Midterm CPTS 515

Lubah Nelson

October 28, 2024

Question 1

$M_1(G)$ is calculated by aggregating the weighted nondeterministic choices $d'(u)$ across all nodes and normalizing this total by the number of states in the deterministic representation of G.

$$M_1(G) = \frac{1}{|S_D|} \sum_{S \in S_D} \sum_{u \in S} d'(u)$$

where:

- $|S_D|$ is the total number of states in the DFA constructed from G, - S represents each subset of nodes in S_D (each DFA state), - u is a node within the subset S, - $d'(u) = \text{Centrality}(u) \times d(u)$ is the weighted nondeterminism degree of node u, - $d(u) = \sum_{c \in C} \max(0, |\delta(u, c)| - 1)$ is the nondeterminism degree of u, which counts the "excess" transitions for each color c at node u.

Justification

The metric $M_1(G)$ effectively captures nondeterminism in G by combining local and global structural aspects. Locally, $d(u)$ measures the "excess" choices for each node under each color, reflecting nondeterministic options directly. We then weight $d(u)$ by each node's centrality, ensuring that nondeterministic choices at structurally significant nodes influence the metric more heavily. Converting G to a DFA enables $M_1(G)$ to account for all reachable node subsets, capturing global nondeterminism without dependency on specific walks. Finally, normalizing by DFA states $|S_D|$ ensures scale invariance, making $M_1(G)$ comparable across graphs of different sizes.

So the reason this metric is not a function of the walk, but the graph is because it measures nondeterminism across the entire graph by evaluating all possible node subsets in the DFA representation, capturing global nondeterministic behavior without reliance on specific paths. By aggregating weighted nondeterministic choices for each node, $M_1(G)$ reflects the graph's overall structure and ambiguity, independent of individual walks.

Constructing $M_1(G)$

To compute $M_1(G)$, we begin by assigning a centrality score to each node using the PageRank algorithm, which captures the influence or importance of each node within the structure of G. This centrality weighting ensures that nondeterministic choices at more central, structurally significant nodes contribute more to the metric, reflecting their higher impact on the graph's overall behavior.

Centrality, in this context, acts as a scaling factor for nondeterministic choices at each node. Intuitively, nondeterminism at a more central node affects the graph's overall behavior more significantly than at a peripheral node.

Mathematically, we define the PageRank centrality for each node u iteratively:

1. **Initialize**: Each node u is assigned an equal probability score, typically $\frac{1}{|V|}$, where $|V|$ is the total number of nodes.
2. **Update**: In each iteration, update each node u's score by distributing its current score across its outgoing edges to adjacent nodes, applying a damping factor $d = 0.85$ to prevent score accumulation.
3. **Convergence**: Continue this iterative update until centrality scores stabilize, yielding Centrality(u) for each node.

These centrality values Centrality(u) are then used to weight the nondeterministic choices at each node, allowing $M_1(G)$ to reflect both the local nondeterministic choices and the global importance of each node within the graph.

Step 2: Calculate the Nondeterminism Degree $d(u)$ for Each Node

After computing the centrality scores, we calculate the nondeterminism degree $d(u)$ for each node u to quantify the degree of nondeterministic behavior at that node. This degree measures the excess choices available at each node, counting additional outgoing edges under each color. This is achieved by iterating through the graph and examining the outgoing edges of each node.

Mathematical Definition: For each node u, the nondeterminism degree $d(u)$ is defined as:

$$d(u) = \sum_{c \in C} \max(0, |\delta(u, c)| - 1)$$

where $\delta(u, c)$ denotes the set of nodes reachable from u by edges labeled with color c, and $|\delta(u, c)|$ is the count of such edges. By subtracting 1, we account only for the "excess" nondeterministic transitions, ignoring the first deterministic choice.

This approach ensures $d(u) \geq 0$, where $d(u) = 0$ if u has at most one outgoing edge per color, reflecting a deterministic transition structure at u.

Step 3: Calculate the Weighted Nondeterminism Degree $d'(u)$

We then compute the weighted nondeterminism degree $d'(u)$ by combining $d(u)$ with the centrality score Centrality(u). This weighting scales $d(u)$ to reflect both the intensity of nondeterminism and the node's structural significance.

Formula:

$$d'(u) = \text{Centrality}(u) \times d(u)$$

This scaling process adjusts $d(u)$ by the importance of u, putting more weigh on nondeterministic behavior at more influential nodes. The weighted degree $d'(u)$ ensures that nondeterministic choices at central nodes have a greater impact on the final metric, aligning with the concept of a global nondeterminism measure for G.

Step 4: Convert G to a Deterministic Finite Automaton (DFA) Using Subset Construction

To obtain a global measure of nondeterminism across G, we convert it into a Deterministic Finite Automaton (DFA) through subset construction. In this process, each DFA state represents a subset of nodes reachable through nondeterministic paths in G, enabling us to account for all possible transitions.

Subset Construction Process:

1. **Initialize**: The DFA's states are initialized with the initial node subset containing only v_0.
2. For each subset S in the DFA states:
    - For each color c, determine the set of nodes reachable by edges labeled c from nodes in S.
    - If this reachable set (new DFA state) is not already in the DFA states, add it and process it further.
3. Repeat until all reachable DFA states are identified.

The result, S_D, represents the DFA states, which include all possible nondeterministic node combinations reachable from v_0. This step ensures that $M_1(G)$ captures the nondeterministic behavior across the entire graph, not just individual walks.

Step 5: Aggregate and Normalize to Obtain $M_1(G)$

Finally, we compute $M_1(G)$ by aggregating the weighted nondeterminism degrees $d'(u)$ across all DFA states in S_D and normalizing by the total number of DFA states to ensure scale invariance.

Formula:

$$M_1(G) = \frac{1}{|S_D|} \sum_{S \in S_D} \sum_{u \in S} d'(u)$$

This normalization by $|S_D|$ ensures that $M_1(G)$ remains independent of graph size, allowing consistent comparison across graphs.

Pseudocode for $M_1(G)$

```
1. Compute PageRank Centrality:
   - Initialize each node's centrality.
   - Distribute centrality scores through damping factor.
   - Iterate until convergence.

2. Calculate Nondeterminism Degree d(u):
   - For each node u, for each color c:
       d(u) += max(0, |(u, c)| - 1)

3. Compute Weighted Nondeterminism Degree d'(u):
   - For each node u, d'(u) = Centrality(u) * d(u)

4. Construct DFA Using Subset Construction:
   - Initialize DFA states with initial node subset.
   - For each color c, compute reachable nodes.
   - Add new DFA states as subsets.

5. Aggregate and Normalize:
   - M1(G) = (1 / |S_D|) * sum(S in S_D) * sum(u in S) d'(u)
```

Question 2

This algorithm uses a *Markov chain model* to analyze nondeterminism in a directed, edge-colored graph G, with each color transition treated as a state transition. This setup provides a framework to understand the variability in color sequences—capturing nondeterministic characteristics through transition probabilities, stationary distributions, and entropy calculations.

Overview and Justification

1. Modeling with a Markov Chain

The *Markov chain model* serves as an abstraction for analyzing color transitions within the graph G, allowing each color to be treated as a distinct state. By focusing on color transitions rather than specific walks, this model captures the inherent variability in the sequences produced by the graph.

2. Constructing the Transition Matrix

The transition matrix P is constructed by normalizing observed transition counts between colors. This matrix represents the probability of transitioning from one color to another, transforming the observed data into a probabilistic model which ideal for statistical analysis.

3. Calculating the Stationary Distribution π

The *stationary distribution* π reflects the long-term probabilities of each color state, independent of initial conditions. Solving $\pi P = \pi$ provides a steady-state perspective that incorporates the relative prevalence of each color, allowing for a more accurate calculation of the graph's nondeterminism.

4. Using Entropy Rate as a Measure of Nondeterminism

The entropy rate h quantifies the average uncertainty or unpredictability in the transitions between color states. By weighting the entropy with the stationary distribution, this calculation captures the overall nondeterminism present in the color sequences produced by the graph. A higher entropy rate signifies greater nondeterminism, indicating that there are more equally probable color transitions, which translates to higher unpredictability in the color sequences. Conversely, a lower entropy rate reflects more deterministic transitions with less variability, suggesting a more predictable structure.

5. Assigning Transition Probabilities

Transition probabilities can be assigned based on different assumptions about the graph's edge-colored structure:

- **Uniform Transition Probabilities:** When no frequency data is available, assume each outgoing transition from a color has an equal likelihood. For a color c_i with k outgoing colors, assign a transition probability of $\frac{1}{k}$ for each.
- **Frequency-Based Transition Probabilities:** If multiple edges exist between colors, transition probabilities are based on observed counts. For transitions from c_i to c_j, the probability is given by:

$$P[c_i][c_j] = \frac{T[c_i][c_j]}{\sum_{k=1}^{n} T[c_i][k]}$$

  where $T[c_i][c_j]$ is the transition count from c_i to c_j.

Mathematical Derivation of the Metric $M_2(G)$

Given a directed, edge-colored graph G with color set $C = \{c_1, c_2, \ldots, c_n\}$, the following steps define the metric $M_2(G)$:

1. Transition Count Matrix T

For each observed transition from color c_i to c_j, increment the entry $T[c_i][c_j]$ in the transition count matrix:

$$T[c_i][c_j] = \text{Number of transitions from } c_i \text{ to } c_j.$$

2. Transition Probability Matrix P

Normalize each row in T to construct the transition probability matrix P:

$$P[c_i][c_j] = \frac{T[c_i][c_j]}{\sum_{k=1}^{n} T[c_i][c_k]},$$

ensuring that each row in P sums to 1.

3. Stationary Distribution π

To obtain the stationary distribution, solve $\pi P = \pi$, subject to the constraint $\sum_{i=1}^{n} \pi_i = 1$. The result, $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$, represents the steady-state probability distribution over colors.

4. Entropy Rate Calculation

The entropy rate h quantifies the average uncertainty per color transition:

$$h = -\sum_{i=1}^{n} \pi_i \sum_{j=1}^{n} P[c_i][c_j] \log_2 P[c_i][c_j],$$

where:

- The inner sum $\sum_{j=1}^{n} P[c_i][c_j] \log_2 P[c_i][c_j]$ calculates the conditional entropy for transitions from c_i to c_j,
- The outer sum weights this entropy by the stationary probability π_i.

5. Metric Definition $M_2(G)$

The nondeterminism metric $M_2(G)$ is defined as the entropy rate h:

$$M_2(G) = h$$

where:

- Higher $M_2(G)$: Indicates a high level of unpredictability in color transitions.
- Lower $M_2(G)$: Reflects a more deterministic structure in transitions.

Example

Let's apply the algorithm to a sample graph with colors $C = \{\text{Red}, \text{Yellow}, \text{Green}\}$ and the following transition probabilities:

- Red $\rightarrow$ Yellow (0.7)
- Red $\rightarrow$ Green (0.3)
- Yellow $\rightarrow$ Green (1.0)
- Green $\rightarrow$ Red (0.4)
- Green $\rightarrow$ Yellow (0.6)

Assuming we observe 10 transitions from each state, the transition count matrix T and normalized matrix P are calculated as follows:

$$T = \begin{bmatrix} 0 & 7 & 3 \\ 0 & 0 & 10 \\ 4 & 6 & 0 \end{bmatrix}$$

where each row corresponds to the counts of transitions from each color to others (Red, Yellow, Green).

Transition Probability Matrix P

Each entry $P[c_i][c_j]$ is obtained by dividing the transition count $T[c_i][c_j]$ by the total transitions from c_i. Therefore, the normalized matrix P is:

$$P = \begin{bmatrix} 0 & \frac{7}{10} & \frac{3}{10} \\ 0 & 0 & 1.0 \\ \frac{4}{10} & \frac{6}{10} & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0.7 & 0.3 \\ 0 & 0 & 1.0 \\ 0.4 & 0.6 & 0 \end{bmatrix}$$

Stationary Distribution π

To calculate the stationary distribution π, we solve for π in $\pi P = \pi$ with the constraint $\sum \pi_i = 1$.

Let $\pi = (\pi_{\text{Red}}, \pi_{\text{Yellow}}, \pi_{\text{Green}})$. Then:

$$\begin{cases} \pi_{\text{Red}} = 0.4\pi_{\text{Green}} \\ \pi_{\text{Yellow}} = 0.6\pi_{\text{Green}} \\ \pi_{\text{Red}} + \pi_{\text{Yellow}} + \pi_{\text{Green}} = 1 \end{cases}$$

Substitute $\pi_{\text{Red}} = 0.4\pi_{\text{Green}}$ and $\pi_{\text{Yellow}} = 0.6\pi_{\text{Green}}$ into the sum constraint:

$$0.4\pi_{\text{Green}} + 0.6\pi_{\text{Green}} + \pi_{\text{Green}} = 1$$

$$2\pi_{\text{Green}} = 1 \Rightarrow \pi_{\text{Green}} = \frac{1}{2}$$

Then:

$$\pi_{\text{Red}} = 0.4 \times \frac{1}{2} = \frac{1}{4}, \quad \pi_{\text{Yellow}} = 0.6 \times \frac{1}{2} = \frac{3}{10}$$

Thus, the stationary distribution is:

$$\pi = (\pi_{\text{Red}}, \pi_{\text{Yellow}}, \pi_{\text{Green}}) = \left(\frac{1}{4}, \frac{3}{10}, \frac{1}{2}\right)$$

Entropy Rate Calculation h

The entropy rate h is calculated as:

$$h = -\sum_{i=1}^{3} \pi_i \sum_{j=1}^{3} P[c_i][c_j] \log_2 P[c_i][c_j]$$

Breaking it down by color:

1. For Red ($\pi_{\text{Red}} = \frac{1}{4}$):

$$h_{\text{Red}} = -\frac{1}{4} \left(0.7 \log_2 0.7 + 0.3 \log_2 0.3\right)$$

where:

$$0.7 \log_2 0.7 \approx 0.7 \times (-0.5146) = -0.3602$$
$$0.3 \log_2 0.3 \approx 0.3 \times (-1.737) = -0.5211$$

So,

$$h_{\text{Red}} = -\frac{1}{4}(-0.3602 - 0.5211) = \frac{1}{4} \times 0.8813 = 0.2203$$

2. For Yellow ($\pi_{\text{Yellow}} = \frac{3}{10}$):

$$h_{\text{Yellow}} = -\frac{3}{10} \times 1.0 \times \log_2 1.0 = 0$$

3. For Green ($\pi_{\text{Green}} = \frac{1}{2}$):

$$h_{\text{Green}} = -\frac{1}{2} \left(0.4 \log_2 0.4 + 0.6 \log_2 0.6\right)$$

where:

$$0.4 \log_2 0.4 \approx 0.4 \times (-1.322) = -0.5288$$
$$0.6 \log_2 0.6 \approx 0.6 \times (-0.737) = -0.4422$$

So,

$$h_{\text{Green}} = -\frac{1}{2}(-0.5288 - 0.4422) = \frac{1}{2} \times 0.971 = 0.4855$$

Summing these values, the total entropy rate h is:

$$h = h_{\text{Red}} + h_{\text{Yellow}} + h_{\text{Green}} = 0.2203 + 0 + 0.4855 = 0.7058 \text{ bits per transition}$$

Thus, the metric $M_2(G)$ is:

$$M_2(G) = h = 0.7058 \text{ bits per transition}$$

Pseudocode for $M_2(G)$ Algorithm

Algorithm 1 Compute $M_2(G)$

function COMPUTE_M2(G, C)
  Input: Graph G with edges labeled by colors in set C
  Output: Nondeterminism metric $M_2(G)$
  Step 1: Initialize the transition count matrix T
  $T \leftarrow$ Zero matrix of size $|C| \times |C|$
  Step 2: Populate T **with observed transitions**
  for each edge (u, v, c_i) in G **do**
    for each outgoing edge (v, w, c_j) **do**
      $T[c_i][c_j] \leftarrow T[c_i][c_j] + 1$
    end for
  end for
  Step 3: Normalize T **to get transition matrix** P
  for each color c_i in C **do**
    total_transitions $\leftarrow \sum T[c_i][*]$
    if total_transitions > 0 **then**
      for each color c_j in C **do**
        $P[c_i][c_j] \leftarrow \frac{T[c_i][c_j]}{\text{total\_transitions}}$
      end for
    end if
  end for
  Step 4: Compute stationary distribution π
  Initialize $\pi \leftarrow$ uniform distribution over colors
  while not converged **do**
    $\pi\_new \leftarrow \pi \times P$
    Check convergence: $\pi\_new \approx \pi$
    $\pi \leftarrow \pi\_new$
  end while
  Step 5: Calculate entropy rate h
  $h \leftarrow 0$
  for each color c_i in C **do**
    for each color c_j in C **do**
      if $P[c_i][c_j] > 0$ **then**
        $h \leftarrow h + \pi[c_i] \times P[c_i][c_j] \times \log_2(P[c_i][c_j])$
      end if
    end for
  end for
  Step 6: Return the final metric
  return $M_2(G) = -h$
end function

Question 3.1

Our goal is to measure the level of nondeterminism in a blackbox program by quantifying the unpredictability in its outputs. As shown in class we can model the program as a NFA, where each state transition corresponds to outputs labeled with colors from a finite set. With this setup in place, this algorithm estimates nondeterminism by analyzing multiple outputs generated from repeated identical inputs, using a combination of Fourier analysis and spectral entropy.

First, let's give a high level overview of the algorithm:

1. First, as stated in the problem, we collect outputs by executing repeated test cases to generate multiple output sequences. These sequences are used to observe the variability in the program's behavior.

2. Next, we transform the colors to numerical time series. We do this by converting each color-labeled output to numerical values, which facilitates Fourier analysis

3. Using Fourier analysis, we analyze the frequency domain representation of each sequence to identify deterministic or random patterns.

4. Next, we calculate entropy from the Fourier-transformed data, using spectral entropy. This allows us to quantify randomness by assessing how power is distributed across frequencies.

5. Finally, we aggregate the entropy values across tests to yield a final nondeterminism metric. This will give us an idea on how high or low the non-determinism is.

1 Methods

Fourier analysis and spectral entropy are used here to analyze time-series data. Fourier analysis is a powerful tool for decomposing data into frequency components, revealing hidden periodicities that are not immediately apparent in the time domain [**?**]. Deterministic signals manifest as strong, discrete peaks in the frequency spectrum, whereas random or nondeterministic signals produce a more uniform power distribution [**?**]. Additionally, time-frequency analysis techniques help assess the randomness of signals, with diffuse energy distributions in the frequency domain often indicating random behaviors [**?**].

Spectral entropy, as introduced by Inouye et al. [**?**], serves as a quantitative measure of randomness by evaluating the spread of power across frequency components. Low spectral entropy indicates structured signals, while high spectral entropy corresponds to unpredictable, random signals. Further, the combined use of symbolic dynamics and entropy measures can provide insights into the complex dynamics of experimental data [**?**]. Together, Fourier analysis and spectral entropy offer a robust framework for distinguishing structured from

random patterns, enabling a comprehensive understanding of nondeterministic behaviors in the blackbox program.

1.1 Justification for Chosen Methods

Fourier analysis and spectral entropy were selected as they effectively reveal periodicities and measure randomness in time-series data. Fourier analysis decomposes data into its constituent frequency components, allowing for the identification of deterministic patterns, which appear as strong peaks in the frequency spectrum, in contrast to nondeterministic patterns, which exhibit uniform power distribution. Spectral entropy then quantifies this randomness by assessing the spread of power across frequencies, distinguishing between structured and unpredictable outputs. Combined, these techniques provide a quantitative measure of nondeterminism.

Pseudocode for Algorithm

Algorithm 2 MeasureNondeterminism

Require: Set of input sequences $\{I_1, I_2, \ldots, I_N\}$, repetitions M, color set C
Ensure: Nondeterminism metric N_{total}

Initialize empty list `entropy_values`
for each input sequence I_i in $\{I_1, I_2, \ldots, I_N\}$ **do**
    for each repetition $m = 1$ to M **do**
        Generate output sequence O_{im} from blackbox for input I_i
        Map O_{im} to time series T_{im} using mapping function $\mathcal{M}$
        Compute DFT $X_{im}(k)$ of T_{im} for $k = 0$ to $n - 1$
        Calculate power spectrum $P_{im}(k) = |X_{im}(k)|^2$
        Normalize P_{im} to obtain probability distribution $p_{im}(k)$
        Calculate spectral entropy $H_{\text{spectral}}(O_{im})$
        Normalize entropy $H_{\text{normalized}}(O_{im}) = H_{\text{spectral}}(O_{im})/H_{\max}$
    end for
    Compute $H_{\text{average}}(I_i)$ as the mean of $H_{\text{normalized}}(O_{im})$ over M trials
    Append $H_{\text{average}}(I_i)$ to `entropy_values`
end for
Compute N_{total} as the mean of `entropy_values` **return** N_{total}

In Depth Walkthrough and Justification of Algorithm

1. First, we generate a diverse set of input sequences $\{I_1, I_2, \ldots, I_N\}$ from the color set C, allowing us to explore the program's full range of behaviors. For each input sequence I_i, execute the blackbox M times, yielding a collection of output sequences $\{O_{i1}, O_{i2}, \ldots, O_{iM}\}$. This is part of the given setup. Where each output sequence is represented as:

$$O_{im} = [o_{im,1}, o_{im,2}, \ldots, o_{im,n}]$$

where $o_{im,j} \in C$ is the output at position j in the sequence.

2. To perform Fourier analysis, we need to represent categorical outputs as numerical time series. This transformation is necessary to retain the output sequence's order and frequency, enabling the identification of deterministic or random patterns. We define a mapping function $\mathcal{M} : C \to \mathbb{R}$ to convert each color $c \in C$ into a unique real number $\mathcal{M}(c)$. Using this, each sequence O_{im} is mapped to a numerical time series:

$$T_{im} = [t_{im,1}, t_{im,2}, \ldots, t_{im,n}]$$

where $t_{im,j} = \mathcal{M}(o_{im,j})$ for $j = 1, \ldots, n$. For example, if $O_{im} = [\text{Red}, \text{Green}, \text{Red}, \text{Blue}]$, it might map to $T_{im} = [1, 2, 1, 3]$ based on a predefined mapping.

We can justify treating an FA walk as a time series because each transition in the walk corresponds to a sequential output tied to an input, forming an ordered sequence that reflects the program's evolving state over time, much like a time-dependent signal. Everything that moves is a FA!

3. Next, we apply Fourier analysis to each numerical time series. Fourier analysis decomposes time series data into its frequency components, making it possible to detect periodic patterns or uniform randomness. We compute the Discrete Fourier Transform (DFT) for each sequence T_{im}:

$$X_{im}(k) = \sum_{j=0}^{n-1} t_{im,j} \cdot e^{-\frac{2\pi i k j}{n}}, \quad k = 0, 1, \ldots, n-1$$

We use a DFT and not a Fourier Transform because our data consists of discrete, finite sequences, not continuous signals. The DFT transforms the time series into the frequency domain, where the FFT algorithm is used for computational efficiency.

4. We then calculate the magnitude $|X_{im}(k)|$ for each frequency component k, forming the power spectrum:

$$P_{im}(k) = |X_{im}(k)|^2$$

This spectrum reveals energy concentration across frequencies, allowing us to distinguish deterministic from random behaviors. For deterministic outputs, concentrated energy at specific frequencies suggests periodic patterns, while nondeterministic outputs exhibit a more uniform energy distribution across frequencies, indicating randomness..

The power spectrum's structure categorizes behavior as follows:

- Strong peaks indicate deterministic responses.
- A uniform spread implies high randomness, suggesting nondeterminism.

5. To quantify the uncertainty within the frequency distribution, we calculate spectral entropy, which provides a precise measure of variability in the

power spectrum. First, the power spectrum is converted into a probability distribution, allowing us to interpret power as probabilities, which is necessary for entropy calculation. For each frequency component k, we normalize the power:

$$p_{im}(k) = \frac{P_{im}(k)}{\sum_{k=0}^{n-1} P_{im}(k)}$$

6. Spectral entropy $H_{\text{spectral}}(O_{im})$ is calculated as:

$$H_{\text{spectral}}(O_{im}) = -\sum_{k=0}^{n-1} p_{im}(k) \log_2 p_{im}(k)$$

This calculation provides:

- **Low Entropy**: Concentrated energy at certain frequencies, indicating determinism.
- **High Entropy**: Uniform energy distribution, indicating nondeterministic, random responses.

8. We normalize by dividing by the maximum possible entropy:

$$H_{\text{normalized}}(O_{im}) = \frac{H_{\text{spectral}}(O_{im})}{H_{\max}}$$

where $H_{\max} = \log_2 n$. This normalization constrains the entropy value between 0 and 1, providing a consistent measure across all output sequences.

9. To smooth out variations across trials, calculate the average normalized entropy for each input sequence I_i:

$$H_{\text{average}}(I_i) = \frac{1}{M} \sum_{m=1}^{M} H_{\text{normalized}}(O_{im})$$

This average reflects the consistency (or lack thereof) of the blackbox's response for each input sequence, with higher averages indicating greater variability.

10. Finally, calculate the total nondeterminism metric by averaging entropy values across all input sequences:

$$N_{\text{total}} = \frac{1}{N} \sum_{i=1}^{N} H_{\text{average}}(I_i)$$

where:

- $N_{\text{total}} \approx 0$: Indicates deterministic behavior.
- $N_{\text{total}} \approx 1$: Suggests highly nondeterministic behavior, indicating significant unpredictability in outputs.

Mini Paper: Application of Algorithm 3.1 in Spintronics

1. Introduction to Spintronics and Ferromagnets

Spintronics is a field that leverages the intrinsic spin of electrons, along with their charge, to enable advanced data storage and processing in solid-state devices [2]. Ferromagnetic materials are fundamental components in spintronic devices due to their ability to maintain a net magnetization without an external magnetic field. Understanding the behavior of spin states in ferromagnets is essential for improving device performance and reliability.

A significant challenge in spintronics is managing the randomness introduced by thermal fluctuations and quantum effects, which result in nondeterministic spin transitions [4]. This paper illustrates how the algorithm can be applied to quantify nondeterminism in the spin states of ferromagnetic materials.

2. Experimental Studies of Spin in Ferromagnets

Research on the spin dynamics of ferromagnets often involves observing how spin states respond to external stimuli, such as magnetic fields or electric currents [**?**]. Common experimental techniques include:

- **Spin Polarized Scanning Tunneling Microscopy (SP-STM)**: Allows for imaging and manipulating spin states at the atomic scale [**?**].
- **Ferromagnetic Resonance (FMR)**: Measures the collective precession of spins in response to microwave-frequency magnetic fields [5].
- **Pump-Probe Experiments**: Utilize ultrafast laser pulses to study spin dynamics on femtosecond timescales [7].

In these experiments, researchers often apply a consistent stimulus and record the resulting spin states over multiple trials to understand the effects of thermal fluctuations and other sources of randomness.

3. Converting the Experiment into a FA

In order to apply the algorithm to study spin dynamics in ferromagnets, we model the experimental setup as a finite automaton (FA). In this context, the FA serves as an abstract representation that captures the essential features of the spin system, including its states, inputs, transitions, and outputs.

The possible spin configurations of the ferromagnet constitute the states of the FA. For simplicity, we consider each electron spin to be either in the spin-up ($\uparrow$) or spin-down ($\downarrow$) state. These two states represent the fundamental states of the FA.

The inputs to the FA correspond to the external stimuli applied during the experiment, such as magnetic field pulses or electric current injections. These

stimuli are intended to influence the spin states, inducing transitions between them. In the FA model, these inputs form the alphabet that triggers state transitions.

Transitions in the FA represent the changes in spin states induced by the inputs. Due to thermal fluctuations and quantum effects, these transitions are inherently probabilistic. This means that applying the same input multiple times may lead to different state transitions, reflecting the nondeterministic nature of the system.

The outputs of the FA are the observed spin states after each input is applied. By measuring the spin state following each stimulus, we obtain a sequence of outputs that can be analyzed to understand the behavior of the system.

4. Applying the Algorithm to Experimental Data

To begin, we can consider a simple experimental scenario where a scientist, Dr. Ferro, aims to quantify the nondeterminism in spin state transitions of a ferromagnetic material.

Suppose a ferromagnetic thin film is subjected to a uniform magnetic field pulse. The experiment involves the following steps:

- The same magnetic field pulse of fixed magnitude and duration is applied repeatedly to the sample.
- After each pulse, the spin state of a specific region in the material is measured using a sensitive detection method, such as magneto-optical Kerr effect (MOKE) microscopy [1].
- The measurement records whether the net spin orientation in that region is spin-up ($\uparrow$) or spin-down ($\downarrow$).
- This process is repeated M times to obtain a sequence of spin states under identical conditions.

We assume that thermal fluctuations at room temperature contribute to the randomness in the spin transitions, leading to variations in the measured spin states despite the consistent input. [**?**, **?**]

Data Collection

The collected data consists of N trials, each containing M measurements:

$$\{O_i = [o_{i1}, o_{i2}, \ldots, o_{iM}] \mid i = 1, 2, \ldots, N\}$$

where $o_{ij} \in \{\uparrow, \downarrow\}$ represents the spin state observed in the j-th measurement of the i-th trial.

Mapping to Numerical Time Series

To apply Fourier analysis, we convert the categorical spin states into numerical values using a mapping function:

$$\mathcal{M}(\uparrow) = +1, \quad \mathcal{M}(\downarrow) = -1$$

Each output sequence O_i is transformed into a numerical time series:

$$T_i = [t_{i1}, t_{i2}, \ldots, t_{iM}] \quad \text{with} \quad t_{ij} = \mathcal{M}(o_{ij})$$

Fourier Analysis

For each time series T_i, we compute the Discrete Fourier Transform (DFT) to analyze the frequency components:

$$X_i(k) = \sum_{j=0}^{M-1} t_{ij} \cdot e^{-2\pi i k j / M}, \quad k = 0, 1, \ldots, M-1$$

The power spectrum $P_i(k)$ is calculated as:

$$P_i(k) = |X_i(k)|^2$$

The power spectrum reveals how the energy of the signal is distributed across different frequencies, which helps identify any periodicities or patterns in the spin state transitions.

Spectral Entropy Calculation

To quantify the randomness in the frequency domain, we compute the spectral entropy. First, we normalize the power spectrum to form a probability distribution:

$$p_i(k) = \frac{P_i(k)}{\sum_{k=0}^{M-1} P_i(k)}$$

The spectral entropy $H_{\text{spectral}}(T_i)$ is then calculated:

$$H_{\text{spectral}}(T_i) = -\sum_{k=0}^{M-1} p_i(k) \log_2 p_i(k)$$

A higher spectral entropy indicates a more uniform distribution of power across frequencies, corresponding to greater randomness in the spin state transitions.

Aggregating Entropy Values

We normalize the spectral entropy by the maximum possible entropy $H_{\max} = \log_2 M$:

$$H_{\text{normalized}}(T_i) = \frac{H_{\text{spectral}}(T_i)}{H_{\max}}$$

The overall nondeterminism metric N_{total} is obtained by averaging the normalized entropy across all trials:

$$N_{\text{total}} = \frac{1}{N} \sum_{i=1}^{N} H_{\text{normalized}}(T_i)$$

Example Application

Let's consider a simplified example with $N = 5$ trials and $M = 10$ measurements per trial. Suppose the mapped time series for one trial is:

$$T_1 = [+1, -1, +1, -1, +1, -1, +1, -1, +1, -1]$$

This sequence exhibits a clear periodic pattern, alternating between spin-up and spin-down. Computing the DFT and power spectrum, we find a strong peak at a specific frequency corresponding to this periodicity. The spectral entropy $H_{\text{normalized}}(T_1)$ would be low, indicating a deterministic pattern.

In contrast, another trial might produce a time series like:

$$T_2 = [+1, +1, -1, +1, -1, -1, +1, -1, -1, +1]$$

This sequence appears more random. Its power spectrum would show a more uniform distribution across frequencies, resulting in a higher spectral entropy $H_{\text{normalized}}(T_2)$.

By calculating $H_{\text{normalized}}(T_i)$ for each trial and averaging them, we obtain N_{total}, which quantifies the overall nondeterminism in the spin transitions.

Final Results

A low N_{total} suggests that the spin transitions are predominantly deterministic, possibly influenced strongly by the external magnetic field. A high N_{total} indicates significant randomness due to thermal fluctuations .

The scientist, Dr.Ferro, can use this metric to assess the effectiveness of the external stimulus in controlling spin states and to understand the extent to which thermal effects impact the system. Thus, this information can be used for designing spintronic devices that require precise control over spin states!

5. Algorithm Complexity

The following is the derivation for the algorithm's time complexity.

1. *Time Series Conversion*: Each of the N sequences of spin transitions, each of length M, is converted into a numerical time series. This process involves mapping each of the M elements in a sequence to a numerical value, resulting in a complexity of $O(M)$ per sequence. Therefore, the complexity for this step across all sequences is:

$$O(N \cdot M). \tag{1}$$

2. *Discrete Fourier Transform (DFT)*: Using the Fast Fourier Transform (FFT), each sequence is transformed from the time domain to the frequency domain, which identifies deterministic and periodic behaviors in the spin responses. The FFT has a time complexity of $O(M \log M)$ per sequence. Thus, for all sequences, the complexity becomes:

$$O(N \cdot M \log M). \tag{2}$$

3. *Spectral Entropy Calculation*: Once the power spectrum is obtained, calculating the spectral entropy for each sequence requires normalizing the power spectrum values and then applying the entropy formula $H = -\sum_{k=0}^{M-1} p(k) \log_2 p(k)$. Both the normalization and entropy calculations have a complexity of $O(M)$ per sequence, yielding a total complexity across sequences of:

$$O(N \cdot M). \tag{3}$$

4. *Aggregation*: Finally, the entropy values for all sequences are averaged to yield a single metric for nondeterminism. This summation and averaging involve N additions and a division, resulting in a complexity of:

$$O(N). \tag{4}$$

Overall Complexity

$$O(N \cdot M \log M). \tag{5}$$

6. Strengths and Weaknesses of the Approach

This algorithm shines when applied to the spintronic study of ferromagnetic systems, where the quantification of nondeterminism can unlock insights into how thermal and quantum fluctuations influence spin states [2, 3]. A primary advantage of this method is its quantitative assessment of randomness: by generating a concrete metric for nondeterminism, it enables clear, objective comparisons across different experimental setups or magnetic materials. This feature is particularly valuable for spintronics, where characterizing the extent of randomness in spin transitions can guide improvements in device stability and performance [4].

Another strength lies in the detection of hidden patterns through Fourier analysis. By converting time-domain data into the frequency domain, this algorithm can uncover periodic behaviors or deterministic tendencies within the otherwise stochastic spin responses, a capability that standard time-domain analyses might miss [5].

Additionally, the algorithm's computational efficiency is a significant strength. Its complexity is primarily driven by the Discrete Fourier Transform (DFT) used in analyzing each sequence of spin transitions. Although the DFT generally has a higher computational cost than the FFT, it remains manageable for datasets in this study, with an overall complexity of

$$O(N \cdot M \log M). \tag{6}$$

where N is the number of trials and M is the length of each time series.

However, some limitations of the algorithm become apparent when considering the constraints of real-world data collection and the intricacies of spintronic materials. High data requirements are a notable challenge; the algorithm's effectiveness depends on large amounts of high-quality data to accurately characterize the spectral entropy and detect patterns. In practical experimental settings, especially at nanoscale or ultrafast timescales, collecting such extensive data can be challenging, if not impractical [7]. Thus, while the algorithm excels in data-rich scenarios, it may yield less reliable results when only small batches of data are available, reducing its utility in time- or resource-constrained experiments. From what I understand, small batches of data is what researchers in spin tend to deal with.

Furthermore, the assumption of stationarity in Fourier analysis can limit the algorithm's applicability. This assumption implies that the statistical properties of the spin transitions remain constant over time, which may not hold in systems where external conditions, like temperature or applied fields, vary during the experiment [2, 4]. When stationarity is not met, the frequency analysis may misinterpret transient effects as deterministic patterns, leading to skewed entropy calculations and inaccurate assessments of nondeterminism.

6. Conclusion

The algorithm presented provides an essential quantitative framework for examining the nondeterminism inherent in ferromagnetic spin transitions. By modeling the system as a finite automaton and applying advanced signal processing techniques, such as Fourier analysis and spectral entropy, this approach captures the nuanced dynamics of spin behavior in ferromagnetic materials.

References

[1] Z. Q. Qiu and S. D. Bader, *Surface magneto-optic Kerr effect*, *Rev. Sci. Instrum.*, vol. 71, no. 3, pp. 1243–1255, 2000.

[2] I. Žutić, J. Fabian, and S. Das Sarma, *Spintronics: Fundamentals and applications*, *Rev. Mod. Phys.*, vol. 76, no. 2, pp. 323–410, 2004.

[3] L. Berger, *Emission of spin waves by a magnetic multilayer traversed by a current*, *Phys. Rev. B*, vol. 54, no. 13, pp. 9353–9358, 1996.

[4] C. Chappert, A. Fert, and F. N. Van Dau, *The emergence of spin electronics in data storage*, *Nature Mater.*, vol. 6, no. 11, pp. 813–823, 2007.

[5] B. Heinrich and J. F. Cochran, *Ultrathin metallic magnetic films: Magnetic anisotropies and exchange interactions*, *Adv. Phys.*, vol. 42, no. 5, pp. 523–639, 1993.

[6] J. A. Katine and E. E. Fullerton, *Device implications of spin-transfer torques*, *J. Magn. Magn. Mater.*, vol. 320, no. 7, pp. 1217–1226, 2008.

[7] A. Kirilyuk, A. V. Kimel, and T. Rasing, *Ultrafast optical manipulation of magnetic order*, *Rev. Mod. Phys.*, vol. 82, no. 3, pp. 2731–2784, 2010.