McMaster University

Advanced Optimization Laboratory

Title:

Reloading Nuclear Reactor Fuel Using Mixed-Integer Nonlinear Optimization

Author:

A.J. Quist, C. Roos, T. Terlaky, R. van Geemert and J.E. Hoogenboom

AdvOl-Report No. 2000/2

May 2000, Hamilton, Ontario, Canada

Reloading Nuclear Reactor Fuel Using Mixed-Integer Nonlinear Optimization

A.J. Quist (a.j.quist@its.tudelft.nl) and C. Roos (c.roos@its.tudelft.nl)
Faculty of Information Technology and Systems
Delft University of Technology
Delft, The Netherlands

T. Terlaky (terlaky@cas.mcmaster.ca)*
McMaster University, Hamilton, Ontario, Canada

R. van Geemert (r.vangeemert@iri.tudelft.nl) and J.E. Hoogenboom (j.e.hoogenboom@iri.tudelft.nl)
Interfaculty Reactor Institute
Delft University of Technology
Delft, The Netherlands

May 2000

Abstract. A nodal nuclear reactor reload pattern optimization model is solved using mixed-integer nonlinear optimization techniques. Unlike currently used heuristic search methods, this method enables continuous optimization of the amount of Burnable Poisons in fresh fuel bundles in a natural way, which is shown in the first part of the article. The second part treats an algorithmic extension using dedicated cuts in a mixed-integer nonlinear optimization algorithm, which push the optimization towards solutions where local power peaks in parts of the core are avoided.

Keywords: Nonlinear Mixed-integer Optimization

1. Introduction

Fuel shuffling optimization is an important issue in the operation of a nuclear reactor. Usually, a number of fuel bundles is discharged at the end of a fuel cycle (EoC), and the same number of fresh bundles is inserted in the core, while all bundles are reshuffled to a configuration that is optimal with respect to some performance criterion. There are several strategies for reloading. We consider the situation in which at each reloading, the number of discharged bundles is the same, being a fixed fraction of the total number of bundles, *e.g.* one third or one fourth.

Having fixed the number of discharged bundles, it is possible to measure the reactivity of all the fuel bundles at each EoC, and then to

* Large part of the research was done while the author was working at Delft University of Technology

find an optimal pattern using a subset of these bundles together with some fresh bundles. Another approach is to apply the same reloading pattern every year. After repeating this for a number of years, the reactor will reach an equilibrium state, in which (ideally) each cycle has the same characteristics. Throughout this paper, such an equilibrium cycle strategy is used. A recent study comparing this equilibrium strategy to the more usual strategy of successive re-optimization at each EoC is made by Yamamoto [25].

Several optimization criteria can be chosen. One way is to search a loading pattern for which the cycle length will be maximal before the reactor becomes *sub-critical*, i.e. the number of neutrons produced is less than the number of neutrons lost by absorption or leakage from the reactor, which stops the reactor operation. Another approach is to minimize the leakage out of the core. In this paper, the cycle length is fixed and a pattern is searched for which the reactivity of the core at EoC is maximal, measured by a property called the uncontrolled effective multiplication factor. All the stated objectives are in some sense equivalent in an equilibrium cycle [25]: a core with maximal EoC reactivity, also would have the longest possible cycle length before becoming sub-critical. Furthermore, the leakage factor is correlated with the effective multiplication factor.

In the literature different methods to solve the reloading problem can be found. Several papers describe solution methods using genetic algorithms and other evolutionary algorithms ([2], [4], [16], [17]), simulated annealing ([12], [14], [15], [20], [21], [23]), pairwise interchange ([6], [8]), tabu search ([13]) and other neighborhood search heuristics. An overview is presented by [3]. Since the problem can be stated as an assignment problem with nonlinear side-constraints, it can also be viewed as a mixed-integer nonlinear optimization problem, for which linear, nonlinear and integer programming techniques can be applied. Variants using linear and nonlinear programming to solve subproblems have been applied ([22], [24]). Results have been published on a mixed-integer linear programming (MILP) approach ([9], [10]) and also on a mixed-integer nonlinear programming (MINLP) approach ([11], [26]).

In a previous paper [19], we compare a pairwise interchange heuristic to algorithms using MINLP solvers. Some years ago it would have been impossible to solve the whole model as one big nonlinear discrete optimization problem. At present, nonlinear programming (NLP) solvers have been improved to the extent that the results for realistic models compare favorably with combinatorial search heuristics. In addition, with this approach it is possible to make the model more realistic by including additional continuous optimization problems in a natural

manner. These include for example the addition of burnable poisons, as is shown in the current paper, and also the insertion of control rods.

The model developed in the current paper is an extension of the model presented in [19]. In that paper, we showed that a simple nodal optimization model can be solved using standard nonlinear optimization techniques, after transforming it into a suitable optimization model. In the current paper, the model is extended to include the optimization of burnable poison concentration. This is a quite natural step when using nonlinear optimization techniques. We only have to think about a good formulation of the algebraic equations; the optimization technique itself remains the same. Another extension compared to our previous paper is the use of improved discretization of the time-axis. Use of central discretization instead of forward discretization gives much more accurate answers, and implementation in nonlinear optimization models is straightforward.

On the algorithmic side, cuts are added to enhance the solution quality. Application of cuts is a well-known technique in the area of nonlinear mixed-integer optimization. ([1], [7]). A dedicated application to the fuel management problem increases solution quality without much increase in the solution time. In our paper, we apply cuts to prevent the solution process from exploring reload patterns where the safety limitations are violated and to drive the solution into feasible, high quality solutions.

This paper is organized as follows. In the next section, the mathematical model is derived. This is the model as used in one of our previous papers [18] with extensions to include the new features. Also, the notation has been improved. Section 3 describes algorithmic details and results of the algorithm used to solve the model including Burnable Poisons. In Section 4, the cutting algorithm is discussed, and test results on the basic model are reported. The paper is concluded in Section 5.

2. Model description

In order to be able to apply mixed-integer nonlinear optimization methods, the model has to be specified as a set of algebraic equations, which are defined over a suitable set of variables. This set consists of variables specifying the loading pattern, and variables describing the corresponding physical behavior of the core.

The algebraic equations are divided into five different types.

- The *first* class of equations describes, for a given loading pattern, the evolution of the core during the cycle from BoC to EoC. This is our physical core model.

- The *second* class of equation specifies valid loading patterns.
- The *third* class describes the equilibrium cycle condition.
- The *fourth* class are the operational constraints, such as safety limitations.
- The *fifth* class are constraints that arise from the specific optimization method that is being chosen. These include cuts that cut off 'bad' solutions and help finding good solutions.

The model described here is based on a nodal core model. The evolution of the core during a cycle is computed in a fixed number of discrete time steps. Calculations are performed on equilibrium cycle reload patterns, which means that the same reloading pattern is applied in a number of successive cycles. At each EoC, the same fixed number of old fuel bundles is discharged, (typically one fourth), which are all of the same age. The advantage of this approach is that, ideally, the behavior of the reactor will converge and eventually be the same during all subsequent cycles. The initial power of fresh bundles, causing high power peaks just after BoC, can be reduced by addition of burnable poisons. Contrary to most current solution methods, the method shown in this paper makes it possible to optimize in one stage both the reload pattern and the fraction of burnable poison that has to be added for this reload pattern.

The following problem dimensions are used throughout this paper:

- I – the number of nodes in the core;
- M – the number of discharged bundles at EoC;
- L – the number of cycles that each bundle stays in the core before being discharged; note that $L = I/M$;
- T – the number of discretization points in time. A cycle is divided in $T-1$ time-intervals of equal length. The first time step describes the state of the core at BoC, the next time steps describe the core at the end of the subsequent time intervals.

A popular way to describe the neutron flux in the reactor core is to consider the production, absorption and transport of neutrons as a diffusion process. This approach forms the basis of the model used in this paper. Neutrons in a reactor core have different amounts of energy. Roughly, the neutrons can be divided into two groups, a *fast* group of neutrons with high energy levels, and a *thermal* group with low energy levels. For the neutron flux, we use the following notation.

– $\phi_{i,t}^{(g)}$: The (integrated) neutron flux in node i at time t for the fast group ($g = 1$) and the thermal group ($g = 2$).

Neutrons in the reactor core are absorbed by nuclei to cause different reactions. The rates in which the nuclei are absorbed are represented by the *macroscopic cross sections* Σ (dimension cm^{-1})[1]. For the purpose of this paper, we define the following cross sections:

– $\Sigma_{\mathrm{f},i,t}^{(1)}$, $\Sigma_{\mathrm{f},i,t}^{(2)}$: *Fission cross section* of the fast and thermal group, *i.e.,* the probability per unit path length that a neutron will be absorbed to cause a fission reaction.

– $\Sigma_{\mathrm{a}}^{(1)}$, $\Sigma_{\mathrm{a}}^{(2)}$: *Absorption cross section* of the fast and thermal group, *i.e.,* the probability per unit path length that a neutron will be absorbed by a nucleus, either to cause a fission reaction or due to another capture process other than scattering.

– $\Sigma_{\mathrm{s}}^{(1,2)}$: *Downscattering cross section*, *i.e.,* the probability per unit path length that a fast neutron interacts with a nucleus, resulting in the immediate re-emission of the neutron at thermal energy level.

Strictly speaking, the absorption and downscattering cross section also depend on the position in the core and on time. In the sequel, it is argued that this dependency can be ignored without too much loss of accuracy. Throughout this paper, the two-group nodal diffusion model is reduced to a one-group model using the $1^1/_2$-group approximation technique (see e.g. [5]). In this approximation, diffusion of thermal neutrons is neglected, which implies that all thermal neutrons are absorbed. The only thermal neutron source is the downscattering from the fast group, hence the fast and thermal neutron flux are related directly by the thermal neutron balance:

$$\Sigma_{\mathrm{s}}^{(1,2)}\phi_{i,t}^{(1)} \;=\; \Sigma_{\mathrm{a}}^{(2)}\phi_{i,t}^{(2)}. \tag{1}$$

Using (1), the thermal neutron flux can be eliminated, so that essentially a 1-group model remains. In this model, the *infinite multiplication factor* $k_{i,t}^{\infty}$ of a fuel bundle i at time t is defined as

$$k_{i,t}^{\infty} \;=\; \left(\frac{\text{local fast neutron production rate}}{\text{local fast neutron removal rate}}\right) \text{averaged over node } i \text{ at time } t.$$

[1] In order to avoid confusion between the cross section Σ and the summation sign $\sum$, we consequently place indices straight under and above each summation sign, and sub- and superscripts after macroscopic cross sections.

The neutron production and removal rates are determined by the material cross-sections, the neutron yield per fission ν, and the actual neutron flux. Neutrons are produced by fission of fast or thermal neutrons:

$$\text{fast neutron production rate} = \nu^{(1)}\Sigma^{(1)}_{\mathrm{f},i,t}\phi^{(1)}_{i,t} + \nu^{(2)}\Sigma^{(2)}_{\mathrm{f},i,t}\phi^{(2)}_{i,t}.$$

Here the thermal neutron flux can be eliminated by (1). Fast neutrons are removed by absorption, and by downscattering to the thermal group. The total infinite multiplication factor can therefore be written as

$$k^{\infty}_{i,t} = \frac{\left(\nu^{(1)}\Sigma^{(1)}_{\mathrm{f},i,t} + \nu^{(2)}\Sigma^{(2)}_{\mathrm{f},i,t}\dfrac{\Sigma^{(1,2)}_{\mathrm{s}}}{\Sigma^{(2)}_{\mathrm{a}}}\right)\phi^{(1)}_{i,t}}{(\Sigma^{(1)}_{\mathrm{a}} + \Sigma^{(1,2)}_{\mathrm{s}})\phi^{(1)}_{i,t}}, \tag{2}$$

where $\nu^{(1)}$ and $\nu^{(2)}$ are the yields per fission for the fast and thermal group. The flux $\phi^{(1)}_{i,t}$ cancels from (2), which shows that $k^{\infty}_{i,t}$ is independent of the actual flux, and only depends on the material composition.

Since the thermal neutron flux will not be used in the sequel, the superscript from the flux is removed, and any flux or other neutron related property will be assumed to be the fast neutron property, unless explicitly stated.

2.1. Kernel equation

In order to calculate the neutron flux in the whole core, the interaction between the different fuel bundles should be described. To describe the total core behavior at discretized time intervals during the operation cycle, the following quantities are introduced.

- G_{ij}: The probability for a neutron produced in node j to be removed (i.e. absorbed, or downscattered and then thermally absorbed) in node i.
- $R_{i,t}$: The fast neutron removal rate, averaged over node i at time t.
- $k^{\infty}_{i,t}$: $\dfrac{\text{neutron production rate}}{\text{neutron removal rate}}$, being the infinite multiplication factor averaged over node i at time t.
- k^{eff}_{t} : $\dfrac{\text{total neutron production rate}}{\text{total neutron loss rate}}$ over the whole core at time t.
- $\phi_{i,t}$: The average (fast) neutron flux in node i at time t.

The product $k_{i,t}^{\infty} R_{i,t}$ represents the fast neutron production rate in node i at time t, and the production and removal rates are related by the time-independent kernel equations

$$R_{i,t} = \frac{1}{k_t^{\text{eff}}} \sum_j G_{i,j} k_{j,t}^{\infty} R_{j,t}. \tag{3}$$

In this equation, k_t^{eff} acts as an eigenvalue. Clearly, $k_t^{\text{eff}} > 1$ means that the core is super-critical in its uncontrolled state; $k_t^{\text{eff}} < 1$ defines a subcritical core.

2.2. Fuel burnup

During reactor operation, the fission reactions cause burnup of the fuel. The amount of fissionable material, and therefore the fission cross-sections $\Sigma_{\text{f}}^{(1)}$ and $\Sigma_{\text{f}}^{(2)}$ decrease. In our current $1^1/_2$ group approximation it is assumed that the fast neutron removal cross section $\Sigma_{\text{a}}^{(1)} + \Sigma_{\text{s}}^{(1,2)}$ remains approximately constant. This is a reasonable assumption, since most of the neutron removal is caused by slowing down, and hence the downscattering to the thermal group $\Sigma_{\text{s}}^{(1,2)}$, which is determined by the moderator instead of the fuel. Moreover, it is assumed that the thermal absorption cross section $\Sigma_{\text{a}}^{(2)}$ remains approximately constant. This cross section decreases, because of the decreasing atom density of fissionable nuclides, but it also increases, since the fission products absorb neutrons, without causing a fission or scattering reaction.

The only two remaining time-dependent cross sections are the fission cross sections. The decrease of these cross sections is described by the time-dependent equations

$$\begin{aligned} \frac{d\Sigma_{\text{f}}^{(1)}}{dt} &= -\sigma_{\text{a}}^{\text{fuel}} \Sigma_{\text{f}}^{(1)} \phi, \\ \frac{d\Sigma_{\text{f}}^{(2)}}{dt} &= -\sigma_{\text{a}}^{\text{fuel}} \Sigma_{\text{f}}^{(2)} \phi. \end{aligned} \tag{4}$$

Here, $\sigma_{\text{a}}^{\text{fuel}}$ is a the *microscopic absorption cross section*, which is a constant that depends on the fuel composition. Note that the indices on the variables are temporarily left out since we deal with non-discretized equations here.

In the sequel it turns out to be handy to substitute ϕ out and to replace it by R using the relation

$$R = (\Sigma_{\text{a}}^{(1)} + \Sigma_{\text{s}}^{(1,2)})\phi. \tag{5}$$

Combining (2), (4) and (5) gives the time-dependence of k^∞ in terms of R:

$$\frac{dk^\infty}{dt} \;=\; -\,\sigma_{\rm a}^{\rm fuel} k^\infty \phi \;=\; \frac{-\sigma_{\rm a}^{\rm fuel}}{\Sigma_{\rm a}^{(1)} + \Sigma_{\rm s}^{(1,2)}} k^\infty\, R. \tag{6}$$

This equation can be discretized by either forward or central discretization, to obtain

$$k_{i,t+1}^\infty - k_{i,t}^\infty \;= \\ \frac{-\sigma_{\rm a}^{\rm fuel}\Delta_t}{\Sigma_{\rm a}^{(1)}+\Sigma_{\rm s}^{(1,2)}} \cdot \begin{cases} k_{i,t}^\infty R_{i,t} & \text{(forward discretization)} \\ \frac{1}{2}(k_{i,t}^\infty R_{i,t} + k_{i,t+1}^\infty R_{i,t+1}) & \text{(central discretization).} \end{cases} \tag{7}$$

The introduction of central discretization is new with respect to previous papers. In the nonlinear optimization approach, this can be implemented smoothly. In local search methods, special constructions have to be devised to use central discretization. Central discretization leads to a slightly larger nonlinear optimization problem (there are two nonlinear terms in the right hand side of (7)), but gives more precision. Figure1 shows the objective function for a fixed reload pattern, computed with both central and forward discretization for different numbers of time steps. The time steps are equally distributed over the fixed cycle length. It is seen that the forward difference solutions converge very slowly. Although using central differences results in a longer solution time for the same number of time steps, the overall number of time steps needed to obtain the same accuracy is much smaller.

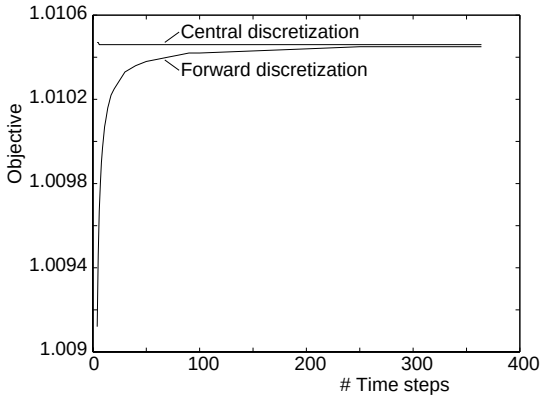

Figure 1. Objective values obtained by central and forward discretization, respectively, plotted as function of the number of time steps.

2.3. Power normalization

The kernel equations (3) describe, up to a constant, the removal rates. The remaining degree of freedom is the power level at which the core is operated. This power level is an external requirement, and should be fixed to the value P_{c}. The actual power level can be expressed in terms of the model variables $k^{\infty}_{i,t}$ and $\phi_{i,t}$ since it is equal to the fission rate, multiplied by the amount of energy μ_f released per fission. The fission rate in node i at time t is

$$\left(\Sigma^{(1)}_{\mathrm{f},i,t} + \Sigma^{(2)}_{\mathrm{f},i,t}\frac{\Sigma^{(1,2)}_{\mathrm{s}}}{\Sigma^{(2)}_{\mathrm{a}}}\right)\phi_{i,t}.$$

This is almost the same as the nominator of (2), except for the multiplication factors $\nu^{(1)}$ and $\nu^{(2)}$. Since these factors are almost equal, and moreover $\Sigma^{(1)}_{\mathrm{f}}$ is a few orders of magnitude smaller than $\Sigma^{(2)}_{\mathrm{f}}$, only a very small error is introduced if $\nu^{(1)}$ and $\nu^{(2)}$ are replaced by one properly averaged factor ν. Using this ν and (2), the fission rate can be described in terms of k^{∞} and R:

$$\left(\Sigma^{(1)}_{\mathrm{f},i,t} + \Sigma^{(2)}_{\mathrm{f},i,t}\frac{\Sigma^{(1,2)}_{\mathrm{s}}}{\Sigma^{(2)}_{\mathrm{a}}}\right)\phi_{i,t} = \frac{k^{\infty}_{i,t}\,R_{i,t}}{\nu}.$$

Using this expression, the power level restriction can be written as a relation between $k^{\infty}_{i,t}$ and $\phi_{i,t}$:

$$\sum_i k^{\infty}_{i,t}R_{i,t} \;=\; \nu\frac{P_{\mathrm{c}}}{\mu_f}, \quad t = 1, ..., T. \tag{8}$$

2.4. Safety limitation

For safety reasons, not all loading patterns are allowed. It is required that the power density in any bundle does not become too high, compared to the average bundle power. This power peaking constraint is applied for every node and at any time. Conform (8), it can be stated as follows:

$$k^{\infty}_{i,t}R_{i,t} \;\leq\; \nu\frac{P_{\mathrm{c}}}{\mu_f}\cdot\frac{f^{\mathrm{lim}}}{I}, \quad i = 1, ..., I,\;\; t = 1, ..., T.$$

These constraints ensure that the power level in node i is at most f^{lim} times the average power level.

2.5. Objective

The objective of the reload pattern optimization problem is to maximize the effective multiplication factor at EoC:

$$\max \quad k_T^{\text{eff}}.$$

2.6. Burnable Poisons

In this section, it is explained how the use of burnable absorbers can be included in a nonlinear optimization model. This is an extension to the model as developed in [18]. Optimization of the concentration of continuously variable burnable absorbers (*e.g.* WABA) is not very well possible in local search algorithms. The purpose of this section is to show that they can be implemented in a natural way in the nonlinear optimization model as we propose.

The use of burnable absorbers can be understood as follows. At BoC, the power difference between fresh and older bundles is quite large. Since fresh bundles generally have a higher burnup rate than older ones, this difference becomes less and less pronounced during the cycle. Hence, the highest power peaks are likely to occur in fresh bundles, during a short period after BoC. This may prohibit a loading pattern that is otherwise appreciated, just because of the safety limitations during a short period near BoC. It also may happen that fresh bundles burn up too rapidly.

In order to overcome these effects, it is possible to add additional absorbing material to fresh fuel bundles, that initially absorbs many neutrons, but burns up rather rapidly, so that it does not affect the next cycles. This implies that the local infinite multiplication factor is decreased, since the removal rate increases due to extra absorption of (mainly) thermal neutrons, while the production rate decreases because of a smaller neutron flux. Ideally, this will reduce the local power density in the bundle and its vicinity. An example of this effect is shown in Figure2, where the power as function of time is plotted for all bundles in a sample core. The loading pattern is initially not feasible, but the addition of absorbing material makes it feasible.

Burnable poisons are described in the model by an extra absorption cross section, that only plays a role for the thermal group.

- $\Sigma^{\text{p}}_{\text{a},i,t}$: The absorption cross section of burnable absorbers in node i at time t.

In the sequel, the infinite multiplication factor k^∞ depends not only on 'standard' fission and absorption cross sections, but also on burnable

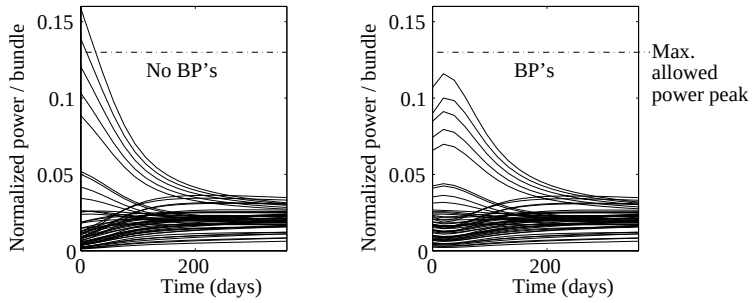

Figure 2. Power as function of time for each bundle, without and with burnable poisons.

absorbers. In order to separate the effect of the BP, an additional variable is introduced:

– $\overline{k}^{\infty}_{i,t}$: The (theoretical) infinite multiplication factor of the fuel if BP is not taken into account.

This variable will be related to the real multiplication factor. Due to burnable absorbers, an extra term in the definition of k^{∞} (2) is needed:

$$k^{\infty} \;=\; \frac{(\nu^{(1)}\Sigma_{\rm f}^{(1)} + \nu^{(2)}\Sigma_{\rm f}^{(2)}\dfrac{\Sigma_{\rm s}^{(1,2)}}{\Sigma_{\rm a}^{(2)}+\Sigma_{\rm a}^{\rm p}})\phi^{(1)}}{(\Sigma_{\rm a}^{(1)}+\Sigma_{\rm s}^{(1,2)})\phi^{(1)}} \tag{9}$$

Let $\overline{k}^{\infty}$ be the infinite multiplication factor as defined in (2), *i.e.,* without BP. Since the fast fission cross section is a few orders of magnitude smaller than the thermal fission cross section, only a negligible error is introduced when k^{∞} as defined in (9) is related to $\overline{k}^{\infty}$ in the following way:

$$k^{\infty} \;=\; \overline{k}^{\infty}\frac{1}{1+\Sigma_{\rm a}^{\rm p}/\Sigma_{\rm a}^{(2)}}. \tag{10}$$

Due to the absorption, the amount of burnable poison decreases at a rate which depends on the *microscopic absorption cross section* $\sigma_{\rm a}^{\rm p}$ of the poison (compare to (4)):

$$\frac{d\Sigma_{\rm a}^{\rm p}(x,t)}{dt} \;=\; -\,\sigma_{\rm a}^{\rm p}\Sigma_{\rm a}^{\rm p}\phi \;=\; -\,\frac{\sigma_{\rm a}^{\rm p}}{\Sigma_{\rm a}^{(1)}+\Sigma_{\rm s}^{(1,2)}}\Sigma_{\rm a}^{\rm p}\,R.$$

Combining all elements that are derived thus far, we arrive at a model where $\overline{k}^{\infty}_{i,t}$ satisfies the (forward or central) discretized burnup equations (7), where

$$\Sigma_{{\rm a},i,t+1}^{\rm p} - \Sigma_{{\rm a},i,t}^{\rm p} \;= \tag{11}$$

$$\frac{-\sigma_{\rm a}^{\rm p}\Delta_t}{\Sigma_{\rm a}^{(1)}+\Sigma_{\rm s}^{(1,2)}} \cdot \begin{cases} \Sigma_{{\rm a},i,t}^{\rm p} R_{i,t} & \text{(forward), or} \\ (\Sigma_{{\rm a},i,t}^{\rm p} R_{i,t} + \Sigma_{{\rm a},i,t+1}^{\rm p} R_{i,t+1})/2 & \text{(central),} \end{cases} \tag{12}$$

and where k^∞ is related to $\overline{k}^\infty$ and $\Sigma_{\rm a}^{\rm p}$ through (10).

2.7. Loading pattern specification

All equations derived in the previous sections describe the physical behavior for a given loading pattern. In our optimization model, the loading pattern itself is also described by a set of variables. A number of constraints describe what are the valid loading patterns. In this paper, it is assumed that equilibrium cycle reload patterns are used. At each EoC, the same number of old fuel bundles is discharged, and all those removed bundles are of the same age. Assuming that a quarter of the bundles is removed at each EoC, all those bundles have spent 4 cycles in the core. The initial configuration of the core will be the same at each BoC. The trajectory notation [6] describes how the bundles are moved during reloading. For example, suppose there are 12 nodes in the core, and at each EoC, three bundles are discharged. The core may for example be described by the following three trajectories:

$$\begin{array}{rcrcrcr} 4 & \rightarrow & 6 & \rightarrow & 8 & \rightarrow & 10 \\ 2 & \rightarrow & 3 & \rightarrow & 1 & \rightarrow & 5\,, \\ 7 & \rightarrow & 11 & \rightarrow & 9 & \rightarrow & 12 \end{array}$$

In this example, the bundle in node 4 is moved to node 6, and a fresh bundle is supplied in node 4, *etc.* In order to be able to formulate the problem in optimization context, the following binary variables are used that describe the trajectories:

$$x_{i,\ell,m}, \quad i = 1, ..., I, \ \ell = 1, ..., L, \ m = 1, ..., M :$$

$$x_{i,\ell,m} = \begin{cases} 1 & \text{if node } i \text{ contains the bundle of age } \ell \text{ from trajectory } m, \\ 0 & \text{otherwise.} \end{cases}$$

The index m will be denoted as the 'bundle-number' of the bundle. The variables $x_{i,\ell,m}$ are simply assignment variables, restricted by the following assignment constraints:

$$\sum_{\ell=1}^{L} \sum_{m=1}^{M} x_{i,\ell,m} \ = \ 1, \quad i = 1, ..., N;$$

$$\sum_{i=1}^{N} x_{i,\ell,m} \ = \ 1, \quad \ell = 1, ..., L, \ m = 1, ..., M;$$

$$x_{i,\ell,m} \in \{0,1\}.$$

2.8. Equilibrium cycle condition

The core evolution description and the loading pattern assignment are to be coupled by equations that specify the reloading operation. That is, it should be specified how the infinite multiplication factors at BoC depend on the EoC state of the previous cycle.

The two properties that are to be specified at BoC are $\overline{k}^{\infty}$, which is a measure for the current 'fuel state' of a bundle, and $\Sigma_{\mathrm{a}}^{\mathrm{p}}$, which is a measure for the amount of burnable poison in the fuel bundle. In the reloading equations, it is stated that the $\overline{k}^{\infty}$ of a bundle at BoC is $\overline{k}^{\infty}$ at EoC in its previous position, while the $\overline{k}^{\infty}$ at BoC of a fresh bundle is k^{fresh}. In the same way, fresh bundles may have a nonzero burnable poison concentration, represented by a $\Sigma_{\mathrm{a}}^{\mathrm{p}}$ that is equal to a BoC-value b^{fresh}. Since the burnable poison concentration is negligible at EoC, its reloading to the next position of the bundle at EoC is neglected in the model. It is simply assumed that non-fresh bundles have a BP concentration of 0. The value b_m^{fresh}, which may be different for the fresh bundles of the different trajectories m, is to be determined by the optimization.

The reloading operation for $\overline{k}^{\infty}$ is given by equation (13) below. The right hand side consists of two terms. In the first term, the variables $x_{i,1,m}$ determine whether node i contains a fresh bundle after reloading, and assign to $\overline{k}_{i,1}^{\infty}$ the value k^{fresh} if this is the case. Note that in our model, there is only one type of fresh bundle with one value k^{fresh} (apart from the BP that is added to it). If node i contains an older bundle, then the second term of (13) determines from $x_{i,\ell,m}$ which bundle (ℓ, m) is in node i. It then finds the node where this bundle was located at age $\ell - 1$ and assigns the $\overline{k}_{j,T}^{\infty}$ of that node to $\overline{k}_{i,1}^{\infty}$ of the current node. The total reloading equation is given by

$$\overline{k}_{i,1}^{\infty} = \sum_{m=1}^{M} x_{i,1,m} k^{\mathrm{fresh}} + \sum_{\ell=2}^{L} \sum_{m=1}^{M} x_{i,\ell,m} \sum_{j=1}^{I} x_{j,\ell-1,m} \overline{k}_{j,T}^{\infty}, \quad i = 1, ..., I \tag{13}$$

The loading equation for the burnable poison concentration is similar, except that the second term is not needed for reasons discussed above. Moreover, concentration in fresh bundles is not a pre-determined value, but b_m^{fresh} is a model variable, and may be different for the fresh bundle of each trajectory m:

$$b_{i,1} = \sum_{m=1}^{M} x_{i,1,m} b_m^{\mathrm{fresh}}, \quad i = 1, ..., I \tag{14}$$

2.9. Octant symmetry

To reduce the model size, octant symmetry is assumed in the core, as is shown in Figure3. From this figure, one can see that special care

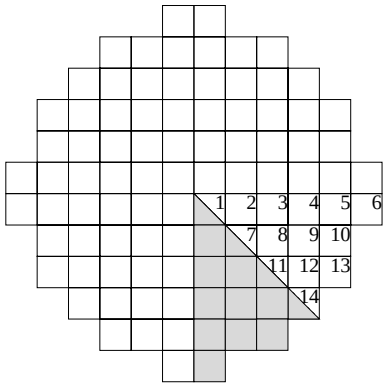

Figure 3. Octant symmetric core.

is needed for the bundles on the diagonal, since they fall only half into the octant. We added the slightly artificial constraint that two adjacent diagonal nodes contain the same bundle, which is split into two halves. This introduces volume factors V_i into the core, which are 1 for non-diagonal bundles, and $^1/_2$ for the diagonal bundles.

2.10. Model summary

The complete model for the octant core is listed below. In this model, we applied a scaling of R, dividing it by $\nu P_{\rm c}/\mu_f$. In this way, the power normalization constraint (8) scales to 1. To simplify notation further, the following two constants are introduced:

$$\alpha \;=\; \frac{-\sigma_{\rm a}^{\rm fuel}\,\nu}{\mu_f(\Sigma_{\rm a}^{(1)}+\Sigma_{\rm s}^{(1,2)})}\,, \quad \alpha^{\rm p} \;=\; \sigma_{\rm a}^{\rm p}\frac{\nu P_{\rm c}}{\mu_{\rm f}(\Sigma_{\rm a}^{(1)}+\Sigma_{\rm s}^{(1,2)})}\,.$$

In the complete model, as it is given here, forward time discretization is used.

$$\max_{x,\overline{k}^{\infty},\Sigma_{\rm a}^{\rm p},b^{\rm fresh},k^{\infty},R,k^{\rm eff}} \quad k_T^{\rm eff}$$

subject to:

the linear constraints:

$$\sum_{i=1}^{I} V_i x_{i,\ell,m} \;=\; 1, \qquad\qquad \ell=1,\cdots,L,\;\; m=1,\cdots,M \qquad (15)$$

$$\sum_{\ell=1}^{L}\sum_{m=1}^{M} x_{i,\ell,m} = 1, \qquad i = 1,\cdots,N \tag{16}$$

the nonlinear constraints:

$$\overline{k}^{\infty}_{i,1} = \sum_{\ell=2}^{L}\sum_{m=1}^{M} x_{i,\ell,m}\sum_{j=1}^{I} V_j\, x_{j,\ell-1,m}\overline{k}^{\infty}_{j,T} + \sum_{m=1}^{M} x_{i,1,m}k^{\mathrm{fresh}}, \qquad i = 1,\cdots,N \tag{17}$$

$$\Sigma^{\mathrm{p}}_{\mathrm{a},i,1} = \sum_{m=1}^{M} x_{i,1,m}b^{\mathrm{fresh}}_{m}, \qquad i = 1,\cdots,N \tag{18}$$

$$(1+\frac{\Sigma^{\mathrm{p}}_{\mathrm{a},i,t}}{\Sigma^{(2)}_{\mathrm{a}}})k^{\infty}_{i,t} = \overline{k}^{\infty}_{i,t}, \qquad i = 1,\cdots,I,\ t = 1,\cdots,T \tag{19}$$

$$k^{\mathrm{eff}}_{t}R_{i,t} = \sum_{j=1}^{I} G_{i,j}k^{\infty}_{j,t}R_{j,t}, \quad i = 1,\cdots,I,\ t = 1,\cdots,T \tag{20}$$

$$\sum_{i=1}^{I} V_i k^{\infty}_{i,t}R_{i,t} = 1, \qquad t = 1,\cdots,T \tag{21}$$

$$\overline{k}^{\infty}_{i,t+1} = \overline{k}^{\infty}_{i,t} - \alpha\Delta_t\overline{k}^{\infty}_{i,t}R_{i,t}, \qquad i = 1,\cdots,I,\ t = 1,\cdots,T-1 \tag{22}$$

$$\Sigma^{\mathrm{p}}_{\mathrm{a},i,t+1} = \Sigma^{\mathrm{p}}_{\mathrm{a},i,t} - \alpha^{\mathrm{p}}\Delta_t\Sigma^{\mathrm{p}}_{\mathrm{a},i,t}R_{i,t}, \qquad i = 1,\cdots,I,\ t = 1,\cdots,T-1 \tag{23}$$

$$k^{\infty}_{i,t}R_{i,t} \le \frac{f^{\mathrm{lim}}}{\sum_{j=1}^{I} V_j}, \qquad i = 1,\cdots,I,\ t = 1,\cdots,T \tag{24}$$

the variable bounds:

$$k^{\mathrm{eff}}_{t} \ge 0, \qquad t = 1,\cdots,T \tag{25}$$

$$\overline{k}^{\infty}_{i,t}, k^{\infty}_{i,t} \ge 0, \qquad i = 1,\cdots,I,\ t = 1,\cdots,T \tag{26}$$

$$\Sigma^{\mathrm{p}}_{\mathrm{a},i,t}, R_{i,t} \ge 0, \qquad i = 1,\cdots,I,\ t = 1,\cdots,T \tag{27}$$

$$b^{\mathrm{fresh}}_{m} \in [0, \Sigma^{\mathrm{p}}_{\mathrm{a,max}}], \qquad m = 1,\cdots,M \tag{28}$$

$$x_{i,\ell,m} \;=\; x_{j,\ell,m}, \;\; \text{all adjacent diagonal pairs } (i,j), \quad \ell = 1,\cdots,L,\;\; m = 1,\cdots,M \tag{29}$$

$$x_{i,\ell,m} \;\in\; \{0,1\}, \quad i = 1,\cdots,I,\;\; \ell = 1,\cdots,L,\;\; m = 1,\cdots,M. \tag{30}$$

The main difference with the model that was used in an earlier paper [19] is the inclusion of burnable poison. The model of [19] was an extension of an earlier model, presented in [11]. This model did not treat individual bundles of one age group separately. It also had a different objective function. Let us stress that the main purpose of the current paper is to discuss solution approaches, and to demonstrate that burnable poison optimization can be integrated into the MINLP model in a natural way.

3. Optimization approach

As is shown in [19], use of mixed-integer nonlinear optimization (MINLP) is competitive in comparison to a local search method like pairwise interchange (PI) for the model without Burnable Poisons. One advantage of MINLP optimization is that it is rather straightforward to implement continuous optimization extensions like the BP optimization. The current section shows the implementation and the results of a MINLP algorithm.

3.1. Time-discretization

One important aspect of the model is the choice of the time-discretization. Following (7), the calculation of the burnup can be done by either forward discretization or by central discretization. In iterative methods, central discretization requires the availability of an estimate of $R_{i,t+1}$ at time t. But when solving the system of equations as a whole, like in Newton-based nonlinear optimization algorithms, the use of central discretization is as straightforward as forward discretization.

The time-discretization has to be fine enough to ensure that the solution remains stable and accurate. An overestimate on the length of one time step Δ_t is computed by requiring that all $\Sigma^{\mathrm{p}}_{\mathrm{a},i,t}$ should be positive. Suppose that $\Sigma^{\mathrm{p}}_{\mathrm{a},i,t}$ and $R_{i,t}$ are given for a given time t. Then forward calculation of $\Sigma^{\mathrm{p}}_{\mathrm{a},i,t+1}$ according to (23) requires that

$$1 - \alpha^{\mathrm{p}} \Delta_t R_{i,t} \;\geq\; 0$$

for each node i, which means that

$$\Delta_t \;\leq\; \frac{1}{\alpha^{\mathrm{p}} \max_i(R_{i,t})}.$$

To get an estimate for Δ_t in advance, an estimate of the maximal $R_{i,t}$ is needed. Using (24) and assuming that the power peak will not occur at a node where $k_{i,t}^{\infty} < 1$, it follows that for feasible patterns,

$$R_{i,t} \;\leq\; \frac{f^{\lim}}{\sum\limits_j V_j}.$$

Experiments show that for bad patterns, the power peak can be four times the maximum allowed power peak, especially for smaller cores. Thus, to get stable solutions for all possible patterns, we have the estimation that

$$\Delta_t \;\leq\; \frac{\sum\limits_j V_j}{4\alpha^{\mathrm{p}} f^{\lim}}.$$

In practice, the power peak occurs at the beginning of a cycle, or, when BPs are added, slightly thereafter. Hence, the time steps can be longer towards the end of the cycle:

$$\Delta_t = \frac{\text{Total cycle length}}{T-1}\left(\frac{1}{2} + \frac{t-1}{T-2}\right), \quad t = 1, ..., T-1. \tag{31}$$

The length of the time steps can be further increased when central discretization is used. In that case, (23) is replaced by

$$\Sigma^{\mathrm{p}}_{\mathrm{a},i,t+1} \;=\; \Sigma^{\mathrm{p}}_{\mathrm{a},i,t} - (\alpha^{\mathrm{p}}\Delta_t)(\Sigma^{\mathrm{p}}_{\mathrm{a},i,t}R_{i,t} + \Sigma^{\mathrm{p}}_{\mathrm{a},i,t+1}R_{i,t+1})/2,$$

which is equivalent to

$$\Sigma^{\mathrm{p}}_{\mathrm{a},i,t+1} \;=\; \frac{1 - \alpha^{\mathrm{p}}R_{i,t}\frac{\Delta_t}{2}}{1 + \alpha^{\mathrm{p}}R_{i,t+1}\frac{\Delta_t}{2}}\Sigma^{\mathrm{p}}_{\mathrm{a},i,t}.$$

This implies that Δ_t may be twice as large as in the case of forward discretization. In the sequel we will show test results on three different core layouts (Small, medium and large). For a cycle length of 350 days, this leads to the requirements:

$$\begin{array}{ll} \text{Small problems:} & \Delta_t \leq 11.5 \text{ days} \\ \text{Medium-size problems:} & \Delta_t \leq 8.5 \text{ days} \\ \text{Large problems:} & \Delta_t \leq 14 \text{ days} \end{array} \tag{32}$$

Note that the maximum Δ_t is not monotone in core-size, which is due to the fact that the $f^{\lim}$-value and α^{p} are different in the test problems of various sizes. Using central discretization with time steps divided as in (31), the width of the first time-step as function of the number of time steps T is computed as follows:

T	10	11	12	13	14
Δ_1 (days)	9.7	8.7	7.9	7.2	6.7

Based on these numbers, the number of time steps used is 12.

3.2. MINLP WITH ROUNDING

The MINLP approach uses a nonlinear optimization routine to solve for all variables at once. The system of inequalities that has to be solved is very nonlinear, and hence it is difficult for the routine to find a good starting point from scratch. Starting point in this context does not only mean: a loading pattern, but also the physical variables belonging to this loading pattern. The latter are easily obtained by running an iterative method for finding an approximate solution for a fixed loading pattern, which is a very fast procedure. The starting loading pattern itself should be non-integer, since many integer solutions are local optima. We derived a reasonable starting pattern that is fractional, which is described in [19]. In short, this starting point is generated as follows. For each of the different age groups of fuel bundles in the core (fresh, once burned, twice burned and three times burned), a number of likely positions in the core is identified by engineering knowledge, as it is for example shown in Figure4.

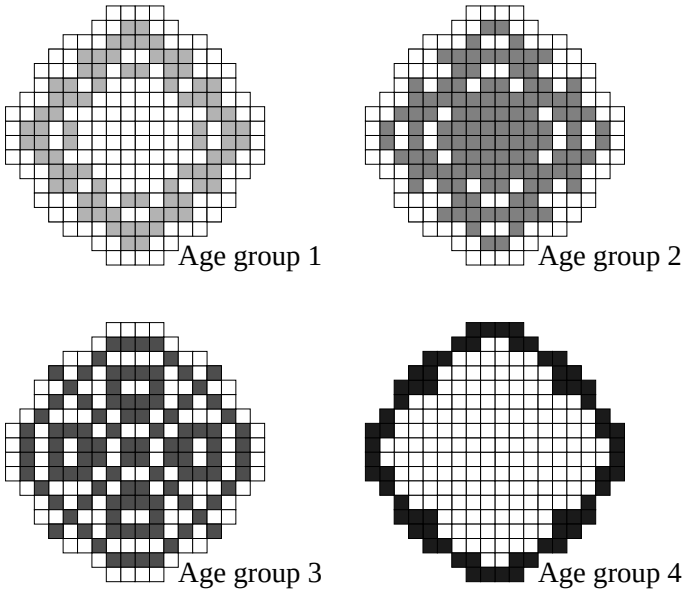

Figure 4. Likely locations for bundles in different age groups, obtained by engineering knowledge.

If the NLP solver returns a (local) optimal solution that is non-integer, a rounding procedure is applied. Starting from the relaxed

solution, this procedure enumerates all possible integer solutions such that

- ones in the relaxed solution remain ones, and
- zeros in the relaxed solution remain zeros.

It turns out in practice that the number of rounded solutions according to these properties is very small (usually 1 up to 10), so that we can evaluate them using a fast iterative algorithm. If there exists feasible rounded patterns, the best of them is returned. If not, then the infeasible pattern with best objective is returned. An outline of the algorithm is supplied in the algorithm of Figure 5. The optimization may be followed by a second step, in which a neighborhood search heuristic is used to check whether better solutions in the neighborhood of the local optimal solution can be found.

Procedure MINLP(x, f_x)

```
get_starting_pattern(x);
EvalFixed(x, f_x);
{NLP subproblems are solved using CONOPT}
NLPSolve(x, f_x);
if x is non-integer then
    f_B := 0; f_I := 0;
    for each rounding x̂ of x do
        EvalFixed(x̂, f_x̂);
        if x̂ is feasible and f_x̂ > f_B then        f_B := f_x̂; x_B := x̂;
        else if x̂ is not feasible and f_x̂ > f_I then f_I := f_x̂; x_I := x̂;
        endif;
    endfor;
    if f_B > 0 then  x := x_B; f := f_B;
    else             x := x_I; f := f_I;
    endif;
endif;
```

Figure 5. MINLP algorithm using a simple rounding procedure

3.3. Results

The algorithm was tested on 30 different test sets. One test set consists of a core layout, a value k^{fresh} of fresh bundles, a maximum amount of

BP to be added in fresh bundles, and additionally the safety limit $f^{\lim}$, the required power level P_c and the microscopic cross sections σ_a^{fuel} and σ_a^{p}. The test sets are divided into 10 problems with a small core layout (12 nodes in an eighth core), 10 problems in a medium sized core layout (28 nodes in one eighth of the core) and 10 large problems (48 nodes in one eight of the core).

Results are shown in Figure6 and Figure7. These graphs show the results for three different algorithms for all test sets. The first bar for each test problem shows the results for the model where addition of BP is not allowed. The second bar shows the result for the model with use of BP. The third bar shows the result of a two-phase algorithm. In the first phase, addition of BP is not allowed, but the safety limit is relaxed. In the second phase, the result of the first phase is used as starting point, and from there an optimization is started with use of BP. A star under the bar means that no BP is present in the final solution. One cross through the bar means that the NLP found a solution, but it was impossible to create a feasible integer solution out of this solution by use of our simple rounding procedure. Two crosses through the bar are used to indicate that even the NLP phase did not find a feasible fractional solution. As the attainable values of k_{Eoc}^{eff} for the various test sets are uncorrelated, the solution value k_{EoC}^{eff} is scaled, such that the best out of the three solutions is given the value 1, and the other two are scaled accordingly.

The graphs show some interesting results. In 24 out of the 30 cases, a better solution was found with one of the two algorithms where BP was included (11 for the second algorithm and 14 for the third algorithm). In one of the 6 cases where the first algorithm is the best, it only returns a fractional solution (M5). Another observation is that in relatively many cases, the third algorithm ends in a fractional solution with no feasible rounding. Apparently, the initial relaxation of the power peaking constraints leads to solutions where the power peak is so severely violated, that it cannot be repaired by the addition of BP only.

A striking result is that the BP algorithms give several times solutions where no BP is added, but that still have a better objective value than the solution of the first algorithm. These improved solutions are feasible solutions to the first model as well, but they were not found by the optimization algorithm. This may be explained by the larger feasible area in the latter two algorithms. On the other hand, both the size and the nonlinear complexity of the problem is increased when BP is taken into account, hence the computation time increases. This increase is not drastical; on average it is less than a factor 2. Compared to the computation time of a local search algorithm, as is described in [19], this is still reasonable.

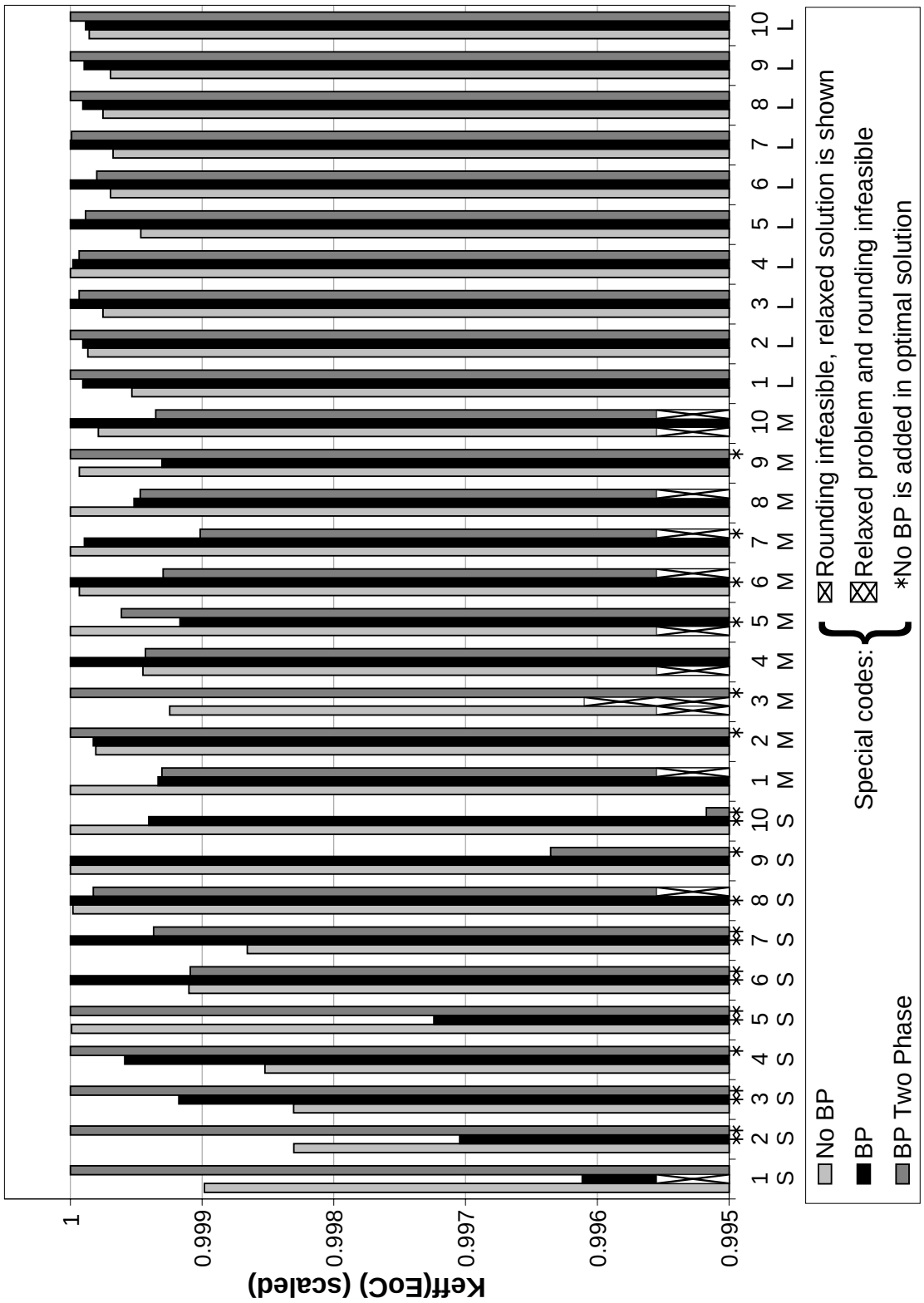

Figure 6. Results with and without use of Burnable Poisons

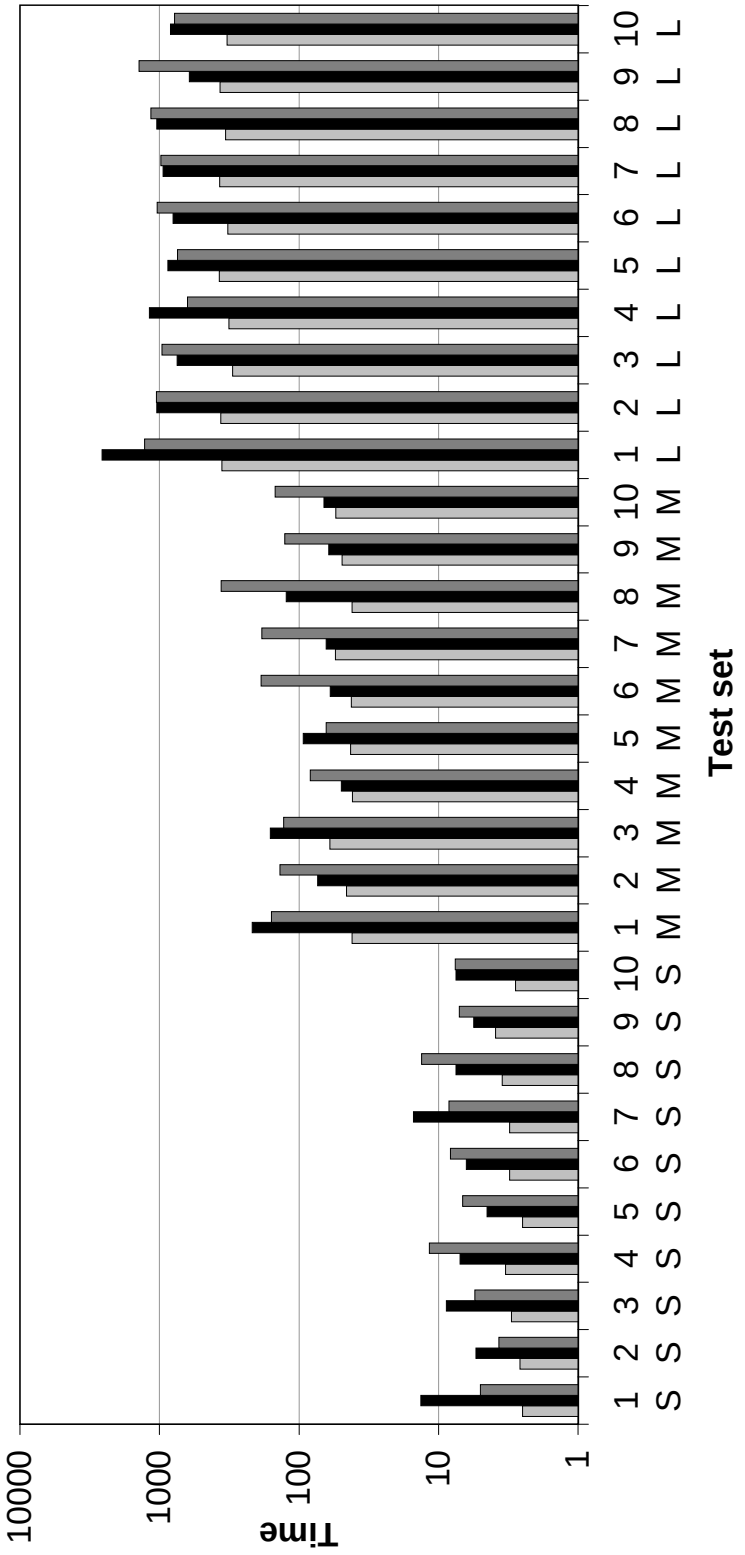

Figure 7. CPU time (on logarithmic scale) of the runs with and without use of Burnable Poisons

The most important observation is that the solutions are more stable for the larger problems, and that for the large cores the BP models consistently outperform the results of the basic model without BP. Further, BP is used in all of these solutions. In the small test cores, the placement of one or two bundles has much influence on the maximal power peak and the objective function of the core. The violation of the power peaking constraints caused by a single bundle at a 'wrong' position is so strong that it cannot be corrected by addition of BP. For the larger cores, the placement of one bundle has much less influence on the total power distribution, and hence addition of BP may better damp out this power peak. These results also underline our observation that the MINLP approach proofs its real power on large size problems. The larger the problems are, the more stable are the results. Other methods applied to this problem in the literature, like local search methods, have much difficulty to find good solutions for larger problems, and computation times increase very rapidly when problem sizes grow.

4. Cutting planes

The previous section discussed an extension of the basic model for nuclear reactor reload pattern optimization, while the underlying algorithm was more or less the same as the algorithm described in [19]. The current section deals with the model as of [19], but an algorithmic improvement is discussed. Although the basic algorithm turns out to be amazingly effective, we investigated whether cutting plane techniques can possibly improve the results. When NLP-solutions are fractional because of limiting power-peaking constraints, or rounding solutions violate some power peak, the nodes can be identified where the power peak constraint is active or violated. Cuts are then added that heuristically force the solution process into a direction where the power peak in these nodes will be smaller.

The largest power peak in a node i can be identified as

$$\max_t (k_{i,t}^{\infty} R_{i,t}).$$

In order to decrease this power peak, either $k_{i,t}^{\infty}$ or $R_{i,t}$ must be reduced.

These two quantities are related by the kernel equation (20):

$$k_t^{\text{eff}} R_{i,t} = \sum_{j=1}^{I} G_{i,j} k_{j,t}^{\infty} R_{j,t}.$$

Since the matrix G is diagonally dominant, reducing $k_{i,t}^{\infty}$ for some i will in general reduce the corresponding $R_{i,t}$, although it may occasionally

happen, due to normalization and influences in other parts of the core, that $R_{i,t}$ increases. The general idea of the cuts type 1, 2 and 3, that are described in the sequel, is based on reduction of $k^{\infty}_{i,t}$ in nodes i where the power peak is too high. So, if node i has the maximum power peak in the current iteration p, the basic form of such a cut is:

$$k^{\infty}_{i,t} \leq \delta k^{\infty,p}_{i,t}, \tag{33}$$

for some suitable $\delta < 1$. Since in our model k^{∞} is a variable that depends indirectly on the control variables $x_{i,\ell,m}$, the cuts are applied to these control variables instead of $k^{\infty}_{i,t}$ directly. Here the property is used that k^{∞} decreases during the lifetime of the bundle, and that bundles of the same age have k^{∞}-values that are rather close. The restriction (33) is replaced by the restriction that the age of the bundle in node i and/or its neighbors should be above a given value.

General cut-algorithm

---

repeat
  `Find_initial_pattern`(x);
  `NLP_solve`(x, f_x);
  `Process_solution`(x, f_x);
  Derive a set X of integer solutions from x;
  for each $\hat{x}$ in X **do**
    `EvalFixed`$(\hat{x}, f_{\hat{x}})$;
    `Process_solution`$(\hat{x}, f_{\hat{x}})$;
  endfor
  Remove cuts that are redundant or timed-out.
until stopping criterion met;

Figure 8. General outline of the cut-algorithm. The sub-algorithm Process_solution evaluates the quality of the solution and derives cuts when needed and possible

The general algorithm with cuts is given in the algorithm of Figure 8. In each iteration we start by computing an initial pattern. Using this starting pattern, the continuous NLP is solved. If the solution is feasible, integer and better than the best found solution thus far, it is stored. If it is not feasible then cuts are derived from this solution. If the solution is not integer, a number of integer solutions are derived from this fractional solution. In our implementation this is done by simple rounding. These solutions are 'processed' in the same way as the NLP solution. Redundant cuts are then removed. This algorithm

is repeated until some stopping criterion is met. In the next sections, three cut variants are shortly described.

Process_solution(x, f_x)

---

if x feasible **and** x integer **and** $f_x > f^{\text{best}}$ **then**
  Store x as x^{best} and f_x as f^{best}
if x not feasible **then**
  Derive cuts from x.

Figure 9. Sub-algorithm that evaluates the solution

4.1. Type 1 cuts

The power peak in some node i is mainly caused by the reactivity of itself and the adjacent nodes. High reactivity and a large flux in the surrounding nodes also causes a large flux in node i (see Figure10). In type 1 cuts, it is assumed that the power peak in node i can be

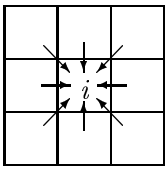

Figure 10. High fluxes in surrounding nodes influence the power in node i.

reduced by putting less active bundles in one or more of its neighbors, and it does not matter which one of the nodes this is. This constraint is implemented by computing the sum of the ages of the bundles in i and its surrounding nodes in the current solution. In the next iteration it is required that this age-sum should be larger than in the current solution, so that the average age in the relevant nodes increases.

Let the assignment variables in the current iteration p be denoted as $x^p_{i,\ell,m}$, denoting the fraction of the bundle with age ℓ from trajectory m that is placed in node i at the pth iteration. Let i be the node on which the cut is posed, and let $\mathcal{N}(i)$ be the set of neighbor nodes including i itself, consisting of all nodes as given in Figure10. We compute

$$A^1_i = \sum_{j \in \mathcal{N}(i)} \sum_{\ell=1}^{L} \sum_{m=1}^{M} \ell x^p_{j,\ell,m} \tag{34}$$

and add the constraint

$$\sum_{j\in\mathcal{N}(i)}\sum_{\ell=1}^{L}\sum_{m=1}^{M}\ell x_{j,\ell,m}\geq\lfloor A_i^1+\delta^1\rfloor \tag{35}$$

for some suitable δ^1 that is initially chosen greater than 1. The rounding is applied since in a fractional solution, A_i may be non-integer, but the final solution should always be integer. The choice of δ^1 is in principle free. In our implementation it consists of two elements: an element that depends on the magnitude of the power peak violation at node i in iteration p, and a fixed factor:

$$\delta^1=\alpha^1\delta^{\mathrm{PP}}+\varepsilon^1 \tag{36}$$

where α^1 and ε^1 are constants, and

$$\delta^{\mathrm{PP}}=\max_t(k_{i,t}^{\infty}R_{i,t})\,\frac{\sum_{j=1}^{I}V_j}{f^{\mathrm{lim}}}. \tag{37}$$

A value $\delta^{\mathrm{PP}}>1$ corresponds to a violation of the power peaking constraint. The values of α^1 and ε^1 should be tuned by experiment. As an alternative to the definition (36) of δ^1 we also tested

$$\delta^1=\alpha^1\log(\delta^{\mathrm{PP}})+\varepsilon^1. \tag{38}$$

This is motivated by the fact that the constraint (35) can be too strong for large violations of the power peak. In such a case it may occur that no assignment can be made that satisfies all added cuts. This is especially true for smaller cores. In those cores, the power peak can be very high since placing one or two fresh bundles at center positions has major influence on the whole core (we actually encountered values $\delta^{\mathrm{PP}}>5$). Also, these small cores have less flexibility to satisfy the cuts since there are less bundles of each age.

4.2. Type 2 cuts

In the type 1 cuts, it is implicitly assumed that the node i itself and its direct neighbors have the same influence on the power peak in i, whereas the influence of bundles that are further away is completely neglected. This assumption can be refined by taking into account the matrix $G_{i,j}$. As follows from the kernel equation, the contribution of the different nodes j to the flux value ϕ, hence to the removal rate R, and finally to the power peak in node i, is proportional to $G_{i,j}$:

$$R_{i,t}=\frac{1}{k_t^{\mathrm{eff}}}\sum_{j=1}^{I}G_{i,j}k_{j,t}^{\infty}R_{j,t}.$$

As in the type 1 cut, we assume that the power in node j is related to the age of the bundle, but instead of counting the ages of all neighbors as in (34), we count the ages with relative weights $G_{i,j}$:

$$A_i^2 = \sum_{j=1}^{I} G_{i,j} \sum_{\ell=1}^{L} \sum_{m=1}^{M} \ell x_{j,\ell,m}^p \tag{39}$$

and the cut is formulated as

$$\sum_{j=1}^{I} G_{i,j} \sum_{\ell=1}^{L} \sum_{m=1}^{M} \ell x_{j,\ell,m} \geq A_i^2 + \delta^2, \tag{40}$$

where δ^2 is defined in a similar way as δ^1. Note that the right hand side of (40) is not rounded to integer. Contrary to (35), the left hand side can be non-integer even when all x-variables are integer, thus non-integer right hand sides make sense. The relative importance of the neighbor nodes decreases about one order of magnitude for each step away from the center node i. Therefore, this cut leaves generally more flexibility for the assignment variables than the type 1 cut. For some relatively small power peak violations it may be possible to increase the age of the center node, while the age of a neighbor node is decreased, which still has the net effect of a decreasing left hand side of (40).

4.3. Type 3 cuts

The cut of type 2 may be refined further by taking into account the height of the power peak in node i and its neighbors. Suppose the power peak is exceeded in node i. If neighbor j_1 has a relatively large power, while neighbor j_2 with the same $G_{i,j}$-value has a much lower power, the bundle j_1 contributes more to the power peak in i than the bundle j_2. The power peak in i might be suppressed by putting an older bundle in j_1, even while a less old bundle is placed in j_2. This is accomplished by making the cut also dependent on the power peak in each node j. Let

$$P_j^c = \max_t (k_{j,t}^{\infty,p} R_{j,t}^p)$$

be the power peak in node j in the current solution p, then define

$$A_{i,p}^3 = \sum_{j=1}^{I} G_{i,j} P_j^p \sum_{\ell=1}^{L} \sum_{m=1}^{M} \ell x_{j,\ell,m}^p \tag{41}$$

leading to the cut definition

$$\sum_{j=1}^{I} G_{i,j} P_j^p \sum_{\ell=1}^{L} \sum_{m=1}^{M} \ell x_{j,\ell,m} \geq A_{i,p}^3 + \delta^3, \tag{42}$$

where δ^3 may be defined as in the previous cut types. Contrary to the previous cuts, the left hand sides in the different iterations are not necessarily linearly dependent, hence cuts from different subsequent iterations may be kept active for the same node i. On the other hand, this cut is very solution-specific, and therefore it can only be effective for solutions that are not too far away from the current solution. In practice, it only pays off to keep them during one or two iterations.

4.4. Starting point after addition of cuts

After the addition of cuts, the current solution may become infeasible. Starting from this point may lead to a new solution that is very close to the old one, but just far enough to satisfy the cut. This may still be in the attraction area of the local optimum that is cut off. We have devised a small convex minimization problem, that generates a starting point that satisfies the cuts, and that is a sort of weighted average of the original starting point (obtained by engineering knowledge) and the best solutions found until now. To be more precisely, define the following patterns:

1. $\overline{x}^{(1)}$: the current solution;
2. $\overline{x}^{(2)}$: the original starting point;
3. $\overline{x}^{(3)}, ..., \overline{x}^{(2+P)}$: the P best feasible patterns found until now, where P is a model parameter to be found by experimentation.

Let furthermore C be the number of cuts currently active, ε a positive 'proximity' parameter. Then the small minimization problem is given by

$$\begin{aligned}
\underset{x,\zeta\geq 0}{\text{minimize}} \quad & \sum_{p=1}^{2+P} \sum_{i=1}^{I} \sum_{\ell=1}^{L} \sum_{m=1}^{M} (x_{i,\ell,m} - \overline{x}^{(p)}_{i,\ell,m})^2 \;-\; \sum_{c=1}^{C} \log(\zeta_c + \varepsilon^{\log}) \\
\text{subject to} \quad & \sum_{i=1}^{I} V_i x_{i,\ell,m} = 1, \quad \ell = 1, \cdots, L, \;\; m = 1, \cdots, M \\
& \sum_{\ell=1}^{L} \sum_{m=1}^{M} x_{i,\ell,m} = 1, \quad i = 1, \cdots, I \\
& (\text{Value of cut } c) \geq (\text{RHS of cut } c) + \zeta_c, \quad c = 1, \cdots, C.
\end{aligned}$$

Depending on the values ζ_c and ε, the logarithmic penalty term in the objective drives the solution away from the boundary of the feasible area. This is especially advantageous when the lastly added cuts do not cut off the attraction area of the last obtained local optimum completely. In that case a model without this penalty term would very likely result in a solution at which the last added cut is active, and

that is still in the attraction area of the last obtained local optimum. This was observed quite frequently in experiments without this penalty term.

4.5. Results

In order to test the different cuts and combinations of cuts, we created a script around our program that generated random settings (within pre-specified ranges) for the different cut parameters. Based on experience with smaller test runs for each cut separately, upper and lower bounds on the α^i and ε^i were set for the different types of cuts.

The tests were executed for all 30 Small, Medium and Large test problems. For each test set, 200 test runs were performed. The results were compared to the results of the rounding algorithm without cuts, applied to the same basic model. Unfortunately, in most of the cases, the cuts did not help to find better optimal solutions. In many cases, either the cut algorithm returned to the attraction area of the previous local optimum, or it found a different local optimum with a worse objective value. Only in 10 from the 30 test sets, significantly better solutions were found, and these were obtained with different settings of the cut parameters. The best of these solutions are given in Figure11. The left bar is the solution without cuts. The middle column is the best solution out of the 200 test runs of the cut algorithm, and the right bar is the solution obtained by a Pairwise Interchange algorithm. The solution values of the best cut solutions are scaled to one, the other two values are scaled relative to this solution. As can be seen, the extra time required for the cut algorithm compared to the rounding algorithm is limited. This is because the first NLP optimization without cuts takes much time, while the subsequent re-optimizations are very fast.

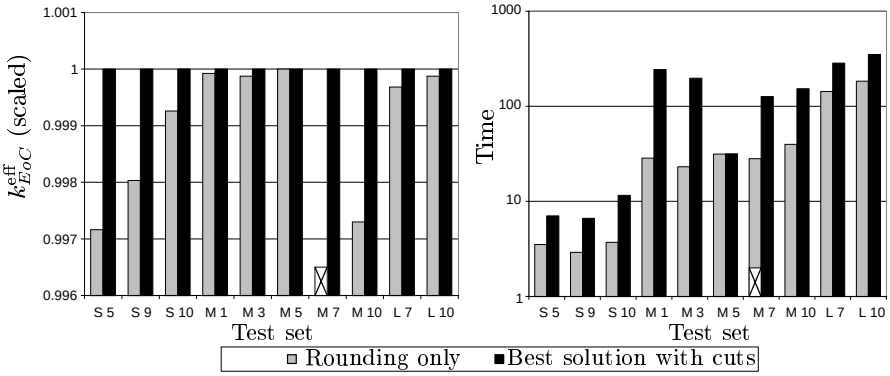

Figure 11. Results for Cuts.

There was no very clear correlation between the cut parameter settings and the results. From a statistical view, a few relations could be found, like the following:

- Settings where the sum of α^1 and ε^1 is around 0.2 - 0.3 gives in all but a few cases better results than settings where this sum is lower than 0.2 or above 0.3;
- The logarithmic variant of cut 2 gives on average better results than the linear variant;
- Most of the good results are obtained when $\alpha^2 \in [0.3, 0.4]$ and ε^2 is about $0.4 - \alpha^2$.

5. Conclusion

This paper describes how a basic nuclear reactor reload pattern optimization model can be extended as to include the use of Burnable Poisons. It is shown that this extended model can well be solved using a mixed integer nonlinear optimization algorithm. Unlike local search algorithms, this algorithm can handle both the shuffling of fuel bundles, and the addition of burnable poisons in one pass, even when the amount of burnable poison can be chosen arbitrarily.

An extension of the algorithm to further improve the results using linear cuts is less successful until now. The cuts described here are only defined on the assignment variables. One may use more of the problem knowledge and include cuts that also act on the infinite multiplication factors or other variables. This may improve the quality of the cuts, but makes them more dependent on the specific model being used. Another promising variant is the embedding of the cuts in a Branch-and-Cut algorithm. One may design deeper cuts, with the consequence that there is more risk to cut off the best solution. However, when embedding them in a Branch-and-Cut algorithm, one may evaluate both sides. Considering the limited increase of computation time with the current implementation, this should be feasible with a reasonable amount of computation time. An efficient restarting procedure on both sides may lead to further improved results.

References

1. I.P. Androulakis, C.D. Maranas, and C.A. Floudas. αBB: A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, 7:337–363, 1995.

2. J.K. Axmann. Parallel adaptive evolutionary algorithms for pressurized water reactor reload pattern optimizations. *Nuclear Technology*, 119:276–292, 1997.
3. J.N. Carter. Genetic algorithms for incore fuel management and other recent developments in optimisation. *Advances in Nuclear Science and Technology*, 25:113–154, 1997.
4. M.D. DeChaine and M.A. Feltus. Fuel management optimization using genetic algorithms and expert knowledge. *Nuclear Science and Engineering*, 124:188–196, 1996.
5. J.J. Duderstadt and L.J. Hamilton. *Nuclear Reactor Analysis*. Wiley & Sons, 1976.
6. R. van Geemert, A.J. Quist, and J.E. Hoogenboom. Reload pattern optimization by application of multiple cyclic interchange algorithms. In *Proceedings of PHYSOR96*, pages I38–I47. Mito, Japan, 1996.
7. R. Horst and H. Tuy. *Global Optimization; Deterministic Approaches*. Springer-Verlag, Berlin, 1990.
8. A.J. de Jong. Reloading pattern design for batch refuelled nuclear reactors. Technical Report IRI 131-95-010, Delft University of Technology, 1995.
9. T.K. Kim and C.H. Kim. Determination of optimized PWR fuel loading pattern by mixed integer programming. In *Proceedings of PHYSOR96*, pages I76–I85. Mito, Japan, 1996.
10. T.K. Kim and C.H. Kim. Mixed integer programming for pressurized water reactor fuel-loading-pattern optimization. *Nuclear Science and Engineering*, 127:346–357, 1997.
11. E. de Klerk, T. Illés, A.J. de Jong, C. Roos, T. Terlaky, J. Valkó, and J.E. Hoogenboom. Optimization of nuclear reactor reloading patterns. *Annals of Operations Research*, 69:65–84, 1997.
12. D.J. Kropaczek and P.J. Turinsky. In-core nuclear fuel management optimization for pressurized water reactors utilizing simulated annealing. *Nuclear Technology*, 95:9–32, 1991.
13. C. Lin, J-I. Yang, K-J. Lin, and Z-D. Wang. Pressurized water reactor loading pattern design using the simple tabu search. *Nuclear Science and Engineering*, 129:61–71, 1998.
14. G.I. Maldonado, P.J. Turinsky, and D.J. Kropaczek. Employing nodal generalized perturbation theory for the minimization of feed enrichment during pressurized water reactor in-core fuel management optimization. *Nuclear Science and Engineering*, 121:312–325, 1995.
15. G.T. Parks. An intelligent stochastic optimization routine for in-core fuel cycle design. *Transactions of the American Nuclear Society*, 57:259–260, 1988.
16. G.T. Parks. Multi-objective pressurized water reactor reload core design by nondominated genetic algorithm search. *Nuclear Science and Engineering*, 124:178–187, 1996.
17. P.W. Poon and G.T. Parks. Application of genetic algorithms to in-core fuel management optimization. In *Proceedings Joint International Conference on Mathematics and Supercomputing in Nuclear Applications*, pages Vol. 1, 777. Karlsruhe, 1993.
18. A.J. Quist, R. van Geemert, J.E. Hoogenboom, T. Illés, E. de Klerk, C. Roos, and T. Terlaky. Application of nonlinear optimization to reactor core fuel reloading. *Annals of Nuclear Energy*, 26(5):423–448, 1998.
19. A.J. Quist, R. van Geemert, J.E. Hoogenboom, T. Illés, E. de Klerk, C. Roos, and T. Terlaky. Finding optimal nuclear reactor core reload patterns us-

ing nonlinear optimization and search heuristics. *Engineering Optimization*, 32:143–176, 1999.
20. T. Šmuc, D. Pevec, and B. Petrović. Annealing strategies for loading pattern optimization. *Annals of Nuclear Energy*, 21:325–336, 1994.
21. J.G. Stevens, K.S. Smith, K.R. Rempe, and T.J. Downar. Optimization of pressurized water reactor shuffling by simulated annealing with heuristics. *Nuclear Science and Engineering*, 121:67–88, 1995.
22. J.S. Suh and S.H. Levine. Optimized automatic reload program for pressurized water reactors using simple direct optimization techniques. *Nuclear Science and Engineering*, 105:371–382, 1990.
23. F.C.M. Verhagen and M. van der Schaar. Simulated annealing in LWR fuel management. In *TOPNUX'93 Proceedings*, pages Vol. 2, 37–39, 1993.
24. S. Wei and C. Pingdong. A new approach for low-leakage reload core multicycle optimization design. In *Proceedings of PHYSOR96*, pages I86–I95. Mito, Japan, 1996.
25. A. Yamamoto. Comparison between equilibrium cycle and successive multicycle optimization methods for in-core fuel management of pressurized water reactors. In *Proceedings of the Joint International Conference on Mathematical Methods and Supercomputing for Nuclear Applications*, pages 769–781, Saratoga Springs, New York, 1997. American Nuclear Society, Inc.
26. N. Zavaljevski. A model for fuel shuffling and burnable absorbers optimization in low leakage PWR's. *Annals of Nuclear Energy*, 17(4):217–220, 1990.