Pressurized water reactor in-core nuclear fuel management by tabu search

Natasha J. Hill[a], Geoffrey T. Parks[a,*]

[a] *Department of Engineering, University of Cambridge*
Cambridge, CB2 1PZ, United Kingdom
Phone: +44 1223 332600
Fax: +44 1223 332662

Abstract

Optimization of the arrangement of fuel assemblies and burnable poisons when reloading pressurized water reactors has, in the past, been performed with many different algorithms in an attempt to make reactors more economic and fuel efficient. The use of the tabu search algorithm in tackling reload core design problems is investigated further here after limited, but promising, previous investigations. The performance of the tabu search implementation developed was compared with established genetic algorithm and simulated annealing optimization routines. Tabu search outperformed these existing programs for a number of different objective functions on two different representative core geometries.

Keywords: Tabu search, Reload core design, Pressurized water reactor

1. Introduction

The design of pressurized water reactors (PWR) reload cores is a formidable combinatorial optimization problem. The designer's task is to find the configuration of fresh and partially burnt fuel and burnable poisons (BPs) that optimizes the performance of the reactor over the subsequent cycle, while ensuring that various operational constraints are satisfied. Such problems have a number of different possible objectives, constraints and many local optima (Galperin, 1995).

Over the years this problem has been tackled in many different ways. Naft and Sesonske (1972) sought to minimize the ratio of peak-to-average power by direct search using heuristic shuffling rules. Federowicz and Stover (1973) also tried to minimize power peaking by successive application of integer linear programming. Ahn and Levine (1985) used a gradient projection method and linear programming in a series of stepwise optimization calculations to minimize the cost of the reload core. Hobson and Turinsky (1986) coupled a first-order accurate perturbation theory model to a Monte Carlo integer programming algorithm to

*Corresponding author
Email address: gtp10@cam.ac.uk (Geoffrey T. Parks)

search for loading patterns (LPs) that maximized the energy production over a cycle, subject to constraints on power peaking and fuel burn-up. Kim et al. (1987) developed a two-stage procedure for maximizing cycle length, subject to power peaking constraints, by decoupling the fuel and BP placement problems. Stillman et al. (1989) used the backward diffusion calculation (Chao et al., 1986) and successive linear programming to determine theoretically optimal fuel and two-dimensional (2D) power distributions for a PWR, minimizing fissile material and BP inventories. Kropaczek and Turinsky (1991) combined the simulated annealing (SA) stochastic optimization technique with a core physics model based on second-order accurate generalized perturbation theory (GPT) to find near-optimal LPs for a variety of different objectives and constraints.

Since the pioneering work of Kropaczek and Turinsky (1991), other researchers, including Mahlers (1994), Šmuc et al. (1994) and Stevens et al. (1995), have developed SA variants to optimize PWR LPs or applied other stochastic/heuristic optimization methods to this problem and/or the closely related boiling water reactor (BWR) LP optimization problem. These other methods have included: genetic algorithms (GAs) (Poon and Parks, 1993, DeChaine and Feltus, 1995, Chapot et al., 1999, François and López, 1999, Ortiz and Requena, 2004, Martín-del-Campo et al., 2004); estimation of distribution algorithms (Jiang et al., 2006); ant colony optimization (De Lima et al., 2008, Esquivel-Estrada et al., 2011, Wang and Lin, 2009, Lin and Lin, 2012); particle swarm optimization (Alvarenga de Moura et al., 2009, Khoshahval et al., 2010, Liu and Cai, 2012); and harmony search (Poursalehi et al., 2013).

A couple of studies have previously investigated the performance of tabu search (TS) on PWR reload core design problems (Lin et al., 1998, Ben Hmaida et al., 1999). These both considered the problem of minimizing the power peaking factor, identifying small improvements in performance compared to a GA implementation. TS implementations have also been applied to various BWR applications: fuel lattice design (François et al., 2003), reload core design (Castillo et al., 2004), control rod design (Castillo et al., 2005) and a combination of fuel loading and control rod pattern optimization (Castillo et al., 2007).

This paper investigates the performance of a TS implementation on representative PWR reload core design problems, seeking optimal values for the parameters that control the algorithm for a range of different objective functions, and then comparing the performance of the resulting TS implementation with that of established SA and GA implementations.

2. Tabu search

Originally developed by (Glover and McMillan, 1986), TS is a meta-heuristic algorithm based on local (or neighborhood) search which has found wide application (Glover and Laguna, 1997), particularly for combinatorial optimization problems. Meta-heuristic algorithms iteratively try to improve the solution but cannot guarantee that the optimum is ever found.

TS evaluates a set of solutions which are, by some definition, next to the current solution and moves to the best of these solutions, even if the objective function value deteriorates as a result of the move. A short-term memory (or *tabu list*) is used to store the most recently visited solutions, and these are not allowed to be revisited for a number of iterations equal to the *tabu tenure*. This feature allows the search to escape from local optima.

Intensification and *diversification* are two further strategies employed in many TS implementations when the progress of the search slows. These rely on medium-term and long-term memories. The medium-term memory (MTM) stores a selection of the best solutions visited in the search. The long-term memory (LTM) records information on how frequently different regions of the search space have been visited.

The aim of *intensification* is to more thoroughly explore the search space close to the locations of the best solutions found. When *intensification* is performed, the search is returned to a solution determined by those in the MTM and search parameters can be adjusted.

Diversification aims to visit insufficiently explored regions of the search space. A random solution in an infrequently visited region (identified using the LTM) is selected and the search is restarted from there. A rudimentary *diversification* strategy does not use a LTM and instead just restarts from random locations in the search space.

3. PWR reload core design

A typical PWR core contains 193 fuel assemblies arranged with quarter-core (reflective or rotational) symmetry. At each refueling between one third and one quarter of these may be replaced. It is common practice for fresh fuel assemblies to carry BPs. It is also usual to rearrange old fuel in order to improve the characteristics of the new core. This shuffling can entail the exchange of corresponding assemblies between core quadrants, which is equivalent to changing the assembly 'orientations', or the exchange of different assemblies, which changes their locations and possibly their orientations also. Examples of each exchange are shown in Fig. 1.

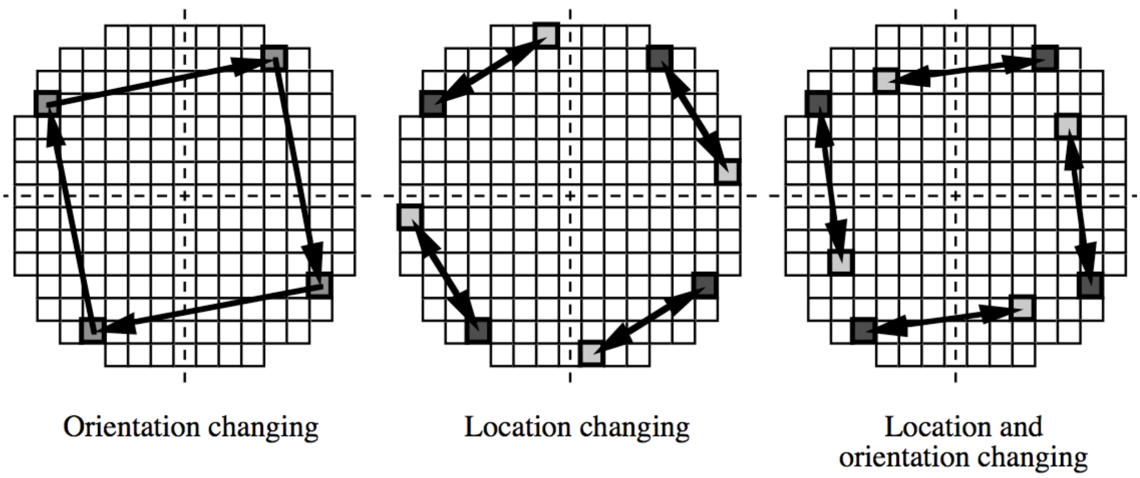


Fig. 1. Typical fuel assembly exchanges.

Thus, a candidate LP of predetermined symmetry must specify:

- the fuel assembly to be loaded in each core location,
- the BP loading with each fresh fuel assembly, and
- the orientation of each burnt assembly.

One interesting point to emerge from a review of past work on PWR reload core design is the diversity in objective functions chosen for optimization. These have included:

1. Maximization of end-of-cycle (EOC) reactivity
2. Maximization of discharge burn-up
3. Minimization of feed enrichment
4. Minimization of power peaking
5. Minimization of the fresh fuel inventory

An effective LP optimization method should ideally work well for any objective function chosen by the user, rather than enforce a choice of objective function on the user.

4. Algorithm implementation

4.1. Framework

The TS implementation was developed within the optimization framework provided in the Fuel Optimization for Reloads: Multiple Objectives by Simulated Annealing for PWRs (FORMOSA-P) nuclear fuel management optimization code (Kropaczek and Turinsky, 1991, Kropaczek et al., 1994, Maldonado et al., 1995). The original version of FORMOSA-P combined SA-based optimization with a 2D (radial) nodal expansion method simulator coupled with an assembly power response GPT LP evaluator. A GA implementation was subsequently added to FORMOSA-P (Poon and Parks, 1993, Parks, 1996).

Within FORMOSA-P each candidate LP is represented by three 2D arrays, corresponding to the physical layout of the fuel assemblies (with identifiers indicating unique fresh or burnt fuel designs), their BP loadings (with identifiers indicating individual options from the range available) and their orientations, respectively, as shown in Fig. 2.

Fuel Assemblies

1	32	6	19	21	20	29	32
32	17	23	10	32	12	32	32
6	24	22	32	13	32	28	32
19	9	32	16	25	31	32	8
21	32	14	26	15	32	4	.
20	11	32	30	32	18	3	.
29	32	27	32	5	2	.	.
32	32	32	7	.	.	.	.

Burnable Poisons

0	1	0	0	0	0	0	0
1	0	0	0	1	0	3	0
0	0	0	2	0	3	0	0
0	0	2	0	0	0	0	0
0	1	0	0	0	0	0	.
0	0	3	0	0	0	0	.
0	3	0	0	0	0	.	.
0	0	0	0	.	.	.	.

Orientations

0	0	3	1	1	1	2	0
1	3	3	3	0	3	0	0
0	1	2	0	3	0	3	0
2	1	0	2	2	1	0	0
2	0	1	2	2	0	2	.
2	1	0	3	0	0	1	.
3	0	1	0	2	3	.	.
1	0	0	0	.	.	.	.

Fig. 2. A representation of a loading pattern (with rotational quarter-core symmetry).

For combinatorial optimization problems such as PWR reload core design, application-specific crossover operators are required in GA implementations to guarantee that valid offspring are produced; in this case, to ensure that the fuel assembly inventory is maintained. The FORMOSA-P GA implementation uses Poon and Parks' heuristic tie-breaking crossover (HTBX) operator (Poon and Parks, 1993). The HTBX maps the parent fuel assembly arrays to reactivity-ranked arrays based on the assemblies' beginning-of-cycle (BOC) reactivities. It then combines randomly selected complementary parts of these arrays through a 'cut and paste' operation and uses a simple tie-breaking algorithm to produce valid offspring reactivity-ranked arrays. Finally, the assembly-ranking mapping is reversed to produce the offspring assembly LPs. The BP loadings and assembly orientations are all inherited from one or other parent. Thus, the BOC reactivity distribution of an offspring LP resembles, but is not necessarily identical to, parts of both parents. The performance comparisons presented in Sections 6.2 and 6.3 are, of course, specific to this GA implementation.

The mutation operator from the FORMOSA-P GA implementation is used extensively in our TS implementation. The mutation operator performs a binary exchange of fuel assemblies and randomly changes the BP loading and orientation of the two fuel assemblies from within the ranges of values for these parameters allowed by the specified core symmetry and geometry and fuel and BP inventories and options.

The objective functions and constraints are handled in the same way as in FORMOSA-P and the reactor core analysis is also performed using GPT (Kropaczek et al., 1994, Maldonado et al., 1995). Four objective functions are available:

1. Maximization of the EOC soluble boron concentration (equivalent to maximizing the EOC reactivity)
2. Minimization of the radial power peaking
3. Maximization of the discharge burn-up
4. Minimization of the enrichment of fresh fuel

Within FORMOSA-P the calculation of each of the objectives is of the form:

$$f_{\mathrm{pen}} = (-)f_{\mathrm{raw}} + c - c_{\mathrm{ref}} \qquad (1)$$

where f_{raw} is the raw objective function (suitably scaled), the factor of -1 [the $(-)$ term] is included if the original objective function is to be maximized, c is a term quantifying the extent of constraint violation for the current solution (candidate LP), c_{ref} is a term quantifying the extent of constraint violation for the original, reference LP, and f_{pen} is thus a suitably penalized objective function to be minimized.

4.2. Tabu search implementation

In our TS implementation, before the search begins, a random starting LP is found by taking the user-specified reference LP and then mutating it 1000 times. The resulting LP is then evaluated, and if it is grossly infeasible (as defined in FORMOSA-P), then the mutation process is repeated until a suitable (not grossly infeasible) starting LP is found. An LP is defined as grossly infeasible if it violates one or more of a number of possible user-defined constraints, such as maximum acceptable radial power peaking, maximum acceptable feed enrichment, maximum acceptable soluble boron concentration etc. This search initialization feature means that the starting LP for individual optimization runs depends on the random number generator seed specified, and is helpful when conducting performance comparisons between different optimization strategies. It can easily be suppressed if the user wants to run an individual optimization using the reference LP as the starting LP.

A basic TS local search iteration proceeds as follows:

1. A neighbor is generated by mutating the current LP once. This neighbor is then evaluated. If it is grossly infeasible or classed as tabu (by comparison with LPs in the tabu list), it is discarded; otherwise it is stored.
2. This process is repeated until the desired neighborhood size (number of stored LPs) has been generated.
3. The best of these neighbors is selected and replaces the current LP. This LP is added to the tabu list. If it meets the criteria for being added to the MTM, it is also added to this.
4. If the criteria for diversification or intensification are not met, then the process repeats, selecting neighbors of this new LP.

This process is terminated when a maximum number of LPs have been evaluated.

The tabu list is an array of the most recently selected LPs, with the number of elements equal to the tabu tenure. When a new LP is selected, it is added to the tabu list, and, if the list is full, another LP is removed on a first-in first-out basis.

The MTM records a fixed number of the best LPs visited, along with their objective function values. When a new LP is selected, its objective function value (f_{pen}) is compared with those of the LPs in the MTM. If it is better than any of the existing MTM LPs, it is added and the worst LP (that with the highest f_{pen} value) is removed from the MTM.

A counter is incremented if a TS local search iteration does not result in the identification of a new best LP and reset to zero when a new best LP is found. Diversification takes place when this counter reaches a user-specified value. When the search is diversified, the current LP is mutated 500 times in order to create a random new LP to search from. The search is then restarted from this LP.

Initially intensification was also designed to take place when the counter of the number of consecutive non-improving iterations reached a user-specified value, a different value to that used for diversification. At intensification the neighborhood size was increased by a factor and the search returned to a randomly selected LP from the MTM. The neighborhood size is reset to its initial value at the next diversification.

This implementation requires a number of parameters to be chosen by the user:

- The neighborhood size
- The tabu list length
- The number of consecutive non-improving iterations when diversification is performed
- The number of consecutive non-improving iterations when intensification is performed
- The MTM size
- The factor by which the neighborhood size is increased during intensification

Optimal values for these parameters were investigated. The results of these investigations can be found in Section 6.1.

A different intensification timing method was also implemented and tested, for reasons that are explained in Section 6.1. This consisted of performing one intensification stage at a specified iteration number in the search. This iteration number, of course, represents another parameter to be specified by the user.

5. Testing protocol

As explained in the previous section, our TS implementation has stochastic elements, as do the methods with which it will be compared. The outcomes of individual runs of stochastic optimization methods depend on the random number generator seed specified. Therefore to compare the performance of different stochastic optimization methods or different configurations of the same stochastic optimization method performance must be measured across a number of runs (in which only the random number generator seed is varied) to draw meaningful conclusions.

Figure 3 shows how the mean and standard deviation over a number of runs of the objective function values of the best LP found after 50 000 objective function evaluations varies for one particular implementation of our TS algorithm as the number of runs increases. This test was repeated with a number of different set-ups (algorithm configurations and objective function choices) and similar results were seen. Based on these results, a sample size of 50 runs was chosen as providing adequately converged measures of the mean and standard deviation of the objective function to allow the comparison of optimization methods and configurations.

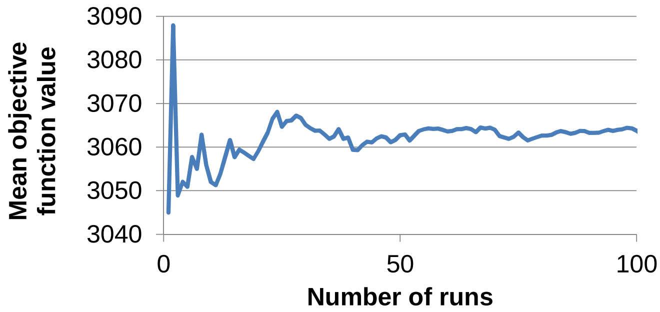
6. Results and discussion

As described in Sections 1 and 3, a number of different methods have been used to optimize PWR LPs with respect to a number of different objective functions. The FORMOSA-P code offers a range of objective function options and a choice of established SA and GA implementations. In fact, three SA implementations are available: SA1 – 'global' SA search; SA2 – 'local' SA search; SA3 'traditional' SA search.

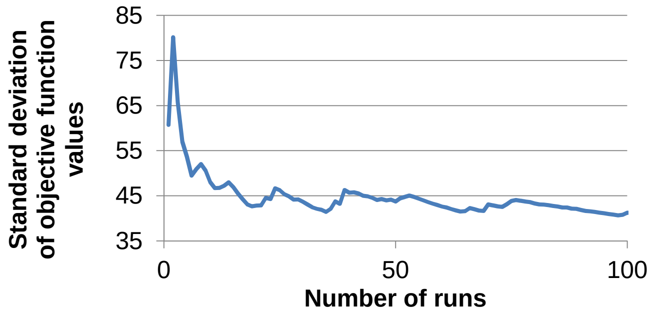
Two test problems were constructed. The first sought optimal LPs for a 3-loop Westinghouse PWR with eighth-core symmetry. The second sought optimal LPs for a 4-loop Westinghouse PWR with only quarter-core symmetry. The reference LPs for both problems are shown schematically in Fig. 4. The second problem has a much larger search space, due both to the larger core size and the lower degree of symmetry specified.

Once a basic implementation of the TS algorithm had been developed, the effects of varying a number of the algorithm's control parameters were investigated in order to find an optimal set of parameters for Problem 1. The results of these experiments are presented in Section 6.1.

The performance of the TS algorithm with this optimal set of parameters was then compared to that of the GA and SA implementations in FORMOSA-P for this problem. The results of this investigation are presented and discussed in Section 6.2.



(a) Mean objective function values



(b) Standard deviation of objective function values

Fig. 3. Convergence of performance measures with the number of optimization runs executed.

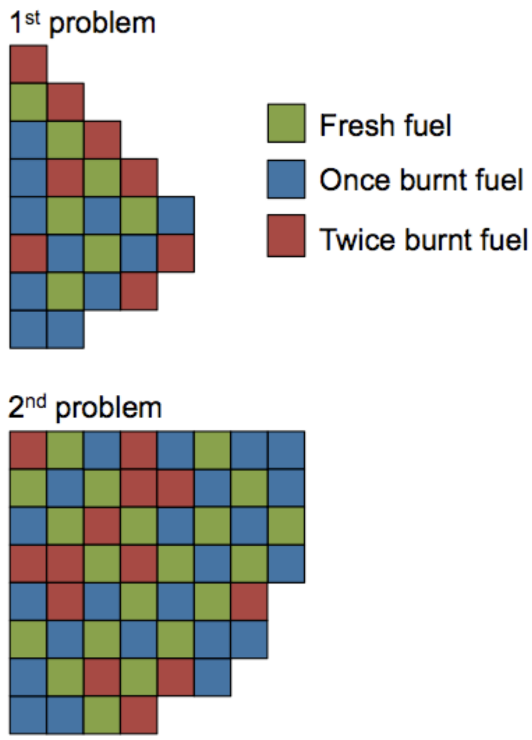


Fig. 4. Fuel maps of the reference LPs for the two problems considered.

The performance of the same TS algorithm configuration was then compared with the FORMOSA-P algorithms for Problem 2. The results of this investigation are summarized in Section 6.3.

These tests were conducted using all four of the objective function options available in FORMOSA-P:

1. Maximization of the EOC soluble boron concentration
2. Minimization of the radial power peaking
3. Maximization of the discharge burn-up
4. Minimization of the fresh fuel enrichment

6.1. Parameter identification

6.1.1. Neighborhood size

In this experiment intensification and diversification strategies were not used and the tabu list length was set to 10. As discussed in Section 6.1.3, subsequent testing showed that neither intensification nor diversification have a significant effect on the performance of the TS implementation. If this had not been the case, then it would, of course, have been straightforward to conduct an iterative investigation in which the neighborhood size was optimized for a TS implementation without intensification and diversification, next those strategies were optimized for that neighborhood size, and then the neighborhood size was optimized for the implementation with 'optimal' intensification and diversification strategies, and so on recursively until convergence in the parameter settings was achieved.

For samples of 50 runs, each 50 000 objective function evaluations long, the mean and standard deviation of the objective function value were compared for various neighborhood sizes. Figure 5 shows the results of this experiment for objective function 2 (minimization of radial power peaking). The performance improves (lower mean and lower standard deviation) for increasing neighborhood size up to 9, and deteriorates beyond 15. Between 9 and 15 the mean improves (reduces) but there is a small increase in variability. This increase in variability was deemed acceptable and a value of 15 was therefore chosen as the neighborhood size for this objective.

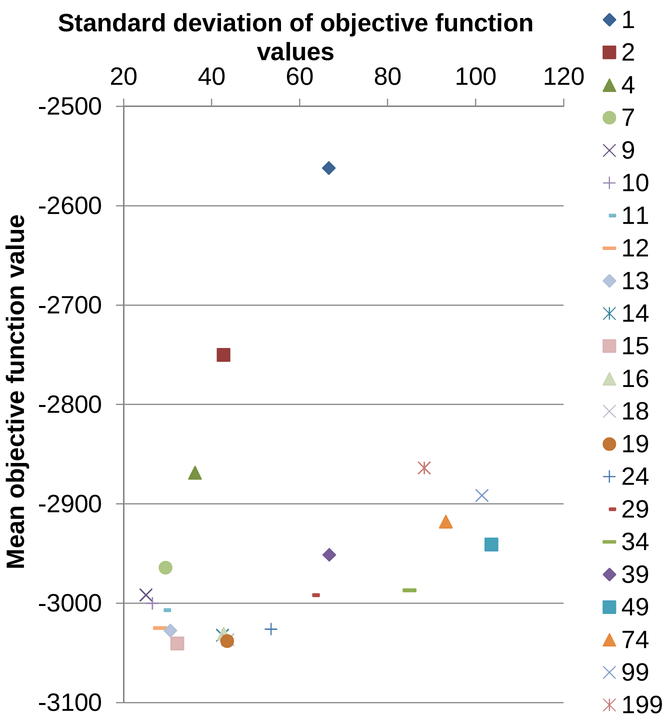


Fig. 5. Effect of neighborhood size (Objective 2).

This approach was repeated for the other three objective functions. The optimal values of the neighborhood size were found to be different for the four objective functions, as summarized in Table 1.

Although this investigation showed that the optimal value of the neighborhood size varied for different objective function choices, the tests also showed that algorithm performance was reasonably robust for choices of this parameter near the optimal value. For subsequent tests, it was therefore decided to test the performance of the algorithm for a single value of neighborhood size which gives reasonably good performance for all four objective functions.

Table 1. Best neighborhood size for each objective function on Problem 1.

Objective	Best neighborhood size
1	9
2	15
3	19
4	37

The value judged to be best for this was 15. Using a single value for this parameter, rather than an objective-specific one, has the advantage of making the algorithm easier to use "out of the box".

6.1.2. Tabu list length

To determine an appropriate tabu list length, and hence tabu tenure, 50 runs of 50 000 objective function evaluations were performed for each objective with a number of different tabu list lengths. The same sets of random number generator seeds were used in each case so that runs for a given seed would be identical with the different tabu tenures unless cycling (that is returning to recently visited solutions) occurred. Cycling is detrimental to performance as it wastes search time. It was found that in almost all cases a tabu tenure of 10 was sufficient to prevent cycling and therefore this value was chosen. Values less than 10 produced more instances of cycling.

6.1.3. Intensification and diversification parameters

Intensification and diversification were investigated with a range of different control parameters. However, neither strategy was found to significantly improve the performance of the TS implementation. Intensification was found to be most effective when applied very close to the end of the search, and thus a different implementation was tested where intensification occurs at a fixed iteration number, near the end of the search. This was found to produce an improvement in performance, but since the length of the search is often not predetermined, it was decided that this strategy should not be used. As such, in the final implementation created, neither intensification nor diversification are used.

6.2. Performance comparison for Problem 1

A TS implementation using the best neighborhood size for each objective (TS), as given in Table 1, and an implementation using a neighborhood size of 15 (TS15) for all objectives, for the reasons discussed in Section 6.1.1, were considered. These were compared to two of the SA implementations (SA1 and SA3) and the GA implementation in FORMOSA-P.

The FORMOSA-P algorithms have parameter values, including the maximum number of objective function evaluations in a run, that are automatically determined based on the size of the problem under consideration, as measured by the number of individual LP perturbations possible. Therefore to compare the performance of the algorithms the mean and standard deviation of the best objective function values found after each objective function evaluation in 50 runs of each algorithm were calculated. When the mean objective function values are plotted, see Fig. 6, it is possible to determine which algorithm is performing best at any point in the search.

The results in Fig. 6 clearly show that TS performs best on average throughout the duration of the search for all four objective functions on this problem. It is also clear that TS reaches good solutions much faster than the other algorithms as the average objective function reduces much faster initially before leveling off.

It is important to also consider the standard deviation of the results. In practice, the optimization would not be repeated many multiple times. Therefore it is important that optimizer performance is reasonably consistent (i.e. that the standard deviation of the results is low). The error bars in Fig. 6 show how the standard deviation of the best objective function value found varies through the search for all of the algorithms tested. It is clear that the standard deviation of the results is lowest throughout the search for TS.

As one would expect, TS performance is best for each objective for the optimal neighborhood size parameter (the TS lines). Figure 6 shows that TS performance for a neighborhood size of 15 (the TS 15 lines) is equally good for Objective 1, and although not quite as good for Objectives 3 and 4, it is nevertheless clearly better than that achieved on these problems by the FORMOSA-P SA and GA implementations. There is no TS 15 line shown for Objective 2 because the optimal neighborhood size is 15 in this case.

6.3. Performance comparison for Problem 2

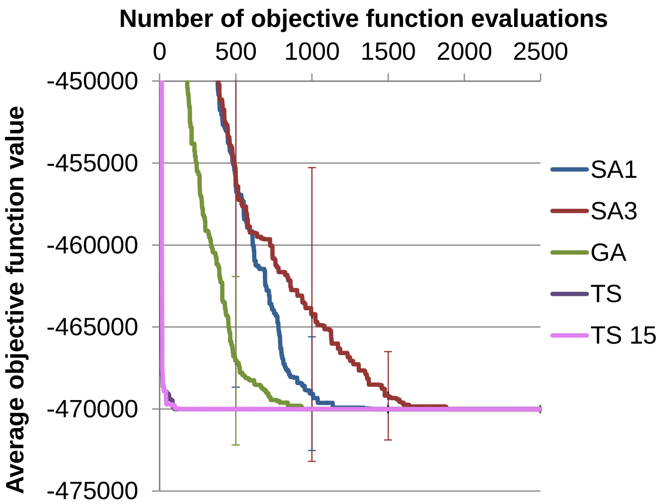
The best TS set-ups found for Problem 1 were used on the much larger (in terms of search space size) Problem 2. It is to be expected that the optimal TS parameters for Problem 2 will be different from those for Problem 1, but it is interesting to see how well TS performs on Problem 2 using the optimal parameters for Problem 1.

As previously mentioned, the FORMOSA-P code automatically changes the SA and GA control parameters such that they are appropriate for the size of the problem being considered, and thus the parameters they use on Problem 2 are different to those used on Problem 1.

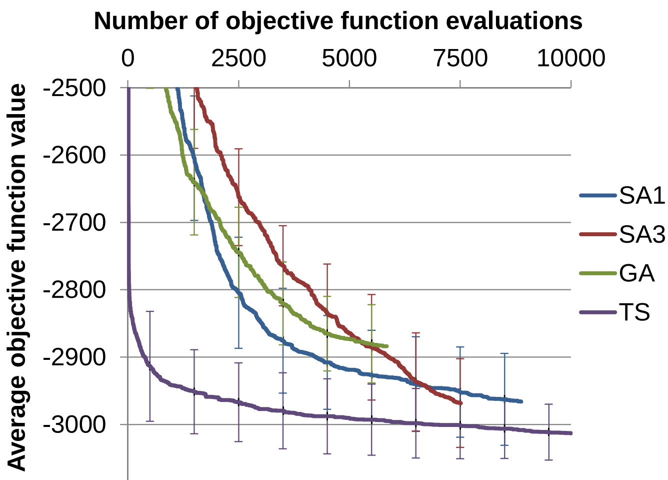
Figure 7 shows the performance of the algorithms on Problem 2. The TS parameters are not optimized for this problem yet it is clear that TS nevertheless outperforms the two versions of SA and the GA implementation in FORMOSA-P by some margin.

For Objective 1 TS with a neighborhood size of 15 now clearly outperforms TS with an optimal (for Problem 1) neighborhood size of 9. For Objectives 3 and 4 TS with the optimal (for Problem 1) neighborhood size (19 for Objective 3, 37 for Objective 4) outperforms TS with a neighborhood size of 15 again. These results imply that while the optimal neighborhood size may well depend on the objective function, it is not obviously a strong function of the problem size. The results for Objective 1 for the two problems imply that a neighborhood size of 15 may overall be a better choice.

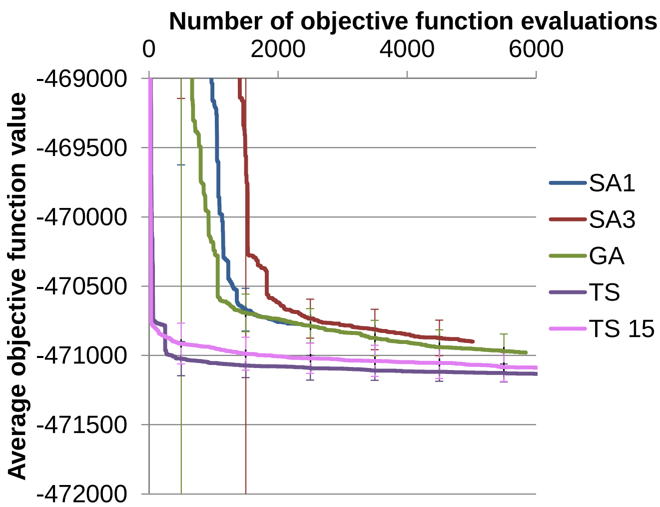
The ability of TS to outperform a GA on PWR reload core design optimization problems observed here is consistent with a similar observation made recently in the context of BWR reload core design by François et al. (2013).



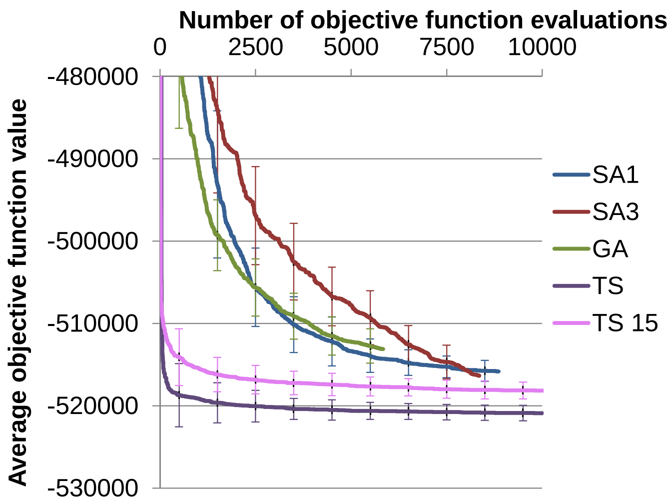
(a) Objective 1



(b) Objective 2

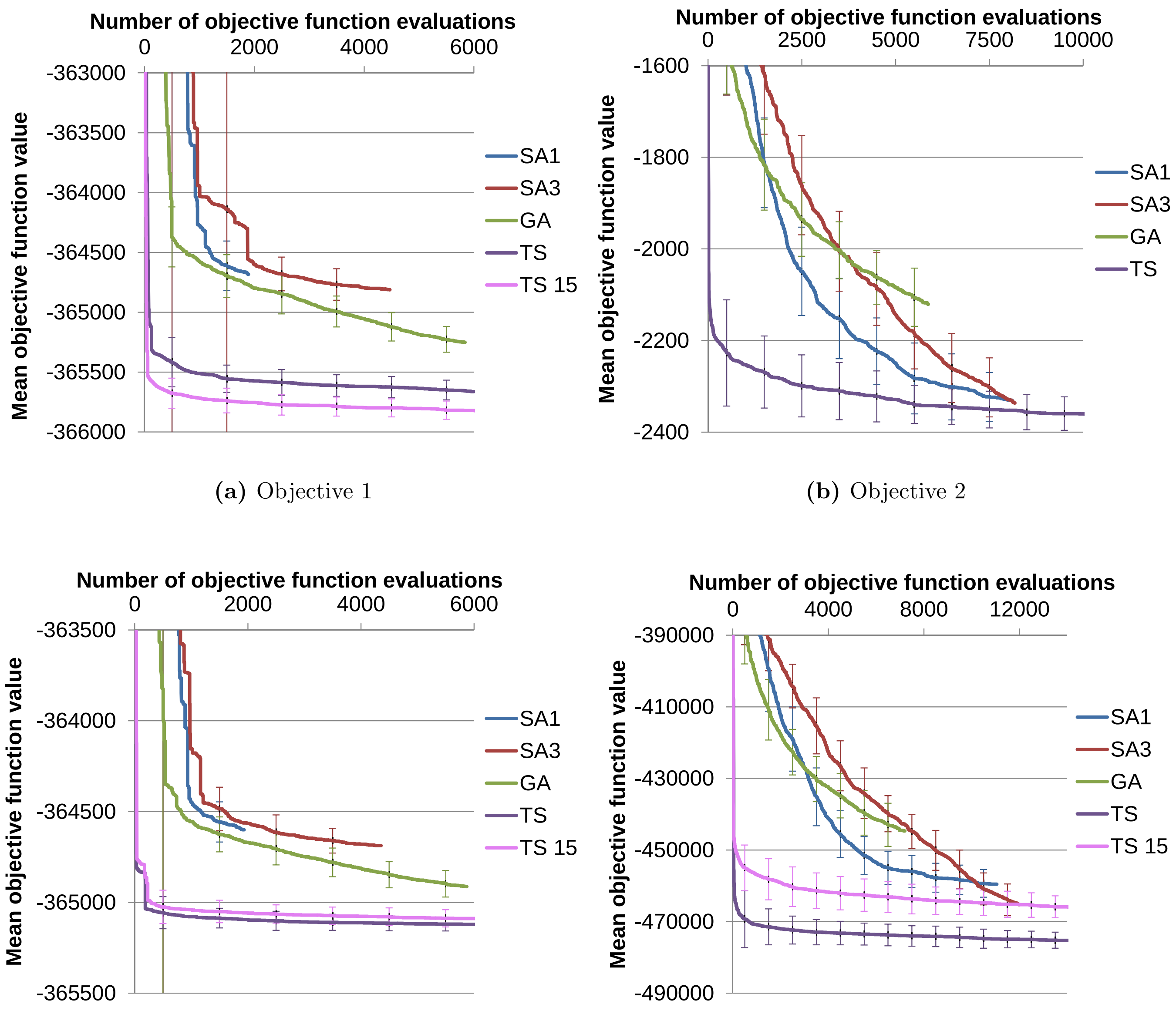


(c) Objective 3



(d) Objective 4

Fig. 6. Comparison of mean objective function values with different optimization algorithms and objective functions for Problem 1, error bars represent one standard deviation.



(a) Objective 1

(b) Objective 2

(c) Objective 3

(d) Objective 4

Fig. 7. Comparison of mean objective function values with different optimization algorithms and objective functions for Problem 2, error bars represent one standard deviation.

7. Conclusions

A TS algorithm for tackling PWR reload core design problems has been implemented within the optimization framework of the FORMOSA-P code and tested on a couple of representative PWR core geometries for four different commonly used objective functions. Testing revealed that the diversification and intensification strategies implemented within the TS algorithm provided negligible performance benefit. The resulting TS implementation therefore has only two control parameters to be determined: the tabu list length and the neighborhood size. Tests showed that a tabu list length of 10 worked well for all four objective functions, but that the optimal neighborhood size did depend on the objective function under consideration.

The performance of the resulting TS implementation was compared with that of the established SA and GA implementations in FORMOSA-P. The TS implementation was found to perform best for both core geometries and all four objective functions. The TS implementation performed better than SA and GA even when using the same neighborhood size for each objective function. This implies that the performance of this TS implementation is reasonably robust to its parameter settings. The ability to perform well "out of the box" is an attractive one for optimizers, as there may not always be the time or expertise available to tune them to the problem at hand.

These findings indicate that our TS implementation is a promising method for solving PWR reload core design problems and worthy of further investigation. Further work that could usefully be undertaken includes the investigation of the method's performance on other representative problems, investigating in particular the question of how sensitive performance is to the choice of neighborhood size for different core geometries and objective functions. Performance comparisons with other state-of-the-art stochastic optimization methods, such as ant colony optimization and particle swarm optimization, would also be instructive.

Acknowledgements

The support of Professor Paul Turinsky of North Carolina State University in allowing the use of the FORMOSA-P code in this research is gratefully acknowledged.

References

Ahn, D., Levine, S., 1985. Automatic optimized reload and depletion method for a pressurized water reactor. Nucl. Technol. 71, 535–547.

Alvarenga de Moura, M., Dornellas, M., Schirru, R., 2009. Particle swarm optimization applied to the nuclear reload problem of a pressurized water. Prog. Nucl. Energy 51, 319–326.

Ben Hmaida, I., Carter, J., De Oliveira, C., Goddard, A., Parks, G., 1999. Nuclear in-core fuel management optimization using the tabu search method, in: Proc. Mathematics & Computation, Reactor Physics and Environmental Analysis in Nuclear Applications, Madrid, Spain. pp. 1658–1666.

Castillo, A., Alonso, G., Morales, L., Martín del Campo, C., François, J., del Valle, E., 2004. BWR fuel reloads design using a Tabu search technique. Ann. Nucl. Energy 31, 151–161.

Castillo, A., Ortiz, J., Montes, J., Perusquía, R., 2007. Fuel loading and control rod patterns optimization in a BWR using tabu search. Ann. Nucl. Energy 34, 207–212.

Castillo, J., Ortiz, J., Alonso, G., Morales, L., del Valle, E., 2005. BWR control rod design using tabu search. Ann. Nucl. Energy 32, 741–754.

Chao, Y., Hu, C., Suo, C., 1986. A theory of fuel management via backward diffusion calculation. Nucl. Sci. Eng. 93, 78–87.

Chapot, J., Da Silva, F., Schirru, R., 1999. A new approach to the use of genetic algorithms to solve the pressurized water reactor's fuel management optimization problem. Ann. Nucl. Energy 26, 641–655.

De Lima, A., Schirru, R., Carvalho da Silva, F., Canedo, J., 2008. A nuclear reactor core fuel reload optimization using artificial ant colony connective networks. Ann. Nucl. Energy 35, 1606–1612.

DeChaine, M., Feltus, M., 1995. Nuclear fuel management optimization using genetic algorithms. Nucl. Technol. 111, 109–114.

Esquivel-Estrada, J., Ortiz-Servin, J., Castillo, J., Perusquía, R., 2011. Azcaxalli: a system based on Ant Colony Optimization algorithms, applied to fuel reloads design in a Boiling Water Reactor. Ann. Nucl. Energy 38, 103–111.

Federowicz, A., Stover, R., 1973. Optimization of PWR loading patterns by reference design perturbations. Trans. Amer. Nucl. Soc. 17, 308–309.

François, J., López, H., 1999. SOPRAG: a system for boiling water reactors reload pattern optimization using genetic algorithms. Ann. Nucl. Energy 26, 1053–1063.

François, J., Martín-del-Campo, C., François, R., Morales, L., 2003. A practical optimization procedure for radial BWR fuel lattice design using tabu search with a multiobjective function. Ann. Nucl. Energy 30, 1213–1229.

François, J., Ortiz-Servin, J., Martín-del-Campo, C., Castillo, A., Esquivel-Estrada, J., 2013. Comparison of metaheuristic optimization techniques for BWR fuel reloads pattern design. Ann. Nucl. Energy 51, 189–195.

Galperin, A., 1995. Exploration of the search space of the in-core fuel management problem by knowledge-based techniques. Nucl. Sci. Eng. 119, 144–152.

Glover, F., Laguna, M., 1997. Tabu Search. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Glover, F., McMillan, C., 1986. The general employee scheduling problem: an integration of MS and AI. Comput. Oper. Res. 13, 563–573.

Hobson, G., Turinsky, P., 1986. Automatic determination of pressurized water reactor core loading patterns that maximize beginning-of-cycle reactivity within power-peaking and burnup constraints. Nucl. Technol. 74, 5–13.

Jiang, S., Ziver, A., Carter, J., Pain, C., Goddard, A., Franklin, S., Phillips, H., 2006. Estimation of distribution algorithms for nuclear reactor fuel management optimisation. Ann. Nucl. Energy 33, 1039–1057.

Khoshahval, F., Zolfaghari, A., Minuchehr, H., Sadighi, M., Norouzi, A., 2010. PWR fuel management optimization using continuous particle swarm intelligence. Ann. Nucl. Energy 37, 1263–1271.

Kim, Y., Downar, T., Sesonske, A., 1987. Optimization of core reload design for low-leakage fuel management in pressurized water reactors. Nucl. Sci. Eng. 96, 85–101.

Kropaczek, D., Turinsky, P., 1991. In-core nuclear fuel management optimization for pressurized water reactors utilizing simulated annealing. Nucl. Technol. 95, 9–32.

Kropaczek, D., Turinsky, P., Parks, G., Maldonado, G., 1994. The efficiency and fidelity of the in-core nuclear fuel management code FORMOSA-P, in: Proc. Int. Conf. on Reactor Physics and Reactor Computations, Tel Aviv, Israel. pp. 572–579.

Lin, C., Lin, B.F., 2012. Automatic pressurized water reactor loading pattern design using ant colony algorithms. Ann. Nucl. Energy 43, 91–98.

Lin, C., Yang, J., Lin, K., Wang, Z., 1998. Pressurized water reactor loading pattern design using the simple tabu search. Nucl. Sci. Eng. 129, 61–71.

Liu, S., Cai, J., 2012. Studies of fuel loading pattern optimization for a typical pressurized water reactor (PWR) using improved pivot particle swarm method. Ann. Nucl. Energy 50, 117–125.

Mahlers, Y., 1994. Core loading pattern optimization for pressurized water reactors. Ann. Nucl. Energy 21, 223–227.

Maldonado, G., Turinsky, P., Kropaczek, D., Parks, G., 1995. Employing nodal generalized perturbation theory for the minimization of feed enrichment during pressurized water reactor in-core nuclear fuel management optimization. Nucl. Sci. Eng. 121, 312–325.

Martín-del-Campo, C., François, J., Avendaño, L., González, M., 2004. Development of a BWR loading pattern design system based on modified genetic algorithms and knowledge. Ann. Nucl. Energy 31, 1901–1911.

Naft, B., Sesonske, A., 1972. Pressurized water reactor optimal fuel management. Nucl. Technol. 14, 123–132.

Ortiz, J., Requena, I., 2004. An order coding genetic algorithm to optimize fuel reloads in a nuclear boiling water reactor. Nucl. Sci. Eng. 146, 88–98.

Parks, G., 1996. Multiobjective pressurized water reactor reload core design by nondominated genetic algorithm search. Nucl. Sci. Eng. 124, 178–187.

Poon, P., Parks, G., 1993. Application of genetic algorithms to in-core nuclear fuel management optimization, in: Proc. Joint Int. Conf. Mathematical Methods and Super-computing in Nuclear Applications, Karlsruhe, Germany. pp. 777–786.

Poursalehi, N., Zolfaghari, A., Minuchehr, A., 2013. PWR loading pattern optimization using Harmony Search algorithm. Ann. Nucl. Energy 53, 288–298.

Stevens, J., Smith, K., Rempe, K., Downar, T., 1995. Optimization of pressurized water reactor shuffling by simulated annealing with heuristics. Nucl. Sci. Eng. 121, 67–88.

Stillman, J., Chao, Y., Downar, T., 1989. The optimum fuel and power distribution for a pressurized water reactor burnup cycle. Nucl. Sci. Eng. 103, 321–333.

Šmuc, T., Pevec, D., Petrović, B., 1994. Annealing strategies for loading pattern optimization. Ann. Nucl. Energy 21, 325–336.

Wang, C.D., Lin, C., 2009. Automatic boiling water reactor loading pattern design using ant colony optimization algorithm. Ann. Nucl. Energy 36, 1151–1158.