

# Report from project SAD2

## 1 Datasets and Ground Truth

Datasets (simulated trajectories) and their corresponding ground-truth Boolean networks are generated by sweeping the following parameter grid:

Parameter	Values
<i>Network parameters</i>	
n_nodes	{5, 8, 11, 16}
network_seed	{0, 1}
<i>Trajectory parameters</i>	
n_trajectories	{1, 5, 20}
sync transition	{True, False}
trajectory_len	{5, 20, 100}
sampling_frequency	{1, 3}

Table 1: Parameter grid used for generating datasets and ground-truth networks.

Larger networks (e.g., `n_nodes` > 12) significantly increased runtime for both dataset generation and BNFiner inference, so instead of exhaustively testing all sizes we used a representative set spanning the range [5, 16]. In total, the sweep contains  $4 \times 2 \times 2 \times 3 \times 3 \times 2 = 288$  parameter combinations, which already requires substantial computational effort during evaluation. In addition, we tracked the `attractor_state_percentage`, which varies between 0 and 1 depending on the trajectory. Overall, the chosen grid allowed us to assess which conditions favor accurate graph-structure inference.

## 2 Evaluation

- **Reconstruction setup.** We used the simplest BNFiner invocation: `bnf -e input1.txt -n output1.sif -l 3`. The key adjustment was `-l 3`, which limits each Boolean function to at most three parents; this greatly reduces the search space and speeds up reconstruction.
- **Scoring functions.** We evaluated candidate network structures using two scores recommended for BNFiner: *Minimal Description Length (MDL)* and *BDe (Bayesian–Dirichlet equivalence)*.
- **Loss functions (accuracy metrics).** We measured structural prediction error using `edge jaccard distance` and `graph edit distance (GED)`: together they capture both *local edge overlap* (exact wiring agreement) and *global topological discrepancy* (how many edits are needed to transform one graph into the other). We chose these metrics because they are widely used in graph-structure evaluation and provide complementary views of reconstruction quality.

### 3 Dataset parameters vs. reconstruction quality

To assess whether parameter values were significantly associated with final reconstruction accuracy, Spearman's rank correlation coefficients and their corresponding p-values were computed between the loss function values and the parameter values. The results are presented in the table below:

<i>Loss function</i>	<i>Jaccard</i>		<i>GED</i>	
<i>Scoring function</i>	<i>BDe</i>	<i>MDL</i>	<i>BDe</i>	<i>MDL</i>
<code>sync</code>	0.01	0.02	-0.12	<b>-0.13</b>
<code>sampling_frequency</code>	<b>0.34</b>	<b>0.33</b>	0.09	0.07
<code>n_trajectories</code>	<b>-0.40</b>	<b>-0.37</b>	-0.10	-0.07
<code>trajectory_len</code>	<b>-0.31</b>	<b>-0.29</b>	-0.04	-0.03
<code>n_nodes</code>	0.00	<b>0.13</b>	<b>0.76</b>	<b>0.79</b>
<code>attractor_state_percentage</code>	<b>-0.24</b>	<b>-0.26</b>	<b>-0.31</b>	<b>-0.33</b>

Table 2: Spearman's rank correlation coefficient values between loss function and parameter values. Coefficient values which had their corresponding *p-value* < 0.05 (observed correlation is unlikely due to random chance) were marked in **bold**.

**Important note.** For both loss functions, lower values correspond to improved reconstruction accuracy. Consequently, a negative correlation coefficient indicates that as the parameter values increase, reconstruction accuracy improves.

### 4 Analysis

Correlation analysis between loss function values and parameter values revealed several noteworthy patterns.

- **Scoring functions.** There are no notable differences in correlation trends between MDL and BDe BNFiner scoring functions.
- **Loss functions.** As `edge jaccard distance` and `graph edit distance` capture different aspects of graph reconstruction accuracy, their correlations with parameter values varied. Nonetheless, employing both metrics allowed us to obtain complementary perspectives on reconstruction quality, as each parameter exhibited statistically significant correlations with one or both measures.
- **Parameters:**
  - `sync` (0 for False, 1 for True) - slightly negative correlation with GED suggests a positive effect of synchronously generated trajectories on reconstruction accuracy.
  - `sampling_frequency` - a relatively strong positive correlation with `edge jaccard distance` indicates that as the number of time steps between consecutive sampled states increases, reconstructing the graph becomes more difficult.
  - `attractor_state_percentage` - a consistent negative correlation with both `edge jaccard distance` and `graph edit distance` suggests that a higher proportion of attractor states relative to transient states in the dataset facilitates graph structure inference.

- **n\_trajectories** and **trajectory\_len** - a relatively strong negative correlation with **edge jaccard distance** (Figure 3) indicates that larger datasets facilitate more accurate graph reconstruction.

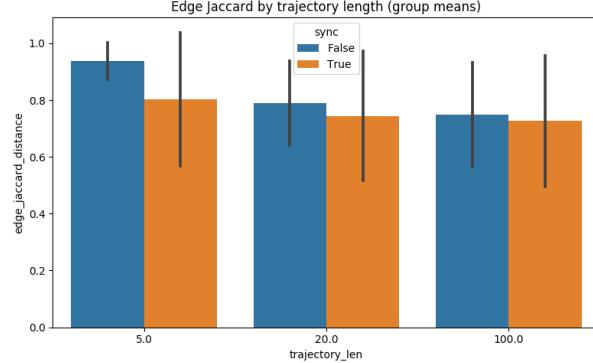


Figure 1: \*  
Jaccard vs. trajectory length (MDL)

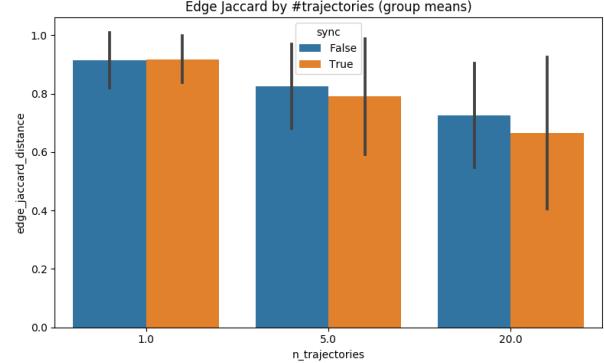


Figure 2: \*  
Jaccard vs. number of trajectories (MDL)

Figure 3: Edge Jaccard distance under MDL for two dataset axes.

- **n\_nodes** - network size exhibited a very strong positive correlation with GED (Figure 4), indicating that larger networks are more difficult to infer accurately.

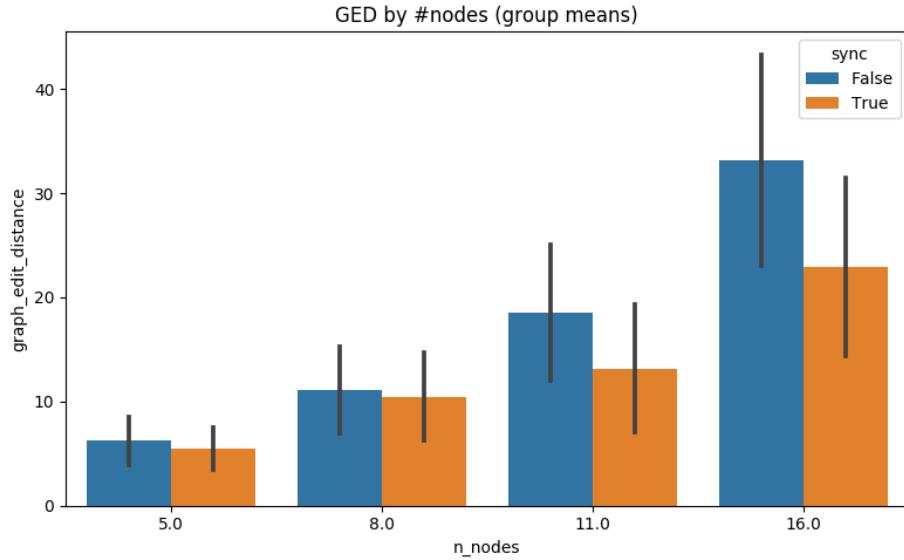


Figure 4: GED under MDL as a function of network size (**n\_nodes**).

**Final conclusions.** We evaluated a wide range of parameter configurations; however, presenting all resulting plots would reduce the readability of this report. Therefore, we selected only the most informative visualizations for inclusion here. All plots are available in the accompanying repository under the `plots/` directory.

The results varied substantially, ranging from near-perfect reconstructions to cases with almost no correctly inferred edges. For networks of fixed size, the most influential factor was dataset size:

increasing both the number of trajectories and their length consistently improved reconstruction performance (Figure 3). This observation aligns with intuition, as larger datasets provide more information for accurate inference. Conversely, reducing the number of states in the dataset by increasing the sampling frequency had a detrimental effect on performance.

We also observed a slight advantage for synchronous transitions, which is expected since synchronous updates are deterministic and, at each step, constrain the Boolean transition functions of all nodes simultaneously. The number of attractor states was another important factor, with a higher proportion of attractor states leading to improved reconstruction quality. We hypothesize this effect may arise because the algorithm can analyze the same set of transitions multiple times, thereby establishing stronger and more reliable connections between nodes.

The evaluation metrics (`edge_jaccard_distance` and `graph_edit_distance`) provided complementary insights into the relationships between model parameters and graph reconstruction quality. In contrast, we did not observe a consistent difference in performance between the MDL and BDe scoring methods. Finally, a strong negative correlation between network size and reconstruction accuracy was observed (Figure 4), which is expected given the increased difficulty of accurately inferring larger networks.

**Insights for the next task.** Based on our analysis, we gained several important insights that guided the design of the subsequent task. For the inference of a real biological model, we therefore propose generating a dataset using the following parameter settings: `sync = True`, `sampling_frequency = 1`, `attractor_state_percentage` - best to choose a random seed that allows us to generate a dataset with a high percentage of attractor states, `n_trajectories = 20` (or higher), `trajectory_len = 100` (or higher), `n_nodes` - the smaller the dataset the easier it will probably be to infer.

## 5 Challenges encountered

- **Legacy environment constraints.** One of the biggest challenges was setting up a fully compatible Python 2.7 environment (libraries, type checking, and tooling), since BNFiner does not support Python 3+. This turned out to be a valuable lesson in reproducibility: some of us managed the setup via `pyenv`, while others had to rely on Docker due to limited compatibility of their local machines with that early of a Python version.
- **Computational cost of the sweep.** To meaningfully assess how inference quality depends on key parameters (e.g., transition mode, network size etc.), we needed to run evaluations for a few hours, which limited how many additional combinations we could explore. To reduce runtime, we parallelized runs where possible and used BNFiner’s `-l` option to cap the maximum number of parents per node, substantially decreasing computational burden.