Factor Graphs

Ruediger Urbanke, EPFL

February 15th, 2010

~ 2-3 hours

Distributive Law

$$ab + ac = a(b + c)$$

Distributive Law

$$ab + ac = a(b + c)$$

$$\sum_{i,j} a_i b_j \qquad\qquad \left(\sum_i a_i\right)\left(\sum_j b_j\right)$$

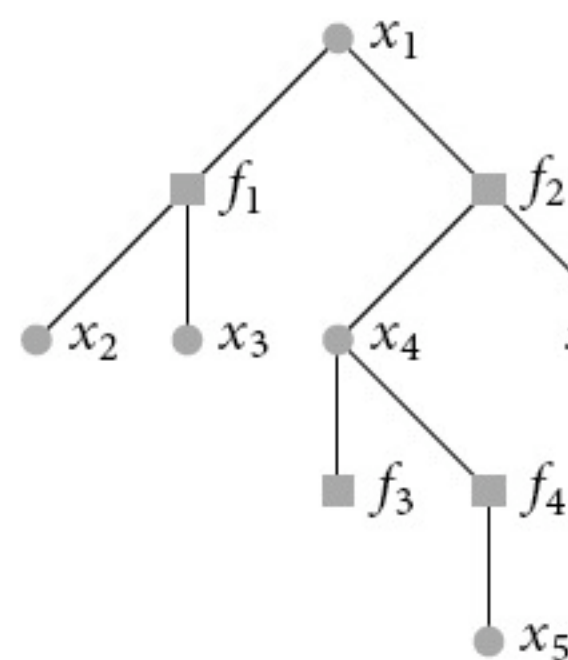
Example

Example

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = f_1(x_1, x_2, x_3) f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$$

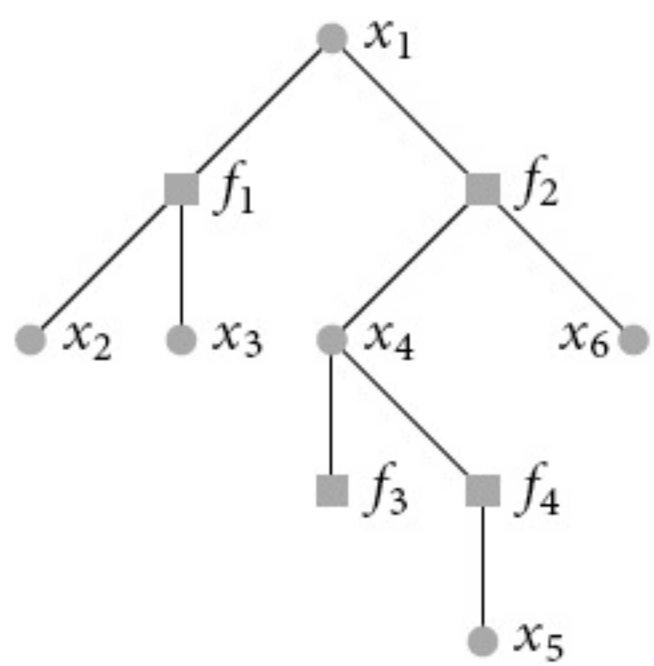
Example

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = f_1(x_1, x_2, x_3) f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$$



Example

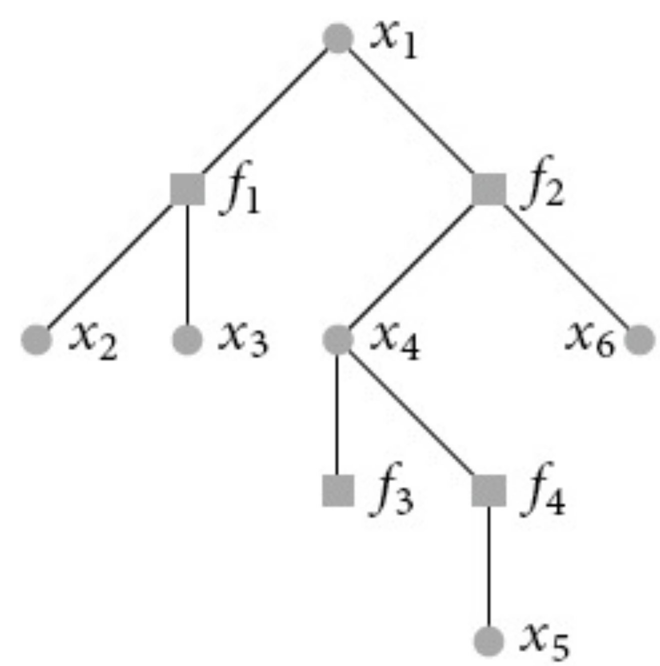
$$f(x_1, x_2, x_3, x_4, x_5, x_6) = f_1(x_1, x_2, x_3) f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$$



$$f(x_1) = \sum_{x_2, x_3, x_4, x_5, x_6} f(x_1, x_2, x_3, x_4, x_5, x_6) = \sum_{\sim x_1} f(x_1, x_2, x_3, x_4, x_5, x_6)$$

Example

$$f(x_1,x_2,x_3,x_4,x_5,x_6) = f_1(x_1,x_2,x_3)\,f_2(x_1,x_4,x_6)\,f_3(x_4)\,f_4(x_4,x_5)$$

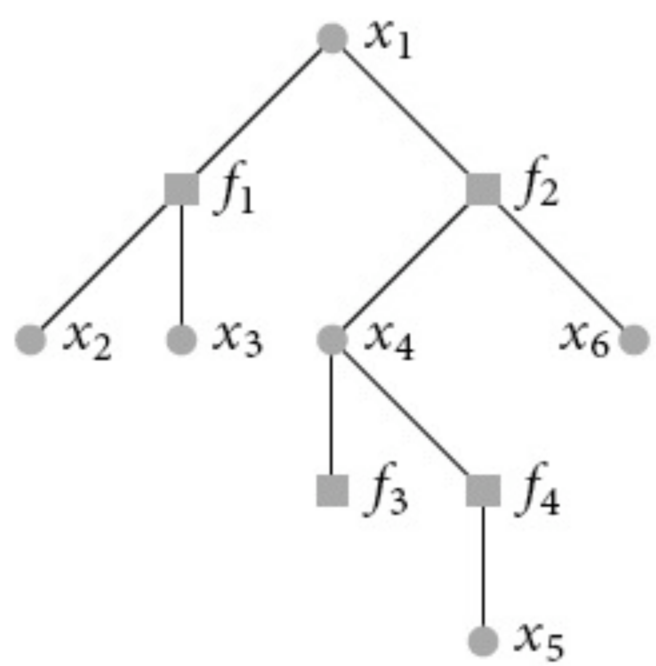


$$f(x_1) = \sum_{x_2,x_3,x_4,x_5,x_6} f(x_1,x_2,x_3,x_4,x_5,x_6) = \sum_{\sim x_1} f(x_1,x_2,x_3,x_4,x_5,x_6)$$

Note: $f(x_1)$ is a function; therefore, it takes on a distinct value for each value of x_1

Example

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = f_1(x_1, x_2, x_3) f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$$



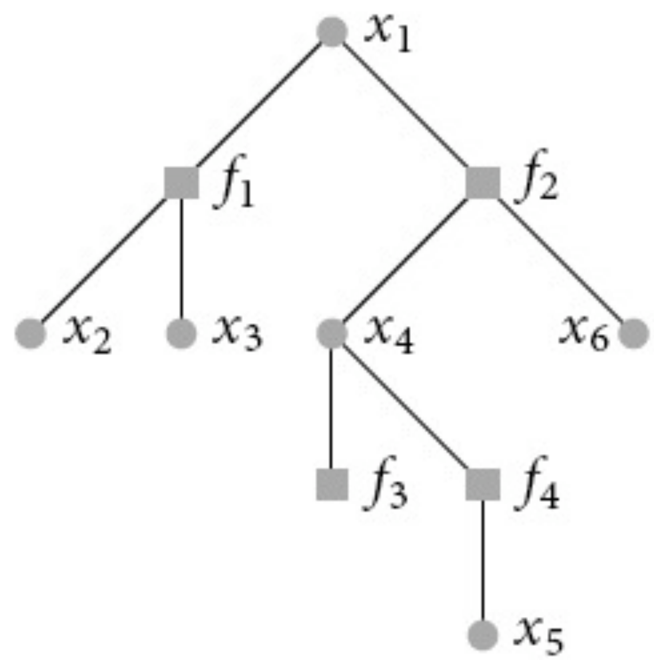
$$f(x_1) = \sum_{x_2, x_3, x_4, x_5, x_6} f(x_1, x_2, x_3, x_4, x_5, x_6) = \sum_{\sim x_1} f(x_1, x_2, x_3, x_4, x_5, x_6)$$

Note: $f(x_1)$ is a function; therefore, it takes on a distinct value for each value of x_1

$|\mathcal{X}|$ alphabet

Example

$$f(x_1, x_2, x_3, x_4, x_5, x_6) = f_1(x_1, x_2, x_3) f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$$



$$f(x_1) = \sum_{x_2, x_3, x_4, x_5, x_6} f(x_1, x_2, x_3, x_4, x_5, x_6) = \sum_{\sim x_1} f(x_1, x_2, x_3, x_4, x_5, x_6)$$

Note: $f(x_1)$ is a function; therefore, it takes on a distinct value for each value of x_1

$|\mathcal{X}|$ alphabet

$\Theta\left(|\mathcal{X}|^6\right)$ brute force complexity

Example

$$f(x_1) = \Big[\sum_{x_2,x_3} f_1(x_1,x_2,x_3)\Big]\Big[\sum_{x_4} f_3(x_4)\Big(\sum_{x_6} f_2(x_1,x_4,x_6)\Big)\Big(\sum_{x_5} f_4(x_4,x_5)\Big)\Big]$$

Example

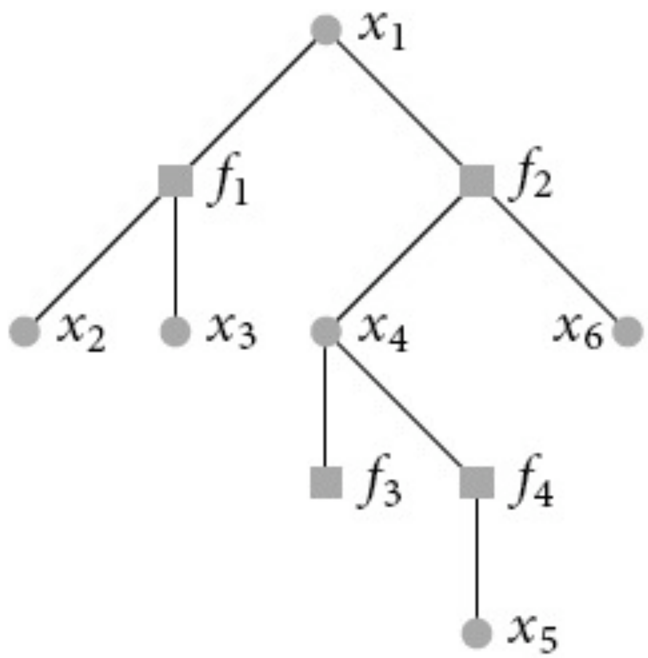
$$f(x_1) = \Big[\sum_{x_2,x_3} f_1(x_1, x_2, x_3)\Big]\Big[\sum_{x_4} f_3(x_4)\Big(\sum_{x_6} f_2(x_1, x_4, x_6)\Big)\Big(\sum_{x_5} f_4(x_4, x_5)\Big)\Big]$$

$\Theta\left(|\mathcal{X}|^3\right)$ complexity

Example

$$f(x_1) = \Big[\sum_{x_2,x_3} f_1(x_1,x_2,x_3)\Big]\Big[\sum_{x_4} f_3(x_4)\Big(\sum_{x_6} f_2(x_1,x_4,x_6)\Big)\Big(\sum_{x_5} f_4(x_4,x_5)\Big)\Big]$$

$\Theta\left(|\mathcal{X}|^3\right)$ complexity



Does there exist a systematic way to find this low complexity scheme using the structure of the graph?

Marginalization via Message Passing for Trees

Marginalization via Message Passing for Trees

$g(z) = \sum_{\sim z} g(z, \ldots)$

Marginalization via Message Passing for Trees

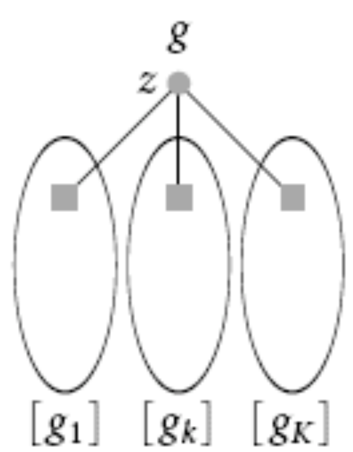
$$g(z) = \sum_{\sim z} g(z, \ldots)$$

$$g(z, \ldots) = \prod_{k=1}^{K} [g_k(z, \ldots)]$$

Marginalization via Message Passing for Trees

$$g(z) = \sum\nolimits_{\sim z} g(z, \dots)$$

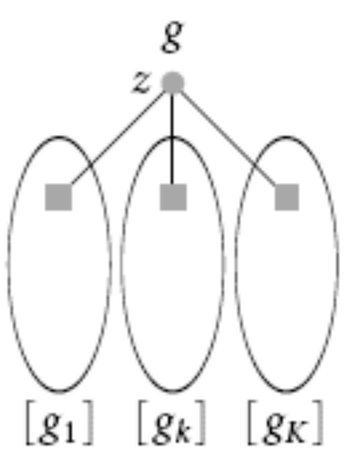
$$g(z, \dots) = \prod_{k=1}^{K} [g_k(z, \dots)]$$



Marginalization via Message Passing for Trees

$$g(z) = \sum_{\sim z} g(z, \ldots)$$

$$g(z, \ldots) = \prod_{k=1}^{K} [g_k(z, \ldots)]$$

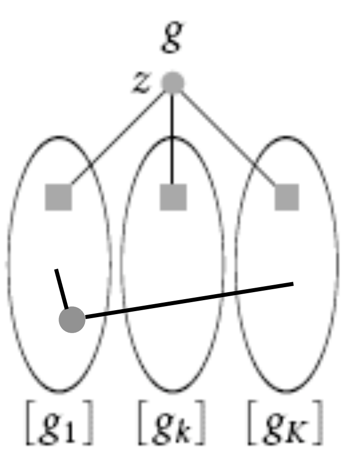


Note: the individual functions $g_k(z, \ldots)$
only share the variable z; all other
variables are "private"

Marginalization via Message Passing for Trees

$g(z) = \sum_{\sim z} g(z, \ldots)$

$$g(z, \ldots) = \prod_{k=1}^{K} [g_k(z, \ldots)]$$



Note: the individual functions $g_k(z, \ldots)$
only share the variable z; all other
variables are "private"

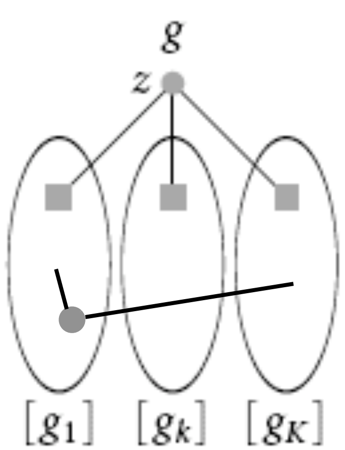
otherwise ... graph is not a tree

Marginalization via Message Passing for Trees

$g(z) = \sum_{\sim z} g(z, \dots)$

$$f(x_1) = \sum_{\sim x_1} f(x_1, x_2, x_3, x_4, x_5, x_6)$$

$$g(z, \dots) = \prod_{k=1}^{K} [g_k(z, \dots)]$$



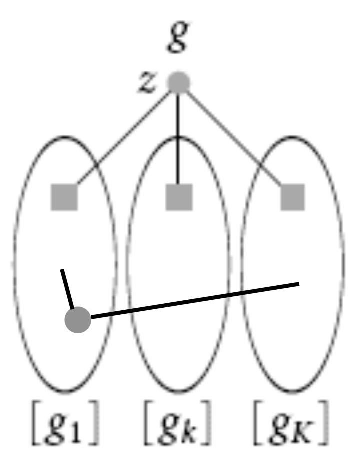
Note: the individual functions $g_k(z, \dots)$
only share the variable z; all other
variables are "private"

otherwise ... graph is not a tree

Marginalization via Message Passing for Trees

$$g(z) = \sum_{\sim z} g(z, \dots)$$

$$g(z, \dots) = \prod_{k=1}^{K} [g_k(z, \dots)]$$



Note: the individual functions $g_k(z, \dots)$ only share the variable z; all other variables are “private”

otherwise ... graph is not a tree

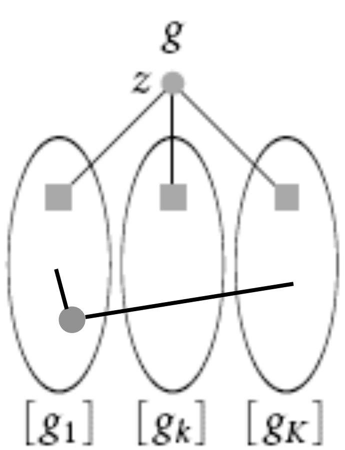
$$f(x_1) = \sum_{\sim x_1} f(x_1, x_2, x_3, x_4, x_5, x_6)$$

$$f(x_1, \dots) = [f_1(x_1, x_2, x_3)]\,[f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)]$$

Marginalization via Message Passing for Trees

$g(z) = \sum_{\sim z} g(z, \ldots)$

$$g(z, \ldots) = \prod_{k=1}^{K} [g_k(z, \ldots)]$$

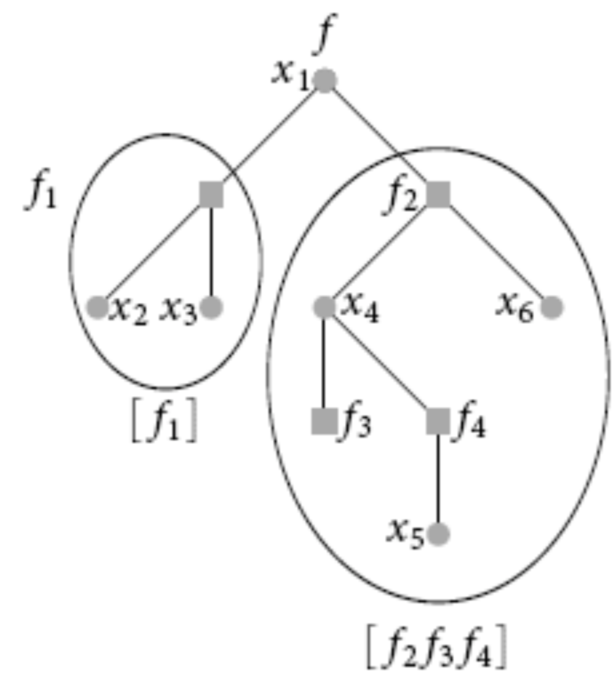


Note: the individual functions $g_k(z, \ldots)$
only share the variable z; all other
variables are "private"

otherwise ... graph is not a tree

$$f(x_1) = \sum_{\sim x_1} f(x_1, x_2, x_3, x_4, x_5, x_6)$$

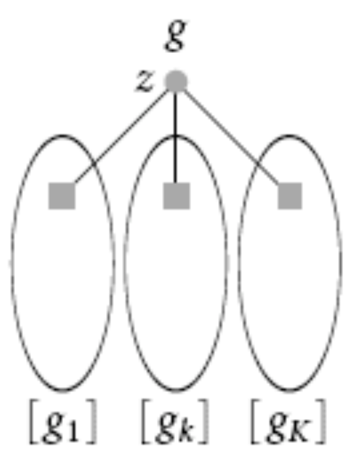
$$f(x_1, \ldots) = [f_1(x_1, x_2, x_3)] [f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)]$$



Marginalization via Message Passing for Trees

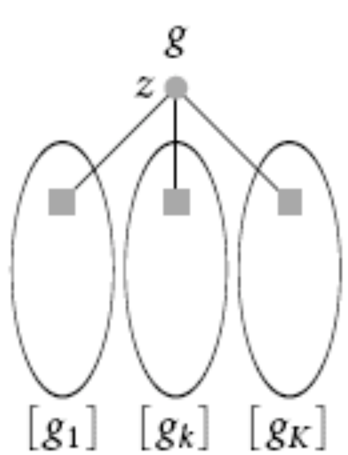
Marginalization via Message Passing for Trees

$$\sum_{\sim z} g(z, \dots) = \underbrace{\sum_{\sim z} \prod_{k=1}^{K} [g_k(z, \dots)]}_{\text{marginal of product}} = \underbrace{\prod_{k=1}^{K} \Big[\sum_{\sim z} g_k(z, \dots) \Big]}_{\text{product of marginals}}$$



Marginalization via Message Passing for Trees

$$\sum_{\sim z} g(z, \ldots) = \underbrace{\sum_{\sim z} \prod_{k=1}^{K} [g_k(z, \ldots)]}_{\text{marginal of product}} = \underbrace{\prod_{k=1}^{K} \Big[\sum_{\sim z} g_k(z, \ldots)\Big]}_{\text{product of marginals}}$$

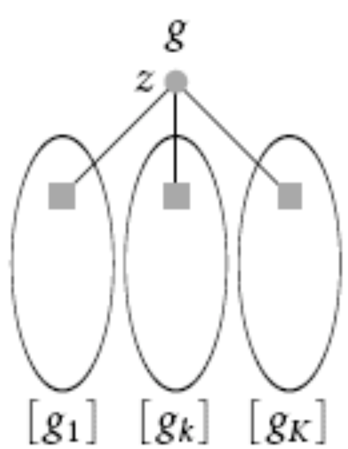


marginal $\sum_{\sim z} g(z, \ldots)$ is the product of the individual marginals

Marginalization via Message Passing for Trees

recall that g(z) is a function, taking a distinct value for each value of z

$$\sum_{\sim z} g(z,\ldots) = \underbrace{\sum_{\sim z} \prod_{k=1}^{K} [g_k(z,\ldots)]}_{\text{marginal of product}} = \underbrace{\prod_{k=1}^{K} \Big[\sum_{\sim z} g_k(z,\ldots) \Big]}_{\text{product of marginals}}$$



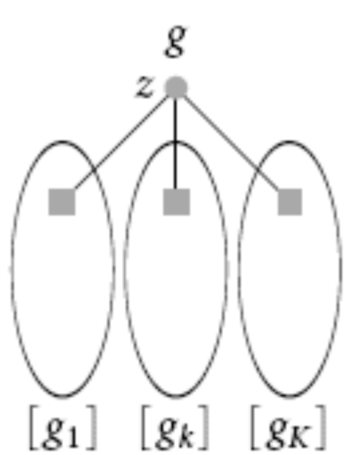
marginal $\sum_{\sim z} g(z,\ldots)$ is the product of the individual marginals

Marginalization via Message Passing for Trees

recall that g(z) is a function, taking a distinct value for each value of z

instead of computing g(z) directly by brute force we can first compute each of the functions $g_k(z)$; we then get g(z) by multiplying these functions $g_k(z)$

$$\sum_{\sim z} g(z,\ldots) = \underbrace{\sum_{\sim z} \prod_{k=1}^{K} [g_k(z,\ldots)]}_{\text{marginal of product}} = \underbrace{\prod_{k=1}^{K} \Big[\sum_{\sim z} g_k(z,\ldots)\Big]}_{\text{product of marginals}}$$



marginal $\sum_{\sim z} g(z,\ldots)$ is the product of the individual marginals

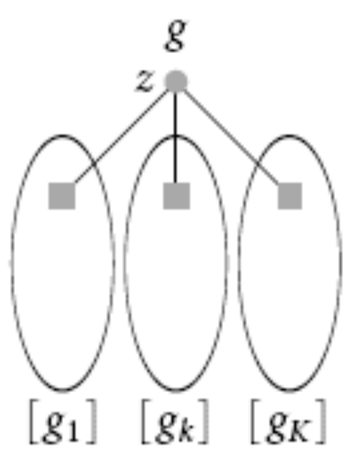
Marginalization via Message Passing for Trees

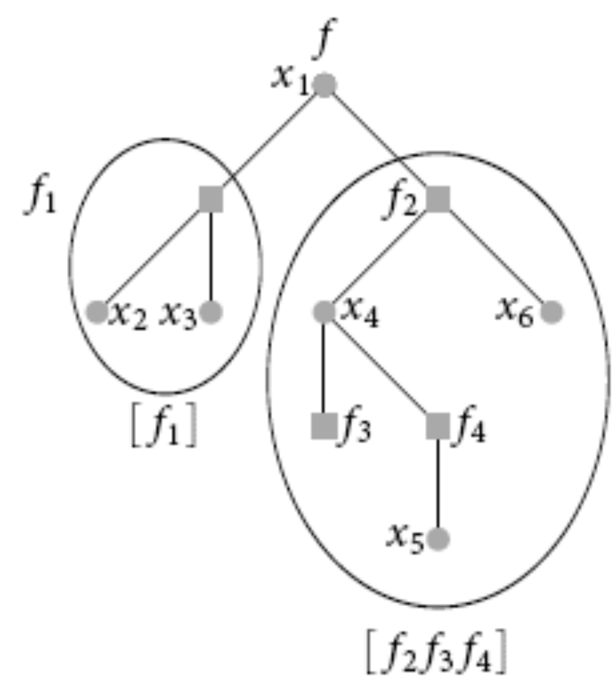
recall that g(z) is a function, taking a distinct value for each value of z

instead of computing g(z) directly by brute force we can first compute each of the functions $g_k(z)$; we then get g(z) by multiplying these functions $g_k(z)$

$$\sum_{\sim z} g(z,\ldots) = \underbrace{\sum_{\sim z} \prod_{k=1}^{K} [g_k(z,\ldots)]}_{\text{marginal of product}} = \underbrace{\prod_{k=1}^{K} \Big[\sum_{\sim z} g_k(z,\ldots)\Big]}_{\text{product of marginals}}$$

$$f(x_1) = \Big[\sum_{\sim x_1} f_1(x_1,x_2,x_3)\Big]\Big[\sum_{\sim x_1} f_2(x_1,x_4,x_6) f_3(x_4) f_4(x_4,x_5)\Big]$$





marginal $\sum_{\sim z} g(z,\ldots)$ is the product of the individual marginals

Marginalization via Message Passing for Trees

$$\sum_{\sim z} g(z,\dots) = \underbrace{\sum_{\sim z} \prod_{k=1}^{K} [g_k(z,\dots)]}_{\text{marginal of product}} = \underbrace{\prod_{k=1}^{K} \Big[\sum_{\sim z} g_k(z,\dots)\Big]}_{\text{product of marginals}}$$

Marginalization via Message Passing for Trees

$$\sum_{\sim z} g(z,\ldots) = \underbrace{\sum_{\sim z} \prod_{k=1}^{K} [g_k(z,\ldots)]}_{\text{marginal of product}} = \underbrace{\prod_{k=1}^{K} \Big[\sum_{\sim z} g_k(z,\ldots)\Big]}_{\text{product of marginals}}$$

$$g_k(z,\ldots)$$

Marginalization via Message Passing for Trees

$$\sum_{\sim z} g(z,\dots) = \underbrace{\sum_{\sim z} \prod_{k=1}^{K} [g_k(z,\dots)]}_{\text{marginal of product}} = \underbrace{\prod_{k=1}^{K} \Big[\sum_{\sim z} g_k(z,\dots)\Big]}_{\text{product of marginals}}$$

$$g_k(z,\dots)$$

$$g_k(z,\dots) = \underbrace{h(z,z_1,\dots,z_J)}_{\text{kernel}} \prod_{j=1}^{J} \underbrace{\big[h_j(z_j,\dots)\big]}_{\text{factors}}$$

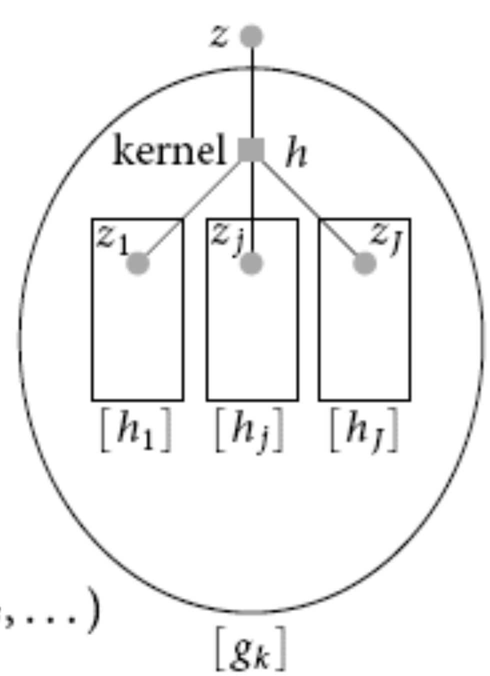
"kernel" $h(z,z_1,\dots,z_J)$

Marginalization via Message Passing for Trees

$$g_k(z,\ldots) = \underbrace{h(z,z_1,\ldots,z_J)}_{\text{kernel}} \prod_{j=1}^{J} \underbrace{[h_j(z_j,\ldots)]}_{\text{factors}}$$

"kernel" $h(z,z_1,\ldots,z_J)$

$g_k(z,\ldots)$



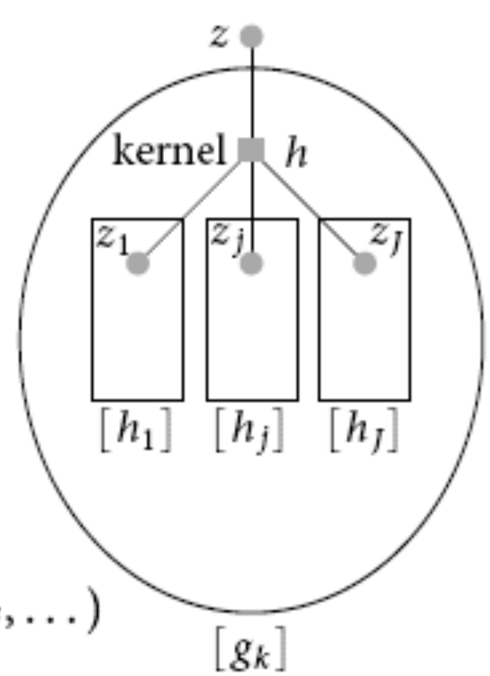
factors $h_j(z_j,\ldots)$

Marginalization via Message Passing for Trees

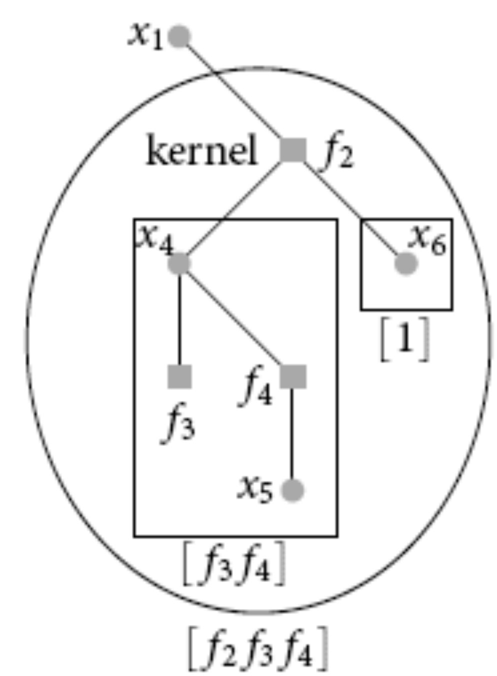
$$g_k(z,\dots) = \underbrace{h(z,z_1,\dots,z_J)}_{\text{kernel}} \prod_{j=1}^{J} \underbrace{[h_j(z_j,\dots)]}_{\text{factors}}$$

"kernel" $h(z,z_1,\dots,z_J)$

$g_k(z,\dots)$



factors $h_j(z_j,\dots)$



$$f_2(x_1,x_4,x_6)\, f_3(x_4)\, f_4(x_4,x_5) = \underbrace{f_2(x_1,x_4,x_6)}_{\text{kernel}} \underbrace{[f_3(x_4) f_4(x_4,x_5)]}_{x_4} \underbrace{[1]}_{x_6}.$$

Marginalization via Message Passing for Trees

$$\sum_{\sim z} g_k(z, \ldots) = \sum_{\sim z} h(z, z_1, \ldots, z_J) \prod_{j=1}^{J} \left[h_j(z_j, \ldots) \right]$$

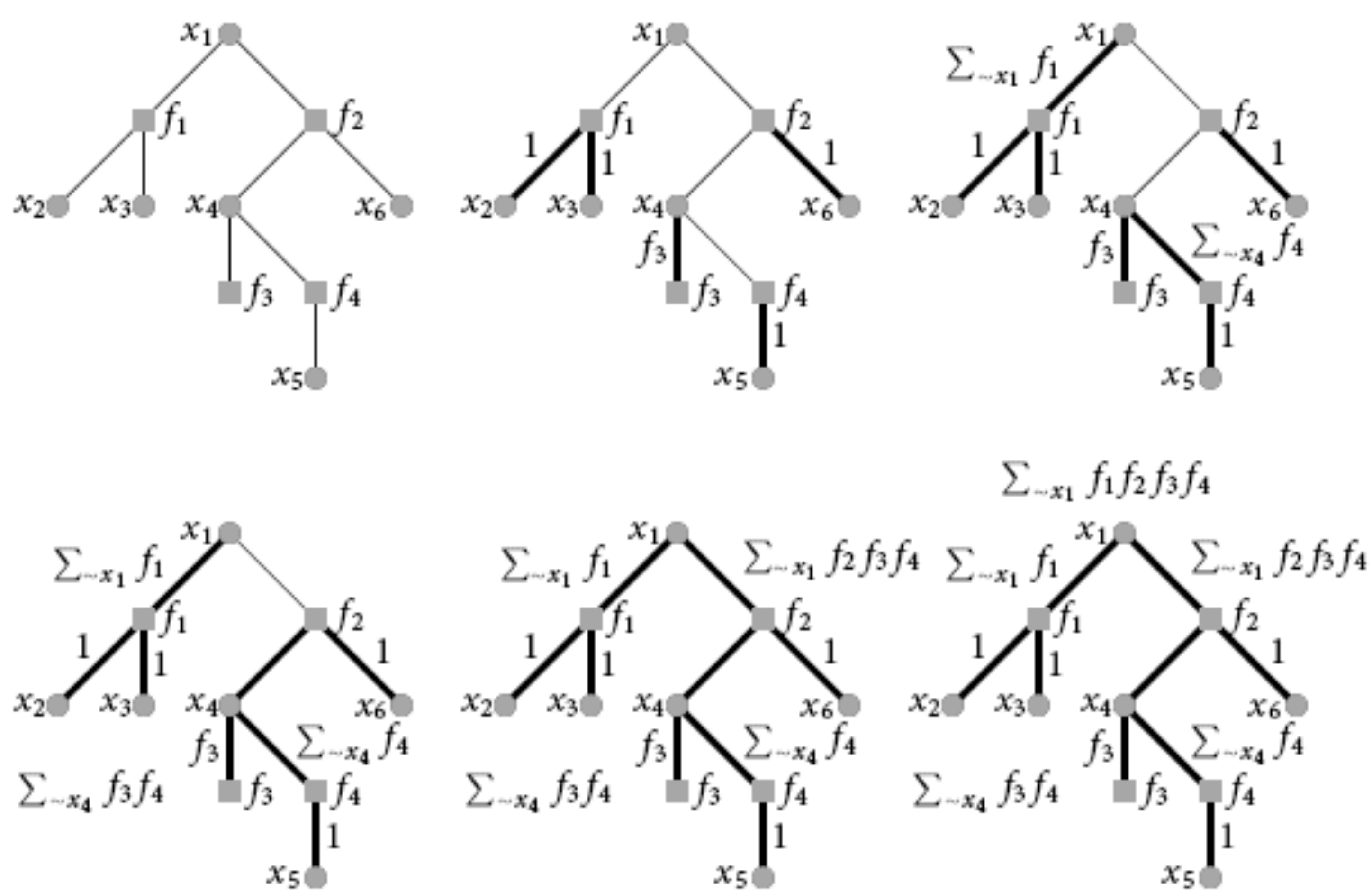
$$= \sum_{\sim z} h(z, z_1, \ldots, z_J) \underbrace{\prod_{j=1}^{J} \Big[\sum_{\sim z_j} h_j(z_j, \ldots) \Big]}_{\text{product of marginals}}$$

Marginalization via Message Passing for Trees

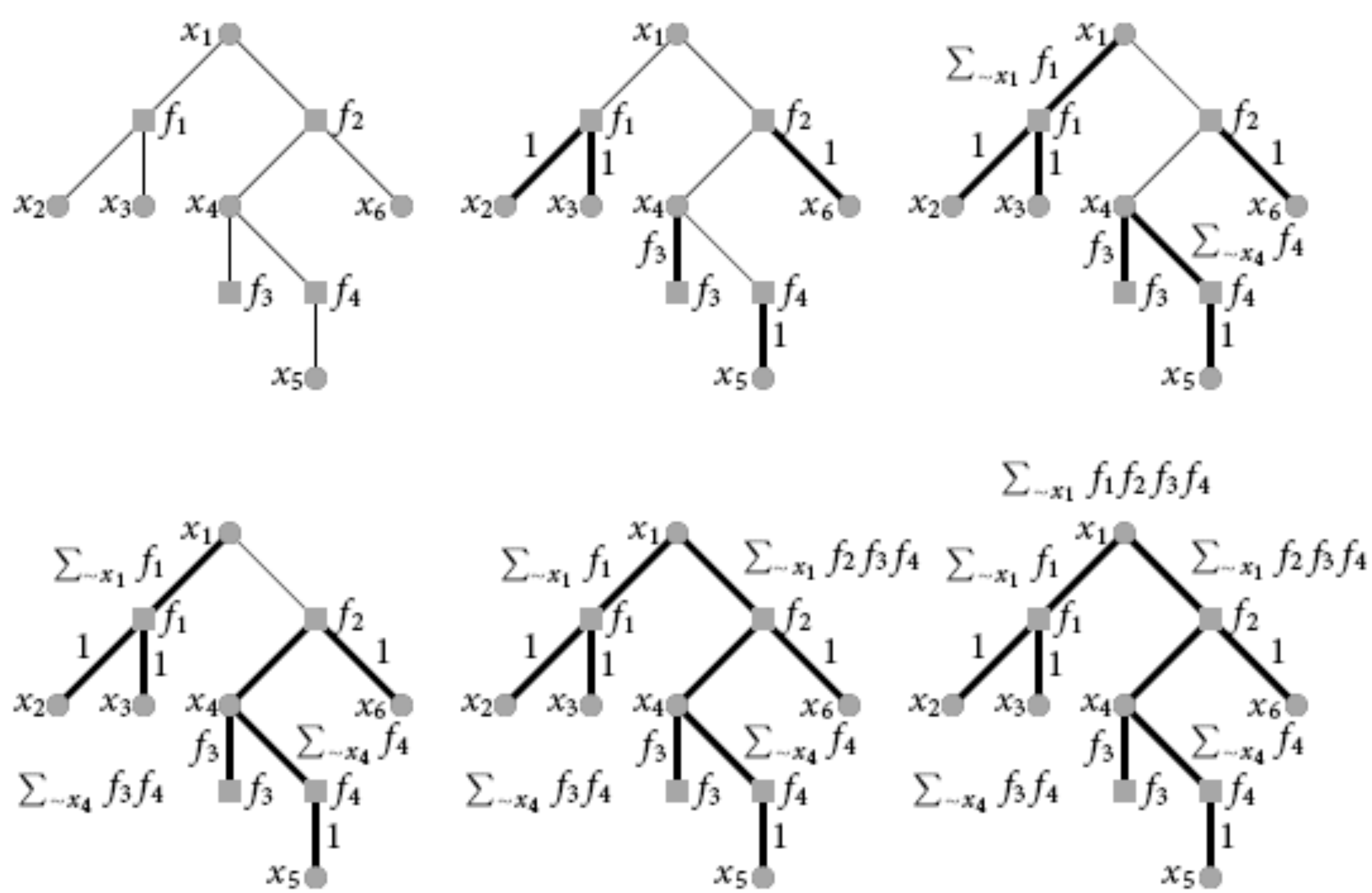
$$\sum_{\sim z} g_k(z,\dots) = \sum_{\sim z} h(z,z_1,\dots,z_J) \prod_{j=1}^{J} \left[h_j(z_j,\dots)\right]$$
$$= \sum_{\sim z} h(z,z_1,\dots,z_J) \underbrace{\prod_{j=1}^{J} \Big[\sum_{\sim z_j} h_j(z_j,\dots)\Big]}_{\text{product of marginals}}$$

$$f(x_1) = \Big[\sum_{\sim x_1} f_1(x_1,x_2,x_3)\Big]\Big[\sum_{\sim x_1} f_2(x_1,x_4,x_6) f_3(x_4) f_4(x_4,x_5)\Big]$$

Marginalization via Message Passing for Trees

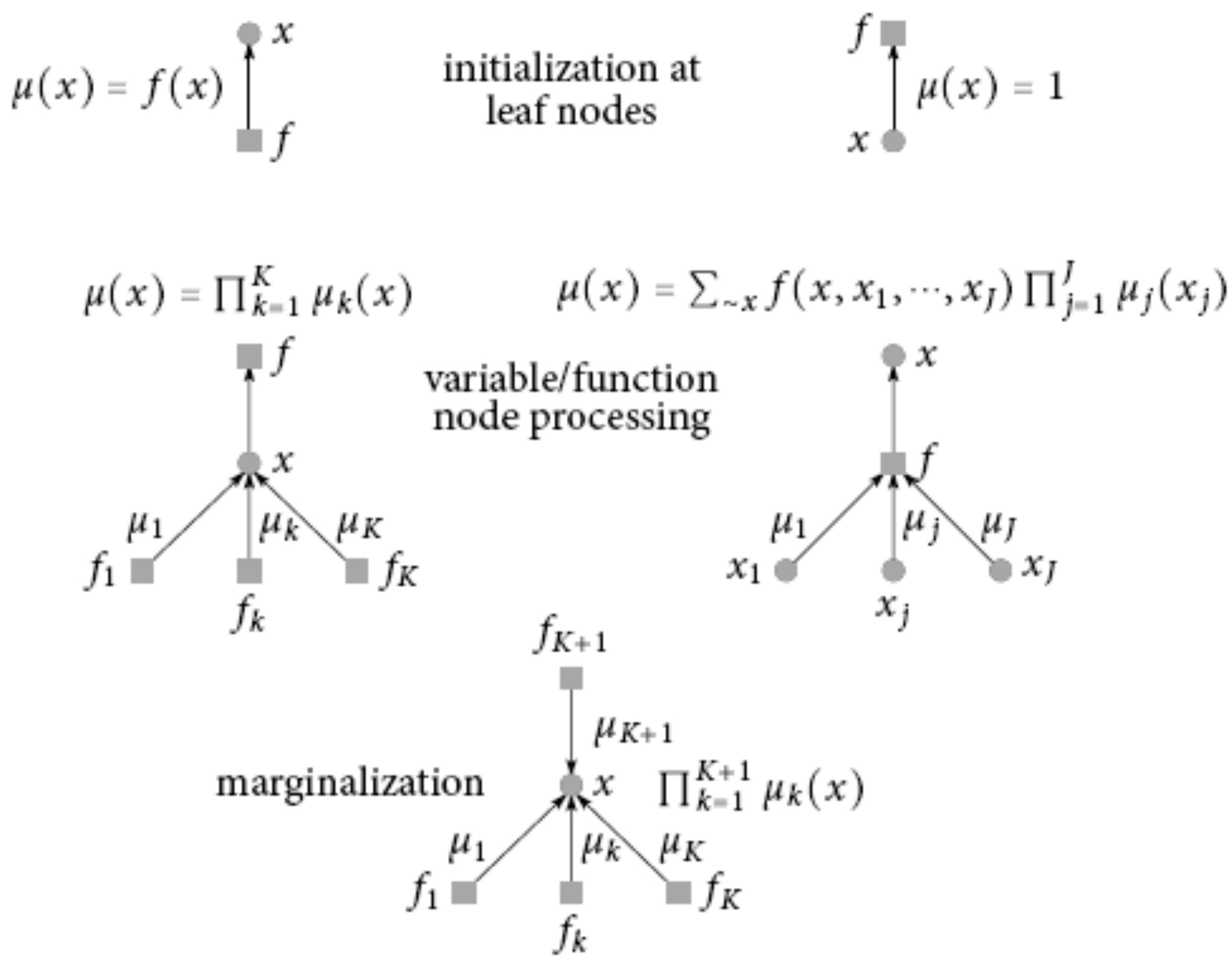


Marginalization via Message Passing for Trees



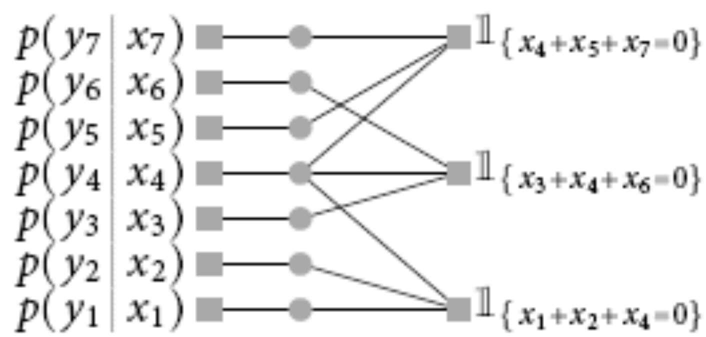
complexity proportional to highest degree

Message Passing Rules



Summary and Limitations

Summary and Limitations



References

R. G. Gallager, *Low-density parity-check codes*, IRE Trans. Inform. Theory, 8 (1962), pp. 21–28. [33, 66, 159, 264, 419, 434]

A. J. Viterbi, *Error bounds of convolutional codes and an asymptotically optimum decoding algorithm*, IEEE Trans. Inform. Theory, 13 (1967), pp. 260–269. [33, 34, 66]

L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, *Optimal decoding of linear codes for minimizing symbol error rate*, IEEE Trans. Inform. Theory, 20 (1974), pp. 284–287. [34, 39, 66]

R. M. Tanner, *A recursive approach to low complexity codes*, IEEE Trans. Inform. Theory, 27 (1981), pp. 533–547. [65, 161, 261, 263]

N. Wiberg, *Codes and Decoding on General Graphs*, PhD thesis, Linköping University, S-581 83, Linköping, Sweden, 1996. [65, 66, 263, 266, 366]

S. M. Aji and R. J. McEliece, *The generalized distributive law*, IEEE Trans. Inform. Theory, 46 (2000), pp. 325–343. [65, 66]

F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, *Factor graphs and the sum-product algorithm*, IEEE Trans. Inform. Theory, 47 (2001), pp. 498–519. [65, 66]