Introduction to Hybrid algorithms

Ashish
Department of Information Technology
Dronacharya College Of Engineering,
Farrukhnagar, Gurgaon, India

Email- ashishposwal2001@gmail.com

Ashwani Kumar
Faculty of Computer Science andInformation technology
Dronacharya College Of Engineering
Farrukhnagar, Gurgaon, India

ABSTRACT

Algorithms are the building blocks of the programming industry created by the human civilization . From automating a bunch of easy tasks to predicting values from an existing datasets by recognizing patterns , algorithms are almost every where . We have successfully created solutions to some pretty difficult problems using algorithms . However , there still remains a class of problems unsolved by applied human knowledge . Hybrid algorithms address those specific types of problems .

Hybrid algorithms play a prominent role in improving the search capability of algorithms. Hybridization aims to combine the advantages of each algorithm to form a hybrid algorithm, while simultaneously trying to minimize any substantial disadvantage. In general, the outcome of hybridization can usually make some improvements in terms of either computational speed or accuracy.

This chapter surveys the detail overview , types and motivations behind the very existence of hybrid algorithms . Types of hybrid algorithms , the discription of each type , the different classifications and subclassifications of algorithms are the main topics of discussion in this chapter .

1 . Introduction

Evolutionary computation has become an important problem solving methodology among many researchers. The population-based collective learning process, self adaptation, and robustness are some of the key features of evolutionary algorithms when compared to other global optimization techniques. Even though evolutionary computation has been widely accepted for solving several important practical applications in engineering, business, commerce, etc., yet in practice sometimes they deliver only marginal performance. Inappropriate selection of various parameters, representation, etc. are frequently blamed.

Evolutionary algorithm behavior is determined by the exploitation and exploration relationship kept throughout the run. All these clearly illustrates the need for hybrid evolutionary approaches where the main task is to optimize the performance of the direct evolutionary approach.Hybrid algorithms are two ormore algorithms that run together to complement each other to create a profitable synergy from their integration.

2. What is Hybrid Algorithm ?

Hybrid algorithms play a prominent role in improving the search capability of algorithms. Hybridization aims to combine the advantages of each algorithm to form a hybrid algorithm, while simultaneously trying to minimize any substantial disadvantage. In general, the outcome of hybridization can usually make some improvements in terms of either computational speed or accuracy. This chapter surveys recent advances in the area of hybridizing different algorithms. Based on this survey, some crucial recommendations are suggested for further development of hybrid algorithms.

In computer science, hybrid algorithms are very common in optimized real-world implementations of recursive algorithms, particularly implementations of divide-and-conquer or decrease-and-conquer algorithms, where the size of the data decreases as one moves deeper in the recursion. In this case, one algorithm is used for the overall approach (on large data), but deep in the recursion, it switches to a different algorithm, which is more efficient on small data. A common example is in sorting algorithms, where the insertion sort, which is inefficient on large data, but very efficient on small data (say, five to ten elements), is used as the final step, after primarily applying another algorithm, such as merge sort or quicksort. Merge sort and quicksort are asymptotically optimal on large data, but the overhead becomes significant if applying them to small data, hence the use of a different algorithm at the end of the recursion. A highly optimized hybrid sorting algorithm is Timsort, which combines merge sort, insertion sort, together with additional logic (including binary search) in the merging logic.

A general procedure for a simple hybrid recursive algorithm is short-circuiting the base case, also known as arm's-length recursion. In this case whether the next step will result in the base case is checked before the function call, avoiding an unnecessary function call. For example, in a tree, rather than recursing to a child node and then checking if it is null, checking null before recursing. This is useful for efficiency when the algorithm usually encounters the base case many times, as in many tree algorithms, but is otherwise considered poor style, particularly in academia, due to the added complexity.

2.1 The Past

Evolutionary algorithms (EAs) are stochastic global optimizers that mimic the metaphor of biological evolution . They are almost always population-based algorithms that learn from the past searches by using a group of individuals or agents . These algorithms often possess behaviour inspired by social or biological behaviour in the natural world . Loosely speaking , there are three categories of EAs , which are :

(i) Evolutionary Programming (EP)
(ii) Evolutionary Strategies (ES)
(iii) Genetic Algorithms (GA)

These algorithms were among the first to offer great advantages in terms of locating the global optimality in vast and complex search spaces, especially when gradient information is unavailable. The implementation of these algorithms are relatively straightforward and easy , based upon simple and easy to understand concepts . They are also reasonably flexible as parameters can be changed easily for better performance .

There were many hybrid algorithms or variants about various evolutionary algorithms . However , Key issues such as slow convergence or premature convergence still exist . In addition , these algorithms can be computationally extensive and requires a lot of iterations for highly complex problems .

2.2 The Present

Current developments often provide some development based on broader improvements over the past few decades, and researchers are still actively trying to develop new algorithms. For example, Rodriguez et al. developed mixed metaheuristic integration with EA and Simulated Annealing (SA). In their review, they found that there were approximately 312 books identified by ISI Web of Science using EA and SA algorithms. By comparison, there were only 123 publications that included EAs and other metaheuristics such as greedy search, repeated local searches, descent declines, and taboo searches. However, Rodriguez et al survey was limited to EAs and South African methods.

In current literature, hybrid algorithms appear to have been widely developed. Using Particle Swarm Optimization (PSO) as an example, the combination of PSO and other search techniques that help it seems to be very effective in improving its performance. Genetic algorithm hybrids (or the use of genetic mutants in other ways) are the most widely studied. Genetic operators such as selection, crossover, and conversion are integrated into the PSO to produce better candidates. Unique evolution, ant colonization improvements and common search techniques used to integrate with the PSO.

In addition, to avoid locating previously detected solutions, techniques such as deflection, sketching, repulsion , self-organizing, and dissipative methods have also been used in the hybrid PSO algorithms. Some biology-inspired operators such as niche and specification technique are introduced into PSO to prevent the swarm from crowding too closely and to locate as many good solutions as possible. A cellular particle swarm optimization, in which a cellular automatic mechanism is integrated in the velocity update to modify the trajectories of particles, is proposed in .

The PSO is just one example, and other hybrid algorithms about the emergence of differences and simulated simulations are also widely studied. Current trends seem to be a combination of new standard / well-established algorithms. For example, a new eagle strategy has been integrated with different variables and improved performance has been achieved.

2.3 The Future

Many new algorithms have been developed in recent years. For example, bio-inspired algorithms such as Artificial Bee Colony Algorithm (ABC), Bat Algorithm (BA), Cuckoo Search (CS), Fire fl y Algorithm (FA), Flower Pollination Algorithm (FPA), Glowworm Swarm Algorithm (GlowSA) , Hunting Search Algorithm (HSA), Eagle Strategy (ES), Roach Infestation Optimization (RIO), Gravitational Search Algorithm (GravSA), Artificial Fish School Algorithm (AFS), Bacterial Evolutionary Algorithm (BEA), Artificial Plant Optimization Algorithm (APO), Krill Herd Algorithm (KHA) and others.

The list is expanding rapidly. These algorithms may possess entities and some novel characteristics for hybridization that remain to be discovered in the near future. However, it is worth pointing out that simple, random hybridization should not be encouraged.

Research efforts should focus on new real, intelligent and effective ways. New strategies and methods should be based on mathematical theory and intelligent analysis of key processes in algorithms so that new mixed algorithms can truly provide more effective solutions to major problems - scale, real-world applications.

2. Why Hybrid Algorithms ?

Optimization is needed in different fields of Engineering to obtain better solutions . To solve optimization problems practically some efficient , effective computational algorithms are very essential.
The main objective of the optimization problem is to derive a optimal solution to the main problem .
An optimization problem is clearly based on the factors : firstly they solve some minimization and maximization objective functions , secondly , a set of unknown variables those are involve in the objective functions and thirdly , a set of constraints that permit the unknowns for taking some specific values but exclude others .

Choosing an optimization method is very important as most of the optimization problems may have more than one local solution . The optimization method must not be greedy and the searching process will not be localized in the neighbourhood of the best solution as it may stick at a local solution and that will misguide the search process . An optimization algorithm should make a balance between global and local search . The problems with large search space or more complex in nature will become difficult

to solve using conventional techniques .

2.1 A heuristic method

A heuristic, or heuristic technique, is any approach to problem-solving that uses a practical method or various shortcuts in order to produce solutions that may not be optimal but are sufficient given a limited time frame or deadline. It can be noted as the way of solving , learning or discovery of a problem using practical methods that ultimately derive immediate near optimal results rather than exact results .

Heuristics can lead to poor decision making based on a limited data set , but the speed of decisions can sometimes make up for the disadvantages. Heuristics can be defined as problem-dependent algorithms, which are developed or adapted to the particularities of a specific optimization problem or problem instance.

A heuristic, or heuristic technique, is any approach to problem-solving that uses a practical method or various shortcuts in order to produce solutions that may not be optimal but are sufficient given a limited timeframe or deadline.

3.1.1 Popular heuristic Algorithms

Optimization heuristics can be categorized into two broad classes depending on the way the solutiondomain is organized:

3.1.1.1 Construction Methods

The greedy algorithm works in stages, where the algorithm makes the right choice for each step as it tries to find the perfect way to solve the whole problem. It is a method used to solve the famous "traveling merchant problem" in which the following heuristic following states: "For each step of the journey, visit a nearby uninhabited city."

Example: Scheduling problem

You are given a set of N schedules for one-day lectures at the university. The specific course schedule is a type (s time, f time) where s time represents the start time of that lesson, and similarly, f time represents the completion time. Given the list of N lectures, we need to select a large set of lectures to be conducted during the day so that there are no overlapping lessons that is when Li and Lj are included in our choice of start time j ≥ time to finish i or vice versa. The best solution is to break your fears or problems into a series of smaller steps. We would sort out the intervals according to the growing order of their expiration dates and start selecting intervals from the beginning.

3.1.1.2 Local Search Methods

The Local Search Method follows a recurring method where we start with the first solution, check the location of the current solution, and replace the current solution with a better solution. In this approach,

the "mobile trader problem" will follow heuristic when the solution is a cycle that contains all the graphnodes and the goal is to reduce the total length of the cycle.

Typically, heuristics systematically perform evaluations, although utilizing stochastic elements. Heuristics use this principle to provide fast, not necessarily exact (i.e., not optimal) numerical solutions to optimization problems. Moreover, heuristics are often greedy to provide fast solutions but get trapped in local optima and fail to find a global optimum.

3.1.1.3 Genetic Algorithms

The name Genetic Algorithm was first used by John Holland. They are designed to mimic Darwin's theory of evolution, which states that living things evolved to produce complex living things and to adapt to life on Earth. Genetic algorithms work on character unit structures, such as biological structures, which change over time according to the survival system of the most powerful through informal but systematic information exchange. Thus, for all generations, a new set of cables is built, using parts of the strong members of the old set. The algorithm is terminated when a satisfactory qualification level has been reached in humans or a higher generation has been reached.

The typical steps are:

1. Choose an initial population of candidate solutions

2. Calculate the fitness, how well the solution is, of each individual

3. Perform crossover from the population. The operation is to randomly choose some pair of individuals like parents and exchange so parts from the parents to generate new individuals

4. Mutation is to randomly change some individuals to create other new individuals

5. Evaluate the fitness of the offspring

6. Select the survive individuals

7. Proceed from 3 if the termination criteria have not been reached

2.2 A meta heuristic method

Meta heuristics can be defined as problem-independent, general-purpose optimization algorithms. They apply to a wide range of problems and problem instances. The term meta describes the higher-level general methodology, which is utilized to guide the underlying heuristic strategy .

A meta heuristic method is an iterative generation procedure to solve a subordinate heuristic by syndicating intelligently different concepts to explore and exploit the search space , learning strategies are applied to structure information in order to find efficiently near-optimal solutions.

As such, they do not take advantage of any specificity of the problem and, therefore, can be used as blackboxes. In general, they are not greedy. In fact, they may even accept a temporary deterioration of the solution which allows them to explore more thoroughly the solution space and thus to get a hopefully better solution .

Metaheuristic is an approach method based on a heuristic method that does not rely on the type of theproblem. The metaheuristic method can be distinguished into two which are metaheuristic with single- solution based (local search) and metaheuristic based on population (random search).

3.2.1 Some Popular Metaheuristic algorithms

3.2.1.1 Ant Colony Optimization

Ant Colony Optimization is a popular example of metaheuristic design ways to solve various integration problems. In 1991, Dorigo et al. introduced the main method that can be collected internally the building and, since then, a number of researchers have introduced many different variations of the basic principle in the literature. Priority a basic goal, inspired by the food behavior of real ants struggling to collect food helped to acquire the concept of ACO. Real ants they have no sharp vision. Real ants are trying to find a shortcut from there their nest in the fountain of food. They release a chemical, known as “pheromones down to share information about food sources among them. If an ant picks the right path to follow, releases more chemicals in that path, leading to a stronger pathway, motivating others to follow the same path.

3.2.1.2 Particle Swarm Optimization

Particle swarm optimization is a popular human-inspired, natural-inspired process, founded by Eberhart and Kennedy in 1995. Inspiration behind the development of PSO social behavior and the flexibility of the movement of fish, insects or birds. PSO is appropriate and effective in solving ongoing evolving problems. This metaheuristic is easy to implement and is not based on free. Contains a very limited set of parameters for demographics. It begins to work with a set of randomly distributed particles, sometimes called potential solutions, metaheuristic attempts to improve the quality of solutions on the basis of quality measurement, called qualitative function. Improvement is achieved by moving the participating particles around search the space with ant for a few simple mathematical expressions. Wise generation, the PSO promotes particles in the best position (achieved by that particle) and the best particle so far. The idea is to allow birds, insects or fish to locate an unfamiliar place in every search by exploiting the same information.

```
begin
   itn:= 0; initialSwarm
   Q(itn); evaluate Q(itn);
   repeat
      begin
         itn:= itn + 1; sltPbst
         Q(itn); sltGbst Q(itn);
         calpVelocity Q(itn);
         updatePosition Q(itn);
         evaluate Q(itn);
       end
     until itn >MxGen
   end
```

The InitialSwarm function is used to randomly generate a series of experimental solutions (Q). Then, the test function determines the value of each qualification particles in Q. The sltPbst and sltGbst functions are used to determine the best fit for each particle and the best qualification value to date. The calpVelocity and update position functions are used to modify the location of each particle in the search field. This process is used for MxGen number of generations

2.1 A Hyper heuristic method

Hyper-heuristics represent a novel search methodology that is motivated by the goal of automating the process of selecting or combining simpler heuristics in order to solve hard computational search problems. An extension of the original hyper-heuristic idea is to generate new heuristics which are not currently known. These approaches operate on a search space of heuristics rather than directly on a search space ofsolutions to the underlying problem which is the case with most meta-heuristics implementations.

Using hyper-heuristics, we are attempting to find the right method or sequence of heuristics in a given situation rather than trying to solve a problem directly. Indeed, using hyper-heuristic methods, we are searching for a generally applicable methodology rather than solving a single problem instance. Hyper- heuristic approaches aim to be generic methods, which should produce solutions of acceptable quality fora set of problems.

Hyper-heuristics represent a novel search methodology that is motivated by the goal of automating the process of selecting or combining simpler heuristics in order to solve hard computational search problems. An extension of the original hyper-heuristic idea is to generate new heuristics which are not currently known.

3.3.1 .Classifications of approaches

Hyper -heuristic approaches so far can be classified into two main categories . In the first class , capturedby the phrase **heuristics to choose heuristics** . The hyper -heuristic framework is provided with a set of pre-existing , generally widely known heuristics for solving the target problem . The task is to discover a good sequence of applications of these heuristics for efficiently solving the problem . At each decision stage, a heuristic is selected through a component called selection mechanism and applied to an incumbent solution. The new solution produced from the application of the selected heuristic is accepted/rejected based on another component called acceptance criterion. Rejection of a solution meansit is simply discarded while acceptance leads to the replacement of the incumbent solution.

In the second class, **heuristics to generate heuristics**, the key idea is to "evolve new heuristics by making use of the components of known heuristics." The process requires, as in the first class of hyper-heuristics, the selection of a suitable set of heuristics known to be useful in solving the target problem. However, instead of supplying these directly to the framework, the heuristics are first decomposed into their basic components.

These two main broad types can be further categorised according to whether they are based on constructive or perturbative search. An additional orthogonal classification of hyper-heuristics considers the source providing feedback during the learning process, which can be either one instance (on-line learning) or many instances of the underlying problem studied (off-line learning).

3.3.2 .Methodologies to choose heuristics

Discover good combinations of fixed, human-designed, well-known low-level heuristics.

- Based on constructive heuristics
- Based on perturbative heuristics

Methodologies to generate heuristics
Generate new heuristic methods using basic components of previously existing heuristic methods.

- Based on basic components of constructive heuristics
- Based on basic components of perturbative heuristics

3.3.1 On-line learning hyper-heuristics

The learning takes place while the algorithm is solving an instance of a problem, therefore, task-dependentlocal properties can be used by the high-level strategy to determine the appropriate low-level heuristic to apply. Examples of on-line learning approaches within hyper-heuristics are: the use of reinforcement learning for heuristic selection, and generally the use of metaheuristics as high-level search strategies overa search space of heuristics

3.3.1 Off-line learning hyper-heuristics

The idea is to gather knowledge in form of rules or programs, from a set of training instances, which would hopefully generalize to the process of solving unseen instances. Examples of off-line learning approaches within hyper-heuristics are: learning classifier systems, case-base reasoning and genetic programming.

3.3.1 Heuristic selection methodologies

3.3.1.1 Approaches based on construction of low level heuristics

These approaches form the solution increasingly. Starting with an empty solution, the goal is to choosewisely and use the construction heuristics to gradually build a complete solution. The hyper-heuristic framework is provided by a set of existing construction heuristics (usually specified problem), and the challenge is to select the heuristic in some way most appropriate to the current problem situation. This process continues until a final state (complete solution) is found.

Note that there is a natural end to the construction process, that is, when a complete solution is reached.The sequence of heuristic options is therefore complete and the size of the basic integration problem is determined. In addition, there is an interesting opportunity to study organizations between the categoriesof the fixed solution and the adequate heuristics of those sections.

3.3.2.1 Approaches based on perturbation of low level heuristics

These methods start with a complete solution, either randomly generated or using simple construction heuristics, and then we have repeatedly tried to improve the current solution. The hyper-heuristic framework is provided by a set of local structures and / or simple local search engines, and the goal is torepeatedly select and apply the complete current solution. This process continues until the suspension condition is met.

Note that these methods are different from those based on construction heuristics, as they do not have the nature of extinction. The sequence of heuristic options, legally, can be extended illegally. This hyper- heuristics class has the potential to be used effectively in a variety of integration problems, as standard neighborhood structures or simple local search engines can be made available. Hyper-heuristics based on

distraction has been used in staff planning, time planning, shelf space allocation, packing and car routeproblems.

3.3.5 Heuristics generation methodologies

Many methods of generating heuristics use genetic programming, a branch of evolutionary computation that deals with the automatic production of computer programs. Apart from certain representations, it differs from other forms of evolution in its use.

Although many applications of evolutionary algorithms deal with performance problems, genetic predisposition may be put in place for machine learning. Genetic planning has been used successfully in automated production of heuristics that solves complex integration problems, such as boolean satisfaction, drum packaging, a mobile retailer problem and production planning.

4. Motivations For Hybridization

In a hybrid algorithm, two or more algorithms solve jointly and collaboratively with a predefined problem. In some hybrids, one algorithm may be included as a sub-algorithm to obtain parameters that fit the otheralgorithm, while in some cases, different components of such algorithms for conversion and crossover areused to develop another algorithm in a hybrid structure. In this regard, hybrid algorithms can be freely divided into two categories:

(i) *Unified purpose hybrids* Under this category, all sub-algorithms are used to solve the same problem directly; as well as the various sub-algorithms used by the non-target search categories. Hybrid metaheuristic algorithms with local search are a rare example. Global search explores the search area, while local search is used to filter out areas that may contain the best content in the world.

(ii) *Multiple purpose hybrids* One main algorithm is used to solve the problem, while a sub-algorithm is used to define the parameters of the main algorithm. For example, the PSO can be used to determine the total conversion rate of GAs. With this, the PSO does not solve the problem, but helps to find better solutions by searching for the right parameter for better performance. Hyper-heuristic algorithms can be considered as a form of mixed methods. In hyper-heuristic methods, parameters are selected

5. Taxonomy of Hybrid Algorithms

The goal of the general taxonomy is to provide a mechanism to allow comparisons of hybrid algorithms in aqualitative way . It is hoped that the categories and their relationships to each other have been chosen carefully enough to indicate areas in need of future work as well to help classify future work . Among existing taxonomies in other domains , one can find examples of flat and hierarchical classification schemes .The taxonomy could usefully be employed to classify any hybrid optimization algorithm . Metaheuristics are a general heuristics applicable to a large optimization problems .

hybrid algorithms can be grouped into two categories .

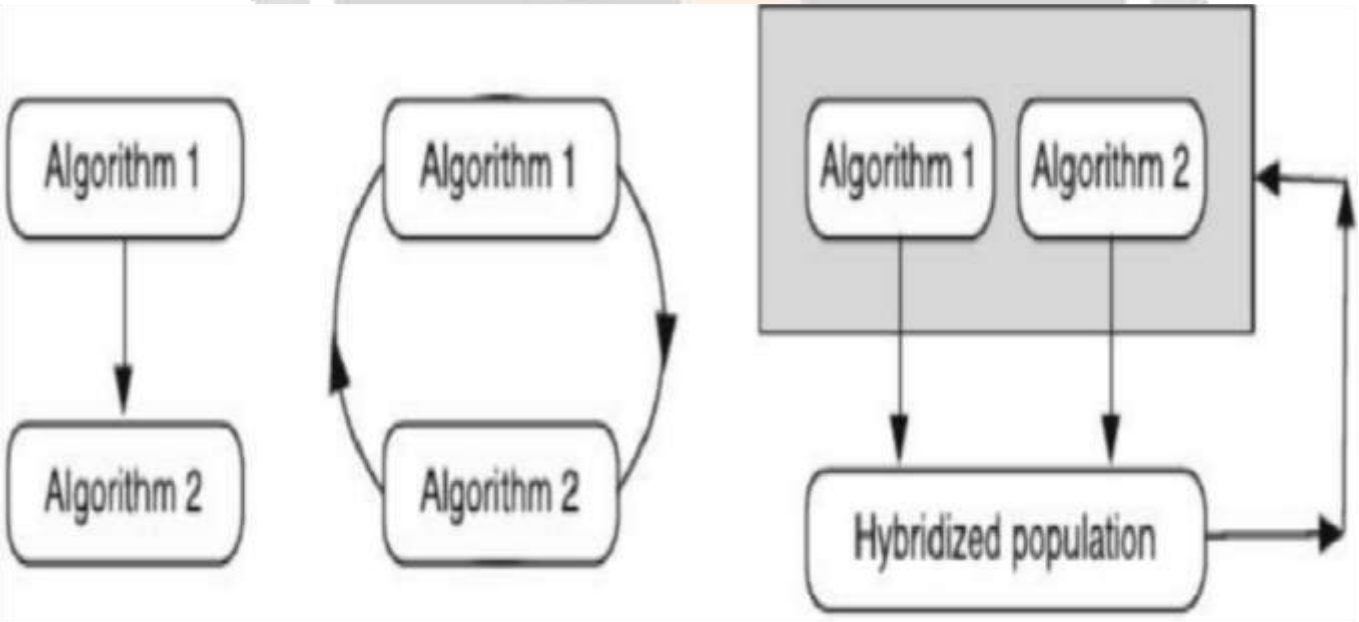
5.1 Collaborative Hybrids

This involves a combination of two or more algorithms that work in sequence or in sequence. The contributing weight of each participating algorithm can be taken as a fraction and a half in the simplest case.

(i) Multi-stage. There are two stages involved in this case. The first algorithm acts as the global optimizer whereas the second algorithm performs local search. The first algorithm is capable of exploring the search space globally to locate promising area of convergence . Then the second algorithm will perform a deep local search like climbing a hill and a simplex descending path. A challenging problem in using such a system is knowing when to switch to a second algorithm. Steps such as diversification should be integrated to assist in the issue of change. The latter applies to the Genetic Algorithm used as a global algorithm (first algorithm), with Particle Swarm Optimization (PSO) as a local detector (second algorithm).

(i) Sequential . In this structure, both algorithms are used separately until one integration process is met. For convenience, both algorithms will be used with the same number of repetitions before moving on to the next algorithm.

(ii) Parallel. Two algorithms are used simultaneously, using the same number of people. One of the algorithms may be applied to the previously specified percentage of the algorithm.



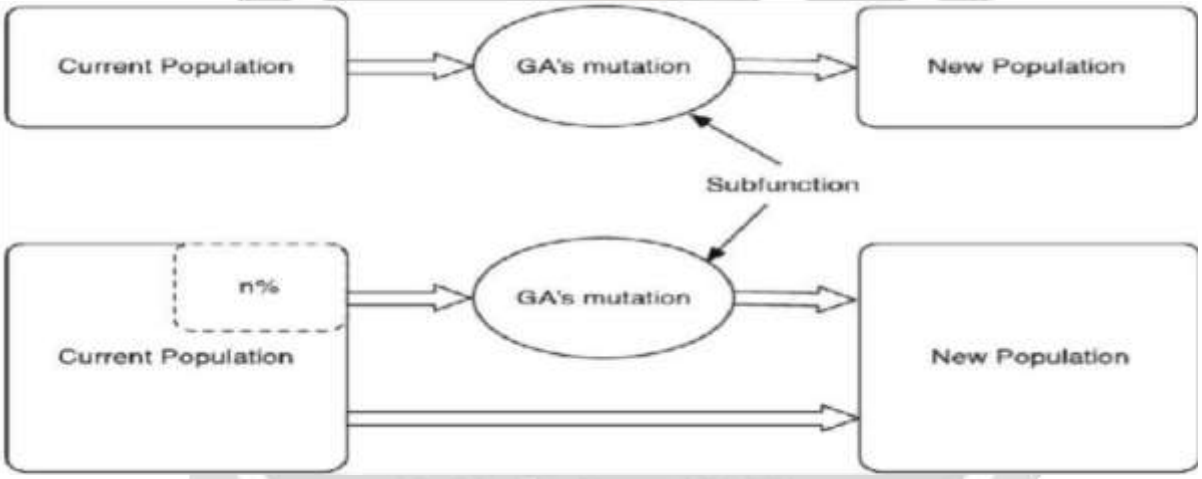
5.1 Interactive hybrids

In this feature, a single algorithm is considered to be subordinate, embedded in the master metaheuristic. At this stage, the contributing weight of the second algorithm is estimated at 10-20%. This involves the installation of a deceptive user from the second algorithm to the main algorithm. For example, many algorithms have used a variable operator from GA to PSO, which has resulted in what is called Genetic PSO or Mutated PSO. Some may include gradient techniques such as hill climbs, steep descents, and Newton- Raphson in the main algorithm.

There are two possible approaches :

(i) Full manipulation . All demographics are used regularly .This function can be integrated with existing source code, usually as subroutine / sub function.

(ii) Partial manipulation . In this deception, only a fraction of the total population is accelerated using local search methods such as gradient methods. Choosing the right component and the right candidateto accelerate poses a major challenge in ensuring the success of this hybrid structure.



6. Disadvantages and Challenges of Hybrid Algorithms

Although hybrid algorithms offers great advantage of increasing the diversity in a population and hence enhancing the search capability of the developed hybrid algorithm, some drawbacks do exist, which will bediscussed in the following subsections.

6.1 Naming Convention

The inclusion of another algorithm often leads to the problem of naming. Some researchers adopt very different names from their hybrid algorithms. For example, the GAAPI algorithm is a summary of the Hybrid Ant Colony-Genetic Algorithm, which confuses some researchers. A hybrid word like HPSO-BFGS seems to be a stressful summary, which is hard to read. the interoperable type of hybrid algorithm seems to create complex words. For example, it may be interesting to compare the terms Hybrid GA-PSO (co- operative) with Mutated PSO (integrated), even though the two hybrids combine GA and PSO.

6.2 Complexity of Hybrid Algorithms

In terms of algorithm structure, the hybridization process tends to create additional components in the overall hybrid algorithm architecture .This increases the complexity of the hybrid algorithm .Because of the complex structure, algorithms are a combination that has certain resistance to be accepted by researchers.In the literature, two popular hybrid algorithms are the Hybrid Taguchi-Genetic Algorithm and the Hybrid Genetic Algorithm, both published in 2004. From the quote, it seems to have been well received. It is interesting to note that both algorithms fall into a combination of a mixed algorithm, with simple taxonomy / structures.

6.3 Computational Speed

In many applications, hybrid algorithms appear to improve the results in terms of compliance speed and absolute accuracy. However, these aggregation graphs are usually sorted according to the repetition rate.This means that rapid mixing does not mean a real mixing level because the mixture usually uses a high number of repetitions (internal or indistinct). For example, in a compound (sequential type) a mixed algorithm such as GA-PSO, cycle, or single multiplication includes GA and PSO. For good comparison, this should be considered two cycles instead of one on the merge graph. To avoid this problem, last-time

performance should be used as a metaphor for comparing mixed algorithms with non-hybrid algorithms. Besides, due to the very complex structures in hybrid algorithms, the overhead appears next to its weight,which is usually unavoidable.

This affects the overall performance and thus reduces its durability. The time spent with overheads should be taken into account for proper comparison. Also, this is possible by recording the actual number of repetitions taken to achieve the predetermined direction, although the complexity of time must be compared again.

There are other issues related to hybrid algorithms. For example, many hybrid algorithms will increase the number of parameters in the algorithms, thus making it difficult to adjust their parameters. In addition, the complex structure of the hybrid often makes it difficult to analyze, and thus we gain little understanding ofthe reasons why such hybrids work. In addition, hybrid algorithms are less difficult to perform, and thus more prone to errors. Therefore, caution should be exercised when interpreting results from integrated algorithms.

7. Recommendations for Future Developments

Based on the above analysis and observations we can highlight some insights for future developments :

(i) Simpler algorithm are preferred than more complex algorithms . algorithms should be made as simple as possible . People tend to use a robust algorithm that has simpler architecture for the ease ofimplementation and is yet efficient to be useful to real world applications .

(ii) Shorter names are much preferred in the scientific community .

(iii) New hybrids should have a clear structure that is easier for implementations . Any combination should be based on clear thinking , novel feature and insightful mechanisms which is more likely toproduce better hybrids in the long run .

8. Conclusions

In this chapter, we have reviewed a variety of algorithms and investigated the causes of their development. We also categorized these algorithms, based on hybridization techniques. In addition, some barriers to hybridization have been discussed. The latest examples of hybrid algorithm from the literature are presented, with a brief summary of some of the outstanding applications. Finally, some suggestions were recommended that could be useful in the future development of integrated algorithms.

9. References:

https://www.sciencedirect.com/topics/computer-science/hybrid-algorithm

https://en.wikipedia.org/wiki/Hyper-heuristic

Hybrid Evolutionary Algorithms: Methodologies, Architectures, and Reviews Crina Grosan & Ajith Abraham

A Taxonomy of Hybrid Metaheuristics : E.-G. TALBI Laboratoire d' Informatique Fondamentale de Lille, URA CNRS 369, Cit´e scientifique , 59655 Villeneuve d' Ascq Cedex, France

A Classification of Hyper-heuristic Approaches Edmund K. Burke, Matthew Hyde, Graham Kendall, Gabriela Ochoa, Ender¨