A General Algorithm for Distributing Information in a Graph

Srinivas M. Aji and Robert J. McEliece[1]
California Institute of Technology

Abstract — **We present a general "message-passing" algorithm for distributing information in a graph. Besides being interesting in its own right, this algorithm may help us to understand the approximate correctness of both the Gallager-Tanner-Wiberg algorithm, and the turbo-decoding algorithm.**

I. Introduction

In this paper we discuss a general information-distributing algorithm, similar to a class of algorithms well-known to the AI community [1, 2, 4], which has the Gallager-Tanner-Wiberg algorithm, the BCJR algorithm, the FFT algorithm, Viterbi's algorithm, the turbo decoding algorithm, and many other algorithms as special cases. Although this algorithm is guaranteed to give exact answers only in certain cases, unfortunately not including the cases of GTW or turbo-decoding, we hope that a close study of it will provide answers to the important question of why certain iterative decoding algorithms perform as well as they do.

II. The problem

Let $A_1, \ldots, A_n$ be finite sets, and let $x_1, \ldots, x_n$ be variables taking values in these sets. If $v = \{i_1, \ldots, i_r\}$ is a subset of $\{1, \ldots, n\}$, we denote the product $A_{i_1} \times \cdots \times A_{i_r}$ by A_v and the variable list $(x_{i_1}, \ldots, x_{i_r})$ by x_v. If $v = \{1, \ldots, n\}$, we denote the product A_v simply by A, and the variable list $\{x_1, \ldots, x_n\}$ simply by x.

Now let $V = \{v_1, \ldots, v_M\}$ be M subsets of $\{1, \ldots, n\}$, which, for reasons to be explained below, we call *vertices*. Suppose that for each $v \in V$, there is a function $\phi_v : A_v \to R$, where R is a *semiring*. (In particular, besides the expected interpretations, in a semiring, "+" may be interpreted as "min" and "·" may be interpreted as "+".) The functions ϕ_v are called the *local kernels*. We define the *global kernel* $F : A \to R$, as follows:

$$F(x_1, \ldots, x_n) = \prod_{v \in V} \phi_v(x_v). \tag{1}$$

With this setup, we can state our general problem: For one or more of the vertices $v \in V$, compute a table of the values of the *v-marginalization* of the global kernel F, which is the function F_v mapping $A_v \to R$, defined by

$$F_v(x_v) = \sum_{x_{v^c} \in A_{v^c}} F(x), \tag{2}$$

where in (2) v^c denotes the complement of v, i.e., $v^c = \{1, \ldots, n\} \setminus v$.

III. The algorithm

If the elements of V are viewed as the vertices of an undirected graph (V, E), an algorithm for solving the above problem can be based on the notion of "message passing." For an edge $\{v, w\} \in E$, the "message" sent from v to w is a table containing the values of a function $\mu_{v,w} : A_{v \cap w} \to R$. Initially, all such functions are defined to be identically one, but when a particular message $\mu_{v,w}$ is updated, the following rule is used:

$$\mu_{v,w}(x_{v \cap w}) = \sum_{x_{v \setminus w} \in A_{v \setminus w}} \phi_v(x_v) \prod_{\substack{\{u,v\} \in E \\ u \neq w}} \mu_{u,v}(x_{u \cap v}). \tag{3}$$

Similarly the "state" of a vertex $v \in V$ is defined to be a table containing the values of a function $\mu_v : A_v \to R$. Initially, μ_v is defined to be the local kernel $\phi_v(x_v)$, but when μ_v is updated, the following rule is used.

$$\mu_v(x_v) = \phi_v(x_v) \prod_{\{u,v\} \in E} \mu_{u,v}(x_{u \cap v}). \tag{4}$$

The hope is that after sufficiently many messages have been passed, the states of the selected vertices $v \in V$ will be, exactly or approximately, the desired marginalization tables.

When the vertices $v \in V$ are, or can be, organized as a hypertree [2], or equivalently, a junction tree [4], a message-passing algorithm of the type cited above can be shown to provide an exact solution to the stated problem, with complexity $O(\sum_{v \in V} d(v)|A_v|)$, where $d(v)$ denotes the number of vertices adjacent to v. This exact algorithm can be specialized to give the BCJR algorithm, Viterbi's algorithm, Pearl's "belief propagation" algorithm [1], and the FFT on any finite Abelian group. Interestingly, this algorithm can also be specialized to give the Gallager-Tanner-Wiberg algorithm [3], and the turbo decoding algorithm [5, 6], although an explanation of its "approximate correctness" in these cases has yet to be found.

[1]This work was supported by NSF grant no. NCR-9505975 and a grant from Pacific Bell.

References

[1] J. Pearl, *Probabilistic Reasoning in Intelligent Systems.* San Mateo, CA: Morgan Kaufmann, 1988.

[2] G. R. Shafer and P. P. Shenoy, "Probability propagation," *Ann. of Math. and Art. Intel.*, vol. 2 (1990), pp. 327–352.

[3] N. Wiberg, *Codes and Decoding on General Graphs.* Linköping Studies in Science and Technology, Dissertations no. 440. Linköping, Sweden, 1996.

[4] F. V. Jensen, *An Introduction to Bayesian Networks.* New York: Springer-Verlag, 1996.

[5] F. Kschischang and B. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," submitted to *IEEE J. Sel. Areas Comm.*, Sept. 1996.

[6] D. J. C. MacKay, R. J. McEliece, and J. -F. Cheng, "Turbo decoding as an instance of Pearl's 'belief propagation' algorithm," submitted to *IEEE J. Sel. Areas Comm.*, Sept. 1996.