中華民國八十四年全國計算機會議論文集
Proceedings of National Computer Symposium 1995

退火基因演算法的收斂分析
THE CONVERGENCE ANALYSIS OF THE ANNEALING-GENETIC ALGORITHM

林豐澤
Feng -Tse Lin

中國文化大學應用數學系
Department of Applied Mathematics, Chinese Culture University, Taipei, Taiwan

摘要

退火基因演算法(AG)是一個很有用的最佳化工具，可求解很難的組合性最佳化問題。此法是我們兩年前提出的。AG 的途徑是將基因演算法加入模擬退火中，以來改進模擬退火以及基因演算法兩者的執行績效。此法具有下列之特性：(1)它可被視是一個模擬退火演算法，但具有人口基底的狀態遞移以及基因運算為主的準平衡控制功能。(2)它可被視為是一個基因演算法，但具有波茲曼型式的挑選運算。實驗結果顯示，AG演算法求解多重限制的 0/1 背包問題的誤差率少於1%，求解集合分割問題是少於 0.1%，求解銷售員旅行與箱櫃包裝問題是接近 3%。在所有的測試案例中，AG 演算法比起模擬退火與基因演算法有較佳的績效表現。

於本篇論文，我們提出一種綱目理論來探討 AG 演算法的效率性。我們不僅証明 AG 演算法的全體收斂性，更分析了演算法的收斂行為與其執行時間之複雜度。演算法的收斂時間最多是 $O(\log\Delta(Sc)^*(1-\varepsilon)N+1/N\beta)$，其中$\Delta(Sc^*)$ 是最佳綱目 Sc* 繁殖到下一代的最小挑選機率，N 是每一代人口數目，β是曾經達到最佳解的機率。

關鍵詞：模擬退火，基因演算法，退火基因演算法，全體收斂，綱目理論，波茲曼分佈

Abstract

The Annealing-Genetic (AG) algorithm is a powerful optimization tool for solving hard combinatorial optimization problems. The AG approach was proposed by author two years ago. This approach incorporates genetic algorithms into simulated annealing for improving both the performance of simulated annealing and genetic algorithms. The AG approach has the following features: (1) it can be viewed as a simulated annealing algorithm with the population-based state transition and with the genetic-operator-based quasi-equilibrium control, (2) it can be viewed as a genetic algorithm with the Boltzmann-type selection operator. Empirically, the error rate of the AG algorithm for solving the multiconstraint zero-one knapsack problem is less than 1%, for solving the set partitioning problem is less than 0.1%, and for solving both the traveling salesman as well as bin-packing problems are around 3%. In all the test cases, the AG approach obtains much better performance than either simulated annealing or genetic algorithm does. In this paper, we present the schemata theory for discussing the efficiency of the AG algorithm. We not only show the global convergence of the AG algorithm but also include the analysis of the convergence behavior and the running time complexity of the algorithm. The ratio of convergence is bounded by $O(\log_{\Delta(Sc^*)(1-\varepsilon)}N + 1/N\beta)$, where $\Delta(Sc^*)$ is the minimum selection ratio of optimal schema S_c^* for propagating to the next generation, N is the size of the population and β is the probability of ever visiting the optimum.

keyword : simulated annealing, genetic algorithms, annealing-genetic algorithm, global convergence, schemata theory, Boltzmann distribution.

1. Introduction

Both simulated annealing (SA) and genetic algorithms (GAs) are nature-based, stochastic computation techniques, the former is based on thermodynamics and the latter is based on natural evolution. During the past few years, they have been applied to solve many combinatorial optimization problems in diverse areas[1, 2, 5]. Theoretically, the SA can be viewed as an algorithm that generates a sequence of Markov chains for a sequence of decreasing temperature values. At each temperature, the generation process is repeated again and again until the probability distribution of the system states approaches the Boltzmann distribution[8]. If the temperature is decreased slowly enough, the Boltzmann distribution tends to converge to a uniform distribution on the set of globally optimal states. Thus, SA may require a long computation time in order to converge to the optimal solutions. However, in any implementation of the algorithm, asymptotic convergence can only be approximated [1, 7, 8, 12]. Due to these approximations, the SA algorithm is no longer guaranteed to find a global optimum with probability 1. Moreover, without a carefully designed annealing schedule, SA may easily be trapped into a local optimum.

GAs were pioneered by John Holland [6] that based on the survival-of-the-fitness principle and try to inherit more genetic information from generation to generation. Whenever some individuals exhibit better than average performance, the genetic information of these individuals will be reproduced more often. GAs work with a rich database of population and simultaneously climb many peaks in parallel during the search so that the probability of trapping into a local optimum is reduced significantly. However, there are several drawbacks prohibiting GAs from becoming a powerful optimization tool for various problems[2,5]. First, it needs a chromosomal representation of solutions to the problem. Second, if the highly fit, short, low-order schemata (also called building blocks) are misled due to coding used or the function itself, the problem may require a long evolution time to arrive at near optimal solutions. Third, it is widely recognized that genetic algorithms are not well suited to perform finely-tuned local search.

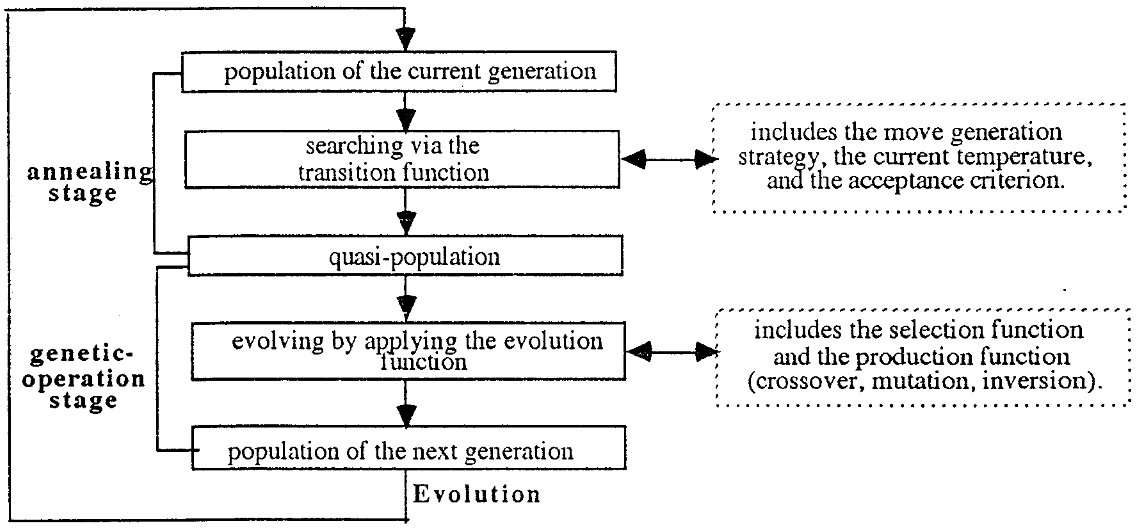
Consequently, we propose an approach that incorporating the advantages of GAs into SA so that it is very similiar to a combination of steepest descent and Newton's method to improve each other for the constrainted optimization problem of the continuous type. Our approach is called annealing-genetic (AG)[9]. The AG approach has the following features: (1) it can be viewed as a SA algorithm with the population-based state transition and with the genetic-operator-based quasi-equilibrium control, (2) it can be viewed as a GA with the Boltzmann-type selection operator. Our previous empirical results [9,10,11] show that the proposed approach seems to facilitate the exhaustive and parallel treatment of the hard problems and to increase the probability of finding global optimums. Empirically, the error rate of the AG algorithm for solving the multiconstraint zero-one knapsack problem is less than 1%, for solving the set partitioning problem is less than 0.1%, and for solving both the traveling salesman as well as the bin-packing problems are around 3%. In all the test cases show that, the AG approach obtains much better performance than either SA or GA does.

In this paper, we present the schemata theory for discussing the efficiency of the algorithm. We not only show the global convergence of the AG algorithm but also include the analysis of the convergence behavior and the running time complexity of the algorithm. The ratio of convergence is bounded by $O(\log_{\Delta(Sc^*)(1-\varepsilon)^N} + 1/N\beta)$, where $\Delta(Sc^*)$ is the minimum selection ratio of optimal schema S_c^* for propagating to the next generation, N is the size of the population and β is the probability of ever visiting the optimum. In the remainder of this paper, we briefly describe the AG approach in Section 2. Section 3 shows why the schemata exhibit high efficiency in a large search space to guide the searches of the search paths. Then, we show the global convergence of the AG algorithm by a serial theorems in Section 4. The conclusions are finally given in Section 5.

2. The Annealing-Genetic Algorithm

The AG approach consists of a set of two-stage cycles. The first is the search stage by the annealing process and the second is the evolution stage by the genetic operations. Fig.1 shows the basic concept of the AG approach and it is described as follows. Starting off with a set of arbitrary points (individuals) from the population of current generation at the current temperature, our approach creates a number of different search paths that trying to find better solutions simultaneously. A heuristic with a probability measurement is used to guide these search paths when they climb in the search space. The heuristic includes a probabilistic generation mechanism and a probabilistic acceptance criterion. The generation mechanism is implemented by a transition function. At each temperature, the points visited by these search paths become the members of a set called the quasi-population. When the size of the quasi-population reaches a predefined population size, the search stage is complete. This action is an implicit parallelism which is absent in the other approaches to implement SA. It is worthwhile to note that our approach is search from a population instead of search from a single point as in the other approaches. After completing the quasi-population of the first stage, the second stage is started. The genetic-operation stage is an evaluation of the evolution function. The evolution function consists of a selection function and a production function. After applying the evolution function to the quasi-population, the population of the next generation is created. The next generation is more mature than the last one in the sense that the average performance of the population has been improved. The new structures of the population and the new temperature are then evaluated, and the cycle repeats until a specified number of generations have been done or the temperature approaches the frozen condition.

The AG algorithm is formally described as follows. Let t_i be an arbitrary point in the search space Ω. The population of current generation P is a fixed size subset of Ω, $P \subset \Omega$. Actually, the annealing stage is the search by SA. SA evaluates the population of the current generation, determines the current temperature T, and creates many search paths at the temperature T. Initially, one search path starts



population of the current generation
searching via the transition function
includes the move generation strategy, the current temperature, and the acceptance criterion.
annealing stage
quasi-population
genetic-operation stage
evolving by applying the evolution function
includes the selection function and the production function (crossover, mutation, inversion).
population of the next generation
Evolution

Fig. 1 The basic concept of the AG approach.

with a randomly selected point t_i in P and the remaining points of the search path are proceeded and based on the following searching procedure. Given a point t_i, the next point t_j is obtained by the move generation strategy M_g. Let $\Delta C = C(t_j) - C(t_i)$, the probability for point t_j to be the next point of the search path is determined by the Metropolis criterion M_c at the current temperature T. The search path is continued only when t_j is accepted; otherwise another search path is started. Let $SP = \{t_j \mid t_j \in \Omega;\ t_j$ is accepted by $M_c\}$ denote a search path and the search paths have been indexed by $SP_1, SP_2, \ldots$. Then, the quasi-population is $P_q = SP_1 \cup SP_2 \cup \ldots$. We define the above search procedure as a *transition function* of $F_t = [M_g, M_c, T]$ which means the search paths are determined by the move generation strategy, the Metropolis criterion, and the current temperature, then $F_t(P) = P_q$. The genetic-operation stage is the evolution by GAs. The selection function F_s is to choose a set of parents from P_q to the mating pool for generating the offsprings, i.e. mating pool = $\{(t_i, t_j) \mid (t_i, t_j) \in F_s(P_q)\}$. The production function F_p includes the crossover (cro), the mutation (mut), and the inversion (inv) operators. F_p is used to produce the children from the set of parents in the mating pool and to create the population of the next generation, i.e., $F_p(t_i, t_j) = (cro + mut + inv)\ (t_i, t_j) = \{t_k \mid t_k \in \Omega,\ t_k$ is the offspring of $(t_i, t_j)\} = P$, where the operator "+" means "or".

The genetic-operation stage is defined as an *evolution function* of $F_o(P_q) = (F_p(F_s(P_q)) = P$. The AG algorithm can be formulated as a sequence of interaction between the transition function and the evolution function. Let $\{w_n \mid n \geq 0\}$ denote the cycles (repetitions) of the AG algorithm. The cycles can be defined as the following iteration and also shown in Fig. 2.

for n = 0, w_0 = P, is the initial population.

for $n \geq 1$, $w_n = F_o(F_t(P)) = F_o(SP_1 \cup SP_2 \cup \ldots) = F_o(P_q) = (F_p(F_s(P_q)) = P$, is the population of n-th cycle.

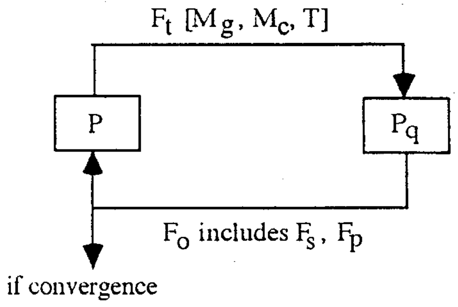


Fig.2 The execution of AG is a sequence of the interaction of F_t and F_o.

If we let the time index include in the population P, i.e. P(0) denotes the initial generation, P(1) denotes the first generation, we have $w_0 = P(0)$, $w_1 = F_o(F_t(P_0)) = P(1), \ldots, w_n = F_o(F_t(P_{n-1})) = P(n)$. Let Avg(P(k)) denote the average performance of the population in the kth generation. We say that the evolution satisfies the monotone restriction if $Avg(P(k)) \geq Avg(P(k+1))$, for $k \geq 0$. The AG algorithm will satisfy the monotone restriction if we let the genetic operators to be performed according to the following steps.

Step 1. The parents are selected from the population based on their fitness values. Then, applying the the crossover operator to the parents to produce their offsprings. The offsprings must have their performance not worse than the average performance of the old generation; otherwise, the parents continuffollowing steps.

Step 2.Applying the inversion operator to the parents for reordering their own sequence to obtain the new structures. If the performance of the new structures are better than the parents, then they come into the next generation; otherwise, the parents continue the last step.

Step 3. Applying the mutation operator to the parents for expecting a better performance structure.otherwise, the parents remain in the next generation.

3. The schemata theory

In this section, we discuss why the schemata exhibit high efficiency in a large search space to guide the searches of the search paths. The discussion will be given by examining a simple example of Multiconstraint Zero-One Knapsack Problem (MCKP). Formally, MCKP is a maximization problem and has the following form :

$$\text{Maximize } Z = \sum_{j=1}^{n} p_j x_j \, ,$$

$$\text{Subject to } \sum_{j=1}^{n} w_{ij} x_j \leq b_i \text{ and } x_j = 0 \text{ or } 1,$$

$$\text{where } i = 1, \ldots, m \text{ and } j = 1, \ldots, n.$$

We give a simple instance of the MCKP with the problem size $n = 6$ and the constraints $m = 10$ which is shown in Table I. The optimal solution is a vector $X = (0, 1, 1, 0, 0, 1)$ with cost = 3800. We define a schema is a building block describing a subset of points with similarities at certain positions in a point. By examining the fitness of any one point, one might expect to obtain information about other points which have a structure similar to it. Usually, we can create 3^n schemata over the extended alphabet $\{0, 1, *\}$ with length = n in each point. The notion of schemata arises from the observation that in evaluating the fitness of a point we can also derive implicit knowledge about the schemata which describe that point. The accuracy of this extrapolation depends on the specification of the given schema. It is important to us to identify the sets of schemata which have above average fitness in the population. Because the important similarities among highly fit points can help guide a search during the search stage by SA,

we can imagine sets of schemata as the hyperplanes. These high performance hyperplane schemata also play an important role in investigating the structure of large, complex search spaces. However, the high performance schemata is limited in each hyperplane. Table II shows the high performance schemata in term of $\#(S_c)$, the number of specific positions contained in the schema, in increasing order of the given example.

Table I Test data with n = 6 and m = 10.

j	p_j	w_{1j}	w_{2j}	w_{3j}	w_{4j}	w_{5j}	w_{6j}	w_{7j}	w_{8j}	w_{9j}	w_{10j}
1	100	8	8	3	5	5	5	0	3	3	3
2	600	12	12	6	10	13	13	0	0	2	2
3	1200	13	13	4	8	8	8	0	4	4	4
4	2400	64	75	18	32	42	48	0	0	0	8
5	500	22	22	6	6	6	6	8	8	8	8
6	2000	41	41	4	12	20	20	0	0	4	4
	b_j =	80	96	20	36	44	48	10	18	22	24

optimal solution = 3800 ;
$x_i = 1$ (i = 2, 3, 6); all others = 0.

In order to identify the high performance schemata, it is necessary to study the dynamics of schemata frequency as a function of search progress. By examining which schemata have above average fitness values, we can foresee the population of subsequent generations due to the fact that these schemata are expected to proliferate. Then, the structure of the future populations can be estimated through schema frequency and fitness. It means that the expected proliferation of above average schemata are at the rate of $O(\log_{N} * \Delta(Sc) * (1 - \varepsilon)^{N})$ which we will be proved in the next section. Fig. 3 shows the growth of high performance schemata $\#(S_c) = 1, 2, 3, 4, 5$, and 6, which are indicated in Table II, as a function of search progress in term of generations. At the first generation, only $\#(S_c) = 1$ and $\#(S_c) = 2$ have the frequency of occurrences 12 and 8, respectively. At the second generation, the frequency of $\#(S_c) = 1$ is decreased but $\#(S_c) = 3$ is appeared, and the frequency of $\#(S_c) = 2$ is increased. While at the third generation the frequency of $\#(S_c) = 3$ become dominant in the population and the high performance schema $\#(S_c) = 5$ is appeared. Eventually, at the fifth generation only the high performance schemata $\#(S_c) = 5$ and $\#(S_c) = 6$ exist in the population.

As we have stated in the previous paragraph, the concept of schemata leads to the quality of the search

Table II The high performance schema structures.

$\#(S_c)$	Schemata	cost value
1	(* * * * * 1)	2000
	(* * * 1 * *)	2400
2	(* * * 0 * 1)	2000
	(* 1 * * * 1)	2600
	(* * 1 * * 1)	3200
3	(* * 1 0 * 1)	3200
	(* * 1 * 1 1)	3700
	(* 1 1 * * 1)	3800
4	(* * 1 0 1 1)	3700
	(* 1 1 0 * 1)	3800
5	(0 * 1 0 1 1)	3700
	(* 1 1 0 0 1)	3800
6	(0 1 1 0 0 1)	3800

$\#(S_c)$ means the number of specific positions contained in the schema.

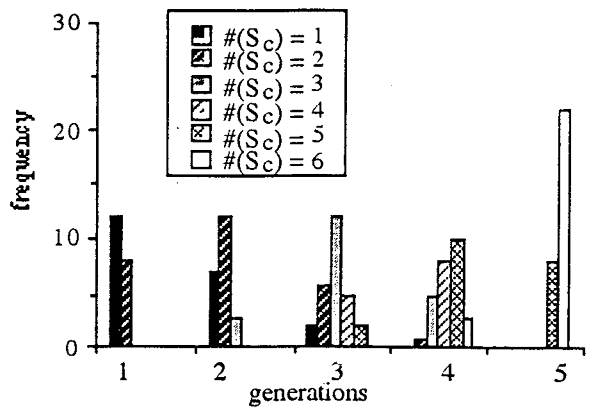


Fig 3. Frequency histogram of the six schemata.

space. The underlying concept of schemata is that schemata are a degenerate description of the entire space resulting the search paths are also degenerated so that the search paths will eventually converge to an optimal or near-optimal point. Fig. 4 shows the percentage of reduction in the search space vs. the number of specific positions contained in the schema. As $\#(S_c) = 1$, the percentage of reduction in the search space is 50%. While $\#(S_c) = 6$, the percentage of reduction in the search space is 100%; this means that the search path is staying at the optimal point X = (0, 1, 1, 0, 0, 1). The choice of the population size is essential to the quality of the final solution as well as the total running time required by the AG algorithm. According to the concept of schemata and the operations of genetic operators, the AG algorithm guarantees an increasing order of performance generations after generations. The larger population size may obtain higher performance than the smaller population size but at the expense of prolonging the execution time. Fig. 5 shows curves of the average cost of the population obtained by the AG algorithm in six generations according to population size are 6, 12, 18, and 24 respectively. The high performance population is appeared in the larger population size. However, all the four population size converge to the optimal solution while at the sixth generation. Fig. 6 shows the total running time required by the AG algorithm in all these cases. The larger population size needs more running time than the smaller ones. From our empirical results the population size should depend on the structure of the underlying problem. Usually, the choice of population size is simply as a constant of the problem size.

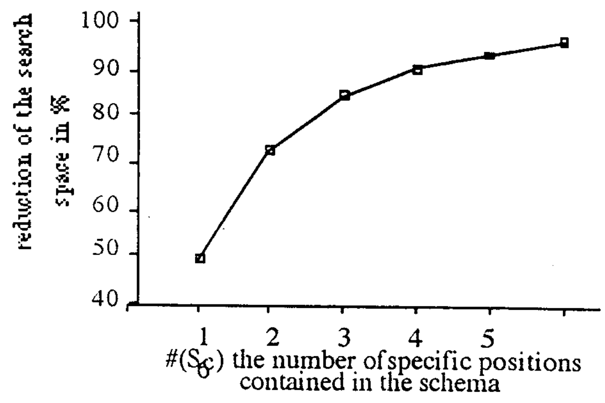


Fig. 4 The percentage of reduction in the search space.

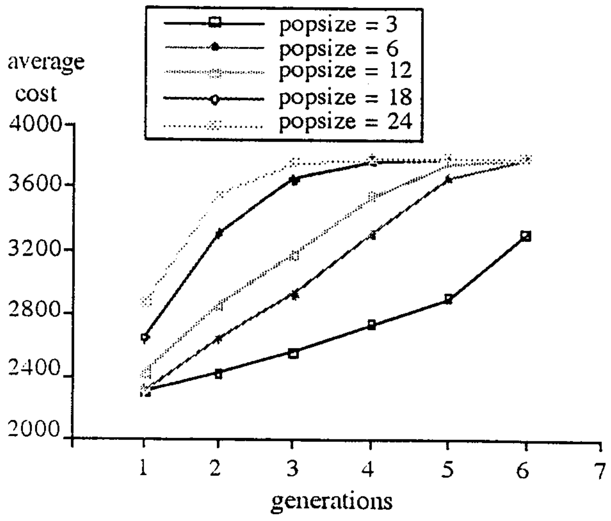


Fig. 5 The average cost of the population in different population sizes.

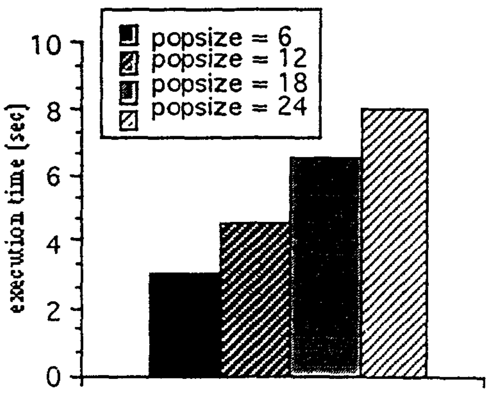


Fig. 6 The comparison of the running time required by the AG algorithm .

4. Convergence of The AG Algorithm

The schemata theory states that the growth ratio of a particular schema S_c is the ratio of the average cost of points belonging to S_c to the average cost of the current population P(k). For the typical maximization problem, it implies that a S_c in P(k) with above the average cost of P(k) will receive an increasing number of points in the next generation. On the other hand, a S_c in P(k) with below the average cost of P(k) will receive a decreasing number of points in the next generation. In the following lemma we show an above average schema S_c will receive exponentially increasing trials in the subsequent generations by the selection function F_s of the AG algorithm. That is, the above average schema will live and the below average will die in the evolution process.

Definition 1: The growing ratio of a schema S_c in the population of the next generation P(k+1) is defined as $\mu_k(S_c) = C'_{avg}(k)/C_{avg}(k)$, where $C_{avg}(k)$ is the average cost of P(k), and $C'_{avg}(k)$ is the average cost of all points belonging to S_c in P(k).

Definition 2: The frequency of a schema S_c appears in P(k) is denoted by $\#(S_c, P(k))$.

Lemma 1 *The selection function F_s allocates exponentially increasing numbers of trials of the above average schema S_c for reproducing the offsprings to the next generation.*

Proof. The propagation of a schema S_c can be written as $\#(S_c, P(k+1)) = \#(S_c, P(k)) * \mu_k(S_c)$. Let the initial population be P(0). Then, we can obtain the following sequence

$P(1) = \#(S_c, P(0)) * \mu_0(S_c)$,
$P(2) = \#(S_c, P(0)) *, \mu_0(S_c)* \mu_1(S_c)$, and finally
$P(k) = \#(S_c, P(0)) * \mu_0(S_c)* \mu_1(S_c)* ... * \mu_{k-1}(S_c)$.
Because of $\mu_k(S_c) = C'_{avg}(k) / C_{avg}(k)$, the above average schema should have $\mu_k(S_c) > 1$ and the below average schema should have $\mu_k(S_c) < 1$. If we let $\mu = \min\{\mu_0(S_c), \mu_1(S_c), ... , \mu_{k-1}(S_c)\}$, then the last equation becomes $P(k) = \#(S_c, P(0)) * \mu^k$. It is obvious that if S_c is an above average schema then it will be allocated exponentially increasing numbers of trials to the subsequent generations. *Q.E.D.*

However, a S_c may be disrupted due to the fact that the genetic-operation stage applies the production function F_p to the population P(k) for producing the offsprings of the next generation. Then, the propagation of the frequency of a schema S_c to the next generation should be written by $\#(S_c, P(k+1)) = \#(S_c, P(k)) * \mu_k(S_c) * (1 - \varepsilon)$, where ε is the probability to be disrupted.

Lemma 2 *The approximated propagation of an above average schema S_c to the next generation is $\#(S_c, P(k+1)) = \#(S_c, P(k)) * \mu_k(S_c) * (1 - \varepsilon)$, where ε is the probability to be disrupted due to the production function F_p. Then, a high performance schema S_c will be propagated exponentially increasing numbers of trials to the next generation in the AG algorithm.*

Proof. By our definitions, the search procedure is a transition function, $F_t(P) = P_q$, for creating the quasi-population of the next generation, and the evolution function, $F_o(P_q) = (F_p(F_s(P_q)) = P$, for creating the population of the next generation. The production function F_p includes crossover, mutation, inversion operations, i.e. F_p = [cro, mut, inv]. Let a point be represented by a parameter vector of the form **A** = $(a_1, a_2, ..., a_n)$ with size n. If θ_{cro} is the probability for crossover, then the probability of S_c to be disrupted due to crossover is given by $\theta_{cro} * (\delta(S_c) / n\text{-}1)$, where n-1 is the metric distance between the first and the last positions of a point. If θ_{mut} is the probability for a mutation, then the probability of S_c to be disrupted due to a mutation is given by $\theta_{mut} * \#(S_c)$. If the probability of a inversion is θ_{inv} and θ_p is the proportion of the population undergoing inversion in a given generation, then the probability of S_c to be disrupted due to a inversion is given by $\theta_{inv} * \theta_p * (\delta(S_c) / n\text{-}1)$. Let $\varepsilon = \theta_{cro} * (\delta(S_c) / n\text{-}1) + \theta_{mut} * \#(S_c) + \theta_{inv} * \theta_p * (\delta(S_c) / n\text{-}1)$ be the probability to be disrupted due to the production function Fp. Then, the approximated growth of an above average schema Sc is obtained by #(Sc, P(k+1)) = #(Sc, P(k)) * μ_k(Sc) * (1 - ε). However, due to the AG algorithm satisfies the monotone restriction, for the high performance schemata, the disruption is minimized and ε will be very small. Therefore, a high performance S_c will be propagated exponentially increasing numbers of trials to the next generation. *Q.E.D.*

Definition 3: The ratio of expected frequencies of S_c propagating from the P(k) to P(k+1) is defined as

$$\Delta_k(S_c) = \frac{\sum_{j \in S_c} C(j)}{C_{avg}(k)} .$$

Then, the above average schema has the larger ratio for propagating than the below average schema and the Δ_k(Sc) is proportional to μ_k(Sc).

Since the crossover is the main operator for propagating high performance schemata to the subsequent generations, one should set θ_{cro} to 1. On the other hand, in order to minimize ε, one should set θ_{mut} and θ_{inv} to a very small value. However, the value of θ_{mut} and θ_{inv} are not critical in the AG algorithm. In fact, one can set θ_{mut} and θ_{inv} to 1 in AG. It is due to the fact that the AG approach adopts the monotone restriction in the genetic-operation stage which rejects all below average offsprings. As a result, only those above average offsprings can survive to the next generation. Thus, ε is a small value in the AG approach. From the equation of lemma 2, #(Sc, P(k+1)) = #(Sc, P(k)) * μ_k(Sc) * (1 - ε), if ε is small enough and $\mu_k(S_c) > 1$, then the high performance Sc will be reproduced more often. However, the exact frequency of S_c appears in the next generation should depend on the selection ratio Δ_k(Sc). Thus, we have the following theorem to prove which stage that the convergence of a high performance schema S_c in the genetic-operation stage of the AG algorithm is O($\log_{\Delta(Sc)*(1 - \varepsilon)}$ N), where N is the population size.

Theorem 1 *The convergence of the genetic-operation stage is O($\log_{\Delta(Sc)*(1 - \varepsilon)}$ N), where N is the population size, Δ(Sc) is the minimum selection ratio of the schema S_c for propagating to the next generation and ε is the probability of disrupting the schemata.*

Proof. From the definition 3 the equation of #(Sc, P(k+1)) = #(Sc, P(k)) * μ_k(Sc) * (1 - ε) should be rewritten as #(Sc, P(k+1)) = #(Sc, P(k)) * Δ_k(Sc) * (1 - ε). This recurrence relation states that
#(Sc, P(1)) = #(Sc, P(0)) * Δ_0(Sc) * (1 - ε),
#(Sc, P(2)) = #(Sc, P(1)) * Δ_1(Sc) * (1 - ε)
= #(Sc, P(0)) * $(1 - \varepsilon)^2$ * Δ_0(Sc) * Δ_1(Sc).
Then, we have #(Sc, P(k)) = #(Sc, P(0)) * $(1 - \varepsilon)^k$ * Δ_0(Sc) * Δ_1(Sc) * ... * Δ_{k-1}(Sc).
Let the frequency of a particular schema Sc in the initial population P(0) be 1, i.e. #(Sc, P(0)) = 1, and let Δ(Sc) = min{Δ_0(Sc), Δ_1(Sc), ..., Δ_{k-1}(Sc)}. The convergence of the evolution is the frequency of the highest performance schema S_c grows to be N, the population size. Therefore, the last equation becomes
N = 1 * $\{\Delta(Sc) * (1 - \varepsilon)\}^k$.
Finally, we obtain k = O($\log_{\Delta(Sc)*(1 - \varepsilon)}$ N).
Q.E.D.

Following the theorem 1, we can prove that the global convergence of the AG algorithm is O($\log_{\Delta(Sc^*)*(1 - \varepsilon)} \alpha^{-1}$ + i), where S_c^* is the schema containing the optimal solution and α is the probability of S_c^* first appears in the *i*th generation, i = 1, 2, ...

Theorem 2 *The global convergence of the AG algorithm is O($\log_{\Delta(Sc^*)*(1 - \varepsilon)} \alpha^{-1}$ + i), where S_c^* is the schema containing the optimal solution, α is the probability of S_c^* first appears in the ith generation and $\Delta(Sc^*)$ is the minimum selection ratio of S_c^* for propagating to the next generation.*

Proof. We assume that the schema S_c^* first appears in a certain generation P(i), $0 \le i < k$, has probability α and N is the population size. The frequency of reproducing S_c^* to the subsequent generation is given by #(Sc*, P(i+1)) = #(Sc*, P(i)) * Δ_i(Sc*) * (1 - ε),
#(Sc*,P(i+2)) = #(Sc*,P(i+1)) * Δ_{i+1}(Sc*) * (1 - ε), ...
Then, at the *k*th generation we obtain
#(Sc*, P(k)) = #(Sc*, P(i)) * $(1 - \varepsilon)^{k-i}$ * Δ_i(Sc*) * Δ_{i+1}(Sc*) * ... * Δ_{k-i}(Sc*).
Since α is the probability of S_c^* first appears at P(i), the frequency of S_c^* in P(i) is #(Sc*, P(i)) = N * α.
Let Δ(Sc*) = min{Δ_i(Sc*) * Δ_{i+1}(Sc*) * ... * Δ_{k-i}(Sc*)}. The convergence condition of the evolution is #(Sc*, P(k)) = N. Thus, the equation becomes N = N * α * $((1 - \varepsilon) * \Delta(Sc^*))^{k-i}$. We obtain k = O($\log_{\Delta(Sc^*)*(1 - \varepsilon)} \alpha^{-1}$ + i). *Q.E.D.*

Theorem 3 *If β is the probability of ever visit the optimal point t_0 by the annealing stage, then the global convergence of the AG algorithm is $O(\log_{\Delta(Sc^*)} * (1 - \varepsilon)^N + 1/N\beta)$, where N is the population size and $\Delta(Sc^*)$ is the minimum selection ratio of the schema S_c^* for propagating to the next generation.*

Proof. We can assume there is one search path ever visit the optimal point t_0 once by the probability $\beta = 1 / |V|$, where $|V|$ is the number of points in the search space. Then, the maximum number of transitions for visiting t_0 is $1/\beta$. That is, t_0 appears at the $1/N\beta$-th generation in the worst case, N is the population size. The equation of theorem 2, $\#(Sc^*, P(k)) = \#(Sc^*, P(i)) * ((1 - \varepsilon) * \Delta_i(Sc^*))^{k-i}$ becomes $N = 1 * (\Delta_i(Sc^*) * (1 - \varepsilon))^{k-i}$ where $i = 1/N\beta$. Finally, we obtain
$O(\log_{\Delta(Sc^*)} * (1 - \varepsilon)^N + 1/N\beta)$. *Q.E.D.*

5. Conclusions

We have shown the global convergence and analysis the running time complexity of the AG algorithm. Goldberg [4] indicates that the connection between GAs and SA have been explored elsewhere [2, 3] but only loosely, and GAs have sometimes been criticized because they lack of convergence proofs, asymptotic or otherwise. Goldberg proposes a selection procedure for GAs called Boltzmann tournament selection and claims that the asymptotic convergence in GAs is guaranteed by Boltzmann distribution and thermal equilibrium of SA without any further proofs [4]. However, the AG algorithm has tighter connection between GAs and SA than what Goldberg proposed. We not only adopt the Metroplois criterion for approaching the Boltzmann distribution but use the annealing schedule for improving the convergence of the algorithm. Although the concept of AG algorithm is equivalent to Goldberg's Boltzmann tournament selection, the implementation of AG is more sophisticated than the Boltzmann tournament selection does. The AG algorithm is used for improving the performance of either SA or GAs and used for solving combinatorial optimization problems. Further work in comparing the AG approach and Goldberg's Boltzmann tournament selection is in progress.

References

[1] Aarts, E., and Korst, J., *Simulated annealing and Boltzmann machines - A stochastic approach o combinatorial optimization and neural computing.* John Wiley and Sons publishing, 1989.

[2] Davis, L., *Genetic algorithms and Simulated annealing*, Los Altos, California: Morgan Kaufmann Publishers, Inc., 1987.

[3] Eiben, A. E., Aarts, E. H. L., Van Hee, K. M., "Global convergence of genetic algorithms: a Markov chain analysis", *Parallel Problem Solving from Nature*, Springer: LNCS # 496, pp.5-12, 1990.

[4] Goldberg, D. E., "A note on Boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing", *Complex Systems* 4, pp.445-460, 1990.

[5] Goldberg, D.E., *Genetic algorithms: In search, Optimization and Machine Learning*, Addison-Wesley Publishing Company, Reading Mass., 1989.

[6] Holland, J. H., *Adaptation in natural and articifial systems*, Ann Arbor: The university of Michigan Press, 1975.

[7] Huang, M. D., Romeo, F., and Sangiovanni-Vincentelli, A., "An efficient general cooling schedule for simulated annealing", *Proceeding of IEEE international Conference on Computer-Aided Design*, Santa Clara, pp. 381-384, 1986.

[8] van Laarhoven, P. J. M., and Aarts, E. H. L. *Simulated annealing : theory and applications*, Dordrecht, Holland: D. Reidel Publishing Company, 1987.

[9] Lin, F. T., Kao, C. Y., and Hsu, C. C., "Incorporating genetic algorithms into simulated annealing", *Proceedings of the 4th International Symposium on Artificial Intelligence*, Cancun, Mexico, pp.290-297, 1991.

[10] Lin, F. T., Kao, C. Y., and Hsu, C. C., "Applying The Genetic Approach to Simulated Annealing in Solving Some NP-hard Problems",IEEE Transactions on Systems, Man, and Cybernetics, Vol. 23, No. 6, Nov/Dec, pp. 1752-1767, 1993.

[11] Lin, F. T, and Kao, C. Y, "The Annealing-Genetic Approach for Solving The Bin-Packing Problems", the Proceedings of International Computer Symposium 1994, Hsinchu, Taiwan, pp. 126-131, 1994.

[12] White, S. R., "Concepts of scale in simulated annealing", *Proceedings of IEEE International Conference on Computer Design*, Port Chester, pp. 645-651, 1984.