Generic Heuristics for Combinatorial Optimization Problems

Michael Affenzeller, Rene Mayrhofer

Institute of Systems Science
Systems Theory and Information Technology
Johannes Kepler University
Altenbergerstrasse 69
A-4040 Linz - Austria
email: ma@cast.uni-linz.ac.at
phone: ++43 732 2468 9200
fax: ++43 732 2468 8893

Keywords: combinatorial optimization, heuristic, evolutionary computation, hybrid models.

Abstract. This paper discusses the use of several heuristic techniques for problems of combinatorial optimization. We especially consider the advantages and disadvantages of naturally inspired generic techniques like Simulated Annealing, Evolution Strategies, or Genetic Algorithms. This reflection gives a quite intuitive motivation for hybrid approaches that aim to combine advantageous aspects of the certain strategies. Among those we formulate some new hybrid multidisciplinary ideas that are mainly based upon Genetic Algorithms and Evolution Strategies. These algorithms aim to improve the global solution quality by retarding the effects of unwanted premature convergence. The experimental part of the paper gives a brief overview of achieved results.

1 Introduction

Heuristic techniques that mimic natural processes, developed over the last thirty years and have produced 'good' results in reasonable short runs for combinatorial optimization problems. Even though those bionic heuristics are much more flexible regarding modifications in the problem description when being compared to classical problem specific heuristics they are often superior in their results.
Those bionic heuristics have been developed following the principles of natural processes: In that sense, Genetic Algorithms (GAs) try to imitate the biological evolution of a species in order to achieve an almost optimal state whereas Simulated Annealing (SA) was initially inspired by the laws of thermodynamics in order to cool down a certain matter to its lowest energetic state.

During the last decades plenty of work has been investigated in order to introduce new coding standards and operators especially for Genetic Algorithms. Almost all of these approaches have one thing in common: They are quite problem specific and often they do not challenge the basic principle of Genetic Algorithms. Considering the advantages and disadvantages of certain heuristic methods in order to combine their favorable attributes in a generic or problem specific way leads to a generic respectively a problem specific hybrid heuristic. In the following we will exemplarily consider the main aspects when designing a hybrid heuristic method. Furthermore, we propose a new approach by looking upon the concepts of a Standard Genetic Algorithm as an artificial self organizing process in order to overcome some of the fundamental problems Genetic Algorithms are concerned with in almost all areas of application.

2 Optimization Techniques

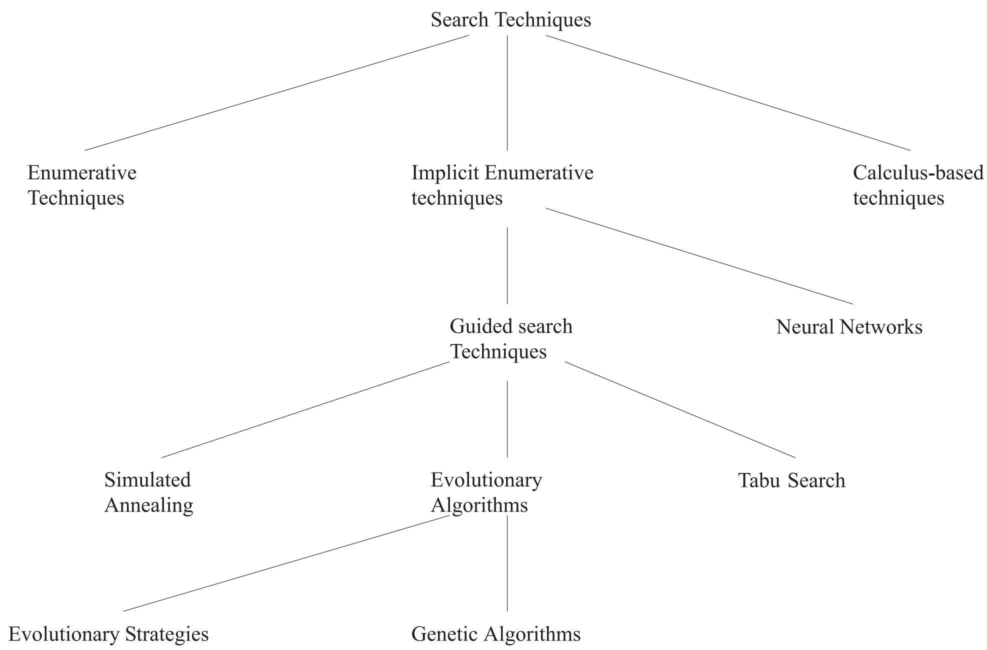

Fig. 1. Search Techniques for Optimization.

Fig. 1 shows the most popular search techniques for optimization in the context of the present paper.

For many practical combinatorial optimization problems there are potentially a number of global optima and many more locally optimal solutions. The introduction of a neighborhood concept allows a systematic approach for the distinction between local and global optima. In a simple neighborhood search each solution has an associated set of neighbors which can be reached directly by an operation respectively by a set of operations. The neighborhood also needs to be produced. In the general case the designer of the heuristics determines the appropriate neighborhood. Furthermore, it is usually assumed that the neighborhood is symmetric.It is obvious that this approach tends to find a local, rather than a global, optimal solution and the number of solutions to be evaluated can be very large. However, it should be mentioned that there exist more complicated versions of neighborhood searches to escape from local optima and to reduce the number of neighbor solutions that are admissible and therefore need to be evaluated. One of the most popular optimization search methods derived from neighborhood search is Simulated Annealing. Simulated Annealing introduces chance into its climb to avoid the traps of local optima. Compared to neighborhood search, identical random start and mutation are applied; but the best evaluation is not always selected. The probability of choosing a new solution is a function of both old and new evaluations and an additional parameter T known as the temperature. Initially the temperature is high and so is the probability of selecting a new solution independent of its evaluation. This continues until the system reaches some sort of balance with little more progress being made. The temperature is lowered and it is allowed to settle again closer to the optimum peak. At a temperature of zero the system acts like a hill climber to find the local and desired global optimum.

Evolutionary Algorithms are search and optimization procedures that find their origin and their afflatus in the biological world. The Darwinian theory of evolution with the survival of the fittest in a changing environment is generally accepted. Evolutionary Algorithms try to abstract and mimic some of the features of natural evolution in order to treat problems that require adaptation, search, and optimization.
Evolutionary Algorithm is a general term effectively including a number of related but not identical methodologies that all exploit ideas from natural evolution and selection. Genetic Algorithms, Evolution Strategies, and Evolutionary Programming are the prominent approaches with hybrid techniques rapidly coming into play.
Evolutionary computing offers many possibilities for parallel and distributed execution because many steps are independent. In fact, if the natural model is to be followed, evolutionary algorithms are parallel on the first place since evolution takes place with individuals acting simultaneously in spatially extended domains. Therefore, a sequential execution setting appears as an unnecessary constraint.

Both attempts, Evolution Strategies as well as Genetic Algorithms, work with a population-model whereby the genetic information of each individual of a population is different in general. Among other things this genotype contains the parameter vector which contains all necessary information about the fitness of a certain individual. Before the intrinsic evolutionary process takes place, the initial population is initialized arbitrarily. Evolution, i.e. replacement of the old generation by a new generation, proceeds until a certain termination criterion is fulfilled.

The major differences between Evolution Strategies and Genetic Algorithms lie in the form of the genotype, the calculation of the fitness and the operators (mutation, recombination, selection). In contrast to Genetic Algorithms where the main-role of the mutation operator is simply to avoid stagnation, mutation is the main operator of Evolution Strategies. As a further difference between the two major representatives of Evolutionary Computation, selection in case of Evolution strategies is absolutely deterministic which is not the case in the context of Genetic Algorithms or in nature. Therefore, in case of Evolution Strategies, arbitrary small differences in fitness can decide about the survival of an individual. The mapping between a bit-string and the real numbers often is a problem when applying Genetic Algorithms whereas Evolution Strategies resign in copying nature that exactly.

3 Hybrid Optimization Techniques

The combination of certain aspects of different generic heuristics and/or problem-specific heuristics is a newer field of research which becomes more and more important and feasible due to increasing computational power. Tabu-Search[1], Simulated Annealing, and Genetic Algorithms are variants of a general search technique in that sense that those techniques do not use problem specific information. Thus, in order to implement a search, generic decisions that govern the operations of the algorithm, and problem specific decisions that govern how a particular problem is modeled to fit into the search framework, need to be made. This observation allows consideration of hybridization of the techniques.
Hybridization allows searches that display particular properties to be produced. It may for instance be desirable to have an element of memory in a Simulated Annealing approach, using Tabu methods. Also it may be that parallelization is desired for Simulated Annealing which suggests a combination with Genetic Algorithms. Similarly many Simulated Annealing algorithms have prohibited moves. For instance, in the allocation problem some tasks may be restricted to a subset of possible processors in order to connect to peripherals. A self-evident hybridization could be formalized by adding Tabu-Search mechanisms to the Simulated Annealing algorithm. More importantly Tabu Search allows expert knowledge to be incorporated into the blind search techniques; thus it becomes possible for the designers to bring in their experience.
Of particular interest is the analysis of the theoretical convergence rates of hybrid algorithms. A key concept in such a theory is likely to be given by Markov Chains [14].

Simulated Annealing (SA) is one of the easiest techniques considered in this paper to understand and to implement. In Simulated Annealing the generic decisions mainly relate to controlling the temperature parameter and the process by which it is changed. The problem specific decisions relate to the solution space, the neighborhood structure, the cost function, and the initial solution. All of these elements are open to hybridization.
Sampling of the neighborhood in Simulated Annealing is usually assumed to be uniform random sampling with replacement. Tabu Search can be used to restrict the neighborhood being considered as the temperature falls. For instance, if a penalty function element that is positive if a threshold is exceeded, is used, it may be more efficient to restrict moves to those that involve variables with positive penalty values. Hybrids can be realized with tabu structures that allow such restrictions to be implemented.
The cost function is also open to hybridization. Simulation can be used to produce a strategy to incorporate the use of estimates of the cost function for a proportion of moves. In constraint satisfaction, such as used in the allocation search to allow for limited resources, both a penalty function to penalize the breaking of constraints, and a selection process that does not attempt most moves which break constraints, can be employed. Attempts have been made to combine the acceptance criteria for Simulated Annealing with population based approaches, including:

- Use an approximation

[1] A detailed survey of Tabu Search is given in [6] or [13].

– Use two different functions in parallel
– Solve the problem iteratively
– Solve in phases
– Combine with features from other meta-heuristics, such as Tabu Search.

A number of differences can be identified between Simulated Annealing(SA) and Tabu Search(TS). First, Tabu Search emphasizes scouting successive neighborhoods to identify high quality moves, while Simulated Annealing randomly samples from neighborhoods. Second, Tabu Search evaluates the relative attractiveness of moves in relation to both the fitness function and influence factors. Finally, Tabu Search relies on guiding the search by use of multiple thresholds, Simulated Annealing uses only one (temperature). Hybrids that allow temperature to be strategically manipulated in Simulated Annealing have been shown to produce improved performance.
A typical approach to integrate SA and TS is proposed by Fox [5] where the SA moves are no more blind as TS is incorporated in order to integrate 'intelligent' moves.

A number of authors have attempted to integrate the Simulated Annealing(SA) and Genetic Algorithm(GA) approaches. On the one hand the SA-community has borrowed elements from Genetic Algorithms in an attempt to introduce some parallelism into the concept of Simulated Annealing whereas on the other hand the GA-community uses SA concepts as a randomising element. In this section two approaches, GESA and PRSA, are considered. Other work in this field has been produced by Lin [10].

Methods to overcome problems associated with Genetic Algorithms are often conflicting and a compromise is usually required. Attempts have been made to use problem-specific knowledge to direct the search. The hybrids have often produced incorporate elements of other neighborhood search techniques. A number of theoretical objections have been raised about hybrid Genetic Algorithms. However, in practice, hybrid Genetic Algorithms do well at optimization tasks.
HGATA (Hybrid GA for Task Allocation) [11] is a hybrid between Genetic Algorithms and Hill Climbing for the task allocation problem in parallel computing. The search is undertaken in three stages: a clustering stage, a calculation-balancing stage, and a tuning stage when the population is near convergence. The primary aspect of hybridization in this context is to introduce a simple problem-specific hill-climbing procedure that can increase the fitness of individuals.
GENOCOP (GENetic Algorithm for Numerical Optimization for COnstrained Problems) [12] considers three approaches to constrained problems. These approaches use penalty functions, decoders/repairers, and modified GA data structures, respectively. The preferred GENOCOP approach is to eliminate the equalities present in a set of constraints and to design special genetic operators to keep all chromosomes within the constrained solution space. This can be done efficiently for linear constraints. The bounds produced are dynamic in the sense that they depend on the values of other variables of the current solution. Anyway, a newer area of theoretical GA research is concerned with new bionic aspects of expanding the theoretical concepts of Genetic Algorithms in order to increase their flexibility and to avoid premature convergence as discussed in the following section.

4 New Hybrid Approaches

Applied to problems of Combinatorial Optimization, Evolution Strategies tend to find local optima quite efficiently. But in the case of multimodal test functions, global optima can only be detected by Evolution Strategies if one of the start values is located in the narrower range of a global optimum. Nevertheless, the concept how Evolution Strategies handle the selective pressure has turned out to be very useful for the new GA and its newly introduced variants as presented within the scope of the present paper.

This handling of selective pressure in our context is mainly motivated by (μ, λ)-Evolution Strategies where μ parents produce λ descendants from which the best μ survive. Within the framework of Evolution Strategies, selective pressure is defined as $s = \frac{\mu}{\lambda}$, where a small value of s indicates high selective pressure and vice versa (for a detailed description of Evolution Strategies see for instance [17]). Even if the interaction between the variable selective pressure within our new model and the notion of temperature within the scope of Simulated Annealing [9] is quite different in detail, we have adopted this notation. Applied to our new Genetic Algorithm, this means that from $|POP|$ (population size) number of parents $|POP| \cdot T$ ((size of virtual population) $> |POP|$, i.e. $T > 1$) descendants are generated by crossover and mutation from which the best $|POP|$ survive. Therefore, within our model selective pressure is defined by $s = \frac{1}{T}$. A detailed description of this concept is given in [3] and [1].

The variable selective pressure handling will substantially be needed when segregating and reuniting the entire population in order to avoid or at least to retard premature convergence for achieving better results in terms of global convergence behaviour. In this sense the term 'segregation' is the namegiving element for this newly developed variant of a Genetic Algorithm which we have called Segregative Genetic Algorithm (SEGA). Whereas Island Models (e.g. in [20]) for Genetic Algorithms are mainly driven by the idea of using simultaneous computer systems, SEGA attempts to utilize migration between the subpopulations more precisely in order to achieve superior results in terms of global convergence.

The aim of dividing the whole population into a certain number of subpopulations (segregation) that grow together in case of stagnating fitness within those subpopulations is to combat premature convergence which is the source of GA-difficulties. The principle idea of SEGA is to divide the whole population into a certain number of subpopulations at the beginning of the evolutionary process. These subpopulations evolve independently from each other until the fitness increase stagnates because of too many similar individuals within the subpopulations (premature convergence). Then a reunification from n to $(n-1)$ subpopulations is done. Roughly spoken this means, that there is a certain number of villages at the beginning of the evolutionary process that are slowly growing together resulting in bigger cities, ending up with one big town containing the whole population at the end of evolution. By this approach of width-search, building blocks in different regions of the search space are evolved at the beginning and during the evolutionary process which would disappear early in case of standard Genetic Algorithms and whose genetic information could not be provided at a later date of evolution when the search for global optima is of paramount importance. In this context the above mentioned variable selective pressure is especially important at the time of joining some residents of a village to a neighboring village in order to steer the genetic diversity. A more detailed description of the SEGA concept is given in [2] or [4].

The second newly introduced concept allows the dynamic usage of multiple crossover operators in parallel in order to somehow imitate the parallel evolution of a variety of species that are struggling for limited resources in a common living space. In contrast to natural evolution, it seems reasonable to allow also recombination of individuals that have emerged from different crossover operators, i.e. the total population (or the total subpopulation in case of SEGA) is taken into account for each crossover operator and the living space (habitat) of each virtual (sub)population is defined by its success during the last couple of iterations. Exemplarily considering the properties of the OX (order crossover) and the ERX (edge recombination crossover) operators for crossover it is reported (e.g. in [13]) that the OX-operator significantly outperforms the ERX-operator in terms of speed whereas the ERX-operator surpasses OX in terms of global convergence. Dynamically using multiple crossover operators in parallel utilizes the 'fast' OX-operator for a long evolution period until the performance in terms of solution quality of ERX outperforms OX at a later stage of evolution. Even more, experiments have shown that the described strategy significantly surpasses the results obtained when just using one single operator in terms of solution quality. This dynamic (self-organizing) strategy seems very adopted for problems which consider more than one crossover operator - especially if the properties of the considered operators may change as evolution proceeds.

As an important property of all the newly introduced hybrids it should be pointed out that under special parameter settings the corresponding GA/GAs is/are unrestrictedly included in the new hybrids.

5 Results

The new hybrids have intensely been tested on a variety of benchmark routing problems. In doing so we have tested the new SEGA algorithm and its derivatives on a selection of symmetric and asymmetric benchmark problems taken from the TSPLIB [16] using updated results[2]. Also we have performed a comparison of our new results with a GA using exactly the same operators for crossover and mutation as well as the same parameter settings. Furthermore we have compared our results with the results of the COSA-algorithm [19], an established and successful ambassador of a heuristic especially developed for routing problems. A detailed report of the experiments can be found in [1]. A brief overview is given in table 1 which shows the experimental results of SEGA (with dynamic habitat adaptation), COSA, and GA concerning various types of problems in the TSPLIB. For each problem the algorithms were run three times. The efficiency for each algorithm is quantified in terms of the relative difference of the best's individual fitness after a given number or iterations to the best or best-known solution. In this experiment, the relative difference is defined as relativeDifference $= (\frac{Fitness}{Optimal} - 1) * 100\%$.

The specific parameter settings of the GA with a variable selective pressure (temperature, cooling factor α) and the specific parameter settings of SEGA (temperature, cooling factor α, number of villages, dates of reunification) have been done by

[2] Updates for the best (known) solutions can for example be found on ftp://ftp.zib.de/pub/Packages/mp-testdata/tsp/tsplib/index.html

Table 1. Experimental results of COSA, GA (using OX or ERX for crossover) and the new SEGA together with a dynamic combination of OX- and ERX crossover.

Problem	Iter.No.	Average difference(%)			
		COSA	GA_{OX}	GA_{ERX}	$SEGA_{var.}$
eil76(symm.)	5000	6.36	17.56	7.62	0.32
ch130(symm.)	5000	14.76	84.54	32.44	0.35
kroA150(symm.)	5000	20.91	102.40	71.97	0.74
kroA200(symm.)	10000	48.45	95.69	117.11	1.24
br17(asymm.)	200	0.00	0.00	0.00	0.00
ftv55(asymm.)	5000	44.22	41.34	23.52	0.27
kro124p(asymm.)	10000	26.78	30.61	15.49	0.48
ftv170(asymm.)	15000	187.34	87.12	126.22	1.09

means of testing; nevertheless further parameter tuning should be possible. Recent experimental research is done on several problems of combinatorial optimization like bin-packing, timetabling, or multiprocessor scheduling. Also for those applications it is observable that the new SEGA algorithm significantly outperforms the GA in case of 'hard' problem instances, i.e. huge problem instances that evoke premature convergence when the standard GA is applied.

6 Conclusion

Based on an exemplary overview of optimization techniques we have proposed new combinations of well-known and established heuristics in order to combine their favorable attributes in a systematic and generic way. In doing so we have formulated an enhanced Genetic Algorithm and two upgrades. The newly proposed GA-based techniques couple aspects from Evolution Strategies (selective pressure), Simulated Annealing (temperature, cooling) as well as a special segregation and reunification strategy with crossover, mutation, and selection in a general way, so that established crossover and mutation operators for certain problems may be used analogously to the corresponding Genetic Algorithm. The investigations in this paper have mainly focused on the avoidance of premature convergence and on the introduction of methods which make the algorithm more open for scalability in the sense of convergence versus running time. Concerning the speed of SEGA, it has to be pointed out that the superior performance concerning convergence requires a higher running time, mainly because of the the greater population size $|POP|$ required. This should allow to transfer already developed GA-concepts to increasingly powerful computer systems in order to achieve better results. Using simultaneous computers seems especially suited to increase the performance of SEGA. Anyway, under special parameter settings the corresponding Genetic Algorithm is fully included within the introduced concepts achieving a performance only marginally worse than the performance of the equivalent Genetic Algorithm. In other words, the introduced models can be interpreted as a superstructure to the GA model or as a technique upwards compatible to Genetic Algorithms. Therefore, an implementation of the new algorithm(s) for a certain problem should be quite easy to do, presumed that the corresponding Genetic Algorithm (coding, operators) is known.

However, the efficiency of a variable selective pressure certainly depends on the genetic diversity of the entire population. Ongoing research indicates that it should be a very fruitful approach to define the actual selective pressure depending on the actual genetic diversity of the population.

References

1. Affenzeller, M.: New Hybrid Variants of Genetic Algorithms: Theoretical and Practical Aspects. PhD thesis, Department of System Science, University of Linz (2001)
2. Affenzeller, M.: A New Approach to Evolutionary Computation: Segregative Genetic Algorithms (SEGA). Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence, Lecture Notes of Computer Science 2084 (2001) 594–601
3. Affenzeller, M.: Transferring the Concept of Selective Pressure from Evolutionary Strategies to Genetic Algorithms. Proceedings of the 14th International Conference on Systems Science 2 (2001) 346–353
4. Affenzeller, M.: Segregative Genetic Algorithms (SEGA): A Hybrid Superstructure Upwards Compatible to Genetic Algorithms for Retarding Premature Convergence. Internatinal Journal of Computers, Systems and Signals (IJCSS), Vol. 2, Nr. 1 (2001) 18–32
5. Fox, B.: Integrating and Accelerating Tabu Search, Simualted Annealing and Genetic Algorithms. Annals of Operations Research 41 (1993) 47–67
6. Glower, F., Laguna, M.: Tabu Search. Kluwer Academic Publishers (1997)
7. Goldberg, D. E.: Genetic Alogorithms in Search, Optimization and Machine Learning. Addison Wesley Longman (1989)
8. Holland, J. H.: Adaption in Natural and Artificial Systems. 1st MIT Press ed. (1992)
9. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by Simulated Annealing. Science 220 (1983) 671–680
10. Lin, F., Kao, C., Hsu, C.: Applying the Genetic Algorithm to Simulated Annealing in Solving NP-Hard Problems. IEEE Trans. on Systems, Man, and Cybernetics 23(6) (1993) 1752–1767
11. Mansour, N., Fox, G.: A Hybrid genetic Algorithm for Task Allocation in Multiprocessors. Proceedings of Fourth International Conference on Genetic Algorithms (1991) 466–473
12. Michalewicz, Z., Janikow C. Z.: Handling Constraints in Genetic Algorithms. Proceedings of Fourth International Conference on Genetic Algorithms (1991) 151–157
13. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (1996)
14. Nix, A., Vose M.: Modelling Genetic Algorithms with Markov Chains. Annals of Mathematics and Artificial Intelligence (1992) 79–88
15. Rechenberg, I.: Evolutionsstrategie. Friedrich Frommann Verlag (1973)
16. Reinelt, G.: TSPLIB - A Traveling Salesman Problem Library. ORSA Journal on Computing 3 (1991) 376-384
17. Schneburg, E., Heinzmann, F., Feddersen, S.: Genetische Algorithmen und Evolutionsstrategien. Addison-Wesley (1994)
18. Srinivas, M., Patnaik, L.: Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms . IEEE Transactions on Systems, Man, and Cybernetics 24(4) (1994) 656–667
19. Wendt, O.: Tourenplanung durch Einsatz naturanaloger Verfahren. Deutscher Universitätsverlag (1995)
20. Whitley, D.: A Genetic Algorithm Tutorial. Statistics and Computing 4 (1994) 65–85