Proceedings of Global 2009
Paris, France, September 6-11, 2009
Paper 9166

Studying international fuel cycle robustness with the GENIUSv2 discrete facilities/materials fuel cycle systems analysis tool

Kyle Matthew Oliver, Paul P. H. Wilson, Arnaud Reveillere, Tae Wook Ahn, Kerry Dunn, Katy Huff, Royal Elmore
University of Wisconsin-Madison, Department of Engineering Physics, 419 Engineering Research Building, 1500 Engineering Drive, Madison, WI 53706
Tel: (608) 263-0807, Fax: (608) 263-4499, Email: wilsonp@engr.wisc.edu

Abstract – *GENIUSv2 is a discrete-facilities/discrete-materials nuclear fuel cycle systems analysis tool currently under development at the University of Wisconsin-Madison. It is designed not only to compute the capacity and mass flow data produced by other study tools but also to model the complex relationships between nuclear fuel cycle facilities; the institutions that own them; and the national, intranational, and/or transnational regions in which they operate. This paper describes novel modeling capabilities and methodological contributions present in GENIUSv2, including its treatment of the region-institution-facility hierarchy and its optimization-compatible framework. We then present a number of test problems designed to demonstrate the code's ability to closely reproduce results from another study tool (Idaho National Laboratory's VISION code) and to richly model multi-region scenarios not easily captured by the data models of other codes currently available.*

I. INTRODUCTION

Several properties of global nuclear fuel cycles as dynamic systems make them especially difficult to model and analyze. Among these properties are the number of technologies one needs to simulate, the sensitivity of system-wide properties to small variations in the operation of those technologies, and the variety and complexity of the policy dimensions that directly influence system design and behavior. These problems call for system study tools that are at the same time detailed, flexible, robust, and generative.

By *detailed*, we mean that they model a wide range of information about the nuclear fuel cycle scenarios being considered. Detailed study tools must model very specific facility deployments and facility operation modes. By *flexible*, we mean that they adapt well to new approaches for how those facilities should work together. Flexible tools are as free as possible from assumptions about what an advanced fuel cycle flowsheet will look like and are easy to modify or augment to model new approaches. By *robust*, we mean that they store and process the necessarily large data sets efficiently. Robust tools should use modern computing libraries and other resources. By *generative*, we mean that they can identify new, promising, or optimal fuel cycle scenarios. A generative system analysis tool is in a sense also a system *design* tool.

The GENIUS project (Global Evaluation of Nuclear Infrastructure Utilization Scenarios) started at Idaho National Laboratory as the top-level, nuclear enterprise simulation tool in the Simulation Institute for Nuclear Enterprise Modeling and Analysis (SINEMA) framework[1]. As specified in that framework, GENIUS represented a promising approach with respect to the four criteria outlined above. The first implementation of the GENIUS code (hereafter "GENIUSv1") was developed by Chris Juchau and Mary Lou Dunzik-Gougar and has been described in detail elsewhere[2,3]. An important step in this work was Juchau's review of existing nuclear fuel cycle systems analysis tools[4], wherein he identified important gaps in current modeling capabilities, gaps that would be filled by a GENIUS tool that functions as specified by SINEMA. GENIUSv1 met many of the requirements in that specification and demonstrated the kind of supply-and-demand data one can generate with a multi-region, discrete-facilities/discrete-materials (DF/DM) code[5].

Work on GENIUSv2 has proceeded at the University of Wisconsin-Madison since January 2007. In light of our four criteria for improved system study tools, it has been completely redesigned and re-implemented as an object-oriented C++ application with Python-based pre- and post-processing. This paper presents results from the GENIUSv2 testing and demonstration suite, after briefly discussing the following list of improvements to GENIUS modeling capability and computational methodology:

1. To improve the detail of the GENIUS tool, an intermediate level was added to the model hierarchy. GENIUSv2 models nuclear fuel cycle facilities (reactors, fuel fabrication, enrichment, etc.), the institutions that own them (e.g., utilities and governments), and the regions that those institutions operate in (e.g., countries or subdivisions thereof). In light of the economic and political questions raised by international proposals like the Global Nuclear Energy Partnership program, the need to model multi-region, multi-institution scenarios is especially pronounced.
2. To improve detail and flexibility, the material-modeling hierarchy has been elaborated. GENIUSv2 tracks individual "lots" of fuel cycle feedstocks (yellowcake, UF_6, separated materials, etc.) as well as individual fuel assemblies, but it still groups assemblies into batches (the smallest unit of material in GENIUSv1) for ease of use.
3. To improve the tool's generative capability, a user-driven facility deployment mode was added to the more typical demand-driven mode. As will be explained, a user-driven mode is much more consistent with natural methods for optimization of the fuel cycle system.
4. To further support optimization (and as a simple consequence of the DF/DM paradigm combined with a user-driven deployment in which supply of fuel cycle commodities may not always equal demand for them) a general algorithm has been developed for sensibly, consistently, and flexibly matching suppliers of a given fuel cycle commodity to their customers.
5. To improve the tool's robustness and ultimately support end-user data analysis, we have implemented a database methodology for storing and reconstructing the complete material and facility histories, which comprise a large, sparse, multidimensional data set.

II. MODELLING

II.A. Regions, Institutions, and Facilities

Several of the currently available nuclear fuel cycle systems analysis tools—including VISION[6], the Idaho National Laboratory code that is the tool-of-choice for the U.S. Department of Energy—do not easily support multi-region simulations. Given the increasing interest in internationalization of the fuel cycle[7], this difficulty is something of a liability, especially since a primary goal of multi-region systems analysis (where regions in this case would mostly likely represent separate countries) is to investigate potential policy mechanisms for inter-region cooperation. For instance, one especially interesting question multi-region codes can help answer involves fuel sale and/or leasing agreements between nations that own and operate their own fuel cycle facilities and those that do not. Nations that agree to forego developing fuel cycle infrastructures are likely to be quite concerned about the robustness of their fuel supply to interruptions caused by the changing winds of international relations.

GENIUSv1 was designed from the start to be a multi-region model. However, we determined for GENIUSv2 that another hierarchical level of granularity made sense in light of the facility-ownership structure in place in many large nuclear states. Thus, we introduced institutions, which represent the private and governmental organizations that own nuclear facilities in a particular region. As we go forward, this region-institution-facility (R-I-F) hierarchy will allow fairly detailed financial modeling of the global nuclear enterprise. For instance, the financial behavior and performance of two nuclear reactors operating in two different regions, or for two different institutions in the same region, are likely to be measured and evaluated quite differently based on debt structures, regulatory environments, etc. Thus, each region, institution, and facility can be assigned financial parameters such as tax and interest rates, in addition to a great deal of technical information about how they actually operate.

Besides capturing the effects of institutional and regional heterogeneities, the R-I-F framework also provides the mechanism for communication and cooperation between the discrete facilities in the simulation. In particular, facilities need a mechanism for sending material orders to one another. As seen in Fig. 1, they do so by writing an offer of or request for material and passing this message up this hierarchy to the simulation manager, which performs a matching algorithm on the set of messages based on some set of rules (see *III.B. Customer-Supplier Matching*) and then passes instructions back down the hierarchy to the appropriate facilities.

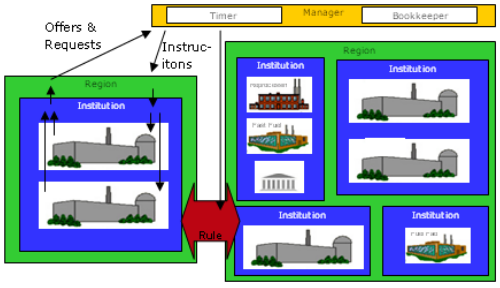


Fig. 1. GENIUSv2 inter-facility communication and cooperation model[8]. Note that the simulation manager makes decisions about material routing based on some set of rules about how the regions, institutions, and facilities in the model's hierarchy are supposed to relate to one another.

The GENIUSv2 implementation makes efficient use of the object-oriented programming concept of inheritance,

taking advantage of the fact that all nuclear fuel cycle facilities share a set of common data types (each has some construction time, operational time, cycle time, characteristic costs, etc.) and generic operational schema (request some feed material from a facility upstream, do some operation on it, and offer it to a facility downstream). All fuel cycle facility types, including reactors, are implemented as subclasses of an abstract facility class which establishes their common data and behavior. This practice eliminates code redundancy and facilitates flexibility and modularity; for each subclass, developers need only implement the data and behaviors unique to that particular type of facility.

II.B. Materials

Because it is intended to be used flexibly to support closed fuel cycle simulations, waste management studies, and non-proliferation analyses, GENIUSv2 must support a wide variety of material types and keep a complete record of the isotopic and facility histories of each individual material object in the simulation. To allow materials to flow to facilities in the smallest reasonably achievable quanta, we decided to model reactor fuel on an assembly-by-assembly basis, rather than at the batch level as in GENIUSv1 (a batch in GENIUSv2 is simply a collection of fuel assembly objects). This decision has the added advantage of leaving the door open for GENIUSv2 to be at least loosely coupled to some sort of heterogeneous core physics package, although for the time being, we have employed the standard "recipe"-based reactor physics approximations used in VISION and other codes.

Except for fuel assemblies, all other fuel cycle commodities are modeled as general "lumps" of material. Thus, in addition to their isotopic and facility histories, these objects keep a record of what type of commodity they are (yellowcake, enriched or unenriched UF_6, separated actinides, etc.) and what form (solid, liquid, gas) they are in. When not being directly operated on, they are stored in "buffers" at each facility, queues of material that represent the holding areas where material is stored before and after enrichment, fabrication, irradiation, reprocessing, etc. but before being sent to another facility.

Two somewhat novel features of the GENIUSv2 materials model are its flexible isotopic vectors and the "decay-on-demand" approach used to track how those vectors change over time. The former feature allows GENIUS to forgo committing to a hard-coded set of specific isotopes to track, which is attractive because comparing results from codes that track slightly different sets of isotopes can be a minor nuisance. The latter feature derives from the observation that, during runtime, facility operations only depend on the decayed isotopics of certain materials, and only at very specific points in the fuel cycle (e.g., when preparing to reprocess a given batch of spent fuel). Thus, GENIUS materials are only decayed at those times, which eliminates the significant computational burden of decaying every material object in the simulation at every time step. The time step-by-time step composition of each object can be reconstructed in post-processing as needed using the material history transaction points (see *III.C. Material and Facility Tracking*) and a special post-processing implementation of the decay algorithm.

III. METHODS

As in any fuel cycle study tool, a wide range of computational methods are required to simulate the operations each facility performs on the different material streams and, more complexly, the way those operations work together as a system. In this section, we will limit the discussion to novel methods that we believe GENIUSv2 is the first to implement.

III.A. Optimization-Compatible Framework

Among the generative goals for the GENIUS code is the ability to identify promising fuel cycles by supporting one or more optimization modes. Of course, the challenges of global optimization of complex dynamic systems are well studied in the operations research literature (and shown to be tremendously difficult in general). Jain and Wilson have discussed the problem in the context of nuclear fuel cycle system study tools, focusing their attention on the elimination, where possible, of so-called heuristics—local methods that make expedient but "short-sighted" decisions when it is difficult to evaluate the consequences of a particular choice with respect to a global objective function[9]. While heuristics are difficult to eliminate entirely, minimizing their use is an important goal when trying to avoid artificially shrinking the decision space of interest and thus eliminating possible globally optimal solutions.

We propose a framework compatible with the general goal of eliminating heuristics where possible and using global methods to search for promising fuel cycle facility deployments. Inspired by the "optimal sub-problem" approach of dynamic programming, we decompose the fuel cycle optimization problem into a facility deployment problem (FDP) and a materials routing problem (MRP). Using optimization algorithms based on network flow theory (see *III.B. Customer-Supplier Matching* below), we solve or approximate the MRP, determining an attractive materials routing (with respect to a global objective function like a time- and region-averaged cost of electricity) *for the given facility deployment*. If for any deployment we can calculate an objective function value representing that deployment's best-case-scenario materials routing, then we can use wrapper-based, iterative optimization technology (such as Sandia National

Laboratories' DAKOTA[10] tools) to gradually converge to better and better solutions to the FDP. Of course, the decision space for the wrapper algorithm need not be limited to facility deployment; other scenario parameters could be included as well.

It may be some time before this global optimization scheme can be implemented. However, we have described it here in some detail because (a) we have taken care to design an approach that we believe is compatible with this very useful but as-yet unavailable study tool capability and (b) that decision has consequences that may seem puzzling outside of the optimization context (such as the main application's reliance on a user-driven, rather than demand-driven, facility deployment).

III.B. Customer-Supplier Matching

A consequence of the decision to model both facilities and materials discretely is the necessity, unlike in fleet-based, continuous-flow codes, to match individual customer facilities with individual suppliers. When combined with a user-driven, rather than demand-driven, deployment scheme, these matching algorithms need to be general enough to handle the very likely event that supply will not equal demand for a given fuel cycle commodity. In this section, we describe a network-flow formulation of such an algorithm.

The nuclear fuel cycle system can be modeled as a multi-commodity network flow (NF) problem. In a network-flow formulation, the facilities of the fuel cycle serve as nodes—sources (supply nodes) and sinks (demand nodes) for the flow of various commodities (yellowcake, unenriched and enriched UF_6, fuel, etc.) along arcs connecting facilities that are allowed to exchange material with one another. Our task in matching suppliers to customers at a given time step is to collect offers and requests, treating the senders of the former as source nodes and of the latter as sink nodes. The solution of the corresponding NF problem can then be translated into a set of instructions to the source-node facilities to send material to the sink-node facilities.

The form for an *M*-commodity NF problem is given by Bertsekas[11], who notes that there are no practical solution methods for the general case. However, the special network structure of the nuclear fuel cycle allows us to reduce the single *M*-commodity problem to *M* single-commodity problems, which can be solved very efficiently with available network solvers or even general linear program (LP) solvers. The key to this decomposition is the observation that the arc set, *A*, and node set, *N*, can be decomposed by commodity. For instance, the arcs connecting conversion plants to enrichment plants will exclusively carry unenriched UF_6, and the arcs connecting reactors to separations plants will only ever carry spent fuel. Thus, we can decompose the arc and node sets by commodity and write the following *M* problems:

$$\begin{aligned} &\forall m \in \{1,2,\ldots,M\}: \\ &\min \sum_{(i,j)\in A_m} a_{ij} x_{ij} \\ &s.t. x_{ij} \in [b_{ij}, c_{ij}], \forall (i,j) \in A_m \\ &\sum_{\{j|(i,j)\in A_m\}} x_{ij} - \sum_{\{j|(j,i)\in A_m\}} x_{ji} = s_i, \forall i \in N_m \end{aligned} \quad (1)$$

where the x_{ij} is the mass of the material traveling from node *i* to node *j* (i.e., on arc *i*,*j*), a_{ij} is the unit cost of that flow, b_{ij} and c_{ij} are the upper and lower bounds of the mass flow on arc *i*,*j*, and s_i is the signed *divergence* of node *i* (s_i is greater than zero and equal to the supply at *i* if *i* is a source and is less than zero and equal to the demand at *i* if *i* is a sink).

In order to guarantee feasibility (i.e., that an optimal solution can exist), and in particular to handle the case where supply does not equal demand, we add an artificial source or sink node for each problem. The divergences of these nodes are chosen to equalize supply and demand. Flow to or from these artificial nodes measures infeasibility of the real network and thus signifies unused supply or unmet demand, so we set arc costs for these nodes very large, in the hope that the algorithm does not need to push any flow on them (flow on these arcs does not correspond to any real instruction to send material from one facility to another). Fig. 2 illustrates a sample network, including the artificial nodes.

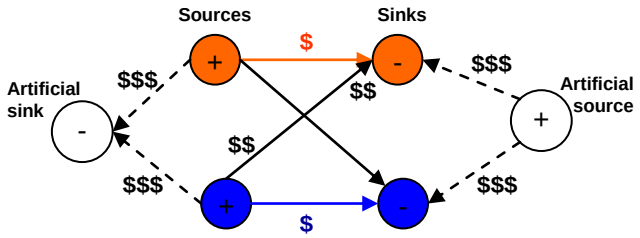


Fig. 2. Schematic of a network flow problem for matching two suppliers of a particular commodity to two customers. Artificial sources and/or sinks are added to ensure feasibility and measure supply/demand imbalance. Arc costs can be lowered between nodes that have a special affinity for trade with each other.

Eventually, arc costs can be dynamically calculated to account for political and economic factors that will introduce constraints and incentives for certain material supply relationships. For example, arc costs may include real material costs, chosen from the *Advanced Fuel Cycle Cost Basis*[12] or other appropriate source. In the meantime, we have implemented a system that sets arc costs in order to match according to *affinities* for trade between elements

in the R-I-F hierarchy. These affinities can be set by the user, but the default is that facilities owned by the same institution have a higher affinity for trading with one another than facilities owned by two different institutions, and facilities in the same region have a higher affinity than facilities in two different regions. At the very ends of the affinity scale are values that cause two facilities to automatically be matched whenever they are both "in the market" or to never be matched even if there is no other option. These latter values can be used to model long-term contracts and embargos, respectively.

III.C. Material and Facility Tracking

Another consequence of our DF/DM approach is that it creates hundreds of thousands if not millions of individual material objects during the life of even modestly-sized systems. And if we want GENIUS to be useful for waste management and non-proliferation studies, we need to be able to fully reconstruct the isotopic histories of each of those materials.

Thus, we have implemented a database solution for efficiently storing isotopic and facility histories of materials that are no longer needed by the system and can be deleted from memory during runtime. When a material is deleted, its history is written to a facility history table (Fig. 3, top). Each entry in the table has a column recording the material's ID number, the time of the transfer, the source and destination facility, and an isotope code. A second table maps those isotope codes to a particular composition via a specially formatted "blob" of binary data.

A transaction-based storage strategy is much more efficient than an array-based one because these data are by nature very sparse (the probability that two random facilities are exchanging materials at a given time step is very small, so most entries in an array representation would be zero). The database approach has the added advantage of being easy to query when the simulation is over, greatly aiding in reconstruction of the simulation. In fact, our light-weight Python post-processor is little more than a wrapper around the database that knows something of the database structure and so can query it as efficiently as possible. Conveniently, other tables in the database double as an input file, thus making it possible to specify very detailed scenarios in a sensible manner (Fig. 3, bottom).

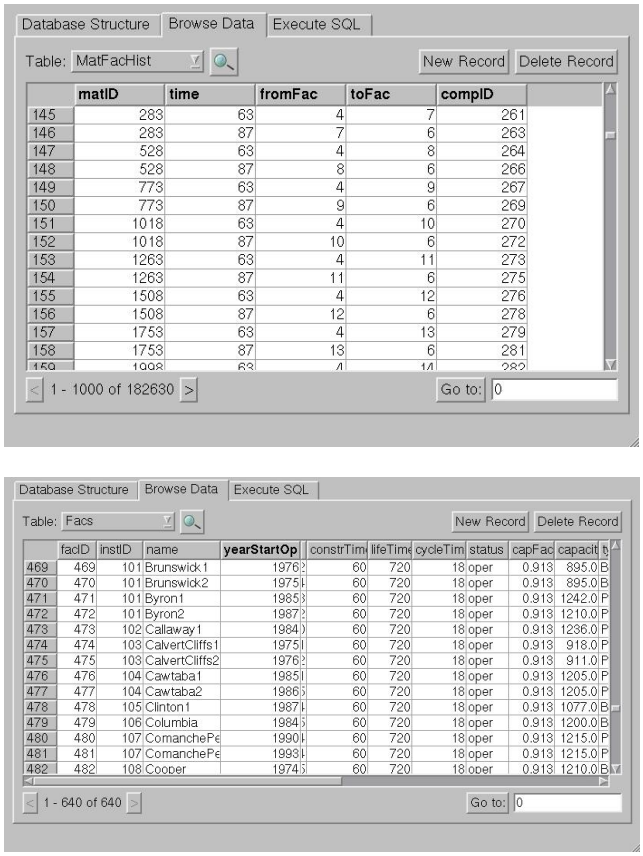


Fig. 3. Screenshots from GENIUS database files. The top image shows the output from the facility history table for a 1000+ reactor problem and almost 200,000 individual material transfers. The bottom image shows part of an input file constructed from the PRIS reactor database.

IV. SAMPLE PROBLEMS AND RESULTS

As the GENIUSv2 source code begins to stabilize and its feature set continues to develop, we are specifying and working through a testing suite that serves several purposes: to confirm that the code is operating as we expect it to, to show that it gives comparable results to other popular codes, to demonstrate the code's performance on large problems, and to highlight some of its key features via problems that are difficult or impossible to model in other codes. We present here a sampling of results from test suite problems currently under investigation.

IV.A. Reactor Benchmarking

Because VISION has emerged as the standard tool for performing fuel cycle systems analysis calculations, we present benchmarking problems comparing GENIUS and VISION results for a series of analogous test problems of increasing complexity. Throughout the discussion it will be important to remember that *analogous* is the operative word. At a certain level of granularity, fleet-based, continuous-flow codes and DF/DM codes are simply

incompatible. However, the overall behavior, including integrated material throughputs, of each model's approximation of any given scenario should and do give nearly to the same answers. Note that each problem was run with VISION's radioactive decay routines turned off, since the analogous routines in GENIUS are still being tested and debugged.

Problem 1 is a single-reactor benchmark with no fuel fabrication constraints, and its purpose is to compare accumulated spent fuel mass and isotopics in GENIUS and VISION. The parameters for this simulation and the three that follow it are given in Table I and represent a combination of parameters that in our experience tend to work well in the VISION model.

TABLE I

VISION-GENIUS Benchmark Reactor Parameters

Parameter	Value
Start Year	2000
End Year	2099
Construction + License Time	6 years
Operating Time, OT	60 years
Power Capacity, P	1050 MWe
Capacity Factor, CF	0.90
Thermal Efficiency, η	0.34
Cycle Time, T	12 months
Fuel Burnup, Bu	51 GWd/tHM
Fuel Batches Per Core, N	5

The fresh and spent fuel isotopics for the GENIUS case are based on the mass fractions from VISION's LWR fuel recipe through interpolation for 51 GWd/tHM of burnup. To discretize this scheme for GENIUS, we need only multiply these fractions by some total fixed core mass, *M*, and then convert each isotopic mass to number density, which is how GENIUS materials store their compositions natively. Since VISION is implemented in a stock-and-flow systems dynamics tool, it calculates a continuous fuel consumption rate by quarter-year time step according to the following formula:

$$\dot{m} = \frac{P(CF)}{\eta(Bu)} \tag{2}$$

This amount of mass, which constitutes (1/*N*)th of the total core, emerges from the reactor during the cycle period T, so the total core mass for a GENIUSv2 reactor using this recipe is

$$M = \dot{m}NT = \frac{PNT(CF)}{\eta(Bu)} = 99.459\frac{tHM}{core} \tag{3}$$

In Problem 1, reactor construction and licensing start in 2000, and once the reactor begins operating it runs for its designated operating time before being decommissioned. We can do a simple hand calculation to compute the total mass we expect to be ejected from a reactor for an idealized real-word refueling scheme: At startup, *N* batches are inserted (total mass *M*). During each normal year of operation one batch is inserted and one ejected (batch mass *M*/*N*). In the decommissioning year, all *N* batches currently in the core are ejected (total mass *M*). Thus, the total ejected mass, M_{ej}, should be (neglecting $E=mc^2$ losses):

$$M_{ej} = \frac{M}{N}(OT-1) + M = 1.273 ktHM \tag{4}$$

Table II shows the results of the hand calculation and the error in the two simulations with respect to that prediction, and Fig. 4 shows the codes' behavior as a function of time. The GENIUS underestimate for the total is due to a conservation of mass violation in our procedure for reproducing the VISION discharge isotopic recipe; we miscalculate the mass of the isotopes that VISION lumps together and labels as "_OTHER" because GENIUS stores number densities rather than masses natively and so cannot compute consistent masses for the _OTHER isotopes. The VISION overestimate results from its particular implementation of modeling a discrete process like refueling in a continuous manner. This analysis suggests that VISION is a suitable reference against which to compare GENIUS, and indicates the magnitude of discrepancy that can be expected in comparing more complex scenarios.

Table II

Total Spent Fuel Mass Comparison – One Reactor

Calculation Method	Total Ejected Mass [kt HM]	Error
Hand Calculation	1.273	--
VISION Simulation	1.293	+1.57%
GENIUS Simulation	1.267	-0.47%

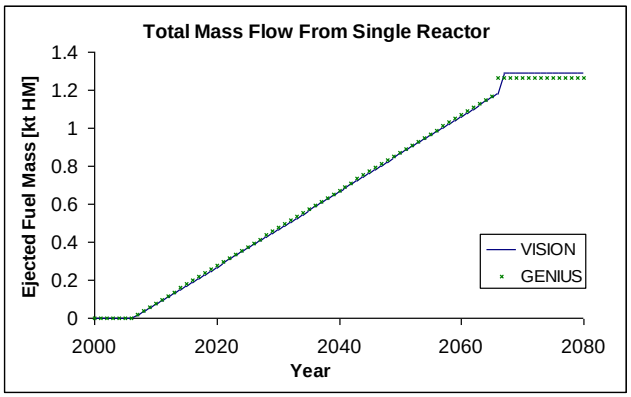


Fig. 4. Integrated total mass flow from the single reactor in Problem 1. Because GENIUS is a discrete-flow code, it handles reactor startup and decommissioning in a straightforward way.

This reasonable agreement extends to the isotopic level. Fig. 5 plots results for the discharge isotopics of the five largest actinide streams in both simulations. The end-of-simulation discrepancies are of the same magnitude as when we compare total masses; the differences in the GENIUS result with respect to the VISION results fall between 1.81% and 2.05% (see first column of Table III).

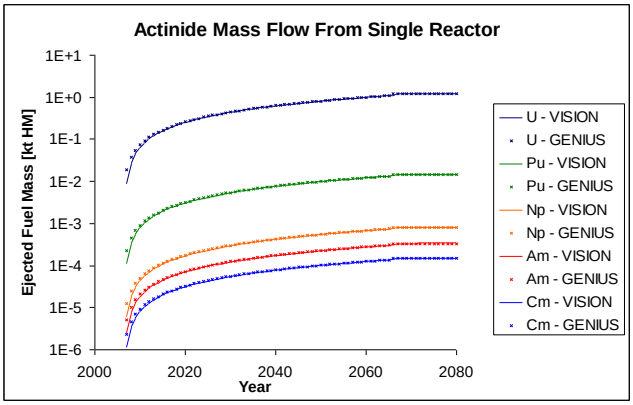


Fig. 5. Integrated mass flow from the single reactor in Problem 1, for the five largest actinide streams (note semi-log scale). The end-of-simulation errors in the GENIUS results with respect to the VISION results for each element are given in the first column of Table III.

To ensure that these results scale appropriately, for Problem 2 we reran the same test but with ten such reactors. Because these reactors were identical to the first one and all begin operating at the same time as in the single case, we expected and observed magnitudes exactly ten times greater than in the single reactor case and with the same end-of-simulation discrepancies.

In Problems 3 and 4, we complicate facility deployment by specifying growth curves. As mentioned earlier, GENIUSv2 requires a user-driven facility deployment in order to avoid using a deployment heuristic. Of course, calculating a reactor deployment to meet an arbitrary demand curve is a fairly trivial problem, so we have written the capability for doing so into the GENIUSv2 pre-processor. Thus, we choose for Problem 3 a stepwise-linear growth case where we start building reactors in 2000 and increase the total capacity by one reactor each year (requiring two new reactors per year starting in 2067 to account for one retirement per year during those final 34 years).

To save space, we forego plotting the straightforward (and identical) capacity curves for the VISION and GENIUS results for this first growth scenario and proceed directly to the mass flow results. These are shown in Fig. 6 (with isotopic breakdowns again in Table III), and they once again show good agreement. In fact, we see that the VISION-GENIUS discrepancy shrinks measurably and that this time GENIUS gives a larger result. Careful examination of Fig. 4 suggests an explanation. Because of the differences in how they handle startup and decommissioning, the time-integrated ejected mass values for GENIUS reactors are higher than for VISION during most of the reactor's lifetime (because VISION only ejects half a batch worth of fuel in the first year) but are lower than for VISION once decommissioning is complete (because VISION reactors consume more total lifetime fuel than their GENIUS counterparts, as we saw in Table II). Thus, we get some error-canceling. And because 94 of the 128 total reactors built during the simulation have not yet been decommissioned in 2100, the VISION fleet lags behind the GENIUS fleet in terms of fuel mass ejected so far, even though in the end each of the VISION reactors will have used slightly more fuel than the GENIUS reactors.

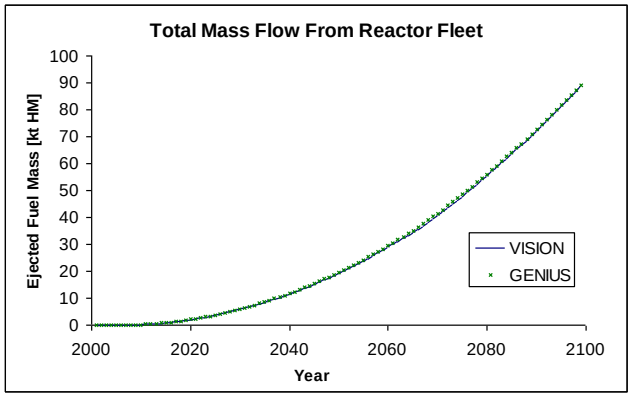


Fig. 6. Integrated total mass flow from the reactor fleet in Problem 3. The end-of-simulation errors in the GENIUS results with respect to the VISION results for total mass and the five main actinide streams are given in the third column of Table III.

Problem 4, the final VISION-GENIUS benchmark problem, is for exponential growth in electricity demand. We choose an initial demand of 10 GWe and a demand growth rate of 2% per year. We include ten "legacy"

reactors that exist when the simulation starts. They retire, one per year, starting in 2029.

The mass flow results for the VISION and GENIUS simulations of this deployment are given in Fig. 7. We see that GENIUS returns a lower total mass than VISION; as we noted in Problem 3, the higher the percentage of total reactors that reach decommissioning by the end of the simulation, the more likely VISION is to compute larger mass flows than GENIUS. In Problem 4, 31.7% of the reactors get decommissioned, as opposed to only 26.6% in the previous problem, so it's not very surprising that GENIUS returns to computing a lower total mass output than VISION. Finally, note that the isotopic discrepancies with respect to VISION case are given in the final column of Table III and once again show reasonable agreement at that level of detail as well.

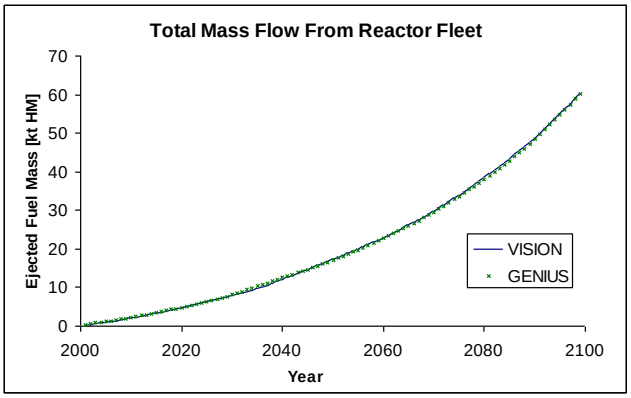


Fig. 7. Integrated total mass flow from the reactor fleet in Problem 4. The end-of-simulation errors in the GENIUS results with respect to the VISION results for total mass and the five main actinide streams are given in the fourth column of Table III.

TABLE III

Summary of Isotopics Results for Benchmark Problems

	VISION-GENIUS Discrepancy (GENIUS error with respect to VISION result)			
Material Stream	1	2	3	4
Total Mass	-2.00%	-2.00%	0.32%	-0.53%
Uranium	-1.93%	-1.93%	0.38%	-0.47%
Plutonium	-2.05%	-2.05%	0.27%	-0.58%
Neptunium	-1.89%	-1.89%	0.43%	-0.43%
Americium	-1.91%	-1.91%	0.40%	-0.45%
Curium	-1.81%	-1.81%	0.51%	-0.34%

IV.B. Large,Three-Region Once-Through Scenario

Finally, we present Problems 5 and 6, two once-through problems that demonstrate GENIUSv2's ability to be scaled up to larger scenarios and that point to the richness and flexibility of the R-I-F model and our formulations for solving the MRP.

The facility deployment for Problems 5 and 6 is given in Table IV. Both of the reactor regions contain three institutions: a small and a large fuel fabricator and reactor operator building either PWRs or PHWRs to match a linear demand curve. The third region contains only a large fabricator with facilities for both LWR and PHWR fuel. The parameters for both types of reactors are given in Table V. The precise fuel fabrication capacities were chosen to match reactor batch sizes in order to avoid the unrealistic scenario of splitting a fuel batch order between two different fabricators. Future work will explore optimization techniques to relax this constraint in a way consistent with our network flow model.

We note briefly that this is a fairly large problem. It calls for construction 748 total reactors (compared to 130 in Problem 3, the largest of our VISION benchmark problems) and records 43,521 individual material transfers (9,612 in Problem 3). We will report more extensively on GENIUSv2 performance data at a later time.

TABLE IV

Three-Region Problem – Facility Deployment

Region	Institution	Facilities
1	1	1 PWR in Jan. 1970 Linear growth: 1.71 GWe/year
	2	1 LWR Fuel Fab (78.66 tHM/month)
	3	1 LWR Fuel Fab (157.3 tHM/month)
2	4	1 PHWR in Jan. 1970 Linear growth: 675 MWe/year
	5	1 PHWR Fuel Fab (333.7 tHM/month)
	6	1 PHWR Fuel Fab (667.3 tHM/month)
3	7	1 LWR Fuel Fab (157.3 tHM/month) 1 PHWR Fuel Fab (667.3 tHM/month)

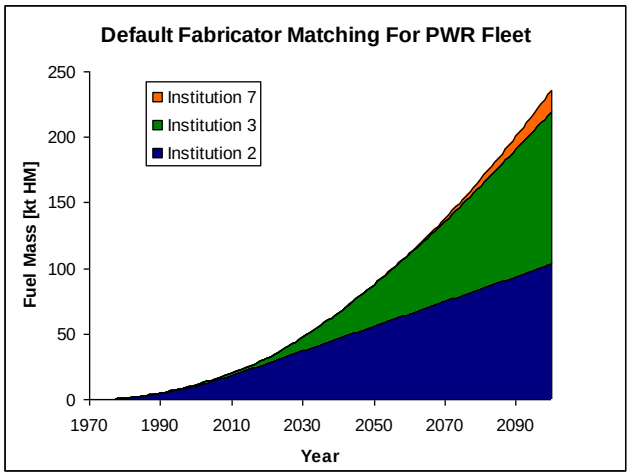
TABLE V

Three-Region Problem – Reactor Parameters

Parameter	Value	
	PWR	PHWR
Start Year	1970	
End Year	2099	
Construction + License Time	5 years	
Operating Time, OT	50 years	
Capacity Factor, CF	0.90	
Power Capacity, P [MWe]	1000	600
Thermal Efficiency, η	0.33	0.30
Cycle Time, T [months]	18	12
Fuel Burnup, Bu [GWd/tHM]	45	7
Fuel Batches per Core, N	4	1

In Problem 5, we allow the GENIUS matching algorithm to solve the MRP according to the default affinities described in *III.B. Customer-Supplier Matching*. In Problem 6, we alter the default behavior by specifying two rules: Institutions 1 and 4 (the reactor operators) get preferentially matched with Institution 7 (the extra-regional

fuel fabricator) as if they were all the same Institution. This affinity assignment could represent any number of modeling decisions, including to simulate a long-term contract between Institutions 1 and 7 and 4 and 7; to signify that all three are, in fact, owned by the same company; or to capture some price advantage benefiting Institution 7. Fig. 8 (PWRs) and Fig. 9 (PHWRs) show results for the cumulative travel of fabricated fuel from the various suppliers to the reactor fleets in the two different problems. In the top of each figure, we see the default behavior at work; the extra-regional fabricator is the supplier of last resort and is only purchased from consistently when the order density is high enough that the fabricators in the reactor regions are always working at capacity. Conversely, the bottom plot of each figure shows that the foreign fabricator is now preferred.



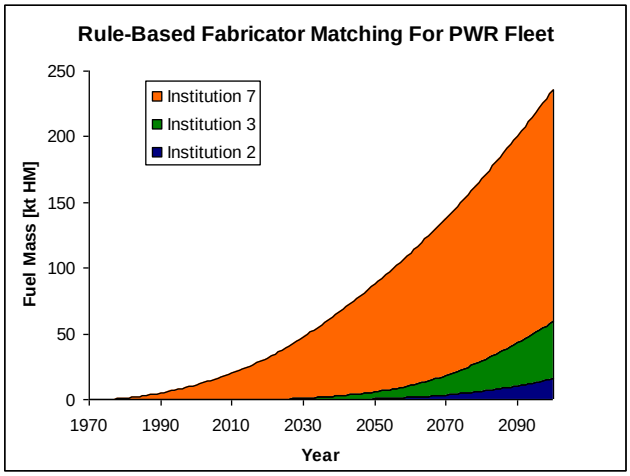
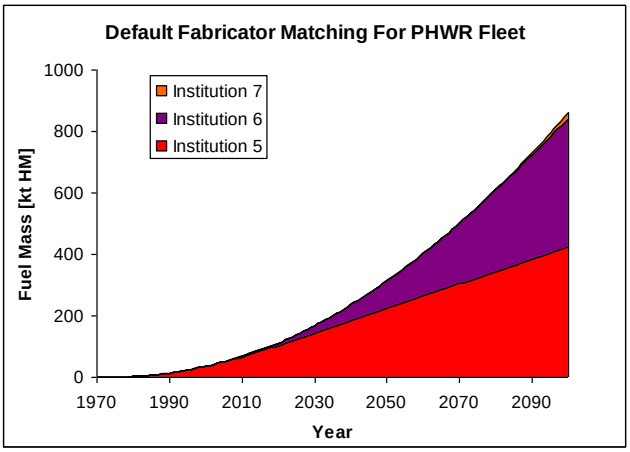


Fig. 8. Change in matching of PWR fuel fabricators to reactors placing orders. When the reactor operator's affinity for trade with Institution 7 is increased sufficiently, it becomes the favored supplier even though it's located in another region.



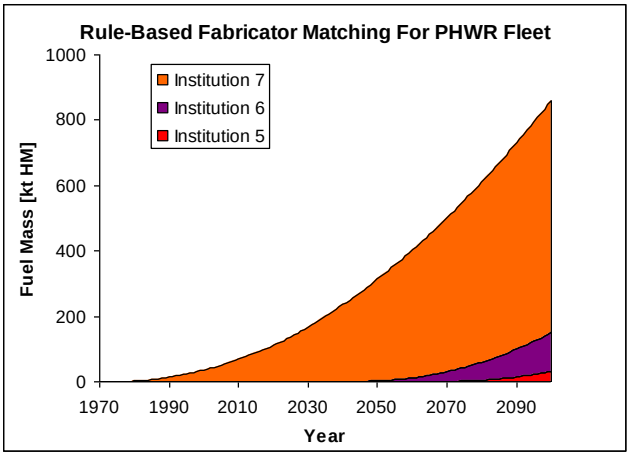


Fig. 9. Same as Fig. 8, except for the PHWR region's fleet. Note the substantially different material routing based on one change to the input file.

We expect this modeling capability to be especially useful for exploring the behavior of the global fuel cycle system under many proposed modes of international cooperation. In particular, R-I-F hierarchical matching under carefully chosen arc costs has the potential to help policy makers understand the effects of tools like trade agreements, tax incentives, long-term contracts, and other mechanisms.

V. CONCLUSIONS AND FUTURE WORK

We conclude from this work that GENIUSv2 is sufficiently detailed to closely approximate the kinds of nuclear fuel cycle scenarios modeled with codes like VISION, sufficiently robust to store and process the large amount of data that accumulate when modeling those scenarios discretely, and sufficiently flexible to model technical and socio-economic relationships between the various entities in a region-institution-facility hierarchy. The latter point is perhaps the most important, since those relationships are both key to improving our understanding

of next-generation fuel cycles and difficult to explore with most of the available tools.

Ongoing and immediate future work on GENIUSv2 includes rigorous testing and benchmarking of its packages for radioactive decay and spent-fuel reprocessing. Combined with the relatively stable and straightforward implementations of the front-end facilities that already exist in the code, these tools will complete GENIUSv2's ability to model closed nuclear fuel cycles. At that point, we hope the code will be useful for our group and other end users interested in studying such systems. In particular, we plan to use GENIUS to probe questions of fuel cycle robustness and to model various scenarios for an international fuel bank. Other planned applications include a study of repository isotopics and radiotoxicity. Our longer-term methodological focus is on continuing to develop techniques for system-wide optimization. In particular, we are interested in the potential for wrapper-based iterative techniques as well as agent-based decision modeling.

ACKNOWLEDGMENTS

The authors would like to thank the Advanced Fuel Cycle Initiative/Global Nuclear Energy Partnership University Fellowship Program for funding Oliver's work on GENIUS and Milad Fatenejad for his help with designing and implementing the GENIUSv2 debugging framework.

REFERENCES

1. K. O. PASAMEHMETOGLU and P. Finck, "SINEMA: Simulation Institute for Nuclear Energy Modeling and Analyses," *Trans. of the American Nuclear Society*, Vol. 95, p. 155-156, American Nuclear Society, La Grange Park, IL, United States (2006).

2. C. A. JUCHAU, M. L. Dunzik-Gougar and K. O. Pasamehmetoglu, "Simulation Institute for Nuclear Energy Modeling and Analyses (SINEMA): Developing a GENIUS," *Trans. of the American Nuclear Society*, Vol. 94, p. 78-79, American Nuclear Society, La Grange Park, IL, United States (2006).

3. C. A. JUCHAU, M. L. Dunzik-Gougar and K. O. Pasamehmetoglu, "Global Evaluation of Nuclear Infrastructure Utilization Scenarios: Initial code development," *Trans. of the American Nuclear Society*, Vol. 95, p. 164-165, American Nuclear Society, La Grange Park, IL, United States (2006).

4. C. A. JUCHAU and M. L. Dunzik-Gougar, "A Review of Nuclear Fuel Cycle Systems Codes," *SINEMA LDRD PROJECT*, Idaho National Laboratory and Idaho State University, Idaho Falls, ID, United States (2006).

5. M. L. DUNZIK-GOUGAR, C. A. Juchau, K. O. Pasamehmetoglu, P. P. H. Wilson, K. M. Oliver, P. J. Turinsky, et al., "Global Evaluation of Nuclear Infrastructure Utilization Scenarios (GENIUS)," *Proc. of GLOBAL 2007: Advanced Nuclear Fuel Cycles and Systems*, American Nuclear Society, La Grange Park, IL, United States (2007).

6. J. JACOBSON, A. M. Yacout, G. Matthern, S. Piet, D. Shropshire and C. Laws. "VISION: Verifiable Fuel Cycle Simulation Model," *Trans. of the American Nuclear Society*, Vol. 95, p. 157-159, American Nuclear Society, La Grange Park, IL, United States (2006).

7. J. F. AHEARN (chair), et al., *Internationalization of the Nuclear Fuel Cycle: Goals, Strategies, and Challenges*, National Academies Press, Washington, DC (2009).

8. P. P. H. Wilson and K. M. Oliver, "Designing GENIUS Version 2 to Model Inter-Facility Relationships [poster]," *Annual Review Meeting*, Advanced Fuel Cycle Initiative/Global Nuclear Energy Partnership, Litchfield Park, AZ, United States (2007).

9. R. JAIN and P. P. H. Wilson, "Transitioning To Global Optimization In Fuel Cycle System Study Tools," *Trans. of the American Nuclear Society*, Vol. 95, p. 162-163, American Nuclear Society, La Grange Park, IL, United States (2006).

10. M. S. ELDRED, B. M. Adams, et al., "DAKOTA, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis (Version 4.2+ User's Manual)," *Report No. SAND2006-6337*, Sandia National Laboratories, Albuquerque, NM, United States (2006).

11. D. P. BERTSEKAS, *Network Optimization: Continuous and Discrete Models*, Athena Scientific, Nashua, NH (1998).

12. D. SHROPSHIRE, et al., *Advanced Fuel Cycle Cost Basis*, *Report No. DE-AC07-05ID14517*, Idaho National Laboratory, Idaho Fall, ID, United States (2007).