Proceedings of the 2014 Winter Simulation Conference
A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, eds.

A TIME AND SPACE COMPLEXITY ANALYSIS OF MODEL INTEGRATION

Michael J. North

Argonne National Laboratory
9700 S. Cass Ave.
Argonne, IL, 60439 USA

ABSTRACT

The computational study of complex systems increasingly requires model integration. The drivers include a growing interest in leveraging accepted legacy models, an intensifying pressure to reduce development costs by reusing models, and expanding user requirements that are best met by combining different modeling methods. There have been many published successes including supporting theory, conceptual frameworks, software tools, and case studies. Nonetheless, on an empirical basis, the published work suggests that correctly specifying model integration strategies remains challenging. This naturally raises a question that has not yet been answered in the literature, namely 'what is the computational difficulty of model integration?' This paper's contribution is to address this question with a time and space complexity analysis that concludes that deep model integration with proven correctness is both NP-complete and PSPACE-complete and that reducing this complexity requires sacrificing correctness proofs in favor of guidance from both subject matter experts and modeling specialists.

1 INTRODUCTION

The computational study of complex systems increasingly requires the integration of multiple computational models (Vangheluwe, de Lara, and Mosterman 2002). Model integration is the construction of larger models by combining smaller models to address questions that cannot be answered by the smaller models in isolation (Villa 2001). The constituent models often use differing solution methods which then produce a hybrid or multi-paradigm integrated model. The *de novo* development of hybrid models can also be seen as a kind of model integration problem, albeit one focused on design integration rather than implementation integration.

Model integration has increasing appeal. The drivers include a growing interest in leveraging divergent sets of accepted legacy models, an intensifying pressure to reduce development costs by reusing existing models, and expanding user requirements that are best met by combining different modeling methods.

Of course, it is possible to connect multiple models in arbitrary ways or to misunderstand the way a given set of models has been integrated. These are potentially significant issues in practice. Nonetheless, the purpose of the paper is not to propose a solution to these problems. Furthermore, the techniques discussed in this paper are not necessarily being recommended to support applied model integration. Rather, this paper explores the minimal computational difficultly of fully documenting the integration of models and of proving the correctness of a model integration strategies. As such, this paper focuses on mathematically rigorous techniques which can be used to derive theoretical bounds on computational difficulty of model integration as opposed to techniques that are easy to apply in practice. Methods for automating the described model integration process, estimate the difficultly for specific models, or skill requirements for modelers the are outside this scope. Furthermore, the paper deals with models based on their requirements rather than their implementation paradigm (e.g., system dynamics, agent-based

modeling, optimization, or discrete event simulation). Thus the conclusions apply equally to all computational modeling techniques.

2 RELATED WORK

There have been many published model integration advances including supporting theory (Vangheluwe, de Lara, and Mosterman 2002), conceptual frameworks (Vangheluwe and de Lara 2003, Liang and Paredis 2003, Brown et al. 2005), software tools (IEEE 2010, IEEE 2012, Villa and Costanza 2000, Villa 2001, Vangheluwe and de Lara. 2003, North et al. 2006, North et al. 2007), and applied case studies (Dubiel and Tsimhoni 2005, Bithell and Brasington 2009, Teose 2011). Despite the documented successes, on an empirical basis, the published works suggest that correctly specifying model integration strategies remains challenging (Rizzoli et al. 2008). This naturally raises a question that has not yet been answered in the literature, namely 'what is the computational difficulty of model integration?' This paper's contribution is to address this question with a time and space complexity analysis of model integration.

3 THE MODEL INTEGRATION PROCESS

Model integration can be factored into a multistep process that begins with a set of candidate models to be integrated and ends with either a single combined model or an incompatibility proof. The combined model will consist of the candidate models joined by selected model integration points.

Model integration points are transient or persistent data structures that can be read by other models, written by other models, or both. Inputs and outputs such as files, databases, or web services are common integration points. Data structures in memory are also candidates if they can be safely accessed or modified during model execution. Integration points are discussed in greater detail later in this paper.

The steps and outcomes in the model integration process shown in Table 1 are as follows:

1. Check the set of candidate models for compatibility. The result is either a certification of compatibility or an incompatibility proof.
2. Group related model integration points in equivalence classes. The result is a set of model integration point clusters, each containing functionally equivalent points.
3. Sequence the model integration point clusters. The result is a model integration strategy showing how the models will execute and communicate.
4. Implement the model integration strategy using appropriate software. The result is an integrated, but untested, model.
5. Perform appropriate verification and validation on the integrated model relative to the questions to be answered. The result is an integrated model usable for the questions at hand.

Table 1: The model integration process.

Preconditions	Process	Postconditions	Addressed in this Paper?
A Combined Set of Assumptions for the Candidate Models	Model Compatibility Check	Either a Certification of Compatibility or an Incompatibility Proof	Yes
A Combined Set of Model Integration Points	Group Related Model Integration Points	A Set of Model Integration Point Clusters	Yes
A Set of Model Integration Point Clusters	Sequence the Model Integration Point Clusters	A Model Integration Strategy	Yes
A Model Integration Strategy	Implement the Model Integration Strategy	An Integrated, but Untested, Model	No
An Integrated, but Untested, Model	Perform Appropriate Verification and Validation Relative to the Questions to be Answered	The Result is an Integrated Model Usable for the Questions at Hand	No

4 ASSUMPTIONS

Several assumptions need to be made about the model integration processes to support the time and space analysis in the next section. These assumptions are considered in this section.

First, we seek to determine the minimum effort needed to integrate models at the desired level of functionality. Efforts that are avoidable *a priori* should not be included in the analysis.

Correct model integration is required. This eliminates provably incorrect integration efforts (e.g., connecting two models in implementable way that, nonetheless, violates their underlying design assumptions). Naturally, correctness is relative to the modeling question or questions to be answered.

The list of candidate integration points will be recorded in a form equivalent to predicate logic. The information will include a qualifier and any relevant preconditions and postconditions. The qualifier describes the integration point including the kind of data (e.g., length or mass), the units (e.g., meters or kilograms), the type of data (e.g., integer or real number), and other metadata as needed. The preconditions define what must be true before the value is accessed or changed (e.g., the simulation clock time must be greater than 12:00 noon). The postconditions define what becomes true after the integration point (e.g., the simulation clock time is 1:00 p.m.). Postconditions reprise any value in the preconditions or qualifiers that remain true. It is possible to have several integration points defined for a specific model data element. For example, a value that can be both read and written might have an integration point defined for each function.

This paper focuses on characterizing the difficulty of the overall process of model integration. The features of particular integration software (i.e., model integration step four) and domain-specific verification and validation requirements (i.e., step five) are important for real world model integration implementations, but are outside the scope of this analysis. As such this paper will analyze model integration steps one, two, and three. It is thus assumed that an integration strategy is the desired product. The goal of a strategy is deep model integration. This eliminates trivial integration efforts from the analysis (e.g., running two unconnected models side-by-side) and focuses attention on the central problem of combining multiple models.

A deep model integration strategy is a maximal-length self-consistent chain of model integration points. A chain of model integration points is an ordered list of ordered triplets. Each triplet includes a source integration point (e.g., for reading), a destination integration point (e.g., for writing), and the

relative time of the transfer. Cyclic or repeating events can also be represented in compact form. A chain is inconsistent if:

- The preconditions for a point later in the chain are not consistent with the accumulated postconditions (i.e., all of the postconditions that still remain true) for points earlier in the chain, or
- The qualifiers and postconditions for a source point are not consistent with the corresponding preconditions and qualifiers destination point.

Of course, maximal-length self-consistent chains are not necessarily unique. Finding any maximal-length self-consistent chain will suffice for this analysis.

Given a specific chain, each constituent model is free to execute until an input is needed from another model. This implicitly assumes pessimistic model execution scheduling in the event of concurrent model execution. It is also possible to assume optimistic scheduling with suitable rollback facilities. This does not change the time or space complexity analysis presented later. Simulation scheduling is discussed in more detail in Perumulla (2006).

It is assumed that each model is accompanied by a complete list of assumptions expressed in linear temporal logic (LTL) (Pnueli 1977). The version of LTL used in this paper requires the operators *eventually*, *invariantly*, *next-time*, *until*, and *since* as first described by Pnueli (1977).

Why LTL? Variations of LTL are widely used for model checking of parallel programs, particularly for deterministic systems (Baulanda 2009). Alternatives include computation tree logic for nondeterministic problems (Schnoebelen 2002). An inherently parallel approach was chosen since integrated models are highly likely to be at least partially concurrent. Few models come with sufficiently detailed documentation for model integration of any kind (Rizzoli et al. 2008, Müller et al. 2014), much less LTL. This does not limit the applicability of this paper since the goal here is to determine the time and space complexity of correct model integration independent of specific implementations.

Finally, it is assumed that the goal is proving the minimum time and space requirements for model integration under the other assumptions, not developing a practical model integration methodology. In other words, the model integration approach discussed in the paper is not necessarily being recommended as a model integration methodology. It is being used to prove computational bounds on all possible approaches that satisfy the other assumptions.

Building on the previous discussion, we can now formally define the model integration problem. For subscripts i and u indexing models; subscript j indexing assumptions for models; subscripts q and l indexing model integration points; and t representing model time, the model integration problem can be stated as follows:

> Starting with a set of M separate models, $m_i(A_i, P_i) \in M$, each with assumptions $a_{i,j} \in A_i$ in LTL form and integration points $p_{i,q}(b_{i,q}, q_{i,q}, e_{i,q}) \in P_i$ that have predicate logic preconditions $b_{i,q}$, qualifiers $q_{i,q}$, and postconditions $e_{i,q}$, find a maximal-length ordered chain, $C = \{(p_{i,q}, p_{u,l}, t) \mid i \neq u,\ m_i \in M, m_u \in M, t \in T\}$, of integration point triplets over model time schedule T or an empty chain if $\{A_i \cup A_u\}$ is inconsistent.

It should be noted that if separate instances of a single model are used repeatedly in an ordered chain then they are recorded as $m_i \in M$ with distinct indices i.

5 COMPLEXITY ANALYSIS

A time and a space complexity analysis of model integration steps one to three is presented in this section. Both analyses are based on the assumptions discussed in the previous section.

Checking the mutual compatibility of the candidate model set is a type of model checking using the superset of all model assumptions. Sistla and Clarke (1985) have proven that resolution of LTL with *eventually* has NP-complete time complexity and PSPACE-complete space complexity. Therefore model integration step one is both NP-complete and PSPACE-complete. Reducing these times will likely require domain-specific optimizations (Wolpert and Macready 1997).

The grouping of elements with attributes is a type of clustering (Estivill-Castro 2002, Xu and Wunsch 2005, Rai and Singh 2010). The preconditions, qualifiers, and postconditions are used as the attributes to be clustered. Elements with compatible preconditions, qualifiers, and postconditions form clusters. For set size N, the sum of Bell's number, B_n, between one and N gives the number of possible strict partition clustering assignments[1], D_n, as shown in Equations 1 to 3 (Rota 1964). Dobinski's formula, as shown in Equation 4, can also be used:

$$B_1 = 1 \tag{1}$$

$$D_1 = 1 \tag{2}$$

$$D_N = \sum_{i=1}^{n-1} \sum_{k=0}^{\infty} \binom{i}{k} B_k \quad \text{for } n \geq 2 \tag{3}$$

$$D_N = \frac{1}{e} \sum_{k=0}^{\infty} \frac{k^n}{k!} \quad \text{for } n \geq 1 \tag{4}$$

Many different algorithms are commonly used for clustering (Xu and Wunsch 2005, Rai and Singh 2010). These algorithms vary according to the criteria or inductive principles used to perform the clustering. Clustering algorithms have a wide range of time complexities from low order polynomial (Ward 1963) to NP-hard (Dasgupta 2008). Most practical clustering techniques run in cubic or faster time (MacQueen 1967, Sibson 1973, Defays 1977, Bock 2008) and use PSPACE space. Therefore, step two, model integration point grouping, can be taken to be in PTIME and PSPACE for the number of points. However, expert-driven domain-specific choices of clustering algorithms will need to be made to reduce the problem to PTIME and PSPACE.

Once model integration points are grouped, the groups need to be put into a relative sequence. Consistent with the goal of deep model integration, sequencing is essentially a single processor scheduling problem with precedence constraints. This inherently NP-complete problem can be approximately solved in PTIME and PSPACE (Agrawal, Klein, and Ravi 1991).

6 CONCLUSIONS

This paper has explored the minimum time and space needed for deep model integration with proven correctness. Combining results, we draw several conclusions[2]. First, model integration is NP-complete. Second, model integration is PSPACE-complete. Third, reducing model integration time and space complexity requires sacrificing correctness proofs in favor of expert guidance that is unlikely to be fully automatable. These three conclusions suggest that while model integration may continue to be supported by a wide range of software such as DIS (IEEE 2012) or HLA (IEEE 2010), the essential task of

[1] Strict partitioning is also sometimes called hard partitioning. Alternatives that allow missing or multiple assignments are sometimes called soft partitions. Model integration requires hard partitioning so all integration points are accounted for, even if they are not ultimately used for integration.

[2] It should be noted that these conclusions do not obviate the potential value of integration frameworks such as DIS (IEEE 2012) or HLA (IEEE 2010). Frameworks can productively automate a wide range of supporting functions without necessarily reducing the overall theoretical time or space complexity of the model integration process.

determining how to connect diverse models will remain a largely manual activity that requires the services of both subject matter experts and modeling specialists.

ACKNOWLEDGMENTS

The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory ("Argonne"). Argonne, a U.S. Department of Energy Office of Science Laboratory, is operated under contract number DE-AC02-06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the government. Any opinions, findings, conclusions, or recommendations are solely those of the author.

REFERENCES

Agrawal, A., P. Klein, and R. Ravi. 1991. "Ordering Problems Approximated: Register Sufficiency, Single-Processor Scheduling, and Interval Graph Completion." *Lecture Notes in Computer Science*. Springer. 510: 751–762.

Bauland, M., M. Mundhenkb, T. Schneiderc, H. Schnoord, I. Schnoord, and H. Vollmere. March 2009. "The Tractability of Model-checking for LTL: The Good, the Bad, and the Ugly Fragments." *Proceedings of the 5th Workshop on Methods for Modalities*. Elsevier Electronic Notes in Theoretical Computer Science. 231 (25): 277–292.

Bithell, M. and J. Brasington. 2009. "Coupling Agent-based Models of Subsistence Farming with Individual-based Forest Models and Dynamic Models of Water Distribution." *Environmental Modelling & Software*. 24: 173–190.

Bock, H. H. December 2008. "Origins and Extensions of the K-means Algorithm in Cluster Analysis." *Electronic Journal for History of Probability and Statistics*. 4 (2): 1–18.

Brown, D. G., R. Riolo, D. T. Robinson, M. North, and W. Rand. 2005. "Spatial process and data Models: Toward Integration of Agent-based Models and GIS." *Journal of Geographical Systems*. 7: 25–47.

Defays, D. 1977. "An Efficient Algorithm for a Complete Link Method." *The Computer Journal of the British Computer Society*. 20 (4): 364–366.

Dasgupta, S. 2008. *The Hardness of K-means Clustering*. Technical Report CS2008-0916. Department of Computer Science and Engineering. University of California, San Diego, CA, USA.

Dubiel, B. and O. Tsimhoni. 2005. "Integrating Agent Based Modeling into a Discrete Event Simulation." In *Proceedings of the 2005 Winter Simulation Conference* edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines. 1029–1037. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Estivill-Castro, V. June 2002. "Why So Many Clustering Algorithms — A Position Paper." *Association for Computing Machinery's Special Interest Group on Knowledge Discovery and Data Mining Explorations Newsletter*. 4(1): 65–75.

Grimmer, J. and G. King. 2009. "Quantitative Discovery from Qualitative Information: A General-Purpose Document Clustering Methodology." *American Psychological Association 2009 Toronto Meeting Paper*. 1–43.

Hagerup, T., and M Maas. 1994. "Generalized Topological Sorting in Linear Time." *Nordic Journal of Computing*. 1: 38–49.

IEEE. 2010. *1516TM-2010 IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Framework and Rules*.

IEEE. 2012. *1278.1-2012 IEEE Standard for Distributed Interactive Simulation – Application Protocols*.

Kahn, A. B. 1962. "Topological Sorting of Large Networks." *Communications of the Association for Computing Machinery*. 5 (11): 558–562.

Liang, V. and C. J. J. Paredis. 2003. "A Port Ontology for Automated Model Composition." *Proceedings of the 2003 Winter Simulation Conference*, edited by S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice. 613–622. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

MacQueen, J. B. 1967. "Some Methods for Classification and Analysis of Multivariate Observations." *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*. University of California Press, Ewing, NJ, USA.

Müller, B., S. Balbi, C. M. Buchmann, L. de Sousa, G. Dressler, J. Groeneveld, C. J. Klassert, Q. Bao Le, J. D. A. Millington, H. Nolzen, D. C. Parker, J. G. Polhill, M. Schlüter, J. Schulze, N. Schwarz, Z. Sun, P. Taillandier, and H. Weise. 2014. "Standardised and Transparent Model Descriptions for Agent-based Models: Current Status and Prospects." *Environmental Modelling & Software*. 55: 156–163

North, M. J., P. Sydelko, J. R. Vos, T. R. Howe, and N. T. Collier. October 2006. "Legacy Model Integration with Repast Simphony," *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects* edited by D. L. Sallach, C. M. Macal, and M. J. North. Argonne National Laboratory, Argonne, IL USA. 95–106.

North, M. J., T. R. Howe, N. T. Collier, and J. R. Vos. 2007. "A Declarative Model Assembly Infrastructure for Verification and Validation." In *Advancing Social Simulation: The First World Congress* edited by S. Takahashi, D.L. Sallach and J. Rouchier. Springer, Heidelberg, FRG. 129–140.

Perumulla K. S. 2006. "Parallel and Distributed Simulation: Traditional Techniques and Recent Advances." In *Proceedings of the 2006 Winter Simulation Conference* edited by L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto. 84–95.Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Pnueli, A., October 1977. "The Temporal Logic of Programs." In *18th Annual Symposium on the Foundations of Computer Science*. 46–57.

Ria, P., and S. Singh. October 2010. "A Survey of Clustering Techniques." *International Journal of Computer Applications*, 7(12): 1–5.

Rizzoli A. E., M. Donatelli, I. N. Athanasiadis, F. Villa, and D. Huber. 2008. "Semantic Links in Integrated Modelling Frameworks." *Mathematics and Computers in Simulation*. 78: 412–423.

Rota, G. C. May 1964. "The Number of Partitions of a Set." *American Mathematical Monthly*. 71(5): 498–504.

Sibson, R. 1973. "SLINK: An Optimally Efficient Algorithm for the Single-link Cluster Method," *The Computer Journal of the British Computer Society*, 16 (1): 30–34.

Sistla, A. P., and E. M. Clarke. July 1985. "The Complexity of Propositional Linear Temporal Logics." *Journal of the Association for Computing Machinery*. 32 (3)733–749.

Schnoebelen, P. 2002. "The Complexity of Temporal Logic Model Checking." *Advances in Modal Logic* World Scientific Publishing Co. Pte. Ltd. 4: 1–44.

Teose, M., Z. Lu, K. Ahmadizadeh, S. P. Ellner, E. O'Mahony, C. Gomes, R. L. Smith, and Y. Grohn. 2011. "Embedding System Dynamics in Agent Based Models for Complex Adaptive Systems." In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*. 2531–2538.

Vangheluwe, H., J. de Lara and P. J. Mosterman. April 2002. "An Introduction to Multi-paradigm Modelling and Simulation." In *Proceedings of the AIS 2002 Conference*, edited by F. Barros and N. Giambiasi. Lisboa, Portugal. 9–20.

Vangheluwe, H. and J. de Lara. 2003. "Computer Automated Multi-Paradigm Modelling: Meta-Modelling and Graph Transformation." In *Proceedings of the 2003 Winter Simulation Conference*, edited by S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice. 595–603. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Villa, F. 2001. "Integrating Modelling Architecture: A Declarative Framework for Multi-Paradigm, Multi-Scale Ecological Modeling." *Ecological Modelling*. 137: 23–42.

Villa, F. and R. Costanza. 2000. "Design of Multi-Paradigm Integrating Modelling Tools for Ecological Research." *Environmental Modelling & Software.* 15: 169–177.

Ward, J. H., Jr. 1963. "Hierarchical Grouping to Optimize an Objective Function." *Journal of the American Statistical Association*. 58: 236–244.

Wolpert, D. H. and W. G. Macready. 1997. "No Free Lunch Theorems for Optimization." *IEEE Transactions On Evolutionary Computation*. 1 (1): 67–82.

Xu, R. and D. Wunsch. 2005. "Survey of Clustering Algorithms." *IEEE Transactions on Neural Networks*. 16(3): 645–678.

AUTHOR BIOGRAPHY

MICHAEL J. NORTH, MBA, Ph.D. is the Deputy Director of the Center for Complex Adaptive Agent Systems Simulation within the Decision and Information Sciences Division of Argonne National Laboratory. He is also a Senior Fellow in the joint Computation Institute of the University of Chicago and Argonne. Dr. North has over 20 years of experience developing and applying models for industry, government, and academia. Dr. North has published two books, five conference proceedings, one journal special issue, eight book chapters, 20 journal articles, four invited encyclopedia entries, and over 70 conference papers. Dr. North is also the lead developer of free and open source Repast agent-based modeling suite. His email address is north@anl.gov.