Contents lists available at ScienceDirect

Nuclear Instruments and Methods in Physics Research A

journal homepage: www.elsevier.com/locate/nima

Multi-objective particle swarm and genetic algorithm for the optimization of the LANSCE linac operation

X. Pang*, L.J. Rybarcyk

Los Alamos National Laboratory, NM 87544, USA

ARTICLE INFO

Article history:
Received 23 October 2013
Received in revised form
5 December 2013
Accepted 17 December 2013
Available online 30 December 2013

Keywords:
MOPSO
MOGA
Linac
Optimization

ABSTRACT

Particle swarm optimization (PSO) and genetic algorithm (GA) are both nature-inspired population based optimization methods. Compared to GA, whose long history can trace back to 1975, PSO is a relatively new heuristic search method first proposed in 1995. Due to its fast convergence rate in single objective optimization domain, the PSO method has been extended to optimize multi-objective problems. In this paper, we will introduce the PSO method and its multi-objective extension, the MOPSO, apply it along with the MOGA (mainly the NSGA-II) to simulations of the LANSCE linac and operational set point optimizations. Our tests show that both methods can provide very similar Pareto fronts but the MOPSO converges faster.

Published by Elsevier B.V.

1. Introduction

The half mile long LANSCE 800-MeV linac provides both H^+ and H^- beams for a variety of user programs. The tune-up procedure of the linac usually begins with direct low-power beam measurements and an envelope code based physics model to establish operating set points of the machine. However, in the transition to high power operations of the facility, linac parameters are adjusted empirically with the goal of achieving minimal loss of the beam over the accelerator. This is done without any direct measurement of the beam. The lengthy adjustment process is highly subjective and more like a random walk with varying degrees of success. To better understand our position in the whole solution space, we have applied multi-objective optimization algorithms to explore the optimal operating set points for our linac.

The multi-objective genetic algorithm (MOGA) has been extensively applied to various accelerator design problems [1–3]. Its applications have been summarized in Ref. [4]. The algorithm is based on the genetic algorithm (GA) which is a well-established single objective optimization method whose original form was conceived in 1975 [5]. It emulates the genetic evolution of species to evolve solutions. Like GA, particle swarm optimization (PSO) is also inspired by nature. It was first proposed in 1995 [6] and was initially inspired by the self-organizing behaviors of social animals living in groups. However, it has a simpler computational paradigm and has shown faster convergence and better computational efficiency than GA in the single objective optimization domain [7,8]. Promising results have also been reported for its applications to MO problems [9]. In this paper, we will introduce the PSO and multi-objective PSO (MOPSO) algorithms, apply both MOGA and MOPSO to real-world optimization problems encountered in linac operation and compare their performance.

2. Multi-objective optimization methods

MO optimization algorithms always strive to find the optimal trade-off possibilities between different and oftentimes conflicting objectives. The optimal final solutions lie on the Pareto front [10,11]. It represents a compromise of various objectives where improving any objective results in a degradation of one or more of the others. The solutions on the Pareto front are the so-called *non-dominated* set. The concept of *domination* in the presence of constraints is defined as follows [12]:

Definition 1. Solution x *dominates* solution y if any of the following conditions are satisfied:

- Solution x is feasible (no constraint violation) but solution y is not.
- Both solutions are infeasible. Solution x has a smaller overall constraint violation.
- Both solutions are feasible. Solution x is no worse than solution y in all objectives and is better than y in at least one of the objectives.

The ultimate goal of any MO optimization algorithm is to push the estimated front toward the true Pareto front and keep it as diverse as possible. An effective way to quantify the diversity of solutions

* Corresponding author.
E-mail address: xpang@lanl.gov (X. Pang).

0168-9002/$ - see front matter Published by Elsevier B.V.
http://dx.doi.org/10.1016/j.nima.2013.12.042

is to calculate the crowding distance [12] which estimates the population density surrounding one particular solution. A non-dominated solution with a larger crowding distance lies in a less dense area of the solution space. This solution would be preferred in the search process so that the under-represented areas of the Pareto front can be explored.

2.1. Multi-objective genetic algorithm (MOGA)

MOGA maintains a population of solutions and uses computational models of evolutionary processes, such as natural selection, survival of the fittest, and chromosome crossover and mutation, to guide the exploration of multiple parts of the Pareto front simultaneously. Several GA algorithms have been developed e.g. NSGA-II [12], PAES [13], and SPEA2 [14] and adopted by accelerator physicists to tackle various design problems [1–4]. In this study, the NSGA-II is employed due to our familiarity with the algorithm. Its algorithm is briefly listed here.

Algorithm 1. NSGA-II.

- Population initialization.
- Repeat the following until the maximum number of iterations is reached or a convergence criteria is met.
  1. Use binary tournament to select parents and simulated binary crossover (SBX) to generate children.
  2. Apply polynomial mutation on children.
  3. Apply non-dominated sorting to combined parents and children.
  4. Pick the top half of the merged population as the new population.

The parameters associated with different steps of the algorithm that can eventually affect the overall outcome and efficiency include crossover rate, mutation rate, and parameters that control the shapes of the probability density functions for the SBX and polynomial mutation [10,12].

2.2. PSO and MOPSO

The PSO heuristic was first introduced by Kennedy and Eberhart in 1995 [6]. Its initial inspiration came from the “graceful but unpredictable choreography of a bird flock”. This kind of self-organizing behavior can be observed in a wide range of species living in groups e.g. bird flock, fish school, and ant colony. Despite the limited knowledge of the external environment by each individual animal within the group, they are able to move together toward food sources, avoid dangers and thrive in harsh natural environments. The key to their success lies in social influence and learning. Each individual′s behavior is greatly influenced by both its own personal experience and the social standard. This is the basic principle that PSO is trying to emulate.

Within a swarm, each individual particle′s position refers to a point in the variable space. It is updated by adding a velocity dependent correction as depicted in Fig. 1. Let $\mathbf{x}_i^t$ denotes particle i′s position at time step t in the variable space, and $\mathbf{v}_i^{t+1}$ denotes its velocity at a unit time step later $t+1$, then

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{v}_i^{t+1}. \tag{1}$$

In PSO, it is the velocity term that drives the optimization and reflects both the personal experience of the particle and the socially exchanged formation within the swarm. $\mathbf{v}_i^{t+1}$ is updated as follows:

$$\mathbf{v}_i^{t+1} = w\mathbf{v}_i^t + c_1 r_1(\mathbf{pbest}_i^t - \mathbf{x}_i^t) + c_2 r_2(\mathbf{gbest}^t - \mathbf{x}_i^t). \tag{2}$$

The parameter w is the inertia of the particle. It reflects the effect of the particle′s current motion. c_1 is a positive number used to scale the

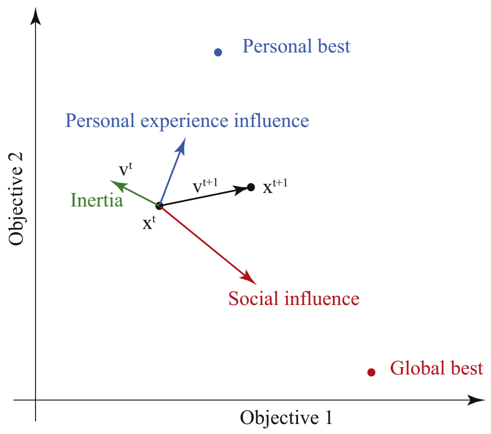

Fig. 1. Velocity and position updates in PSO.

contribution of a particle′s personal experience. $\mathbf{pbest}_i^t$ is the personal best position particle i has visited since the first time step. The second term of the equation shows that the further away the particle is from its personal best the more it will be pulled towards it in the next time step. This is conceptually analogous to “nostalgia” in that the individual tends to return to the place it encountered in the past that best satisfied the objectives. The third term of the equation represents the effects of publicized knowledge or social norms which individuals seek to attain. c_2 is a positive scale factor. $\mathbf{gbest}^t$ is the best position ever for the entire swarm. r_1 and $r_2 \in [0,1]$ are random numbers sampled from a uniform distribution. They are the stochastic elements of the algorithm. With position and velocity defined, the PSO algorithm simply consists of three steps: position and velocity initialization, velocity update according to Eq. (2), and position update according to Eq. (1). The parameters w, c_1 and c_2 are chosen by the user at the start of the problem. In Eq. (1), if a particle′s position $\mathbf{x}_i^{t+1}$ is driven outside of its allowed range by a large velocity change, its updated position is replaced by the value at the boundary.

The PSO algorithm we just introduced is the so-called global best PSO where the social knowledge used in the velocity update is the global best of the entire swarm. Another kind of PSO is the local best PSO [11], where a swarm is divided into several smaller but overlapping neighborhoods so that more diverse solution spaces can be explored and the global optimal can be reached. For local best PSO, the $\mathbf{gbest}^t$ in Eq. (2) is replaced by the best position found within the neighborhood of the particle. The overlap of the neighborhoods facilitates the spread of the global information among them. In this study, the global best PSO is employed as the base algorithm in multi-objective optimization due to its simplicity and faster convergence.

In order to extend the PSO to solve MO problems, the single global best $\mathbf{gbest}^t$ is extended into a fixed-sized archive of non-dominated solutions accumulated during the search process. Several different MOPSO algorithms have been developed over the years [11]. Most efforts go to the maintenance of the archive. With a large archive, the *dominated tree* [15] might become an efficient data structure for archive management. To simplify the algorithm, a small archive size equal to the swarm size is chosen for our MOPSO algorithm. The algorithm can be summarized as

Algorithm 2. MOPSO.

- Position and velocity initialization.
- Initialize each particle′s personal best.
- Store the non-dominated solutions into the global best archive.
- Repeat the following until the maximum number of iterations is reached or a convergence criterion is met.