the objective function $\varphi : \{0,1\}^S \to \mathbb{R}$ such that

$$\forall y \in \{0,1\}^S\colon \quad \varphi(y) = \sum_{\{a,a'\} \in S} c_{\{a,a'\}}\, y_{\{a,a'\}} \tag{7.17}$$

#### Greedy joining

The greedy joining algorithm starts from any initial decomposition and searches for decompositions with lower objective value by joining pairs of components recursively. By this procedure, components can only grow, and the number of components decreases by precisely one in every step. Thus, one typically starts from the finest decomposition $\Pi_0$ of $G = (A, E)$ into one-elementary components.

**Definition 22** For any graph $G = (A, E)$ and any disjoint sets $B, C \subseteq A$, the pair $\{B, C\}$ is called *neighboring* in $G$ iff there exist nodes $b \in B$ and $c \in C$ such that $\{b, c\} \in E$.

For any decomposition $\Pi$ of a graph $G = (A, E)$, we define

$$\mathcal{E}_\Pi = \left\{ \{B, C\} \in \tbinom{\Pi}{2} \,\middle|\, \exists b \in B\, \exists c \in C\colon \{b, c\} \in E \right\} \ . \tag{7.18}$$

**Definition 23** For any decomposition $\Pi$ of $G = (A, E)$ and any $\{B, C\} \in \mathcal{E}_\Pi$, let $\text{join}_{BC}[\Pi]$ be the decomposition of $G$ obtained by joining the sets $B$ and $C$ in $\Pi$, i.e.

$$\text{join}_{BC}[\Pi] = (\Pi \setminus \{B, C\}) \cup \{B \cup C\} \ . \tag{7.19}$$

**Algorithm 4** The greedy joining algorithm is defined by the recursion below.

```
$\Pi' = \text{greedy-joining}(\Pi)$
------------------------------------------------------------
choose $\{B,C\} \in \operatorname{argmin}_{\{B',C'\} \in \mathcal{E}_\Pi} \varphi(y^{\text{join}_{B'C'}[\Pi]}) - \varphi(y^\Pi)$
if $\varphi(y^{\text{join}_{BC}[\Pi]}) - \varphi(y^\Pi) < 0$
    $\Pi' := \text{greedy-joining}(\text{join}_{BC}[\Pi])$
else
    $\Pi' := \Pi$
```

**Exercise 7** *a) Write the difference $\varphi(y^{\text{join}_{B'C'}[\Pi]}) - \varphi(y^\Pi)$ in terms of the $c$ defined in* (7.16).
*b) Implement greedy joining efficiently.*
*c) Establish a bound on the time complexity of your implementation.*

### Greedy moving

The greedy moving algorithm starts from any initial decomposition, e.g., the fixed point of greedy joining. It seeks to lower the objective value by recursively moving individual nodes from one component to a neighboring component, or to a new component. When a node is moved to a new component, the number of components can increase. When the last node is moved from a component, the number of components decreases.

**Definition 24** For any graph $G$, a component $G'$ of $G$ is called *maximal* iff there is no subgraph of $G$ that is both a strict supergraph of $G'$ and a component of $G$.

**Definition 25** For any graph $G = (A, E)$ and any decomposition $\Pi$ of $G$, the decomposition $\Pi$ is called *coarsest* iff, for every $U \in \Pi$, the component $(U, E \cap \binom{U}{2})$ induced by $U$ is maximal.

**Lemma 12** *For any graph $G$, the coarsest decomposition is unique.*

For any graph $G$, we denote the coarsest decomposition by $\Pi^*_G$.

**Definition 26** For any graph $G = (A, E)$, any decomposition $\Pi$ of $A$ and any $a \in A$, choose $U_a$ to be the unique $U_a \in \Pi$ such that $a \in U_a$, and let

$$\mathcal{N}_a = \{\varnothing\} \cup \{W \in \Pi \,|\, a \notin W \,\wedge\, \exists w \in W \colon \{a, w\} \in E\} \tag{7.20}$$

$$G_a = \Big(U_a \setminus \{a\},\, E \cap \tbinom{U_a \setminus \{a\}}{2}\Big) \tag{7.21}$$

For any $U \in \mathcal{N}_a$, let $\text{move}_{aU}[\Pi]$ the decomposition of $A$ obtained by moving the node $a$ to the set $U$, i.e.

$$\text{move}_{aU}[\Pi] = \Pi \setminus \{U_a, U\} \cup \{U \cup \{a\}\} \cup \Pi^*_{G_a} \quad . \tag{7.22}$$

**Algorithm 5** The greedy moving algorithm is defined by the recursion below.

```
$\Pi' = \text{greedy-moving}(\Pi)$
choose $(a, U) \in \operatorname{argmin}_{(a',U') \in A \times (\Pi \cup \{\varnothing\})} \varphi(y^{\text{move}_{a'U'}[\Pi]}) - \varphi(y^{\Pi})$
if $\varphi(y^{\text{move}_{aU}[\Pi]}) - \varphi(y^{\Pi}) < 0$
    $\Pi' := \text{greedy-moving}(\text{move}_{aU}[\Pi])$
else
    $\Pi' := \Pi$
```

**Exercise 8** *a) Write the difference $\varphi(y^{\text{move}_{aU}[\Pi_t]}) - \varphi(y^{\Pi_t})$ in terms of the $c$ defined in* (7.16).
*b) Implement greedy moving.*

### Greedy moving using the technique of Kernighan and Lin (1970)

Both algorithms discussed above terminate as soon as no transformation (join and move, resp.) leads to a partition with strictly lower objective value. This can be sub-optimal in case transformations that increase the objective value at one point in the recursion can lead to transformations that decrease the objective value at later points in the recursion and the decrease overcompensates the increase. A generalization of greedy local search introduced by Kernighan and Lin (1970) can escape such sub-optimal fixed points. It is applied to greedy moving below.

**Algorithm 6** The greedy moving algorithm generalized by the technique of Kernighan and Lin (1970) is defined by the recursion below.

```
$\Pi' = $ greedy-moving-kl$(\Pi)$
$\Pi_0 := \Pi$
$\delta_0 := 0$
$A_0 := A$
$t := 0$
repeat                                                           (build sequence of moves)
    choose $(a_t, U_t) \in \operatorname*{argmin}_{(a,U) \in A_t \times (\Pi \cup \{\varnothing\})} \varphi(y^{\mathrm{move}_{aU}[\Pi_t]}) - \varphi(y^{\Pi_t})$
    $\Pi_{t+1} := \mathrm{move}_{a_t U_t}[\Pi_t]$
    $\delta_{t+1} := \varphi(y^{\Pi_{t+1}}) - \varphi(y^{\Pi_t}) < 0$
    $A_{t+1} := A_t \setminus \{a_t\}$                                     (move $a_t$ only once)
    $t := t + 1$
until $A_t = \varnothing$
$\hat{t} := \min \operatorname*{argmin}_{t' \in \{0,\dots,|A|\}} \sum_{\tau=0}^{t'} \delta_\tau$                          (choose sub-sequence)
if $\sum_{\tau=0}^{\hat{t}} \delta_\tau < 0$
    $\Pi' := $ greedy-moving-kl$(\Pi_{\hat{t}})$                                     (recurse)
else
    $\Pi' := \Pi$                                                       (terminate)
```

**Exercise 9** *a) Implement greedy moving using the technique of Kernighan and Lin (1970).*
*b) Generalize the greedy joining algorithm using the technique of Kernighan and Lin (1970).*
*c) Implement greedy joining using the technique of Kernighan and Lin (1970).*

# Chapter 8

# Ordering

## 8.1 Orders

Throughout this chapter, we consider some finite set $A \neq \varnothing$ that we seek to order. Hence, our feasible solutions are *strict (total) orders* on $A$. A strict order on $A$ is a binary relation $< \subseteq A \times A$ that is trichotomous and transitive, i.e., it satisfies the following conditions:

$$\forall a \in A\colon \quad \neg a < a \tag{8.1}$$

$$\forall \{a,b\} \in \tbinom{A}{2}\colon \quad a < b \ \text{xor} \ b < a \tag{8.2}$$

$$\forall \{a,b,c\} \in \tbinom{A}{3}\colon \quad a < b \ \wedge \ b < c \ \Rightarrow \ a < c \tag{8.3}$$

On the one hand, the strict orders on $A$ are characterized by the bijections $\alpha : \{0, \ldots, |A|-1\} \to A$. For any such bijection, consider the order $<_\alpha$ such that

$$\forall a, b \in A\colon \quad a < b \ \Leftrightarrow \ \alpha^{-1}(a) < \alpha^{-1}(b) \ . \tag{8.4}$$

On the other hand, the strict orders on $A$ are characterized by those

$$y : \{(a,b) \in A \times A \mid a \neq b\} \to \{0,1\} \tag{8.5}$$

that satisfy the following conditions:

$$\forall \{a,b\} \in \tbinom{A}{2}\colon \quad y_{ab} + y_{ba} = 1 \tag{8.6}$$

$$\forall \{a,b,c\} \in \tbinom{A}{3}\colon \quad y_{ab} + y_{bc} - 1 \leq y_{ac} \tag{8.7}$$

We reduce the problem of learning and inferring orders to the problem of learning and inferring decisions, by choosing constrained data with

$$S = \{(a,b) \in A \times A \mid a \neq b\} \tag{8.8}$$

$$\mathcal{Y} = \left\{y \in \{0,1\}^S \;\middle|\; (8.6) \wedge (8.7)\right\} \ . \tag{8.9}$$

One can think of the set $S$ as the set of edges in the complete digraph with nodes $A$ and without self-edges.

## 8.2 Linear functions

### 8.2.1 Data

Throughout Section 8.2, we consider some finite set $A \neq \varnothing$ and constrained data $(S, X, x, \mathcal{Y})$ with $S = \{(a,b) \in A \times A \mid a \neq b\}$ as in (8.8), $X = \mathbb{R}^V$ and $\mathcal{Y}$ as in (8.9). As a special case, we consider labeled data, i.e., just one $\mathcal{Y} = \{y\}$ with $y$ satisfying the constraints (8.6) and (8.7).

### 8.2.2 Learning

The learning of linear orders is analogous to the learning of equivalence relations (Chapter 6).

### 8.2.3 Inference problem

**Corollary 6** *For any constrained data as defined above and any $\theta \in \mathbb{R}^V$, the inference problem takes the form of* LINEAR-ORDERING, *i.e.*

$$\min_{y \in \{0,1\}^S} \sum_{(a,b) \in S} (-\langle \theta, x_{ab} \rangle)\, y_{ab} \tag{8.10}$$

$$\text{subject to} \quad \forall \{a,b\} \in \tbinom{A}{2}: \quad y_{ab} + y_{ba} = 1 \tag{8.11}$$

$$\forall \{a,b,c\} \in \tbinom{A}{3}: \quad y_{ab} + y_{bc} - 1 \leq y_{ac} \ . \tag{8.12}$$

LINEAR-ORDERING has been studied intensively. A comprehensive survey is by Martí and Reinelt (2011). Important early research is by Grötschel et al. (1984).

**Lemma 13** LINEAR-ORDERING *is* NP-*hard.*

### 8.2.4 Inference algorithms

Below, we discuss two local search algorithms for the linear ordering problem, i.e., heuristics whose outputs need not be optimal.

For simplicity, we define $c : S \to \mathbb{R}$ such that

$$\forall (a,b) \in S: \quad c_{ab} = -\langle \theta, x_{ab} \rangle \ , \tag{8.13}$$

and write the objective function $\varphi : \{0,1\}^S \to \mathbb{R}$ such that

$$\forall y \in \{0,1\}^S: \quad \varphi(y) = \sum_{(a,b) \in S} c_{ab}\, y_{ab} \tag{8.14}$$

#### Greedy transposition

The greedy transposition algorithm starts from an initial strict order and searches for strict orders with lower objective value by swapping pairs of elements.

**Definition 27** For any bijection $\alpha : [|A|] \to A$ and any $j, k \in [|A|]$, let $\text{transpose}_{jk}[\alpha]$ the bijection obtained from $\alpha$ by swapping $\alpha_j$ and $\alpha_k$, i.e.

$$\forall l \in [|A|]: \quad \text{transpose}_{jk}[\alpha](l) = \begin{cases} \alpha_k & \text{if } l = j \\ \alpha_j & \text{if } l = k \\ \alpha_l & \text{otherwise} \end{cases} \ . \tag{8.15}$$

**Algorithm 7** The greedy transposition algorithm is defined by the recursion below.

```
$\alpha' = \text{greedy-transposition}(\alpha)$
choose $(j,k) \in \operatorname{argmin}_{0 \leq j' < k' \leq |A|} \varphi(y^{\text{transpose}_{j'k'}[\alpha]}) - \varphi(y^\alpha)$
if $\varphi(y^{\text{transpose}_{jk}[\alpha]}) - \varphi(y^\alpha) < 0$
    $\alpha' := \text{greedy-transposition}(\text{transpose}_{jk}[\alpha])$
else
    $\alpha' := \alpha$
```

**Exercise 10** *a) Write the difference $\varphi(y^{\text{transpose}_{j'k'}[\alpha]}) - \varphi(y^\alpha)$ in terms of the $c$ defined in (8.13).*
*b) Implement greedy transposition.*

### Greedy transposition using the technique of Kernighan and Lin (1970)

As in the case of greedy heuristics for the set partition problem, the greedy transposition algorithm for the linear ordering problem has been generalized using the idea of Kernighan and Lin (1970).

**Algorithm 8 (Martí and Reinelt (2011))** The greedy transposition algorithm generalized by the technique of Kernighan and Lin (1970) is defined by the recursion below.

```
$\alpha' = \text{greedy-transposition-kl}(\alpha)$
$\alpha^0 := \alpha$
$\delta_0 := 0$
$J_0 := [|A|]$
$t := 0$
repeat                                                    (build sequence of swaps)
    choose $(j,k) \in \operatorname*{argmin}_{\{(j',k') \in J^2 | j' < k'\}} \varphi(y^{\text{transpose}_{j'k'}[\alpha^t]}) - \varphi(y^{\alpha^t})$
    $\alpha^{t+1} := \text{transpose}_{jk}[\alpha_t]$
    $\delta_{t+1} := \varphi(y^{\alpha^{t+1}}) - \varphi(y^{\alpha^t}) < 0$
    $J_{t+1} := J_t \setminus \{j,k\}$                          (move $\alpha_j$ and $\alpha_k$ only once)
    $t := t+1$
until $|J_t| < 2$
$\hat{t} := \min \operatorname*{argmin}_{t' \in \{0,\ldots,|A|\}} \sum_{\tau=0}^{t'} \delta_\tau$                    (choose sub-sequence)
if $\sum_{\tau=0}^{\hat{t}} \delta_\tau < 0$
    $\alpha' := \text{greedy-transposition-kl}(\alpha^{\hat{t}})$                      (recurse)
else
    $\alpha' := \alpha$                                               (terminate)
```

**Exercise 11** *Implement greedy transposition using the technique of Kernighan and Lin (1970).*

# Chapter 9

# Supervised structured learning

## 9.1 Intuition

Even in the most general learning and inference problem w.r.t. constrained data $(S, X, x, \mathcal{Y})$ we have considered so far, attributes $x_s \in X$ are defined for single elements $s \in S$ only, and solutions are such that decisions $y_s, y_{s'} \in \{0, 1\}$ for distinct $s, s' \in S$ are independent unless they are tied by constraints of a feasible set $\mathcal{Y} \subset \{0, 1\}^S$.

This mathematical abstraction of learning is too restrictive for certain applications. For example, consider a task where we are given a digital image and need to decide for every pixel $s \in S$, by the contents of the image around that pixel, whether the pixel is of interest ($y_s = 1$) or not of interest ($y_s = 0$). Typically, decisions at neighboring pixels $s, s' \in S$ are more likely to be equal ($y_s = y_{s'}$) than unequal ($y_s \neq y_{s'}$), and we may wish to learn how this increased probability depends on the contents of the image. None of the mathematical abstractions of learning we have considered so far is sufficient to express this dependency.

In order to lift this restriction, we will now define a supervised learning problem as well as an inference problem in which attributes are associated with subsets of $S$, and in which decisions can be tied by probabilistic dependencies. Therefor, we will introduce a family $H : \Theta \to \mathbb{R}^{X \times Y}$ of functions that quantify by $H_\theta(x, y)$ how incompatible attributes $x \in X$ are with a combination of decisions $y \in \{0, 1\}^S$. We will define supervised structured learning as a problem of finding one function from this family. We will define structured inference as the problem of finding a combination of decisions $y \in \{0, 1\}^S$ that minimizes $H_\theta(x, \cdot)$.

## 9.2 Definition

**Definition 28** A triple $(S, F, E)$ is called a *factor graph* with *variable nodes* $S$ and *factor nodes* $F$ iff $S \cap F = \varnothing$ and $(S \cup F, E)$ is a bipartite graph such that $\forall e \in E \, \exists s \in S \, \exists f \in F \colon e = \{s, f\}$.

For any factor node $f \in F$, we denote by $S_f = \{s \in S \mid \{s, f\} \in E\}$ the set of those variable nodes that are neighbors of $f$. For any variable node $s \in S$, we denote by $F_s = \{f \in F \mid \{s, f\} \in E\}$ the set of those factor nodes that are neighbors of $s$.

**Definition 29** A tuple $T = (S, F, E, \{X_f\}_{f \in F}, x)$ is called *unlabeled structured data* iff $(S, F, E)$ is a factor graph, every set $X_f$ is non-empty, called the *attribute space* of $f$, and $x \in \prod_{f \in F} X_f$, where the Cartesian product $\prod_{f \in F} X_f$ is called the *attribute space* of $T$. A tuple $(S, F, E, \{X_f\}_{f \in F}, x, y)$ is called *labeled structured data* iff $(S, F, E, \{X_f\}_{f \in F}, x)$ is unlabeled structured data, and $y \in \{0, 1\}^S$.

**Definition 30** For any labeled structured data $T = (S, F, E, \{X_f\}_{f \in F}, x, y)$, the attribute space $X = \prod_{f \in F} X_f$, the set $Y = \{0, 1\}^S$, any $\Theta \neq \varnothing$ and family of functions $H : \Theta \to \mathbb{R}^{X \times Y}$, any $R : \Theta \to \mathbb{R}_0^+$, called a *regularizer*, any $L : \mathbb{R}^Y \times Y \to \mathbb{R}_0^+$, called a *loss function*, and any

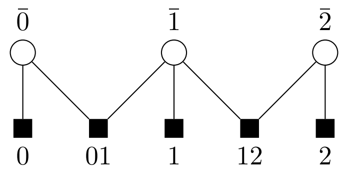


Figure 9.1: The factor graph with $S = \{\bar{0}, \bar{1}, \bar{2}\}$ and $F = \{0, 1, 2, 01, 12\}$ depicted above makes explicit that a function $H : \{0,1\}^S \to \mathbb{R}$ factorizes according to $H(y) = h_0(y_{\bar{0}}) + h_1(y_{\bar{1}}) + h_2(y_{\bar{2}}) + h_{01}(y_{\bar{0}}, y_{\bar{1}}) + h_{12}(y_{\bar{1}}, y_{\bar{2}})$.

$\lambda \in \mathbb{R}_0^+$, called a *regularization parameter*, the instance of the *supervised structured learning problem* w.r.t. $T, \Theta, H, R, L$ and $\lambda$ is defined as

$$\inf_{\theta \in \Theta} \quad \lambda R(\theta) + L(H_\theta(x, \cdot), y) \tag{9.1}$$

Intuitively, $H_\theta$ is a function that quantifies by $H_\theta(x, y)$ how incompatible attributes $x \in X$ are with a combination of decisions $y \in \{0,1\}^S$. Consequently, $H_\theta(x, \cdot)$ is a functional that assigns an incompatibility to every combination of decisions.

**Definition 31** For any unlabeled structured data $T = (S, F, E, \{X_f\}_{f \in F}, x)$ and any $\hat{H} \colon X \times \{0,1\}^S \to \mathbb{R}$, the instance of the *inference problem* w.r.t. $T$ and $\hat{H}$ is defined as

$$\min_{y \in \{0,1\}^S} \quad \hat{H}(x, y) \tag{9.2}$$

## 9.3 Conditional graphical models

### 9.3.1 Data

Throughout Section 9.3, we consider labeled data $(S, F, E, \{X_f\}_{f \in F}, x, y)$ and an attribute space $X = \prod_{f \in F} X_f$ such that, for every $f \in F$, there is an $n_f \in \mathbb{N}$ and $X_f = \mathbb{R}^{n_f}$.

### 9.3.2 Family of functions

**Definition 32** For any factor graph $G = (S, F, E)$, a function $H : \{0,1\}^S \to \mathbb{R}$ is said to *factorize* w.r.t. $G$ iff, for every $f \in F$, there exists a function a function $h_f : \{0,1\}^{S_f} \to \mathbb{R}$, called a *factor* of $H$, such that

$$\forall y \in \{0,1\}^S \colon \quad H(y) = \sum_{f \in F} h_f(y_{S_f}) \ . \tag{9.3}$$

An example is shown in Fig. 9.1.

**Definition 33** A tuple $(S, F, E, \{X_f\}_{f \in F}, \Theta, \{h_f\}_{f \in F})$ is called a *conditional graphical model* with *attribute space* $\prod_{f \in F} X_f = X$ and *parameter space* $\Theta$ iff $(S, F, E)$ is a factor graph, $\Theta \neq \emptyset$ and, for every $f \in F$, $X_f \neq \emptyset$, called the *attribute space* of $f$, and $h_f : \Theta \to \mathbb{R}^{X_f \times \{0,1\}^{S_f}}$, called a *factor*.

The $H : \Theta \to \mathbb{R}^{X \times \{0,1\}^S}$ defined below is called the *energy function* of the conditional graphical model.

$$\forall \theta \in \Theta \ \forall x \in X \ \forall y \in \{0,1\}^S \colon \quad H_\theta(x, y) = \sum_{f \in F} h_{f\theta}(x_f, y_{S_f}) \tag{9.4}$$

Throughout Section 9.3, we consider such a conditional graphical model. We make two additional assumptions: Firstly, we assume that $\Theta$ is a finite-dimensional, real vector space, i.e., there exists a finite, non-empty set $J$ and $\Theta = \mathbb{R}^J$. Secondly, we assume that every function $h_f$ is linear in

$\Theta$, i.e., for every $f \in F$, there exists a $\varphi_f : X_f \times \{0,1\}^{S_f} \to \mathbb{R}^J$ such that for any $x_f \in X_f$, any $y_{S_f} \in \{0,1\}^{S_f}$ and any $\theta \in \Theta$:

$$h_{f\theta}(x_f, y_{S_f}) = \langle \theta, \varphi_f(x_f, y_{S_f}) \rangle \tag{9.5}$$

For convenience, we define $\xi : X \times \{0,1\}^S \to \mathbb{R}^J$ such that for any $x \in X$ and any $y \in \{0,1\}^S$:

$$\xi(x,y) = \sum_{f \in F} \varphi_f(x_f, y_{S_f}) \tag{9.6}$$

Thus, we obtain for any $\theta \in \Theta$, any $x \in X$ and any $y \in Y$:

$$\begin{aligned} H_\theta(x,y) &= \sum_{f \in F} h_{f\theta}(x_f, y_{S_f}) \\ &= \sum_{f \in F} \langle \theta, \varphi_f(x_f, y_{S_f}) \rangle \\ &= \left\langle \theta, \sum_{f \in F} \varphi_f(x_f, y_{S_f}) \right\rangle \\ &= \langle \theta, \xi(x,y) \rangle \end{aligned} \tag{9.7}$$

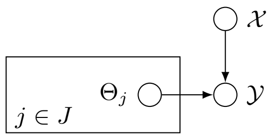
### 9.3.3 Probabilistic model

#### Random Variables

- Let $\mathcal{X}$ be a random variable whose realization is an element $x \in X$ of the attribute space.
- Let $\mathcal{Y}$ be a random variable whose realization is a combination of decisions $y \in \{0,1\}^S$
- For any $j \in J$, let $\Theta_j$ a random variable whose realization is a $\theta_j \in \mathbb{R}$

#### Conditional independence assumptions

We assume a probability distribution that factorizes according to the Bayesian net depicted below.



#### Factorization

- Firstly:

$$P(\mathcal{X}, \mathcal{Y}, \Theta) = P(\mathcal{Y} \mid \mathcal{X}, \Theta)\, P(\mathcal{X}) \prod_{j \in J} P(\Theta_j) \tag{9.8}$$

- Secondly:

$$\begin{aligned} P(\Theta \mid \mathcal{X}, \mathcal{Y}) &= \frac{P(\mathcal{X}, \mathcal{Y}, \Theta)}{P(\mathcal{X}, \mathcal{Y})} \\ &= \frac{P(\mathcal{Y} \mid \mathcal{X}, \Theta)\, P(\mathcal{X}) \prod_{j \in J} P(\Theta_j)}{P(\mathcal{X}, \mathcal{Y})} \\ &\propto P(\mathcal{Y} \mid \mathcal{X}, \Theta) \prod_{j \in J} P(\Theta_j) \end{aligned} \tag{9.9}$$

**Forms**

**Definition 34** For any conditional graphical model, the *partition function* $Z\colon X \times \Theta \to \mathbb{R}$ and *Gibbs distribution* $p\colon X \times \{0,1\}^S \times \Theta \to [0,1]$ are defined by the forms

$$Z(x,\theta) = \sum_{y \in \{0,1\}^S} e^{-H_\theta(x,y)} \tag{9.10}$$

$$p(y,x,\theta) = \frac{1}{Z(x,\theta)} e^{-H_\theta(x,y)} \tag{9.11}$$

We consider in (9.9) the Gibbs distribution of our conditional graphical model, i.e.

$$p_{\mathcal{Y}|\mathcal{X},\Theta}(y,x,\theta) = \frac{1}{Z(x,\theta)} e^{-H_\theta(x,y)} \quad . \tag{9.12}$$

Moreover, we consider in (9.9) a $\sigma \in \mathbb{R}^+$ and, for every $j \in J$, the *normal distribution*

$$p_{\Theta_j}(\theta_j) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\theta_j^2/2\sigma^2} \quad . \tag{9.13}$$

### 9.3.4 Learning problem

**Lemma 14** *Estimating maximally probable parameters $\theta$, given attributes $x$ and decisions $y$, i.e.,*

$$\operatorname*{argmax}_{\theta \in \mathbb{R}^J} \quad p_{\Theta|\mathcal{X},\mathcal{Y}}(\theta,x,y)$$

*is identical to the supervised structured learning problem w.r.t. $L$, $R$ and $\lambda$ such that*

$$L(H_\theta(x,\cdot),y) = H_\theta(x,y) + \ln Z(x,\theta) \tag{9.14}$$

$$= H_\theta(x,y) + \ln \sum_{y' \in \{0,1\}^S} e^{-H_\theta(x,y')} \tag{9.15}$$

$$= \langle \theta, \xi(x,y) \rangle + \ln \sum_{y' \in \{0,1\}^S} e^{-\langle \theta, \xi(x,y') \rangle} \tag{9.16}$$

$$R(\theta) = \|\theta\|_2^2 \tag{9.17}$$

$$\lambda = \frac{1}{2\sigma^2} \tag{9.18}$$

**Exercise 12** *Prove Lemma 14.*

**Lemma 15** *The first and second partial derivatives of the logarithm of the partition function have the forms*

$$\frac{\partial}{\partial \theta_j} \ln Z = \frac{1}{Z(x,\theta)} \sum_{y' \in \{0,1\}^S} (-\xi_j(x,y')) e^{-\langle \theta, \xi(x,y') \rangle} \tag{9.19}$$

$$= \mathbb{E}_{y' \sim p_{\mathcal{Y}|\mathcal{X},\Theta}}(-\xi_j(x,y')) \tag{9.20}$$

$$\frac{\partial^2}{\partial \theta_j \, \partial \theta_k} \ln Z = \mathbb{E}_{y' \sim p_{\mathcal{Y}|\mathcal{X},\Theta}}(\xi_j(x,y')\xi_k(x,y')) - \mathbb{E}_{y' \sim p_{\mathcal{Y}|\mathcal{X},\Theta}}(\xi_j(x,y'))\mathbb{E}_{y' \sim p_{\mathcal{Y}|\mathcal{X},\Theta}}(\xi_k(x,y'))$$

$$= \mathrm{COV}_{y' \sim p_{\mathcal{Y}|\mathcal{X},\Theta}}(\xi_j(x,y'), \xi_k(x,y')) \tag{9.21}$$

**Exercise 13** *Prove Lemma 15.*

**Lemma 16** *Supervised structured learning of a conditional graphical model is a convex optimization problem.*

**Exercise 14** *Prove Lemma 16 using* (9.21).

### 9.3.5 Inference problem

**Lemma 17** *Estimating maximally probable decisions $y$, given attributes $x$ and parameters $\theta$, i.e.*

$$\operatorname*{argmax}_{y \in \{0,1\}^S} \quad p_{\mathcal{Y}|\mathcal{X},\Theta}(x, y, \theta) \tag{9.22}$$

*is identical to the structured inference problem with $\hat{H}(x, y) = H_\theta(x, y)$.*

**Exercise 15** *Prove Lemma 17.*

### 9.3.6 Learning algorithm

On the on hand, the supervised structured learning problem for conditional graphical models can be solved exactly by means of the steepest descent algorithm, due to its convexity (Lemma 16).

**Algorithm 9** Steepest descent with tolerance parameter $\epsilon \in \mathbb{R}_0^+$

```
θ := 0
repeat
    $d := \nabla_\theta L(H_\theta(x, \cdot), y)$
    $\eta := \operatorname{argmin}_{\eta' \in \mathbb{R}} L(H_{\theta - \eta' d}(x, \cdot), y)$    (line search)
    $\theta := \theta - \eta d$
    if $\|d\| < \epsilon$
        return $\theta$
```

On the other hand, the time complexity of computing the gradient is $O(2^{|S|})$, due to the summations involved in computing the partition function $Z(x, \theta)$ and expectation values (9.19). More specifically, computing a derivative

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} \ln Z &= -\mathbb{E}_{y' \sim p_{\mathcal{Y}|\mathcal{X},\Theta}}(\xi_j(x, y')) \\
&= -\frac{1}{Z(x,\theta)} \sum_{y' \in \{0,1\}^S} \xi_j(x, y')\, e^{-\langle \theta, \xi(x,y') \rangle} \\
&= \frac{1}{Z(x,\theta)} \sum_{y' \in \{0,1\}^S} \sum_{f \in F} \varphi_{fj}(x_f, y'_{S_f})\, e^{-\langle \theta, \xi(x,y') \rangle} \\
&= \frac{1}{Z(x,\theta)} \sum_{f \in F} \sum_{y'_{S(f)} \in \{0,1\}^{S(f)}} \sum_{y'_{S \setminus S(f)} \in \{0,1\}^{S \setminus S(f)}} \varphi_{fj}(x_f, y'_{S(f)})\, e^{-\langle \theta, \xi(x,y') \rangle} \\
&= \sum_{f \in F} \sum_{y'_{S(f)} \in \{0,1\}^{S(f)}} \varphi_{fj}(x_f, y'_{S(f)}) \frac{1}{Z(x,\theta)} \sum_{y'_{S \setminus S(f)} \in \{0,1\}^{S \setminus S(f)}} e^{-\langle \theta, \xi(x,y') \rangle} && (9.23) \\
&= \sum_{f \in F} \sum_{y'_{S(f)} \in \{0,1\}^{S(f)}} \varphi_{fj}(x_f, y'_{S(f)})\, p_{\mathcal{Y}_{S(f)}|\mathcal{X},\Theta}(y'_{S(f)} \mid x, \theta) && (9.24) \\
&= \sum_{f \in F} \mathbb{E}_{y'_{S(f)} \sim p_{\mathcal{Y}_{S(f)}|\mathcal{X},\Theta}}(\varphi_{fj}(x_f, y'_{S(f)})) && (9.25)
\end{aligned}$$

requires computing

- the partition function

$$Z(x, \theta) = \sum_{y' \in \{0,1\}^S} e^{-\langle \theta, \xi(x,y') \rangle} \tag{9.26}$$

- for every factor $f \in F$, the so-called *factor marginal*

$$p_{\mathcal{Y}_{S(f)}|\mathcal{X},\Theta}(y'_{S(f)} \mid x, \theta) = \frac{1}{Z(x,\theta)} \sum_{y'_{S\setminus S(f)} \in \{0,1\}^{S\setminus S(f)}} e^{-\langle \theta, \xi(x,y') \rangle} \tag{9.27}$$

- for every factor $f \in F$, the expectation value

$$\sum_{y'_{S(f)} \in \{0,1\}^{S(f)}} \varphi_{fj}(x_f, y'_{S(f)})\, p_{\mathcal{Y}_{S(f)}|\mathcal{X},\Theta}(y'_{S(f)} \mid x, \theta) \enspace . \tag{9.28}$$

In the special case where the degree $\max_{f\in F} S(f)$ of the conditional graphical model is bounded by a constant, computing (9.28) from the factor marginal takes constant time. The challenge in (9.27) or all (9.26) is to sum the function

$$\psi_\theta(x, y') := e^{-\langle \theta, \xi(x,y') \rangle} \tag{9.29}$$

over assignments to some (9.27) or all (9.26) variables $y'$. Defining

$$\psi_{f\theta}(x_f, y'_{S(f)}) = e^{-\langle \theta, \varphi_f(x_f, y'_{S(f)}) \rangle} \tag{9.30}$$

and exploiting factorization (9.6), we obtain

$$\begin{aligned} & e^{-\langle \theta, \xi(x,y') \rangle} \\ &= e^{-\sum_{f\in F} \langle \theta, \varphi_f(x_f, y_{S(f)}) \rangle} && (9.31) \\ &= \prod_{f \in F} e^{-\langle \theta, \varphi_f(x_f, y_{S(f)}) \rangle} && (9.32) \\ &= \prod_{f\in F} \psi_{f\theta}(x_f, y_{S(f)}) \enspace . && (9.33) \end{aligned}$$

Thus, the challenge in (9.27) and (9.26) is to compute a sum of a product of functions. Specifically:

$$Z(x,\theta) = \sum_{y' \in \{0,1\}^S} \prod_{f\in F} \psi_{f\theta}(x_f, y_{S(f)}) \tag{9.34}$$

$$p_{\mathcal{Y}_{S(f)}|\mathcal{X},\Theta}(y'_{S(f)} \mid x, \theta) = \frac{1}{Z(x,\theta)} \sum_{y'_{S\setminus S(f)} \in \{0,1\}^{S\setminus S(f)}} \prod_{f\in F} \psi_{f\theta}(x_f, y_{S(f)}) \tag{9.35}$$

One approach to tackle this problem is to sum over variables recursively. In order to avoid redundant computation, Kschischang et al. (2001) define partial sums:

**Definition 35 (Kschischang et al. (2001))** For any variable node $s \in S$ and any factor node $f \in F$, the functions

$$m_{s\to f}, m_{f\to s} \colon \{0,1\} \to \mathbb{R} \enspace , \tag{9.36}$$

called *messages*, are defined such that for all $y_s \in \{0,1\}$:

$$m_{s\to f}(y_s) = \prod_{f' \in F(s)\setminus\{f\}} m_{f'\to s}(y_s) \tag{9.37}$$

$$m_{f\to s}(y_s) = \sum_{y_{S(f)\setminus\{s\}}} \psi_{f\theta}(x_f, y_{S(f)}) \prod_{s' \in S(f)\setminus\{s\}} m_{s'\to f}(y_{s'}) \tag{9.38}$$

**Lemma 18** *If the factor graph is acyclic, messages are defined recursively by* (9.37) *and* (9.38)*, beginning with the messages from leaves. Moreover, for any $s \in S$ and any $f \in F$:*

$$Z(x,\theta) = \sum_{y_s \in \{0,1\}} \prod_{f' \in F(s)} m_{f' \to s}(y_s) \tag{9.39}$$

$$p_{\mathcal{Y}_{S(f)} \mid \mathcal{X},\Theta}(y'_{S(f)} \mid x,\theta) = \frac{1}{Z(x,\theta)} \psi_{f\theta}(x_f, y_{S(f)}) \prod_{s' \in S(f)} m_{s' \to f}(y_{s'}) \tag{9.40}$$

**Exercise 16** *Prove Lemma 18.*

The recursive computation of messages is known as *message passing*.

For conditional graphical models whose factor graph is acylic, the supervised structured learning problem can be solved efficiently by means of the steepest descent algorithm and message passing, by Lemma 16 and Lemma 18.

For conditional graphical models whose factor graph is cyclic, the definition of messages by (9.37) and (9.38) is cyclic as well. The partition function and marginals cannot be computed by message passing in general. A heuristic without guarantee of correctness or even convergence is to initialize all messages as normalized constant functions and update messages according to some schedule, e.g., synchronously. This heuristic is known as *loopy belief propagation* and has proven suitable for some applications.

## 9.3.7 Inference algoritms

### Iterated conditional modes (ICM)

For the inference problem

$$\operatorname*{argmin}_{y \in \{0,1\}^S} \; H_\theta(x,y) \;, \tag{9.41}$$

a heuristic that is guaranteed to converge and terminate in a (possibly sub-optimal) feasible solution is local search w.r.t. transformations that change one variable at a time:

**Definition 36** For any $s \in S$, let $\mathrm{flip}_s \colon \{0,1\}^S \to \{0,1\}^S$ such that for any $y \in \{0,1\}^S$ and any $t \in S$:

$$\mathrm{flip}_s[y](t) = \begin{cases} 1 - y_t & \text{if } t = s \\ y_t & \text{otherwise} \end{cases} . \tag{9.42}$$

**Algorithm 10** Greedy local search w.r.t. transformations that change one variable at a time is defined by the recursion below. In the context of graphical models and probabilistic inference, this algorithm is also known as *iterated conditional modes*, or ICM (Besag, 1986).

```
$y' = \mathrm{icm}(y)$
choose $s \in \operatorname*{argmin}_{s' \in S} \; H_\theta(x, \mathrm{flip}_{s'}[y]) - H_\theta(x,y)$
if $H_\theta(x, \mathrm{flip}_s[y]) < H_\theta(x,y)$
    $y' := \mathrm{icm}(\mathrm{flip}_s[y])$
else
    $y' := y$
```

### Message passing

The inference problem

$$\operatorname*{argmin}_{y \in \{0,1\}^S} \; \sum_{f \in F} h_{f\theta}(x_f, y_{S(f)}) \tag{9.43}$$

consists in computing the minimum of a sum of of functions. This problem is analogous to that of computing the sum of a product of functions (Section 9.3.6) in that both $(\mathbb{R}, \min, +)$ and $(\mathbb{R}, +, \cdot)$ are commutative semi-rings. This analogy is sufficient to transfer the idea of message passing, albeit with messages adapted to the $(\mathbb{R}, \min, +)$ semi-ring:

**Definition 37 (Kschischang et al. (2001))** For any variable node $s \in S$ and any factor node $f \in F$, the functions

$$\mu_{s \to f}, \mu_{f \to s} \colon \{0,1\} \to \mathbb{R} \ , \tag{9.44}$$

called *messages*, are defined such that for all $y_s \in \{0,1\}$:

$$\mu_{s \to f}(y_s) = \sum_{f' \in F(s) \setminus \{f\}} \mu_{f' \to s}(y_s) \tag{9.45}$$

$$\mu_{f \to s}(y_s) = \min_{y_{S(f) \setminus \{s\}}} \psi_{f\theta}(x_f, y_{S(f)}) + \sum_{s' \in S(f) \setminus \{s\}} \mu_{s' \to f}(y_{s'}) \tag{9.46}$$

**Lemma 19** *If the factor graph is acyclic, messages are defined recursively by* (9.45) *and* (9.46)*, beginning with the messages from leaves. Moreover, for any $s \in S$:*

$$\min_{y \in \{0,1\}^S} \sum_{f \in F} h_{f\theta}(x_f, y_{S(f)}) = \min_{y_s \in \{0,1\}} \sum_{f' \in F(s)} \mu_{f' \to s}(y_s)$$

Proof Analogous to Lemma 18.

For conditional graphical models whose factor graph is acyclic, the inference problem can be solved efficiently by means of message passing, by Lemma 19.

For conditional graphical models whose factor graph is cyclic, the definition of messages by (9.45) and (9.46) is cyclic as well. The inference problem cannot be solved by message passing in general. A heuristic without guarantee of correctness or even convergence is to initialize all messages as constant zero and update messages according to some schedule, e.g., synchronously. This heuristic is also known as *loopy belief propagation* and has proven suitable for some applications.

# Appendix A

# Combinatorial problems

## A.1 Satisfiability

**Definition 38** For any CNF defined by $V$ and $\theta$ as in Definition 4, and for the Boolean function $f_\theta$ defined by this form as in Definition 4, deciding whether there exists an $x \in \{0,1\}^V$ such that $f_\theta(x) = 1$ is called the instance of the *satisfiability problem (*SAT*)* with respect to $V$ and $\theta$.

Any instance of SAT with respect to a 3-CNF defined by $V$ and $\theta$ as in Definition 4 is additionally called an instance of the *3-satisfiability problem (*3-SAT*)* with respect to $V$ and $\theta$.

**Theorem 3 (Cook (1971))** SAT *is* NP*-complete.*

**Theorem 4 (Karp (1972))** SAT $\leq_p$ 3-SAT.

**Lemma 20 (Karp (1972))** 3-SAT *is* NP*-complete.*

PROOF 3-SAT $\in$ NP, as solutions can be verified efficiently.

3-SAT is NP-hard by Theorems 3 and 4.

## A.2 Matching

**Definition 39** For any $m \in \mathbb{N}$, any finite, non-empty sets $S_0, \ldots, S_{m-1}$ and any $T \subseteq S_0 \times \cdots \times S_{m-1}$ the set $T$ is called a *matching* of $S_0, \ldots, S_{m-1}$ iff, for all distinct $(s_0, \ldots, s_{m-1}), (s'_0, \ldots, s'_{m-1}) \in T$ and all $j \in [m]$, we have $s_j \neq s'_j$.

The matching is called *perfect* iff, in addition, for every $j \in [m]$ and every $s_j \in S_j$, there exists precisely one $s' \in T$ such that $s'_j = s_j$.

**Lemma 21** *If a perfect matching as in Definition 39 exists, we have* $|S_0| = \cdots = |S_{m-1}|$.

**Definition 40** For any $m \in \mathbb{N}$, any finite, non-empty sets $S_0, \ldots, S_{m-1}$ such that $|S_0| = \cdots = |S_{m-1}|$ and any $T \subseteq S_0 \times \cdots \times S_{m-1}$, deciding whether there exists a perfect matching $T' \subseteq T$ of $S_0, \ldots, S_{m-1}$ is called the instance of the *$m$-dimensional perfect matching problem ($m$-*PM*)* with respect to $m$, $S_0, \ldots, S_{m-1}$ and $T$.

**Theorem 5** 3-SAT $\leq_p$ 3-PM

PROOF Consider any instance of 3-SAT defined by a 3-CNF defined by $V$ and $\theta$ as in Definition 4.

Let $n = |\theta|$ the number of clauses of the 3-CNF.

Choose any order on $\theta$ to obtain a bijection $\theta' : [n] \to \theta$.

Define an instance of 3-PM by three sets $A, B, C$ of equal cardinality and one set $T \subseteq A \times B \times C$ of triples constructed as follows:

- For any clause index $c \in [n]$ and any variable $v \in V$, let

$$(a_c^v,\ b_c^v,\ \bar{x}_c^v) \in T \tag{A.1}$$
$$(a_{c+1 \bmod n}^v,\ b_c^v,\ x_c^v) \in T \tag{A.2}$$

- For any clause index $c \in [n]$ and the $(V_0, V_1) = \theta'_c$, distinguish between the variables in the clause, that is, the set $V_0 \cup V_1$, and the variables not in the clause, that is, the set $V \setminus (V_0 \cup V_1)$:
  - For each variable $v \in V \setminus (V_0 \cup V_1)$, let

$$(\alpha_c^v,\ \beta_c^v,\ \bar{x}_c^v) \in T \tag{A.3}$$
$$(\alpha_c^v,\ \beta_c^v,\ x_c^v) \in T \tag{A.4}$$

  - For each variable $v \in V_0$, let

$$(a_c,\ b_c,\ \bar{x}_c^v) \in T \tag{A.5}$$

  - For each variable $v \in V_1$, let

$$(a_c,\ b_c,\ x_c^v) \in T \tag{A.6}$$

  - If $|V_0 \cup V_1| \geq 2$, then, for each variable $v \in V_0 \cup V_1$, let

$$(\alpha_c,\ \beta_c,\ x_c^v) \in T \tag{A.7}$$
$$(\alpha_c,\ \beta_c,\ \bar{x}_c^v) \in T \tag{A.8}$$

  - If $|V_0 \cup V_1| = 3$, then, for each variable $v \in V_0 \cup V_1$, let

$$(\alpha'_c,\ \beta'_c,\ x_c^v) \in T \tag{A.9}$$
$$(\alpha'_c,\ \beta'_c,\ \bar{x}_c^v) \in T \tag{A.10}$$

In order to verify that $|A| = |B| = |C|$, we count the elements defined in the triples above:

| Triples | Elements | $\|A\|$ | $\|B\|$ | $\|C\|$ |
|---|---|---|---|---|
| (A.1), (A.2) | $a_c^v$ | $n\|V\|$ | | |
| (A.1), (A.2) | $b_c^v$ | | $n\|V\|$ | |
| (A.1), (A.2) | $\bar{x}_c^v$ | | | $n\|V\|$ |
| (A.1), (A.2) | $x_c^v$ | | | $n\|V\|$ |
| (A.3), (A.4) | $\alpha_c^v$ | $\sum_{(V_0,V_1)\in\theta} (\|V\| - \|V_0\| - \|V_1\|)$ | | |
| (A.3), (A.4) | $\beta_c^v$ | | $\sum_{(V_0,V_1)\in\theta} (\|V\| - \|V_0\| - \|V_1\|)$ | |
| (A.5)–(A.10) | $a_c, \alpha_c, \alpha'_c$ | $\sum_{(V_0,V_1)\in\theta} (\|V_0\| + \|V_1\|)$ | | |
| (A.5)–(A.10) | $b_c, \beta_c, \beta'_c$ | | $\sum_{(V_0,V_1)\in\theta} (\|V_0\| + \|V_1\|)$ | |
| | Total | $2n\|V\|$ | $2n\|V\|$ | $2n\|V\|$ |

Next, we show that the instance of 3-SAT defined by $V$ and $\theta$ has a solution iff the instance of 3-PM defined by $A, B, C$ and $T$ has a solution.

($\Rightarrow$) Let $\hat{x} \in \{0,1\}^V$ a solution to the instance of 3-SAT defined by $V$ and $\theta$.

Hence, for any clause $(V_0, V_1) \in \theta$, we can choose one variable $v_c \in V_0 \cup V_1$ such that

$$(v_c \in V_0 \ \wedge\ \hat{x}(v_c) = 0) \ \vee\ (v_c \in V_1 \ \wedge\ \hat{x}(v_c) = 1) \ . \tag{A.11}$$

Consider any $T' \subseteq T$ such that:

- For any clause index $c \in [n]$ and any variable $v \in V$:

$$(a_c^v,\ b_c^v,\ \bar{x}_c^v) \in T' \ \Leftrightarrow\ \hat{x}(v) = 1 \tag{A.12}$$
$$(a_{c+1 \bmod n}^v,\ b_c^v,\ x_c^v) \in T' \ \Leftrightarrow\ \hat{x}(v) = 0 \tag{A.13}$$

- For any clause index $c \in [n]$, the $(V_0, V_1) = \theta'_c$ and any variable $v \in V \setminus (V_0 \cup V_1)$:

$$(\alpha_c^v,\ \beta_c^v,\ \bar{x}_c^v) \in T' \ \Leftrightarrow\ \hat{x}(v) = 0 \tag{A.14}$$
$$(\alpha_c^v,\ \beta_c^v,\ x_c^v) \in T' \ \Leftrightarrow\ \hat{x}(v) = 1 \tag{A.15}$$

- For any clause index $c \in [n]$:
  - For the $v_c \in V_0 \cup V_1$ chosen in (A.11):

$$(a_c,\ b_c,\ \bar{x}_c^{v_c}) \in T' \ \Leftrightarrow\ \hat{x}(v_c) = 0 \tag{A.16}$$
$$(a_c,\ b_c,\ x_c^{v_c}) \in T' \ \Leftrightarrow\ \hat{x}(v_c) = 1 \tag{A.17}$$

  - If $|V_0 \cup V_1| \geq 2$, for precisely one variable $v'_c \in (V_0 \cup V_1) \setminus \{v_c\}$:

$$(\alpha_c,\ \beta_c,\ \bar{x}_c^{v'_c}) \in T' \ \Leftrightarrow\ \hat{x}(v'_c) = 0 \tag{A.18}$$
$$(\alpha_c,\ \beta_c,\ x_c^{v'_c}) \in T' \ \Leftrightarrow\ \hat{x}(v'_c) = 1 \tag{A.19}$$

  - If $|V_0 \cup V_1| = 3$, for the remaining variable $v''_c \in (V_0 \cup V_1) \setminus \{v_c, v'_c\}$:

$$(\alpha'_c,\ \beta'_c,\ \bar{x}_c^{v''_c}) \in T' \ \Leftrightarrow\ \hat{x}(v''_c) = 0 \tag{A.20}$$
$$(\alpha'_c,\ \beta'_c,\ x_c^{v''_c}) \in T' \ \Leftrightarrow\ \hat{x}(v''_c) = 1 \tag{A.21}$$

We show that $T'$ is a solution to the instance of 3-PM defined by $A, B, C$ and $T$, by verifying that each element of $A, B$ and $C$ occurs in precisely one triple in $T'$:

- For any $c \in [n]$, the $(V_0, V_1) = \theta'_c$ and any $v \in V$:

| Element | Triple | |
|---|---|---|
| $a_c^v$ | $(a_c^v,\ b_c^v,\ \bar{x}_c^v)$ | if $\hat{x}(v) = 1$ |
| | $(a_c^v,\ b_{c-1 \bmod n}^v,\ x_{c-1 \bmod n}^v)$ | if $\hat{x}(v) = 0$ |
| $b_c^v$ | $(a_c^v,\ b_c^v,\ \bar{x}_c^v)$ | if $\hat{x}(v) = 1$ |
| | $(a_{c+1 \bmod n}^v,\ b_c^v,\ x_c^v)$ | if $\hat{x}(v) = 0$ |
| $\bar{x}_v^c$ | $(a_c^v,\ b_c^v,\ \bar{x}_c^v)$ | if $\hat{x}(v) = 1$ |
| | $(\alpha_c^v,\ \beta_c^v,\ \bar{x}_c^v)$ | if $\hat{x}(v) = 0$ and $v \in V \setminus (V_0 \cup V_1)$ |
| | $(a_c,\ b_c,\ \bar{x}_c^{v_c})$ | if $\hat{x}(v) = 0$ and $v = v_c$ |
| | $(a_c,\ b_c,\ \bar{x}_c^{v'_c})$ | if $\hat{x}(v) = 0$ and $\lvert V_0 \cup V_1 \rvert \geq 2$ and $v = v'_c$ |
| | $(a_c,\ b_c,\ \bar{x}_c^{v''_c})$ | if $\hat{x}(v) = 0$ and $\lvert V_0 \cup V_1 \rvert = 3$ and $v = v''_c$ |
| $x_v^c$ | $(a_{c+1 \bmod n}^v,\ b_c^v,\ x_c^v)$ | if $\hat{x}(v) = 0$ |
| | $(\alpha_c^v,\ \beta_c^v,\ x_c^v)$ | if $\hat{x}(v) = 1$ and $v \in V \setminus (V_0 \cup V_1)$ |
| | $(a_c,\ b_c,\ x_c^{v_c})$ | if $\hat{x}(v) = 1$ and $v = v_c$ |
| | $(a_c,\ b_c,\ x_c^{v'_c})$ | if $\hat{x}(v) = 1$ and $\lvert V_0 \cup V_1 \rvert \geq 2$ and $v = v'_c$ |
| | $(a_c,\ b_c,\ x_c^{v''_c})$ | if $\hat{x}(v) = 1$ and $\lvert V_0 \cup V_1 \rvert = 3$ and $v = v''_c$ |

- For any $c \in [n]$, the $(V_0, V_1) = \theta'_c$ and any $v \in V \setminus (V_0 \cup V_1)$:

| Element | Triple | |
|---|---|---|
| $\alpha_c^v$ | $(\alpha_c^v,\ \beta_c^v,\ \bar{x}_c^v)$ | if $\hat{x}(v) = 0$ |
| | $(\alpha_c^v,\ \beta_c^v,\ x_c^v)$ | if $\hat{x}(v) = 1$ |
| $\beta_c^v$ | $(\alpha_c^v,\ \beta_c^v,\ \bar{x}_c^v)$ | if $\hat{x}(v) = 0$ |
| | $(\alpha_c^v,\ \beta_c^v,\ x_c^v)$ | if $\hat{x}(v) = 1$ |

- For any $c \in [n]$:

| Element | Triple | |
|---|---|---|
| $a_c$ | $(a_c,\ b_c,\ \bar{x}_c^{v_c})$ | if $\hat{x}(v_c) = 0$ |
| | $(a_c,\ b_c,\ x_c^{v_c})$ | if $\hat{x}(v_c) = 1$ |
| $b_c$ | $(a_c,\ b_c,\ \bar{x}_c^{v_c})$ | if $\hat{x}(v_c) = 0$ |
| | $(a_c,\ b_c,\ x_c^{v_c})$ | if $\hat{x}(v_c) = 1$ |

- For any $c \in [n]$ with $(V_0, V_1) = \theta'_c$ such that $|V_0 \cup V_1| \geq 2$:

| Element | Triple | |
|---|---|---|
| $\alpha_c$ | $(\alpha_c,\ \beta_c,\ \bar{x}_c^{v'_c})$ | if $\hat{x}(v'_c) = 0$ |
| | $(\alpha_c,\ \beta_c,\ x_c^{v'_c})$ | if $\hat{x}(v'_c) = 1$ |
| $\beta_c$ | $(\alpha_c,\ \beta_c,\ \bar{x}_c^{v'_c})$ | if $\hat{x}(v'_c) = 0$ |
| | $(\alpha_c,\ \beta_c,\ x_c^{v'_c})$ | if $\hat{x}(v'_c) = 1$ |

- For any $c \in [n]$ with $(V_0, V_1) = \theta'_c$ such that $|V_0 \cup V_1| = 3$:

| Element | Triple | |
|---|---|---|
| $\alpha'_c$ | $(\alpha'_c,\ \beta'_c,\ \bar{x}_c^{v''_c})$ | if $\hat{x}(v''_c) = 0$ |
| | $(\alpha'_c,\ \beta'_c,\ x_c^{v''_c})$ | if $\hat{x}(v''_c) = 1$ |
| $\beta'_c$ | $(\alpha'_c,\ \beta'_c,\ \bar{x}_c^{v''_c})$ | if $\hat{x}(v''_c) = 0$ |
| | $(\alpha'_c,\ \beta'_c,\ x_c^{v''_c})$ | if $\hat{x}(v''_c) = 1$ |

($\Leftarrow$) Let $T' \subseteq T$ be any solution to the instance of 3-PM defined by $A, B, C$ and $T$. Moreover, let $\hat{x} \in \{0,1\}^V$ such that

$$\forall v \in V: \quad \hat{x}_v = \begin{cases} 1 & \text{if } (a_0^v,\ b_0^v,\ \bar{x}_0^v) \in T' \\ 0 & \text{otherwise} \end{cases} . \tag{A.22}$$

We show that $\hat{x}$ is a solution to the instance of 3-SAT defined by $V$ and $\theta$:

To begin with, we observe the clause index 0 in (A.22) being an arbitrary choice, because, for any clause index $c \in [n]$, we have by construction of $T$:

$$\begin{aligned} & (a_c^v,\ b_c^v,\ \bar{x}_c^v) \in T' \\ \Rightarrow\ & (a_{c+1 \bmod n}^v,\ b_c^v,\ x_c^v) \notin T' && \text{(A.23)} \\ \Rightarrow\ & (a_{c+1 \bmod n}^v,\ b_{c+1 \bmod n}^v,\ \bar{x}_{c+1 \bmod n}^v) \in T' && \text{(A.24)} \end{aligned}$$

By induction, we conclude for any clause index $c \in [n]$:

$$(a_c^v,\ b_c^v,\ \bar{x}_c^v) \in T' \ \Leftrightarrow\ (a_0^v,\ b_0^v,\ \bar{x}_0^v) \in T' \tag{A.25}$$

For any clause index $c \in [n]$ and the $(V_0, V_1) = \theta'_c$, there exists a triple in $T'$ that contains $a_c$ (because $T'$ is a solution to the instance of 3-PM).

Thus, and by construction of $T$, there exists a $v \in V$ such that one of the following statements holds:

$$v \in V_0 \;\wedge\; (a_c,\ b_c,\ \bar{x}^v_c) \in T' \tag{A.26}$$

$$v \in V_1 \;\wedge\; (a_c,\ b_c,\ x^v_c) \in T' \tag{A.27}$$

We consider the two cases separately:

- If $v \in V_0$, we have

$$\begin{aligned} & (a_c,\ b_c,\ \bar{x}^v_c) \in T' \\ \Rightarrow\quad & (a^v_c,\ b^v_c,\ \bar{x}^v_c) \notin T' && \text{(A.28)} \\ \Rightarrow\quad & (a^v_0,\ b^v_0,\ \bar{x}^v_0) \notin T' && \text{by (A.25)} \quad \text{(A.29)} \\ \Rightarrow\quad & \hat{x}(v) = 0\ . && \text{(A.30)} \end{aligned}$$

  The clause indexed by $c$ and defined by $(V_0, V_1)$ is satisfied because $v \in V_0$ and $\hat{x}(v) = 0$.

- If $v \in V_0$, we have

$$\begin{aligned} & (a_c,\ b_c,\ x^v_c) \in T' \\ \Rightarrow\quad & (a^v_{c+1 \bmod n},\ b^v_c,\ x^v_c) \notin T' && \text{(A.31)} \\ \Rightarrow\quad & (a^v_c,\ b^v_c,\ \bar{x}^v_c) \in T' && \text{(A.32)} \\ \Rightarrow\quad & (a^v_0,\ b^v_0,\ \bar{x}^v_0) \in T' && \text{by (A.25)} \quad \text{(A.33)} \\ \Rightarrow\quad & \hat{x}(v) = 1\ . && \text{(A.34)} \end{aligned}$$

  The clause indexed by $c$ and defined by $(V_0, V_1)$ is satisfied because $v \in V_1$ and $\hat{x}(v) = 1$.

**Exercise 17** *Follow the proof of Theorem 5 in order to:*

*(a) construct the instance of* 3-PM *for the instance of* 3-SAT *given by the 3-CNF* $(x_1 \vee (1 - x_2) \vee x_3) \cdot ((1 - x_1) \vee x_2 \vee x_4)$.

*(b) construct, for any solution to this instance of* 3-SAT*, the solution to the instance of* 3-PM.

**Lemma 22** 3-PM *is* NP*-complete.*

PROOF 3-PM $\in$ NP, as solutions can be verified efficiently.

3-PM is NP-hard, by Theorem 5 and Lemma 20.

## A.3 Packing

**Lemma 23** *For any finite set $S$ and any $\varnothing \notin \Sigma \subseteq 2^S$, we have*

$$\left| \bigcup_{U \in \Sigma} U \right| \leq \sum_{U \in \Sigma} |U|\ . \tag{A.35}$$

PROOF For $|\Sigma| = 0$, the statement is obviously correct.

For $|\Sigma| > 0$, there exists a $V \in \Sigma$ and

$$\begin{aligned} \left| \bigcup_{U \in \Sigma} U \right| = \left| V \cup \bigcup_{U \in \Sigma \setminus \{V\}} U \right| &= |V| + \left| \bigcup_{U \in \Sigma \setminus \{V\}} U \right| - \left| V \cap \bigcup_{U \in \Sigma \setminus \{V\}} U \right| \\ &\leq |V| + \left| \bigcup_{U \in \Sigma \setminus \{V\}} U \right| \\ &\leq |V| + \sum_{U \in \Sigma \setminus \{V\}} |U| && \text{by induction} \\ &= \sum_{U \in \Sigma} |U|\ . \end{aligned}$$

**Definition 41** For any finite set $S$ and any $\varnothing \notin \Sigma \subseteq 2^S$, the set $\Sigma$ is called a *packing* of $S$ iff the elements of $\Sigma$ are pairwise disjoint, i.e.:

$$\forall \{U, U'\} \in \binom{S}{2}: \quad U \cap U' = \varnothing \ . \tag{A.36}$$

**Lemma 24** *For any finite set $S$ and any $\varnothing \notin \Sigma \subseteq 2^S$, the set $\Sigma$ is a packing of $S$ iff*

$$\left| \bigcup_{U \in \Sigma} U \right| = \sum_{U \in \Sigma} |U| \ . \tag{A.37}$$

Proof (⇒) For $|\Sigma| = 0$, the statement (A.37) is obviously correct.

For $|\Sigma| > 0$, there exists a $V \in \Sigma$ and

$$\begin{aligned}
\left| \bigcup_{U \in \Sigma} U \right| = \left| V \cup \bigcup_{U \in \Sigma \setminus \{V\}} U \right| &= |V| + \left| \bigcup_{U \in \Sigma \setminus \{V\}} U \right| - \left| V \cap \bigcup_{U \in \Sigma \setminus \{V\}} U \right| \\
&= |V| + \left| \bigcup_{U \in \Sigma \setminus \{V\}} U \right| - \left| \bigcup_{U \in \Sigma \setminus \{V\}} \underbrace{(V \cap U)}_{\varnothing} \right| \\
&= |V| + \left| \bigcup_{U \in \Sigma \setminus \{V\}} U \right| && \text{by (A.36)} \\
&= |V| + \sum_{U \in \Sigma \setminus \{V\}} |U| && \text{by induction} \\
&= \sum_{U \in \Sigma} |U|
\end{aligned}$$

(⇐) For $|\Sigma| = 0$, the statement (A.37) is obviously correct.

For $|\Sigma| > 0$ and any $V \in \Sigma$:

$$\begin{aligned}
\left| \bigcup_{U \in \Sigma} U \right| &= \left| V \cup \bigcup_{U \in \Sigma \setminus \{V\}} U \right| \\
&= |V| + \left| \bigcup_{U \in \Sigma \setminus \{V\}} U \right| - \left| V \cap \bigcup_{U \in \Sigma \setminus \{V\}} U \right| \\
&= |V| + \left| \bigcup_{U \in \Sigma \setminus \{V\}} U \right| - \left| \bigcup_{U \in \Sigma \setminus \{V\}} (V \cap U) \right|
\end{aligned}$$

Therefore:

$$\begin{aligned}
\left| \bigcup_{U \in \Sigma \setminus \{V\}} (V \cap U) \right| &= |V| + \left| \bigcup_{U \in \Sigma \setminus \{V\}} U \right| - \left| \bigcup_{U \in \Sigma} U \right| \\
&\leq |V| + \sum_{U \in \Sigma \setminus \{V\}} |U| - \left| \bigcup_{U \in \Sigma} U \right| && \text{by Lemma 23} \\
&= \sum_{U \in \Sigma} |U| - \left| \bigcup_{U \in \Sigma} U \right| \\
&= \sum_{U \in \Sigma} |U| - \sum_{U \in \Sigma} |U| && \text{by (A.37)} \\
&= 0
\end{aligned}$$

Thus:

$$\forall U, V \in \binom{\Sigma}{2}: \quad U \cap V = \varnothing$$

## A.4 Covering

**Definition 42** For any set $S$ and any $\varnothing \notin \Sigma \subseteq 2^S$, the set $\Sigma$ is called a *cover* of $S$ iff

$$\bigcup_{U \in \Sigma} U = S \ . \tag{A.38}$$

A cover $\Sigma$ of $S$ is called *exact* (and a *partition* of $S$) iff it is also a packing of $S$, i.e., if the elements of $\Sigma$ are pairwise disjoint, i.e.:

$$\forall U, U' \in \binom{\Sigma}{2}: \quad U \cap U' = \varnothing \tag{A.39}$$

**Definition 43** Let $S$ be any set, and let $\varnothing \notin \Sigma \subseteq 2^S$.

Deciding whether there exists a $\Sigma' \subseteq \Sigma$ such that $\Sigma'$ is an exact cover of $S$ is called the instance of the *exact cover problem (*EC*)* with respect to $S$ and $\Sigma$.

Additionally, if $|S|$ is an integer multiple of three and any $U \in \Sigma$ is such that $|U| = 3$, the instance of EC with respect to $S$ and $\Sigma$ is also called the instance of the *exact cover by 3-sets problem (*EC-3*)* with respect to $S$ and $\Sigma$.

**Lemma 25** 3-PM $\leq_p$ EC-3

PROOF Consider any instance of 3-PM defined by $A, B, C$ and $T$. Without loss of generality, assume that $A, B, C$ are pairwise disjoint.

Now, consider the instance of EC-3 defined by

$$S = A \cup B \cup C \tag{A.40}$$
$$\Sigma = \{\{a, b, c\} \subseteq S \mid (a, b, c) \in T\} \tag{A.41}$$

Firstly, $|S|$ is an integer multiple of three, as $|A| = |B| = |C|$.

Secondly, every $U \in \Sigma$ is such that $|U| = 3$, by construction.

Thirdly, $\Sigma' \subseteq \Sigma$ is an exact cover of $S$ iff the triples defined uniquely by the elements of $\Sigma'$ are a perfect 3-dimensional matching of $A \times B \times C$, by construction.

**Lemma 26** EC-3 *is* NP*-complete.*

PROOF EC-3 $\in$ NP, as solutions can be verified efficiently.

EC-3 is NP-hard, by Lemmata 22 and 25.

**Lemma 27** EC *is* NP*-complete.*

PROOF EC $\in$ NP, as solutions can be verified efficiently.

EC is NP-hard, as EC-3 is NP-hard and EC-3 $\subseteq$ EC.

**Definition 44** For any $k \in \mathbb{N}$, deciding whether there exists a $\Sigma' \subseteq \Sigma$ such that (i) $\Sigma'$ is a cover of $S$ and (ii) $|\Sigma'| \leq k$ is called the instance of the *set cover problem (*SET-COVER*)* with respect to $S$, $\Sigma$ and $k$.

**Lemma 28** EC-3 $\leq$ SET-COVER

Proof For any instance $(S, \Sigma)$ of EC-3, let $m \in \mathbb{N}$ such that $|S| = 3m$.

Consider the instance of SET-COVER defined by $(S, \Sigma, m)$.

We show for any $\Sigma' \subseteq \Sigma$: $\Sigma'$ solves the instance of EC-3 iff $\Sigma'$ solves the instance of SET-COVER.

($\Rightarrow$) If $\Sigma'$ is a solution to the instance of EC-3, we have $|\Sigma'| \leq m$ because

$$3m = |S| \overset{(A.38)}{=} \left| \bigcup_{U \in \Sigma'} U \right| \overset{(A.39) \text{ and Lemma 24}}{=} \sum_{U \in \Sigma'} |U| = \sum_{U \in \Sigma'} 3 = 3|\Sigma'| \ .$$

($\Leftarrow$) If $\Sigma'$ is a solution to the instance of SET-COVER, we have

$$\left| \bigcup_{U \in \Sigma'} U \right| = \sum_{U \in \Sigma'} |U|$$

because

$$3m = |S| \overset{(A.38)}{=} \left| \sum_{U \in \Sigma'} U \right| \overset{\text{Lemma 23}}{\leq} \sum_{U \in \Sigma'} |U| = \sum_{U \in \Sigma'} 3 = 3|\Sigma'| \overset{|\Sigma'| \leq m}{\leq} 3m \ .$$

Thus, $\Sigma'$ is a packing of $S$, by Lemma 24.

Hence, $\Sigma'$ is a solution to the instance of EC-3.

## A.5 Coloring

**Definition 45** Let $G = (V, E)$ be any graph.

For any $m \in \mathbb{N}$ and any $c : V \to E$, the map $c$ is called an *m-coloring* of $G$ iff $\forall \{v, w\} \in E$: $c(v) \neq c(w)$. $G$ is called *m-colorable* iff there exists an $m$-coloring of $G$.

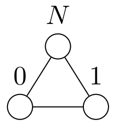
$G$ is called *m-chromatic*, and $m$ is called the *chromatic number* of $G$, iff $m$ is the minimal natural number such that $G$ is *m-colorable*.

**Definition 46** For any graph $G = (V, E)$ and any $m \in \mathbb{N}$, deciding whether $G$ is *m-colorable* is called the instance of the *m coloring problem (m-*COLORING*)* with respect to $G$ and $m$.

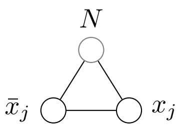
**Theorem 6 (Karp (1972))** 3-SAT $\leq_p$ 3-COLORING

Proof Given any instance of 3-SAT, we define a graph $G = (V, E)$ as follows:

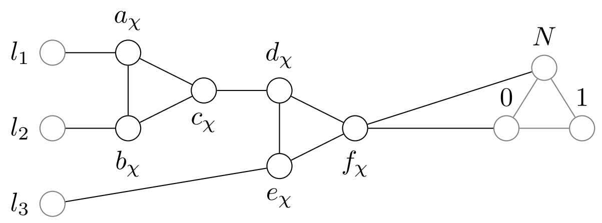
- Let $V$ contain a complete subgraph of three special nodes labeled 0, 1 and $N$. This subgraph is commonly called a *palette*.



- For every variable $x_j$ in the given instance of 3-SAT, let $V$ contain two nodes labeled $x_j$ and $\bar{x}_j$, respectively. Let both these nodes be connected to the palette node labeled $N$. Moreover, let the nodes labeled $x_j$ and $\bar{x}_j$ be connected by an edge:



- For any clause $(l_1 \vee l_2 \vee l_3) = \chi$ in the given instance of 3-SAT, let $V$ contain the following subgraph in which $l_j$ denotes the node labeled $x_k$ iff $l_j = x_k$, and $l_j$ denotes the node labeled $\bar{x}_k$ iff $l_j = 1 - x_k$:



Observe that the size of $G$ is polynomially bounded by the size of the given instance of 3-SAT.

**Exercise 18** *Complete the proof sketched above by showing that the instance of* 3-SAT *has a solution iff (!) $G$ is 3-colorable.*

**Lemma 29 (Karp (1972))** 3-COLORING *is* NP*-complete.*

PROOF 3-COLORING $\in$ NP, as solutions can be verified efficiently.
3-COLORING is NP-hard by Lemma 20 and Theorem 6.

**Lemma 30** $m$-COLORING *is* NP*-complete.*

PROOF $m$-COLORING $\in$ NP, as solutions can be verified efficiently.
$m$-COLORING is NP-hard, as 3-COLORING is NP-hard and 3-COLORING $\subseteq$ $m$-COLORING.

# Bibliography

Bansal, N., A. Blum, and S. Chawla
2004. Correlation clustering. *Machine Learning*, 56(1–3):89–113.

Besag, J.
1986. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, 48(3):259–302.

Chopra, S. and M. Rao
1993. The partition problem. *Mathematical Programming*, 59:87–115.

Cook, S. A.
1971. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*.

Dahlhaus, E., D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis
1994. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894.

Demaine, E. D., D. Emanuel, A. Fiat, and N. Immorlica
2006. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2):172–187.

Grötschel, M., M. Jünger, and G. Reinelt
1984. A cutting plane algorithm for the linear ordering problem. *Operations Research*, 32(6):1195–1220.

Haussler, D.
1988. Quantifying inductive bias: Ai learning algorithms and valiant's learning framework. *Artificial Intelligence*, 36(2):177–221.

Horňáková, A., J.-H. Lange, and B. Andres
2017. Analysis and optimization of graph decompositions by lifted multicuts. In *ICML*.

Karp, R. M.
1972. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, eds., The IBM Research Symposia Series, Pp. 85–103. Plenum Press, New York.

Kernighan, B. W. and S. Lin
1970. An efficient heuristic procedure for partitioning graphs. *Bell Systems Technical Journal*, 49(2):291–307.

Kschischang, F. R., B. J. Frey, and H.-A. Loeliger
2001. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519.

Martí, R. and G. Reinelt
2011. *The Linear Ordering Problem.* Springer.