ECEN 655: Advanced Channel Coding Spring 2014

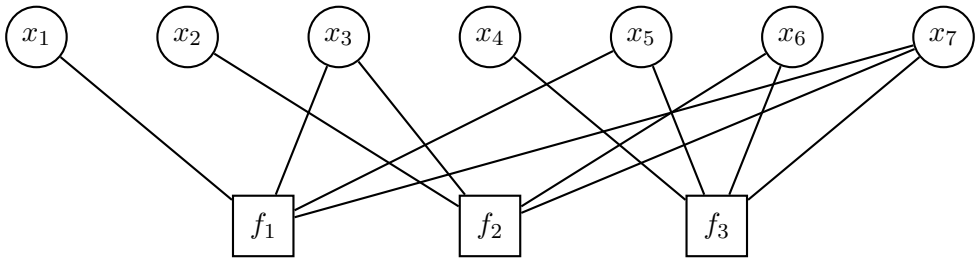
Lecture 5 — 01/28/2014

Prof. Henry Pfister *Scribe: Ying Wang*

1 Overview

In the last lecture, we talked about marginalization via message passing and conditional independence for factor graphs. In this lecture, we'll talk about how to apply the message passing algorithm for the decoding of LDPC codes.

2 Decoding of LDPC codes



LDPC codes are linear codes defined by $xH^T = 0$ for all c.w. $x \in \mathcal{C}$

- H is an $r \times n$ sparse parity-check matrix for the code
- Ensembles of LDPC codes can be defined by specifying the bit and check degrees and a random permutation that connects them

One can build a factor graph (FG) for binary LDPC codes by associating bit and check nodes with columns and rows of H

- Bits are VNs in FG with local factors from channel observations
- Checks are FNs defined by: $f_{\text{even}}(x_1^d) = I(x_1 \oplus \cdots \oplus x_d = 0)$
- The indicator function of the overall code factors into a product of indicators for the local parity checks

$$\mathbf{1}_{\mathcal{C}}(x_1^n) = \prod_{i=1}^{r} f_{\text{even}}(x_{F(i)})$$

2.1 VN and CN updates in LLR domain

Normalized binary messages are given by scalar $\mu(1) = 1 - \mu(0)$. One can also use the likelihood ratio (LR) $\frac{\mu(0)}{\mu(1)}$ or the log likelihood-ratio (LLR) $L = \ln \frac{\mu(0)}{\mu(1)}$.

For inference, LLR messages contain all the information:

$$L_{i\to a}^{(t)} = \ln \frac{\mu_{i\to a}^{(t)}(0)}{\mu_{i\to a}^{(t)}(1)} \qquad \hat{L}_{a\to i}^{(t)} = \ln \frac{\hat{\mu}_{a\to i}^{(t)}(0)}{\hat{\mu}_{a\to i}^{(t)}(1)}$$

Recall that the VN message-passing update is:

$$\mu_{i\to a}^{(t+1)}(x_i) = \prod_{b\in F(i)\backslash a} \hat{\mu}_{b\to i}^{(t)}(x_i)$$

In the LLR domain, this simplifies to

$$L_{i\to a}^{(t+1)} = \ln \frac{\mu_{i\to a}^{(t+1)}(0)}{\mu_{i\to a}^{(t+1)}(1)} = \ln \frac{\prod_{b\in F(i)\backslash a} \hat{\mu}_{b\to i}^{(t)}(0)}{\prod_{b\in F(i)\backslash a} \hat{\mu}_{b\to i}^{(t)}(1)} = \sum_{b\in F(i)\backslash a} \hat{L}_{b\to i}^{(t)}$$

Recall that the CN message-passing update is:

$$\hat{\mu}_{a\to i}^{(t)}(x_i) = \sum_{x_{V(a)}\backslash x_i} f_1(x_{V(a)}) \prod_{j\in V(a)\backslash i} \mu_{j\to a}^{(t)}(x_j)$$

In the LLR domain, this gives

$$\hat{L}_{a\to i}^{(t)} = \ln \frac{\hat{\mu}_{a\to i}^{(t)}(0)}{\hat{\mu}_{a\to i}^{(t)}(1)} = \ln \frac{\sum_{x_{V(a)}:x_i=0} f_1(x_{V(a)}) \prod_{j\in V(a)\backslash i} \mu_{j\to a}^{(t)}(x_j)}{\sum_{x_{V(a)}:x_i=1} f_1(x_{V(a)}) \prod_{j\in V(a)\backslash i} \mu_{j\to a}^{(t)}(x_j)}$$

2.2 Simplified Summation

For binary LDPC codes, many of the summations naively include terms that are known to be zero. For example, the "factor" associated with the variable node,

$$\begin{aligned} f(x_1, \ldots, x_d) &= I(x_1 = x_2 = \cdots = x_d) \\ &= \mathbf{1}_{\{x_1=x_2=\cdots=x_d\}}(x_1^d), \end{aligned}$$

is zero except for the arguments $00\cdots00$ and $11\cdots11$. Therefore, we find that

$$\sum_{x_1^d\backslash x_d} f(x_1, \cdots, x_d) \prod_{i=1}^{d} \hat{\mu}_{\to i}(x_i) = \begin{cases} \prod_{i=1}^{d} \hat{\mu}_{\to i}(0) & \text{if } x_d = 0 \\ \prod_{i=1}^{d} \hat{\mu}_{\to i}(1) & \text{if } x_d = 1 \end{cases}$$

Likewise, the check node factor is given by

$$\begin{aligned} f(x_1, \ldots, x_d) &= I(x_1 \oplus x_2 \oplus \cdots \oplus x_d = 0) \\ &= \mathbf{1}_{\{x_1 \oplus x_2 \oplus \cdots \oplus x_d = 0\}}(x_1^d) \\ &= \begin{cases} 0 & \text{if odd parity} \\ 1 & \text{if even parity.} \end{cases} \end{aligned}$$

In this case, the function is zero for exactly half of the possible arguments.

2.3 Generating functions

In combinatorics, a generating function is an algebraic object that is used to organize combinatorial objects into groups by weight. For example, let $\mathcal{A}$ be a set where each element a has a non-negative integer weight $w(a)$. We use A_h to denote the number of elements in $\mathcal{A}$ with weight h. Then, the associated generating function is

$$A(x) = \sum_{a \in \mathcal{A}} x^{w(a)} = \sum_{h \geq 0} A_h x^h.$$

Likewise, one can define $B(x)$ for the sequence B_h associated with the set $\mathcal{B}$. Now, consider organizing all elements in $\mathcal{C} = \mathcal{A} \times \mathcal{B}$ by weight (assuming the product weight is additive $w((a,b)) = w(a) + w(b)$). This gives

$$\begin{aligned} C(x) &= \sum_{h \geq 0} C_h x^h \\ &= \sum_{a \in \mathcal{A}} \sum_{b \in \mathcal{B}} x^{w((a,b))} \\ &= \sum_{a \in \mathcal{A}} \sum_{b \in \mathcal{B}} x^{w(a)} x^{w(b)} \\ &= \left(\sum_{a \in \mathcal{A}} x^{w(a)} \right) \left(\sum_{b \in \mathcal{B}} x^{w(b)} \right) \\ &= A(x) B(x). \end{aligned}$$

Example: Consider $A(x) = 2x + x^2$ and $B(x) = x^2 + x^3 + x^4$. Then,

$$(2x + x^2)(x^2 + x^3 + x^4) = (1 \cdot x + 1 \cdot x + 1 \cdot x^2) \cdot (1 \cdot x^2 + 1 \cdot x^3 + 1 \cdot x^4)$$

has 9 terms in the expanded product. There is a one-to-one correspondence between these 9 terms and the ways to pick one object from each set. Multiplying, we find that $C(x) = 2x^3 + 3x^4 + 3x^5 + x^6$.

2.4 Characteristic Function

Let $A(x) = \sum_h A_h x^h$ and assume that $\mathbb{P}(X = h) = A_h/A(1)$. Thus, X is the weight of an object drawn randomly from the set $\mathcal{A}$. Recall that the characteristic function of X is

$$\phi(s) = \mathbb{E}\left[e^{sX}\right] = \sum_{h\geq 0} \frac{A_h}{A(1)} e^{sh} = \frac{A(e^s)}{A(1)}$$

Example: Enumerating weighted patterns of bits with odd/even weight.

Let $\mu_{i\to}(0)$, $\mu_{i\to}(1)$ for $i = 1, \ldots, d$ represent real numbers associated binary values 0,1. Consider the following expression

$$\begin{aligned}
\phi(z) &= \prod_{i=1}^{d} \sum_{x_i\in\{0,1\}} \mu_{i\to}(x_i) z^{x_i} \\
&= \prod_{i=1}^{d} (\mu_{i\to}(0) + z\mu_{i\to}(1)) \\
&= \sum_{x_1^d\in\{0,1\}^d} \prod_{i=1}^{d} \mu_{i\to}(x_i) \cdot z^{w_H(x_1^d)}
\end{aligned}$$

From this, we can find the even-parity terms of $\phi(z)$ (i.e. $z^0, z^2, z^4 \ldots$)

$$\frac{1}{2}[\phi(z) + \phi(-z)] = \sum_{x_1^d\in\{0,1\}^d} f_{\text{even}}(x_1^d) \prod_{i=1}^{d} \mu_{i\to}(x_i) z^{w_H(x_1^d)}$$

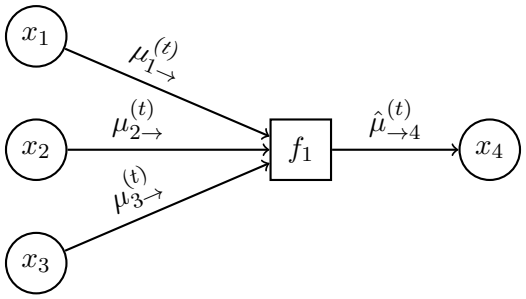
The simplified even parity sum is

$$\begin{aligned}
\frac{1}{2}\left[\phi(1) + \phi(-1)\right] &= \sum_{x_1^d\in\{0,1\}^d} \prod_{i=1}^{d} \mu_{i\to}(x_j) \frac{1}{2}\left[(1)^{w_H(x_1^d)} + (-1)^{w_H(x_1^d)}\right] \\
&= \sum_{x_1^d\in\{0,1\}^d} \prod_{i=1}^{d} \mu_{i\to}(x_j)\, I\left(w_H(x_1^d) \text{ is even}\right) \\
&= \sum_{x_1^d\in\{0,1\}^d} f_{\text{even}}(x_1^d) \prod_{i=1}^{d} \mu_{i\to}(x_j)
\end{aligned}$$

Similarly, the simplified odd parity sum is

$$\begin{aligned}\frac{1}{2}\left[\phi(1)-\phi(-1)\right] &= \sum_{x_1^d\in\{0,1\}^d}\prod_{i=1}^{d}\mu_{i\to}(x_j)\frac{1}{2}\left[(1)^{w_H(x_1^d)}-(-1)^{w_H(x_1^d)}\right]\\ &= \sum_{x_1^d\in\{0,1\}^d}\prod_{i=1}^{d}\mu_{i\to}(x_j)\, I\left(w_H(x_1^d)\text{ is odd}\right)\\ &= \sum_{x_1^d\in\{0,1\}^d} f_{\text{odd}}(x_1^d)\prod_{i=1}^{d}\mu_{i\to}(x_j)\end{aligned}$$

2.5 Simplified CN update for even parity constraint in LLR domain



For degree 4, the CN message-passing update is:

$$\begin{aligned}\hat{L}_{\to 4}^{(t)} = \ln\frac{\hat{\mu}_{\to 4}^{(t)}(0)}{\hat{\mu}_{\to 4}^{(t)}(1)} &= \ln\frac{\sum_{x_1^3\in\{0,1\}^3} f_{\text{even}}(x_1^3)\prod_{i=1}^{3}\mu_{j\to}^{(t)}(x_j)}{\sum_{x_1^4:x^4=1} f_{\text{even}}(x_1^4)\prod_{i=1}^{3}\mu_{j\to}^{(t)}(x_j)}\\ &= \ln\frac{\sum_{x_1^3\in\{0,1\}^3} f_{\text{even}}(x_1^3)\prod_{i=1}^{3}\mu_{j\to}^{(t)}(x_j)}{\sum_{x_1^3\in\{0,1\}^3} f_{\text{odd}}(x_1^3)\prod_{i=1}^{3}\mu_{j\to}^{(t)}(x_j)}\end{aligned}$$

For the general degree d:

$$\begin{aligned}\hat{L}_{\to d}^{(t)} &= \ln\frac{\prod_{j=1}^{d-1}(\mu_{j\to}^{(t)}(0)+\mu_{j\to}^{(t)}(1))+\prod_{j=1}^{d-1}(\mu_{j\to}^{(t)}(0)-\mu_{j\to}^{(t)}(1))}{\prod_{j=1}^{d-1}(\mu_{j\to}^{(t)}(0)+\mu_{j\to}^{(t)}(1))-\prod_{j=1}^{d-1}(\mu_{j\to}^{(t)}(0)-\mu_{j\to}^{(t)}(1))}\\ &= \ln\frac{1+\prod_{j=1}^{d-1}\frac{\mu_{j\to}(0)-\mu_{j\to}(1)}{\mu_{j\to}(0)+\mu_{j\to}(1)}}{1-\prod_{j=1}^{d-1}\frac{\mu_{j\to}(0)-\mu_{j\to}(1)}{\mu_{j\to}(0)+\mu_{j\to}(1)}}\\ &= \ln\frac{1+\prod_{j=1}^{d-1}\tanh(\frac{1}{2}L_{j\to})}{1-\prod_{j=1}^{d-1}\tanh(\frac{1}{2}L_{j\to})}\\ &= 2\tanh^{-1}(\prod_{j=1}^{d-1}\tanh(\frac{1}{2}L_{j\to}))\end{aligned}$$

Note that we used the following properties in the computation.

$$\tanh(\frac{1}{2}\ln\frac{a}{b}) = \frac{a-b}{a+b}$$
$$2\tanh^{-1}(z) = \ln\frac{1+z}{1-z}$$
$$h(z) = \ln\coth(\frac{|z|}{2})$$
$$h(h(z)) = z$$

3 Extension to a Commutative Semiring

A *commutative semiring* is a commutative ring where addition may not have inverse. It turns out that the distributive operations used to reduce the complexity of marginalization work similarly in any commutative semiring. For example, consider the maximization

$$\max_{x_1^n} f(x_1, \ldots, x_n) = \max_{x_1^n} \prod_{a \in F} f_a(x_{V(a)}).$$

In this case, the max operation (instead of the sum) can be pushed to the right to minimize computations.

The following table lists some important commutative semirings. For each operation, the identity is listed along with the mathematical definition.

Table 1: Commutative semiring

Set	Addtion	Multiplication	Name
$\mathbb{R} \geq 0$	$(+, 0)$	$(\cdot, 1)$	sum-product
$\mathbb{R} \geq 0$	$(\max, 0)$	$(\cdot, 1)$	max-product
$\mathbb{R} \geq 0 \bigcup \{\infty\}$	$(\min, \infty)$	$(+, 0)$	min-sum