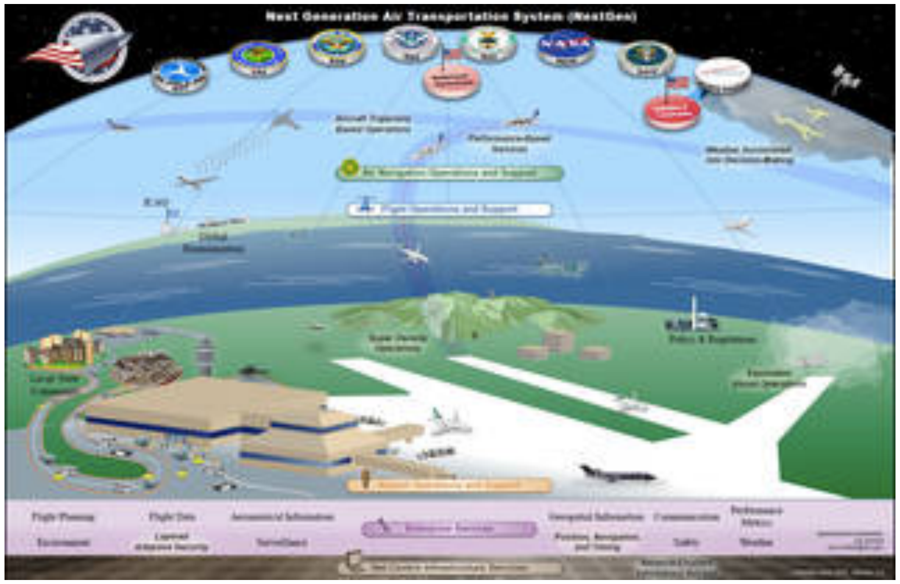
NASA INNOVATIVE ADVANCED CONCEPTS (NIAC): NNH15ZOA0001N. PHASE I

Verifying Safety of NextGen Models: A Rational Approach



WHY

- Air spaces gets more crowded
- Safety concerns become more critical.
- Need better reasoning about how pilots navigate crowded skies.
- Principle of rationality: experts know that only a few actions are most relevant
- Our hypothesis:
  (1) these few choices mark out a small space;
  (2) in that small space, formal verification tools can thoroughly explore a model

WHAT

JPF= Java Pathfinder = model exploration tool.
GALE= optimizers=like a smart pilot controlling a craft.

- Step1: Run GALE to find rational decisions.
- Step2: Run JPF for a few steps, constrained to regions found by GALE.
- Step3: Unleash JPF on regions accepted by Step2.

If JPF finds problems, ask GALE for mitigations:

- Step 4: run GALE with the constraint that it should strive to avoid JPF's counter-example states.

HOW

Case studies using recent work at NASA Ames models: modeling of pilots in the Brahms multi-agent framework for NAS and the NextGen Air Transportation Systems

WHO

PI Prof Tim Menzies CS NC State, tjmenzie@ncsu.edu
Dr. Neha Rungta, NASA Ames, neha.s.rungta@nasa.gov

Menzies = co-author of the GALE heuristic optimizer.
Rungta = experienced modeler of pilot systems at NASA

Verifying Safety of NextGen Models: A Rational Approach

Tim Menzies (PI), Computer Science, NC State, USA tim.menzies@gmail.com
Neha Rungta, SGT Inc/NASA Ames, neharungta@gmail.com

ABSTRACT: The current National Airspace (NAS) is quite safe and accidents are at a record low. Over the past few years, the US has embarked in a transformation of the Air Transportation System (ATS) to address the expected increase of air traffic in the US. The goal is to increase efficiency without compromising safety. There is a critical need for new safety evaluation techniques to verify and validate the interactions of humans and autonomy in the context of new complex software systems and changing roles of the human operator. The current state of the art to assess interactions of humans and autonomy is to model the air transportation system as a distributed, interactive system of systems with authority and autonomy assigned to both humans and automation at multiple levels. The scale and complexity of these models make the use of traditional verification and validation techniques intractable.

Hence, we propose **exciting and unexplored** research to develop a **new concept** that combines heuristic optimizers and formal verification tools in order to better understand huge models. In this approach, optimizers quickly select the smaller, most interesting regions of the model. While previous work has explored these two approach separately, **no previous work has studied their combination**. Creating this combination is **much more than a mere tool or process project**. Rather, it requires **fundamentally new ideas** for utilizing a variety of signals collected from complex systems. We propose to use the optimizers to find guides that can assist model-checking based verification when exploring these huge models. Our approach is based on the *principle of rationality* [16] which states that experienced operators of any system know that certain actions are most relevant to important outcomes. That is, within the very large decision space of huge models, there exists a smaller space of relevant and rational actions. Our hypothesis (to be tested in this work) is in this smaller space, it is possible that formal verification can thoroughly explore a model.

THE DOMAIN: Our work is **based on a current NASA aerospace architecture**. NASA and the FAA agree that the US need to modernize the ATS and to implement NextGen (Next Generation Air Transportation System) to accommodate the traffic increase over the next 15 years; Europe is going through a similar effort with SESAR. NextGen needs to provide at least the same, if not a better, level of safety. The NextGen IWP has a requirement (R-1440) that calls for new and improved V&V techniques for complex systems.

Recent work at NASA Ames models several scenarios in the Brahms multi-agent framework [5, 24]. Some examples include, Brahms models for conditions that led to the Air France 447 accident [7], sequence of events that led to the Überlingen collision [21], and performed a comparative evaluation of operator workload between a single pilot operations concept versus a traditional two pilot operation [27]. More recently, Brahms models have been developed for arrivals and departures at the La Guardia airport based on the work on Departure Sensitive Arrival Scheduling (DSAS) concept developed by the Airspace Operations Lab (AOL) at NASA Ames [13]. Through simulations of these models the interactions between pilots, controllers, and automation several issues regarding safety (Air France 447 and Überlingen crashes); performance (DSAS concept), and human operator workload (single pilot operation) have been discovered, characterized, and fixed.

THE PROBLEM: ACT-R, SOAR, and other architectures provide the ability to model actual low-level cognitive processes [1,11,12,14,15]. However, such modeling languages cannot describe complex systems and interactions at a higher-level of abstraction, or the decision making process of the human [20]. Other modeling tools, like the Brahms tool used at NASA Ames [4,21,24–26] can build more extensive models that know about higher-level human cognitive reasoning. Hence, they are the tool of choice at Ames for modeling air traffic management (ATM), NAS operations, and even manned space flight scearios.

The problem with models is that even at a higher-level of abstraction, they can contain fairly extensive interactions as well as a large number of rational agents including their beliefs, goals, and actions. Hence, when verifying them, it is often required to severely constrain the analysis by either creating simpler models or bounding the scope of the verification. There is a pressing need to scale automated verification tools to meet the demands of NextGen. Decades of research at NASA has lead to rigorous and formal verification methods for model-based reasoning– all of which suffer from problems of scale.

OUR PROPOSED SOLUTION: To make formal verification tools practical for large models, we use the principle of rationality to find the relevant parts of a model. Our proposal is to use the GALE *heuristic optimizers* [9] to "run ahead" of the formal verification tools to "shout back" the most important parts of the model to explore. GALE mimics the actions of a smart human operator of a model that finds the most useful regions within a model. GALE is an innovative combination of data mining and optimization algorithms:

- The data mining divides the data into tiny regions with *best* and *worst* ends.
- The optimizer builds new solutions by mutating the solutions in a region towards the *best* end.

GALE inputs a random selection of possible model inputs. These inputs are clustered to reduce n inputs into $\sqrt{n}$ clusters. These are then sub-divided into a tree of clusters and sub-clusters and sub-sub-sub-clusters, etc. Next, GALE walks down that tree performing *binary culling*; i.e. prune away sub-trees that contain the undesired model outputs (for the NASA models, undesired outputs include: conditions that lead to safety violations; or dangerous spikes in operator workload; or undesired large deviations in performance; e.g. flight delays).

When GALE finishes, it returns just the remaining tiny regions containing the most promising inputs. From these regions, we can extract four sets of input decisions that:

- *Never* appear in the promising regions;
- Appear at the *Same* frequency at the *best*and *worst* ends;
- Are *More* frequent at the best end than the worst end;
- Are *Less* frequent at the best end than the rowst end.

These tiny ranges, the above four sets, could be used to enable better model verification using automatic verification tools. Firstly, the restrictions on the model space mean that the verification tools need to only search a small percentage of the total model space. Secondly, the above four sets can be used to heuristically order the internal execution of the verification.

In this work we propose to use the Java Pathfinder (JPF) [28] framework to perform GALE-directed verification of Brahms models. JPF is an extensible Java Virtual Machine (JVM) for developing and exploring different verification techniques and application domains. In previous work we have developed an extension to JPF that allows us to systematically explore all possible behaviors of Brahms models [**?**]. JPF uses a generic model of the program state space consisting of *States*, *Choices* and *Transitions*. *States* are restorable snapshots of a program execution along a particular path. We leverage the choice generation mechanism within JPF to interact with the Brahms simulator and generate all possible choices at points of non-determinism within a Brahms model. This is an important advantage afforded by JPF, most model checkers do not provide user-configurable options to manage domain-specific types of non-determinism.

In this work we propose to use the output generated by GALE to direct the exploration within JPF. Specifically, we propose use constraints generated by GALE to perform a targeted exploration of the space using JPF. We will develop techniques that allow us to ensure:

- Any exploration that reaches a *Never* can be instantly curtailed;
- Similarly, any exploration that changes a *Some* decision can also be instantly curtailed;
- The remaining bounded explorations can be sorted according to how much they use *More* and how little they use *Less*.

Once sorted, the verification tools can then be unleased to explore deeper into the model but only on this smaller, most relevant portion of the model.

TECHNICAL DETAILS: The goal is to use the output of GALE to battle the state space explosion problem when using JPF for the verification of Brahms models. When JPF analyses a Brahms model along with its semantics, a new state is generated when the search within JPF encounters a point of non-determinism that has not yet been explored. There are three main points of non-determinism in the Brahms semantics (1) probabilistic updates to facts and beliefs of agents, (2) a range of activity durations, and (3) picking the next task to perform. In large models this leads even with a controlled exploration of choices, the state space explosion is a problem. With GALE we hope to overcome this by exploring relevant choices rather a brute force exploration. We will use constraints from GALE to select states (the beliefs and facts of agents and objects) and choices (relevant decisions). This we believe will allow us to mitigate the state space explosion problem and yet be able to characterize the safety, efficiency, and other properties of the model.

We recommend GALE for this process since it runs very fast. In prior experiments with large NASA models [10], GALE found optimizations comparable to more standard methods, but hundreds of times faster; i.e. four minutes, not 7.5 hours.

To achieve its fast runtimes, GALE uses a top-down *binary spectral clustering* algorithm [8] that recursively divides the state space along their eigenvectors. Standard top-down spectral learners require $O(n^2)$ calculations at each level of the recursion to find the eigenvectors [3]. A faster method, called uses Faloutsos' heuristic [6] to approximate eigenvectors in $O(n)$ time [19]. GALE uses these eigenvectors for a top-down binary clusterer. At each level, GALE projects all vectors onto a line connecting two distant vectors called P, Q (generating this line takes $O(n)$ time[1]). The line is then divided in half and GALE recurses on each half. GALE is very fast since it uses the above clustering to find and cull much of the search space. When dividing the data and evaluating P, Q, if P produces better model performance scores than Q, then GALE culls the worst half of the data (this approach cuts back n examples to $\sqrt{n}$ using just $2 * log_2\sqrt{n}$ evaluations). For the final leaf cluster, GALE says that the *direction of improvement* us towards the pole that generates better performance scores. GALE can then build n new vectors by mutate all the vectors in that leaf along the direction of improvement. These mutants become generation $i + 1$ that can be clustered and then mutated again to form generation $i + 2$, etc. Note that GALE make no assumptions of continuous or discrete-space vectors– it runs on either using a distance predicate that can comment on the the similarity of pairs of input states.

SUCCESS CRITERIA: To test the value of this approach, we only need to run the verification tools with and without GALE. This work would be a success if, for very large models, GALE+verification finds errors missed by just formal verification.

WHY THIS TEAM? Our proposed approach is **credible and reasonable** as it is based on mature research tools (JPF and GALE) being used by researchers with extensive experience in these specific tools, and with NASA systems. Dr. Rungta is an internationally recognized expert in methods for applying automatic verification methods to models [2, 7, 17, 21–23]. She is also skilled in "mixed methods" that combine multiple techniques (such as the GALE+JPF approach proposed here) [18]. Also, Dr. Menzies is one of the co-authors of the original GALE paper [9]. He also supervises a research lab of eight Ph.D. students, exploring GALE and its extensions.

Note that much of the case study material required for this work is readily available. For the last 12 months, Dr. Rungta has been conducting extensive studies with BRAHMS models. Hence, she is well aware of current concerns at NASA.

[1]Faloutsos heuristic: Pick any vector O at random. Let one pole P by the vector farthest from O. Let the other pole Q be the vector farthest from P. Note that finding P, Q requires only $2n$ distance calculations across n vectors.

References

[1] John R Anderson, Michael Matessa, and Christian Lebiere. ACT-R: A theory of higher level cognition and its relation to visual attention. *Human-Computer Interaction*, 12(4):439–462, 1997.

[2] John Backes, Suzette Person, Neha Rungta, and Oksana Tkachuk. Regression verification using impact summaries. In *SPIN '13*, 2013.

[3] Daniel Boley. Principal direction divisive partitioning. *Data Min. Knowl. Discov.*, 2(4):325–344, December 1998.

[4] William J. Clancey, Patricia Sachs, Maarten Sierhuis, and Ron van Hoof. Brahms: Simulating practice for work systems design. *Int. J. Hum.-Comput. Stud.*, 49(6):831–865, December 1998.

[5] W.J. Clancey, P. Sachs, M. Sierhuis, and R. Van Hoof. Brahms: Simulating practice for work systems design. *International Journal of Human-Computer Studies*, 49(6):831–865, 1998.

[6] Christos Faloutsos and King-Ip Lin. FastMap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163–174, 1995.

[7] J. Hunter, F. Raimondi, N. Rungta, and R. Stocker. A synergistic and extensible framework for multi-agent system verification. In Maria L. Gini, Onn Shehory, Takayuki Ito, and Catholijn M. Jonker, editors, *AAMAS*, pages 869–876. IFAAMAS, 2013.

[8] Sepandar Kamvar, Dan Klein, and Christopher Manning. Spectral learning. In *IJCAI'03*, pages 561–566, 2003.

[9] Joe Krall, Tim Menzies, and Misty Davies. Gale: Geometric active learning for search-based software engineering. *IEEE Transactions on Software Engineering*, pages to–appear, 2015.

[10] Joseph Krall, Tim Menzies, and Misty Davies. Better model-based analysis of human factors for safe aircraft approach. *IEEE Transactions on Human-Machine Systems, accepted, to appear*.

[11] John E Laird, Allen Newell, and Paul S Rosenbloom. Soar: An architecture for general intelligence. *Artificial intelligence*, 33(1):1–64, 1987.

[12] C. Lebiere, F. Jentsch, and S. Ososky. Cognitive models of decision making processes for human-robot interaction. In *Virtual Augmented and Mixed Reality. Designing and Developing Augmented and Virtual Environments*, pages 285–294. Springer, 2013.

[13] Paul U Lee, Nancy M Smith, Jeffrey Homola, Connie Brasil, Nathan Buckley, Chris Cabrall, Eric Chevalley, Bonny Parke, and Hyo-Sang Yoo. Reducing departure delays at laguardia airport with departure-sensitive arrival spacing (dsas) operations. In *Eleventh USA/Europe Air Traffic Management Research and Development Seminar (ATM)*, 2015.

[14] Yili Liu. QN-ACES: Integrating queueing network and ACT-R, CAPS, EPIC, and Soar architectures for multitask cognitive modeling. *International Journal of Human-Computer Interaction*, 25(6):554–581, 2009.

[15] Mikael Lundin. Simulating the effects of mental workload on performance in tankcrew. Technical Report SE-172 90, Swedish Defense Research Institute (FOI), 2004.

[16] Allen Newell. Readings from the ai magazine. chapter The Knowledge Level, pages 357–377. American Association for Artificial Intelligence, Menlo Park, CA, USA, 1988.

[17] Corina S Păsăreanu and Neha Rungta. Symbolic Pathfinder: symbolic execution of Java bytecode. In *Proceedings of the IEEE/ACM international conference on Automated software engineering*, pages 179–180. ACM, 2010.

[18] Corina S Păsăreanu, Neha Rungta, and Willem Visser. Symbolic execution with mixed concrete-symbolic solving. In *Proceedings of the 2011 International Symposium on Software Testing and Analysis*, pages 34–44. ACM, 2011.

[19] John C. Platt. FastMap, MetricMap, and Landmark MDS are all Nyström algorithms. In *In Proceedings of 10th International Workshop on Artificial Intelligence and Statistics*, pages 261–268, 2005.

[20] A. R. Pritchett and K. M. Feigh. Simulating first-principles models of situated human performance. In *Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2011 IEEE First International Multi-Disciplinary Conference on*, pages 144–151. IEEE, 2011.

[21] Neha Rungta, Guillaume Brat, William J. Clancey, Charlotte Linde, Franco Raimondi, Chin Seah, and Michael Shafto. Aviation safety: Modeling and analyzing complex interactions between humans and automated systems. In *Proceedings of the 3rd International Conference on Application and Theory of Automation in Command and Control Systems*, ATACCS '13, pages 27–37, New York, NY, USA, 2013. ACM.

[22] Neha Rungta, Eric G Mercer, and Willem Visser. Efficient testing of concurrent programs with abstraction-guided symbolic execution. In *Model Checking Software*, pages 174–191. Springer Berlin Heidelberg, 2009.

[23] Neha Rungta, Suzette Person, and Joshua Branchaud. A change impact analysis to characterize evolving program behaviors. In *Software Maintenance (ICSM), 2012 28th IEEE International Conference on*, pages 109–118. IEEE, 2012.

[24] M. Sierhuis. *Modeling and Simulating Work Practice. BRAHMS: a multiagent modeling and simulation language for work system analysis and design*. PhD thesis, Social Science and Informatics (SWI), University of Amsterdam, SIKS Dissertation Series No. 2001-10, Amsterdam, The Netherlands, 2001.

[25] R. Stocker, L. Dennis, C. Dixon, and M. Fisher. Verification of brahms human-robot teamwork models. In *Proceedings of 13th European Conference on Logics in Artificial Intelligence*, JELIA'12, 2012.

[26] R. Stocker, M. Sierhuis, L. Dennis, C. Dixon, and M. Fisher. A formal semantics for brahms. In *Proceedings of the 12th international conference on Computational logic in multi-agent systems*, CLIMA'11, pages 259–274, Berlin, Heidelberg, 2011. Springer-Verlag.

[27] Richard Stocker, Neha Rungta, Eric Mercer, Franco Raimondi, Jon Holbrook, Colleen Cardoza, and Michael Goodrich. An approach to quantify workload in a system of agents. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '15, pages 1041–1050, Richland, SC, 2015. International Foundation for Autonomous Agents and Multiagent Systems.

[28] W. Visser, K. Havelund, G. P. Brat, S. Park, and F. Lerda. Model checking programs. *Automated Software Engineering*, 10(2):203–232, 2003.