A Data-Parallel Monte Carlo Framework for Large-Scale PRA using Probabilistic Circuits

Preliminary Exam

Arjun Earthperson

Department of Nuclear Engineering
North Carolina State University

March 6, 2025

Education

- **MS, Nuclear Engineering**, North Carolina State University (2023)
  Thesis: ”Integrating Dual Error Propagation into Dynamic Event Trees to Support Fission Battery Probabilistic Risk Assessments”
- **BS, Electrical Engineering**, University of California, Los Angeles (2017)
  Capstone: ”Integer Hardware Optimizations on TI-C6000 DSPs for Low-Power IoT Applications”

Work Experience

- **Intern**, Idaho National Laboratory (Summer 2021)
  Project: Coupled OpenPRA’s OpenEPL engine with EMRALD.
- **Programmer**, The B. John Garrick Institute for the Risk Sciences, UCLA (2018–2020)
  Developed the Hybrid Causal Logic and Phoenix human reliability assessment web applications.

Awards

- **1st Place Graduate Student Winner**, ASME SERAD Student Safety Innovation Challenge (2023)
  Paper: Introducing OpenPRA: A Web-Based Framework for Collaborative Probabilistic Risk Assessment

Software

[1] *openpra-org/canopy*. Probability Estimation using Data-Parallel Monte Carlo, forked from SCRAM. URL: `https://github.com/openpra-org/canopy`.

[2] *openpra-org/inverse-canopy*. Tensorflow-based library for learning PRA model parameters. URL: `https://github.com/openpra-org/inverse-canopy`.

[3] *openpra-org/monorepo*. End-to-end PRA modeling and quantifcation solution. URL: `https://github.com/openpra-org/openpra-monorepo`.

[4] *openpra-org/openepl-engine*. Probablistic model checking on Dual-Error Propagation Models. URL: `https://github.com/openpra-org/openepl-engine`.

[5] *openpra-org/PRAcciolini*. PRA model meta-conversion utility. URL: `https://github.com/openpra-org/PRAcciolini`.

Journal Articles

[1] **Arjun Earthperson**, Mihai A. Diaconeasa. "Integrating Commercial-Off-The-Shelf Components into Radiation-Hardened Drone Designs for Nuclear-Contaminated Search and Rescue Missions". en. In: *Drones* 7.8 (2023). Number: 8 Publisher: Multidisciplinary Digital Publishing Institute, p. 528. ISSN: 2504-446X. DOI: `10.3390/drones7080528`. URL: `https://www.mdpi.com/2504-446X/7/8/528` (visited on 12/03/2024).

[2] **Arjun Earthperson**, Courtney M. Otani, Daniel Nevius, Steven R. Prescott, Mihai A. Diaconeasa. "A combined strategy for dynamic probabilistic risk assessment of fission battery designs using EMRALD and DEPM". en. In: *Progress in Nuclear Energy* 160 (2023), p. 104673. ISSN: 0149-1970. DOI: `10.1016/j.pnucene.2023.104673`. URL: `https://www.sciencedirect.com/science/article/pii/S0149197023001087` (visited on 03/30/2023).

[3] Elaheh Rabiei, Lixian Huang, Hao-Yu Chien, **Arjun Earthperson**, Mihai A Diaconeasa, Jason Woo, Subramanian Iyer, Mark White, Ali Mosleh. "Method and software platform for electronic COTS parts reliability estimation in space applications". en. In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* (2021), p. 1748006X2199823. ISSN: 1748-006X, 1748-0078. DOI: `10.1177/1748006X21998231`. URL: `http://journals.sagepub.com/doi/10.1177/1748006X21998231` (visited on 03/25/2021).

Conference Papers

[1] Egemen Aras, **Arjun Earthperson**, Hasibul Rasheeq, Asmaa Salem Farag, Stephen Wood, Jordan Boyce, Mihai Diaconeasa. "Introducing OpenPRA's Quantification Engine: Exploring Capabilities, Recognizing Limitations, and Charting the Path to Enhancement". In: Las Vegas, NV: American Nuclear Society, 2024. DOI: `doi.org/10.13182/T130-43362`.

[2] **Arjun Earthperson**, Egemen Aras, Asmaa Salem Farag, Mihai Diaconeasa. "Towards a Deep-Learning based Heuristic for Optimal Variable Ordering in Binary Decision Diagrams to Support Fault Tree Analysis". In: Las Vegas, NV: American Nuclear Society, 2024.

[3] **Arjun Earthperson**, Priyanka Pandit, Mihai Diaconeasa. "Implementing Multiple Control Paths in the Dual Error Propagation Graph for Stochastic Failure Analysis of Digital Instrumentation and Control Systems". In: Las Vegas, NV: American Nuclear Society, 2024. DOI: `doi.org/10.13182/T130-43400`. URL: `https://epubs.ans.org/?a=56419`.

[4] **Arjun Earthperson**, Priyanka Pandit, Mihai Diaconeasa. "Quantifying Safety System Unavailability Margins via Inverse Estimation of Event Sequence Frequencies". In: Las Vegas, NV: American Nuclear Society, 2024.

[5] Priyanka Pandit, **Arjun Earthperson**, Mihai Diaconeasa. "Laying the Foundations for the Development of a Probabilistic Model Checking Method to Quantify Common Cause Failure Parameters in Digital Instrumentation and Control". In: Las Vegas, NV: American Nuclear Society, 2024. DOI: `doi.org/10.13182/T130-43603`.

[6] Priyanka Pandit, **Arjun Earthperson**, Mihai Diaconeasa. "Leveraging Inverse Uncertainty Quantification to Enhance the Resilience of Nuclear Power Plant Construction Duration Estimation". In: Las Vegas, NV: American Nuclear Society, 2024. DOI: `doi.org/10.13182/T130-43602`.

[7] Asmaa Salem Farag, Amir Afzali, Yasir Arafat, Matt Loszak, **Arjun Earthperson**, Mihai Diaconeasa. "Aalo Atomics' High Level Licensing Strategy". In: Las Vegas, NV: American Nuclear Society, 2024.

[8] Asmaa Salem Farag, S Wood, **Arjun Earthperson**, Egemen Aras, Jordan Boyce, Mihai Diaconeasa. "Evaluating PRA Tools for Accurate and Efficient Quantifications: A Follow-Up Benchmarking Study Including FTREX". In: Las Vegas, NV: American Nuclear Society, 2024. DOI: doi.org/10.13182/T130-43377.

[9] Egemen Aras, Asmaa Farag, **Arjun Earthperson**, Mihai Diaconeasa. "Method of Developing a SCRAM Parallel Engine for Efficient Quantification of Probabilistic Risk Assessment Models". en. In: *18th International Probabilistic Safety Assessment and Analysis (PSA 2023)*. Knoxville, TN: American Nuclear Society, 2023, pp. 134–140. ISBN: 978-0-89448-792-7. DOI: 10.13182/PSA23-41054. URL: https://www.ans.org/pubs/proceedings/article-53973/ (visited on 04/02/2024).

[10] Egemen Aras, Asmaa Farag, **Arjun Earthperson**, Mihai Diaconeasa. "Methodology and Demonstration for Performance Analysis of a Probabilistic Risk Assessment Quantification Engine: SCRAM". en. In: *18th International Probabilistic Safety Assessment and Analysis (PSA 2023)*. Knoxville, TN: American Nuclear Society, 2023, pp. 452–459. ISBN: 978-0-89448-792-7. DOI: 10.13182/PSA23-41053. URL: https://www.ans.org/pubs/proceedings/article-54005/ (visited on 04/02/2024).

[11] **Arjun Earthperson**, Egemen M. Aras, Asmaa S. Farag, Mihai A. Diaconeasa. "Introducing OpenPRA: A Web-Based Framework for Collaborative Probabilistic Risk Assessment". In: *Volume 13: Research Posters; Safety Engineering, Risk and Reliability Analysis*. New Orleans, Louisiana, USA: American Society of Mechanical Engineers, 2023, V013T15A014. ISBN: 978-0-7918-8770-7. DOI: 10.1115/IMECE2023-111708. URL: https://asmedigitalcollection.asme.org/IMECE/proceedings/IMECE2023/87707/V013T15A014/1196307 (visited on 04/02/2024).

[12] **Arjun Earthperson**, Priyanka Pandit, Mihai Diaconeasa. "Introducing Multiple Control Paths in the Dual Error Propagation Graph for Stochastic Failure Analysis of Digital Instrumentation and Control Systems". en. In: *18th International Probabilistic Safety Assessment and Analysis (PSA 2023)*. Knoxville, TN: American Nuclear Society, 2023, pp. 209–218. ISBN: 978-0-89448-792-7. DOI: 10.13182/PSA23-41248. URL: https://www.ans.org/pubs/proceedings/article-53981/ (visited on 04/02/2024).

[13] Asmaa Farag, Egemen Aras, **Arjun Earthperson**, S. Wood, Jordan Boyce, Mihai Diaconeasa. "Preliminary Benchmarking of SAPHSOLVE, XFTA, and SCRAM Using Synthetically Generated Fault Trees with Common Cause Failures". en. In: *18th International Probabilistic Safety Assessment and Analysis (PSA 2023)*. Knoxville, TN: American Nuclear Society, 2023, pp. 40–49. ISBN: 978-0-89448-792-7. DOI: 10.13182/PSA23-41031. URL: https://www.ans.org/pubs/proceedings/article-53964/ (visited on 04/02/2024).

[14] Priyanka Pandit, **Arjun Earthperson**, Han Bao, Mihai Diaconeasa. "Analyzing Hardware and Software Common Cause Failures in Digital Instrumentation and Control Systems Using Dual Error Propagation Method". en. In: *18th International Probabilistic Safety Assessment and Analysis (PSA 2023)*. Knoxville, TN: American Nuclear Society, 2023, pp. 12–21. ISBN: 978-0-89448-792-7. DOI: 10.13182/PSA23-41062. URL: https://www.ans.org/pubs/proceedings/article-53961/ (visited on 04/02/2024).

[15] Priyanka Pandit, **Arjun Earthperson**, Daniel Nevius, Mihai Diaconeasa. "Quantifying the Likelihood of Nuclear Supply Chain Shortage Risk". en. In: *18th International Probabilistic Safety Assessment and Analysis (PSA 2023)*. Knoxville, TN: American Nuclear Society, 2023, pp. 668–674. ISBN: 978-0-89448-792-7. DOI: 10.13182/PSA23-41229. URL: https://www.ans.org/pubs/proceedings/article-54029/ (visited on 04/02/2024).

[16] Molly Prins, Thomas M. O'Connell, **Arjun Earthperson**, Yahya A. Alzahrani, Mihai A. Diaconeasa. "Leveraging Probabilistic Risk Assessment and Machine Learning for Safety and Cost Optimization in HAZMAT Transportation". In: *Volume 13: Research Posters; Safety Engineering, Risk and Reliability Analysis*. New Orleans, Louisiana, USA: American Society of Mechanical Engineers, 2023, V013T15A012. ISBN: 978-0-7918-8770-7. DOI: 10.1115/IMECE2023-114273. URL: https://asmedigitalcollection.asme.org/IMECE/proceedings/IMECE2023/87707/V013T15A012/1196286 (visited on 04/02/2024).

[17] Egemen M. Aras, Asmaa S. Farag, **Arjun Earthperson**, Mihai A. Diaconeasa. "Benchmark Study of XFTA and SCRAM Fault Tree Solvers Using Synthetically Generated Fault Trees Models". In: *Volume 9: Mechanics of Solids, Structures, and Fluids; Micro- and Nano-Systems Engineering and Packaging; Safety Engineering, Risk, and Reliability Analysis; Research Posters*. Columbus, Ohio, USA: American Society of Mechanical Engineers, 2022, V009T14A016. ISBN: 978-0-7918-8671-7. DOI: 10.1115/IMECE2022-95783. URL: https://asmedigitalcollection.asme.org/IMECE/proceedings/IMECE2022/86717/V009T14A016/1157443 (visited on 02/15/2023).

[18] Bineh Ndefru, Karthik Sankaran, Theresa Stewart, Ali Mosleh, **Arjun Earthperson**, Natalie Zawalick. "Risk-Informed Decision-Making Tool for Covid-19 Community Behavior and Intervention Scenario Assessment". English. In: *Proceedings of the 16th International Conference on Probabilistic Safety Assessment and Management (PSAM)*. Vol. 3. Honolulu, Hawaii, USA: Curran Associates, Inc., 2022. ISBN: 978-1-71386-375-5. URL: https://www.iapsam.org/PSAM16/papers/KS45-PSAM16.pdf.

[19] Courtney Mariko Yang Hui Otani, Robby Christian, Steven R Prescott, Mihai Diaconeasa, **Arjun Earthperson**. "Probabilistic Methods for Cyclical and Coupled Systems with Changing Failure Rates". English. In: *Report Number: INL/CON-22-66839-Rev000*. Research Org.: Idaho National Lab. (INL), Idaho Falls, ID (United States), 2022. URL: https://www.osti.gov/biblio/1885929.

[20] **Arjun Earthperson**, Mihai A. Diaconeasa. "Verification Study of the Nuclear PRA for the Mars 2020 Mission Following Accidental Orbital Re-Entry". In: *Volume 13: Safety Engineering, Risk, and Reliability Analysis; Research Posters*. Virtual, Online: American Society of Mechanical Engineers, 2021, V013T14A019. ISBN: 978-0-7918-8569-7. DOI: 10.1115/IMECE2021-71359. URL: https://asmedigitalcollection.asme.org/IMECE/proceedings/IMECE2021/85697/V013T14A019/1133290 (visited on 02/04/2022).

[21] Priyanka Pandit, **Arjun Earthperson**, Alp Tezbasaran, Mihai A. Diaconeasa. "A Quantitative Approach to Assess the Likelihood of Supply Chain Shortages". In: *Volume 13: Safety Engineering, Risk, and Reliability Analysis; Research Posters*. Virtual, Online: American Society of Mechanical Engineers, 2021, V013T14A023. ISBN: 978-0-7918-8569-7. DOI: 10.1115/IMECE2021-73696. URL: https://asmedigitalcollection.asme.org/IMECE/proceedings/IMECE2021/85697/V013T14A023/1133260 (visited on 02/04/2022).

[22] Priyanka Pandit, Alp Tezbasaran, **Arjun Earthperson**, Mihai A. Diaconeasa. "Evaluating the Implementation of Distributed Ledger Technology for the Licensing and Regulation of Nuclear Power Plants". In: *Volume 8B: Energy*. Virtual, Online: American Society of Mechanical Engineers, 2021, V08BT08A016. ISBN: 978-0-7918-8564-2. DOI: 10.1115/IMECE2021-71730. URL: https://asmedigitalcollection.asme.org/IMECE/proceedings/IMECE2021/85642/V08BT08A016/1132953 (visited on 02/04/2022).

Long-Standing Needs in PRA Quantification

- Large-scale PRA models ($\geq \approx 10^4$ components) remain computationally taxing.
- To ease burden, approximations are used, implicating accuracy.
- No knobs for controlling trade-off between accuracy and speed.

Long-Standing Needs in PRA Quantification

- Large-scale PRA models ($\geq\approx 10^4$ components) remain computationally taxing.
- To ease burden, approximations are used, implicating accuracy.
- No knobs for controlling trade-off between accuracy and speed.

Large-scale PRA models still takes days to quantify.

Evolving Hardware Landscape

Industry responding to emerging ML compute challenges by investing heavily in data-parallel hardware

- GPUs, tensor cores provide high throughput for integer operations.
- Current-gen consumer hardware already supports specialized ops (Intel AMX, VNNI).

Designed for Massive Workloads

- $\approx 10^9$ parameters on mobile devices, $\approx 10^{12}$ on HPC/cloud.
- Comparatively, largest PRA models: $\approx 10^6$ parameters.

Evolving Hardware Landscape

Industry responding to emerging ML compute challenges by investing heavily in data-parallel hardware

- GPUs, tensor cores provide high throughput for integer operations.
- Current-gen consumer hardware already supports specialized ops (Intel AMX, VNNI).

Designed for Massive Workloads

- $\approx 10^9$ parameters on mobile devices, $\approx 10^{12}$ on HPC/cloud.
- Comparatively, largest PRA models: $\approx 10^6$ parameters.

But PRA models have no overlap with ML models.

Evolving Hardware Landscape

Industry responding to emerging ML compute challenges by investing heavily in data-parallel hardware

- GPUs, tensor cores provide high throughput for integer operations.
- Current-gen consumer hardware already supports specialized ops (Intel AMX, VNNI).

Designed for Massive Workloads

- $\approx 10^9$ parameters on mobile devices, $\approx 10^{12}$ on HPC/cloud.
- Comparatively, largest PRA models: $\approx 10^6$ parameters.

But PRA models have no overlap with ML models(?)

Evolving Hardware Landscape

Industry responding to emerging ML compute challenges by investing heavily in data-parallel hardware

- GPUs, tensor cores provide high throughput for integer operations.
- Current-gen consumer hardware already supports specialized ops (Intel AMX, VNNI).

Designed for Massive Workloads

- $\approx 10^9$ parameters on mobile devices, $\approx 10^{12}$ on HPC/cloud.
- Comparatively, largest PRA models: $\approx 10^6$ to 10^9 parameters.

But PRA models have no overlap with ML models (?) - **Research Question**
Probability estimation analogous to inference in feed-forward networks.

Research Questions

- How can large-scale PRA models be quantified efficiently?
- What are the overlaps between PRA models and Probabilistic Circuits?
- What guarantees can be made about tractability and accuracy when using Monte Carlo for probability estimation?
- What techniques can be developed to exploit native hardware parallelism for PRA quantification?

Research Contribution

1. **Bridge PRA modeling semantics with Probabilistic Circuits**
2. **Develop data-parallel Monte Carlo methods for evaluating Boolean circuits**
3. **Open-source implementations and benchmarks**
4. **Develop Monte Carlo sampling techniques for computing partial-derivatives on Boolean circuits**

Research Objective 1

Bridge PRA Modeling Semantics with Probabilistic Circuits

The Triplet Definition of Risk

- Define risk as a set of triplets, each representing:
  1. What can go wrong? (S_i)
  2. How likely is it to happen? (L_i)
  3. What are the consequences? (X_i)

- $$R = \{\langle S_i, L_i, X_i \rangle\}_c, \quad (1)$$

  c represents completeness in enumerating *all* relevant scenarios.

Scenario S_i Modeling in PRA

- Each scenario unfolds from initiating events (IEs), followed by conditional branching events.
- Fundamental goal: assign probabilities to these scenarios and assess resulting outcomes (e.g., core damage, large release).
- Implementation typically uses structured diagrams such as:
  - Event Trees (ETs): forward chaining from IE to various end states.
  - Fault Trees (FTs): top-down decomposition to basic events (component failures).

Event Trees

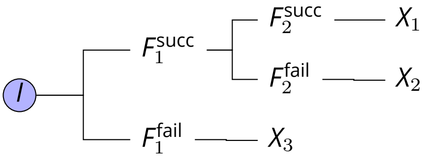


Figure: Illustrative event tree with an initiating event I, two functional events F_1 and F_2, and three end-states X_1, X_2, X_3.

- An Event Tree represents how an initiating event I can branch into multiple functional-event outcomes.

Event Trees (cont.)

- Each functional event F_k may succeed or fail, driving the path toward a distinct end-state X_j.
- If ω_j denotes one branch leading to X_j, then the branch probability often factors as

$$p(\omega_j) = p(I) \times \prod_{k=1}^{n} p\left(F_k^{\alpha_k} \mid \text{all previous outcomes}\right).$$

- As a logical expression, each ω_j translates to an AND of success/failure literals, and the overall set of end-states is an OR of these branches:

$$\Omega = \omega_1 \vee \omega_2 \vee \ldots \vee \omega_m.$$

Event Trees (cont.)

- Consequently, ETs are in *sum of products* (SOP) or *disjunctive normal form* (DNF), where each product term identifies one success/failure path and the scenario-level outcome is the logical OR across all such paths.
- Graphically straightforward, but can combinatorially expand for deep branching.

Fault Trees

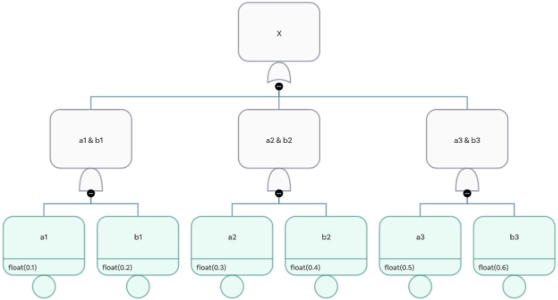


Figure: Fault tree with 6 basic events, top event $X = (a_1 \bigwedge b_1) \bigvee (a_2 \bigwedge b_2) \bigvee (a_3 \bigwedge b_3)$

Fault Trees (cont.)

- A Fault Tree describes how a top event (system failure) can result from lower-level component or subsystem failures.
- Internal gates (AND, OR, k-of-n, etc.) combine events in logical fashion:

$$\text{Output} = \begin{cases} \bigwedge_{i=1}^{k} e_i, & \text{(AND)} \\ \bigvee_{i=1}^{k} e_i, & \text{(OR)} \\ \dots \end{cases}$$

- Basic events (BEs) in the leaves have assigned probabilities $p(b)$. Independence often assumed unless modeling common-cause failures.

Fault Trees (cont.)

- The top event failure probability can be written as:

$$\Pr[\text{Top Fail}] = \sum_{S \subseteq \mathcal{B}} \left[\pi_F(S, \text{Top}) \prod_{b \in S} p(b) \prod_{b \notin S} [1 - p(b)] \right]. \tag{2}$$

Linking Event Trees and Fault Trees in PRA

- Real systems often combine:
  - Forward branching dynamics via Event Trees (ET).
  - Subsystem or component reliability logic via Fault Trees (FT).
- An event tree branch may call a specific FT top event to collect system failure or success.
- Conversely, a fault tree output may feed back into an event tree branch as an initiating event or functional node outcome.
- This multi-level interconnection $\implies$ a need for a unified model capturing both forward branching (ET) and hierarchical failure logic (FT).

Probabilistic Directed Acyclic Graph (PDAG)

- A PDAG is a Directed Acyclic Graph whose edges carry either:
  - Conditional probabilities (e.g., for event tree branches).
  - Logical dependencies (e.g., for fault tree gates).
- Nodes may include:
  - Basic events (BEs) with known probabilities.
  - ET or FT intermediate events storing partial results.
  - Any top-level node (e.g., a final end-state) with no children.
- The absence of cycles guarantees consistent flow from initial seeds (basic events, initiating events) to final outcomes.
- PDAG forms the structural backbone for bridging scenario-based expansions with gate-based logic in a single coherent representation.

Probabilistic Circuits: A Brief Overview

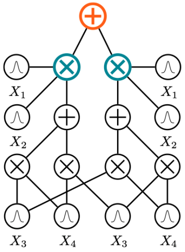


Figure: Probabilistic circuit with 4 inputs X_i, product gates (blue), sum gate (orange)

- A **probabilistic circuit** is a directed acyclic graph (DAG) that encodes a joint probability distribution through *sum-gates* and *product-gates*.
- **Sum-gates** approximate mixture distributions:

$$p_v(\mathbf{x}) = \sum_{u \in \mathrm{ch}(v)} \theta_{v,u}\, p_u(\mathbf{x}), \quad \text{with} \sum_{u \in \mathrm{ch}(v)} \theta_{v,u} = 1.$$

  Each child distribution is weighted by a nonnegative parameter $\theta_{v,u}$.
- **Product-gates** factorize independent variable sets:

$$p_v(\mathbf{x}) = \prod_{u \in \mathrm{ch}(v)} p_u(\mathbf{x}_u),$$

  assuming disjoint subsets of variables for each child u.

- **Leaf nodes** (inputs) often correspond to known base distributions. When evaluated upward through the DAG, each internal node yields its own distribution, culminating in a root node that represents the full model.
- **Key motivation**:
  - Tractable inference: evaluation and certain marginal or conditional queries can be performed in time proportional to circuit size.
  - Decomposability and modularity: separate, interpretable substructures that can be reused or combined for large-scale systems.

Compiled PRA Graphs are PDAGs

- A unified PRA model can be viewed as

$$\mathcal{M} = \langle \mathcal{V}, \mathcal{A}, \{p(b)\}, \{\theta_{u \to v}\}, \pi_F \rangle,$$

where $\mathcal{V}$ includes basic events, ET nodes, and FT gates, and $\mathcal{A}$ is the acyclic edge set.

- Event-tree edges carry transitional probabilities $\theta_{u \to v}$ summing to 1 from each node.
- Fault-tree nodes embed Boolean logic π_F that checks if inputs fail under a subset of basic events.

Compiled PRA Graphs are PDAGs (cont.)

- This PDAG perspective:
  - Ensures systematic accounting of all scenario paths and subsystem logic.
  - Aligns naturally with tractable Probabilistic Circuits, since each node's distribution/function can be embedded in a sum-product style DAG.
  - Offers a foundation for efficient Monte Carlo or advanced inference methods.
- Ongoing work to bridge PRA PDAG semantics with Probabilistic Circuits:
  - PDAGs can be transformed to equivalent canonical forms (there are tradeoffs).

Research Objective 2

Develop data-parallel Monte Carlo methods for evaluating Boolean circuits

Boolean Functions: Basic Concepts

- Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ be a vector of n Boolean variables, each $x_i \in \{0, 1\}$.
- A *Boolean function* is any map $F(\mathbf{x}) : \{0, 1\}^n \to \{0, 1\}$.
- Example: If F encodes "system fails," then $F(\mathbf{x}) = 1$ signifies a failure mode, where $\mathbf{x}$ captures component states.
- Modeling perspective:
  - AND, OR, NOT, k-of-n gates allow composing complex logic.
  - Each F can be evaluated deterministically if we know $\mathbf{x}$.

Exact Probability Estimation: Inclusion-Exclusion

- Suppose each x_i has a probability $p_i = \Pr[x_i = 1]$, assuming independence.
- We want $\Pr[F(\mathbf{x}) = 1]$, which is

$$\Pr\left[F(\mathbf{X}) = 1\right] = \sum_{\mathbf{x}\in\{0,1\}^n} F(\mathbf{x}) \prod_{i=1}^{n} \left[p_i^{x_i}(1-p_i)^{1-x_i}\right].$$

- For sets of events, using the *inclusion-exclusion principle*:

$$\Pr\left(\bigcup_{i=1}^{n} E_i\right) = \sum_{k=1}^{n} (-1)^{k+1} \sum_{1\leq i_1<\cdots<i_k\leq n} \Pr\left(E_{i_1} \cap \cdots \cap E_{i_k}\right).$$

Approximation Methods: REA and MCUB

For large *n*, exact enumeration of subsets is exponential, making it impractical for large Boolean circuits.

- **Rare-Event Approximation (REA):**
  - Assumes each event has small probability $p_i \ll 1$.
  - Overlaps (intersections of multiple failures) are deemed negligible.
  - Probability of the union $\approx \sum_i \Pr[E_i]$, ignoring higher-order terms.

Approximation Methods: REA and MCUB (cont.)

- **Min-Cut Upper Bound (MCUB):**

$$\Pr\left[\bigcup_{C\in\{\mathrm{MCS}\}} C\right] \leq \sum_{C\in\{\mathrm{MCS}\}} \prod_{b\in C} p_b, \tag{3}$$

  - Interprets each minimal cut set (MCS) as a distinct mechanism for failure.
  - Sums (over)estimate total failure if MCSs share components.
  - Often used as a conservative bound in safety analyses.

- Both methods reduce complexity but can misestimate the true probability when events are not truly rare or heavily intersect.

Monte Carlo Sampling

- Rather than summing or bounding all combinations of failures, *simulate* random draws of $\mathbf{X}$.
- Each Monte Carlo iteration:
  1. Sample $x_1, x_2, \ldots, x_n \overset{\text{i.i.d.}}{\sim} \prod p(x_i)$.
  2. Evaluate the Boolean function $F(\mathbf{x})$ (cost is just logical gate evaluation).
  3. Collect whether $F(\mathbf{x}) = 1$ (failure) or 0 (success).
- Repeating for many samples $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ yields a *sample average* estimate of the probability.
- Benefits:
  - Bypasses explicit inclusion-exclusion expansions.
  - Straightforward to parallelize (evaluate each draw in separate threads or blocks).

Estimator for the Expected Value (i.e., Probability)

- A Boolean function $F(\mathbf{x})$ can be viewed as an indicator function: $F(\mathbf{x}) \in \{0, 1\}$.
- The event $\{F(\mathbf{X}) = 1\}$ has probability $\mathbb{E}[F(\mathbf{X})]$.
- **Monte Carlo estimator:**

$$\widehat{P}_N = \frac{1}{N} \sum_{i=1}^{N} F(\mathbf{x}^{(i)}),$$

  where each $\mathbf{x}^{(i)}$ is a random draw from the input distribution.
- By the Law of Large Numbers,

$$\lim_{N \to \infty} \widehat{P}_N = \Pr\left[F(\mathbf{X}) = 1\right], \quad \text{almost surely.}$$

- Error decreases at rate $\mathcal{O}(1/\sqrt{N})$, analyzed via the Central Limit Theorem.

Boolean Derivatives: Definition and Interpretation

- **Boolean Derivative Concept:** For a Boolean function $F(\mathbf{x})$ with $\mathbf{x} = (x_1, \ldots, x_n)$, the derivative with respect to x_i is defined via XOR:

$$\frac{\partial F}{\partial x_i} = F(x_i = 0, \mathbf{x}_{-i}) \oplus F(x_i = 1, \mathbf{x}_{-i}),$$

  where $\oplus$ denotes the exclusive-OR operation, and $\mathbf{x}_{-i}$ are all variables except x_i.
- **Interpretation:**
  - $\frac{\partial F}{\partial x_i}(\mathbf{x}) = 1$ whenever *flipping* x_i changes the value of F under the specific configuration $\mathbf{x}_{-i}$.
  - Captures *sensitivity*: if $\frac{\partial F}{\partial x_i}$ rarely equals 1, then F is robust to changes in x_i.

Extension to Monte Carlo Estimation of Boolean Derivatives

- **Key Idea:** Estimate $\mathbb{E}[\,\partial F/\partial x_i\,]$ by sampling random configurations $\mathbf{x}^{(s)}$ of the Boolean inputs, then checking how F changes when x_i is flipped.
- **Sampling Procedure:**
  1. Draw $\mathbf{x}^{(s)} = (x_1^{(s)}, \ldots, x_n^{(s)})$ from the distribution of interest.
  2. Form $\mathbf{x}^{(s)} \oplus \mathbf{e}_i$ by flipping the ith coordinate.
  3. Compute:

$$\frac{\partial F}{\partial x_i}(\mathbf{x}^{(s)}) = F(\mathbf{x}^{(s)}) \oplus F(\mathbf{x}^{(s)} \oplus \mathbf{e}_i).$$

- **Insight:**
  - Sensitivity and importance analysis using sampling methods.
  - Gradient computation opens a path towards learning-based tasks.

Avoiding Inclusion-Exclusion via Monte Carlo

- Exact expansions for large circuits require enumerating all subsets of failing components or gates, which is computationally huge.
- In contrast, *Monte Carlo* draws a sample $\mathbf{x} \in \{0,1\}^n$ and directly evaluates $F(\mathbf{x})$ without enumerating *all* subsets.
- Each run picks a single draw of failed components from the distribution. After many runs, the frequency of $F = 1$ approximates its probability.
- Results:
  - No exponential blow-up in the number of terms.
  - Straightforward extension to complex gate structures, correlated variables.
  - Parallelizable on modern CPU/GPU architectures.

Data-Parallel Implementation using SYCL

Data-Parallel Monte Carlo for Boolean Circuits:

- Simultaneous evaluation of *all* intermediate gates, success, and failure paths.
- Relax coherence constraints - arbitrary shapes with NOT gates permitted.
- Vectorized bitwise hardware ops for logical primitives (AND, OR, XOR, etc.)
- Specialized treatment of k/n logic, without expansion.
- Simultaneous use of all available compute - GPUs, multicore CPUs.

Preliminary Case Study

Overview: Aralia Dataset

- **Dataset Composition:** The Aralia collection consists of 43 distinct fault trees, each with varying numbers of basic events (BEs), gate types (AND, OR, K/N, XOR), and minimal cut-set counts.
- **Diverse Problem Sizes:** Small trees (e.g. 25–32 BEs) through large models with over 1,500 BEs.
- **Wide Probability Range:** Top-event probabilities spanning from rare events near 10^{-13} to fairly likely failures with probability above 0.7.
- **Model Variability:** Some trees are primarily AND/OR, others incorporate more advanced gates (K/N, XOR, NOT), providing thorough coverage of typical (and atypical) fault tree logic structures.

#	Fault Tree	Basic Events	Logic Gates					Minimal Cut Sets	Top Event Probability
			Total	AND	K/N	XOR	NOT		
1	baobab1	61	84	16	9	-	-	46,188	1.01708E-04
2	baobab2	32	40	5	6	-	-	4,805	7.13018E-04
3	baobab3	80	107	46	-	-	-	24,386	2.24117E-03
4	cea9601	186	201	69	8	-	30	130,281,976	1.48409E-03
5	chinese	25	36	13	-	-	-	392	1.17058E-03
6	das9201	122	82	19	-	-	-	14,217	1.34237E-02
7	das9202	49	36	10	-	-	-	27,778	1.01154E-02
8	das9203	51	30	1	-	-	-	16,200	1.34880E-03
9	das9204	53	30	12	-	-	-	16,704	6.07651E-08
10	das9205	51	20	2	-	-	-	17,280	1.38408E-08
11	das9206	121	112	21	-	-	-	19,518	2.29687E-01
12	das9207	276	324	59	-	-	-	25,988	3.46696E-01
13	das9208	103	145	33	-	-	-	8,060	1.30179E-02
14	das9209	109	73	18	-	-	-	8.20E+10	1.05800E-13
15	das9601	122	288	60	36	12	14	4,259	4.23440E-03
16	das9701	267	2,226	1,739	-	-	992	26,299,506	7.44694E-02
17	edf9201	183	132	12	-	-	-	579,720	3.24591E-01
18	edf9202	458	435	45	-	-	-	130,112	7.81302E-01
19	edf9203	362	475	117	-	-	-	20,807,446	5.99589E-01
20	edf9204	323	375	106	-	-	-	32,580,630	5.25374E-01
21	edf9205	165	142	30	-	-	-	21,308	2.09351E-01

22	edf9206	240	362	126	-	-	-	385,825,320	8.61500E-12
23	edfpa14b	311	290	70	-	-	-	105,955,422	2.95620E-01
24	edfpa14o	311	173	42	-	-	-	105,927,244	2.97057E-01
25	edfpa14p	124	101	42	-	-	-	415,500	8.07059E-02
26	edfpa14q	311	194	55	-	-	-	105,950,670	2.95905E-01
27	edfpa14r	106	132	55	-	-	-	380,412	2.09977E-02
28	edfpa15b	283	249	61	-	-	-	2,910,473	3.62737E-01
29	edfpa15o	283	138	33	-	-	-	2,906,753	3.62956E-01
30	edfpa15p	276	324	33	-	-	-	27,870	7.36302E-02
31	edfpa15q	283	158	45	-	-	-	2,910,473	3.62737E-01
32	edfpa15r	88	110	45	-	-	-	26,549	1.89750E-02
33	elf9601	145	242	97	-	-	-	151,348	9.66291E-02
34	ftr10	175	94	26	-	-	-	305	4.48677E-01
35	isp9601	143	104	25	1	-	-	276,785	5.71245E-02
36	isp9602	116	122	26	-	-	-	5,197,647	1.72447E-02
37	isp9603	91	95	37	-	-	-	3,434	3.23326E-03
38	isp9604	215	132	38	-	-	-	746,574	1.42751E-01
39	isp9605	32	40	8	6	-	-	5,630	1.37171E-05
40	isp9606	89	41	14	-	-	-	1,776	5.43174E-02
41	isp9607	74	65	23	-	-	-	150,436	9.49510E-07
42	jbd9601	533	315	71	-	-	-	150,436	7.55091E-01
43	nus9601	1,567	1,622	392	47	-	-	unknown	unknown

Benchmarking Setup: Hardware and Environment

- **Target Hardware:**
  - GPU: NVIDIA® GeForce GTX 1660 SUPER (6 GB GDDR6, 1,408 CUDA cores).
  - CPU: Intel® Core™ i7-10700 (2.90 GHz, turbo-boost, hyperthreading).
- **Software Stack:**
  - SYCL-based (AdaptiveCpp/HipSYCL), with LLVM-IR JIT for kernel compilation.
  - Compiler optimization at `-O3` for efficient code generation.
  - Repeated runs (5+) to mitigate transient variations.
- **Measured Time:** Includes entire wall-clock duration, from host-device transfers and JIT compilation to final result collection.

Monte Carlo Execution and Implementation

- **Sampling Strategy:**
  - Single pass per fault tree, generating as many samples as fit in 6 GB GPU memory.
  - 128-bit Philox4x32x10 pseudo-random number generator, parallel threads.
- **Bit-Packing Optimization:**
  - Each group of 64 Monte Carlo outcomes stored in a single 64-bit word.
  - Enables vectorized instructions (e.g. `popcount`) and reduces memory I/O.
- **Data Types:**
  - Tallies in 64-bit integers.
  - Probability accumulations in double precision (64-bit float).

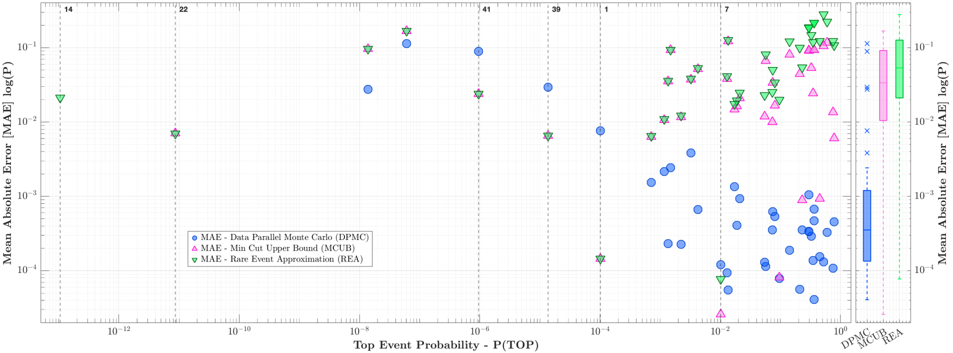
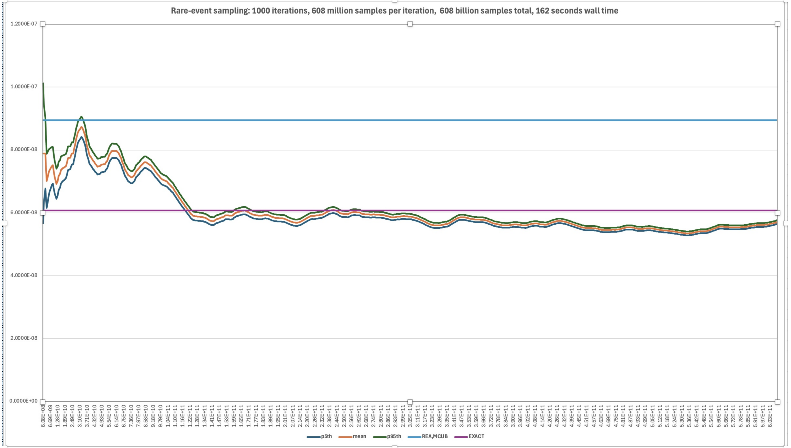


Figure: Mean Absolute Error – Exact (BDD) vs Approximate Methods


Rare-event sampling: 1000 iterations, 608 million samples per iteration, 608 billion samples total, 162 seconds wall time
1.2000E-07
1.0000E-07
8.0000E-08
6.0000E-08
4.0000E-08
2.0000E-08
0.0000E+00
p5th
mean
p95th
REA,MCUB
EXACT

#	Fault Tree	Mean Absolute Error - log(P)				Runtime [sec]
		REA	MCUB	Monte Carlo		
1	baobab1	1.45156×10^{-4}	1.45156×10^{-4}	7.61880×10^{-3}	2.5×10^{8}	0.262
2	baobab2	6.48628×10^{-3}	6.34705×10^{-3}	1.54436×10^{-3}	2.5×10^{8}	0.209
3	baobab3	1.21509×10^{-2}	1.16701×10^{-2}	2.24843×10^{-4}	2.4×10^{8}	0.259
4	cea9601	9.36195×10^{-2}	9.32207×10^{-2}	2.41802×10^{-3}	1.2×10^{8}	0.262
5	chinese	1.08742×10^{-2}	1.06354×10^{-2}	2.14601×10^{-3}	9.4×10^{8}	0.277
6	das9201	1.26649×10^{-1}	1.22765×10^{-1}	5.49963×10^{-5}	2.3×10^{8}	0.279
7	das9202	7.72743×10^{-5}	2.57596×10^{-5}	1.20232×10^{-4}	5.2×10^{8}	0.295
8	das9203	3.59019×10^{-2}	3.55935×10^{-2}	2.31768×10^{-4}	5.2×10^{8}	0.292
9	das9204	1.68086×10^{-1}	1.68087×10^{-1}	1.13495×10^{-1}	6.1×10^{8}	0.292
10	das9205	9.63825×10^{-2}	9.63725×10^{-2}	2.76190×10^{-2}	3.3×10^{9}	0.958
11	das9206	5.43561×10^{-2}	8.89660×10^{-4}	3.51548×10^{-4}	2.0×10^{8}	0.269
12	das9207	1.18486×10^{-1}	2.45492×10^{-2}	1.36519×10^{-4}	9.5×10^{7}	0.282
13	das9208	4.12808×10^{-2}	3.81968×10^{-2}	9.34017×10^{-5}	2.5×10^{8}	0.307
14	das9209	2.11242×10^{-2}	1.70245×10^{1}		-	-
15	das9601	5.29285×10^{-2}	5.19122×10^{-2}	6.67174×10^{-4}	1.1×10^{8}	0.256
16	das9701	5.02804×10^{-2}	3.37565×10^{-2}	6.22978×10^{-4}	2.3×10^{7}	0.273
17	edf9201	1.48012×10^{-1}	5.36182×10^{-2}	2.88906×10^{-4}	1.8×10^{8}	0.315
18	edf9202	1.07181×10^{-1}	6.05976×10^{-3}	4.53900×10^{-4}	7.8×10^{7}	0.271

19	edf9203	$2.221\,46 \times 10^{-1}$	$1.172\,93 \times 10^{-1}$	$3.279\,93 \times 10^{-4}$	8.0×10^{7}	0.302
20	edf9204	$2.795\,31 \times 10^{-1}$	$1.055\,91 \times 10^{-1}$	$1.314\,16 \times 10^{-4}$	8.7×10^{7}	0.298
21	edf9205	$9.943\,39 \times 10^{-2}$	$4.462\,60 \times 10^{-2}$	$5.601\,46 \times 10^{-5}$	1.9×10^{8}	0.284
22	edf9206	$6.987\,97 \times 10^{-3}$	$7.077\,75 \times 10^{-3}$			-
23	edfpa14b	$1.855\,74 \times 10^{-1}$	$9.159\,83 \times 10^{-2}$	$1.047\,67 \times 10^{-3}$	9.4×10^{7}	0.267
24	edfpa14o	$1.864\,82 \times 10^{-1}$	$9.186\,65 \times 10^{-2}$	$3.390\,49 \times 10^{-4}$	9.8×10^{7}	0.275
25	edfpa14p	$3.400\,10 \times 10^{-2}$	$1.662\,83 \times 10^{-2}$	$5.350\,99 \times 10^{-4}$	2.1×10^{8}	0.294
26	edfpa14q	$1.856\,09 \times 10^{-1}$	$9.153\,66 \times 10^{-2}$	$3.332\,92 \times 10^{-4}$	9.6×10^{7}	0.282
27	edfpa14r	$2.480\,88 \times 10^{-2}$	$2.097\,29 \times 10^{-2}$	$9.338\,65 \times 10^{-4}$	2.1×10^{8}	0.294
28	edfpa15b	$2.163\,29 \times 10^{-1}$	$9.370\,65 \times 10^{-2}$	$4.678\,81 \times 10^{-4}$	1.1×10^{8}	0.283
29	edfpa15o	$2.165\,02 \times 10^{-1}$	$9.376\,27 \times 10^{-2}$	$4.068\,46 \times 10^{-5}$	1.1×10^{8}	0.282
30	edfpa15p	$2.525\,68 \times 10^{-2}$	$1.003\,82 \times 10^{-2}$	$3.543\,44 \times 10^{-4}$	2.6×10^{8}	0.299
31	edfpa15q	$2.163\,29 \times 10^{-1}$	$9.370\,65 \times 10^{-2}$	$6.747\,36 \times 10^{-4}$	1.1×10^{8}	0.284
32	edfpa15r	$1.946\,93 \times 10^{-2}$	$1.626\,68 \times 10^{-2}$	$4.049\,24 \times 10^{-4}$	2.5×10^{8}	0.290
33	elf9601	$1.981\,07 \times 10^{-2}$	$8.089\,25 \times 10^{-5}$	$7.866\,00 \times 10^{-5}$	2.3×10^{8}	0.274
34	ftr10	$1.220\,76 \times 10^{-1}$	$9.272\,68 \times 10^{-4}$	$1.548\,44 \times 10^{-4}$	2.1×10^{8}	0.297
35	isp9601	$8.083\,92 \times 10^{-2}$	$6.630\,74 \times 10^{-2}$	$1.132\,64 \times 10^{-4}$	1.8×10^{8}	0.271
36	isp9602	$1.745\,72 \times 10^{-2}$	$1.477\,82 \times 10^{-2}$	$1.352\,80 \times 10^{-3}$	2.3×10^{8}	0.281
37	isp9603	$3.823\,37 \times 10^{-2}$	$3.748\,15 \times 10^{-2}$	$3.823\,44 \times 10^{-3}$	2.7×10^{8}	0.278
38	isp9604	$1.208\,89 \times 10^{-1}$	$8.143\,13 \times 10^{-2}$	$1.886\,65 \times 10^{-4}$	1.4×10^{8}	0.280
39	isp9605	$6.573\,44 \times 10^{-3}$	$6.570\,32 \times 10^{-3}$	$2.934\,72 \times 10^{-2}$	5.0×10^{8}	0.262
40	isp9606	$2.278\,11 \times 10^{-2}$	$1.189\,83 \times 10^{-2}$	$1.303\,07 \times 10^{-4}$	3.4×10^{8}	0.289
41	isp9607	$2.388\,80 \times 10^{-2}$	$2.388\,80 \times 10^{-2}$	$1.281\,36 \times 10^{-1}$	3.8×10^{8}	0.282

42	jbd9601	$1.220\,01\times10^{-1}$	$1.353\,43\times10^{-2}$	$1.081\,16\times10^{-4}$	5.7×10^{7}	0.279
43	nus9601				1.6×10^{7}	0.289

The End