Diversity Maintenance Perspective: An Analysis of Exploratory Power and Function Optimization in the Context of Adaptive Genetic Algorithms

Sunanda Gupta and M. L. Garg

Abstract In order to increase the probability of finding optimal solution, GAs must maintain a balance between the exploration and exploitation. Maintaining population diversity not only prevents premature convergence but also provides a better coverage of the search space. Diversity measures are traditionally used to analyze evolutionary algorithms rather than guiding them. This chapter discusses the applicability of updation phase of binary trie coding scheme [BTCS] in introducing as well as maintaining population diversity. Here, the robustness of BTCS is compared with informed hybrid adaptive genetic algorithm (IHAGA), which works by adaptively changing the probabilities of crossover and mutation based on the fitness results of the respective offsprings in the next generation.

Keywords Genetic algorithm · Multidimensional knapsack problem · Diversity maintenance

1 Introduction

Genetic algorithms (GAs) have been successfully applied to various optimization problems where one intends to find an optimum or approximate solution to a problem that has a huge size of solution space. However, one of the major concerns in using evolutionary algorithms to search a complex state space is the problem of premature convergence, especially for combinatorial optimization problems like multidimensional knapsack problem (MKP), where the landscape is multipeaked; the probability of search sticking to local optima is all the more high. Because genetic

S. Gupta (✉) · M. L. Garg
School of Computer Science and Engineering, S.M.V.D.U, Katra, India
e-mail: sunanda.gupta@smvdu.ac.in; gupta.sunanda@gmail.com

M. L. Garg
e-mail: garg.ml@smvdu.ac.in

B. V. Babu et al. (eds.), *Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28–30, 2012*, Advances in Intelligent Systems and Computing 236, DOI: 10.1007/978-81-322-1602-5_4,

programming is highly stochastic, we do not expect to obtain clear rules about exact levels of diversity. We aim to draw general conclusions and "rules of thumb" from the investigation of evolving populations with different measures of diversity. Given a specific landscape structure—defined by the search space, objective function, then relying on problem-specific knowledge for navigating this structure in order to extract helpful information from the search space, would make the optimization faster and more effective. This paper investigates the characteristic issues of BTCS [1] (which incorporates this strategy) and compares it with IHAGA [2] for solving test instances of combinatorial optimization problem on MKP. The simulation results show that the proposed strategy significantly improves the computational efficiency of GAs. The rest of the chapter is organized as follows. In the section that follows, a brief review of the BTCS scheme and the research work going on in the field of using adaptive crossover and mutation operators for achieving diversity is provided. Section 3 describes the BTCS bucket updation phase vis-a-vis population diversity. Experimental results are presented in Sect. 4. Section 5 summarizes the main contributions of the chapter.

2 Related Work

2.1 Binary Trie Coding Scheme

Binary trie coding scheme [BTCS] creates and maintains a diverse population of highly fit individuals capable of adapting quickly to fitness landscape change [1]. BTCS provides three major contributions related to duplicate elimination and premature convergence in a steady-state GA. The first contribution of BTCS is the virtually compressed binary trie structure (VCBT). VCBT when integrated with GA proves to be beneficial in determining duplicates among all the generations and replacing them with unique individuals [1, 3]. The second contribution is to demonstrate that preventing duplicates results in improved performance. It effectively avoids what is usually a difficult trade-off between achieving fast search and sustaining diversity and thereby provides means to avoid premature convergence. The third contribution of BTCS is that it relies on problem-specific knowledge in fragmenting the search space into feasible and infeasible regions and then pruning the infeasible regions. This chapter discusses as to how bucket updation phase of BTCS incorporates the effective measures pertaining to population diversity without using adaptive crossover and mutation operators.

2.2 Adaptive Crossover and Mutation Rate

The significance of crossover operator in controlling GA performance has long been acknowledged in GA research which can be dated back to 1980s [4]. A number

of guidelines exist in the literature for setting the values of crossover probability [5]. Some studies particularly focused on finding optimal crossover rates [6]. These heralded the need for self-adaption of the crossover or mutation rates. In [7], an adaptive genetic algorithm was proposed, in which crossover and mutation probabilities were varied according to the fitness values of the solutions. There were also works on devising adaptive crossover operators instead of varying the crossover rates [8]. Several operators were employed, and the probabilities of applying each operator were adapted according to the performance of the offsprings generated by the operator. Since then, several similar works have also been done [9].

The choice of mutation rate is also critical to GA's performance [10]. Various researchers have come up with novel approaches to implement the adaptive mutation into a GA. Some approaches to adaptive mutation control employ parent fitness in determining mutation probability [11]. If selected, highly fit individuals undergo low levels of mutation (minimal disruption), while low-fitness individuals are subjected to large rates of disruptive mutation. A measure of population diversity is employed by [12] and [13] in adapting mutation probabilities. In a similar vein, Zhang et al. [14] adapt crossover and mutation according to parameters extracted from a K-means clustering algorithm. Thus, many researchers have emphasized on using adaptive mutation so as to improve GA's performance as it facilitates the finding of global optimum more efficiently [15].

Although the adaptive crossover and mutation rates are hot spots in the study of genetic search, the BTCS scheme proposed by us [1] does not explicitly employ any scheme to adaptively mutate or crossover. For analyzing the robustness of BTCS, we compare it with informed hybrid adaptive genetic algorithm (IHAGA). This scheme works by adjusting its cross-adaptive rate and mutation rate according to the situation surrounding the fitness of the individual [2]. In the course of crossover and mutation, the probabilities of crossover and mutation are adjusted adaptively according to the following formulas:

$$Pc = \begin{cases} \dfrac{Pc_{\max} - Pc_{\min}}{1 + \exp\left(Ax\left(\dfrac{2\left(f' - f_{\text{avg}}\right)}{f_{\max} - f_{\text{avg}}} - 1\right)\right)} + Pc_{\min} & f' \geq f_{\text{avg}} \\ Pc_{\min} & f' \leq f_{\text{avg}} \end{cases}$$

$$Pm = \begin{cases} \dfrac{Pm_{\max} - Pm_{\min}}{1 + \exp\left(Ax\left(\dfrac{2\left(f' - f_{\text{avg}}\right)}{f_{\max} - f_{\text{avg}}} - 1\right)\right)} + Pm_{\min} & f' \geq f_{\text{avg}} \\ Pm_{\min} & f' \leq f_{\text{avg}} \end{cases}$$

where $Pc_{\max}$ and $Pc_{\min}$ denote the lower limit and the upper limit of probability of crossover, respectively. $f_{\max}$ and f_{avg} denote the maximal fitness and the average fitness of population, respectively, f' denotes the higher fitness of two crossover individuals, f denotes the fitness of the individuals, and $A = 9.903438$ [2].

3 BTCS Bucket Updation Phase

3.1 Buckets and Their Significance

The buckets correspond to the leaf nodes in a VCBT [1, 3]. The aim of buckets is to maintain a continuous presence on as many peaks as possible. Population's spatial information is obtained with computationally inexpensive buckets. It provides important information in addition to the address of the trie structure existing under it. This information is used to identify potentially local convergence. Buckets are significant in dividing the population into an exploration section and exploitation section. It monitors and measures diversity at synchronized time intervals and accordingly attempts to control or promote diversity during the evolution. Identifying such measures allows better prediction for run performance and improved understanding of the population and enables the design of efficient operators.

3.2 Bucket Updation

The contribution of updation phase in the BTCS scheme is twofold [1, 3]. The first contribution of updation phase is to manage the size of VCBT structure. The size of VCBT structure can be kept small by pruning fully explored regions of the search space. The second contribution is to monitor convergence and introduce diversity so as to avoid local entrapment.

3.2.1 Guided Crossover Operator

The proposed procedure works by randomly selecting buckets with criterion value 1, and then exchanges information by copying their best strings [1, 3]. The copying of best feasible boundary solution of one to another is done only if (*new bucket_sum* + *old bucket_best_sum*) are feasible and *new bucket_sum* is greater than *old bucket_sum*. Doing this restricts the copying of strings between any two selected buckets randomly. Bucket_sum and bucket_best_sum are two variables that are unique to each bucket. They are used for storing the sum of included objects from root to bucket_position and sum of bucket + 1 position till n (the number of objects), respectively. GA with the proposed method distributes the individuals more widely compared to simple GA, where the individuals represent the local optima for that region. The aim is to identify feasible regions in the landscape that could replace less fit individuals by more promising samples from the unexplored sections of the search space. This prevents entrapment in local optima by including new individuals from the unexplored regions of the search space.

Table 1 Average execution time of BTCS in comparison with IHAGA

n	m	α	Simple GA		BTCS_IMO		IHAGA	
			A.B.S.T	A.E.T	A.B.S.T	A.E.T	A.B.S.T	A.E.T
100	5	0.25	9.6	345.9	10.85	109.47	8.14	**31.13**
		0.5	23.5	347.3	26.32	120.23	19.92	**76.41**
		0.75	26.9	361.7	33.36	123.02	23.19	**90.43**
	10	0.25	97.5	384.1	104.20	**115.12**	83.33	192.05
		0.5	97.3	418.9	111.90	143.6	84.61	**129.86**
		0.75	16.8	462.6	19.15	159.56	14.36	**129.53**
	30	0.25	177.4	604.5	198.69	202.68	150.00	**199.49**
		0.5	118	782.1	130.98	247.74	113.90	**218.99**
		0.75	90	904.2	80.10	315.34	80.10	**253.18**
250	5	0.25	50.7	682	34.19	**216.03**	34.48	265.98
		0.5	276.7	709.4	191.59	**257.54**	185.39	333.42
		0.75	195.9	763.3	127.12	**241.78**	137.13	534.31
	10	0.25	359	870.9	258.16	**290.65**	290.43	566.09
		0.5	342.2	931.5	249.91	**295.06**	281.63	596.16
		0.75	129.1	1011.2	95.91	**320.3**	104.44	455.04
	30	0.25	582.9	1499.5	332.51	**493.45**	472.73	794.74
		0.5	901.5	1980	601.20	**643.14**	720.30	1207.80
		0.75	1059.3	2441.4	754.22	**815.23**	840.02	1440.43
500	5	0.5	291.3	1345.9	142.10	**416.09**	236.83	969.05
		0.75	386.2	1412.6	188.40	**499.32**	317.84	974.69
	10	0.5	562.2	1728.8	274.20	**615.35**	490.24	1383.04
		0.75	937.6	1931.7	457.40	**715.34**	792.27	1564.68
	30	0.5	1121.6	3198.9	547.10	**1334.56**	923.08	2600.71
		0.75	1903.3	3888.2	928.40	**1231.49**	1545.48	3110.56

3.2.2 Adaptive Selection Parameter Control

This takes place when there is some form of feedback from the search that serves as input to the mechanism used to determine the change in the strategy parameters. During this phase, the *avg* corresponding to the worst and best individuals within that bucket is checked. It is computed as the average of all the individuals within that bucket. The *new avg* value will drop if more boundary solutions between the worst and average interval are generated and would increase if more boundary solutions between average and best interval are generated. During this phase, that bucket is selected, whose *new avg* has increased and at least approximately more than 60 % of the region within that bucket has already been explored. The aim of phase 2 in bucket updation is to prevent the unnecessary delay caused in exploring those regions of search space where the probability of best solution to exist is very limited. The phase 2 describes the buckets' solution space diversity from a fitness perspective, i.e., a measure of diversity of healthy individuals. *BDS Bucket Updation* employs ASPC

Table 2 Percentage gaps for BTCS and IHAGA

n	m	α	GA	BTCS_IMO	IHAGA
100	5	0.5	0.4564	0.4613	0.46200
		0.75	0.3212	0.2884	0.32119
	10	0.5	0.7982	0.7774	0.79838
		0.75	0.4813	0.4697	0.48100
	30	0.5	1.3457	1.3145	1.36953
		0.75	0.8321	0.8296	0.81546
250	5	0.5	0.1253	0.1183	0.12525
		0.75	0.0811	0.0752	0.08759
	10	0.5	0.2543	0.2362	0.25429
		0.75	0.1572	0.1513	0.15710
	30	0.5	0.5321	0.5267	0.54877
		0.75	0.3112	0.2972	0.32431
500	5	0.5	0.0441	0.0443	0.04631
		0.75	0.0378	0.0429	0.04271
	10	0.5	0.1134	0.0946	0.11907
		0.75	0.0712	0.0501	0.07903
	30	0.5	0.2635	0.2387	0.26908
		0.75	0.1747	0.1738	0.17905

to regulate selection pressure. ASPC's objective is to create a diversity of health in the population, i.e., the diversity of high-fitness individuals.

4 Experimental Results

Tables 1 and 2 illustrate the comparison of results of BTCS with those of IHAGA [2] for solving the MKP. The results of BTCS and IHAGA are based on our own computations. Table 1 provides the average best solution time (ABST) and the average execution time (AET) for both BTCS and IHAGA. The bold highlights in Table 1 show the optimal average execution time among the two for varying n and m values, where n is the number of objects and m is the number of constraints. It is clear from Table 1 that IHAGA outperforms BTCS computationally, for smaller values of n. The cost of constructing VCBT results in an increase in the average execution time. However, for larger instances, despite the time utilized in the construction of VCBT structure, BTCS is effective in reaching the optimal solution in comparison with IHAGA. The ability to work with unique boundary individuals facilitates faster convergence. The probability of recurrence of individuals in the subsequent generations results in deviation from path, leading to optimality, for larger set of individuals in the case of IHAGA despite its ability to guide the search. Table 2 further provides the average percentage gaps for the two approaches. For both, the BTCS and IHAGA, the percentage gaps (100 $\times$ (optimum LP – optimum GA)/optimum LP) relative to

the solution of LP relaxation were computed. Here GA refers to special cases of GA, i.e., BTCS and IHAGA. It can be inferred from the results of Table 2 that BTCS outperforms IHAGA for all test instances under consideration. BTCS has provided better coverage of the search space and has been found to be successful in providing solutions of better quality in comparison with IHAGA.

5 Conclusion

The aim of updation phase in the BTCS scheme has been the exploring of promising regions while concentrating the search on hyperplanes that are likely to contain good solutions. The GCO and ASPC focus on extracting information about the selected buckets before deciding on the introduction of diversity. Its advantage is that at the point of near convergence, late in a GA run, such diversion reduces the probability of GA to entrap in local convergence and thus provides better solutions.

In our approach, we have not used a mutation parameter, which should be adapted explicitly. Instead, it is the principle of working with unique chromosomes (or individuals) in the VCBT structure, which guarantees automatic mutation. Furthermore, our approach still concentrates on using one-point crossover operator with a fixed probability of 0.70. This is attributed to the deeper nature of BTCS scheme, which permits only good optimal solutions from the search space to participate in the process of evolution. Working with unique boundary solutions assists in maintaining an optimum level of diversity among the individuals.

References

1. Sunanda, Garg M.L.: Binary trie coding scheme—An intelligent genetic algorithm avoiding premature convergence. Int. J. Comput. Math. Taylor & Francis. (2012). doi:10.1080/00207160.2012.742514.
2. Yanqin, M., Jianchen W.: Improved hybrid adaptive genetic algorithm for solving knapsack problem. In: Proceedings of the 2nd International Conference and Information Processing, pp. 644–647 IEEE, (2011)
3. Sunanda, Garg, M.L.: GA implementation of the multi dimensional knapsack problem using compressed binary tries. Advances in computational research (ISSN: 0975–3273), 43–46 (2009)
4. Goldberg, D.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley Publishing Company Inc, Massachusetts (1989)
5. Schaffer, J.D., Carvana, R.A., Eshelman, L.J.: R. A study of control parameters affecting online performance of genetic algorithms for function optimization. In: Proceedings of the Third International Conference on Genetic Algorithms, Das (1989)
6. Ochoa, G., Harvey, I., Buxton, H.: On recombination and optimal mutation rates. In: Proceedings of Genetic and Evolutionary Computation Conference, pp. 488–495 (1999)
7. Srinivas, M., Parnaik, L.: Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Trans. Syst. Man. Cybern. **3**, 1841–1844 (2003)

8. Vekaria, K., Clark, C.: Biases introduced by adaptive recombination operators. In: Proceedings of Genetic and Evolutionary Computation Conference, 670–677 (1999)
9. Ono, I., Kita, H., Kobayashi, S.: A robust real coded genetic algorithm using unimodal normal distribution crossover augmented by uniform crossover: Effects of self-adaptation of crossover probabilities. In: Proceedings of Genetic and Evolutionary Computation Conference, pp. 496–503 (1999)
10. Cervantes, J., Stephens, C.R.: Limitations of existing mutation rate heuristics and how a rank GA overcomes them. IEEE Trans. evol. comput. **13**(2), 369–397 (2009)
11. Liu, D., Feng, S.: A novel adaptive genetic algorithms. Proc. Int. Conf. Mach. Learn. Cybern. **1**, 414–416 (2004)
12. Zhu, K.: A diversity controlling adaptive genetic algorithm for the vehicle routing Problem with time windows. In: Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence, pp. 176–183 (2003)
13. Hagras, H., Pounds-Cornish, A., Cooley, M., Callaghan, V., Clarke, G.: Evolving spiking neural network controllers for autonomous robots. In: Proceedings IEEE International Conference on Robotics and Automation, vol. 5, pp. 4620–4626 (2004)
14. Zhang, J., Chung, H., Lo, W., et al.: Clustering based adaptive crossover and mutation probabilities for genetic algorithms. IEEE Trans. Evol. Comput. **11**(3), 326–335 (Jun. 2007)
15. Lobo, F. G., Lima, C. F., Michalewicz, Z (eds.): Parameter setting in evolutionary algorithms. volume 54 of Studies in Computational Intelligence Springer, (2007)