On Calculating the Capacity of an Infinite-Input Finite (Infinite)-Output Channel

CHEIN-I CHANG, MEMBER, IEEE, AND LEE D. DAVISSON, FELLOW, IEEE

Abstract—**The Arimoto–Blahut algorithm can be used to compute the capacity of a finite-input finite-output channel. A version of the Arimoto–Blahut algorithm also exists for continuous channels; unfortunately, it involves evaluating integrals over an entire input space and thus is not tractable. Two generalized discrete versions of the Arimoto–Blahut algorithm are presented for this purpose. Instead of calculating integrals, both algorithms require only the computation of a sequence of finite sums. This significantly reduces numerical computational complexity.**

I. Introduction

IT IS WELL-KNOWN that for a given finite-input finite-output channel the most efficient and elegant algorithm for computing the channel capacity is the one developed by Arimoto [1] and Blahut [2]. Since its original formulation it has been widely cited in the literature. The main feature of the Arimoto–Blahut algorithm is iterating probability mass functions to achieve convergence. An Arimoto–Blahut algorithm for a continuous channel is established in [3], but in this case the Arimoto–Blahut algorithm requires the calculation of integrals over the entire input and output spaces which, in general is computationally impractical. We propose two algorithms (to be called Algorithm I and Algorithm II) as replacements for the continuous version of the discrete Arimoto–Blahut algorithm. Both algorithms are iterative and are based on a succession of discretizations of the infinite channel inputs. In essence, Algorithm I is the same algorithm suggested in Davisson and Leon-Garcia [4] which was used to calculate the minimax redundancy for the class of the Lth extensions of binary memoryless sources and also shown to be equivalent to finding the capacity of a particular infinite-input finite-output channel. Unfortunately, the convergence proof of this algorithm was not given even though it has been shown to be empirically convergent in [4] and [5]. Thus one of the major results obtained in this paper concerns proving a convergence theorem for this algorithm (i.e., Algorithm I) for all such channels.

Algorithm II is new and slightly different. Whereas Algorithm I bases its iterative processes on a fixed discrete set of properly chosen channel inputs, Algorithm II simply augments its present input set to accommodate newly generated inputs at every iteration cycle. Basically, Algorithm I relies on the fact that a discrete output channel's capacity can be achieved by using only a discrete set of channel inputs that has the same size as the set of channel outputs (see [6, corollary 3, p. 96]), whereas Algorithm II does not. This results in an important difference: Algorithm II can be generalized to *infinite-output* channels, whereas Algorithm I cannot. However, by invoking a theorem which will be proved later, Algorithm I is still applicable through a sequence of finite approximations.

This paper is organized as follows. In Section II the channel capacity problem is formulated as a minimax problem rather than the double maxima problem treated in [2]. In Section III the case of an infinite-input finite-output channel is considered, and Algorithms I and II are presented for calculating its capacity. Associated convergence theorems are then proven. In Section IV a numerical example is studied, and the performances of Algorithms I and II are compared. It is shown that the two algorithms actually yield almost the same results. To extend the results in Section III to the case of an infinite-input infinite-output channel, an auxiliary approximation theorem is proven in Section V. By virtue of this theorem the algorithms proposed in Section III can be further used to compute the capacity of a continuous channel within any specified degree of accuracy. Finally, in Section VI a brief conclusion concerning the advantages and disadvantages of the two algorithm is given.

II. Minimax Formulation of the Channel Capacity Problem

Consider a discrete-time stationary and memoryless channel characterized by a channel input space X, a channel output space Y, and a channel transition probability density function $\{P(y|x)\}$, i.e., a conditional probability density function that the letter y appears at the channel output when the input letter is x. Then under mild conditions the channel capacity is defined as $C =$

Manuscript received October 23, 1986; revised December 12, 1987. This paper was presented at the IEEE International Symposium on Information Theory, Brighton, England, June 1985.

C.-I Chang was with the Department of Electrical Engineering, University of Maryland, College Park, MD. He is now with the Department of Electrical Engineering, University of Maryland—Baltimore County, Baltimore, MD 21228.

L. D. Davisson is with the Department of Electrical Engineering, University of Maryland, College Park, MD 20742.

IEEE Log Number 8823618.

0018-9448/88/0900-1004$01.00

$\sup_{p(x)\in P} I(X;Y)$, where P is the set of all possible probability density functions of X and $I(X;Y)$ is the average mutual information between X and Y. Also, by definition $I(X;Y)$ is given by

$$I(X;Y)=\int_X\int_Y dx\,dy\,p(x)P(y|x)\log\frac{P(y|x)}{q(y)},$$

$$\text{where } q(y)=\int_X P(y|x)p(x)\,dx$$

We now reformulate the definition of the channel capacity as a minimax problem as follows [6, p. 524].

Let Q be the set of all possible probability density functions $q(y)$ of Y. Then for any fixed input probability density function $p(x)$ and channel transition probability density function $P(y|x)$, it is easy to show that the solution $q^*(y)\in Q$ to minimizing the expression

$$\int_X\int_Y dx\,dy\,p(x)P(y|x)\log\frac{P(y|x)}{q(y)}$$

over Q is given by $\int_X p(x)P(y|x)\,dx$. We define

$$q^*(y)\equiv \arg\left\{\min_{q(y)\in Q}\int_X\int_Y dx\,dy\,p(x)P(y|x)\log\frac{P(y|x)}{q(y)}\right\}.$$

Indeed, by using the inequality $\log x\geq 1-(1/x)$ with equality if and only if $x=1$, the following inequality can be derived:

$$\int_X\int_Y dx\,dy\,p(x)P(y|x)\log\frac{P(y|x)}{q(y)}$$
$$\geq\int_X\int_Y dx\,dy\,p(x)P(y|x)\log\frac{P(y|x)}{\int_X p(x)P(y|x)\,dx}$$

with equality if and only if $q(y)\equiv\int_X P(y|x)p(x)\,dx$. This implies that

$$C=\sup_{p(x)\in P} I(X;Y)$$
$$=\sup_{p(x)\in P}\int_X\int_Y dx\,dy\,p(x)P(y|x)\log\frac{P(y|x)}{\int_X p(x)P(y|x)\,dx}$$
$$=\sup_{p(x)\in P}\min_{q(y)\in Q}\int_X\int_Y dx\,dy\,p(x)P(y|x)\log\frac{P(y|x)}{q(y)}$$
$$=\min_{q(y)\in Q}\sup_{p(x)\in P}\int_X\int_Y dx\,dy\,p(x)P(y|x)\log\frac{P(y|x)}{q(y)}.$$

The last equality holds if Q is compact.

Therefore, one can conclude that finding the channel capacity is equivalent to solving a minimax problem. In the following sections, we consider the channel capacity problem in two separate cases: finite output space in Section III and infinite output space in Section V.

III. An Infinite-Input Finite-Output Channel

In this section we consider a channel that has an infinite input space X and a finite output space, i.e., $Y=\{y_1,\cdots,y_L\}$. The capacity is given by

$$C_L=\min_{q\in Q}\sup_{p\in P}\int_X\sum_{k=1}^{L}P(y_k|x)p(x)\log\left[\frac{P(y_k|x)}{q(y_k)}\right]dx. \tag{3-1}$$

A. An Algorithm for Calculating the Channel Capacity

The following recursive algorithm (based on [6, corollary 3, p. 96]) is used.

Algorithm I:

1) Initialization: Given a finite $J\geq L$ and an error threshold ϵ, choose an arbitrary initial input set $X^1=\{x_1^1,\cdots,x_J^1\}$. Set $n=0$.

2) Set $n=n+1$. Apply the discrete version of the Arimoto–Blahut algorithm to the nth test channel specified by the channel transition matrix $\{P(y_k|x_j^n)\}$ with finite input space X^n and the output space Y, and find

a) the channel capacity C^n;

b) an nth input distribution $p^n(x^n)=[p^n(x_1^n)\cdots p^n(x_J^n)]$ achieving the capacity C^n;

c) the nth optimal output probability vector $q_{p^n}^*(y)=[q_{p^n}^*(y_1)\cdots q_{p^n}^*(y_L)]$ on Y with respect to $p^n(x^n)$ where

$$q_{p^n}^*(y_k)=\Sigma_{j=1}^{J}p^n(x_j^n)P(y_k|x_j^n).$$

3) Define

$$I^n(x;q_{p^n}^*)\equiv\sum_{k=1}^{L}P(y_k|x)\log\left[\frac{P(y_k|x)}{q_{p^n}^*(y_k)}\right],$$

the mutual information for arbitrary input x averaged over the outputs in Y, and thus,

$$C^n=\sum_{j=1}^{J}p^n(x_j^n)I^n(x_j^n;q_{p^n}^*)$$
$$=\sum_{j=1}^{J}\sum_{k=1}^{L}p^n(x_j^n)P(y_k|x_j^n)\log\left[\frac{P(y_k|x_j^n)}{q_{p^n}^*(y_k)}\right].$$

Compute $\sup_{x\in X} I^n(x;q_{p^n}^*)$ and check the error

$$\epsilon_n=\sup_{x\in X} I^n(x;q_{p^n}^*)-C^n. \tag{3-2}$$

4) If the error $\epsilon_n<\epsilon$, then stop and output C^n, p^n, and $q_{p^n}^*$. Otherwise, let $\{x_j^n\}$ be arranged in nonincreasing order according to the probability vector p^n so that

$$p^n(x_i^n)>p^n(x_j^n),\qquad \text{if and only if } i<j,$$

or

$$p^n(x_i^n)=p^n(x_j^n)\quad\text{and}\quad x_i^n<x_j^n,$$
$$\text{if and only if } i<j.$$

Let $X^{*,n}$ be the set of all locally maximizing inputs

$\{x_j^{*,n}\}$ with $I^n(x_j^{*,n}, q_{p^n}^*) \geq C^n$, and define X^{n+1} as follows:

a) if $|X^{*,n}| = J$ then let $X^{n+1} = X^{*,n}$;
b) otherwise, let

$$x_j^{n+1} = \begin{cases} x_j^n, & 1 \leq j \leq J - |X^{*,n}| \\ x_j^{*,n}, & J - |X^{*,n}| + 1 \leq j \leq J \end{cases}$$

where $x_j^n \in X^n$ and $x_j^{*,n} \in X^{*,n}$.

Let $X^n = \{x_j^{n+1}\}$. Go to step 2).

It is worth noting that computing (3-2) in step 3) and finding $X^{*,n}$ in step 4) can be accomplished by using the techniques described in [7], e.g., Fibonacci's method. In addition, we assume that only a finite number of points exists in X that achieve the local maxima of $I^n(x; q_{p^n}^*)$.

B. A Second Algorithm for Calculating the Channel Capacity

In the first algorithm the number of input-output pairs is fixed at J. We now present a second algorithm that allows the number of pairs to vary.

Algorithm II:

1) Initialization: Let J^1 be an arbitrary positive integer and ϵ a prescribed error threshold. Choose an arbitrary initial input set $X^1 = \{x_1^1, \cdots, x_{J^1}^1\}$. Set $n = 0$.

2) Set $n = n + 1$. Apply the discrete version of the Arimoto–Blahut algorithm to the nth test channel $\{P(y_k|x_j^n)\}$ to find the following:

a) the nth channel capacity C^n;
b) an nth input distribution $p^n(x^n) = [p^n(x_1^n) \cdots p^n(x_{J^n}^n)]$ on X^n that achieves the capacity C^n;
c) the nth optimal output probability vector on Y with respect to p^n, $q_{p^n}^*(y) = [q_{p^n}^*(y_1) \cdots q_{p^n}^*(y_L)]$, where

$$q_{p^n}^*(y_k) = \sum_{j=1}^{J^n} p^n(x_j^n) P(y_k|x_j^n).$$

3) Define

$$I^n(x; q_{p^n}^*) \equiv \sum_{k=1}^{L} P(y_k|x) \log \left[\frac{P(y_k|x)}{q_{p^n}^*(y_k)} \right],$$

and then

$$C^n = \sum_{j=1}^{J^n} p^n(x_j^n) I^n(x_j^n; q_{p^n}^*) = \sum_{j=1}^{J^n} \sum_{k=1}^{L} p^n(x_j^n) P(y_k|x_j^n) \log \left[\frac{P(y_k|x_j^n)}{q_{p^n}^*(y_k)} \right].$$

Compute $\sup_{x \in X} I^n(x; q_{p^n}^*)$, and check the error

$$\epsilon_n = \sup_{x \in X} I^n(x; q_{p^n}^*) - C^n.$$

4) If the error $\epsilon_n < \epsilon$, then stop and output C^n, p^n, and $q_{p^n}^*$. Otherwise, delete all zero probability inputs in X^n, relabel the remaining inputs, and denote the resulting input set by $\tilde{X}^n$. Let $X^{*,n}$ be the set of all locally maximizing inputs of $I^n(x; q_{p^n}^*)$ over X, and define

$$x_j^{n+1} = \begin{cases} x_j^n, & 1 \leq j \leq |\tilde{X}^n| \text{ for } x_j^n \in \tilde{X}^n \\ x_j^{*,n}, & 1 + |X^{*,n}| \leq j \leq |X^{*,n}| + |\tilde{X}^n| \equiv J^{n+1} \end{cases}$$

where $x_j^{*,n} \in X^{*,n}$.

Let $X^{n+1} = \{x_j^{*,n}\}$. Go to step 2).

C. Convergence Theorem for Algorithm I

Theorem 1: The sequence $\{C^n\}$ is nondecreasing, i.e., for each n, $C^n \leq C^{n+1}$, with equality if and only if

$$\sup_{x \in X} I^n(x; q_{p^n}^*) = C^n. \tag{3-3}$$

Proof: For any arbitrary probability vector w on X^{n+1}, it is obvious that

$$C^n = \sum_{j=1}^{J} p^n(x_j^n) I^n(x_j^n; q_{p^n}^*) \leq \sum_{j=1}^{J} w(x_j^{n+1}) I^{n+1}(x_j^{n+1}; q_{p^n}^*). \tag{3-4}$$

The inequality holds because of the choices of $\{x_j^{n+1}\}$ in step 4) of Algorithm I.

Recall that

$$I^n(x_j^n; q_{p^n}^*) = \sum_{k=1}^{L} P(y_k|x_j^n) \log \frac{P(y_k|x_j^n)}{q_{p^n}^*(y_k)}$$

where $q_{p^n}^*$ is the nth optimal output probability vector with respect to p^n produced by the discrete Arimoto–Blahut algorithm and $q_{p^n}^*(y_k) = \sum_{j=1}^{J} p^n(x_j^n) P(y_k|x_j^n)$ for $1 \leq k \leq L$. In other words,

$$q_{p^n}^* = \arg \left[\min_q I^n(x_j^n; q_{p^n}^*) \right] = \arg \left[\min_q \sum_{k=1}^{L} P(y_k|x_j^n) \log \left[\frac{P(y_k|x_j^n)}{q(y_k)} \right] \right].$$

In the following we show that $q_{p^n}^*$ is not only an nth optimal output probability vector with respect to p^n on X^n but also an $(n+1)$th optimal output probability vector with respect to some input probability vector $\tilde{p}^{n+1}(x^{n+1})$ on X^{n+1}. Note that $\tilde{p}^{n+1}$ is not necessarily an $(n+1)$th input distribution on X^{n+1} which achieves the channel capacity C^{n+1}. More precisely, we wish to find an input probability vector $\tilde{p}^{n+1}(x^{n+1})$ on X^{n+1} so that for every $1 \leq k \leq L$,

$$q_{p^n}^*(y_k) = \sum_{j=1}^{J} p^n(x_j^n) P(y_k|x_j^n) = \sum_{j=1}^{J} \tilde{p}^{n+1}(x_j^{n+1}) P(y_k|x_j^{n+1}) \equiv q_{\tilde{p}^{n+1}}^*(y_k). \tag{3-5}$$

Notice that $q^*_{\tilde{p}^{n+1}}$ defined by (3-5) is an optimum (or Bayes) L-dimensional probability vector on Y with respect to the probability vector $\tilde{p}^{n+1}$ on X^{n+1}. From (3-5) we obtain a set of L simultaneous linear algebraic equations with J unknowns. Hence since $J \geq L$, $\tilde{p}^{n+1}$ exists and is a probability vector on X^{n+1}. Note that this observation offers an alternative view of [5, corollary 3, p. 96], that is that the number of inputs to the channel assigned nonzero probability by a probability vector w which achieves the channel capacity is no more than $|Y| = L$. Equation (3-5) shows that the number of channel outputs L determines the size of the initial input set for Algorithm I.

We now go back to inequality (3-4) and substitute the probability vector $\tilde{p}^{n+1}$ for the probability vector w appearing in the right side of inequality (3-4). Note that the inequality (3-4) is true for any arbitrary probability vector defined on X^{n+1}. Therefore, inequality (3-4) becomes

$$\begin{aligned} C^n &\leq \sum_{j=1}^{J} \tilde{p}^{n+1}\left(x_j^{n+1}\right) I^n\left(x_j^{n+1}; q^*_{p^n}\right) \\ &= \sum_{j=1}^{J} \tilde{p}^{n+1}\left(x_j^{n+1}\right) I^n\left(x_j^{n+1}; q^*_{\tilde{p}^{n+1}}\right) \\ &= I\left(X^{n+1}; Y\right) \qquad \text{(with the input distribution } \tilde{p}^{n+1}\text{)} \\ &\leq \sup_{p(x)} I\left(X^{n+1}; Y\right) \\ &= \sum_{j=1}^{J} p^{n+1}\left(x_j^{n+1}\right) I^{n+1}\left(x_j^{n+1}; q_{p^{n+1}}\right) = C^{n+1}. \end{aligned} \tag{3-6}$$

The equality follows from (3-5) and the last inequality follows from construction of p^{n+1} in the algorithm which is an input distribution on X^{n+1} achieving the capacity of the $(n+1)$th test channel, C^{n+1}. Thus C^n is nondecreasing in n.

Proof of equality — 1) Necessity: Assume that $C^n = C^{n+1}$. Then from inequality (3-6), we have

$$C^n = \sum_{j=1}^{J} \tilde{p}^{n+1}\left(x_j^{n+1}\right) I^{n+1}\left(x_j^{n+1}; q^*_{\tilde{p}^{n+1}}\right) = C^{n+1}.$$

This implies that $\tilde{p}^{n+1}$ is also an $(n+1)$th input probability vector defined on X^{n+1} achieving the $(n+1)$th channel capacity C^{n+1}. Therefore,

$$\begin{aligned} C^{n+1} &= \max_{x \in X^{n+1}} I^{n+1}\left(x; q^*_{p^{n+1}}\right) \\ &= \max_{x \in X^{n+1}} I^{n+1}\left(x; q^*_{\tilde{p}^{n+1}}\right) \\ &\geq \sup_{x \in X} I^n\left(x; q^*_{p^n}\right) \qquad \left(\text{since by (3-5) } q^*_{p^n} = q^*_{\tilde{p}^{n+1}}\right) \\ &\geq C^n = C^{n+1}. \end{aligned}$$

Consequently, $C^n = \sup_{x \in X} I^n\left(x; q^*_{p^n}\right)$.

2) Sufficiency: Assume $C^n = \sup_{x \in X} I^n\left(x; q^*_{p^n}\right)$. Then

$$\begin{aligned} C^{n+1} &= \max_{x \in X^{n+1}} I^n\left(x; q^*_{p^{n+1}}\right) \\ &\leq \max_{x \in X^{n+1}} I^n\left(x; q^*_{p^n}\right) \qquad \left(\text{since } q^*_{p^{n+1}} \text{ is the } (n+1)\text{th minimax output probability vector on } X^{n+1}\right) \\ &\leq \sup_{x \in X} I^n\left(x; q^*_{p^n}\right) \qquad \left(\text{since } X^{n+1} \subset X\right) \\ &= C^n. \end{aligned}$$

However, as shown in the first part of the theorem, C^n is nondecreasing. Therefore, $C^{n+1} \geq C^n$ implies that $C^{n+1} = C^n$. The proof is thus complete.

An immediate consequence of Theorem 1 is that if Algorithm I terminates at some finite step m with equality (3-3), then C^m is equal to the channel capacity C_L. Indeed, from (3-3) in Theorem 1,

$$C^m = \sup_{x \in X} I^m\left(x; q^*_{p^m}\right).$$

On the other hand, since from (3-1) C_L is the minimax quantity, for any probability vector $p(x) \in P$ let $q^*_p(y_k) = \int_X p(x) P(y_k|x)\, dx$; then

$$C_L \leq \sup_{x \in X} \sum_{k=1}^{L} P(y_k|x) \log\left[\frac{P(y_k|x)}{q^*_p(y_k)}\right] = \sup_{x \in X} I(x; q^*_p)$$

(see [5, p. 524]). In particular, choose $p(x) = p^m(x)$. Then,

$$C_L \leq \sup_{x \in X} I^m\left(x; q^*_{p^m}\right) = C^m.$$

However, C^n is nondecreasing according to Theorem 1 and is also bounded from above by C_L because of (3-1). This yields that $C_L = C^m$.

The same treatment can be applied to the case (i.e., (3-2))

$$\sup_{x \in X} I^{m(\epsilon)}\left(x; q^*_{p^{m(\epsilon)}}\right) - C^{m(\epsilon)} < \epsilon$$

at some finite step $m(\epsilon)$ depending on ϵ when an error ϵ is allowed. In this situation, $C_L < C^{m(\epsilon)} + \epsilon < C_L + \epsilon$. Accordingly, as $\epsilon \to 0$, $C^{m(\epsilon)} \to C_L$.

D. Convergence Theorem for Algorithm II

In this section, we prove the same theorem (Theorem 1 in Section III-C) for Algorithm II. Generally speaking, the same convergence proof is still applicable for Algorithm II with the proper modifications. Nevertheless, we intend to give a simple alternative proof for the first part of Theorem 1 which essentially constructs the probability vector $\tilde{p}^{n+1}$ on X^{n+1} without solving a set of linear algebraic equations subject to (3-5) so that this proof is valid for the case of an infinite-output channel which will be discussed in Section V.

Proof of Theorem 1 for Algorithm II: Since $X^n \subset X^{n+1}$ and C^{n+1} is the $(n+1)$th channel capacity specified by the channel with input set X^{n+1}, output set Y, and the channel transition matrix $\{P(y_k|x_j^{n+1})\}$,

$$C^{n+1} = \sup_{\hat{p}^{n+1} \text{ on } X^{n+1}} \sum_{j=1}^{J^{n+1}} \hat{p}^{n+1}\left(x_j^{n+1}\right) I^{n+1}\left(x_j^{n+1}; q^*_{\hat{p}^{n+1}}\right)$$

where

$$I^{n+1}\left(x_j^{n+1}; q^*_{\hat{p}^{n+1}}\right) = \sum_{k=1}^{L} P\left(y_k|x_j^{n+1}\right) \log\left[\frac{P\left(y_k|x_j^{n+1}\right)}{q^*_{\hat{p}^{n+1}}(y_k)}\right]$$

and

$$q^*_{\hat{p}^{n+1}}(y_k) = \sum_{j=1}^{J^{n+1}} \hat{p}^{n+1}\left(x_j^{n-1}\right) P\left(y_k|x_j^{n+1}\right).$$

Without solving (3-5) we can directly define $\tilde{p}^{n+1}$ by means of p^n as follows:

$$\tilde{p}^{n+1}\left(x_j^{n+1}\right) = \begin{cases} p^n\left(x_j^n\right) & \text{for } x_j^{n+1} \in \tilde{X}^n \\ 0, & \text{for } x_j^{n+1} \in X^{*,n}. \end{cases}$$

Then we have

$$\begin{aligned} q^*_{\tilde{p}^{n+1}}(y_k) &= \sum_{j=1}^{J^{n+1}} \tilde{p}^{n+1}\left(x_j^{n+1}\right) P\left(y_k|x_j^{n+1}\right) \\ &= \sum_{j=1}^{J^n} p^n\left(x_j^n\right) P\left(y_k|x_j^n\right) = q^*_{p^n}(y_k) \end{aligned}$$

and therefore,

$$\begin{aligned} C^{n+1} &\geq \sum_{j=1}^{J^{n+1}} \tilde{p}^{n+1}\left(x_j^{n+1}\right) I^{n+1}\left(x_j^{n+1}; q^*_{\tilde{p}^{n+1}}\right) \\ &= \sum_{x_j^{n+1} \in \tilde{X}^n} p^n\left(x_j^n\right) I^n\left(x_j^n q^*_{p^n}\right) \\ &\quad + \sum_{x_j^{n+1} \in X^{*,n}} 0 \times I^n\left(x_j^{n+1}; q^*_{p^n}\right) \\ &= \sum_{x_j^{n+1} \in \tilde{X}^n} p^n\left(x_j^n\right) I^n\left(x_j^n; q^*_{p^n}\right) \\ &= \sum_{j=1}^{J^n} p^n\left(x_j^n\right) I^n\left(x_j^n; q^*_{p^n}\right) = C^n. \end{aligned}$$

This shows that $\{C^n\}$ is nondecreasing in n. The rest of the proof (i.e., equality) can be carried out by using exactly the same argument given in Theorem 1.

Remarks: 1) It is apparent that if C_L were not bounded, then from [6, Prob. 4-17 (i), p. 524] and from Theorem 1, Algorithms I and II would continue indefinitely unless we set in advance the number of iterations needed.

2) A general theory of the two foregoing algorithms was developed in a mathematical setting for a general decision problem in [8]; the associated convergence theorems were also proven in detail. However, the convergence proofs of general versions of Algorithms I and II are by no means as simple as in the proof of Theorem 1. They involve initiating a new property that bridges the gap between two successive input sets X^n and X^{n+1} and plays a key role in the foregoing proofs. In addition, the discrete version of the Arimoto–Blahut algorithm is not applicable to general decision problems. It is replaced by the Nelson algorithm [9] which can be used for a general purpose to generate a least favorable distribution and a corresponding Bayes estimate. More details can be found in [8].

IV. Numerical Results

Although Algorithm I was implemented in [4] and [5], Algorithm II is new. In this section, to compare the relative performance of the algorithms, we consider a generalized binarylike memoryless channel which is specified by the input space $X = [0,1]$, the output space $Y_{L+1} = \{0,1,2,\cdots,L\}$, and the channel transition probabilities $\{P(k|x)\}_{x \in , k \in Y_{L+1}}$ given by

$$P(k|x) = \binom{L}{k} x^k (1-x)^{L-k}$$

where

$$\binom{L}{k} = \frac{L!}{k!(L-k)!}.$$

Then the channel capacity is defined by

$$C_{L+1} = \max_{p(x)} \left\{ \sum_{k=0}^{L} \int_X p(x) P(k|x) \log \frac{P(k|x)}{q^*_p(k)} \right\}$$

where

$$q^*_p(k) = \int_X p(x) P(k|x)\, dx.$$

As shown in [4], for each $L+1$ (i.e., the size of the channel output space) C_{L+1} is bounded and $X^{*,n}$ is finite for all n. In Fig. 1 the numerical results obtained based on Algorithm I and Algorithm II are seen to be essentially identical. Since Algorithm I fixes the channel input size and Algorithm II does not, Algorithm II needs less iterations than Algorithm I (as one would expect). However, this advantage is gained at the expense of increased channel input size (and hence computational requirements) after completing each iteration cycle. This fact can be observed from Table I. Actually, this example further demonstrates the efficiency of Algorithms I and II. In Table I, for $L \leq 49$, Algorithm I needs no more than five

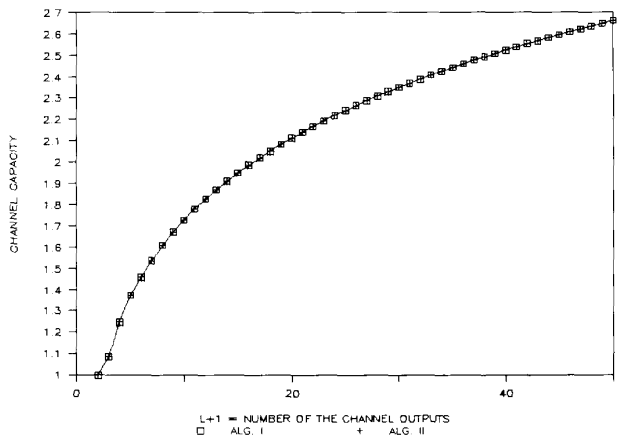

Fig. 1. Channel capacity versus $L+1$ = number of channel outputs for Algorithms I and II.

TABLE I
A COMPARISON OF THE PERFORMANCE OF ALGORITHMS I AND II
(ERROR THRESHOLD $\epsilon = 6 \times 10^{-4}$)

	Algorithm I		Algorithm II		
$L+1$	Channel Capacity	Iterations Required	Channel Capacity	Iterations Required	Number of Inputs
2	1.00000000	1	1.00000000	1	2
3	1.08746283	1	1.08746283	1	3
4	1.24790640	2	1.24790661	2	5
5	1.37227190	1	1.37227190	1	5
6	1.45797721	1	1.45797721	1	6
7	1.53587065	3	1.53586957	3	11
8	1.60795459	3	1.60795116	3	12
9	1.67149199	3	1.67149204	3	13
10	1.72682231	3	1.72682026	3	15
11	1.77802079	3	1.77802110	3	16
12	1.82584637	3	1.82584434	3	15
13	1.87028438	3	1.87028312	3	21
14	1.91139536	3	1.91138040	3	13
15	1.94964308	2	1.94963458	2	18
16	1.98588584	3	1.98587671	3	27
17	2.02021339	3	2.02022505	3	22
18	2.05280604	3	2.05280645	3	21
19	2.08372756	2	2.08372758	2	23
20	2.11305547	2	2.11307271	2	24
21	2.14116808	2	2.14116790	2	23
22	2.16810322	2	2.16810326	2	27
23	2.19397183	2	2.19397166	2	26
24	2.21881660	2	2.21881651	2	28
25	2.24269341	2	2.24269314	2	28
26	2.26568557	2	2.26568556	2	31
27	2.28791704	2	2.28791477	2	32
28	2.30940827	2	2.30940754	2	34
29	2.33020603	2	2.33020748	2	35
30	2.35035823	2	2.35035611	2	35
31	2.36988413	2	2.36988209	2	37
32	2.38882862	2	2.38882774	2	38
33	2.40723350	2	2.40723357	2	39
34	2.42514996	3	2.42514362	2	42
35	2.44258608	3	2.44258050	2	40
36	2.45956822	3	2.45955957	2	43
37	2.47611668	3	2.47610383	2	45
38	2.49225175	3	2.49223741	2	46
39	2.50798912	3	2.50797873	2	47
40	2.52335028	4	2.52334873	2	49
41	2.53837809	4	2.53836604	2	52
42	2.55306776	3	2.55304872	2	53
43	2.56743387	3	2.56741719	2	55
44	2.58149611	3	2.58147673	2	54
45	2.59525659	3	2.59524335	2	57
46	2.60873619	3	2.60872360	2	58
47	2.62194383	3	2.62193244	2	58
48	2.63488601	3	2.63488110	2	60
49	2.64759964	4	2.64758040	2	61
50	2.66006762	5	2.66004004	2	64

iterations, while Algorithm II requires at most three iterations. It is believed that Algorithms I and II have potential applications to a variety of areas other than channel capacity problems (see [8]). More details on this numerical study can be found in [10].

V. AN INFINITE-INPUT INFINITE-OUTPUT CHANNEL

In this section we further assume that X and Y are both compact and infinite. Then the capacity is given by

$$C_\infty = \min_{q \in Q} \sup_{p \in \boldsymbol{P}} \int_X \int_Y p(x) P(y|x) \log \left[\frac{P(y|x)}{q(y)} \right] dy\, dx \tag{4-1}$$

where $\boldsymbol{P}$ is the set of all possible probability density functions on X and Q is the set of all possible continuous probability density functions on Y. It is obvious that Algorithm I is not directly applicable because in (3-4) we solve for a probability vector $\tilde{p}^{n+1}$ with L unknowns $\tilde{p}^{n+1}(x_j^{n+1})$ for $j = 1, \cdots, J$ and $J \geq L$, where L is the cardinality of Y which is now infinite.

To overcome the difficulty, we prove the following theorem which enables us to produce a sequence of conditional probability *mass* functions defined on a finite subset in Y given $x \in X$ such that (4-1) can be approximated by (3-1) to any desirable accuracy. It is also important to note that since the proof of Theorem 1 given for Algorithm II in Section III-D does not rely on the choice of J and does not require the solution of (3-5), Algorithm II still works for this case. This is intuitively reasonable because unlike Algorithm I which presets the size of the input set, Algorithm II simply starts with any arbitrary set of inputs, finds all inputs that locally maximize $I^n(x; q_{p^n}^*)$, and adds them to the present input set after completing an iteration. Thus it always keeps updated information by increasing its present iterated input set to store all available past data. As a consequence, Algorithm II will converge.

Theorem 2: For a given continuous channel transition probability density function $P(y|x)$ with X, Y compact, a *finite* set F in Y and a sequence of polynomials $\{P_n(y_k|x)\}$ exist such that $\{P_n(y_k|x)\}$ approximates $P(y|x)$ uniformly on X for some $y_k \in F$. Moreover, by means of this sequence $\{P_n(y_k|x)\}$ an associated sequence of conditional probability *mass* functions defined on F can be constructed so that the problem of an infinite output space can be reduced to that of a finite output space discussed in Section III; thus Algorithm I is still applicable.

Proof: The proof consists of the following four steps.

1) Since X, Y are compact and $P(y|x)$ is continuous on $X \times Y$, by the Stone–Weierstrauss approximation a sequence of polynomials $P_n(y|x)$ exists which uniformly approximates $P(y|x)$ to any degree of accuracy. The polynomials $P_n(y|x)$ can be made nonnegative by setting the polynomial approximation to zero whenever negative values occur. Namely, for any arbitrarily small positive ϵ, a positive integer N exists such that

$$|P(y|x) - P_n(y|x)| < \frac{\epsilon}{2}, \qquad \text{for all } x \in X,\ y \in Y. \tag{4-2}$$

2) Let

$$K_y = \left\{ z \in Y \middle| |P(z|x) - P(y|x)| < \frac{\epsilon}{2} \text{ for all } x \in X \right\}.$$

Then $Y = \cup_{y \in K} K_y$ and K_y is open because $P(y|x)$ is continuous on Y. Since Y is compact, then by the Heine–Borel theorem a *finite* set $F = \{y_k | y_k \in Y\}$ exists such that $Y = \cup_{y_k \in F} K_{y_k}$, where

$$K_{y_k} \equiv \left\{ y \in Y \middle| |P(y|x) - P(y_k|x)| < \frac{\epsilon}{2} \text{ for all } x \in X \right\} \tag{4-3}$$

i.e., for any y in Y a y_k exists such that $y \in K_{y_k}$. Then combining (4-2) in step 1) with (4-3), we have

$$0 < P(y|x) - \epsilon < P_n(y_k|x) < P(y|x) + \epsilon$$

for all $n \geq N$ and all $y \in K_{y_k}$.

3) In general, the polynomials $P_n(y_k|x)$ constructed in step 2) need not be probability mass functions. However, based on each polynomial $P_n(y_k|x)$, we can define a new conditional probability *mass* function $F_{P_n}(y_k|x)$ on F given that x is in X as follows. Let

$$h(x) = \sum_{y_k \in F} P_n(y_k|x)\mu(K_{y_k}), \qquad \text{where } \mu(K_{y_k}) = \int_{K_{y_k}} dy.$$

Then $F_{P_n}(y_k|x)$ is defined by the following:

$$f_{P_n}(y_k|x) \equiv \frac{P_n(Y_k|x)}{h(x)}$$

and

$$F_{P_n}(y_k|x) \equiv f_{P_n}(y_k|x)\mu(K_{y_k}).$$

Obviously, $\sum_{y_k \in F} F_{P_n}(y_k|x) = 1$ and $F_{P_n}(y_k|x)$ is a conditional probability *mass* function on F given x.

4) By invoking the conditional probability *mass* functions $F_{P_n}(y_k|x)$ obtained in step 3), the problem of calculating an infinite-output channel capacity is reduced to one for a finite-output channel such that its capacity can be computed numerically by Algorithm I, as discussed in Section III.

The procedures for carrying out the foregoing steps are straightforward; we refer all technical details to [8].

Although Theorem 2 was proven under the assumption that the output space Y is compact, it can be also extended to the case where Y is not compact, in particular, when Y is countably infinite. This can be easily justified by truncating the tail of Y and replacing it with a single probability for the truncated tail. An increasing nested sequence of such truncations will converge to the capacity.

In [11], an efficient universal noiseless coding procedure was constructed based on a key lemma (i.e., [11, lemma 1]) where the common source output space was assumed to be finite. As an application of Theorem 2, the lemma can indeed be extended to a broader class of stationary parameterized sources where the common source alphabet space is infinite and the parameter space is compact.

VI. Conclusion

We proposed two algorithms for calculating the capacity of an infinite-input finite (infinite)-output channel. Although both algorithms consist of iterative procedures based on a sequence of finite inputs, there are major differences that characterize their respective advantages.

1) To initialize an input set, Algorithm I presets its size and keeps the same size until termination. In contrast, Algorithm II chooses an arbitrary input set to initialize the algorithm and then keeps track of all desirable inputs by varying the sizes of input sets during its execution.

2) To form the $(n+1)$th input set, Algorithm I replaces a subset of inputs in the nth input set with low probabilities by a set of newly found locally maximizing inputs of $I^n(x; q_{p^n}^*)$ whose average mutual information in Y is no less than the nth test channel capacity; Algorithm II simply collects all possible locally maximizing inputs of $I^n(x; q_{p^n}^*)$ and adds them to the nth input set.

The advantage of using Algorithm I is that throughout its iterations only a finite fixed memory is used, and we do not have to worry about buffer size limitations. This is particularly important for computer implementations when $I^n(x; q_{p^n}^*)$ has a large number of locally maximizing inputs. On the other hand, the advantage of applying Algorithm II is that we need not determine the size of an input set for initialization beforehand but simply select any arbitrary inputs and then store all necessary inputs by adjusting the buffer size after completing an iteration. This suffers from the problem that the buffer size may increase rapidly. Nevertheless, Algorithm II is preferable whenever there is difficulty in determining the size of the initial inputs for Algorithm I.

References

[1] S. Arimoto, "An algorithm for computing the capacity of arbitrary discrete memoryless channels," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 14–20, Jan. 1972.

[2] R. E. Blahut, "Computation of channel capacity and rate-distortion function," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 460–473, July 1972.

[3] ____, "A hypothesis testing approach to information theory," Ph.D. dissertation, Cornell Univ., Ithaca, NY, 1972.

[4] L. D. Davisson and A. Leon-Garcia, "A source matching approach to finding minimax codes," *IEEE Trans. Inform. Theory*, vol. IT-26, no. 2, pp. 166–174, Mar. 1980.

[5] D. H. Lee, "On the source matching approach for Markov sources," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 754–755, Sept. 1983.

[6] R. G. Gallager, *Information and Reliable Communication*. New York: Wiley, 1968.

[7] J. L. Kuter and J. H. Mize, *Optimization Techniques with FORTRAN*. New York: McGraw-Hill, 1973.

[8] C.-I Chang, "A generalized minimax approach to statistical decision problems with applications to information theory," Ph.D. dissertation, Univ. of Maryland, Baltimore, Aug. 1986.

[9] W. Nelson, "Minimax solution of statistical decision problems by iteration," *Ann. Math. Statist.*, vol. 37, pp. 1643–1657, 1966.

[10] S. C. Fan, "A numerical study on calculating the capacity of a continuous-input discrete-output channel," M.S. thesis, Univ. of Maryland—Baltimore County, in preparation.

[11] L. D. Davisson, R. J. McEliece, M. B. Pursley, and M. S. Wallace, "Efficient universal noiseless source codes," *IEEE Trans. Inform. Theory*, vol. IT-27, no. 3, pp. 269–279, May 1981.