SWARM AND EVOLUTIONARY COMPUTING

HYBRID MULTIOBJECTIVE OPTIMIZATION ALGORITHMS

Assignment

Anurag Malyala
2K15/CO/035

HYBRID MULTIOBJECTIVE OPTIMIZATION ALGORITHMS

MULTIOBJECTIVE OPTIMIZATION PROBLEMS

multiobjective problems (MOPs), where the quality of a solution is defined by its performance in relation to several, possibly conflicting, objectives. In practice it turns out that a great many applications that have traditionally been tackled by defining a single objective function (quality function) have at their heart a multiobjective problem that has been transformed into a single-objective function in order to make optimization tractable.

E.g. Selecting a house to buy, manufacturing something keeping the quality high and production costs low.

Two ways of solving:

- present the user with a diverse set of possible solutions, representing a range of different trade-offs between objectives.
- assign a numerical quality function to each objective, and then combine these scores into a single fitness score using some (usually fixed) weighting. This approach, often called scalarization.

Scalarization, drawbacks:

- the use of a weighting function implicitly assumes that we can capture all the user's preferences, even before we know what range of possible solutions exist.
- for applications where we are repeatedly solving different instances of the same problem, the use of a weighting function assumes that the user's preferences remain static, unless we explicitly seek a new weighting every time.

Two common terms which come up when talking about Multiobjective optimization problems are Dominance and Pareto.

Dominance

given two solutions, both of which have scores according to some set of objective values (which, without loss of generality, we will assume to be maximized), one solution is said to dominate the other if its score is at least as high for all objectives and is strictly higher for at least one.

We can represent the scores that a solution A gets for n objectives as an n-dimensional vector a. Using the ≥ symbol to indicate domination, we can define A ≥ B formally as:

$$A \geq B \iff \forall i \in \{1, \ldots, n\}\ ai \geq bi, and\ \exists i \in \{1, \ldots, n\}, ai > bi.$$

For conflicting objectives, there exists no single solution that dominates all others, and we will call a solution nondominated if it is not dominated by any other. All nondominated solutions possess the attribute that their quality cannot be increased with respect to any of the objective functions without detrimentally affecting one of the others.

Pareto

The set of all nondominated solutions is called the Pareto set or the Pareto front.

EVOLUTIONARY ALGORITHM FOR SOLVING MOPS

Nonelitist Approaches

Multiobjective Genetic Algorithm (MOGA): assigns a raw fitness to each solution equal to the number of members of the current population that it dominates, plus one. It uses fitness sharing amongst solutions of the same rank, coupled with fitness-proportionate selection to help promote diversity.

Nondominated Sorting Genetic Algorithm (NSGA): assigns fitness based on dividing the population into several fronts of equal domination. To achieve this, the algorithm iteratively seeks all the nondominated points in the population that have not been labelled as belonging to a previous front. It then labels the new set as belonging to the current front, and increments the front count, repeating until all solutions have been labelled. Each point in each front gets as its raw fitness the count of all solutions in inferior fronts. Again, fitness sharing is implemented to promote diversity, but this time it is calculated considering only members from that individual's front

Niched Pareto Genetic Algorithm (NPGA): uses a modified version of tournament selection rather than fitness proportionate with sharing. The tournament operator works by comparing two solutions first based on whether they dominate each other, and then second on the number of similar solutions already in the new population.\

All these share two common features:

- the performance they achieve is heavily dependent on a suitable choice of parameters in the sharing/niching procedures
- they can potentially lose good solutions.

Elitist Approaches

NSGA-II

Uses the same non-dominated fronts concept but incorporates:

- A crowding distance metric is defined for each point as the average side length of the cuboid defined by its nearest neighbors in the same front. The larger this value, the fewer solutions reside in the vicinity of the point.
- A $(\mu + \lambda)$ survivor selection strategy is used (with $\mu = \lambda$). The two populations are merged, and fronts assigned. The new population is obtained by accepting individuals from progressively inferior fronts until it is full. If not all of the individuals in the last front considered can be accepted, they are chosen on the basis of their crowding distance.
- Parent selection uses a modified tournament operator that considers first dominance rank then crowding distance.

HYBRID MULTIOBJECTIVE OPTIMIZATION

WHY?

The task of EMO algorithms is to find Pareto-optimal solutions as many as possible. It is, however, impractical to try to find true Pareto-optimal solutions of large-scale combinatorial optimization problems. Thus, non-dominated solutions among examined ones are presented to decision makers as a result of the search by EMO algorithms. In this case, EMO algorithms try to drive populations to true Pareto-optimal solutions as close as possible for obtaining a variety of near Pareto-optimal solutions.

One promising approach for improving the search ability of EMO algorithms to find near Pareto-optimal solutions is the hybridization with local search, when dealing with single objective problems this approach is called memetic algorithms.

Framework

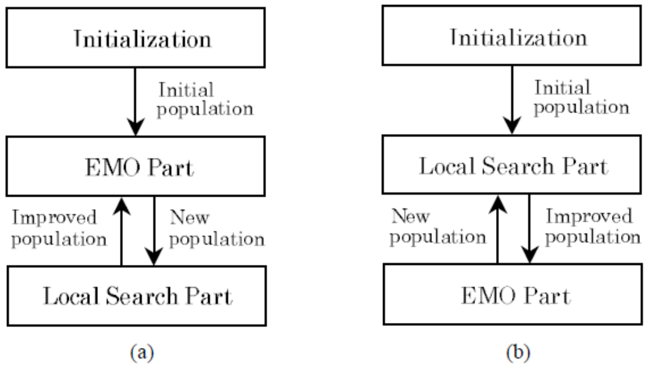

Figure 1

Generic frameworks of hybrid EMO algorithms are shown in Fig. 1. The two frameworks in Fig. 1 are the same except for the order of genetic search and local search. Genetic operations are first applied to an initial population in Fig. 1 (a). On the other hand, genetic operations are applied after an initial population is improved by local search in Fig. 1 (b). The two frameworks are executed in the same manner after the second generation: the EMO part and the local search part are iterated for finding Pareto-optimal solutions. Emphasis is implicitly placed on the local search part in Fig. 1 (b) while the EMO part is implicitly viewed as the main part in Fig. 1 (a). In this paper, we use the framework in Fig. 1 (a) for describing hybrid EMO algorithms. In Fig. 1 (a), the local search part can be also viewed as a special kind of mutation in EMO algorithms.

Algorithm 1

Multiobjective Genetic Local Search Algorithm

0. Initialization: Randomly generate an initial population of Npop solutions.

EMO Part

1. Evaluation: Calculate the n objectives for each solution in the current population. Then update the secondary population where non-dominated solutions are stored separately from the current population.
2. Selection: Repeat the following procedures to select (Npop - Nelite) pairs of parent solutions where Nelite is the number of elite solutions.
    a. Randomly specify the weight values w1, w2, ..., wn .
    b. Select a pair of parent solutions using the scalar fitness function in (3). Theroulette wheel scheme in (4) is used for the selection of parent solutions.
3. Crossover and mutation: Apply a crossover operation to each of the selected (Npop - Nelite) pairs of parent solutions. A new solution is generated from each pair. Then apply a mutation operation to each of the generated new solutions.
4. Elitist strategy: Randomly select Nelite solutions from the secondary population. Then add their copies to the (Npop - Nelite) solutions generated in Step 3 to construct a population of Npop solutions.

Local Search Part

5. Local search: Iterate the following three steps Npop times. Then replace the current population with Npop solutions obtained by the following steps.
    a. Randomly specify the weight values w1, w2, ..., wn
    b. Select a solution from the current population using tournament selection with replacement based on the scalar fitness function in (3) with the current weight vector specified in (a). A copy of the selected solution is

c. used in (c). Thus, no solution is removed from the current population

d. Apply local search to a copy of the selected solution using the current weight vector with the local search probability PLS . When local search is applied, the current solution after local search is included in the next population. As mentioned above, local search is terminated when no better solution is found among k neighbors randomly generated from the current solution. On the other hand, when local search is not applied, a copy of the selected solution in (b) is added to the next population.

6. Return to step 1.

Hybridizing ACO and Firefly algorithm.

Algorithm 1 The pseudo code of evaluating the non-dominated solutions(left) and updating the archive (right)

```
Input: $A^{(t)} = \phi; D = \phi; C = \{C_l\}_{l=1}^{K}, C_l = \{P_j\}_{j=1}^{m}$.
$i = 1$.
If $P_i \prec \forall P_j \ \wedge i \neq j$ then
    $D = D \cup \{P_i\}$
Else
    $A^{(t)} = A^{(t)} \cup \{P_i\}$
End ; $i = i + 1$
output : $A^{(t)}$
```

```
Input: $A^{(t)}, P$
If $\nexists P^1 \in A^{(t)} | P \succ P^1$ then
    $A^{(t)} = A^{(t)} \cup P$
Else if $\exists P^1 \in A^{(t)} | P \succ P^1$ then
    $A^{(t)} = \{A^{(t)} \cup P\} \setminus \{P^1\}$
Else
    $A^{(t)} = A^{(t)}$
End
output : $A^{(t)}$
```

Algorithm 2 The pseudo code of the local search scheme

```
Input: $\mathbf{x} = (x_1, x_2, ..., x_i, ..., x_n); x_i^L; x_i^U$ ; number of maximum iteration ( $N$ ).
Set $n = 0$
Generate $d\mathbf{x}$ ( $d\mathbf{x} = .5 * (U - L) * (\varepsilon)^n$
If $\exists \mathbf{x}' = \mathbf{x} + d\mathbf{x} \,|\mathbf{x}' \in \Omega$ ,then $\mathbf{x}_{new} = \mathbf{x}'$
Else if $\exists \mathbf{x}' = \mathbf{x} - d\mathbf{x} \,|\mathbf{x}' \in \Omega \ \mathbf{x}' \in \Omega$ ,then $\mathbf{x}_{new} = \mathbf{x}'$
Else $\mathbf{x}_{new} = []$ then $n = n + 1$
End if
output : $\mathbf{x}_{new}$
```

Algorithm 3 The pseudo code of the movement local movement

```
Input: $\mathbf{x}_j \in A^{(t)}$, $\mathbf{x}_i \in R^{(t)}$, $\mu = 0.5, \lambda = 1$ and maximum iteration ( $T$ ).
while t$\leq T$ do
      $\mathbf{x}_i = \mathbf{x}_i + rand(0,1).(\mathbf{x}_i - \mu(\mathbf{x}_j - \mathbf{x}_i), \mathbf{x}_j + \mu(\mathbf{x}_j - \mathbf{x}_i))$ ; $\beta_n = (1 + 2\mu) + \mu\,\lambda(t)$.
      If $\exists \mathbf{x}_i = \mathbf{x}_i + \beta_n(\mathbf{x}_j - \mathbf{x}_i) \,|\mathbf{x}_i \in \Omega$ ,
      Else if $\mathbf{x}_i = \mathbf{x}_j + \beta_n(\mathbf{x}_i - \mathbf{x}_j) \,|\mathbf{x}_i \in \Omega$
      Else $\lambda(t) = 0.1 * \lambda$ .
    end
end while
output : $\mathbf{x}_i$
```

Algorithm 4 The pseudo code of the proposed HMO-ACO/FA approach

```
Initialize the parameters for VEACO and FA.
Initialize all colonies $(C_1, C_2, ..., C_K)$.
while the tour not completed do

    HMO-ACO/FA algorithm.
    while the maximum iteration not completed do
        Evaluation for each colony.
        Update the pheromone according to the Equation (9) and Equation (10) for all colonies.
        Construct the solutions for all colonies according to the Equation (11).
        Shuffled the solution of all colonies.
        Apply Algorithm 1 to determine the non-dominated solutions.
        Apply the local search scheme according to Algorithm 2.
    end while

    FA algorithm.
    Create an initial population of fireflies within $n$-dimensional search space.
    while the condition not terminated do
        Separate the infeasible fireflies from the feasible.
        Move the infeasible fireflies toward the solutions stored in Pareto repository by using
        Algorithm 3 until become feasible.
        Evaluate the fitness of the population $(f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_K(\mathbf{x}))$ which is directly proportional to
        light intensity.
        Determine the non-dominated solutions by Algorithm 1.
        Evaluate combined the light intensities according to the Equation (12).
        Rank the solutions.
        Update the movement of the fireflies according to the Equation (13).
    end while
The new solutions randomly divided into $K$ colonies, $(C_1, C_2, ..., C_K)$.
end while
```

1. Unlike others, it uses a vector of colonies and objective functions which give a better generalization to the problem solution
2. The use of both, ACO and Firefly steps in conjunction with the Local search strategy leads to a better non-dominated solution
3. The algorithm captures fronts which get missed by algorithms using only ACO or FA.
4. The algorithm proved to give state-of-art results for the environmental/ economic power dispatch (EED) problem

CONCLUSION

The hybridization of EA for MOPs can be done through various ways, by either including a local search strategy or by having an alternative EA work on the same solution and then mixing results at each step