Proceedings of the 2013 21st International Conference on Nuclear Engineering
ICONE21
July 29 - August 2, 2013, Chengdu, China

ICONE21-15730

STUDY ON IMPROVEMENT OF ANALYTIC DEPLETION CALCULATION METHOD

Kai Huang
School of Nuclear Science and Technology
Xi'an Jiaotong University
Xi,an, Shaanxi, 710049 China
Email: hk_1988@live.cn

Hongchun Wu
School of Nuclear Science and Technology
Xi'an Jiaotong University
Xi,an, Shaanxi, 710049 China

Liangzhi Cao*
School of Nuclear Science and Technology
Xi'an Jiaotong University
Xi,an, Shaanxi, 710049 China

Youqi Zheng
School of Nuclear Science and Technology
Xi'an Jiaotong University
Xi,an, Shaanxi, 710049 China

ABSTRACT

The transmutation trajectory analysis(TTA) method is a traditional analytic depletion calculation algorithm. It's capable of providing very accurate solutions with cutoff value being sufficiently small, but the main drawback is expensive time consumption. In this paper, the depletion problem was rephrased as a directed-graph module with additional properties attributed to vertices and edges. Based on this model a new approach for TTA implementation was conducted, to which dynamic programming technique could be introduced easily. Besides it, a new analytic algorithm is proposed based on the above model, which gives each nuclide a time dependent concentration function instead of a scalar value that other algorithms could only offer. One of the advantages brought by this algorithm is making time integration value calculations straightforward. Both of these two approaches are programmed. The resulting TTA code was proved to be very efficient, the running time for a typical ORIGEN2 problem with cutoff equals to 10^{-15} *could be of the same magnitude with time spent on library reading process. The codes are analyzed in terms of time and space complexities, which could offer a theoretical point of view on their behaves.*

*Corresponding author: +86 2982668692; Email:caolz@mail.xjtu.edu.cn

1 INTRODUCTION

Depletion calculation aims at solving time dependent nuclide concentrations based on given transmutation relations among all nuclides in concern. Under general circumstances, the transmutation paths and associated coefficients are approximated to be unchanged during a calculational step, thus reduce the original problem to a first order differential system. Two different approaches have been proposed in order to solve this ODE system, and they are very different from theoretical point of view. While one of them deals with approximation of matrix exponential, the other converts the transmutation network into forms that could be solved analytically. There is a abundant collection of matrix exponential approximation methods [1–4], but only one analytic method, namely TTA, follows the other approach. Many proposed methods have also been programmed and utilized. For example, Taylor series truncation with scaling and squaring technique was adopted by ORIGEN2 and ORIGEN-S code [4,5], CRAM which is abbreviation for Chebyshev Rational Approximation was introduced to depletion calculation in recent years [1,6], and CINDER code employed TTA method in solving depletion problem. Compared with most of matrix exponential approximation methods, TTA could ensure much higher accuracy of the solutions, and it is also capable of providing integral values at the same level of accuracy with negligible additional

computational spending. However the time consumption of TTA method is very expensive if very small cutoff value being set as a consequence of high accuracy requirement and particle induced transmutations are introduced. This is because complex transmutation network will produce massive amount of linear chains, and direct implementation of TTA must search and solve all these linear chains independently, and combine these partial solutions to get final solution. The main topic of this paper is improving the efficiency of TTA without loss of accuracy. Based on observation, most of the linear chains have some repeated parts with others, and the analytic expressions to Bateman equations have recurrence relations. Exploiting these natural properties possessed by depletion problem gives much more efficient implementation of TTA method. And by the inspiration of the invention of recurrence relations, a new analytic algorithm is proposed, implemented and tested.

2 METHOD OF TTA

The basic equation of depletion system could be written as:

$$\frac{dN}{dt} = AN \tag{1}$$

Where N is a vector containing the concentration data of all nuclides; A is a square matrix, transmutation relations are indicated by nonzero values of this matrix, and transmutation coefficients are recorded in corresponding entries.

Let $N(0)$ denote the initial composition vector, then the solution expression to homogenous system of Eqn. (1) could be written as:

$$N(t) = e^{tA}N(0) \tag{2}$$

The matrix exponential e^{tA} is defined as follow:

$$e^{tA} = I + \sum_{k=1}^{\infty} \frac{t^k A^k}{k!} \tag{3}$$

As mentioned above, TTA does not concern with the matrix exponential directly, alternatively, linear chains are searched and solved on the basis of data stored in A, the solution on the left hand side of Eqn.(2) is actually obtained as the sum of individual contributions from all processed linear chains.

The governing equations of linear chain problem are also called Bateman equations. They could be presented as:

$$\frac{dN_1}{dt} = -\lambda_1 N_1 \tag{4}$$

$$\frac{dN_i}{dt} = \lambda_{i-1}N_{i-1} - \lambda_i N_i \ (i = 2, \ldots, n) \tag{5}$$

or in matrix form:

$$\begin{pmatrix} \frac{dN_1}{dt} \\ \frac{dN_2}{dt} \\ \cdot \\ \cdot \\ \cdot \\ \frac{dN_n}{dt} \end{pmatrix} = \begin{bmatrix} -\lambda_1 & & & & \\ \lambda_1 & -\lambda_2 & & & \\ & \cdot & \cdot & & \\ & & \cdot & \cdot & \\ & & & \lambda_{n-1} & -\lambda_n \end{bmatrix} \begin{pmatrix} N_1 \\ N_2 \\ \cdot \\ \cdot \\ \cdot \\ N_n \end{pmatrix} \tag{6}$$

Assuming the following initial conditions:

$$N_1(0) \neq 0 \quad \text{and} \quad N_i(0) = 0 \quad \text{if } i > 1 \tag{7}$$

Bateman found out the analytic expressions of concentrations for all nuclides by Laplace transformation and inverse laplace transformation [7]:

$$N_m(t) = \frac{N_1(0)}{\lambda_m} \sum_{i=1}^{m} \lambda_i \alpha_i e^{-\lambda_i t} \quad 1 \leq m \leq n \tag{8}$$

$$\alpha_i = \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{\lambda_j}{(\lambda_j - \lambda_i)} \tag{9}$$

However, if any coefficients in the chain coincide, which could be a very common phenomena when particle flux induced transmutations take place, then infinity will be introduced to the expressions. Fortunately, Cetnar had removed these possible infinities and obtained the general solution based on limit calculation [8], the general solution expressions are presented below:

$$N_n(t) = \frac{A(t)N_1(0)}{\lambda_n} \tag{10}$$

$$A(t) = \sum_{i=1}^{n} \lambda_i \alpha_i e^{-\lambda_i t} \cdot \sum_{m=0}^{\mu_i} \frac{(\lambda_i t)^m}{m!} \cdot \Omega_{i,\mu_i - m} \tag{11}$$

$$\alpha_i = \prod_{\substack{j=1,n \\ j \neq i}} \left(\frac{\lambda_j}{\lambda_j - \lambda_i} \right)^{m_j} \tag{12}$$

$$\Omega_{i,j} = \sum_{h_1=0}^{j} \sum_{h_2=0}^{j} \cdots \sum_{h_n=0}^{j} \times \prod_{\substack{k=1 \\ k \neq i}}^{n} \binom{h_k + \mu_k}{\mu_k} \left(\frac{\lambda_i}{\lambda_i - \lambda_k} \right)^{h_k} \delta \left(j, \sum_{\substack{l=0 \\ l \neq i}}^{n} h_l \right) \tag{13}$$

2.1 BASIC CONCEPTS OF TTA

In section 2, the analytic expressions of linear chain problem was discussed. Remaining problem of TTA implementation is how to take apart the transmutation network into linear chains. A straightforward method of carrying out this task is presented here.

Let N represent the set of all nuclides in concern, $N = \{N_i, 1 \leq i \leq n\}$. For one particular nuclide, denoted by N_i, suppose it has n_i daughters—$N_{i,j}$ $(1 \leq j \leq n_i)$, and $N_{i,j} \in N$, associated branch ratios are $b_{i,j}$. Then it is self-evident that the proportion of nuclide N_i that goes though the following path is $B_{chain} = \prod_{x=j}^{end} b_{parent(x),x}$:

$$N_i \rightarrow N_{i,j} \rightarrow N_{i,j,k} \rightarrow \cdots \rightarrow N_{i,j,k,\cdots,end}$$

Based on this idea, each linear chains starting from nuclide N_i has a weight, and according to these weights, the concentration contributions from nuclide N_i to its reachable nuclides(possibly include N_i itself) are found by combining all analytic solutions of linear chains. After all nuclides are processed in this way, the solution of depletion problem is obtained. However, the resulting code following this approach is very inefficient.

2.2 NEW APPROACH TO TTA METHOD

Deeper investigation into the analytic expressions of linear chain and chain splitting process reveal some tempting properties possessed by TTA method.

1. The solution expressions to Bateman equation is very symmetric, the $(i+1)th$ nuclide in the chain can be solved when the ith nuclide's concentration function is already known. Because the ith nuclide is the only direct "source" to $(i+1)th$ nuclide, which could also be understood by examine of the expressions of Bateman equation, this clearly implies that recurrence relations might exist.
2. Most of the linear chains have common parts with other closely related chains. This can be readily seen from the fact that branching point which has more than 2 branches appears very frequently in the chain especially when particle flux involved (otherwise, the number of linear chains should not be enormous).

If these properties of TTA method were exploited, computational effort should be reduced without loss of accuracy.

2.2.1 MODEL THE DEPLETION PROBLEM AS A DIRECTED GRAPH

Due to the inherent property of depletion problem, only nuclide that is not stable, hence $\lambda_{total} = \lambda_{decay} + \lambda_{flux} \neq 0$, has daughters, this guarantee that branching ratios can always be properly defined. Define all nuclides in concern to be vertices, and transmutation relations among them to be directed edges. Then attribute destruction constants $\lambda_{i,total}$ to all vertices, and branch ratios to all edges. The original depletion problem is modeled equivalently as a directed graph.

In the established model, all coefficients that appear in the solution expressions are built in vertices. Which will facilitate the introduction of new algorithm.

2.2.2 ALGORITHM DESCRIPTION

Flow diagram in Fig. 1 is an brief overall description of the proposed TTA algorithm. The concept of dynamic programming was integrated in the designed flow diagram. Most linear chains are not solved from scratch, instead useful information from closely related chains are preserved and utilized.

Some detailed aspects of the algorithm are not shown explicitly in the flow diagram. One of them is about the calculation of coefficients appear in the solution expressions to Bateman equations. At first the following recurrence relations were found and proved for non-general solution expressions, let $\beta_{m,i}$ be the coefficient corresponds to $exp(-\lambda_i t)$ term in the expression for mth nuclide, we have:

$$\beta_{1,1} = N_1(0) \tag{14}$$

$$\beta_{m,i} = \frac{\lambda_{m-1}\beta_{m-1,i}}{\lambda_m - \lambda_i} \quad 2 \leq m \leq n,\ 1 \leq i \leq m-1 \tag{15}$$

$$\beta_{m,m} = N_m(0) - \sum_{i=1}^{m-1} \beta_{m,i} \tag{16}$$

Where $N_m(0)$ denotes the initial number density of mth nuclide. Recurrence relations for general Bateman solution,

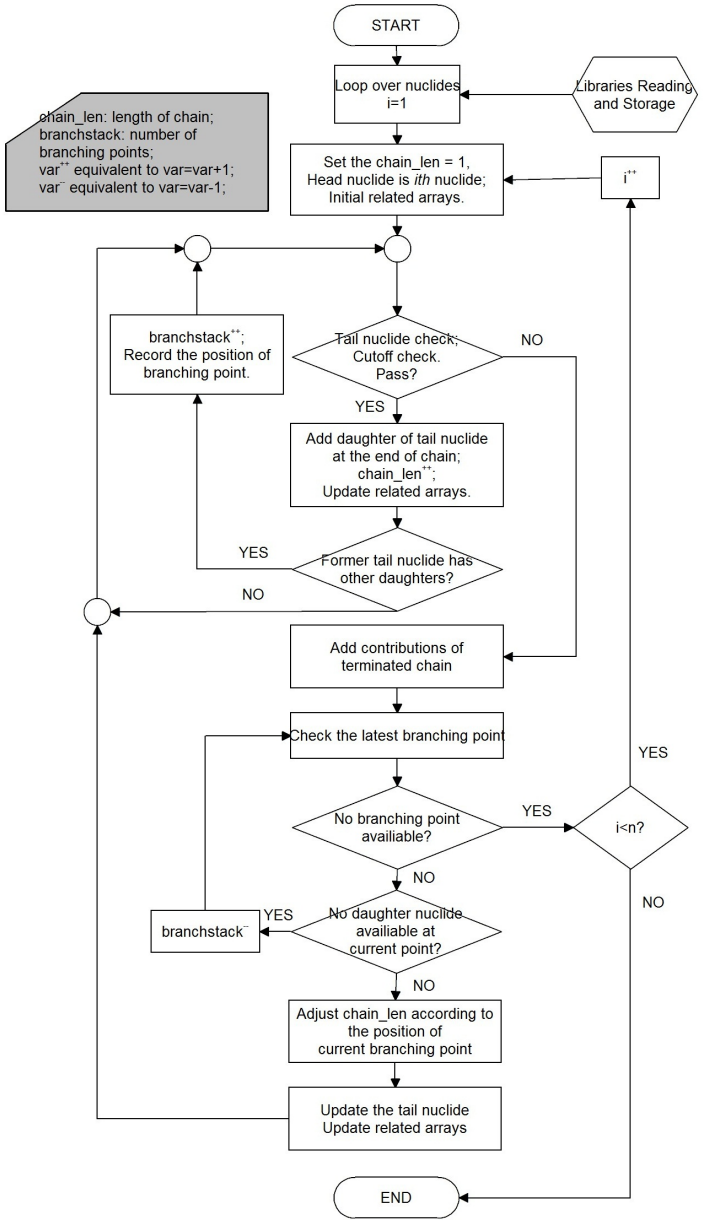

FIGURE 1. FLOW DIAGRAM OF PROPOSED TTA ALGORITHM.

which is somewhat more complicated, could also be derived. At first, ith nuclide's analytic expression has the following form:

$$N_i(t) = \sum_{j=1}^{a_i} \sum_{k=0}^{b_{i,j}} \gamma_{i,j,k} t^k \exp^{-\lambda_j t} \tag{17}$$

In the above presented expression, a_i is maximum number of exponential functions that ith nuclide could have, it has a clear physical meaning as the number of distinct eigenvalues induced by first i nuclides in the chain; $b_{i,j}$ is the multiplicity of jth eigenvalue in ith nuclide, it equals to the number of occurrences of jth eigenvalue. Finally, $\gamma_{i,j,k}$ is the coefficient corresponds to the term $t^k \exp^{-\lambda_j t}$.

Destruction coefficient of jth nuclide is denoted by $\tilde{\lambda}_j$, in order to distinguish it from λ_j that reside in the recurrence expressions. As starting point, the following equations hold naturally:

$$\begin{aligned} a_1 &= 1 \\ b_{1,1} &= 0 \\ \lambda_1 &= \tilde{\lambda}_1 \\ \gamma_{1,1,0} &= N_1(0) \end{aligned} \tag{18}$$

The following recurrence relations settle down mth nuclide directly from $(m-1)th$ nuclide. For all λ_j that satisfy $\lambda_j \neq \lambda_m$:

$$\begin{aligned} \gamma_{m,j,b_{m-1,j}} &= \frac{\lambda_{m-1}\gamma_{m-1,j,b_{m-1,j}}}{\lambda_m - \lambda_j} \\ \gamma_{m,j,b_{m-1,j}-1} &= \frac{\lambda_{m-1}\gamma_{m-1,j,b_{m-1,j}} - b_{m-1,j}\gamma_{m,j,b_{m-1,j}}}{\lambda_m - \lambda_j} \\ &\cdots \\ \gamma_{m,j,0} &= \frac{\lambda_{m-1}\gamma_{m-1,j,0} - \gamma_{m,j,1}}{\lambda_m - \lambda_j} \end{aligned} \tag{19}$$

If there is λ_j that equals λ_m, then:

$$\begin{aligned} a_m &= a_{m-1} \\ b_{m,j} &= b_{m-1,j} + 1 \\ \gamma_{m,j,b_{m,j}} &= \frac{\lambda_{m-1}\gamma_{m-1,j,b_{m-1,j}}}{b_{m,j}} \\ &\cdots \\ \gamma_{m,j,1} &= \lambda_{m-1}\gamma_{m-1,j,0} \\ \gamma_{m,j,0} &= N_m(0) - \sum_{\substack{j=1 \\ \lambda_j \neq \lambda_m}}^{a_m} \gamma_{m,j,0} \end{aligned} \tag{20}$$

If λ_m is different from all eigenvalues λ_j $j = 1, ..., a_{m-1}$, then:

$$
\begin{aligned}
a_m &= a_{m-1} + 1 \\
b_{m,a_m} &= 0 \\
\lambda_{a_m} &= \tilde{\lambda_m} \\
\gamma_{m,j,0} &= N_m(0) - \sum_{j=1}^{a_{m-1}} \gamma_{m,j,0}
\end{aligned} \tag{21}
$$

These recurrence relations should produce exactly the same solutions with respect to "direct" version expressions if roundoff errors are not taken into consideration. It is found by lots of tests that the possible roundoff errors are practically negligible for both solutions built from recurrence relations or "direct" expressions. And apparently the usage of recurrence relations for calculating coefficients in solution expressions will improve the efficiency of resulting TTA code, since most linear chains could retrieve useful information from their closely related relatives.

2.3 EFFICIENCY ANALYSIS

The most time consuming parts of TTA algorithm include calculations of coefficients in solution expressions and scalar exponentials. The former part is the main target to be improved, while the latter one's computational consumption is less important and hard to be compressed further. A back of envelope estimation of algorithm's time complexity and space complexity is presented here.

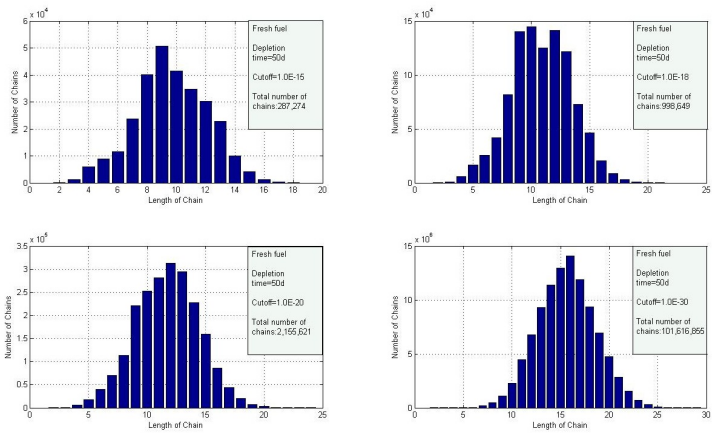

FIGURE 2. LENGTH DISTRIBUTION OF LINEAR CHAINS OF TEST CASE 1.

Based on practice and analysis, for a given cutoff value and a given depletion problem, a properly designed TTA algorithm should search and solve almost a fixed amount of linear chains; and the length distribution of these chains is very like to normal distribution, as illustrated in Fig. 2. The time spending is composed of fractional contributions from all linear chains, so the efficiency analysis on solving a single linear chain is of vital importance.

Let L_i denote the length of ith linear chain, then the calculation of coefficients appear in the solution expressions should require $\sum_{j=1}^{L_i}\sum_{k=1}^{j} j = \Theta(\frac{L_i^3}{3})$ flops. However, with dynamic programming technique and recurrence relations integrated in, the new algorithm described above is able to take advantage of elimination of enormous overlapping subproblems in the process of solving massive amount of linear chains. The overall required flops is bounded by $O(L_i)$. Let N denote the total number of linear chains, then the direct version algorithm needs $c_1 \cdot N \cdot L_{average}^3$ flops to calculate all coefficients, while above described algorithm needs only $c_2 \cdot N \cdot L_{average}$ flops to do the same job. Besides calculation of coefficients appeared in the analytic solution expressions, some additional operations such as searching chains and computing scalar exponentials are also needed. These could be taken into account by adding $d \cdot N$ flops, where d is a constant. The time complexity of direct version algorithm is $\Theta(c_1 \cdot N \cdot L_{average}^3 + d \cdot N)$, newly designed algorithm's time complexity is $\Theta(c_2 \cdot N \cdot L_{average} + d \cdot N)$.

The typical value of L_i is 10 to 20, c_1 is about $\frac{1}{3}$, c_2 is about 2 to 4. At least an order of magnitude of speedup should be observed.

Space complexity analysis is much easier to be carried out, the direct version algorithm only needs arrays to store the identifiers and destruction coefficients of nuclides in the chain. Its space complexity is $\Theta(L_{max})$. The newly designed algorithm needs a little more space to store additional information as mentioned above, its space complexity is $\Theta(L_{max}^2)$. According to practice, this increase of space requirement is smaller than one Megabit.

3 INCOMING SOURCE RESPONSE METHOD

The depletion problem could be modeled as a directed graph with some additional attributes assigned to vertices and edges. If all parents of a vertex(or a nuclide) is solved, the source of that vertex is known, so this vertex is solvable. The incoming source response method(ISR) is based on this fact, and integrated in the depletion calculation code. At first, the directed graph is inverted, facilitating the source building processes; a subroutine will initialize each vertices' expression to be $N_i(0)e^{-\lambda_i t}$(corresponding to the situation of disabling all edges.), and a series of subroutines are responsible for collecting source, and calculating the expression of response for a specific vertex; while the main routine decides the scan scheme over all vertices.

This new method is very suitable for solving decay cases,

where every vertex is a strong connected component on itself. Which means the incoming source response method is capable of giving each vertex a explicit expression after only one round of scan, as illustrated in Fig. 3. In realistic situations, the maximum chain length appear in decay problem is well below 30, so that the problem is not a decay problem if the inverse searching procedure was already done much more times than 30. When particle flux is present, very large size of strong connected components are very likely to exist, which is found by applying corresponding graph algorithm on depletion matrices of real cases. As a consequence, the existence of one round scanning scheme is not guaranteed, currently the scan process is done sequentially over all vertices for some number of rounds under such circumstances. When rth round scan is finished, for a particular vertex, all vertices that lie in the range of distance r from the vertex in concern are taken into consideration for building the concentration expression of this vertex. The typical value of linear chains is 10 to 20, as discussed above, so that $10 \sim 20$ rounds of scan should be enough.

Since each vertex will have an explicit concentration expression after execution of code resulting from this method, and such expression is constituted of functions in the form of $t^{m_i} \cdot e^{\lambda_i t}$, $m_i \geq 0$ and $m_i \in \mathbb{Z}$. Introduction of particle flux might cause the number of functions that needed by vertices to increase aggressively while scan process proceeds. The space complexity of current version of this algorithm is $\Theta(N^2)$ in such situation, in which N is the number of vertices, the space consumption could grow to several hundreds of Megabits for ORIGEN2 scale problems. With so much data to deal with, the computation tends to be slow and numerically unstable. It is for this reason that, only the decay problem is recommended to be solved by this method right now. Because this method is equivalent in solving decay problem with TTA, and the numerical solutions are essentially all the same with TTA, so its performance is not discussed in this paper.

4 NUMERICAL RESULTS OF TTA

Test cases presented below applies ORIGEN2 libraries, and the decay library contains 1307 nuclides. To illustrate the accuracy and efficiency of the developed TTA code, three test cases are selected and the double precision version of TTA is compared with fourfold precision version of TTA having $cutoff = 10^{-30}$ and double precision CRAM of order 14. Both instantaneous values and integral values of concentrations are taken into consideration. The integral concentrations are defined straightly as:

$$N_i^{t_1 \to t_2} = \int_{t_1}^{t_2} N_i(t)\, dt \tag{22}$$

1. Test case 1: Initial composition is fresh UO_2, neutron flux is set to be $3.0 \times 10^{16}/(cm^2 \cdot s)$, depletion time is 50 days. The neutron flux is much higher than most practical values and impose great pressure on TTA code.
2. Test case 2: Same as test case 1, but the initial composition is set to be the end of time composition of test case 1.
3. Test case 3: Pure decay problem. Initial composition is depleted fuel that experienced 350 days of neutron radiation with neutron flux intensity of $3.0 \times 10^{16}/(cm^2 \cdot s)$, and depletion time is one million years.

TABLE 1. DETAILED RUNNING INFORMATION OF TTA WITH VARIOUS CUTOFF VALUES, TEST CASE 1.

Double precision TTA				
Cutoff	Number of linear chains	Average chain length	Maximum chain length	Time consumption(s)
10^{-15}	287274	9.6377	18	0.12
10^{-16}	440577	10.0314	19	0.18
10^{-17}	681840	10.4498	20	0.31
10^{-18}	998649	10.8657	21	0.48
10^{-19}	1455925	11.3081	22	0.72
10^{-20}	2155621	11.6894	24	1.11
Fourfold precision TTA				
10^{-20}	2167242	11.7115	22	3.500662E+01
10^{-30}	101616855	15.6242	29	2.268395E+03

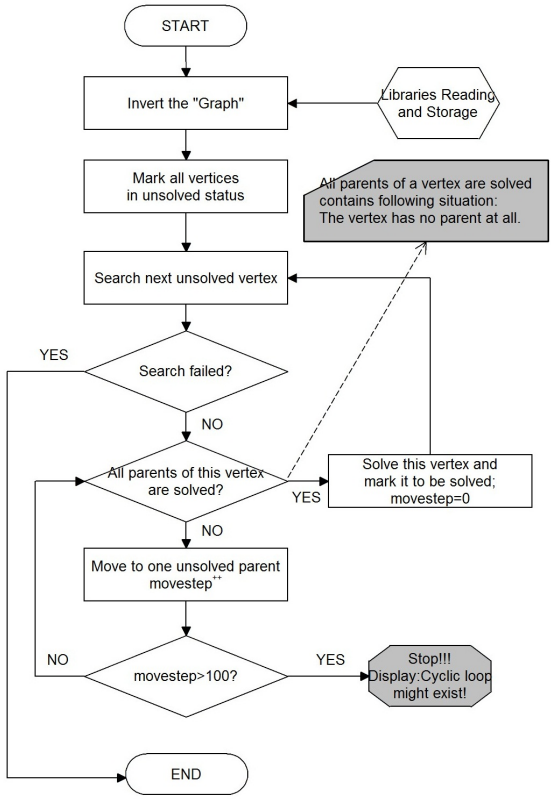

FIGURE 3. FLOW DIAGRAM OF INCOMING SOURCE RESPONSE METHOD FOR SOLVING DECAY PROBLEM.

As long as the accuracy is in concern, the reference solutions are provided by fourfold precision TTA. In order to make sure that this version of TTA code is properly programmed, CRAM of order 14 is employed to offer comparable solutions. And the cross check results support the availability of fourfold precision TTA. The reliability of CRAM code origin from the characteristics of the spectrum of depletion matrices, and was deeply investigated [9, 10].

Fig. 4 and Fig. 5 show the logarithm of relative errors($\log_{10}\left|\frac{n_{i,cmp}-n_{i,ref}}{n_{i,ref}}\right|$) according to relative importance that defined as logarithm of the fraction of total concentration contributed by one particular nuclide($\log\frac{n_{i,ref}}{n_{total,ref}}$). The closer to right border the point is, the lower the point should be, because much higher accuracy is desired if the nuclide is responsible for much more fraction of total concentration in general. Three noticeable phenomena could be found from these two figures. At first, cutoff being set to 10^{-20} rather than 10^{-15} results in better solution; secondly, the solution obtained with cutoff equals 10^{-15} is accurate enough for most applications, as nuclides that count for more than 10^{-8} fraction of the total concentration have their relative errors bounded under 10^{-5}; thirdly, the integral values

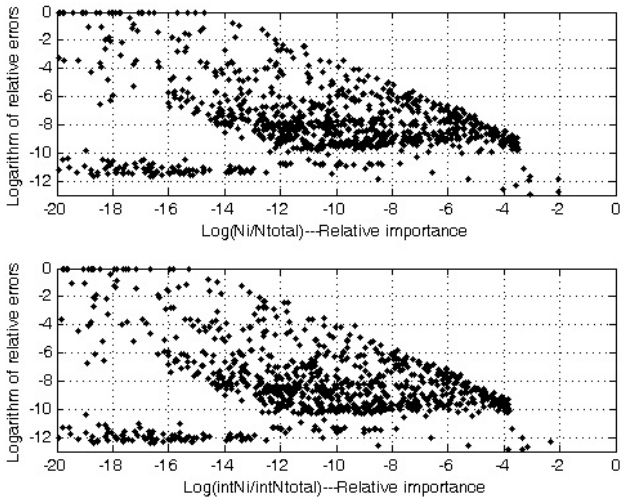

FIGURE 4. TEST CASE 1. REPRESENTATION OF RELATIVE ERRORS OF BOTH INSTANTANEOUS VALUES(UPPER SUBPLOT) AND INTEGRAL VALUES(LOWER SUBPLOT). DOUBLE PRECISION TTA WITH $CUTOFF = 10^{-15}$.

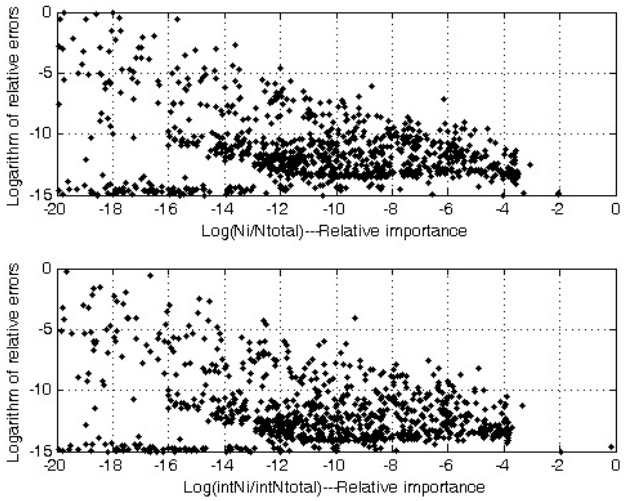

FIGURE 5. TEST CASE 1. REPRESENTATION OF RELATIVE ERRORS. DOUBLE PRECISION TTA WITH $CUTOFF = 10^{-20}$.

maintain the same level of accuracy with instantaneous values.

Fig. 6 and Fig. 7 suggest that the double precision TTA could also deal with radiation or decay calculation of depleted fuel. Test case 3 is a decay problem, the cutoff value is set to be 0, because no cyclic loop exist in the transmutation network, so the number of linear chains does not turn out to be infinity. As a

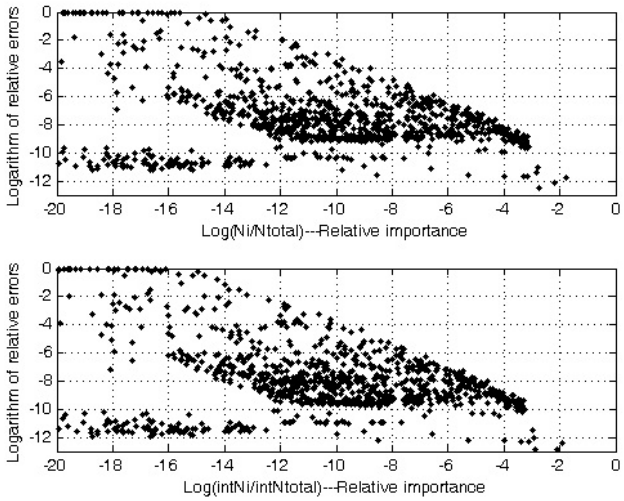

FIGURE 6. TEST CASE 2. REPRESENTATION OF RELATIVE ERRORS. DOUBLE PRECISION TTA WITH $CUTOFF = 10^{-15}$.

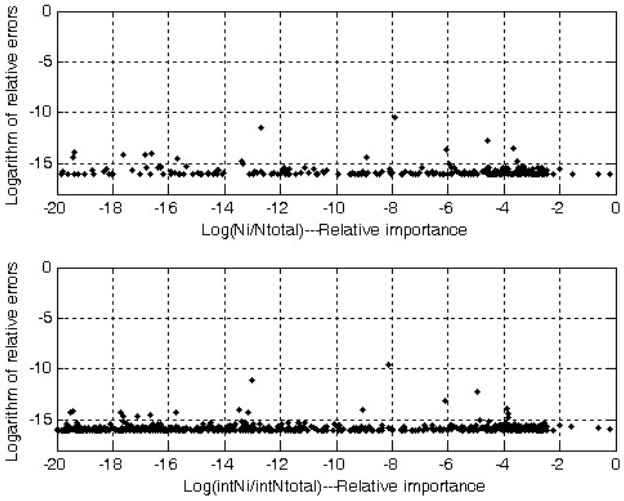

FIGURE 7. TEST CASE 3. REPRESENTATION OF RELATIVE ERRORS. DOUBLE PRECISION TTA WITH $CUTOFF = 0.0$.

matter of fact, several thousands of linear chains is typical in decay problem, and no linear chain is terminated due to cutoff or instability in test case 3. The time consumption for test case 3 is about one millisecond,since the running time of TTA is proportional to the number of linear chains.

The TTA code is executed on a 3.20GHz machine. Table 1 gives out some important execution information of test case 1. As we can see, the TTA code behaves as predicted in section 2.3.

TABLE 2. ADDITIONAL TEST CASE: PWR, $FLUX = 3.0 \times 10^{14}/(cm^2 \cdot s)$, INITIAL COMPOSITION IS FRESH UO_2. TIME SPENDING OF VARIOUS ALGORITHMS

Depletion time	10d	20d	30d	40d	50d
CRAM of order 14	6 - 7ms				
TTA with $cutoff = 10^{-15}$	8ms	12ms	14ms	17ms	20ms
TTA with $cutoff = 10^{-20}$	38ms	61ms	84ms	0.105s	0.128s

And the running time is about 0.1 second if cutoff is 10^{-15}, this is about the same with the time required by ORIGEN2 to read and process the libraries, and time required by libraries reading process in TTA code is about 0.01s since no special treatments is needed. If the initial composition is depleted fuel, a slowdown factor of 2 to 3 should be expected; and if the depletion time is shorter than 50 days or flux level is lower than $3.0 \times 10^{16}/(cm^2 \cdot s)$, speedup at some degree is natural, as shown in Tab. 2. Nevertheless, the resulting TTA code is efficient enough to be applied in practice, it takes nearly the same amount of time in each calculational step with ORIGEN2 and provides better solutions. The integral values are also given with high accuracy that consistent with instantaneous values.

5 CONCLUSION

(1) A new transmutation trajectory analysis(TTA) algorithm exploiting some basic characteristics of depletion problem by integrating dynamic programming concept in code design is proposed, implemented and tested. The TTA algorithm was bombarded with a lot of test cases, and a few cases are chosen to demonstrate its efficiency and accuracy. It is shown that the accuracy does not degenerate as the direct expressions are replaced with recurrence relations. Although the computational spending of TTA is very sensitive to flux level, initial composition, cross-sections, depletion time and cutoff value, compared with ORIGEN2, this new algorithm is capable of producing more accurate solutions with computational spending confined at the same order of magnitude.

(2) The Incoming Source Response method, which provides concentration functions instead of scalar concentration values, is also developed and tested, however it is only suitable for decay calculation as far as now, and it is equivalent to TTA in solving decay problem theoretically and practically.

CRAM method was proved to be both efficient and accurate, however, integral values calculation and long time decay calculation, that can be solved elegantly by TTA, are challenging subjects to CRAM method till now. Generally speaking, the newly developed TTA algorithm is reliable, versatile and efficient. Currently, the two analytic algorithms discussed above and two other matrix exponential algorithms are already integrated in a depletion calculation package, and the package could be easily coupled with neutron transport code.

ACKNOWLEDGMENT

This work is supported by the National High Technology Research and Development Program ("863" program) of China (Approved number: 2013AA051402) and the National Natural Science Foundation of China (Approved number: 91126005).

REFERENCES

[1] M. Pusa, and J. Leppäanen, 2010. "Computing the matrix exponential in burnup calculations". *Nucl. Sci. Eng.,* ***164***, pp. 140–150.

[2] Ding She, PHYSOR 2012. "Using laguerre polynomials to compute the matrix exponential in burnup calculations".

[3] Akio Yamamoto, Masahiro Tatsumi, and Naoki Sugimura, 2007. "Numerical solution of stiff burnup equation with short half lived nuclides by the krylov subspace method". *Journal of Nuclear Science and Technology,* ***44***, pp. 147–154.

[4] A.G. Croff, 1980. *A User's Manual for the ORIGEN2 Computer Code*. Oak Ridge National Laboratory, Oak Ridge, Tennessee 37830, July.

[5] O.W. Hermann, and R.M. Westfall, 1998. *ORIGEN-S: Scale System Module to Calculate Fuel Depletion, Actinide Transmutation, Fission Product Buildup and Decay, and Associated Radiation Source Terms*. Oak Ridge National Laboratory, Oak Ridge, Tennessee, September.

[6] M. Pusa, and J. Leppäanen, 2011. "Rational approximations to the matrix exponential in burnup calculations". *Nucl. Sci. Eng.,* ***169***, pp. 155–167.

[7] H. Bateman, 1910. "The solution of a system of differential equations occurring in the theory of radio-active transformations". *Proc. Cambridge Philos. Soc.*, February, pp. 423–427.

[8] J. Cetnar, 2006. "General solution of Bateman equations for nuclear transmutations". *Annals of Nuclear Energy,* ***33***, April, pp. 640–645.

[9] C. Moler, and V.L. Charles, 2003. "Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later". *SIAM Review,* ***45***, pp. 3–49.

[10] L.N. Trefethen, J.A.C. Weideman, and T. Schmelzer, 2006. "Talbot quadratures and rational approximations". *BIT Numer. Math,* ***46***, p. 653.

APPENDIX A: PSEUDO CODE OF TTA ALGORITHM

◯ Loop over all nuclides: outmost
· i=1,n
  · Add the *ith* nuclide to the head of the chain, by setting $chain(1) = ID_i$, ID_i is the identifier of *ith* nuclide.
  · Set $chain\_len = 1$, $branchstack = 0$, $changepoint = 0$ $cimp(1) = 1.0$, $lamb(1) = \lambda_i$, $transv(1) = e^{\lambda_i t}$.
  ○ Loop to grow the chain: inner1
    · The last nuclide in the chain (tail nuclide) is stable? ⇒ *exit inner1*
    · Cutoff criterion check is met? ⇒ *exit inner1*
    · Number of tail nuclide daughters, if zero ⇒ *exit inner1*
    · Append the first daughter to the tail of the chain. and update all related data, for example: $chain\_len^{++}$, $cimp(chain\_len) = cimp(chain\_len) * b_{chain(chain\_len-1) \to chain(chain\_len)}$
    · If number of daughters greater than one ⇒ $branchstack^{++}$, and store branch point position in $branch(brachstack)$.
  ○ end loop inner1
  · Check the ended chain, record data in counters.
  ○ Loop to add contributions from a ended chain: inner2
  · j=changepoint+1,endpoint
    · Add $concentration(chain(j)) * transv(j) * cimpv(j)$ to nuclide chain(j).
    · If integral value required, add them in the same manner.
  ○ end loop inner2
  ○ Loop turn around:inner3
    · If $branchstack == 0$ ⇒ *exit outmost*
    · Check the last branch point— $brach(branchstack)$.
      More daughters exist at this point? ⇒
        update the tail nuclide;
        update $chain\_len$, $changepoint$, etc;
        ⇒ *exit inner3*
      Otherwise $branchstack^{--}$
  ○ end loop turn around:inner3
  · While the tail nuclide was changed, update all related arrays.
◯ end loop over all nuclides: outmost

chain_len The integer that specify the length of the linear chain.
chain The integer vector that contains nuclide identifiers which form the chain.
ID_i The identifier of *ith* nuclide.
branchstack Integer. Indicate the number of branching points in the linear chain.
branch Integer vector. Keep all branching points' positions.

changepoint Integer. Indicate the latest branching position that share with the last chain in the current chain.

lamb Real vector. Keep all nuclides' destruction coefficients.

transv Real vector. The concentrations of corresponding nuclides that result from the unit concentration of head nuclide, with all branch ratios along the chain being set to 1.0.

cimp Real vector. Defined as $cimp(i) = \prod_{j=1}^{i-1} b_{chain(j)\to chain(j+1)}$, thus, information about branches are stored in it. Combined with $transv$, the concentration of a particular nuclide in the chain could be found.

Cutoff check The cutoff check is done by adding a pseudo stable nuclide at the end of the chain, and the calculated concentration of this nuclide at the end of depletion time is the total "leakage" from original chain.

Meaning of var^{++} and var^{--} var^{++} is equivalent to $var = var + 1$, var^{--} is equivalent to $var = var - 1$.

Type of counters Length of chain, and the reason of termination of the chain.