Dear all,

Thank you very much for your interest in our projects. In this email, we shall introduce the projects and specific tasks further, such that you can take some time to decide on what you are really keen to work on, and prepare yourselves by reading the references.

Since we have received more applications than we can take, **we would like to conduct a quick interview with each of you on Monday, 11 August, between 3:00 and 5:40 PM.** Please let us know which project tasks you are most interested in taking up, along with any relevant expertise you have. We encourage you to select multiple tasks, as there may be overlaps, and we will need to distribute them accordingly.

As you would have already noticed, our questions are intended to be high-impact, novel questions that no one has ever worked on. Therefore, this project will be intrinsically challenging as we are working in uncharted territory. There will be times when you, I, Xianquan, Prof. Lee and Prof. Duane do not know the answers for sure; in those times, we will have to go through this enriching process together, hopefully with your new ideas/initiatives.

In this sense, it's very important to act proactively (which some of you have already shown) and take responsibility for the project. In the end, we eventually want to publish our results in prestigious scientific and Computer Science journals or conference proceedings. I believe each of you—even perhaps after going into industry—will want to have the chance to say: *"When I was doing my MSc, I had the chance to contribute to this unknown question and contributed a droplet to the huge sea of science."* I think this should be intellectually desirable and satisfying for every one of you.

Also, as in any full-fledged AI for Science project, we value high-quality data generation and deep understanding of the system, beyond simply training models. Therefore, we will first be working on creating the data we need, and practising this step is arguably at least as important as learning how to fine-tune neural networks.

Below, let us describe the available projects and the tasks contained therein. In some cases, one student can contribute to more than one project, and more than one student may take part in (different aspects) of the same project :

1– Generating a general percolation library and review paper:

Percolation theory asks a deceptively simple question: given a collection of sites or bonds that are randomly occupied, when does a spanning cluster first appear? From the way water finds its way through coffee grounds, to the robustness of power grids, to the spread of information (or contagion) on social networks, the moment at which isolated domains suddenly knit together marks a universal transition between order and disorder. Remarkably, very different systems share the same critical exponents and scaling laws, making percolation a cornerstone

model for understanding collective behaviour in physics, materials science, biology, and network theory.

Although there are some percolation-related open-source repositories and reviews, there has been no general library containing all the needed features for percolation/criticality analysis. As we will need to generate data for our machine learning project, it's necessary to have the tools to do it. As we are doing this, there is no reason not to add a bit more effort to create a generic percolation library that the scientific community can use. While developing these codes, I will ask for booklet chapters on how your algorithms work, including mathematical details and their meaning. These reports will eventually be combined to give the community an understandable library with working details.

We already have a starter library from our previous project and we will build upon that. Although we will be adding more tasks as we progress in our project, some initial tasks will be:

1. Coding classical percolation models
2. Investigating literature to find all models showing percolation/criticality transition, and implementing
3. Generating arbitrary rules which percolate
4. Graph representation converter of the selected rules
5. Implementing average observable/moment calculations (Newman–Ziff Algorithm)
6. Implementing one cluster growth algorithm (Leath–Alexandrowicz algorithm.
7. Finite size scaling and binder cumulant calculations

2– Converting percolation rules to graph representations:

One big important question in complex systems research is how simple rules can give complex/critical behaviours. Yet despite decades of progress on percolation, predicting which microscopic update rules give rise to macroscopic percolation is an area which has not been investigated before. This investigation has only recently become possible with the emerging field of graph neural networks; prior to this, there was no way to look at vast number of rules and understand which rules give complexity/criticality. Here, GNNs can allow us to tackle this problem for the first time. By converting update rules into graph structures and asking:

- Which rule families necessarily yield percolation or criticality?
- What structural features of a rule decide its universality class?
- Can we diagnose percolation without ever running the dynamics?

To do this, we need to:

1. Select which rules to model as graphs (as there are many different rules that percolate), or find a generic way to represent all rules as graphs.
2. Creating a code that automatically maps rules to graph representations.
3. Simulate the rules and collect classes(percolates or not), and gather statistics on the final grid of the simulation using the library created in step 1
4. Through optimised code acquired by 1-2-3, create a comprehensive dataset of (rule graph, class) pairs
5. Using explainable AI methods(Embedding space investigation, attention scores, etc.) to investigate which features in the rule give rise to each class.
6. Using generative GNN methods to create critical systems.

To better understand this intriguing direction, imagine a periodic checkerboard where black and white boxes are randomly distributed. In one time step, each site on the checkerboard evolves according to its n neighbours, with each neighbour influencing it in a different way. For example, consider a black site with four neighbours—left, right, and top neighbours are white, while the bottom neighbour is black. Suppose the rule specifies that the black site will turn white in the next time step with probability p_left + p_right + p_up, where these probabilities are defined by the rule.

Since we aim to determine whether such a rule leads to critical behaviour, we can train graph neural networks using graphs associated with these local rules. A simple graph representation of such a rule is one where the central node is connected to its four neighbours, with edge features corresponding to the associated probabilities (p_left, p_right, p_up, p_bottom). Combined with the first part of the project (creating a percolation library), this will give us the ground to create the dataset of rule graphs and their percolation/critical properties.

3– Cellular automata ML classification:

Cellular automata (CA) are celebrated as minimalist laboratories for emergence: from a handful of local symbols and update rules, they reproduce patterns as diverse as crystal growth, traffic jams, and universal computation. Four canonical classes—uniform, periodic, chaotic, and complex—summarise these behaviours, yet we still lack an understanding of why certain rules belong to these classes. The fundamental challenge, then, is to explain why only some rules yield complexity, while others collapse into uniformity, periodicity, or chaos. By translating each rule into a graph and analysing these graphs with GNNs, we aim to uncover the structural hallmarks of complexity.

One way to convert these rules into graphs is as follows: since each state evolves either in a probabilistic or deterministic manner, the rules can be represented as Markov chains. In the deterministic case, the Markov chain reduces to a single edge between two nodes; in the

probabilistic case, a node can have multiple outgoing edges, with edge features corresponding to transition probabilities. Given such networks of state transitions for each rule, we can construct graphs that represent the rule dynamics. By creating a dataset of (graph, CA class) pairs, we can train a GNN to learn the relationship between structural features and classes. As in our other projects, we will also employ generative learning to produce complex rules using our GNNs. Although we will be adding more tasks as we progress in our project, some initial tasks will be:

1. Select which Cellular Automata(CA) to model as graphs and find the optimal sampling strategy.
2. Implementing the Hamming distance code by following a recent paper to classify these CA into the 4 classes
3. Creating a code that automatically maps rules to graph representations.
4. Through optimised code acquired by 1-2-3, create a comprehensive dataset of (CA graph, class type) pairs.
5. Implementing a GNN on this dataset to learn graph representations
6. Using explainable AI methods(Embedding space investigation, attention scores, etc) to investigate which features in the CA give rise to each class
7. Using generative GNN methods to complex CAs.

4– Predicting criticality and physical features from the rule sets governing physical systems :

In physics—just as in social networks or biology—global behaviour emerges from local interactions. For example, whether a material becomes magnetised is determined by its local interaction rules, compactly written in the Hamiltonian. Other hallmark critical phenomena follow the same pattern: a liquid and its vapour merge at the fluid critical point, superconductors lose all electrical resistance below their transition temperature, disordered alloys begin conducting once metallic clusters form a spanning network, and ferroelectrics flip their collective polarisation at a Curie point. Similar to percolation, magnetisation can be viewed as a local-probability update process described by the Hamiltonian and its partition function: some rule sets produce magnetisation, others do not.

Exact solutions exist for special systems exhibiting criticality, such as the Ising and Potts models, but these cover only a narrow subset of possible Hamiltonians. For a general rule set (a Hamiltonian with arbitrary combinations of kinetic and potential terms), we still cannot say, in advance, whether the system will be magnetised or remain disordered. To close this gap, we will:

1. **Convert each Hamiltonian into a graph** whose structure captures its local interactions.
2. **Collect labels** indicating global properties (e.g., magnetised vs non-magnetised).
3. **Train graph neural networks** to learn the mapping from local rules to global outcomes.

Using this common pipeline with the other projects, we aim to isolate which ingredients in a Hamiltonian drive magnetisation and other critical phenomena and understand why local interactions give rise to the collective behaviour we observe every day.

1. Select which Hamiltonians to model as graphs and develop a sampling strategy.
2. Creating a code that automatically maps rules to graph representations.
3. Simulate the rules and collect classes(percolates or not), and gather statistics on the final grid of the simulation using the library created in step 1
4. Through optimised code acquired by 1-2-3, create a comprehensive dataset of (Hamiltonian, class) pairs.
5. Using explainable AI methods(Embedding space investigation, attention scores, etc) to investigate which features in the rule give rise to each class.
6. Using generative GNN methods to create critical systems.

Note: For each of the projects, the participants will create reports and supporting code, as well as pedagogical Jupyter notebooks, to help everyone involved coordinate effectively.

I hope you recognise the universal scope of our investigation, which spans probability theory (percolation), physics (criticality), computer science (cellular automata), and the broader field of complex systems. Now, to further motivate you on these interesting topics, let us provide some references. We believe that these topics are certainly accessible to anyone with curiosity and dedication, including non-physicists. Moreover, we will support you every step of the way to help you learn and grow in these areas. Let me share some PDFs and videos that I think will be useful for you.

– For percolation –

Applications of Percolation Theory by Muhammad Sahimi (2nd edition) — please look into the first 3 chapters; this should be enough to familiarise yourself with percolation for the initial interview.

Percolation Theory Using Python by **Anders Malthe-Sørenssen**
(Also, thankfully, Prof. Lee shared his class notebook on percolation—please check that and play around with it as well.)
Our recent Game of Life paper, showing how percolation can appear in deterministic and cellular automata systems: https://arxiv.org/abs/2411.07189

Some fun/informative videos:

- https://www.youtube.com/watch?v=a-767WnbaCQ&t=183s
- https://www.youtube.com/watch?v=baOizaPEalg
- https://www.youtube.com/watch?v=AY_B9gk18Mw
- https://www.youtube.com/watch?v=vaUG9lFs8nA

– For Cellular Automata –

Our recent Game of Life paper, showing how percolation can appear in deterministic and cellular automata systems: https://arxiv.org/abs/2411.07189

A recent paper showing an algorithm to classify these systems into 4 classes: https://arxiv.org/html/2407.06175v1#bib.bib14

Some fun/informative videos:

- https://www.youtube.com/watch?v=XcB_7jv98uE&t=140s (first 5 mins)
- https://www.youtube.com/watch?v=xP5-iIeKXE8
- https://www.youtube.com/watch?v=Kk2MH9O4pXY
- https://www.youtube.com/watch?v=KZeIEiBrT_w
- https://www.youtube.com/watch?v=Ws63I3F7Moc&t=1s

– For Ising Model –

Some fun/informative videos:

- https://www.youtube.com/watch?v=AY_B9gk18Mw
- https://www.youtube.com/watch?v=1WB4ovaNepg
- https://www.youtube.com/watch?v=hjGFp7lMi9A
- https://www.youtube.com/watch?v=vwLb3XlPCB4&t=305s
- https://www.youtube.com/watch?v=fi-g2ET97W8

– For criticality in general –

Some fun/informative videos:

- https://www.youtube.com/watch?v=hjGFp7lMi9A
- https://www.youtube.com/watch?v=vwLb3XlPCB4&t=305s

– For graph neural networks –

https://web.stanford.edu/class/cs224w/

Hamilton, W. L. (2020). Graph Representation Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning

---

I hope I managed to spark at least some excitement toward our creative and impactful journey/research. Please select which projects you want to contribute to and familiarise yourself with the content to prepare yourself for our upcoming discussion. Accordingly, we will start distributing the workload and begin working on our exciting projects together.

Best Regards,

Hakan Akgun
Xianquan Yan
Prof. Ching Hua Lee
Prof. N. Duane Loh