TUTORIAL ON PARATEMPORAL SIMULATION

USING TREE EXPANSION

93rd MORS Symposium Conference

Bernard Zeigler

Christian Koertje

RTSync Corp

www.rtsync.com

June , 2024

RTSync

Context: RTSync Paratemporal Simulation Development

- **Paratemporal simulation algorithms** for stochastic simulations with the goal of decreasing time to execute very high branching, long running models for statistically significant outcome distributions.

- **Tree Expansion simulation algorithms** that can achieve 10^x speed-up and scalability of stochastic systems while increasing knowledge of outcome distributions.

- **ParaDEVS (Paratemporal Simulation in DEVS) integrated in** RTSync's ***Discrete Event System Specification*** (DEVS)-based Simulation platform and Model-Based System Engineering Software for ease of user model development with new performance algorithms.

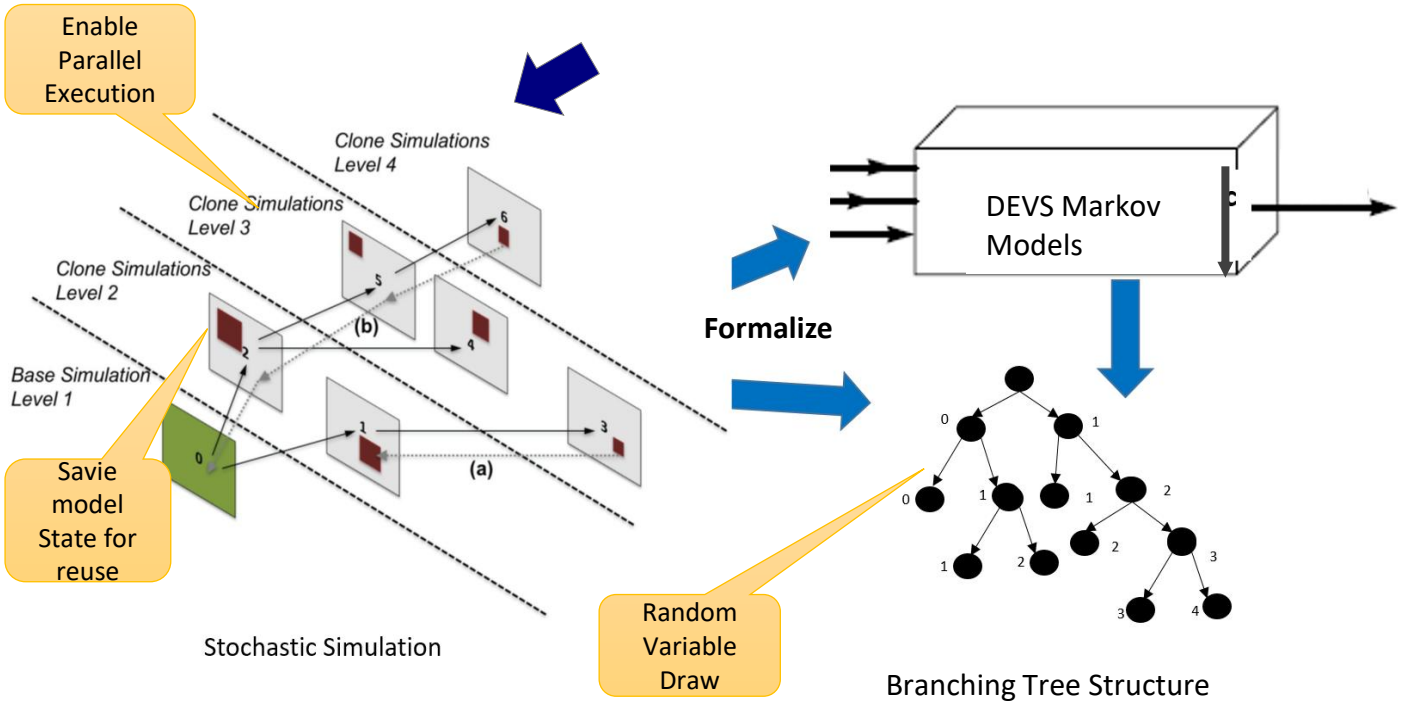
- **ParaDEVS execution on high performance computational architectures** on the cloud and in novel DEVS-chip environments where exploration of scenario spaces in parallel can further reduce execution time.

Agenda

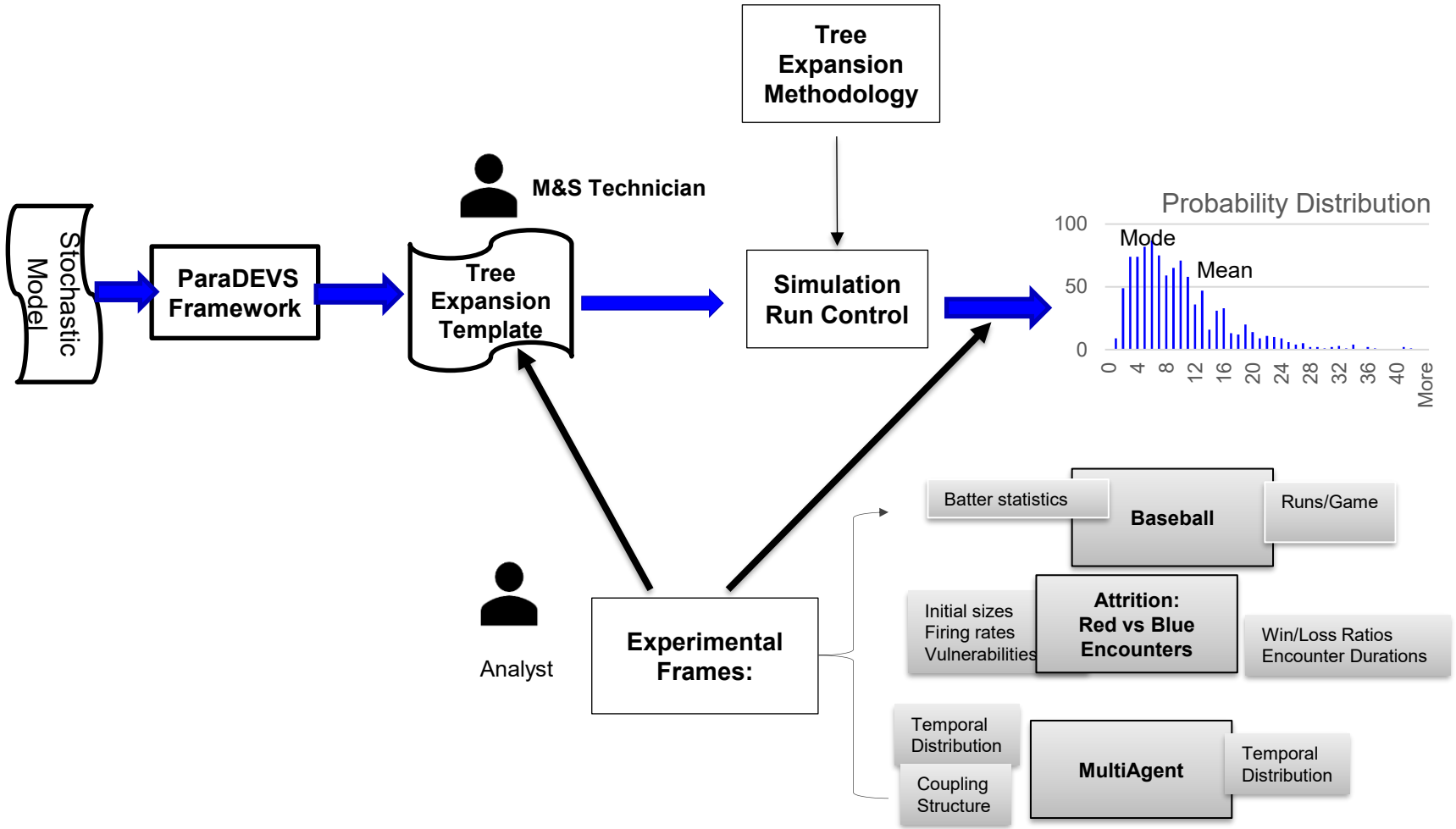
- **ParaDEVS Modeling and Simulation Concept of Operation**
- **Background and Problem Statement**
  - Tree Expansion Concepts
  - Expansion Control via Merging
- **Examples of ParaDEVS Use-Cases**
  - Chemical reaction,
  - Baseball game outcome
  - Financial stock trading
  - Air Battle
  - Kill Chain
- **Implementation Overview**
  - Two-level design, Simulation schema,
- **Summary**
- **Q&A**

DEVS-based Framework For Paratemporal Simulations.

- **Clone simulations for reuse, enabling parallel execution**
- **Employ homomorphism-based node merging to control tree expansion**
- **Execute trees in parallel to get earlier results sooner**



ParaDEVS Modeling and Simulation (M&S) Concept of Operation



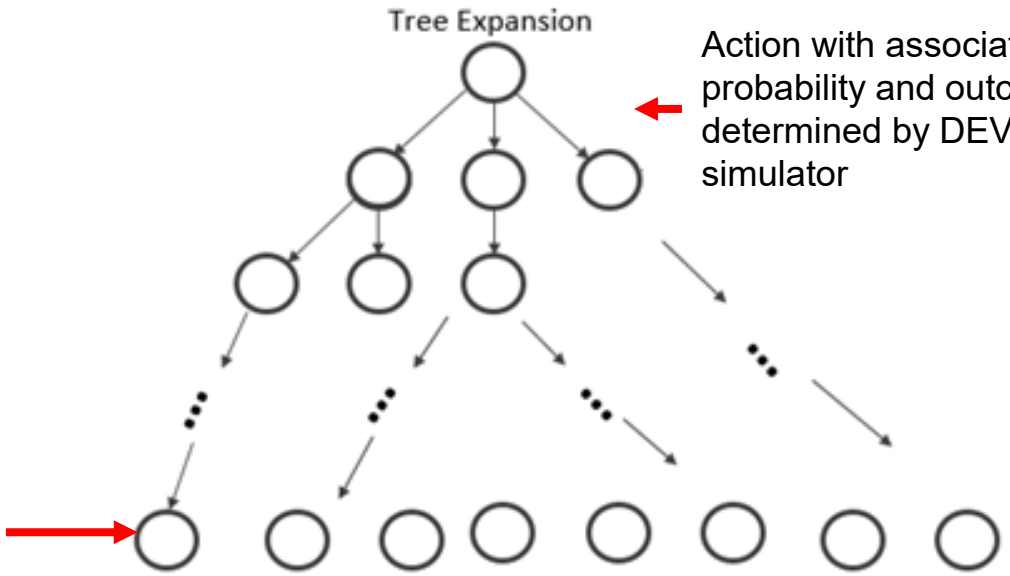
Background and Problem Statement

Tree Expansion Concepts

- Coin Tossing Binary Tree
- Depth vs. Breadth
- Node Merging to Manage Tree Growth

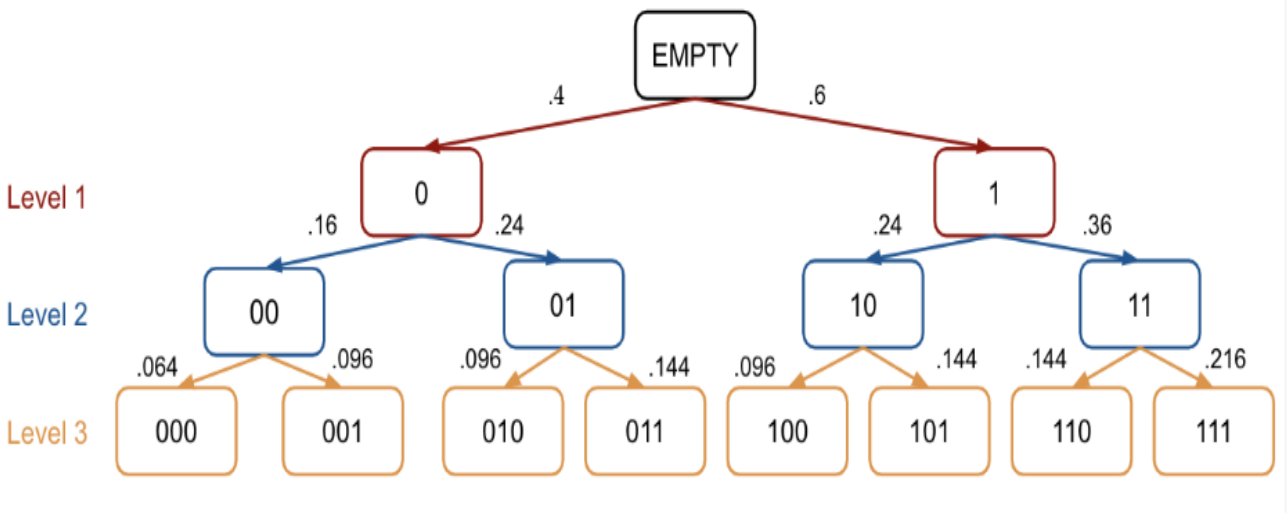
Tree Expansion Problem Statement

- **DEVS process with a corresponding equivalent expansion tree**
  - DEVS := Discrete Event System Specification, Theory of Modeling and Simulation
- **Examples include**
  - Air Battle
  - Baseball team performance
  - Stock trading strategy



Example: Coin Tossing Binary Tree

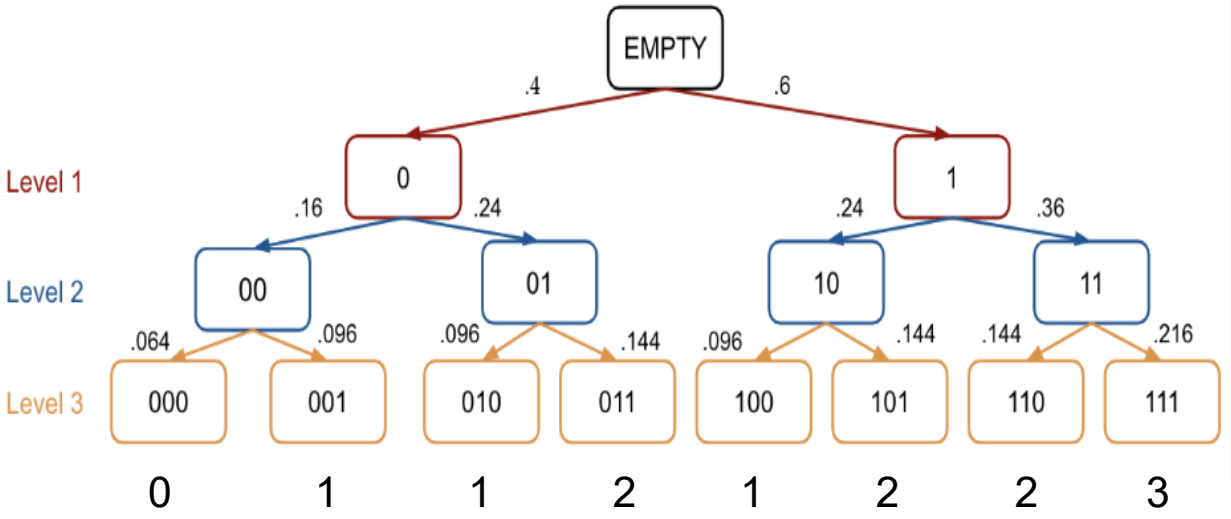
- **Expansion tree with two possible branches at each stage (biased coin)**
  - Nodes are named by the paths taken to reach them
  - Can grow this tree in any manner, e.g., depth first, vs breadth first

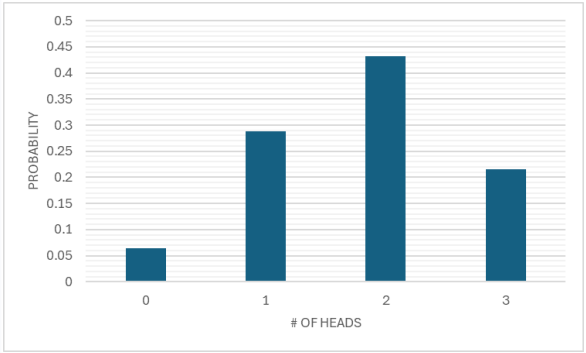


Binary Example Tree Cont'd

- **Now let each of the leaves have an outcome value (macrostate) associated with it**
  - For example, this could be the number of ones in its name (e.g., number of heads)
    - Trajectory (1,0,1) = 2 heads
- **Outcome distribution: all the outcomes with associated path probabilities**
- **Outcome indicators**
  - *Expectation* obtained via accumulating outcomes weighted by path probabilities
  - *Mode* (most likely outcome and its probability), *unlikely* events, tracking different *history dependent* branches (critical events, *chained events*) as ===>

determined by the modeler's objectives ==➔ Experimental Frame

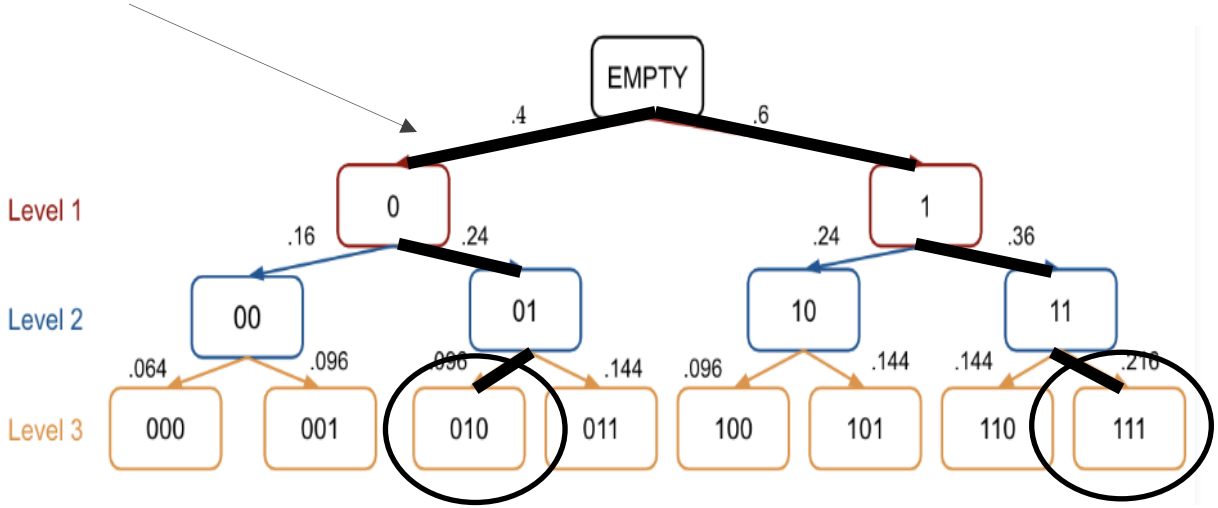




Depth First (Monte Carlo stochastic simulation)

- **Path generated down to last level (sample of the outcome, random variable)**
  - example, three coin tosses are made and the number of ones is obtained as the outcome for that path.
- **Number of samples is determined by the desired statistical significance of the estimate, which is the numerical average of the outcomes produced**

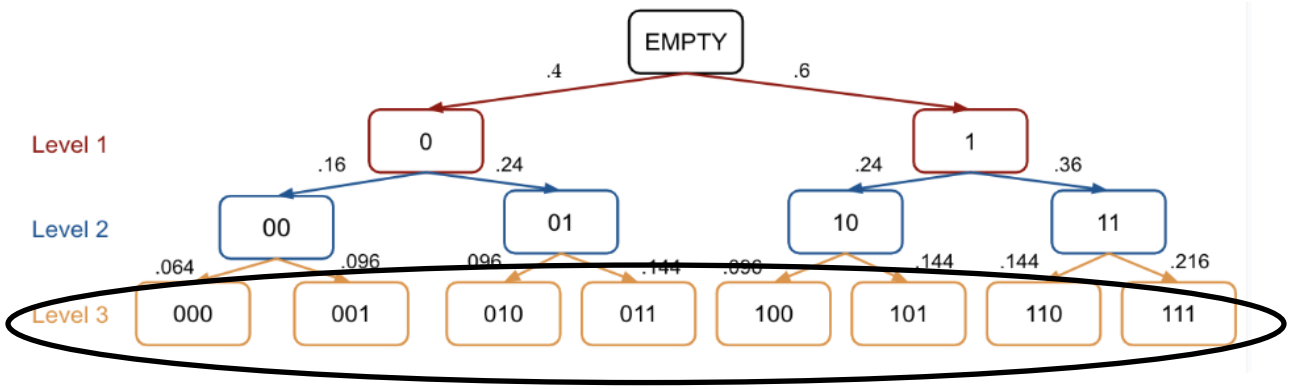
Sample path



- **Inefficient: repeat trajectories, re-simulates the same events, ineffective at studying low probability events**

Breadth First

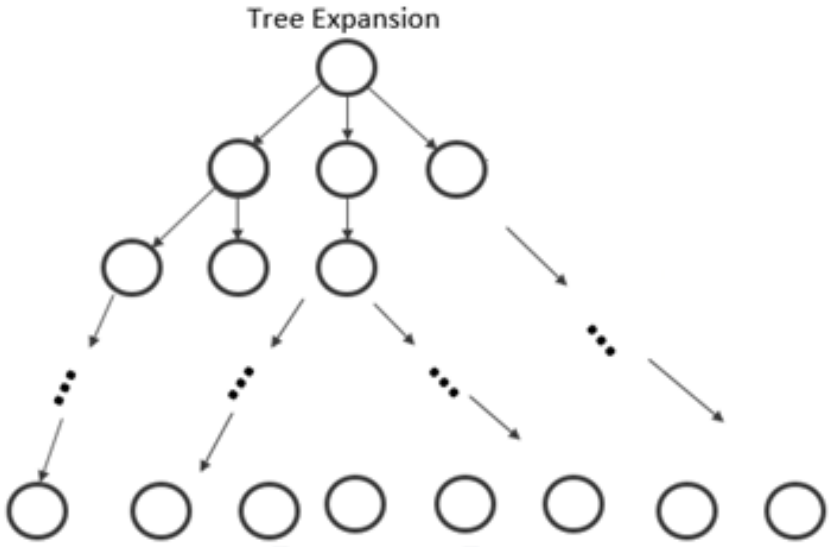
- In a paratemporal simulation this average is obtained by accumulating **all** the *outcomes* weighted by the associated *path probabilities*
  - $E(X) = \sum_{i \in leafs} p_i x_i$
    - $p_i \coloneqq leaf\ path\ probability, x_i\ is$ "total outcome" attributed to leaf
- Yields results that are **exact** at the cost of having to explore potentially **exponentially** large number of paths



Combinatorial explosion limits feasibility

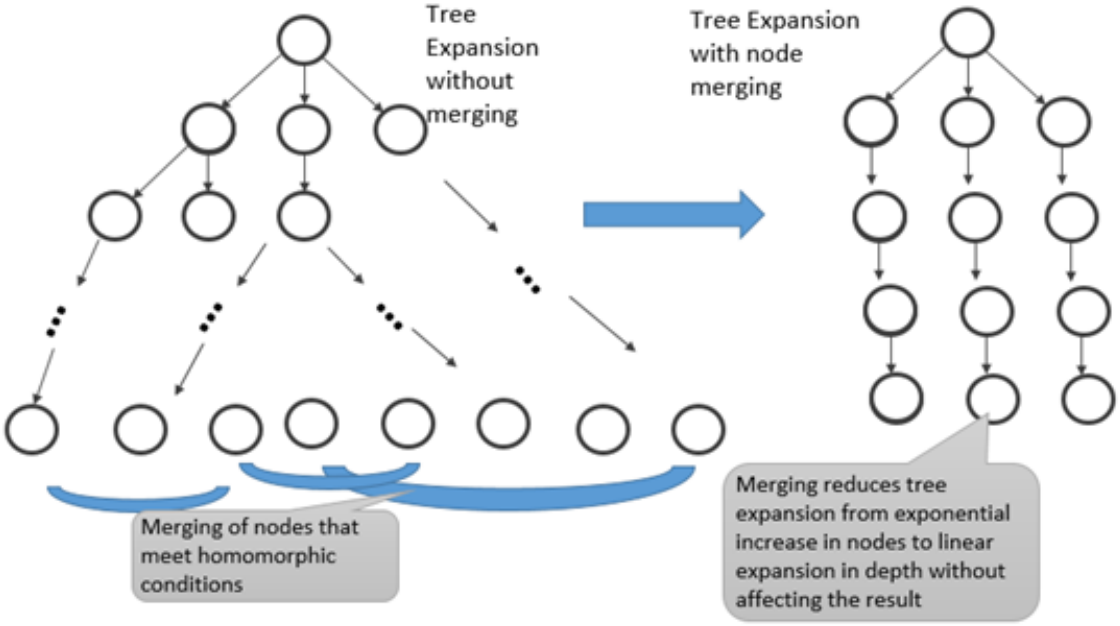
Issues with Handling Expansion Trees

- **Accuracy:**
  - Stochastic sampling simulations require large amounts of time to generate enough trajectories to attain statistical significance
- **Large Search Space:**
  - The event space path are often deep and wide resulting from combinatorial explosion of branching arising from multiple choice points
- **Goal: to leverage Paratemporal simulation method to address these issues**



Node Merging to Control Tree Growth

- **Question: Can we cut down on the size of the tree from 2^d? (d:= $depth$)**
- **Answer: Yes by leveraging the concept of homomorphism**
- **Under the value function already stated 10 and 01 are equivalent and can be merged together to generate one representative node instead**

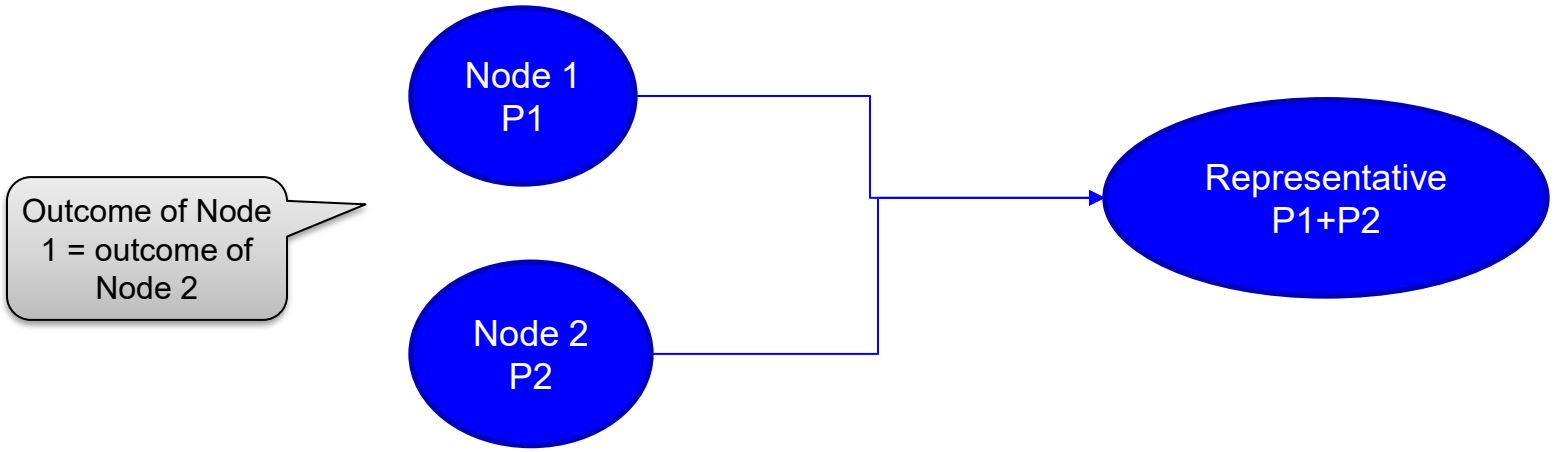


Homomorphism

Homomorphism Accounting

When we merge nodes

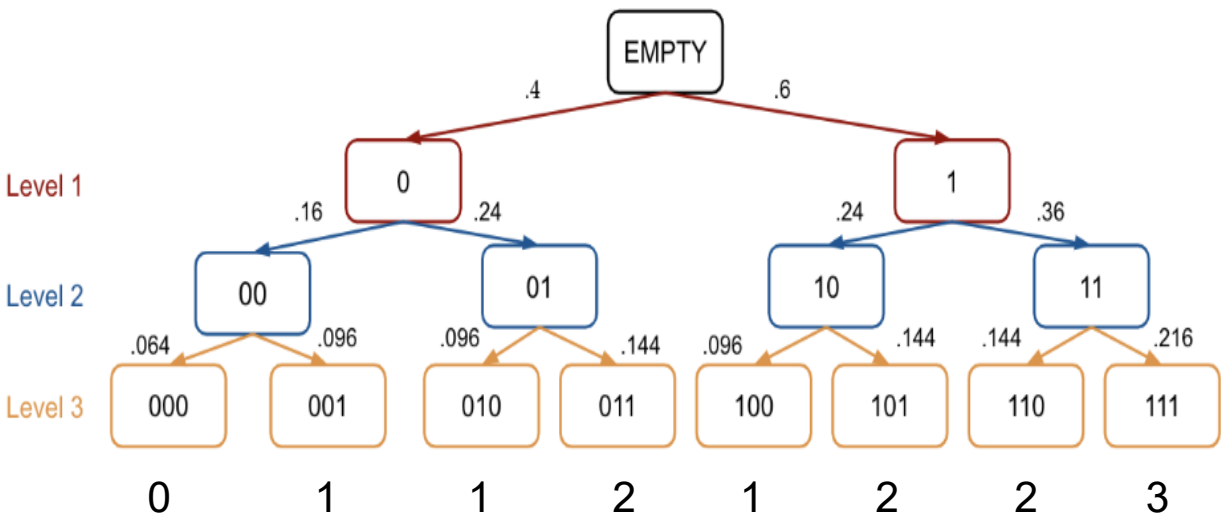
- Account for probability mass of combination, given by $\mathrm{P}_{rep} = \sum_{i \in leafs} p_i$
- Outcome of merged leafs are all equal, so a ***Representative*** takes on that value

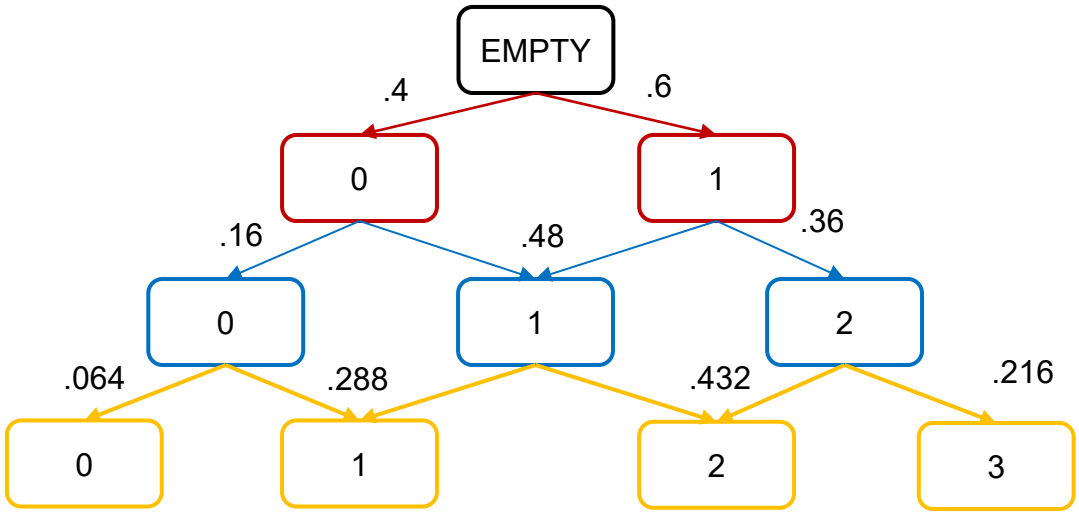


RTSync

Binary Tree Reduction

- **In this example we go from 2^d to d+1 leafs**
  - Exponential ➔ linear



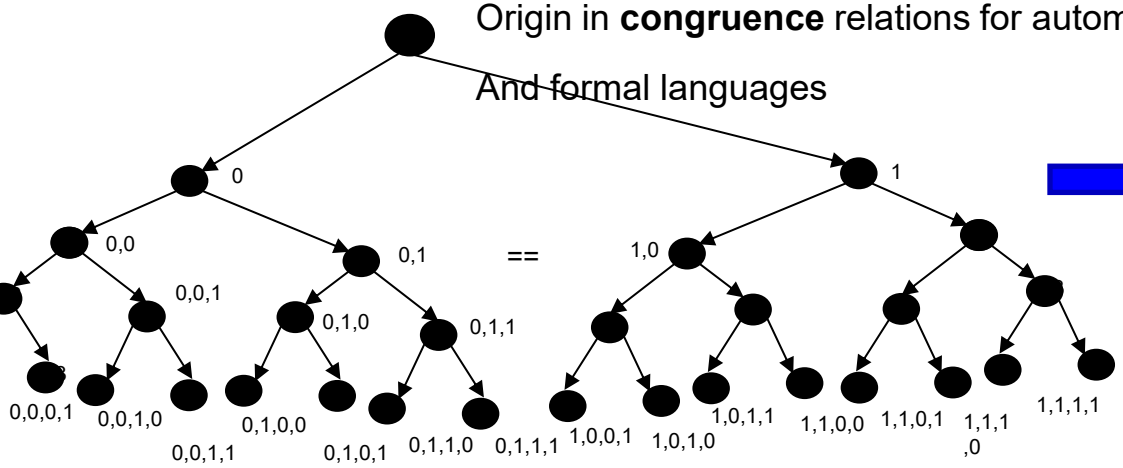


Merging based on # of heads

Homomorphism merging can reduce tree expansion from exponential to polynomial

Origin in **congruence** relations for automata

And formal languages



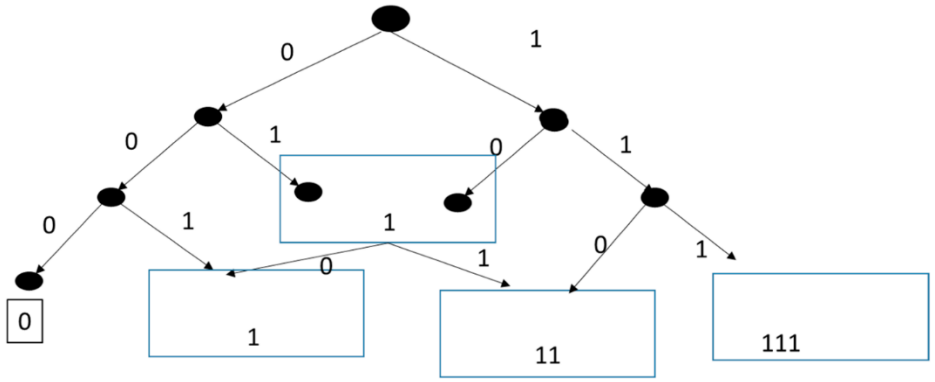
Equivalence Relation and equivalent states:

States are equivalent if:

1) Their outcomes are equal

2) their children are equivalent under the same branching

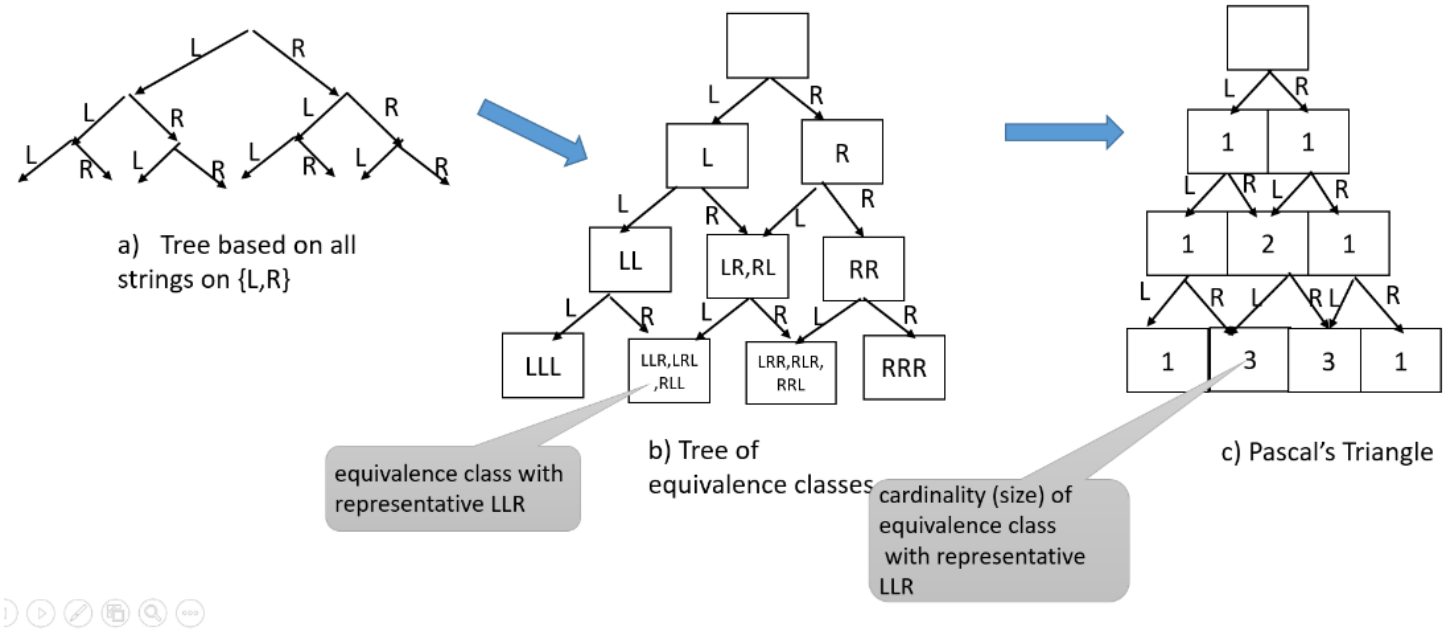
(x1,x2,x3,x4) with the same sum (x1+x2+x3+x4) are grouped into the same equivalence class



RTSync

Homomorphic Tree Merging Underlies Pascal's Triangle

Compute the number of combinations of Ls and Rs in strings of length, n



- **Pascal's computation merges nodes that have the same number of R's and L's**
- **This reduces the computation of all combinations from exponential, to polynomial (square), in the number of places (depth of the tree)**
- **The number of classes at *depth n* is n+1 ➔ tree grows as O (n^2)**

RTSync

ParaDEVS use-cases and Implementation

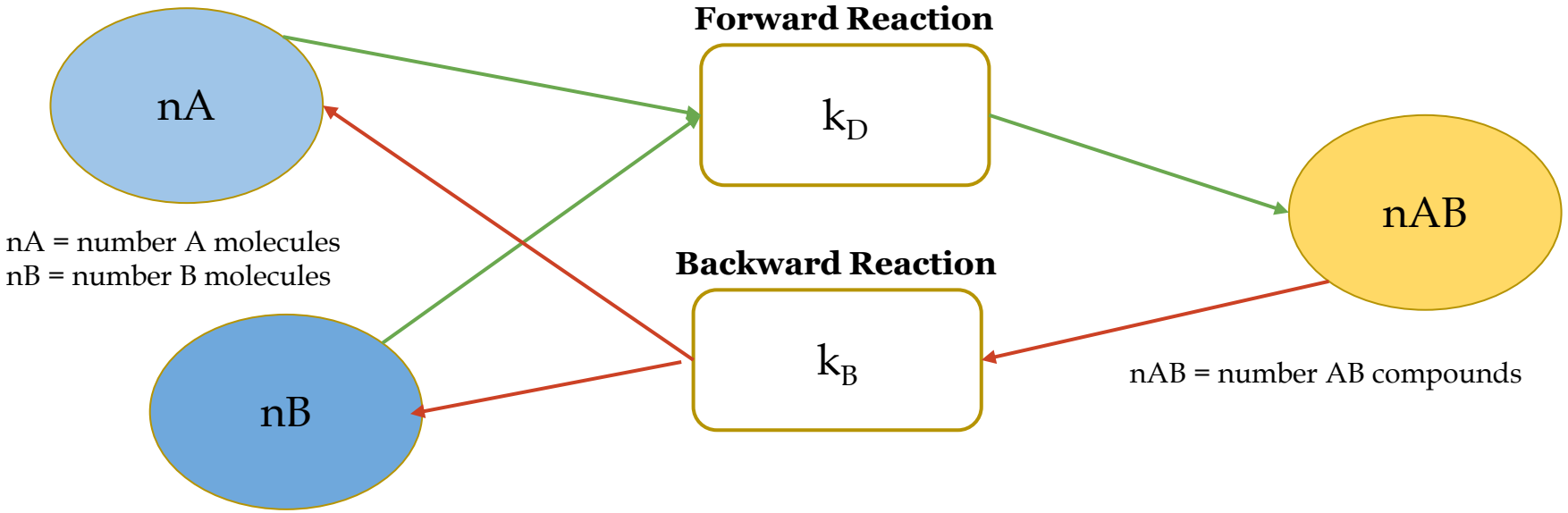
- Stochastic Chemical Reaction Models
- Baseball outcome prediction
- DEVS-based Machine Learning for Financial Investment
- Implementation Architecture

Applications of ParaDEVS

- **Stochastic Chemical Reaction Models**

Example: Dimer Creation Chemical Reaction Model

- **For small numbers of molecules, fluctuations around macro values are large.**
- **The Gillespie algorithm is used fluctuations are important.**
- **Dimer Example:**
  - Molecules A and B react to form a dimer, AB (forward reaction)
  - AB dimer dissociates into molecules A and B (backward reaction)
  - Reaction rate constant of a single A reacting with a single B is k_D
  - Reaction rate constant of an AB dimer breaking up is k_B



Gillespie Algorithm Simulation of Dimer Reaction Model

Converts macro reaction rates to probabilities:

Reaction Rates

$$\text{Rate of dimer formation} = k_D \times nA \times nB$$

$$\text{Rate of dimer dissociation} = k_B \times nAB$$

$$\text{Total rate of reaction} = R_{tot} = (k_D \times nA \times nB) + (k_B \times nAB)$$

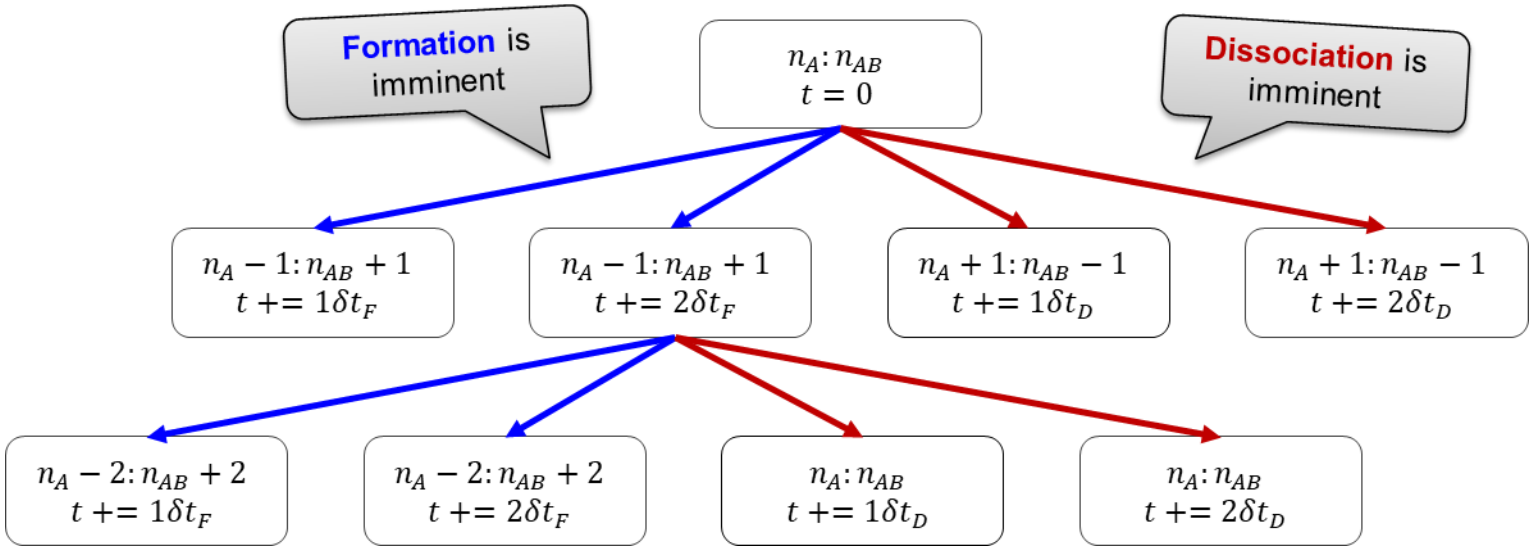
Probability of Reaction Occurring

$$\mathrm{P(A{+}B)} \rightarrow \mathrm{AB} = \frac{k_D \times nA \times nB}{R_{tot}}$$

$$\mathrm{P(AB)} \rightarrow \mathrm{A{+}B} = \frac{k_D \times nAB}{R_{tot}} = 1 - \mathrm{P(A{+}B)} \rightarrow \mathrm{AB}$$

Tree Expansion for Chemical Reaction Example

- **At each event, the imminent reaction is either Formation or Dissociation**
- **Branch in corresponding direction, with time expansion (via discretization)**



- Keep track of state (NA,NAB) and accumulated probabilities
- Get distribution at any depth {averages of NA and NAB, and min, max)
- Need to merge to get fast results

Node Generation and Equivalence

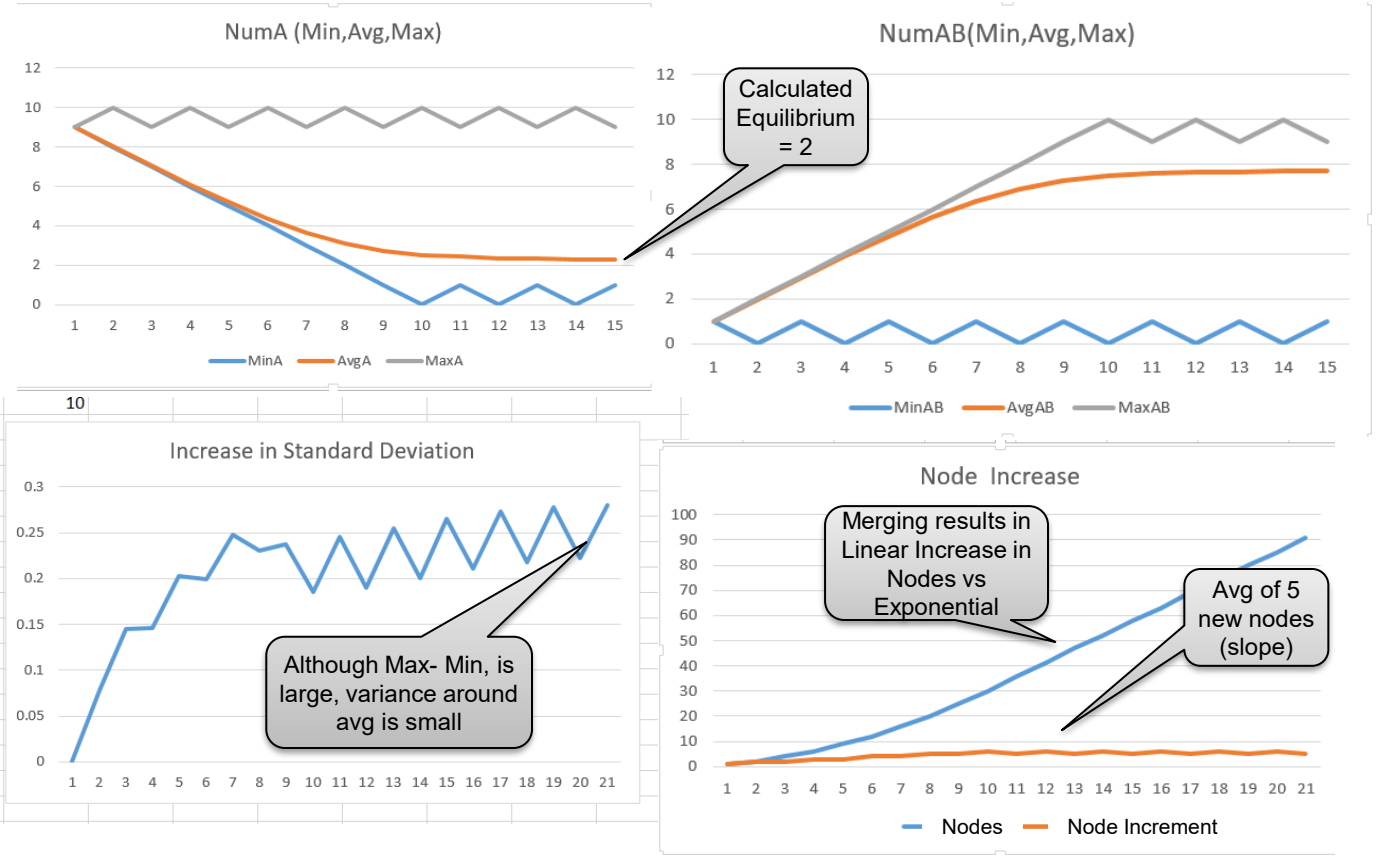
Binary tree expansion:

- **One branch is forward reaction**
  - nA = nA – 1 (nB = nB – 1)
  - nAB = nAB + 1
- **One branch is backward reaction**
  - nA = nA + 1 (nB = nB + 1)
  - nAB = nAB – 1
- **Equivalence:**
  - Merge nodes if they have the same states, nA=n'A, …

(note NA+NB = constant under conservation of matter)

Sample Results: Molecule numbers, Simulation runtimes

Initial NA = NB = 10, NAB = 0, Values at t= 20, NA = 2.28 vs 2 calculated equilibrium
Initial NA = NB = 2, NAB = 10-2, Results = nA = 2.28 vs 2 calculated equilibrium



ParaDEVS Implementation

- Baseball Simulation Example
- Illustrates ParaDEVS Simulation Architecture
- Parallel Simulation of Tree Expansions

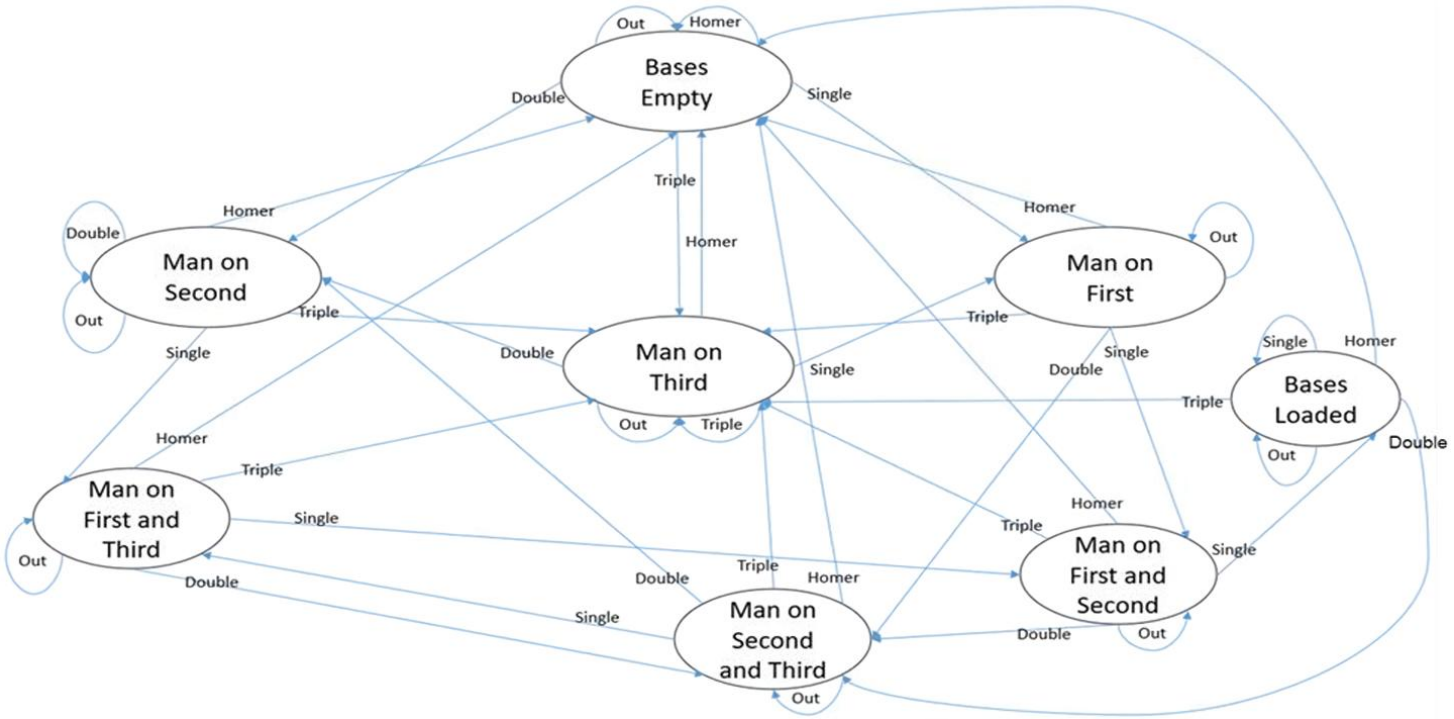
Implementation of ParaDEVS

- **MS4 Me Two-Level Approach**

MS4 Me ParaDEVS Implementation

- **Problem:** Construct ParaDEVS model for Baseball Game
- **Approach: Simulate stochastic game given one team's batter hit avgs**
- **Goal:** Compute expected runs given a batting order
- **Methods Used:** No Merging, Homomorphic Merging, Incremental Merging

Baseball Model

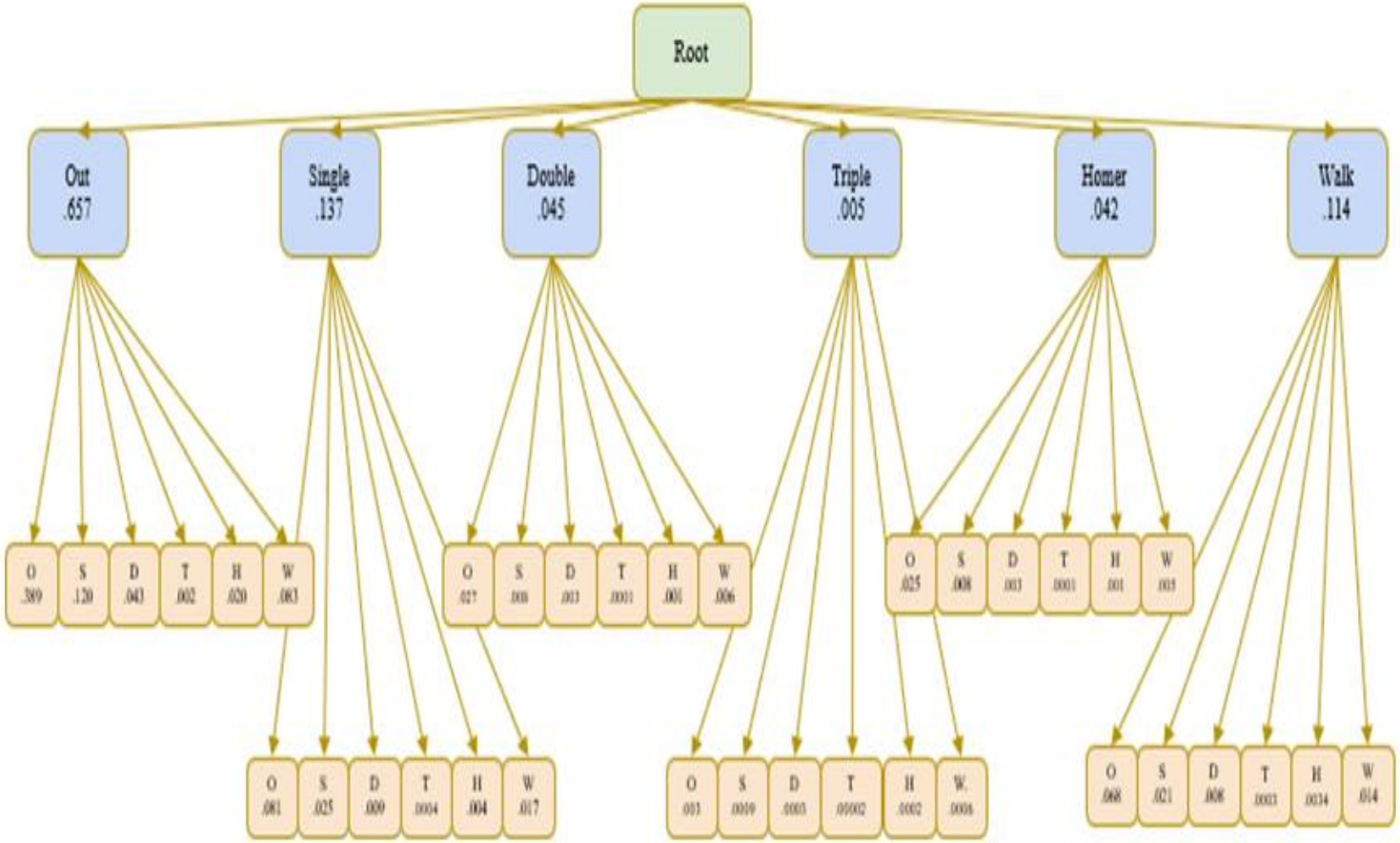


Baseball Example

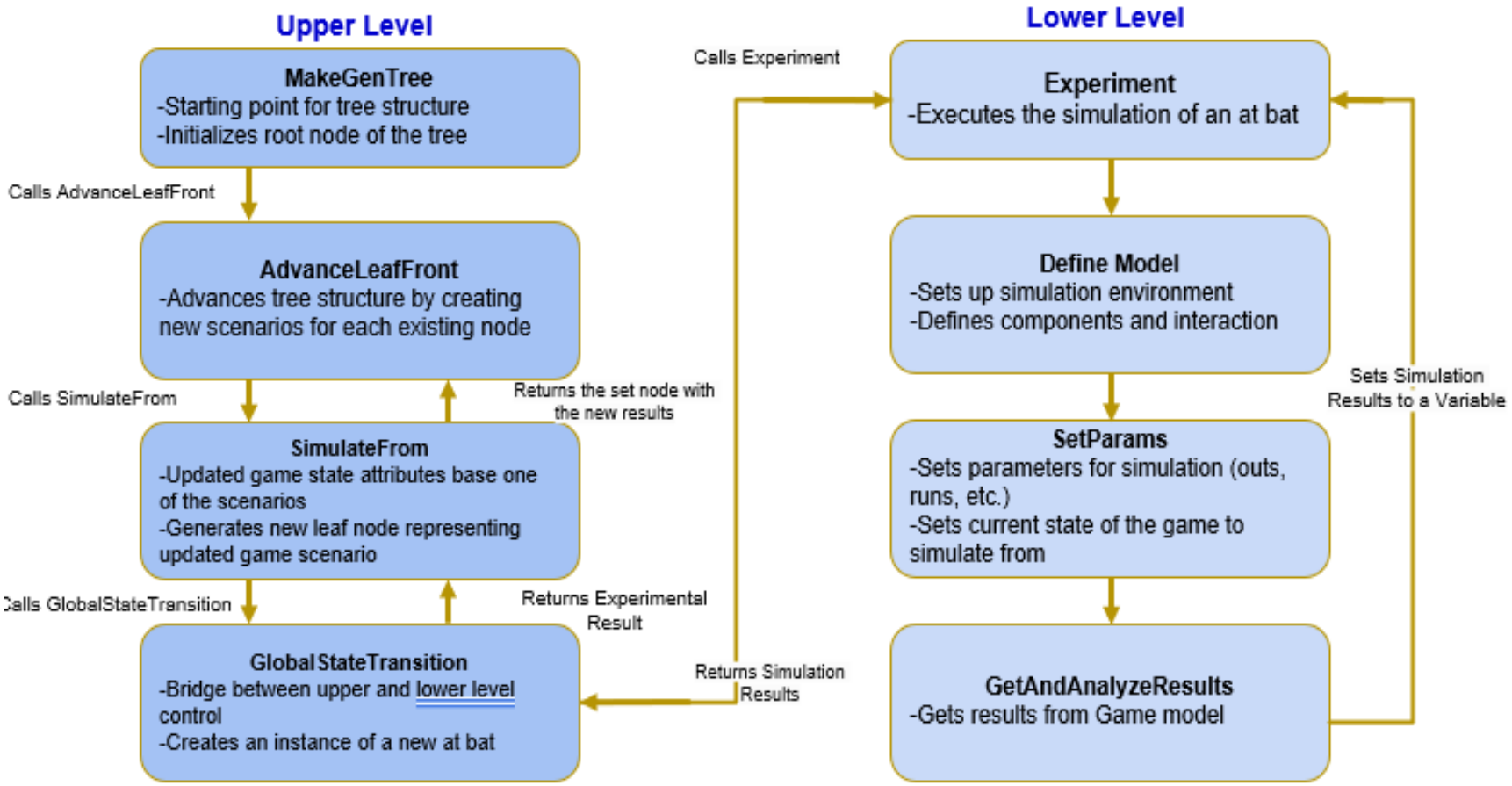
- **Nodes track successive**
  - model states
  - outs,
  - inning,
  - and runs scored

Player 1 at bat

Player 2 at bat

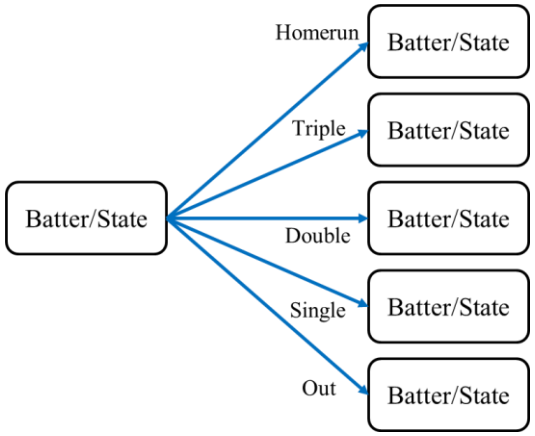


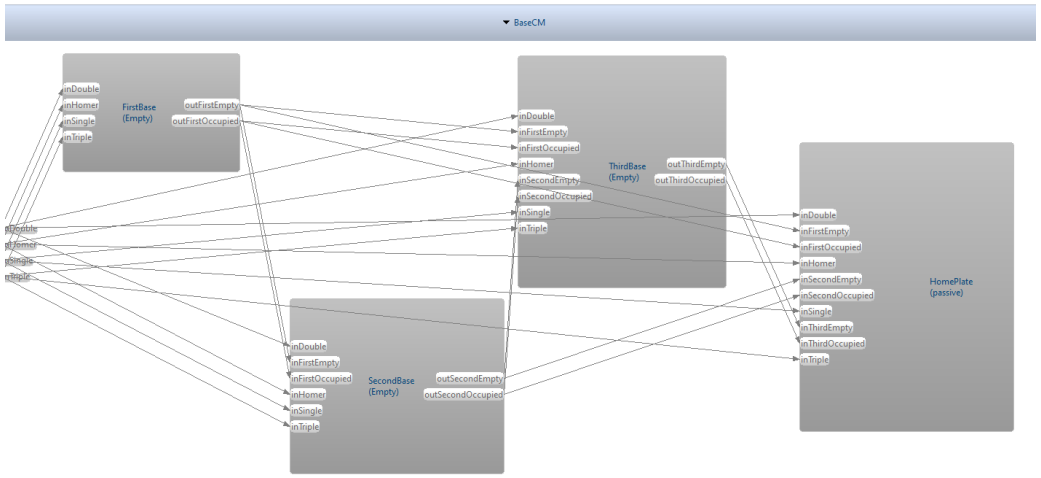
ParaDEVS Simulator Architecture



Integration of Baseball Game with AdvanceLeafFront

- **Each node represents a batter along with the state of the game**
  - State of the game = {runners on base, runs scored, outs, etc.}
  - Branching corresponds to random variable outcome of the batter
  - After branching, a new simulator is initialized and started with the state of the game following previous batter outcome.

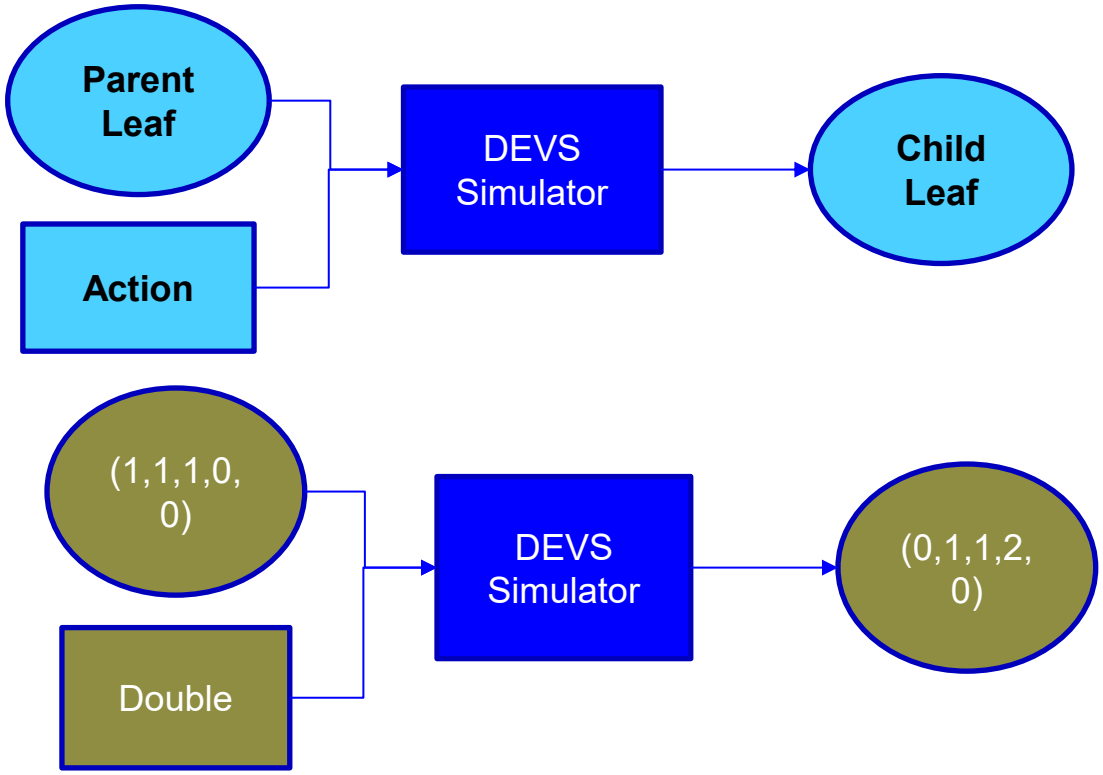




Model for the state of a game

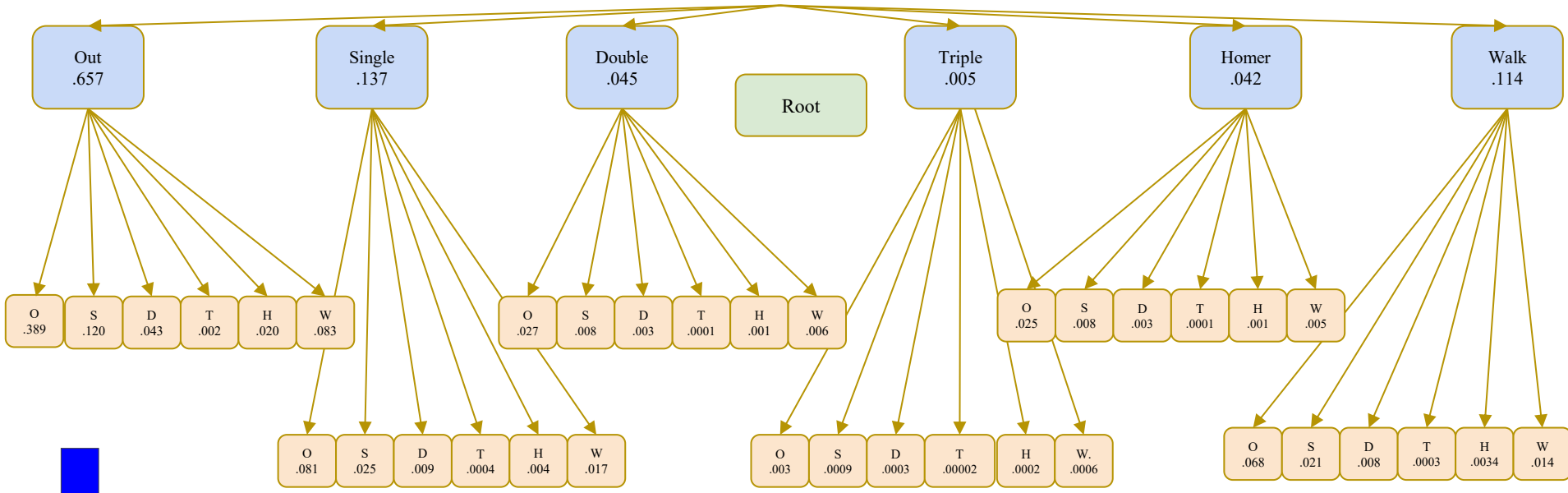
Simulator Scheme

- Nodes track successive model states, outs, inning, and runs scored
- Nodes are expanded and simulated along their branches



Target Computation: $Expected\ Number\ of\ Runs = \sum_{i \in leafs} p_i runs_i$

Baseball Tree Expansion



	Out	Single	Double	Triple	Homer	Walk
Player 1	.657	.137	.045	.005	.042	.114
Player 2	.593	.182	.066	.003	.030	.126

Tree expansion continues down to 3 outs and 9 innings

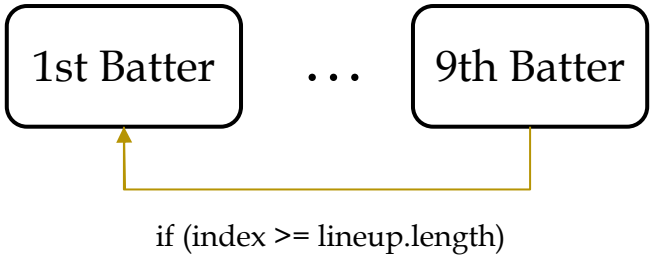
ParaDEVS-Baseball

- **Utilized paratemporal simulation methods to simulate baseball game outcomes**
- **Each state has the following parameters:**
  - Hit type: out, single, double, triple, homerun, walk
  - Current batter
  - Current number of runs
  - Current number of outs
  - Base configuration: man on first, second and third, bases loaded, etc.
  - Probability of current players hit outcome
  - Probability of node occurring
- **9 players in lineup, so 9 different batters with their individual hit probabilities**

Batter	Out	Single	Double	Triple	Homer	Walk
Will Smith	0.6574	0.1367	0.0450	0.0052	0.0415	0.1142
Freddie Freeman	0.5932	0.1822	0.0664	0.0028	0.0297	0.1257
Gavin Lux	0.6539	0.1762	0.0425	0.0149	0.0127	0.0998
Trea Turner	0.6572	0.1841	0.0552	0.0057	0.0297	0.068
Max Muncy	0.6708	0.0832	0.0389	0.0018	0.0372	0.1681
Chris Taylor	0.6960	0.1123	0.0551	0.0066	0.0220	0.1079
Cody Bellinger	0.7345	0.1036	0.0491	0.0055	0.0345	0.0727
Mookie Betts	0.6604	0.1189	0.0626	0.0047	0.0548	0.0986
Justin Turner	0.6504	0.1523	0.0677	0.0000	0.0244	0.1053

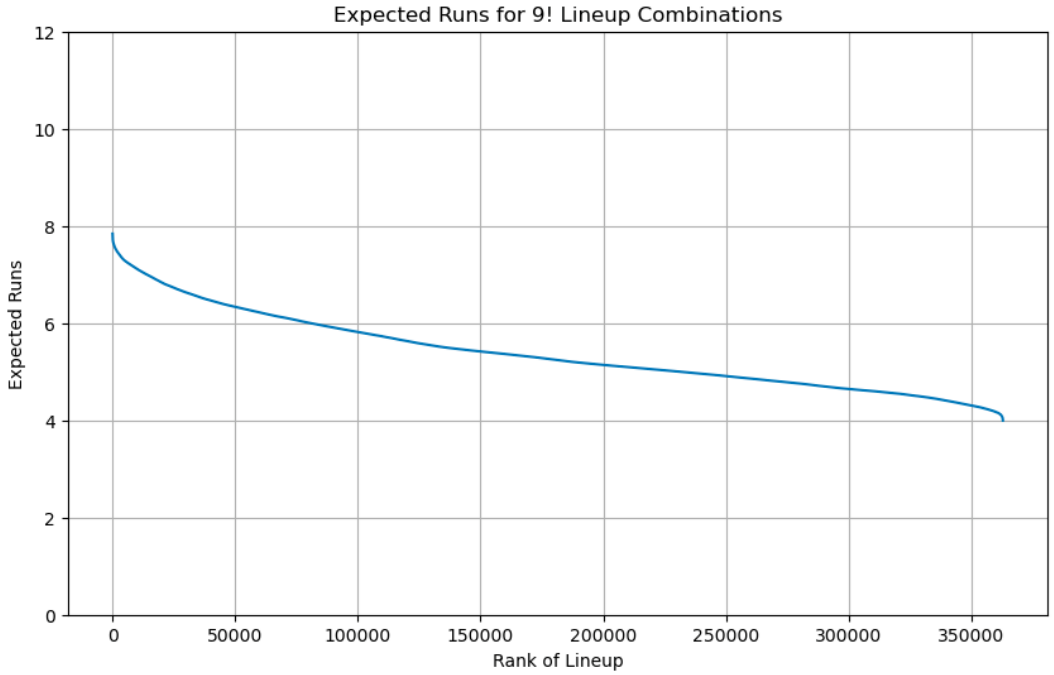
Full Game Simulation

- Incorporated inning aspect of nodes to specify the current inning in which the node is created
- Nodes are terminated when 9 innings are reached or when a node's probability of occurring is less then .000001 = 1 x 10-5
- Once the last batter hits, the lineup is sent back to index 0 (first batter)
- When 3 outs is reached, the inning advanced by 1, and the bases are cleared



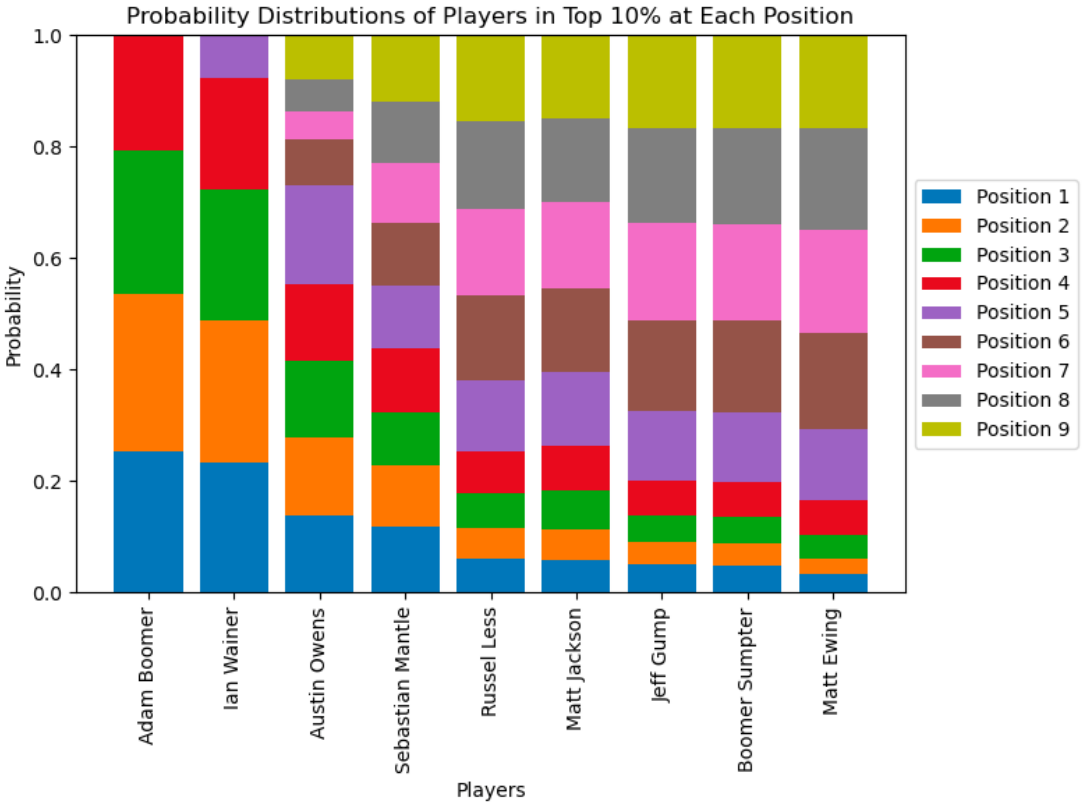
Computing all 9! Team Lineup's Expected Run Production

- Graph displays each individual lineups expected runs ranked from best to worst
- Best Lineup: 7.8397
- Worst Lineup: 3.9969
- Best lineup is 1.9608 times better than the worst lineup
- Computation time: 25227988 ms = 7 hrs 0 min 27 s



Distribution of Players Bar Chart

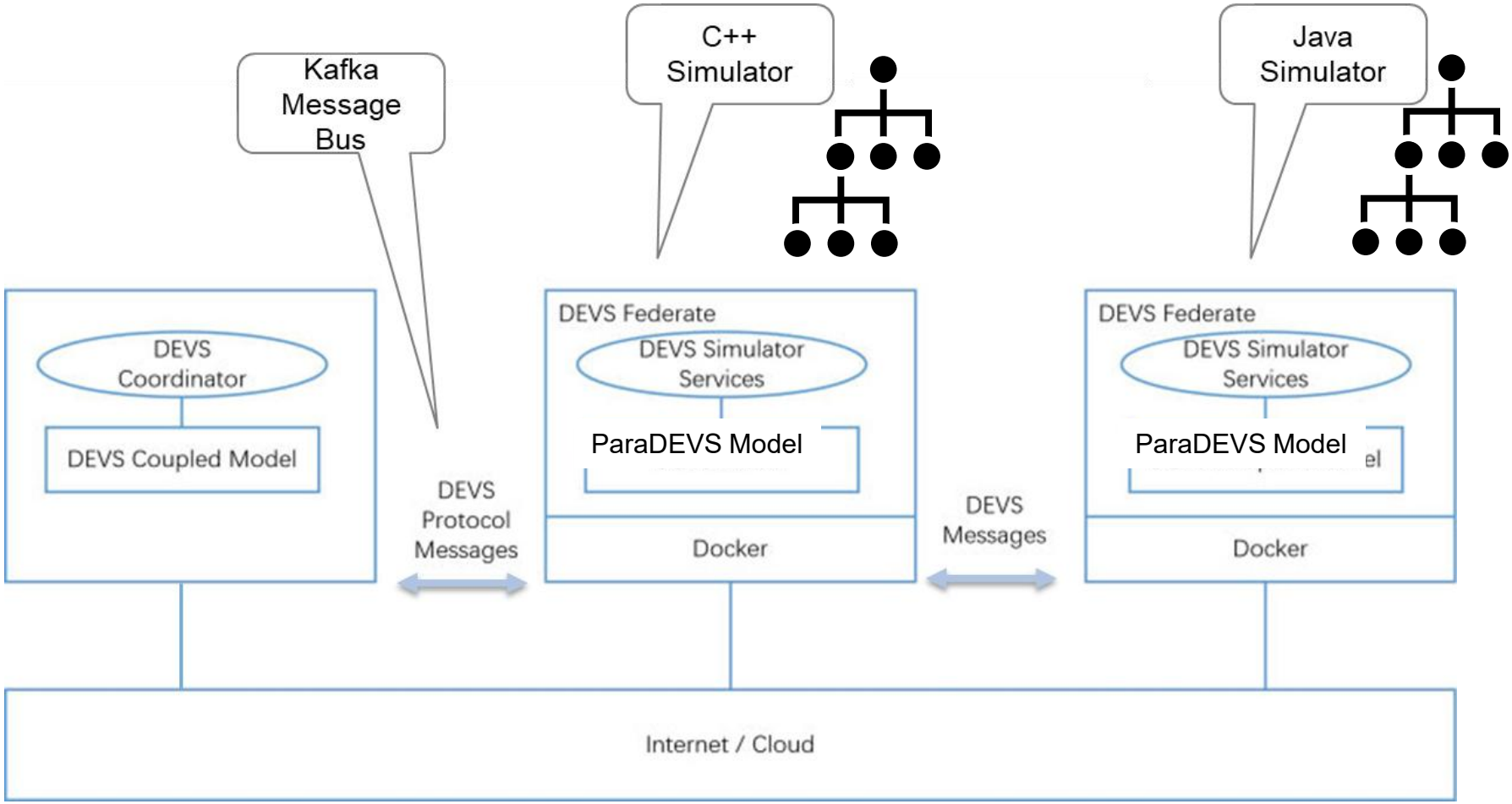
- Bars show the **probability** that the respective player is in each lineup position (color) for the top 10% of lineups
- **Probability** is measured by number of times player occurs in given lineup position
- ***Better*** hitters have larger blue, orange, green, and red bars which represent positions at the beginning of the lineup



Implementation of ParaDEVS

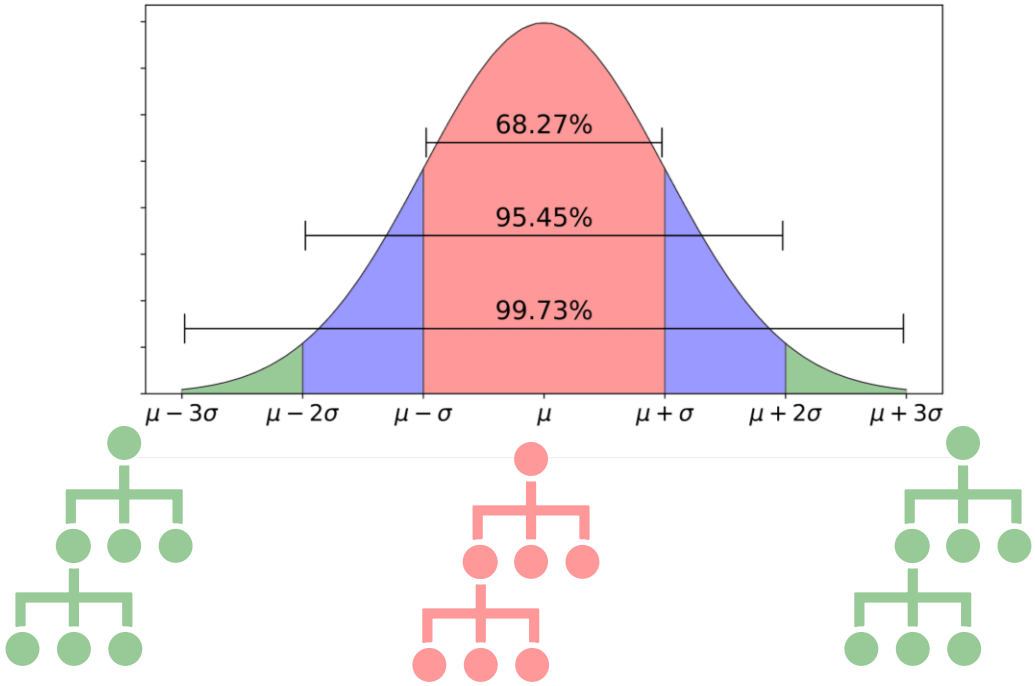
- **Parallel Simulation Architecture via DEVS Streaming Framework**

DEVS Streaming Framework for Interoperable Distributed Simulation



Parallel Simulation of Multiple Tree Expansions

- **Ensemble of sub-trees focusing on selected regions of scenario space with high resolution**
  - Mode: take highest probability branches
  - Mean: span greater range with less resolution
  - low probability events
  - Goal: get earlier results sooner

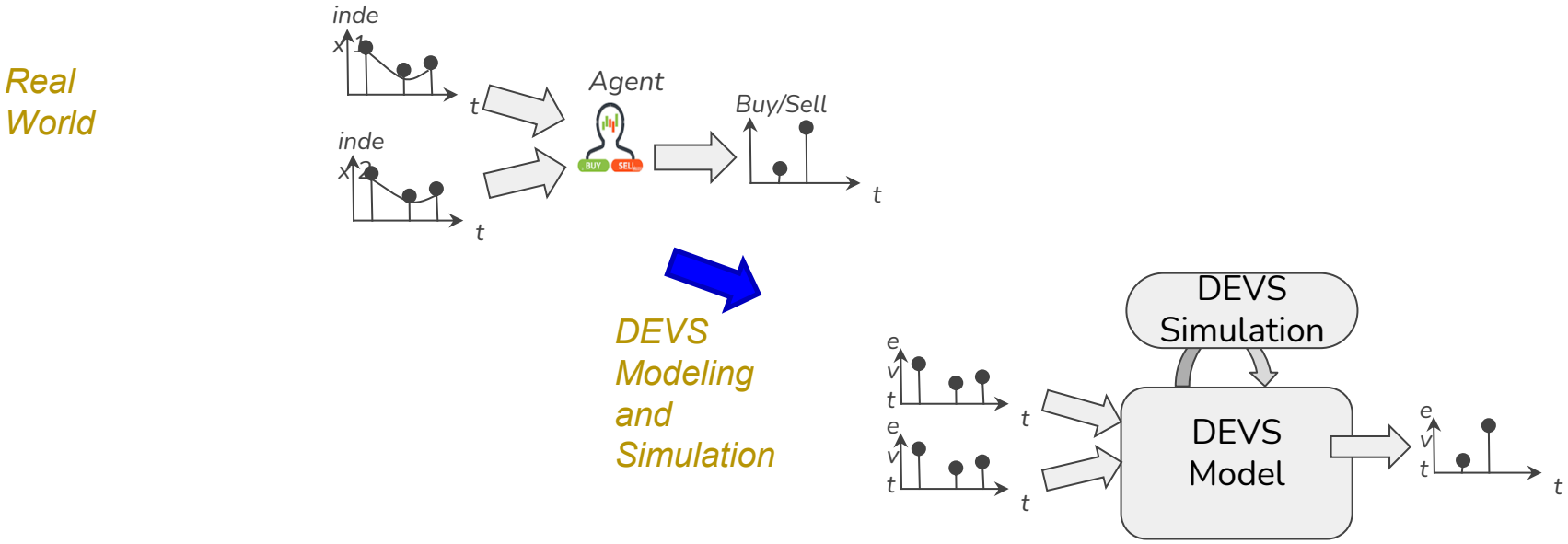


Applications of ParaDEVS

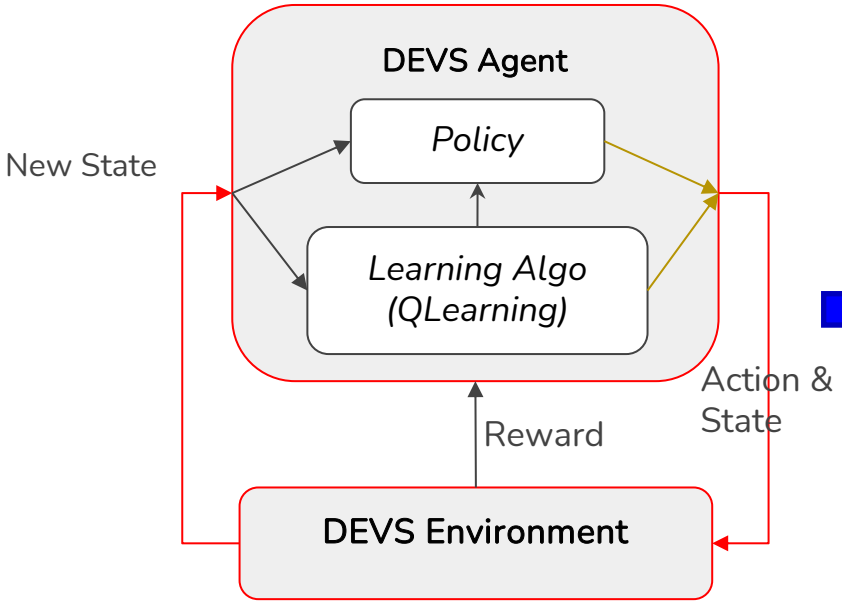
- **DEVS-Based Machine Learning for Financial Investment**

DEVS-Based Machine Learning for Financial Investment

From Trader behavior (Agent) to DEVS simulation model:



DEVS Machine Learning Simulation Approach



Q-Table

State-Action Mapping Table				
	a_0	a_1	a_2	a_3
s_0	$Q(s_0,a_0)$	$Q(s_0,a_1)$	$Q(s_0,a_2)$	$Q(s_0,a_3)$
s_1	$Q(s_1,a_0)$	$Q(s_1,a_1)$	$Q(s_1,a_2)$	$Q(s_1,a_3)$
...	...	...	...	...

DEVS-RL Output : Action policy that tells an agent what to do given a state s

- Given some state “s” , the policy tells us what action “a” to take
- We use Q table to devise the policy do this

Observations

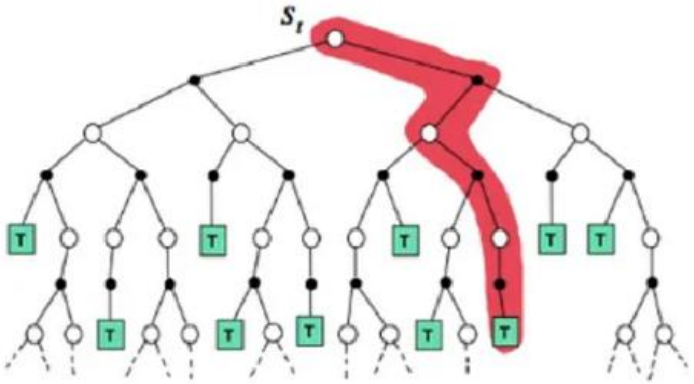
- DEVS-RL produces an approximation of Q so our policy can also be seen as an approximate policy
- We use ParaDEVS to hedge our policy

Barbieri Emanuele. Discrete Event Modeling and Simulation of Large Markov Decision Process : Application to the Leverage Effects in Financial Asset Optimization Processes. Performance [cs.PF]. Université Pascal Paoli, 2023. En

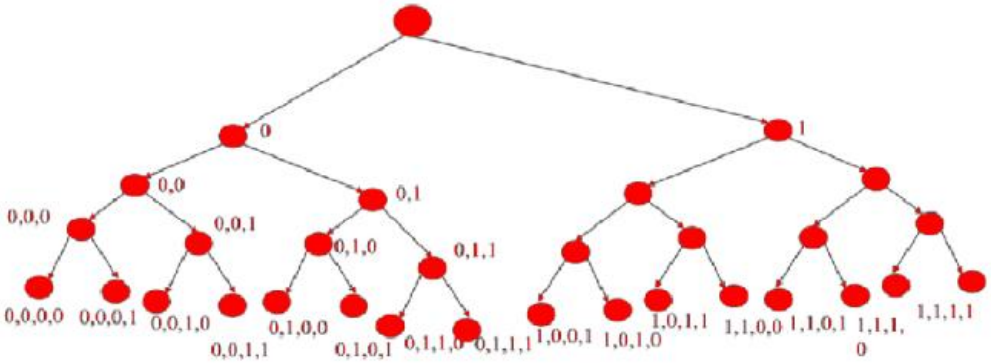
ParaDEVS Financial Application: ParaFinance

- Para DEVS: Stochastic policy does in fact need to consider tree as it's doing multiple actions with non zero probabilities
- Evaluation Tree

Greedy Policy Tree Path



Stochastic Policy Tree Path



Experimental Study

We motivate the following test to verify how the current model performs against down market conditions for four separate NASDAQ stocks as well as it's ability to handle bounce back

Period of Time: April 2nd - April 14

Stocks:

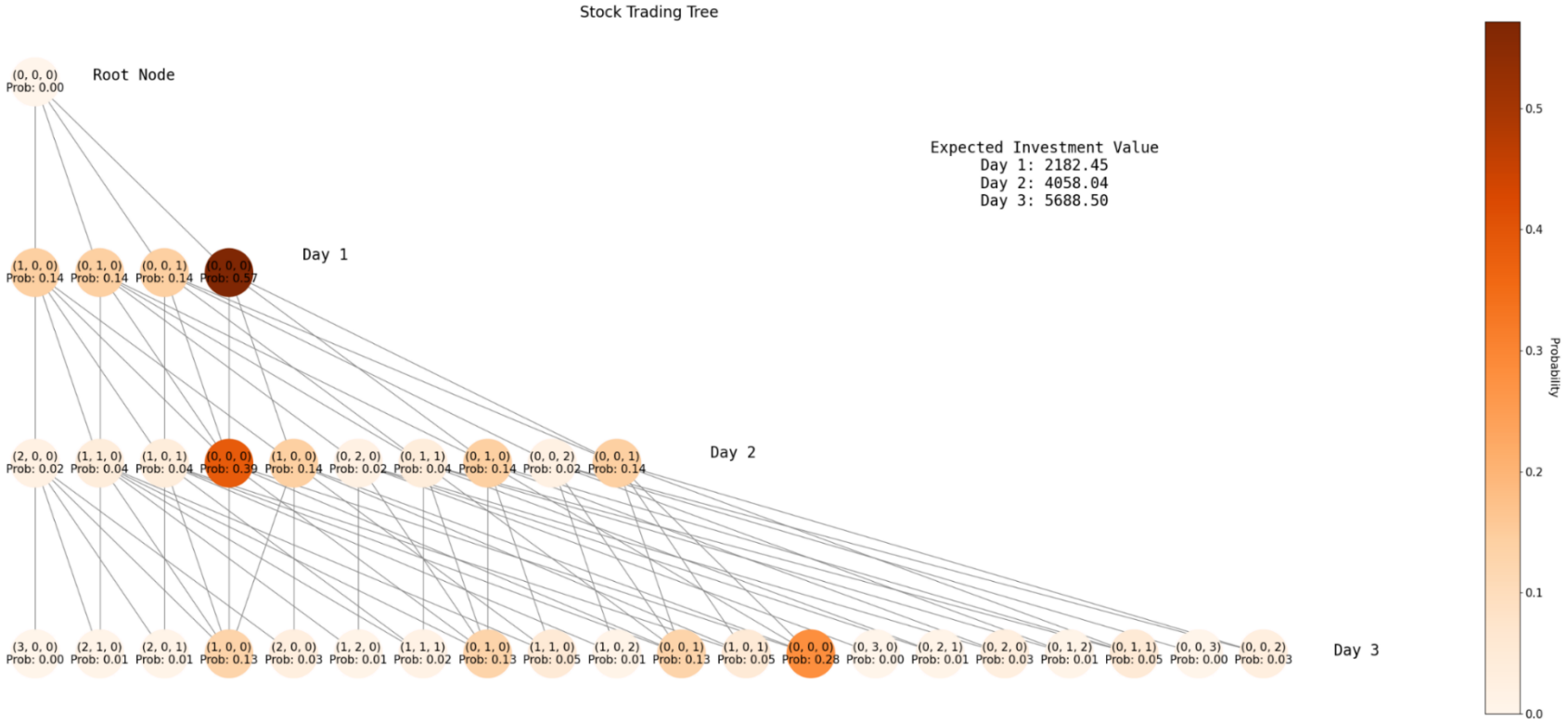
1. Apple
2. GLD
3. SPDR S&P 500
4. WMT

Incremental Homomorphic Merging: Single stock trading

- Actions: Buy,Sell,Hold 1 stock
- Probabilities for Actions derived from Q-table (RL Machine Learning)
- Expand tree to estimate investment value at the end of 3 days

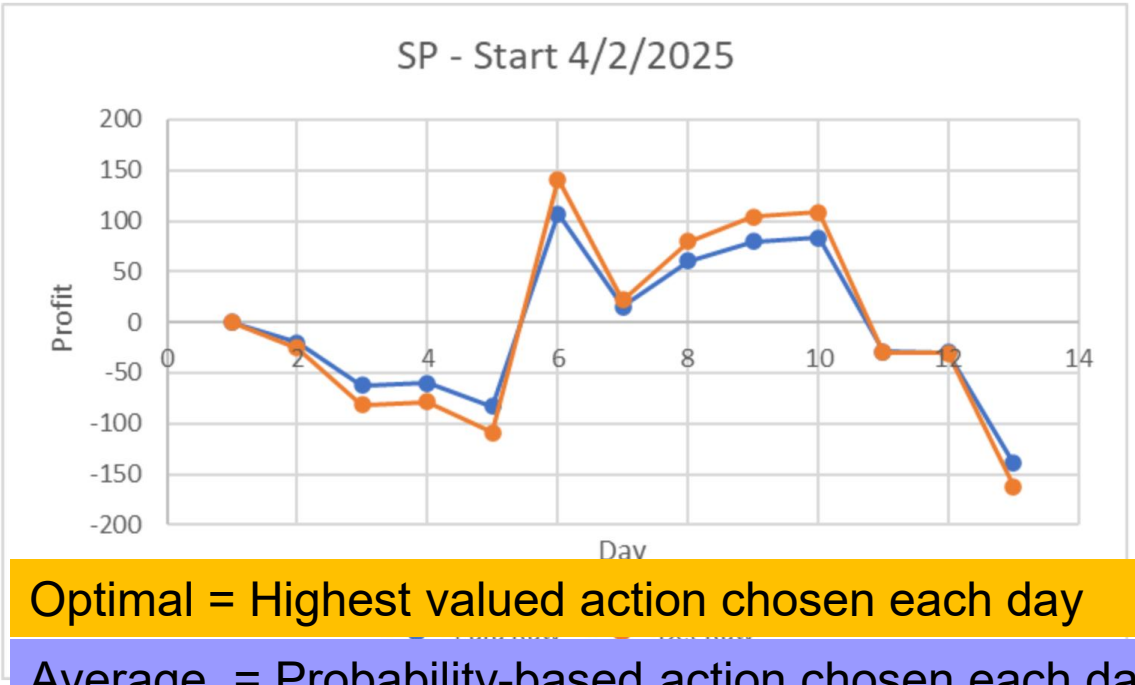
Q-Table defining the transition probabilities

	Buy	Sell	Hold
0	0.06652	0.907657	0.025823
1	0.008867	0.013949	0.977185
2	0.045624	0.037561	0.916815
3	0.060281	0.018281	0.921438
4	0.923601	0.023231	0.053168
5	0.968004	0.021886	0.01011



Example: S&P Investment Profit Optimal and Average Over time

SP Profit

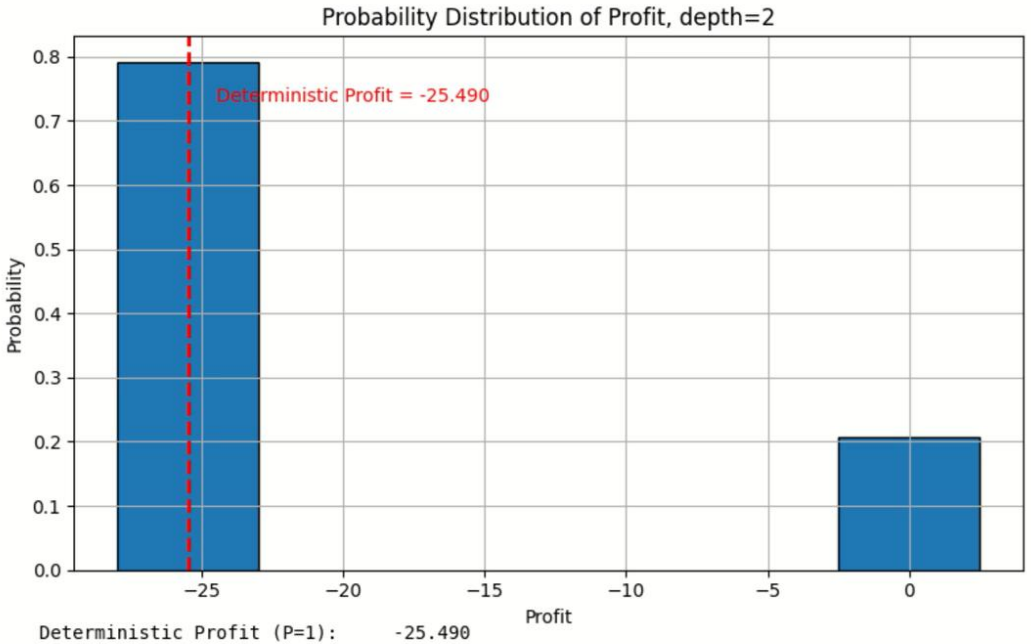


Optimal = Highest valued action chosen each day

Average = Probability-based action chosen each day

Evolution of S&P Profit Probability Distribution over time

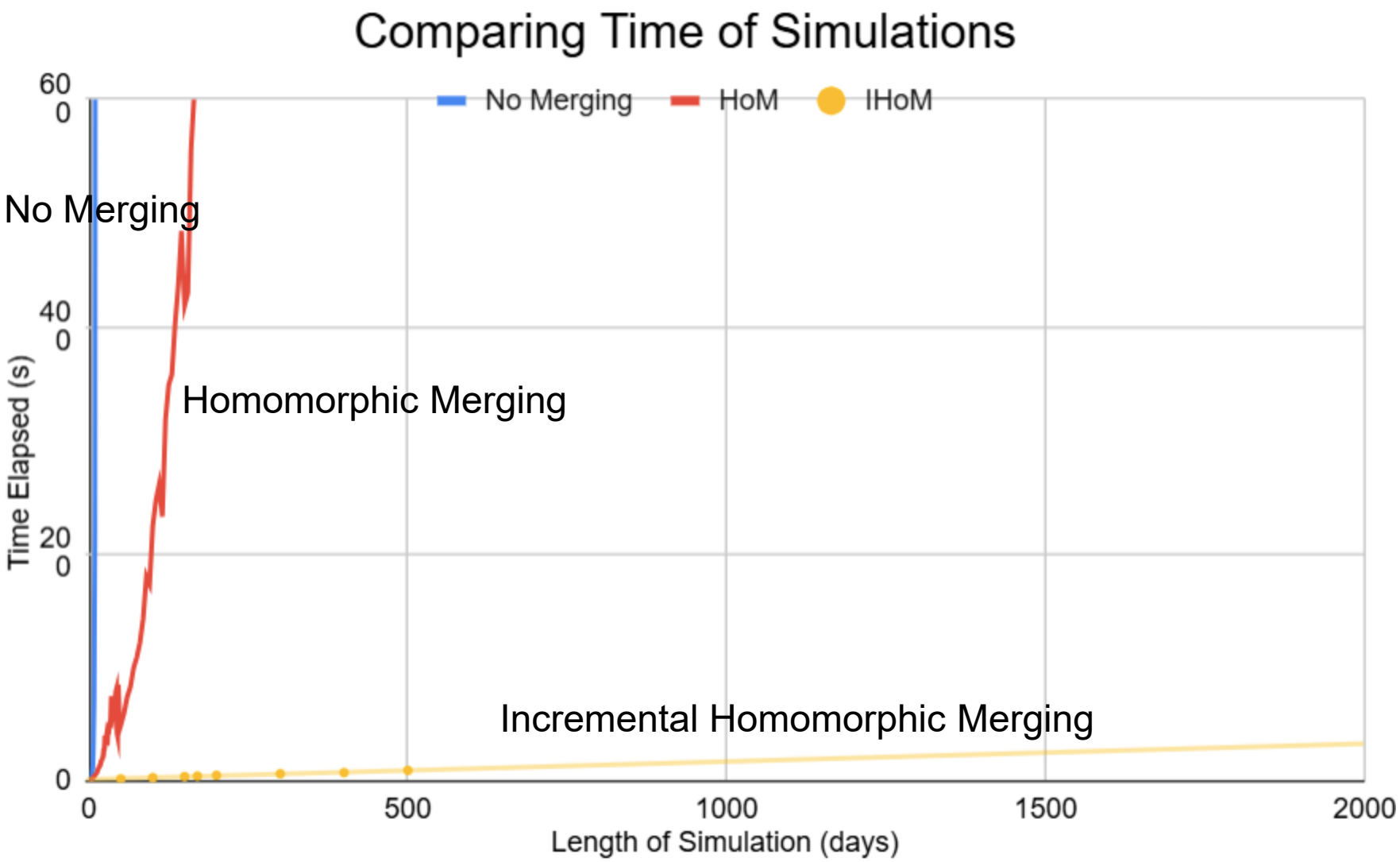
SP PMF (Q′ → Q, Max over 13 days /= Max over 1-12)




```
Deterministic Profit (P=1):     -25.490
Weighted Mean:                  -20.192
Standard Deviation:             10.343
Mode (Most Probable Profit):    -25.490
Max Profit:                     -0.000 (P=0.208)
Min Profit:                     -25.490 (P=0.792)
```

RTSync

ParaDEVS Simulation Results



Summary: Comparison with Monte Carlo Simulation

Feature	ParaDEVS	Monte Carlo Simulation
Speed	Faster (minutes/hours)	Slower (can take years for com plex models)
Accuracy	Higher (tree expansion method)	Lower (random sampling introd uces variability)
Flexibility	Can adjust model structure dynamically	Requires restarting for changes
Computational Efficiency	Optimized for large-scale simulations	Can be resource-intensive
Integration with AI	Seamlessly integrates with AI & ML	Limited AI integration