The Cross Entropy Method and Its Applications

Sarwar Hamad

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mathematics

Eastern Mediterranean University
September 2015
Gazimağusa,North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Serhan Çiftçioğlu
Acting Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Mathematics.

Prof. Dr. Nazım Mahmudov
Chair, Department of Mathematics

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Applied Mathematics and Computer Science.

Asst. Prof. Dr. Arif Akkeleş
Supervisor

Examining Committee

1. Assoc. Prof. Dr. Sonuç Zorlu

2. Asst. Prof. Dr. Arif Akkeleş

3. Asst. Prof. Dr. Mustafa Kara

ABSTRACT

The Cross Entropy (CE) method which was initiated and developed by Reuven Rubinstein has been applied to combinatorial optimization problems with promising results. The CE method is actually a generic approach for solving combinatorial optimization. The CE method has been applied successfully to well known optimization problems such as traveling salesman, quadratic assignment problem, and the maximal cuts. In this study, the solution methodology of Traveling Salesman Problem (TSP) for different CE parameters are considered and tested.

Keywords: Travelling Salesman Problem, Genetic Algorithm, CE parameter

ÖZ

Reuven Rubinstein tarafından geliştirilen Çapraz Entropi (CE) yöntemi umut verici sonuçlar ile kombinatoryel optimizasyon problemlerine uygulanmıştır. Çapraz-Entropi (CE) yöntemi CE gibi yolculuk satıcısı, kuadratik atama problemi ve maksimal kesimler olarak optimizasyon problemleri başarıyla uygulanmış olan bir kombinasyon optimizasyonu için genel bir yaklaşımdır. Bu çalışmada, farklı CE parametreleri için Satıcı Problemi (TSP) çözüm yöntemi olarak uygulandi ve testedildi.

Anahtar Kelimeler: Gezgin Satıcı Problemi, Genetik Algoritma

DEDICATION

I am dedicating this work to my parents

ACKNOWLEDGMENT

I would first of all like to thank Allah without whom nothing is possible.

A special thanks to my supervisor Asst. Prof. Dr. Arif Akkeleş for his support and help in the preparation of this study.

Thank to my family member’s for the love and the support they have been giving to me since the beginning of my study.

I am thankful to all my friends in particular Walat Mohammad who contributed a lot for my success during this Master program.

TABLE OF CONTENTS

LIST OF TABLE

LIST OF FIGURES

LIST OF ABBREVIATIONS

AP	Assignment Problem
CE	Cross-Entropy
GA	Genetic Algorithm
QAP	Quadratic Assignment Problem
TSP	Travelling Salesman Problem
GSP	Gezgin Satıcı Problemi

Chapter 1

INTRODUCTION

In general, an optimization problem needs a very reliable an optimal program to be solved. For a given optimization problem, the search of a robust algorithm is always the goal. Nevertheless, the idea carries by all the solution converges to the global optimization. The global optimization goal's is to find the function's global extremum in solution subspace. The global optimization is usually done through several methods such as simulated annealing, evolution strategies, hill climbing methods and the Cross-Entropy method. In this work, the Cross-Entropy method (CE) is explored with the aim to use it for solving c strained optimization problem.

The CE method was developed between 1999 and 2001 by Rubinstein; the idea of CE comes from a work carried out by Rubinstein in 1997 where the aim was variance minimization. The principal objective of the CE method is the modeling of rare events. Such events (rare events) are those which probability of appearance is less than 10^{-4}. The Monte Carlo method is used for the rare event's probability estimation. In practice, the Monte Carlo estimation requires more effort. Furthermore, the effort seems to be inversely proportional to rare event probability. This means if the probability is less than10^{-4}, then the sample size used for the Monte Carlo estimation should be greater than 10^{4}. The main two processes of the CE method consist primary of the gradual change of the sample size to enable an accurate estimation of the rare event probability and secondly the sequence of sample distribution is constructed using the CE [25] [26].

1.1 Structural Reliability Analysis by Importance Sampling

The time invariant structural reliability problem is defined as follow. A real-valued random vector $x=(x_1,x_2,\ldots,x_n)$, together with a joint probability density function $f(x)$ represents uncertain structural parameters. Structural performance depending on random parameters is defined by the limit state function $g(x)$. The limit state function is always defined to be negative value function for parameters which failure occurs. Therefore, the limit state function defines a subset in the random variable space called the failure domain $\Omega_F=\{x:g(x)\le 0\}$ [25] [15]. Further, the failure's probability is defined by

$$P_F=\int_{\Omega_F} f(x)\mathrm{d}x \tag{1.1}$$

this probability of failure can be estimated using Monte Carlo integration[9]. Generally, for engineering structure a small probability of failure is desired. Thus the Monte Carlo method appears as efficient to solve those type of problem. Thus application of variance reduction techniques, like importance of sampling, is usually targeted. The formula for importance sampling for evaluating P_F is based on (1.1), it is rewritten as follows:

$$P_F=\int_{\Omega_F}\frac{f(x)}{h(x)}h(x)dx=E_h\left[I\left(g(x)\le 0\right)\frac{f(x)}{h(x)}\right] \tag{1.2}$$

where $h(x)$ is an importance sampling density[1][9].

$I(\cdot)$is the indicator function of the failure domain, and E_h is the expectation operation with respect to the density $h(x)$.

Having m independent sample points $x^{(k)}, k=1,\ldots,m$ from the distribution $h(x)$, the expectation in (1.2) can be estimated by:

$$\hat{P}_F = \frac{1}{m}\sum_{k=1}^{m} I\left(g\left(x^{(k)} \le 0\right) \frac{f\left(x^{(k)}\right)}{h\left(x^{(k)}\right)} \right). \qquad (1.3)$$

The optimal density function $h(x)$ which minimizes the variance of this estimator is defined to have the following form:

$$h^*(x) = \begin{cases} \dfrac{f(x)}{P_f}, & if \;\; g(x) \le 0, \\ 0, & otherwise \end{cases} \qquad (1.4)$$

However, this formula is more theoretical, because the generation of independent random variables needs the knowledge of the term of our interest P_F. In practice, the distribution, from which samples are built, is generally chosen to resemble the distribution with density $h^*(x)$ [1][9].

1.2 Adaptive Importance Sampling

The sampling distribution is mostly chosen to be a parametric family of distribution from which independent random samples can be generate easily. The family $F = \{ f(x,v), v \in V \}$[25] [26], where v is a vector of parameters, x the random vector and $f(x,v)$ the probability density function is used. Since we are interesting I evaluating the probability of failure mentioned in (1.3), the parameters v should be chosen such a way to facilitate that estimation. For instance, from a multivariate normal distribution data, the parameters v of the sampling distribution can be generated by minimizing the variance by following formula

$$\min_{v\in V} Var_{f(x,v)}\left[I_{\Omega_f}(x)\frac{f(x,u)}{f(x,v)}\right] \tag{1.5}$$

or alternatively by:

$$\min_{v\in V}\left\{ E_{f(x,v)}\left[I_{\Omega_f}(x)\frac{f^2(x,u)}{f^2(x,v)}\right]\right\}. \tag{1.6}$$

The above optimization problem is solved by estimating the expectation as follows:

$$\min_{v\in V}\left\{ \frac{1}{n}\sum_{i=1}^{n} I_{\Omega_f}\left(x^{(i)}\right)\frac{f\left(x^{(i)},u\right)}{f\left(x^{(i)},v\right)}\frac{f\left(x^{(i)},u\right)}{f\left(x^{(i)},v_1\right)}\right\} \tag{1.7}$$

where the probability density of the random sample $x^{(i)},\ i=1,\dots,n$ .Is defined by $f(x,v_1)$ *[25 [15]].*

An adaptive algorithm for the failure probability estimation using (1.7) is defined as follows

1. Take $f(x,v_1)=f(x,u)$. Generate the sample $x_1,\dots,x_N$ with the density function $f(x,v_1)$ then solve the optimization problem (1.7). Denote the solution by $\hat{v}^*$. Assume $\hat{v}^*$ to be the estimation of the optimal parameter vector v^*.
2. Estimate the failure probability based on (1.3) taking $h(x)=f\left(x,\hat{v}^*\right)$.

Take $v_1=\hat{v}^*$ from the first step of the algorithm to accurate the estimate of v^*.

1.3 The Cross-entropy Method

The cross-entropy introduced and developed by Rubinstein in 1997 is usually used for the selection of an important sampling distribution. The cross-entropy method knowing also as Kullback-Leibler distance is based on two probabilities distribution which densities functions are $f(x)$ and $g(x)$ it is defined as follow

$$D(f,g)=\int f(y)\ln\frac{f(x)}{g(x)}dx \ . \tag{1.8}$$

.

Remark: In general $D(g,f)\neq D(f,g)$ thus the cross-entropy method doesn't define a distance function in the formal sense of the definition of a distance, otherwise $D(g,f)=D(f,g)$ [25].

Consider the distribution h^* given by (1.4) and the distribution $f(x,v)\in F$, the cross-entropy can be defined as follows:

$$\begin{aligned} D\left(h^*(x), f(x,v)\right) &= \int P_f^{-1} I_{\Omega_f}(x) f(u) \ln\frac{P_f^{-1} I_{\Omega_f}(x) f(x,u)}{f(x,v)} dx \\ &= E_{f(x,u)}\left[P_f^{-1} I_{\Omega_f}(x) \ln\frac{P_f^{-1} I_{\Omega_f}(x) f(x,u)}{f(x,v)} \right] \end{aligned} \tag{1.9}$$

The distributions h^* and $f(x,v)$ should be similar, therefore the cross-entropy of h^* and $f(x,v)$ should be minimal, in which case the optimal parameter v^* is the solution of the problem

$$\min_{v\in V}\left\{D\left(h^*(x), f(x,v)\right)\right\} \tag{1.10}$$

or alternatively

$$\max_{v \in V} \left\{ D(v) = E_{f(x,u)} \left[I_{\Omega_f}(x) \ln f(x,v) \right] \right\} \tag{1.11}$$

the solution of (1.10) can be approximated using an importance sampling method:

$$\max_{v \in V} \left\{ \hat{D}_n(v) = \frac{1}{n} \sum_{i=1}^{n} I_{\Omega_f}(x_i) \frac{f\left(x^{(i)}, u\right)}{f\left(x^{(i)}, v_1\right)} \ln f\left(x^{(i)}, v\right) \right\} \tag{1.12}$$

1.4 Basic Cross-entropy Algorithm

In general case, the function D in (1.11) is convex furthermore it is a differentiable function with respect to $\mathbf{v}$, so the solution of (1.12) is found by solving the following system of equations [25]:

$$\nabla \hat{D}_n(v) = \frac{1}{n} \sum_{i=1}^{n} I_{\Omega_f}\left(x^{(i)}\right) \frac{f\left(x^{(i)}, u\right)}{f\left(x^{(i)}, v_1\right)} \nabla \ln f\left(x^{(i)}, v\right) = 0 \quad . \tag{1.13}$$

The system (1.13) has a simple form for independent random variable. Let consider for instance a set of normal variables which are independent and which variables have join probability density functions defined by

$$f(x, \mu, \sigma) = \prod_{i=1}^{n} \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left(-\frac{(x_i - \mu_i)^2}{2\sigma_i^2} \right) \tag{1.14}$$

where $\mu = \{\mu_1, \dots, \mu_n\}$ are mean values and $\sigma = \{\sigma_1, \dots, \sigma_n\}$ are the standard deviations of the components. The gradient of the logarithm of the probability density function are defined as follow

$$\frac{\partial \ln f(x, \mu, \sigma)}{\partial \mu_i} = \frac{\mu_i - x_i}{\sigma_i^2}, \qquad i = 1, \dots n, \tag{1.15}$$

$$\frac{\partial \ln f(x, \mu, \sigma)}{\partial \sigma_i} = \frac{(x_i - \mu_i)^2 - \sigma_i^2}{\sigma_i^3}, \qquad i = 1, \dots n. \tag{1.16}$$

Thus, the following set of equations for optimal parameters of (1.14) can be obtained by substituting formulas (1.15) and (1.16) into (1.13) [7] [10]:

$$\hat{\mu}_i^* = \frac{1}{n}\sum_{i=1}^{n} x_i I_{\Omega_f}\left(x^{(i)}\right)\frac{f\left(x^{(i)},0,1\right)}{f\left(x^{(i)},\mu_1,\sigma_1\right)} \tag{1.17}$$

$$\hat{\sigma}_i^{*2} = \frac{1}{n}\sum_{i=1}^{n}\left(x_i-\mu_i\right)^2 I_{\Omega_f}\left(x^{(i)}\right)\frac{f\left(x^{(i)},0,1\right)}{f\left(x^{(i)},\mu_1,\sigma_1\right)} \tag{1.18}$$

Using equations (1.13), the algorithm proposed for minimum variance criteria can be adapted easily for the probability of failure estimation using the cross-entropy optimal parameters.

Chapter 2

LITERATURE REVIEW

2.1 History

The Cross Entropy (CE) is said to be an efficient and trustful method, for the computation and estimation of probabilities of rare-event. Later on, research in the CE fields made of it a robust to for both combinatorial optimization and for the rare event simulation.

During about a half century, a method called Kullback-Leibler or simply Cross Entropy which has successfully been used for measuring information in various fields of sciences. Therefore, the actual Cross Entropy, was developed and got its name from the Kullback-Leibler. The Kullback-Leibler was particularly used in the field of neural computation.

The Cross Entropy is a method based on iterative computation following two main steps [26].

- The generation of a random data sample (it is generally vectors, trajectories etc.) using a random mechanism.
- The update of data generated in the first step by the random mechanism to produce to produce an accurate sample for the next stage iteration.

The key of the Cross Entropy is that it has a precise and concise mathematical structure and the sample parameters are defined for deriving fast. This makes sense in term of an optimal point of view.

Many combinatorial and optimization problems have got worth solution from the cross entropy method. There are the traveler Selman problems, the quadratic assignment problem, the maximal cut problem, the buffer allocation problem, just to name few of them. Both deterministic problem and noisy problem can be solved using the Cross Entropy method.

Dr. David Wolpert et al have a collection of probability works which aims are related to the Cross Entropy method. His approach is based on information theory like a bridge to put together, statistical physic, game theory, and distributed optimization control system.

Usually in the rare-event simulation field, the Cross Entropy method is used in association with another method called the important Sampling.

Chapter 3

INTRODUCTION EXAMPLES TO THE CE METHOD

In the introduction chapter, we defined the CE method and its algorithm. In this chapter, we will discuss about how the CE method works via a simple case of continuous optimization problem. Those examples will range rare-event simulation, to the combinatorial optimization.

3.1. Some Various Illustration Examples

3.1.1 Example Based on Rare-event Simulation

Consider the following weighted graph [25].

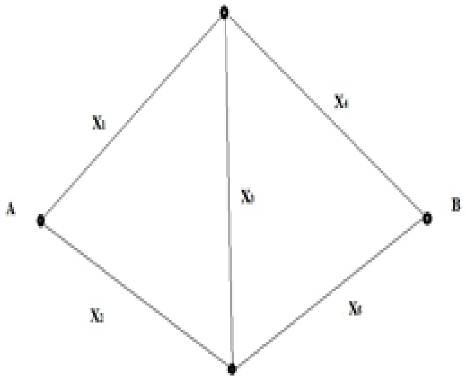

Figure 3.1: weighted graph

Figure 3.1 is a weighted graph which weights are random and denoted by $x_1, \ldots, x_5$. Now let assume that the independent variables $x_1, \ldots, x_5$ (weights) are exponentially distributed with respective means $u_1,\ldots,u_5$. The probability distribution is then defined as

$$f(x,u)=\exp\left(-\sum_{j=1}^{5}\frac{x_j}{u_j}\right)\prod_{j=1}^{5}\frac{1}{u_j} \quad . \tag{3.1}$$

There exists a shortest path to move from the node A to the node B. Let's denote the length of this path by $S(x)$. Based on the aim of this study, simulation is used to estimate [3] [7] [25].

$$l=P\big(S(x)\geq\gamma\big)=EI_{\{S(x)\geq\gamma\}} \tag{3.4}$$

equation is actually the probability that the shortest path length's exceed a given fixed value γ. A common method estimation of the value in the equation (3.4) is the use of Crude Monte Carlo (CMC) simulation. The CMC consist to the draw of a random sample $x_1,\ldots,x_N$ from distribution of x for further use.

$$\frac{1}{N}\sum_{i=1}^{N}I_{\{S(x)\geq\gamma\}} \tag{3.5}$$

in this case (3.5) is considered to be an unbiased estimator of l. The probability l is very small when the value γ is large. Using the CMC in this case leads us to a very large effort. N must be large in order to obtain an precise value of l. The mentioned effort can be avoided by the use of the importance sampling (IS). It consists to consider another probability function $g(x)$ such that $g(x)=0\Rightarrow I_{\{S(x)\geq\gamma\}}f(x)=0$. Using $g(x)$, the probability l can be symbolize as

$$l = \int I_{\{S(x)\geq\gamma\}} \frac{f(x)}{g(x)} g(x)dx = E_g I_{\{S(x)\geq\gamma\}} \frac{f(x)}{g(x)} \quad . \tag{3.6}$$

The inferior g on E is to show the computation is done with respect to g . Here g is the important sampling (IS) [7] [25]. It follows in this case that an unbiased estimator of l is

$$\hat{l} = \frac{1}{N}\sum_{i=1}^{N} I_{\{S(x)\geq\gamma\}} W(x_i) \tag{3.7}$$

here, $\hat{l}$ =importance sampling or likelihood ratio estimator.

$$W(x) = \frac{f(x)}{g(x)} \tag{3.8}$$

is the likelihood ratio (LR).

$x_1,\ldots,x_N$ is a random sample comes from g .

In the particular case where $g = f$, we have $W = 1$. The likelihood ratio estimator in (3.7) becomes the Crude Monte Carlo of (3.5).

Consider g to be such that $x_1,\ldots,x_5$ are independently, exponentially distributed, with respective means $v_1,\ldots,v_5$. Then we have

$$W(x;u,v) = \frac{f(x;u)}{f(x;v)} = \exp\left(-\sum_{j=1}^{5} x_j \left(\frac{1}{u_j} - \frac{1}{v_j}\right)\right)\prod_{j=1}^{5}\frac{v_j}{u_j} \tag{3.9}$$

The vector parameter $v = (v_1, \ldots, v_5)$ is used to determine the "change of measure". The problem here seems to be the reverse of the problem states by the Crude Monte Carlo simulation. This means a certain simulation effort is given, and we try to select the vector parameter v which leads us to the accurate estimation of l [8].

Applying the CE algorithm mentioned in introduction chapter, with initial parameter N , N_1 , ρ between 0.01 and 0.1. Furthermore let consider that the vector parameter $u = (0.25,\ 0.4,\ 0.1,\ 0.3,\ 0.2)$ and assuming that we are computing the probability that $S(x) \geq \gamma = 2$.

Using the CMC method with 10^7 samples leads to an estimated value of $1.65*10^{-5}$ with a relative estimated error of 0.165. Using now a sample of 10^8, the estimated value is $1.30*10^{-5}$ with a relative estimated error of 0.03 [25].

3.1.2 Example Based on Combinatorial Optimization

Let us consider $y = (y_1, \ldots, y_n)$ being a binary vector in which we assume not to know the entrance of y which are 0 as well as those which are 1. However, we assume that there exists an "oracle predictor" which for each input vector $x = (x_1, \ldots, x_n)$ returns the response $S(x) = n - \sum_{j=1}^{n} \left| x_j - y_j \right|$. The CE method can be used here as follow for the combinatorial optimization. Generating a sequence of parameters vectors $\hat{p}_0, \hat{p}_2, \cdots$ and sequence of levels $\hat{\gamma}_1, \hat{\gamma}_2, \ldots$ such that $\hat{\gamma}_1, \hat{\gamma}_2, \ldots$ converges to optimal performance n and $\hat{p}_0, \hat{p}_2, \cdots$ to the optimal degenerated parameters vectors which coincide with y .

Assuming we have the following parameters $\mathbf{y}=(1,1,1,1,1,0,0,0,0,0)$ initial parameters vectors $\hat{p}_0=\left(\frac{1}{2},\frac{1}{2},\ldots,\frac{1}{2}\right)$ and $N=50$, with $\rho=0.1$. The result is shown in the following table. It is clear that the convergence of $\hat{p}_t$ and $\hat{\gamma}_t$ to the respective optimal parameter $p^*=y$ and the optimal performance $\gamma^*=n$ is fast.

The numerical results of this can be found in [9].

3.2 The CE Method for Simulation and Optimization

3.2.1 Case of Rare-event Simulation

The simulation rare event based on the cross entropy method is discussed here. The ideas or roots of the CE method will be explicated here.

Let's consider a random vector $x=(x_1,\ldots,x_n)$ which values are taken in a space x . Let μ be a measure and $\{f(\cdot;v)\}$ a density probability functions family on x with respect to μ . Here v is a parameters vector of real values. It follows that $EH(x)=\int_\chi H(x)f(x;v)\mu(dx)$, with H being any measurable function. In what will follow, let's assume that $\mu(dx)=dx$ for simplicity.

Let's consider now a real value function $S(x)$ on x. Let assume that we want to compute the probability that $S(x)$ is greater than or even equals to a real value γ . Here the value γ is considered to be a threshold (level) under $f(\cdot;v)$. The mentioned probability is computed by

$$l = P_u\left(S\left(x\right) \geq \gamma\right) = E_u I_{\{S(x)\geq\gamma\}} \tag{3.10}$$

if the probability computed in (3.10) is very small, if it is for instance smaller than 10^{-5}, then $\left\{S\left(x\right) \geq \gamma\right\}$ is said to be a rare-event [23] [25].

The estimation of l in (3.10) can be done by the by the Monte-Carlo simulation. In which case, $\frac{1}{N}\sum_{i=1}^{N} I_{\{S(x)\geq\gamma\}}$ is defined to be an unbiased estimator of l. Nevertheless, the crude Monte-Carlo [25] simulation become a brainstorm, when $\left\{S\left(x\right) \geq \gamma\right\}$ is a rare event; because the simulation effort needs to reach the aim is very large.

An important alternative method of solving this problem is based the importance sampling. This is stated as follow. Consider a random sample $x_1,\ldots,x_n$ from one importance sampling density g on χ, next estimate l by the likelihood ratio (LR) estimator.

$$\hat{l} = \frac{1}{N}\sum_{i=1}^{N} I_{\{S(x_i)\geq\gamma\}} \frac{f\left(x_i;u\right)}{g\left(x_i\right)} \tag{3.11}$$

The best estimation of l is done by using the change of measure with the following density function

$$g^*\left(x\right) = \frac{I_{\{S(x)\geq\gamma\}} f\left(x;u\right)}{l} \tag{3.12}$$

it follows from (3.11) and (3.12) that

$$I_{\{S(x_i)\geq\gamma\}} \frac{f\left(x_i;u\right)}{g^*\left(x_i\right)} = l, \qquad \forall i\,. \tag{3.13}$$

Now we have l which is a constant, it follows that the estimator defined by (3.11) has a zero variance. Therefore, we need for the process a number $N = 1$ sample just.

At this level what seems obviously to be a difficulty is that the function g^* depends mainly on the parameter l which is unknown. It is convenient that g is chosen, such to belong to the following densities family $\{f(.;v)\}$. The goal can be reached, if we adopt the following idea. That is to choose v, the reference parameter (also named by tilting parameter) such a way to minimize the distance between the foregoing g^* and the function $f(.;v)$. On the other hand, the Kullback-Leibler distance is a convenient method for measurement of the distance between two densities functions. The following formula defined the Kullback-Leibler distance

$$D(g,h) = E_g \ln \frac{g(x)}{h(x)} = \int g(x) \ln g(x) dx - \int g(x) \ln h(x) dx \qquad (3.14)$$

NB: In (3.14), the word distance is used to call $D(g,h)$, but it is just conceptual. Actually $D(g,h)$ haven't all the fulfillment properties of a distance. For instance $D(g,h)$ is not symmetric [25].

Now recalling the formula (3.12), the Kullback-Leibler [17] [20] distance is minimized between g^* and the function $f(.;v)$ if and only if v is chose in a way that $-\int g^*(x) \ln f(x;v) dx$ is minimized. Which actually lead us to solve the following maximization problem

$$\max_v \int g^*(x) \ln f(x;v) dx \qquad (3.15)$$

recalling g^* from (3.12), and substituting it into (3.15), the following maximization program is obtained

$$\max_v \int \frac{I_{\{S(x)\geq\gamma\}} f(x;u)}{l} \ln f(x;v)\,dx\,. \tag{3.16}$$

Finally recalling (3.14), the problem stated in (3.16) is equivalent to

$$\max_v D(v) = \max_v E_u I_{\{S(x)\geq\gamma\}} \ln f(x;v) \tag{3.17}$$

3.2.2 Case of combinatorial optimization

The CE method algorithm for combinatorial optimization is discussed in this section.

The following maximization problem is use as guide to illustrate the method.

Consider x to be a finite set of class. Let S be the real-valued efficiency function on x. The problem is to find the maximum S of over x and to find the state(s) at which attained S this maximum. Let called the maximum γ^* [13] [16]. It follows that

$$S(x^*) = \gamma^* = \max_{x\in\chi} S(x)\,. \tag{3.18}$$

The first thing to do is the association of the optimization problem stated in (3.18) with an estimation problem which is meaningful. For various levels $\gamma \in R$, a set of indicator functions $\left\{I_{\{S(x)\geq\gamma\}}\right\}$ is defined. Let consider next $\{f(.;v), v\in V\}$ to be a family of probability densities on χ [25] [3]. Where v is a real valued vector which parameterized the probability densities functions $\{f(.;v), v\in V\}$. For a given $u\in V$, we associate to (3.18) the number estimation problem

$$l(\gamma) = P_u\left(S(x) \geq \gamma\right) = \sum_x I_{\{S(x)\geq\gamma\}} f\left(x;u\right) = E_u I_{\{S(x)\geq\gamma\}} \tag{3.19}$$

where P_u is the probability computed when the random state has the probability density function $f\left(.;u\right)$ and is E_u the corresponding expectation operator. The show the association between (3.18) and (3.19), let consider the following assumptions $\gamma^* = \gamma$ and $f\left(.;u\right)$ is the density uniformly defined on χ. It is important to note that typically $l\left(\gamma^*\right) = f\left(x^*;u\right) = {}^{1}\!\!/_{|x|}$. Where $|x|$ is the cardinality of χ. This means,for $\gamma^* = \gamma$, a good way of estimating $l\left(\gamma\right)$ is to use the LR estimator $\hat{l} = \frac{1}{N}\sum_{i=1}^{N} I_{\{S(x)\geq\gamma\}} W\left(x_i;u,\hat{v}_T\right)$ with reference parameter v^* given by

$$v^* = \arg\max_v E_u I_{\{S(x)\geq\gamma\}} \ln f\left(x;v\right) \tag{3.20}$$

The parameter in (3.20) could be estimated by

$$\hat{v}^* = \arg\max_v \frac{1}{N}\sum_{i=1}^{N} I_{\{S(x)\geq\gamma\}} \ln f\left(x_i;v\right) \tag{3.21}$$

In (3.21), x_i are generated from the probability densities function $f\left(.;u\right)$. It is furthermore clear that when γ is almost equals to γ^*, then probability mass assigned by $f(.;v^*)$ are close to x^*. This can therefore be used to compute an approximate solution of the problem stated in equation (3.18). Generally, the estimator given by the formula (3.21) is suitable and useful if and only if $I_{\{S(x)\geq\gamma\}} = 1$ [3] [7]. In which case, one should choose u such that $P_u\left(S\left(x\right) \geq \gamma\right)$ shouldn't be too small. From what preceded it is clear that, there is a close relation between the choice of γ and u.

The entire procedure mentioned is implemented by the following algorithm [9] [22].

Algorithm: (Main Cross Entropy Algorithm for Optimization)

1. Consider $\hat{v}_0^* = u$. Set the counter $t = 1$.

2. Create a sample $x_1, \dots, x_N$ using the density $f(.; v_{t-1})$ then compute the $(1-\rho)$-quantile $\hat{\gamma}_t$ sample's performance.

3. Base on the same initial sample $x_1, \dots, x_N$, solve the following stochastic program $\max_v \hat{D}(v) = \max_v \frac{1}{N} \sum_{i=1}^{N} I_{\{S(x) \ge \hat{\gamma}_t\}} W(x_i; u, \hat{v}_{t-1}) \ln f(x_i; v)$, setting $W = 1$ and denote its solution by $\hat{v}_t$.

4. If for a given $t \ge d$, for instance $d = 5$,

$$\hat{\gamma}_t = \hat{\gamma}_{t-1} = \cdots = \hat{\gamma}_{t-d}$$

(3.22) then stop the process (denote by T the final iteration); else set $t = t + 1$ and iterate the process from the step 2.

Chapter 4

CROSS-ENTROPY METHOD FOR POWERFULL SIMULATION

In sciences, usually the performance of system such as storage system, telecommunication networks, assurance risk, and inventory system is based on rare-event. However, the simulation of rare-event using the crude Monte Carlo method requires a considerably large number of trials and it’s therefore, time and space consuming. To palliate to it, new methods are developed [25] [8].

In chapter some of the techniques used for an optimal rare-event simulation are explored. They are importance sampling, Kullback-Leiber Cross-Entropy [17].

4.1 Importance Sampling

Considered the following stochastic system which the expected performance l is given by:

$$l = E_f H(x) = E_f \varphi\left(S(x), \gamma\right) = \int \varphi\left(S(x), \gamma\right) f(x) \mu(dx) \qquad (4.1)$$

where S represents the sample performance function. $\varphi(., \gamma)$ is the real-valued function based on the sample performance. The expectation E_f is computed here with respect to f that is why f is at the subscript of E_f. One can have for instance the indicator functions [17][12] [25]

$$\varphi\left(S(x);\gamma\right) = I_{\{S(x)\geq\gamma\}} \tag{4.2}$$

and the Boltzmann functions

$$\varphi\left(S(x);\gamma\right) = \exp\left(-S(x)/\gamma\right). \tag{4.3}$$

Considering for instance the framework problem of the stochastic shortest path, the shortest path can be computed by

$$S(x) = \min_{j=1,\dots,p} \sum_{i\in B_j} x_i \tag{4.4}$$

where B_j stands for the j-th complete path that moves from the source to the sink. The exist p completes paths. x_i is the duration or the weight of the links.

Consider another density function g such that it dominated Hf. This means $g(x)=0 \Rightarrow H(x)f(x)=0$. Following the foregoing relation, the performance value l can be computed by

$$l = \int H(x)\frac{f(x)}{g(x)}g(x)\mu(dx) = E_g H(x)\frac{f(x)}{g(x)}. \tag{4.5}$$

In equation (4.5), the expectation is computed with respect to the function g. The function g is said to be the importance sampling density [2].

An estimator of the mean value l which is unbiased is given by

$$\hat{l} = \frac{1}{N}\sum_{i=1}^{N} H(x_i)W(x_i) \tag{4.6}$$

In equation (4.6), the value $\hat{l}$ is called the likelihood ratio (LR) estimator or the importance sampling (IS).

The function W is the ratio of the two functions f and g.

$$W(x) = f(x)/g(x) \ . \tag{4.7}$$

The function W is called usually the likelihood ratio. There exists a single particular case where $(f = g)$, this happens when there is no change of the measure. In such case, we have $W = 1$ [25]. The estimation given in the equation (4.6) is therefore reduced simply to the crude Monte Carlo (CMC) estimator given by

$$\hat{l} = \frac{1}{N}\sum_{i=1}^{N} H(x_i) \tag{4.8}$$

In equation (4.8), the series $x_1,\ldots,x_N$ is a random sample vector coming from the density function f.

While choosing the IS density g, the minimization of the variance of the mean estimator $\hat{l}$ should be considered.

The minimization of $\hat{l}$ variance is stated as follow with the respect to the density function g.

$$\min_g Var_g\left\{H(x)\frac{f(x)}{g(x)}\right\}. \tag{4.9}$$

As reminder, the problem stated by the equation (4.9) has the following solution

$$g^{*}(x)=\frac{|H(x)|f(x)}{\int |H(x)|f(x)\mu(dx)}. \tag{4.10}$$

It is important to notice that in case $H(x)\geq 0$, then

$$g^{*}(x)=\frac{H(x)f(x)}{l} \tag{4.11}$$

and

$$Var_{g^{*}}(\hat{l})=Var_{g^{*}}(H(x)W(x))=Var_{g^{*}}(l)=0. \tag{4.12}$$

The density function g^{*} is named the optimal importance sampling density [25] [17].

4.2 Kullback-Leibler Cross-Entropy

This method can also be used instead of the variance minimization method. Here optimal parameter vector is computed based on a "distance" defined between two probability distribution functions g and h. This distance is defined as [17]

$$\begin{aligned} D(g,h) &= \int g(x)\ln\frac{g(x)}{h(x)}\mu(dx) \\ &= \int g(x)\ln g(x)\mu(dx)-\int g(x)\ln h(x)\mu(dx) \end{aligned} \tag{4.13}$$

reminding that the distance $D(g,h)\geq 0$; i.e. the distance should be positive with the equality $D(g,h)=0$ only if $g=h$ and the $\mu-$measure of the set is 1. The aims of the CE method in this case is to choose the important sample density function h in a way to minimize the kullback-Leibler distance which exist between h and the optimal IS density g^{*} defined in (4.10) . It follows that the solution of the functional optimization problem stated by

$$\min_{h} D(g^{*},h) \tag{4.14}$$

is the CE importance sample density h^{*} . Based on the relation $D(g^{*},h) \geq 0$, it is obvious that solution of the problem stated in (4.14) is given by $h^{*} = g^{*}$. By optimization over all the density function h , the CE importance sampling (IS) and the variance minimization (VM) densities function coincide. On the other hand, using the sampling likelihood ration (SLR) approach, there is a restriction of the densities function class to a family $\{f(.;v), v \in V\}$, in which the nominal density $f(.;u)$ is included. Following the CE method, the aims is now to solve the following parametric optimization problem

$$\min_{h} D\left(g^{*}, f(.;v)\right) \tag{4.15}$$

recalling the equation (4.10), we have $g^{*}(x) = k^{-1}|H(x)|\, f(x)$ with $k = \int |H(x)|\, f(x)\,\mu(dx)$. Considering the formula given by the equation (4.13), the right-hand side is independent on v . Therefore, the process of minimizing the Kullback-Leibler distance between $f(.;v)$ and g^{*} is exactly equivalent to the process of maximizing, the following equation with respect to v.

$$\int |H(x)|\, f(x;u) \ln f(x;v) \mu(dx) = E_{u} |H(x)| \ln f(x;v) . \tag{4.16}$$

Assuming for simplicity that $H(x) \geq 0$, the absolute signs are dropped from the formula (4.16) and the optimal parameter based on the Kullback-Leibler distance is given by the solution of the equation [17]

$$\max_{v} D(v) = \max_{v} E_u H(x) \ln f(x;v) \tag{4.17}$$

which is equivalent to

$$\max_{v} D(v) = \max_{v} E_w H(x) W(x,u,w) \ln f(x;v) \tag{4.18}$$

This $\forall w$ tilting parameter. With $W(x,u,w)$ being the likelihood ratio given by

$$W(x,u,w) = \frac{f(x;u)}{f(x;v)}.$$

The optimal solution v^* can be estimated by computing the optimal solution of the following program [25]

$$\max_{v} \hat{D}(v) = \max_{v} \frac{1}{N} \sum_{i=1}^{N} H(x_i) W(x_i,u,w) \ln f(x_i;v). \tag{4.19}$$

With $x_1, \dots, x_N$ being a random sample from $f(.;w)$.

The program stated by the equation (4.19) is called the stochastic counterpart of the Cross Entropy program states by the equation (4.18). It can also be called the simulated Cross Entropy program. The function $\hat{D}$ is typically a differentiable and concave function with respect to v. Therefore the optimal solution of equation (4.19) may be obtained by solving the following equations system with respect to v.

$$\frac{1}{N} \sum_{i=1}^{N} H(x_i) W(x_i,u,w) \Delta \ln f(x_i;v) = 0. \tag{4.20}$$

The solution of (4.18) may also be obtained by solving the equation

$$E_u H(x) \Delta \ln f(x; v) = 0 \tag{4.21}$$

Chapter 5

COMBINATORIAL OPTIMIZATION

Combinatorial optimization is a subset of optimization that is related to algorithm theory, operation research, and computational complexity theory. Combinatorial optimization algorithms are mainly used in many applications like planning, management, operation of telecommunication and communication networks. To find the optimal path or to find the combination of paths that leads to NP-hard problems is one of the important combinatorial problems. There exist a number of well known methods for finding optimal or near optimal solutions to these problems like simulated annealing [4], tabu search [21], genetic algorithms [23], ant colony system [5] and the cross entropy method [3] [22] [16] [24] [7] [15] [18]. The cross entropy (CE) method (Rubinstein and Kroese, (2004) was motivated by Rubinstein in 1997. It originated from the field of rare event simulation. The CE method is a general Monte-Carlo approach to combinatorial and continuous multi-extremal optimization and importance sampling. The method derives its name from the Kullback-Leibler cross entropy distance [17] [20]. The CE method was modified in [3] by Rubinstein to solve both continuous multi-extremal and various discrete combinatorial optimization problems. There are three main steps of CE method for optimization problems:

1. Translate the optimal problem into an associated estimation problem, the so called associated stochastic problem (ASP).

2. Generate sample data by choosing a probability family, initializing the parameters, and then generating feasible solutions according to the chosen distribution.

3. Update parameters based on the Kullback-Leibler [17] [20] cross entropy distance.

The main goal of this work is to introduce more efficient ranges of CE parameters like sample size (*N*), smoothed parameter (α) and rarity parameter (ρ), and to modify CE algorithm for optimization to achieve optimal solution with less computational time and less iteration number. The Traveling Salesman Problem (TSP) is considered a tutorial problem for the CE method. We used MATLAB to implement cross entropy method for solving the TSP. Numerical experiments were performed with different CE parameters on different sizes of matrices for TSP. With these numerical experiments we conclude our observations. The rest of the work is organized as follows: In the second section the general CE method is described. In the third section, the results of the numerical experiments and observations are presented. And the last section summarizes and concludes this work.

Chapter 6

PRELIMINARIES

In this section we present some background on the CE method and TSP.

6.1 The Cross Entropy Method

As mentioned above CE method was developed by Rubinstein in 1999 for solving optimization problems. The reader is referred to Rubinstein and Kroese (2004), de-Boer at al. (2005) and references therein for context [9], extensions and applications.

The main idea of the CE method for optimization is given below. Suppose that we aims tominimize (maximize) some objective function say $S(x)$ over all $x \in X$. Let take a fixed parameter γ' and set

$$\gamma' = \min_{x \in X} S(x) \tag{6.1}$$

then we define a family of probability density function, $f\{(\cdot), v \in V\}$ on the random vector x [3] [22]. Then the optimal problem translated into an associated stochastic problem which is defined below

$$\ell(\gamma) = p_u\left(S(x) \geq \gamma\right) = E_u I_{(S(x)\geq\gamma)}\,. \tag{6.2}$$

Cross Entropy Method for Combinatorial Optimization Problems. ASP is the expected value of the index set that satisfies the condition which is objective function value is greater than or equal to fixed parameter γ . Converting optimization problem to the estimation problem was the first step of CE method for the optimization. At the second step , the random variables are generated from the probability density function. Then the reference parameters $\{v_t, t \geq 0\}$ and the levels $\{\gamma_t, t \geq\}$ are initialized. After that feasible solutions are generating according to the chosen distribution. At the last step parameters are updated based on the Kullback-Leibler CE distance to produce a better sample in the next iteration. The main CE algorithm for optimization is summarized in Algorithm 6.1 [9] [22].

The CE Algorithm 6.2

Step 1: *Choose some* v_0 *, set* $t = 1$.

Step 2: Generate a sample $x_1, x_2, ..., x_N$ from the density $f(\cdot, v_{t-1})$ and compute the sample $(1-\rho)-$quantile γ_t of the performances according to $\gamma_t = S_{\lceil(1-\rho)\rceil}$.

Step 3: Use the same sample $x_1, x_2, ..., x_N$ and solve the stochastic program

$$\max_v \hat{D}(v) = \max_v \frac{1}{N} \sum_{i=1}^{N} I_{(s(x_i)\geq\gamma)} W(x_i; u, w) In\, f(x_i; v) \qquad (6.3)$$

denote the solution *by* (v_t).

Step 4: Apply smoothed equation $\hat{v}_t = \alpha\tilde{v}_t + (1-\alpha)\hat{v}_{t-1}, \forall i = 1, ..., n$ to smooth out the vector $\hat{v}_t$.

Step 5: If for some $t \geq d$, say $d = 5$, $\hat{\gamma}_t = \hat{\gamma}_{t-1} = \cdots = \hat{\gamma}_{t-d}$ then stop *(*let *T*

denote the final iteration); otherwise set $t = t + 1$ and reiterate from Step *2*.

The above algorithm can be summarized as follows;

At the first step of the algorithm the parameter vector *v* was initialized and the level Counter *t* was set to 1. At the second step, the random sample data was generated from the chosen probability density function to calculate °*t* values that satisfies the following condition

$$p_{v_{t-1}}\left(S\left(x\right)\leq\gamma_t\right)\geq 1-\rho \qquad (2.4)$$

where ρ is the rarity parameter. The choice of rarity parameter plays a critical role to keep CE algorithm close to global extrema with high probability and avoid local one.

At the third step CE algorithm iterates by using v_{t-1} and γ_t to update v_t by solving the stochastic program 2.3 according to Kullback-Lieber cross entropy distance. In the Stochastic program $W\left(X_i;u,w\right)$ is the likelihood ratio of the probability density function. At the fourth step, by using smooth parameter α , algorithm smoothed out the values of v_t to reduce the probability that some component $v_{t,i}$ to be 0 or 1 at the first few iterations, which causes algorithm to converge wrong solution. Last step is the stopping criterion. If the CE algorithm runs with the same objective function values for say *d* = 5 times then the algorithm terminates and set *t*= *T* where *T* is the number of iteration. Otherwise algorithm increases the level counter and continues with step 2.

6.2 Traveling Salesmen Problem (TSP)

In this paper we used MATLAB to implement CE method for solving the TSP [9] [22] which aims to find the shortest tour that visits all the cities exactly once. Let the objective function $Z\left(x\right)$ be the total length of tour $x\in X$ then we calculate

$\min_{x\in X} Z\left(x\right)=\min_{x\in X}\left\{\sum_{i=1}^{n-1}c_{xi,xi+1}+c_{n,1}\right\}$ where $x=\left(x_1,x_2,\cdots,x_n\right)$ with $x_1=1$ denotes permutation of $\left(1,2,\cdots,n\right)$, x_i where $i=1,2,\cdots,n$ is the i'th city to be visited in the tour represented by x, and c_{ij} is the distance (or cost) from city i to city j. The main CE algorithm for TSP is summarized in Algorithm 2.2 [22] [3] [16].

Step 1: Choose an initial reference transition matrix b $\hat{p}_0$ say with all off diagonal elements equal to $\frac{1}{n-1}$. Set $t=1$.

Step 2: Generate a sample $x_1, x_2, \cdots, x_N$ of tours via Trajectory Generation using Node Transitions Algorithm [16], with $P = \hat{P}_{t-1}$ and compute the sample $(1-\rho)$ quantile of the performances according to $\hat{\gamma}_t = \lceil (1-\rho) N \rceil$.

Step 3 : Use the same sample to update $\hat{P}_t$ via $\hat{p}_{t,ij} = \frac{\sum_{k=1}^{N} I_{(S(X_K)\le\gamma_t)} I_{X_K \in X_{ij}}}{\sum_{K=1}^{N} I_{(S(X_K)\le\gamma_t)}}$.

Step 4: Apply smoothed equation to smooth out the matrix.

Step 5: If for some $t \ge d$, say $d = 5, \hat{\gamma}_t = \hat{\gamma}_{t-1} = \cdots = \hat{\gamma}_{t-d}$ then stop, otherwise set $t = t+1$ and reiterate from step 2.

6.3 Implementation Notes

The parameters that are used by the CE method (the CE parameters) are the sample size N, the rarity parameter (ρ) and the smoothing parameter (α). In this paper we tweak the CE parameters and compare the ones that are used by Rubinstein (2004). According to [22] the CE parameters for MATLAB computer program of TSP are chosen: $< N = 10n^2$ (where n is number of nodes), $\rho = 0{:}01$ and $\alpha = 0{:}7$ (smoothed parameter) within the ranges: $5n^2 \le N \le 10n^2, 0.3 \le \alpha \le 0.8$

$\rho = \left\{ \begin{matrix} 0.01, \text{ if } n \geq 100 \\ \frac{\ln n}{n}, \text{ if } n \prec 100 \end{matrix} \right\}$ respectively. In order to achieve optimal or near optimal solution with less computational time and less iteration number, different CE parameters within their ranges are used. By this method different results are obtained and comparisons are done with the known ones. We implement CE method for solving the problem by using a computer with properties Intel (R) Core (TM)i7 CPU and 3.07 GHz processor with at least 10 times reapplication. Many implementations are done for CE applications of TSP by various sizes.

Table 6.1: Br17 with 17 nodes (best known optimal solution is 39)

Sample number(N)	Rarity parameter ρ)	Smoothing parameter α)	Av. Count	Av. Opt. sol	Av. cpu. time	Av. Error (e)
n^2=289	0.01	0.3	13.3	39.8	3.4509	0.0203
		0.7	9	47.6	2.33228	0.2002
		0.8	8.3	48.6	20.5581	0.2457
	0.16	0.3	43.4	41.1	12.17276	0.0533
		0.7	23.9	39.5	7.3576	0.0127
		0.8	21.1	40.1	6.49422	0.0278
5 n^2=1445	0.01	0.3	42.3	44.7	65.22319	0.145
		0.7	10.6	39	13.7305	0
		0.8	9.7	39.1	12.42068	0.0025
	0.16	0.3	44.4	42.4	68.43775	0.0863
		0.7	26.8	39	41.27225	0
		0.8	24.2	39.2	37.20877	0.005
		0.3	16	39	41.517	0

10 n^2*=2890*	0.01	0.7	10.3	39	24.73429	0
		0.8	9.5	39	41.27225	0
	0.16	0.3	47.9	43.2	147.64644	0.169
		0.7	27.6	40	84.95841	0.0254
		0.8	42.3	40.6	74.27452	0.04

The generalization of all results is done by comparing the best known solution. The new obtained results are generalized in known example Br17 [19] with 17 nodes where the best known solution is 39 and P43 with 43 nodes where the best known solution is 5620. Table 6.1 presents 18 different outcomes for TSP for br17 [19].

6.4 Observations

Observations are done for different ranges of CE parameters. In this section some results and observations of CE algorithm stated as follows:

- It follows form the table 6.1 that he following values of CE parameters gives us best known optimal solution which is 39 for Br17 [3] [19] [18] as it is seen in table 6.1.

Table 6.2: CE parameters

Sample number	Rarity parameter	Smoothing parameter	Av. Rel. Error
N=5n^2	ρ =0.01	α =0.7	$\varepsilon = 0$
N=5n^2	ρ =0.16	α =0.7	$\varepsilon = 0$

N=10n^2	ρ =0.01	α =0.3 α =0.7 α =0.8	

- The parameter values which results in exact optimal solution each produce different average CPU times and average iteration numbers. For the following CE parameters algorithm 2.2 produces best average CPU time (13.7305) with smallest average relative experimental error (0): For N=10n^2, ρ = 0.01 and α = 0.3 values algorithm produces highest average CPU time which is 41. 517 seconds.

For N=5n^2, ρ = 0.01 and α = 0.7 values algorithm produces highest average CPU time which is 13.7305 seconds.

As a result of an observations when N=5n^2= 1445, ρ = 0.7 and α = 0.01 algorithm 2.1 runs with best average number of iterations and average number of CPU time with 10.6 and 13.7305 seconds respectively.

- During the numerical experiments we observed that when smoothing parameter α increased both average CPU time and iteration number decreased. Therefore, observations show that CPU time and iteration number with smoothing parameter α are not proportional. Also it is observed that when rarity parameter ρ increases both CPU time and number of iterations increases too. Hence, rarity parameter ρ and number of iterations are proportional.

- Although for small sizes of matrices like n <100, elite sample percentile ρ is suggested $\frac{\ln n}{n}$ [22], our numerical experiments show that by taken elite sample percentile ρ equal to 0:01 instead of $\frac{\ln n}{n}$ it is obtained less average number of iterations and average number of CPU time with zero relative error. These results are better than the results given in [18].

- When $\rho = \frac{\ln n}{n}$ = 0.16 there is only one optimum solution at α = 0.7, N=5n^2=1445 and α = 0.7. Results show that while $\rho = \frac{\ln n}{n}$ when Samples number (N) increases, average number of CPU time, number of iterations and average relative error increases also.
- There must be an increment of smoothing parameter α , when rarity parameter ρ is $\frac{\ln n}{n}$ in order to be a decrement of average number of CPU time and the average number of iterations. In other words, to obtain more effective results, if there is an increment on rarity parameter ½ there must be an increment on smoothing parameter (α)As a result, for the following CE parameters N=5n^2=1445, α = 0.7, ρ = 0.01algorithm runs with best average iteration number and average CPU time with 10.6 and13.7305 in seconds, respectively.

The following table contains results of CE algorithm for TSP for P43 [19] cost matrix where the best known optimal solution is 5620. It can be observed from the table that best known optimal value and smallest relative experimental error of the problem is obtained for the CE parameters N=10n^2=1445 = 18490, α = 0.3 and ρ = 0.01.

Table 6.3: P43 with 43 nodes (best known optimal solution is 5620)

Sample number(N)	Rarity parameter(ρ)	Smoothing parameter α	Av. Count	Av. Opt. sol	Av. cpu. time	Av. Error (e)
n^2=1849	0.01	0.3	62.3	5635	631.92	0.00262
		0.7	26.9	5656.3	277.37368	0.0062
		0.8	24.7	5669.3	248.67647	0.0083
	0.16	0.3	185.5	5628	2053.9	0.00132
		0.7	87.9	5632.8	990.51	0.00195
		0.8	81.3	5634.7	889.67	0.00256
5 n^2=9245	0.01	0.3	62.3	5635	631.92	0.00262
		0.7	38.2	5625.2	1973.411	0.0009255
		0.8	33.6	5627.1	1680.8	0.00128
	0.16	0.3	177.8	5627.9	9911.157	0.001406
		0.7	110.1	5624.9	6117.101	0.000872
		0.8	98.1	5625	5986.155	0.0008563
10 n^2=18490	0.01	0.3	81.7	5623.9	8381.058	0.000694
		0.7	42.9	5624.3	4431.513	0.000765
		0.8	36.6	5629.06	3346.2513	0.0006942
	0.16	0.3	88.6	5624.6	6500.2	0.00139
		0.7	110.1	5624.9	6117.097	0.0008721
		0.8	94.8	5625.2	10259.37	0.0009256

- It follows from the table that CE algorithm archives near optimal value of the problem with smallest average relative experimental error (0.000765) for the following CE parameters: N = 102 = 18490, ρ = 0:01 and α = 0.7

- Numerical results show that if the smoothing parameter α increase from 0.3 to 0.7then the average CPU time and the average iteration number decrease at most by a factor of 0.5 as observed in br17.

- Similar to the br17 results the choice of rarity parameter ρ has a contrast with suggested one by Rubinstein [22] [18]. As mentioned before it was suggested to select

$$\rho = \begin{cases} 0.01 \; if \;\; n \geq 100 \\ \dfrac{\ln n}{n} \;\; if \;\; n < 100 \end{cases}$$

in our numerical experiments it is observed that although for both cases n is less than100 with the choice of $\rho = \dfrac{\ln n}{n}$ algorithm runs with worse CPU time and average iteration number than the value of ρ = 0.01.It is known that the choice of rarity parameter plays an important role to achieve an optimal solution for TSP. This effect can be easily seen in the graphical representation of rarity parameter and CPU time of the iteration of the algorithm to get the optimal solution of the problem. The following graph is rarity parameter versus CPU time for matrix br17 and p43where the other CE parameters are fixed as below: for br17 *5* n^2=1445, α =0.7 and N = *10* n^2=2890, α =0.7, for p43 N = 5 n^2=9245, α =0.7 and N = *10* n^2 =18490, α =0.7.

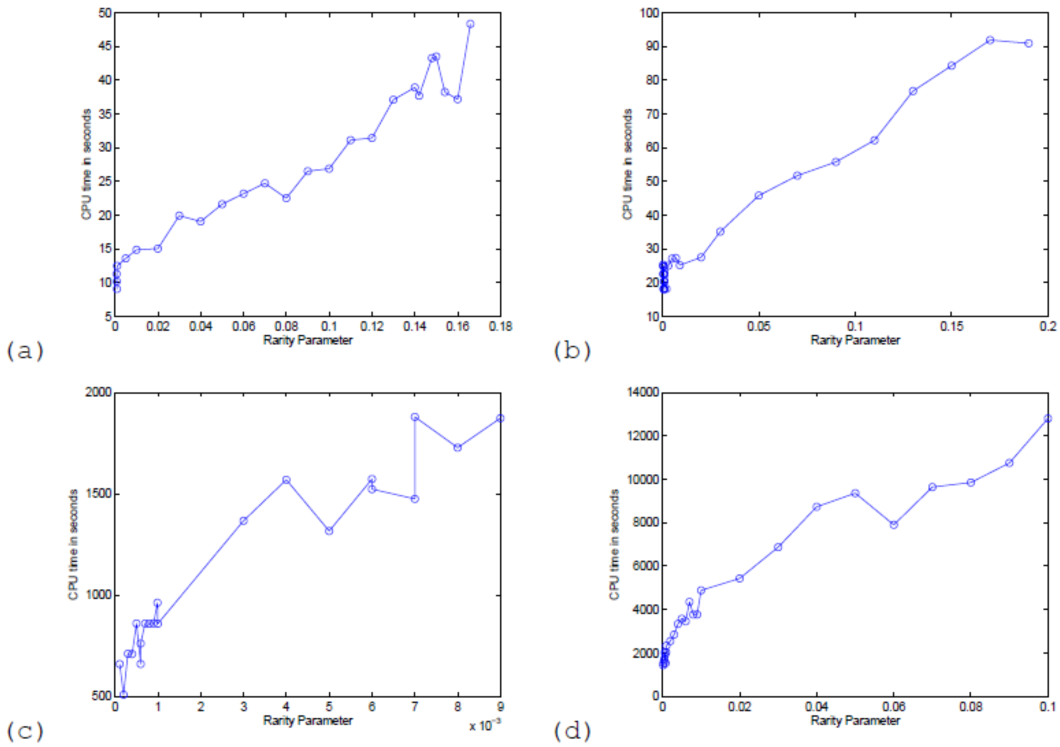

Figure 6.1: Results graph

This figure includes (a) $N = 5\ n^2$ =1445, α =0.7, (b) N = 10 n^2 =2890, α =0.7 for br17 cost matrix of TSP and (c), (d) includes N =5 n^2=9245, α =0.7 and N = 10n^2 =18490, α =0.7 for p43 cost matrix for TSP respectively.

It follows from the graph 3.1 that the CPU times increase during the increments of rarity parameter. Numerical experiments show that the value of the rarity parameter ρ =0.01 gave better CPU time (14.849085) with 0 relative experimental error than the CPU time (37.152770) for $\rho = \frac{\ln n}{n} = \frac{\ln 17}{17}$ =0.16 in graph 3.1 (a). It was also found that the lower bound for ρ is 0.0007, N =5 n^2=1445. In graph 3.1 (b) for N = 10 n^2 =2890, $\alpha = 0.7$ experiments show that the value of the rarity parameter ρ =0.01 gave better CPU time (28.015066) with 0 relative experimental error than the

CPU time (78.66035) for $\rho = \frac{\ln n}{n} = \frac{\ln 17}{17}$ =0.16. It was also found that the lower bound for ρ is 0.00035.In the graph 3.1 (c) and (d) are rarity parameter versus CPU time for matrix p43 where the other CE parameters are fixed. When $N = 5\ n^2$ =9245, α =0.7 it follows from the graph3.1 (in both (c) and (d)) that the CPU times increase during the increments of rarity parameter. Numerical experiments show that the value of the rarity parameter ρ =0.01 gave better CPU time (1817.72464) with lower relative experimental error than the rarity parameter $\rho = \frac{\ln n}{n} = \frac{\ln 43}{43}$ =0.08 where the CPU time=8542.748905 and it was also found that the lower bound for ρ is 0.0001.

When N = 10 n^2 =18490, α =0.7 then from the graph 3.1 (d) shows that the CPU times increase during the increments of rarity parameter. Again as in other numerical experiments also show that the value of the rarity parameter ρ =0.01 gave better CPU time (4883.11913) with lower relative experimental error than the rarity parameter $\rho = \frac{\ln n}{n} = \frac{\ln 43}{43}$ =0.08 where the CPU time=9850.426922. It was also found that the lower bound for ρ ½ is 0.00055.

All these observations can be summarized as below in a table 3.2;

1. It follows from the graphs that the CPU times increase during the increments of rarity parameter.

2. Following table gives the values of CPU times for different values of rarity parameter ρ.

Table 3.3: Rarity parameter relation with CPU time and lower bounds for Cross Entropy Method for Br17 and P43.

	Lower bound for ρ	**Rarity parameter** ρ	**CPU time**	**Optimal solution**
n^2	0.0007	0.01	14.849085	39
		$\frac{\ln 17}{17}$=0.16	37.152770	39
Br17-10 n^2	0.00035	0.01	28.015066	39
		$\frac{\ln 17}{17}$=0.16	78.66035	39
P43-5 n^2	0.0001	0.01	4883.11913	5629
		$\frac{\ln 43}{43}$=0.08	9850.426922	5626
P43-10 n^2	0.00055	0.01	1817.724640	5624
		$\frac{\ln 43}{43}$=0.08	8542.748905	5627

It can be seen from the table that the value of the rarity parameter ρ=0.01 gave better CPU time than the rarity parameter $\rho = \frac{\ln n}{n}$. Also it was observed that the lower bound of ρ decrease at most a factor of 0.5 when the sample size N increase from 5 n^2 to 10 n^2.

Chapter 7

CONCLUSION

In this thesis we investigate Cross-Entropy method and application on the solution methods of TSP.

In order to achieve optimal or near optimal solution with less computational time and less iteration number, different CE parameters within their ranges are used. By this method different results are obtained and comparisons are done with the known ones. We implement CE method for solving the problem by using a computer with properties Intel (R) Core (TM) i7 CPU and 3.07 GHz processor with at least 10 times reapplication. Many implementations are done for CE applications of TSP by various sizes. The generalizations of all results are done by comparing the best known solution. The new obtained results are generalized in known example Br17 with 17 nodes where the best known solution is 39 and P43 with 43 nodes where the best known solution is 5620.

After all simulations of CE algorithm, the obtained results show that parameter choosing plays an important role for obtaining optimal or near optimal solution with less iteration and less computational time. For the following CE parameters algorithm 2.2 runs with best average CPU time (13.7305) with zero average relative experimental error for TSP with matrix br17: $N = 5\, n^2 = 1445$, $\alpha = 0.7$, $\rho = 0.01$. For the following CE parameters algorithm 2.2 runs with best average CPU time

(4431.513) with smallest average relative experimental error for TSP with matrix p43: (N = 10, n^2= 18490, α = 0.7, ρ = 0.01) we also observe that the choice of rarity parameter ρ used in the algorithm plays a critical role to keep CE algorithm close to global extreme with high probability and avoid to local one. Our numerical experiments show that the choice of rarity parameter ρ has a contrast with suggested one by Rubinstein in [22] [3]. It is observed that although for both cases n is less than 100 with the choice of $\rho = \left(\frac{\ln n}{n}\right)$ algorithm runs with worse CPU time and average iteration number than the value of ρ = 0:01 as it is seen in table 3.1.

Same experiments can be done for Quadratic Assignment problem or other well-known combinatorial optimization problems for further research. Some techniques can be applied to speed up CE algorithm for different combinatorial optimization problems. While the CE method has been widely deployed to efficiently solve a wide range of difficult problems, such as the Max-Cut, Quadratic Assignment Problem and Traveling Salesman problems, there still remains a great deal to be understood about the dynamics and convergence properties of the method.

REFERENCES

[1] M.A.Nielsen and I.L.Chuang ,*Quantum Computation and Quantum Information*(Cambridge University Press , Cambridge, England, 2000).

[2] M. Abdel-Aty and H. Moya-Cessa, Phys. Lett. A 369, 372 (2007).

[3] R.Y. Rubinstein, *The Cross Entropy Method for Combinatorial and Continuous Optimization,Methodology and Computing in Applied Probability*, pp 127-190 (1999).

[4] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Optimization by Simulated Annealing*, Science 220, pp. 671680 (1983).

[5] M. Dorigo and L. M. Gambardella, Ant Colony System: *A Cooperative Learning Approach to the Traveling Salesman Problem*, IEEE Transactions on Evolutionary Computing, vol. 1, (1997).

[6] M. Dorigo and G. D. Caro, *Ant Algorithms for Discrete Optimization, Artificial Life*, vol. 5, no. 3, pp. 137172, (1999).

[7] Asmussen, S., Kroese, D.P., Rubinstein, R.Y. Heavy Tails, Importance *Sampling andCross-Entropy*. Stochastic Models 21 No. 1, pp. 57-76 (2005).

[8] L. Margolin, *On the convergence of the Cross-Entropy Method*, Annals of Operations Research, 134, pp. 201-214 (2004).

[9] De Boer, P-T., Kroese, D.P, Mannor, S. and Rubinstein, R.Y. *A Tutorial on the Cross-Entropy Method*. Annals of Operations Research. Vol. 134(1), pp 19-67 (2005).

[10] Costa, A., Jones, O.D, Kroes, D. P. Convergence Properties of Cross-Entropy Method for Discrete Optimization Operations Research Letters 35(5), 573-580 (2007).

[11] Costa, A.,Jones, O.D, Kroese, D. P. *Convergence Properties of the Cross-Entropy Method for Discrete Optimization.* Operations Research Letters, 35(5), 573-580 (2007).

[12] Hu, J., and Hu, P. On The performance of the cross-entropy method, proceedings of the (2009) winter simulation conference (WSC), PP. 459-468 (2009).

[13] Jenny Liu, Global optimization technique using Cross-Entropy and Evolution Algorithms Coursework Master Thesis, Department Of Mathematics, The University of Queensland, (2004).

[14] L. Margolin. On The Convergence Of The Cross-Entropy Method. Annals Of Operations Research, vol. 134,pp 201-214, (2005).

[15] D. Peleg, S. Mannor and R. Y.Rubinstein, The Cross Entropy Method For Classification Proceeding Of The 22-nd International Conference On Machine Learning, Bonn, Germany,(2005).

[16] R. Y. Rubinstein” The Stochastics Minimum Cross Entropy Method For Combinatorial Optimization And Rare-Event Estimation”. “Methodology And Computing In Applied Probability Vol.7, No 1.pp 5-50,(2005).

[17] Kullbck, S.,R. A. Leibler. On information and sufficiency. Ann. Math. Statist. 22, 7986(1915).

[18] Rubinstein, R. Y. Cross Entropy and Rare Events for Maximal Cut And partition Problems. ACM Trans. Model. Comput. Simulation 12, 2753 (2002).

[19] http://www2.iwr.uni-heidlberg.de/groups/comput/software/TSPPLIB95/atsp/

[20] Kullback, S. , ”Letter To The Editor: The Kullback Leibler Distance”. The American Statistician 41 (4), 340341 (1987).

[21] F.Glover, Tabu search. Kluwer, (1996).

[22] R. Rubinstein, D. Kroese. The Cross Entropy Method, Springer (2004).

[23] D. Goldberg, Genetic Algorithms in search, optimization and machine learning Addison Wesley, (1998).

[24] R. Y. Rubinstein, D. p. Cross Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, (2005).

[25] Rubinstein, D. Kroese. The Cross-Entropy Method, Springer (2004).

[26] A.Akkeleş, A.Öneren, F.Bilen ”Cross-Entropy Method And Combinatorial Problems. EMU, (2012).