Annals of Nuclear Energy 37 (2010) 1263–1271

Contents lists available at ScienceDirect

Annals of Nuclear Energy

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

PWR fuel management optimization using continuous particle swarm intelligence

F. Khoshahval, A. Zolfaghari *, H. Minuchehr, M. Sadighi, A. Norouzi

Nuclear Engineering Department, Shahid Beheshti University, G.C, P.O. Box: 1983963113, Tehran, Iran

ARTICLE INFO

Article history:
Received 22 February 2010
Received in revised form 24 May 2010
Accepted 31 May 2010
Available online 23 June 2010

Keywords:
Fuel management
Optimization
PSO
WWER-1000

ABSTRACT

The objective of nuclear fuel management is to minimize the cost of electrical energy generation subject to operational and safety constraints. In the present work, a core reload optimization package using continuous version of particle swarm optimization, CRCPSO, which is a combinatorial and discrete one has been developed and mapped on nuclear fuel loading pattern problems. This code is applicable to all types of PWR cores to optimize loading patterns. To evaluate the system, flattening of power inside a WWER-1000 core is considered as an objective function although other variables such as K_{eff} along power peaking factor, burn up and cycle length can be included. Optimization solutions, which improve the safety aspects of a nuclear reactor, may not lead to economical designs. The system performed well in comparison to the developed loading pattern optimizer using Hopfield along SA and GA.

1. Introduction

Nuclear fuel management involves making the so called the out-of-core and in-core decisions. The out-of core decisions address the attributes of the fresh fuel that will be fabricated and partially burnt fuel to reinsert into the core for additional energy generation. The in-core decisions address where the fresh and burnt fuel along with burnable poisons should be located in the core. The core reload design process encompasses a complex set of decisions, spread over a period of time, with the final goal of specifying a core capable of producing a demand-imposed target cycle energy, at the minimum cost, with appropriate margins to assure that given acceptable fuel design limits are not exceeded during any condition of normal operation, including the effects of anticipated operational occurrences. These margins are detailed in a reactor's final safety analysis report, FSAR. In recent decades there has been an effort to establish computational capability to assist the reload core decision. Several techniques have been developed such as dynamic programming (Wall and Fenech, 1965; Stout and Robinson, 1973), direct search (Motoda et al., 1975), variational techniques (Terney and Williamson, 1982), backward diffusion calculation (Chao et al., 1986), reverse depletion (Downar and Kim, 1986), linear programming (Okafor and Aldemir, 1988; Stillman et al., 1989), simulated annealing (Parks, 1990; Kropaczek and Turinsky, 1991; Smuc et al., 1994; Mahlers, 1995), hopfield neural network along simulated annealing (Sadighi et al., 2002a,b; Fadaei and setayeshi, 2008), genetic algorithms (Yamamuto, 1997), knowledge-based systems (Galperin and Nissan, 1988; Galperin et al., 1989; Galperin and Kimhy, 1991), particle swarm optimization (Meneses et al., 2009; Babazadeh et al., 2009). None of these optimization approaches ensures the global optimum solution because of the limitations of their search algorithms; they can only find near optimum solution (Kim et al., 1993a,b). While these techniques have been successfully implemented to solve loading pattern in PWRs, each has inherent drawbacks. Recent researches show the stochastic methods such as simulated annealing (SA), genetic algorithm (GA) and particle swarm optimization (PSO) seem to be more suitable owing to the evolution in powerful computer hardware.

Poon and parks (1993) compared GAs to simulated annealing, it was found that GA optimization worked better in the initial global search, but simulated annealing was better for local search. Babazadeh et al. (2009) applied discrete version of PSO algorithm to the optimization of the fuel core loading pattern in nuclear power reactors. In this paper the continuous version of PSO is exploited and implemented. The method is simple, uncomplicated and it converges quickly.

Optimization goals vary: maximization of burn up or excess reactivity minimization, minimization of power peaking, cycle costs, etc. In this work flattening of the power distribution is selected as an objective function. Besides power peaking factors other parameters as well as their combination could be implemented in the developed software as an objective function. In spite of significant efforts devoted to the problem, no standard methods, with industry-wide acceptance and adequate compromise between simplicity and accuracy, are in general use to generate acceptance candidate core reload patterns meeting realistic optimizing criteria.

* Corresponding author. Tel.: +98 21 22431596; fax: +98 21 29902546.
E-mail address: a-zolfaghari@sbu.ac.ir (A. Zolfaghari).

2. Swarm intelligence

Swarm intelligence is an attempt to design algorithms or distributed problem solving devices inspired by the collective behavior of social insects and other animal societies (Bonebeau et al., 1999). Particle swarm optimization (PSO) and ant colony optimization (ACO) are the most popular optimization frameworks based on the original notion of swarm intelligence. They are based on the repeated sampling of solutions to the desired problem which means each agent provides a solution (Unler, 2008). PSO is a population-based stochastic approach for solving continuous and discrete optimization problems. PSO was first introduced by Kennedy and Eberhart in 1995, which is inspired by social behavior of bird flocking and fish schooling. PSO has many similarities with genetic algorithm (GA). Both GA and PSO are similar in the sense that both techniques are population-based search but employ different strategies and computational efforts to find best solution. For example both of them start their process with an initial population but in PSO there are no evolution operators. Furthermore, it does not require any cross over or mutation and this is one of the most benefits of the PSO. In addition, its memory and CPU speed requirement are low (Eberhart et al., 1996). In PSO, a set of randomly generated solutions propagates in the design space towards the optimal solution over a number of iterations based on large amount of information about the design space that is assimilated and shared by all members of the swarm.

2.1. Continuous PSO system

Two basic approaches for continuous PSO formulation are used: inertia weight approach, IWA, and constricted factor approach, CFA.

2.1.1. Inertia weight approach

In continuous PSO, each particle moves in the search space with a velocity according to its own previous best solution and their group previous best solution. The position of a particle represents a candidate solution to treat the optimization considered problem. The following equations are used to iteratively modify the particle velocities and positions at each time step, i.e.:

$$v_{id}^{t+1} = w^t v_{id}^t + c_1 r_1^t (pbest_{id} - x_{id}^t) + c_2 r_2^t (gbest_d - x_{id}^t) \quad (1)$$

$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \quad (2)$$

where $i = [1, 2, \ldots, n]$, $d = [1, 2, \ldots, m]$, n = number of particles in a group, m = elements of particle vectors, and v_{id}^t = velocity of the particle at time step t, x_{id}^t = position at time step t, c_1, c_2 = acceleration constants, r_1, r_2 = random number between 0,1, w^t = inertia weight at time step t, $pbest_{id}$ = previous best position of the particle at time step t. (sometimes called best neighbor), $gbest_d$ = best position among all particles at time step t, t = current iteration.

It can be observed the new particle position is obtained by adding the particle's current position and the new velocity using Eq. (1). The investigations show that the continuous PSO has more chance to "fly" into the better solutions region and converge quickly (Khoshahval, 2009), so it can discover reasonable quality solution faster than other evolutionary algorithms. One of the drawbacks of this method is parameter dependency of the method. The continuous PSO parameters are very important since they have significant impact on optimization results. Therefore, a sensitivity analysis on the continuous PSO parameters, especially on the number of particles and maximum iteration, is carried out in this paper. Considering standard PSO for continuous optimization, one usually obtains reasonable results using acceleration constants in such a way that $c_1 + c_2 = 4.1$ (Clerc and Kennedy, 2002), therefore the number of parameters needed by this metaheuristic is quite reduced. An analogous approach is carried out to obtain proper acceleration constants in this paper and it is found that, taking $c_1 = c_2 = 2$ are proper choice. The inertia weight could be picked out as a constant value but it is preferred to define a variable w which decreases linearly during a run, i.e.:

$$w^t = w_{\max} - \left(\frac{w_{\max} - w_{\min}}{t_{\max}} \right) t \quad (3)$$

where $w_{\max}$ is initial weight, $w_{\min}$ is final weight, $t_{\max}$ is maximum iteration (generation) number and t is current iteration number.

The value of the $w_{\max}$, $w_{\min}$, $t_{\max}$ are chosen through the evaluation of the outputs by running the program a few times. Following a survey, we used $w_{\max} = 0.8$ and $w_{\min} = 0.4$. In our program two different values of 80 and 100 are selected for $t_{\max}$ to compare the effect of the iteration number on the results. However, Eberhart and Shi (1998a,b) have illustrated the following parameters are appropriate and do not depend on problems, i.e.:

$$c_i = 2.0, \quad w_{\max} = 0.9, \quad w_{\min} = 0.4$$

Original formulation of continues PSO which developed in 1995 did not include any inertia weight as a coefficient for the velocity v_{id}^t. In the early versions of continues PSO, $V_{\max}$ (a limitation factor for velocity of each particle) had a great significance, the motivation to eliminate the need for $V_{\max}$ lead to a more appropriate version and introduce a weighting factor. To sum up, PSO using (1), (3) is called inertia weight approach, IWA. This method was first reported in the literature in 1998 (Eberhart and Shi, 1998a,b).

2.1.2. Constricted factor approach

In contrast, another development of continuous PSO was undertaken by Clerc and Kennedy (2002) which is called constricted factor approach, CFA, of PSO. A detailed discussion of the constricted factor is beyond the scope of this paper, but in a simplified form the velocity is defined by,

$$V_{id}^{t+1} = cfk(V_{id}^t + c_1 \cdot r_1^t (pbest_{id} - x_{id}^t) + c_2 \cdot r_2^t (gbest_d - x_{id}^t)) \quad (4)$$

where cfk is a function of c_1 and c_2, i.e.:

$$cfk = \frac{2}{\left| 2 - \phi - \sqrt{\phi^2 - 4\phi} \right|}, \quad \text{where } \phi = c_1 + c_2, \ \phi > 4 \quad (5)$$

The convergence characteristic of the system can be controlled by ϕ. In addition the system can search different regions efficiently by avoiding premature convergence (Lee and El-Sharkawi, 2008).

Both IWA and CFA methods are similar except that CFA utilizes a different expression to work out of velocity. Unlike other evolutionary computation methods, PSO with CFA ensures the convergence of the search procedures (Fukuyama, 2000). In the present work, both of these methods are mapped, used and compared for the fuel loading pattern problem of a nuclear reactor core.

3. Neutronic calculation

Reactor physics calculation provides the basic information for in-core fuel management analysis. The major objective of these neutronic calculations is the prediction of core parameters such as reactivity, power density, macroscopic cross sections and burn up. Well developed neutronic codes are available to perform analysis of LWR cores in detail. In our developed software, CRCPSO, the core calculation is performed by CITATION LDI-2 code (Fowler, 1999). CITATION has been developed to solve multi-group diffusion equation in 3-D. In addition, WIMS-D4 code (Winfrith, 1985) is used to generate average group constants for different fuel assemblies (Table 1). A major feature of the above computer codes is their robustness. They are in widespread use and they have been benchmarked against WWER reactor cores experimental data (Faghihi et al., 2007).