Annals of Operations Research 134, 201–214, 2005

On the Convergence of the Cross-Entropy Method

L. MARGOLIN leonid1971@bezeqint.net
Faculty of Industrial Engineering and Management, Technion, Haifa, Israel

Received August 2003; Revised November 2004; Accepted January 2004

Abstract. The cross-entropy method is a relatively new method for combinatorial optimization. The idea of this method came from the simulation field and then was successfully applied to different combinatorial optimization problems. The method consists of an iterative stochastic procedure that makes use of the importance sampling technique. In this paper we prove the asymptotical convergence of some modifications of the cross-entropy method.

Keywords: combinatorial optimization, convergence, cross-entropy

The *cross-entropy* (*CE*) method was developed in Rubinstein (1999) for solving both continuous multi-extremal and various discrete combinatorial optimization problems. The method was successfully examined on a wide range of problems such as the traveling salesman problem (Rubinstein, 1999), the bipartition problem (Rubinstein, 1999), the maximal cut problem (Rubinstein, 2002), the image matching (Dubin, 2002), the image segmentation (Dubin, 2002), etc. This method is an iterative stochastic procedure making use of the importance sampling technique. The stochastic mechanism changes from iteration to iteration according to the Kullback-Leibler cross-entropy approach.

While most of the stochastic algorithms for combinatorial optimization are based on local search (they employ local neighborhood structures), the cross-entropy method is a global random search procedure.

The CE method arises from the rare-event estimation framework (see Rubinstein, 1997) and uses similar techniques. The main idea of the CE method can be stated as follows.

First, we randomize our deterministic problem (by defining a probability distribution on the set of the all possible solutions). Next, we define the following event: "the optimal solution appears within sample of certain size (which is very small in comparison with the number of the all feasible solutions)". For the vast majority of the problems this event is rare, and to estimate its probability is not trivial. For this purpose we make adaptive changes of the probability distribution according to the Kullback-Leibler cross-entropy approach. This procedure significantly increases the chances of the optimal solution (or an almost optimal) to appear within our sample.

In this paper we prove the asymptotical convergence for some modifications of the method. The idea and technique of the proof are similar to the ones from Gutjahr

(2002), where the convergence of two ant algorithms is proved. More precisely, in this paper we derive two cross-entropy algorithms, which are very close to the ant algorithms considered by Gutjahr. Due to this similarity, Gutjahr's proof can be readily modified and adopted to our case. For an alternative convergence prove of CE (see Homem-de-Mello and Rubinstein, 2002).

The rest of the work is organized as follows. In the second section the general cross-entropy method is described. In the third section two modifications of the CE method are introduced and their convergence is proved.

1. The general cross-entropy method

As mentioned above, the initial version of the CE method was presented in Rubinstein (1999). Then, in order to speed up the method and/or improve convergence, several additional modifications were introduced (see Rubinstein, 2001a, 2001b; Margolin, 2002; Rubinstein and Kroese, 2004): for instance, the conservative (smoothed) modification, the CE method with lower bound on probabilities, the CE method with varying sample size, the CE method with memory, etc. In this section, which is based on (Margolin, 2002), we describe the general CE method. For easier understanding, we illustrate the terms and technique with an emphasis on the well-known traveling salesman problem (TSP).

We consider the following combinatorial optimization problem:

$$\min_{\boldsymbol{x}\in\mathcal{X}} L(\boldsymbol{x}), \tag{1}$$

where $\mathcal{X} \subset R^n$ is a finite set, $L(\cdot)$ is a real function from $\mathcal{X}$ to R.

Assume that $\boldsymbol{x}^* = (\boldsymbol{x}_1^*, \ldots, \boldsymbol{x}_n^*)$ is the unique optimal solution of (1).

Additionally, assume that each feasible solution of the problem can be defined by the array $\boldsymbol{X}$ with components X_I, where I belongs to some set of indices $\mathcal{I}$ (that is, $|\mathcal{I}|$ numbers define some feasible solution). Note that $\mathcal{I}$ can be one-dimensional, two-dimensional, etc.

The general cross-entropy method consists of three main steps:

1. Choosing a probability family. Initializing of the distribution parameters and the method parameters. Generating feasible solutions according to the chosen distribution.
2. Updating the distribution parameters, based on the Kullback-Leibler cross-entropy.
3. Checking the stopping rule. Updating the method parameters in the case the stopping rule fails.

At the first step we should choose a "suitable" representation. Finding such "suitable" representation, that is well fitted to the structure of the problem, is of great importance. However, there is no general rule to do so, and our decision is mainly based on intuition.

For example, consider the TSP with n cities on a fully connected graph. As mentioned above, each feasible solution is assumed to be described by the array $\boldsymbol{X}$ with

components X_I, where I belongs to some set of indices $\mathcal{I}$. We would like to give two different representations for this problem.

For instance, we can define feasible solutions using the two-dimensional array $\boldsymbol{X}$ with the following components:

$$X_{(r,s)} = \begin{cases} 1, & \text{the transition from } r \text{ to } s \text{ occurred,} \\ 0, & \text{otherwise.} \end{cases}$$

Obviously, $X_{(r,r)} \equiv 0$. In this case $\mathcal{I} = \{(r,s) \in \{1,\dots,n\} \times \{1,\dots,n\} : r \neq s\}$.

With this representation we can generate feasible solutions randomly, using the following algorithm.

Algorithm 1.1. Solution Generation for the TSP (for the First Representation):

1. Define the probability array $\boldsymbol{P}$:

$$\boldsymbol{P} = \begin{bmatrix} 0 & p_{(1,2)} & \dots & p_{(1,n)} \\ p_{(2,1)} & 0 & \dots & p_{(2,n)} \\ \vdots & \vdots & \vdots & \vdots \\ p_{(n,1)} & p_{(n,2)} & \dots & 0 \end{bmatrix} \tag{2}$$

where the probability $p_{(r,s)}$ corresponds to the transition from the node r to the node s. Assume that for $r \neq s$: $p_{(r,s)} \neq 0$.

2. Set counter $= 1$, $\boldsymbol{U} = \{1\}$, $r = 1$.

3. Generate a next node S from the following distribution:

$$\text{Prob}\{S = s\} = p_{(r,s)}, \quad \forall s \in \{1,\dots,n\}$$

Assume, that S received value s^*. Define

$$X_{(r,s)} = \begin{cases} 1, & s = s^*, \\ 0, & s \neq s^*. \end{cases}$$

4. Set $\boldsymbol{U} = \boldsymbol{U} \cup \{s^*\}$, counter $=$ counter $+ 1$, $r = s^*$.

5. If (counter $< n - 1$)

Update the r-th row of the array:
nullify the probabilities $p_{(r,s)}$ corresponding to the already visited nodes s;
re-normalize the remaining in this row probabilities.
Go to Step 3.

else

Go to Step 6.

end If

6. Set $\{s^*\} = \boldsymbol{V} \setminus \boldsymbol{U}$ (at this step $\boldsymbol{U}$ contains exactly $n-1$ nodes), $X_{(r,s^*)} = 1$.

Remark 1.1. To accelerate solution generation we can use so-called alias method (see Walker, 1977), which is a very effective method for generation random numbers from an arbitrary n-point distribution. More detailed information on using the alias method for the CE algorithms can be found in Margolin (2002).

An alternative representation can be stated as follows. Any feasible solution can be defined by the array $\boldsymbol{X}$ with the following components:

$$X_{(r,s)} = \begin{cases} 1, & \text{the node } r \text{ is arranged to the place } s, \\ 0, & \text{otherwise.} \end{cases}$$

In this case $\mathcal{I} = \{(r, s) \in \{1, \ldots, n\} \times \{1, \ldots, n\}\}$.

The solution generation algorithm corresponding to this representation is similar to the one above (with some minor changes) and can be found in Margolin (2002).

Usually for a specific problem more than one representation can be found. Note that the choice of representation is of great importance for the method convergence.

To proceed with the general CE method, recall that we generate solutions that can be defined by the random array $\boldsymbol{X}$ with components X_I, where I belongs to some set of indices $\mathcal{I}$. For each $I \in \mathcal{I}$ assume that X_I can take values from the following set: $\{a_1(I), \ldots, a_{R(I)}(I)\}$.

Let N_t be the number of the solutions to be considered in the t-th iteration. Generally, N_t may vary from iteration to iteration. Assume, that K_t of the above solutions are the best solutions obtained in the all previous iterations, and the remaining $N_t - K_t$ solutions are the solutions generated in the t-th iteration (clearly, $K_t \in \{0, \ldots, N_t - 1\}$). We will call K_t the "memory size" of the method at the iteration t. Let $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{N_t}$ denote solutions participating in the t-th iteration (for simplicity we drop the index t).

Let Λ_t be the set of the all solutions generated during the iterations $1, \ldots, t$.

Let $p_{t,I}(k)$ be the current probability (after the first t iterations) corresponding to the event $\{X_I = a_k(I)\}$, where $I \in \mathcal{I}$, $k \in \{1, \ldots, R(I)\}$.

Let $\phi(L(\boldsymbol{X}), \Lambda_t, \boldsymbol{\nu}_t)$ be a real positive function, non-increasing in the first argument, where $\boldsymbol{\nu}_t = \boldsymbol{\nu}(\Lambda_t)$ is a vector of *parameters of the function* $\phi(\cdot)$ that may be dependent on the previous history (on Λ_t).

For all k satisfying $p_{t,I}(k) > 0$ define the cross-entropy component:

$$C_{t+1,I}(k) = \frac{\sum_{j:X_{j,I}=a_k(I)} \phi(L(\boldsymbol{X}_j), \Lambda_t, \boldsymbol{\nu}_t)}{\sum_j \phi(L(\boldsymbol{X}_j), \Lambda_t, \boldsymbol{\nu}_t)}, \tag{3}$$

where $X_{j,I}$ denotes the I-th component of $\boldsymbol{X}_j$.

It should be mentioned that the above formula is based on the Kullback-Leibler cross-entropy approach (see Rubinstein, 2001a, 2001b; Margolin, 2002; Rubinstein and

Kroese, 2004). Derivation of the formula using the cross-entropy approach is out of scope of this paper. Instead, we will give some intuitive explanation. The value $\phi(L(\boldsymbol{X}_j), \Lambda_t, \boldsymbol{\nu}_t)$ can be viewed as some quality coefficient of the solution $\boldsymbol{X}_j$. Then

$$\sum_j \phi(L(\boldsymbol{X}_j), \Lambda_t, \boldsymbol{\nu}_t)$$

is the sum of the quality coefficients of the all solutions participating in the iteration t. Similarly,

$$\sum_{j:X_{j,I}=a_k(I)} \phi(L(\boldsymbol{X}_j), \Lambda_t, \boldsymbol{\nu}_t)$$

is the sum of the quality coefficients of those solutions where the component I equals to the k-th possible value. Therefore, $C_{t+1,I}(k)$ can be viewed as *the relative contribution of the such solutions to the sum of the quality coefficients of the all solutions*. It can be considered as some measure of suitability of the k-th possible value of X_I.

Remark 1.2. When dealing with a maximization problem, $\phi(L(\boldsymbol{X}), \Lambda_t, \boldsymbol{\nu}_t)$ must be a real positive function, non-decreasing in the first argument.

The general rule for the probability update can be stated as follows. For every k satisfying $p_{t,I}(k) > 0$:

$$p_{t+1,I}(k) = F(C_{t+1,I}(k), p_{0,I}(k), p_{1,I}(k), \ldots, p_{t,I}(k), \boldsymbol{\mu}_t), \tag{4}$$

where F is a real function taking values from $[0, 1]$ and non-decreasing in the first argument, $\boldsymbol{\mu}_t = \boldsymbol{\mu}(\Lambda_t)$ is a vector of *parameters of the method* that may be dependent on the previous history. Since F is non-decreasing in the first argument, the formula (4) can be considered as some measure of suitability as well.

For example, the following probability updating rule is widely used in the CE literature.

Example 1.1 (Conservative (Smoothed) Modification of the CE Method)**.** In this modification the probabilities are updated according to the following rule:

$$p_{t+1,I}(k) = \alpha_t C_{t+1,I}(k) + (1 - \alpha_t) p_{t,I}(k),$$

where $\alpha_t \in (0, 1]$ is the weight of the cross-entropy component. Obviously, for this modification $\boldsymbol{\mu}_t = (\alpha_t)$. Generally, α_t can be dependent on the previous history.

Below we present the general cross-entropy algorithm. Before implementation of the algorithm a suitable representation of the problem should be chosen. As before, we assume that $\mathcal{I}$ denotes a set of indices of the random array in this representation. Let

$I \in \mathcal{I}$, and assume that the I-th component of the array can take values from the following set: $\{a_1(I), \ldots, a_{R(I)}(I)\}$.

Algorithm 1.2. General CE Algorithm:

Step 1

(a) Initialize $p_{0,I}(k)$ $(\forall I \in \mathcal{I},\ \forall k \in \{1, \ldots, R(I)\})$.

(b) Set $t = 0,\ \Lambda_{-1} = \emptyset$.

Step 2

(a) Let $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{K_t}$ be the best solutions obtained during the previous iterations. Generate solutions $\boldsymbol{X}_{K_t+1}, \ldots, \boldsymbol{X}_{N_t}$ according to the probabilities $\{p_{t,I}(k)\}$.
Set $\Lambda_t = \Lambda_{t-1} \cup \{\boldsymbol{X}_{K_t+1}, \ldots, \boldsymbol{X}_{N_t}\}$.
Set $\boldsymbol{\mu}_t = \boldsymbol{\mu}(\Lambda_t),\ \boldsymbol{\nu}_t = \boldsymbol{\nu}(\Lambda_t)$.

(b) For all k satisfying $p_{t,I}(k) > 0$ obtain $C_{t+1,I}(k)$ from (3), and calculate $p_{t+1,I}(k)$ according to (4).

Step 3

Set $t = t + 1$. Repeat Step 2 until Stopping Rule is satisfied.

2. Convergence

In this section we prove the asymptotical convergence of a certain class of CE algorithms. As mentioned above, the idea and technique of the proof were taken from Gutjahr (2002).

Firstly, we give a definition of so-called construction graph according to Gutjahr (2002).

Definition 2.1. Let an instance of a combinatorial optimization problem be given. By a construction graph for this instance, we understand a directed graph $\boldsymbol{C} = (\boldsymbol{V}, \boldsymbol{E})$ together with a function Φ with the following properties:

1. In $\boldsymbol{C}$, a unique node is marked as start node.
2. Let $\boldsymbol{W}$ be the set of directed paths w in $\boldsymbol{C}$ satisfying the following conditions:
   (i) w starts at the start node of $\boldsymbol{C}$;
   (ii) w contains each node of $\boldsymbol{C}$ at most once;
   (iii) the last node on w has no successor node in $\boldsymbol{C}$ that is not contained in w (i.e., w cannot be prolonged without violating (ii)).

Then function Φ maps a subset $\bar{\boldsymbol{W}}$ of the set $\boldsymbol{W}$ onto the set of feasible solutions of the given problem instance. In other words: to each path w in $\bar{\boldsymbol{W}}$, there corresponds (via Φ) a feasible solution, and to each feasible solution, there corresponds (via Φ^{-1}) at least one path in $\bar{\boldsymbol{W}}$.

The path generation on the construction graph is managed by the following rule. Assuming the current node is r, we generate a successor node according to the probabilities

$$\text{Prob}\{S = s\} = \begin{cases} \dfrac{p_{t,(r,s)}}{\sum_{(r,u)\, is\ feasible} p_{t,(r,u)}}, & \text{if } (r, s) \text{ is feasible,} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Consider the general CE-based algorithm on the construction graph and define the following special case of it (assume we deal with a minimization problem).

Let n denote the number of vertices on the construction graph. For each path $w \in \bar{\boldsymbol{W}}$ we build solution $\boldsymbol{X}$ according to the following rule:

$$X_{(r,s)} = \begin{cases} 1, & \text{the edge } (r, s) \text{ belongs to the path } w, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Add the following artificial components

$$X_{(r,f)} = \begin{cases} 1, & \text{the node } r \text{ doesn't belong to the path } w, \\ 0, & \text{otherwise,} \end{cases} \tag{7}$$

where f does not correspond to any node. (Actually, the last two equations define the function Φ).

Obviously, in our case $\mathcal{I} = \{1, \ldots, n\} \times \{1, \ldots, n+1\}$ (the $(n+1)$-th column corresponds to the artificial node f).

Let the number of solutions participating in any iteration be constant $N_t \equiv N$, and the memory size be $K_t \equiv 1$ (we store only the best solution obtained in the all previous iterations).

Define the parameter of the function $\phi(\cdot)$ by $\boldsymbol{\nu}(\Lambda_t) = (\rho)$. Here the notation ρ comes from the initial versions of the CE method (for example, see Rubinstein (1999)) and denotes the share of the solutions to be used for probability updating. Let $L_{(m)}$ be the m-th order statistic of the following values: $L(\boldsymbol{X}_1), \ldots, L(\boldsymbol{X}_N)$. Additionally, define

$$\phi(L(\boldsymbol{X}), \Lambda_t, \boldsymbol{\nu}(\Lambda_t)) = I_{\{L(\boldsymbol{X}) \le \gamma_t\}},$$

where

$$\gamma_t = L_{(\lceil \rho N \rceil)}.$$

Therefore,

$$\phi(L(\boldsymbol{X}), \Lambda_t, \boldsymbol{\nu}(\Lambda_t)) = \begin{cases} 1, & \boldsymbol{X} \text{ is not worse than the } \lceil \rho N \rceil\text{-th current best solution,} \\ 0, & \text{otherwise.} \end{cases}$$

Particularly, for $\rho = 1/N$ we have:

$$\phi(L(\boldsymbol{X}), \Lambda_t, \boldsymbol{\nu}(\Lambda_t)) = \begin{cases} 1, & \boldsymbol{X} \text{ is one of the current best solutions,} \\ 0, & \text{otherwise.} \end{cases}$$

Obviously,

$$\begin{aligned} C_{t+1,(r,s)} = C_{t+1,(r,s)}(1) &= \frac{\sum_{j=1}^{N} I_{\{L(\boldsymbol{X}_j) \le \gamma_t,\, X_{j,(r,s)}=1\}}}{\sum_{j=1}^{N} I_{\{L(\boldsymbol{X}_j) \le \gamma_t\}}} \\ &= \begin{cases} N^*_{(r,s)} \big/ N^*, & (r,s) \text{ belongs to at least one of the current best solutions,} \\ 0, & \text{otherwise,} \end{cases} \end{aligned} \tag{8}$$

where N^* is the number of solutions within $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N\}$ having the best obtained value, $N^*_{(r,s)}$ is the number of solutions having the best obtained value and containing the edge (r, s).

Define the parameter of the method $\boldsymbol{\mu}_t = \alpha_t$ and let

$$\begin{cases} p_{0,(r,s)} = 1/|\tilde{\boldsymbol{E}}(r)|, \\ p_{t+1,(r,s)} = (1 - \alpha_t) p_{t,(r,s)} + \alpha_t C_{t+1,(r,s)}, \end{cases}$$

where $\tilde{\boldsymbol{E}}(r)$ is the number of arcs outgoing from the vertex r (the artificial arc (r, f) is included).

Formally, we can write the considered algorithm (we will denote this algorithm by the Graph-Based CE Algorithm/Conservative Modification - GBCE/CM) in the following way.

Algorithm 2.1. GBCE/CM:

Step 1

For each (r, s), edge of the construction graph or artificial edge, set $p_{t+1,(r,s)} = 1/|\tilde{\boldsymbol{E}}(r)|$. Set $t = 0$. Generate an arbitrary path $\boldsymbol{X}_1$ using the rule (5), and use it as the current best found path.

Step 2

Generate $N - 1$ solutions by generation corresponding arrays $\boldsymbol{X}_j (j = 2, \ldots, N)$ according to the rule (5). Add $\boldsymbol{X}_1$, which is the best solution found in the previous iterations.

For each (r, s) update the corresponding probability using the following rule:

$$p_{t+1,(r,s)} = (1 - \alpha_t) p_{t,(r,s)} + \alpha_t C_{t+1,(r,s)}, \tag{9}$$

where

$$C_{t+1,(r,s)} = \begin{cases} N^*_{(r,s)}/N^*, & (r, s) \text{ belongs to at least one of the current best solutions,} \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

Step 3
Set $t = t + 1$. Choose α_t. Repeat Step 2 until Stopping Rule is satisfied.

In this step we must note that the above algorithm is very close to the algorithm GBAS/tdev given at Gutjahr (2002). Therefore, we can analyze it using the same technique.

Similarly to Gutjahr, we consider a stochastic process with states $(\boldsymbol{P}_t, \hat{\boldsymbol{X}}_{t-1})$ $(t = 1, 2, \ldots)$, where $\boldsymbol{P}_t$ is the array of probabilities at the beginning of the iteration t, $\hat{\boldsymbol{X}}_{t-1}$ is the best found solution at the end of the iteration $t - 1$. The proof that this process is an inhomogeneous Markov process is analogous to Gutjahr's one.

Theorem 2.1. Consider an optimization (minimization) problem on the construction graph and assume that the optimal solution is unique.

Let, in the algorithm GBCE/CM,

$$\alpha_t \leq 1 - \frac{\log(t + 1)}{\log(t + 2)} \quad (t \geq T) \tag{11}$$

for some $T \geq 0$, and

$$\sum_{t=0}^{\infty} \alpha_t = \infty. \tag{12}$$

Then, with probability one, the states $(\boldsymbol{P}_t, \hat{\boldsymbol{X}}_{t-1})$ of the assigned Markov process converge as $t \to \infty$ to the state $(\boldsymbol{P}^*, \boldsymbol{X}^*)$, where $\boldsymbol{X}^*$ is the optimal solution, and $\boldsymbol{P}^*$ is defined as

$$\boldsymbol{P}^*_{(r,s)} = \begin{cases} 1, & \text{if } (r, s) \in \boldsymbol{X}^*, \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

For a fixed repetition, its probability to obtain $\boldsymbol{X}^*$ at the iteration t tends to 1, as $t \to \infty$.

Proof. The proof is similar to the one given at Gutjahr (2002) with some minor changes.

We have

$$P\{\boldsymbol{X}^* \text{ is never obtained}\} = \prod_{t=0}^{\infty} P\left\{ \begin{array}{l} \boldsymbol{X}^* \text{ isn't obtained} \\ \text{at the iteration } t \end{array} \middle| \begin{array}{l} \boldsymbol{X}^* \text{ wasn't obtained} \\ \text{before the iteration } t \end{array} \right\}. \tag{14}$$

For each fixed arc (r, s), we can calculate the following lower bound on the corresponding probability at the beginning of the iteration t ($t \geq T$). Without lost of generality, let $T \geq 1$. Noting that the lower bound corresponds to the case when the arc (r, s) does not belong to any current best solution until the t-th iteration, we have:

$$p_{t,(r,s)} \geq \left[\prod_{i=0}^{t-1}(1-\alpha_i)\right] p_{0,(r,s)} \geq \left[\prod_{i=0}^{T-1}(1-\alpha_i)\right]\left[\prod_{i=T}^{t-1}\frac{\log(i+1)}{\log(i+2)}\right] p_{0,(r,s)}$$
$$= \left[\prod_{i=0}^{T-1}(1-\alpha_i)\right] p_{0,(r,s)}\frac{\log(T+1)}{\log(t+1)} = \frac{const}{\log(t+1)} \tag{15}$$

(the second inequality follows from the condition (11)).

Hence, the probability to obtain $\boldsymbol{X}^*$ at a fixed repetition of the iteration $t \geq T$ is greater than

$$\left(\frac{const}{\log(t+1)}\right)^{|\boldsymbol{X}^*|},$$

where $|\boldsymbol{X}^*|$ is the number of edges in the path corresponding to $\boldsymbol{X}^*$. Notice that this bound is valid for all scenarios of what can happen before the iteration t, so we can use it for conditional probabilities as well. Consequently,

$$P\left\{\begin{array}{l}\boldsymbol{X}^* \text{ isn't obtained} \\ \quad \text{at the iteration } t\end{array}\middle|\begin{array}{l}\boldsymbol{X}^* \text{ wasn't obtained} \\ \quad \text{before the iteration } t\end{array}\right\}$$
$$= \prod_{n=1}^{N} P\left\{\begin{array}{l}\boldsymbol{X}^* \text{ isn't obtained} \\ \quad \text{at the repetition } n \text{ of the iteration } t\end{array}\middle|\begin{array}{l}\boldsymbol{X}^* \text{ wasn't obtained} \\ \quad \text{before the iteration } t\end{array}\right\}$$
$$\leq \left[1 - \left(\frac{const}{\log(t+1)}\right)^{|\boldsymbol{X}^*|}\right]^N. \tag{16}$$

Hence, the upper bound on (14) can be written as

$$\prod_{t=0}^{T-1} 1 \cdot \prod_{t=T}^{\infty}\left[1 - \left(\frac{const}{\log(t+1)}\right)^{|\boldsymbol{X}^*|}\right]^N = \prod_{t=T}^{\infty}\left[1 - \left(\frac{const}{\log(t+1)}\right)^{|\boldsymbol{X}^*|}\right]^N. \tag{17}$$

Calculating the logarithm of (17), and noting that $\sum_t (\log(t+1))^{-a} = \infty$ for each positive integer a, we obtain

$$N\sum_{t=T}^{\infty}\log\left[1 - \left(\frac{const}{\log(t+1)}\right)^{|\boldsymbol{X}^*|}\right] \leq -N\sum_{t=T}^{\infty}\left(\frac{const}{\log(t+1)}\right)^{|\boldsymbol{X}^*|} = -\infty.$$

Therefore, expression (17) equals to zero, and, consequently,

$$P\{\boldsymbol{X}^* \text{ is never obtained}\} = 0.$$

Hence, the event that in some iteration the optimal solution is reached has probability one.

We have proved that with probability one the considered Markov process $(\boldsymbol{P}_t, \hat{\boldsymbol{X}}_{t-1})$ enters in some iteration into the set $\Theta \times \{\boldsymbol{X}^*\}$, where Θ is a set of all matrices $\boldsymbol{P}$ of dimension $n \times (n+1)$ with the following properties:

$$\begin{aligned} 0 \le p_{(r,s)} \le 1, \\ \sum_s p_{(r,s)} = 1. \end{aligned} \tag{18}$$

(It is not difficult to verify that the last property is valid in the all iterations).

Now we are going to show that the process converges to the state $(\boldsymbol{P}^*, \boldsymbol{X}^*)$.

Notice that when the optimal solution $\boldsymbol{X}^*$ is attained for the first time, formula (10) can be rewritten in the following way:

$$C_{t+1,(r,s)} = \begin{cases} 1, & (r,s) \text{ belongs to the optimal solution,} \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

Let t^* is the iteration where the optimal solution was obtained for the first time and consider an edge (r, s), which *doesn't belong* to the path corresponding to the optimal solution. The probability corresponding to this edge at the beginning of iteration $t^* + \Delta t$ is equal to

$$p_{t^*+\Delta t,(r,s)} = \left[\prod_{t=t^*}^{t^*+\Delta t-1} (1-\alpha_t) \right] p_{t^*,(r,s)}. \tag{20}$$

Using (12) it is not difficult to see that

$$\prod_{t=0}^{\infty} (1-\alpha_t) = 0$$

(to prove it, take the logarithm of the left-hand side).

Hence,

$$\lim_{\Delta t \to \infty} p_{t^*+\Delta t,(r,s)} = \left[\prod_{t=t^*}^{\infty} (1-\alpha_t) \right] p_{t^*,(r,s)} = 0 \tag{21}$$

(for the edges (r, s) that don't belong to the path corresponding to the optimal solution).

Since the normalization property (18) holds and the optimal solution is unique, we have

$$\lim_{\Delta t \to \infty} p_{t^*+\Delta t,(r,s)} = 1 \tag{22}$$

(for the edges (r, s) that belong to the path corresponding to the optimal solution).

This proves that the process $(\boldsymbol{P}_t, \hat{\boldsymbol{X}}_{t-1})$ converges to the state $(\boldsymbol{P}^*, \boldsymbol{X}^*)$.

The last assertion of the theorem (for a fixed repetition, its probability to obtain $\boldsymbol{X}^*$ in the iteration t tends to 1, as $t \to \infty$) follows immediately from the formulae (21) and (22). □

Corollary 2.1. Let, in the algorithm GBCE/CM,

$$\alpha_t = \frac{c_t}{(t+1)\log(t+2)} \quad (t \geq 0) \quad \text{with} \quad 0 < \lim_{t\to\infty} c_t < 1.$$

Then the assertions of Theorem 2.1 hold.

The proof can be found in Gutjahr (2002).

Remark 2.1. The condition $0 < \lim_{t\to\infty} c_t < 1$ can be replaced by a less restrictive one:

$$\liminf_{t\to\infty} c_t > 0 \quad \text{and} \quad \limsup_{t\to\infty} c_t < 1.$$

It involves some minor changes in the proof.

Now consider another version of the CE method, namely, consider an algorithm, which is similar to Algorithm 2.1, but has one difference: instead of (9) the probability updating rule is as follows. Calculate the temporary non-normalized values

$$p^{\text{temp}}_{t+1,(r,s)} = \max\left((1-\alpha)p_{t,(r,s)} + \alpha C_{t+1,(r,s)},\, p^{\min}_{t+1}\right),$$

where $0 < \alpha < 1$ is a constant value, and after that, normalize them:

$$p_{t+1,(r,s)} = \frac{p^{\text{temp}}_{t+1,(r,s)}}{\sum_{(r,u)\textit{is feasible}} p^{\text{temp}}_{t+1,(r,u)}}.$$

This algorithm will be called the Graph-Based CE Algorithm/Conservative Modification with Lower Bound - GBCE/CMLB. (We have to notice that, due to the above normalization, the actual lower bound on the probabilities at the iteration t is less than $p^{\min}_t$). This algorithm is analogous to GBAS/tdlb from Gutjahr (2002).

Theorem 2.2. Let, in the algorithm GBCE/CMLB,

$$p^{\min}_t = \frac{c_t}{\log(t+1)} \quad (t \geq 1) \quad \text{with} \quad \lim_{t\to\infty} c_t > 0.$$

Then the assertions of Theorem 2.1 hold.

Proof. The proof is similar to the one of Theorem 2.1. There are two differences only.

In the part proving that with probability one at some iteration the optimal solution is reached, the lower bound for the probabilities at the iteration t is

$$p_{t,(r,s)} \geq \frac{p_t^{\min}}{|\tilde{\boldsymbol{E}}(r)|} = \frac{c_t}{|\tilde{\boldsymbol{E}}(r)| \log(t+1)} \geq \frac{c}{2|\tilde{\boldsymbol{E}}(r)| \log(t+1)} = \frac{const}{\log(t+1)}$$

for sufficiently large t, where $c = \lim_{t\to\infty} c_t$. This is the same lower bound as in (15).

The proof of convergence of the probability arrays to the optimal one $\boldsymbol{P}^*$ (13) is based on the following argument. Assume the optimal solution was obtained for the first time at the iteration t^*. Consider the edge (r, s) not belonging to the optimal solution. It is not difficult to show that for sufficiently large Δt

$$p_{t+\Delta t,(r,s)} \leq p_{t+\Delta t}^{\min}.$$

Noting that $\lim_{t\to\infty} p_t^{\min} = 0$, we obtain

$$\lim_{\Delta t\to\infty} p_{t+\Delta t,(r,s)} = 0.$$

The remaining part of the proof is analogous to that of Theorem 2.1. □

Remark 2.2. The condition $\lim_{t\to\infty} c_t > 0$ can be replaced by a less restrictive one:

$$\liminf_{t\to\infty} c_t > 0.$$

The proof remains almost the same.

3. Conclusion

We have proved the asymptotical convergence of two cross-entropy algorithms to the optimal solution. This result is analogous to the ones proved for the ant colony optimization and simulated annealing. However, it should be noted that the presented result, as well as the results for the two mentioned above methods, is rather theoretical, because we cannot say anything about convergence rate, as well as about error in each iteration.

References

Dubin, U. (2002). "The Cross-Entropy Method with Applications for Combinatorial Optimization." Master's thesis, The Technion, Israel Institute of Technology, Haifa, Israel.

Gutjahr, W.J. (2002). "ACO Algorithms with Guaranteed Convergence to the Optimal Solution." *Information Processing Letters* 82(3), 145–153.

Homem-de-Mello, T. and R.Y. Rubinstein (2002). "Rare Event Estimation for Static Models Via Cross-Entropy and Importance Sampling." Manuscript.

Margolin, L. (2002). "Cross-Entropy Method for Combinatorial Optimization." Master's thesis, The Technion, Israel Institute of Technology, Haifa, Israel.

Rubinstein, R.Y. (1997). "Optimization of Computer Simulation Models with Rare Events." *European Journal of Operational Research* 99, 89–112.

Rubinstein, R.Y. (1999). "The Cross-Entropy Method for Combinatorial and Continuous Optimization." *Methodology and Computing in Applied Probability* 2, 127–190.

Rubinstein, R.Y. (2001a). "Combinatorial Optimization, Cross-Entropy, Ants and Rare Events." In S. Uryasev and P.M. Pardalos (eds.), *Stochastic Optimization: Algorithms and Applications*, Kluwer, pp. 304–358.

Rubinstein, R.Y. (2001b). "Combinatorial Optimization Via Cross-Entropy." In S. Gass and C. Harris (eds.), *Encyclopedia of Operations Research and Management Sciences*, Kluwer, pp. 102–106.

Rubinstein, R.Y. (2002). "The Cross-Entropy Method and Rare-Events for Maximal Cut and Bipartition Problems." *ACM Transactions on Modelling and Computer Simulation* 12(1), 27–53.

Rubinstein, R.Y. and D.P. Kroese (2004). *The Cross Entropy Method.* Springer-Verlag.

Walker, A.J. (1977). "An Efficient Method for Generating Discrete Random Variables with General Distributions." *ACM Transactions on Mathematical Software* 3, 253–256.