Towards new approaches to cluster detection for cosmology

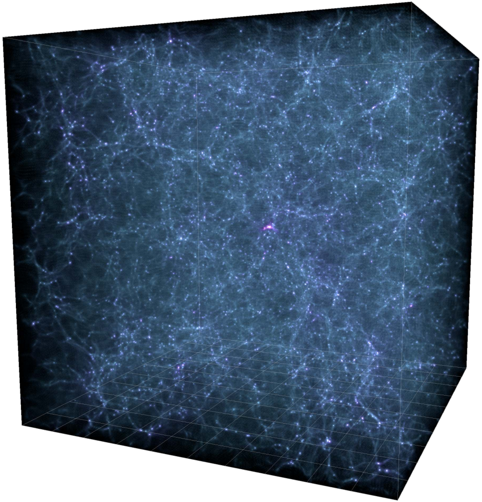
Vincent Reverdy
[vincent.reverdy@lapp.in2p3.fr]
Researcher in Computer Science and Numerical Cosmology
CNRS - French National Centre for Scientific Research
LAPP - Laboratoire d'Annecy de Physique des Particules

November 18th, 2022

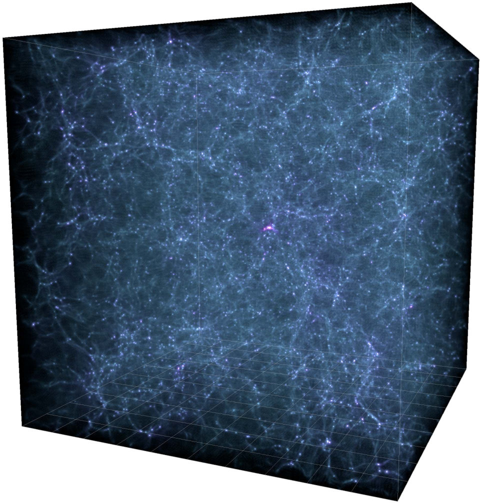
Table of contents

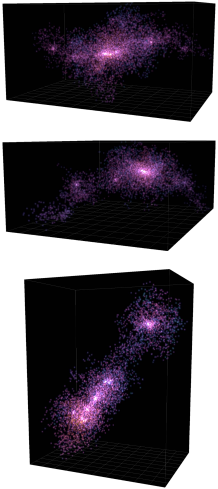
Galaxy clusters

Halos in simulations

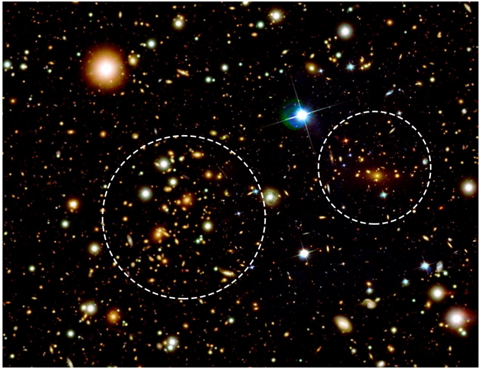


Halos in simulations

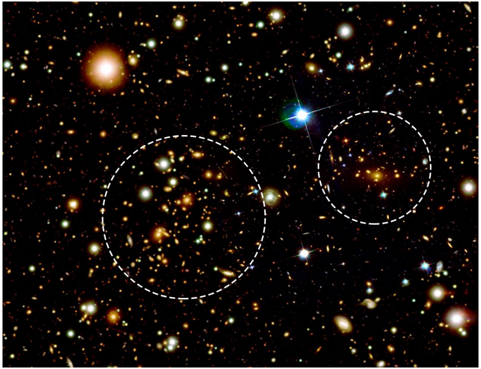




Galaxy clusters in observations



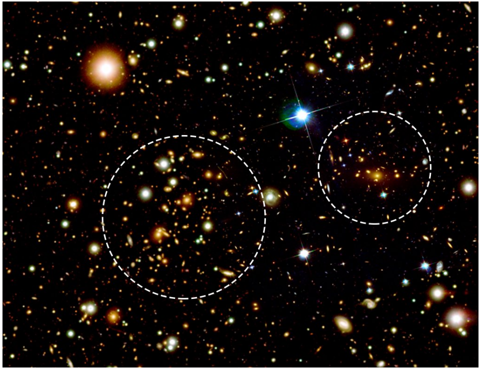
Galaxy clusters in observations



Observational cosmology using galaxy clusters

Galaxy clusters as building blocks of cosmological analyses.

Galaxy clusters in observations



Observational cosmology using galaxy clusters

Galaxy clusters as building blocks of cosmological analyses.

Main problem of this presentation in the context of LSST

How to detect galaxy clusters?

Galaxy cluster detection

Clusters in simulations

- Input: particles

Simulation cluster detectors

- Spherical Over Density (SOD)
- Friends-of-Friends (FoF)
- ...

Clusters in observations

- Input: sources/galaxies

Observational cluster detectors for LSST

- redMaPPer (Rykoff et al. 2013)
- WaZP (Benoist 2014)
- ...

Galaxy cluster detection

Clusters in simulations

- Input: particles

Clusters in observations

- Input: sources/galaxies

Simulation cluster detectors

- Spherical Over Density (SOD)
- Friends-of-Friends (FoF)
- ...

Observational cluster detectors for LSST

- redMaPPer (Rykoff et al. 2013)
- WaZP (Benoist 2014)
- ...

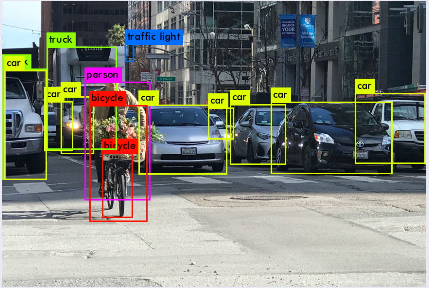
Improving cluster detection for observational cosmology

Can we use machine-learning to improve cluster detection for observational cosmology?

Clusters and computers

Using deep neural networks to detect galaxy clusters

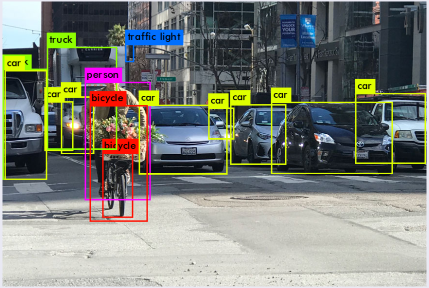
Idea: try to adapt YOLO



You Only Look Once (Redmon et al. 2015)

Using deep neural networks to detect galaxy clusters

Idea: try to adapt YOLO



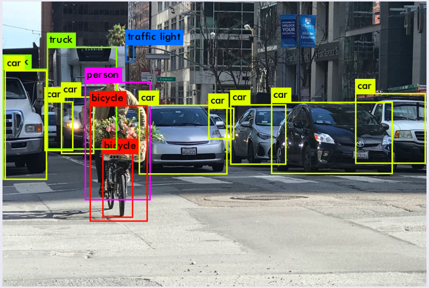
You Only Look Once (Redmon et al. 2015)

Technical problems

- Ultra large images
- More than 3 colors

Using deep neural networks to detect galaxy clusters

Idea: try to adapt YOLO



You Only Look Once (Redmon et al. 2015)

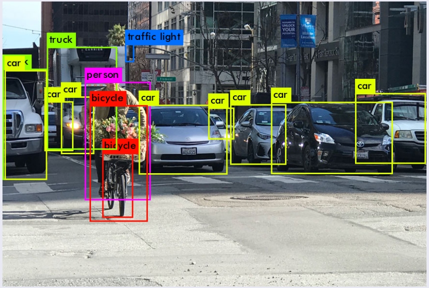
Technical problems

- Ultra large images
- More than 3 colors

Scientific problems

Using deep neural networks to detect galaxy clusters

Idea: try to adapt YOLO



You Only Look Once (Redmon et al. 2015)

Technical problems

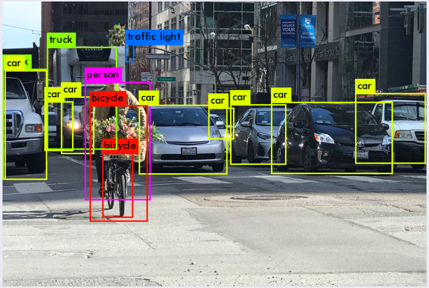
- Ultra large images
- More than 3 colors

Scientific problems

- Training phase: implicit dependency on cluster detection algorithm used in simulations (FoF)

Using deep neural networks to detect galaxy clusters

Idea: try to adapt YOLO



You Only Look Once (Redmon et al. 2015)

Technical problems

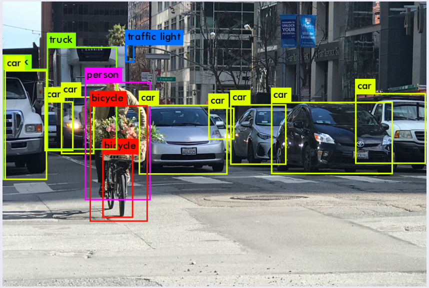
- Ultra large images
- More than 3 colors

Scientific problems

- Training phase: implicit dependency on cluster detection algorithm used in simulations (FoF)
- Performance depends on image rescaling / color treatment

Using deep neural networks to detect galaxy clusters

Idea: try to adapt YOLO



You Only Look Once (Redmon et al. 2015)

Technical problems

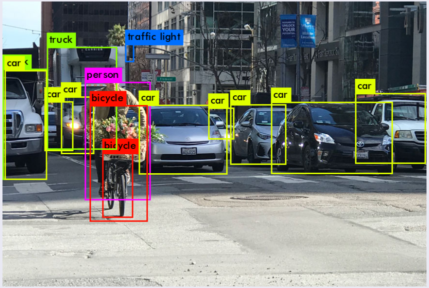
- Ultra large images
- More than 3 colors

Scientific problems

- Training phase: implicit dependency on cluster detection algorithm used in simulations (FoF)
- Performance depends on image rescaling / color treatment
- Blackbox effect: hard to understand, interpret, and adjust performance

Using deep neural networks to detect galaxy clusters

Idea: try to adapt YOLO



You Only Look Once (Redmon et al. 2015)

Technical problems

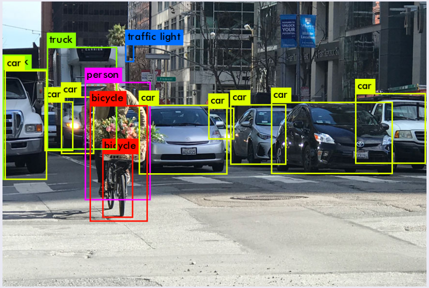
- Ultra large images
- More than 3 colors

Scientific problems

- Training phase: implicit dependency on cluster detection algorithm used in simulations (FoF)
- Performance depends on image rescaling / color treatment
- Blackbox effect: hard to understand, interpret, and adjust performance
- Risk of circular dependency: may just learn how to invert semi-analytical prescription

Using deep neural networks to detect galaxy clusters

Idea: try to adapt YOLO



You Only Look Once (Redmon et al. 2015)

Technical problems

- Ultra large images
- More than 3 colors

Scientific problems

- Training phase: implicit dependency on cluster detection algorithm used in simulations (FoF)
- Performance depends on image rescaling / color treatment
- Blackbox effect: hard to understand, interpret, and adjust performance
- Risk of circular dependency: may just learn how to invert semi-analytical prescription

Direction of investigation

Physics-guided machine-learning?

A problem of definition

The fundamental problem of clusters

- No universal definition: many definitions and parameterization accross simulations and observations
- Computers require definitions: implicit or explicit parameterization in algorithms
- Algorithms $\Leftrightarrow$ Definitions

A problem of definition

The fundamental problem of clusters

- No universal definition: many definitions and parameterization accross simulations and observations
- Computers require definitions: implicit or explicit parameterization in algorithms
- Algorithms $\Leftrightarrow$ Definitions

Physics-based definitions

- Physical distance
- Gravitational potential
- Virialized structures
- ...

A problem of definition

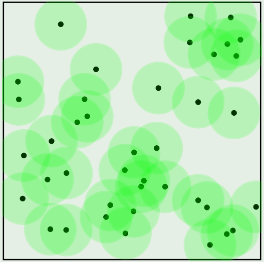
The fundamental problem of clusters

- No universal definition: many definitions and parameterization accross simulations and observations
- Computers require definitions: implicit or explicit parameterization in algorithms
- Algorithms ⇔ Definitions

Physics-based definitions

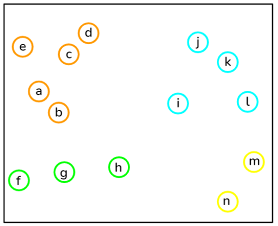
- Physical distance
- Gravitational potential
- Virialized structures
- . . .

Example of FoF: linking-length

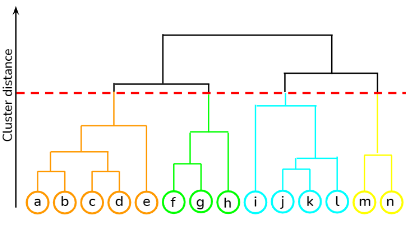


Trees as a universal representation for clustering results

Hierarchical clustering

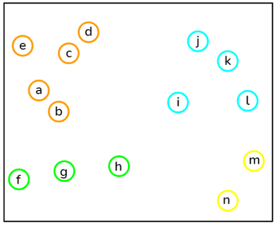


Cluster dendrograms

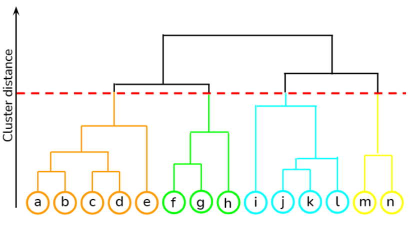


Trees as a universal representation for clustering results

Hierarchical clustering



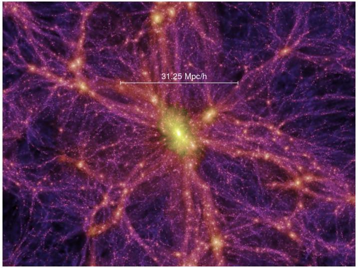
Cluster dendrograms

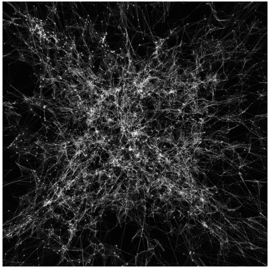


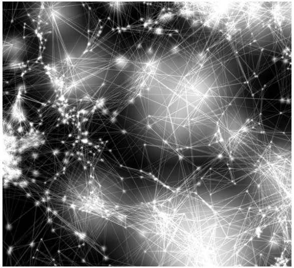
A universal representation

- Traditional galaxy cluster finders ⇒ 3-level trees (root/clusters/sources)
- With subhalo finders ⇒ 4-level trees (root/clusters/subclusters/sources)
- Hierarchical finders ⇒ N-level trees (root/supersuperclusters/superclusters/clusters/. . . /sources)

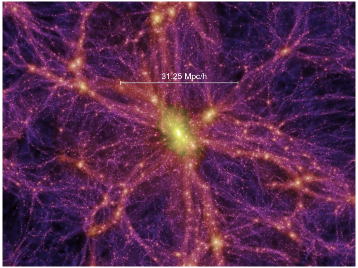
Graph as a numerical representation of the cosmic web

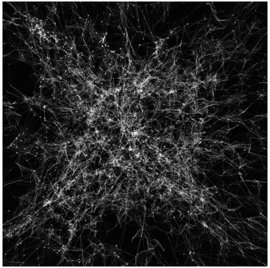


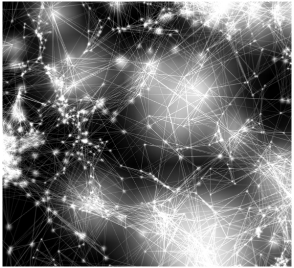




Graph as a numerical representation of the cosmic web







Representing the cosmic web as a graph

- Vertices: particles, sources
- Edges: physical parameterization (physical distance. . .)
- Going beyond: fuzzy graphs to take into account error bars

A computer science perspective on galaxy clustering: graphs and trees

Reframing of the galaxy clustering problem

- Cosmic web $\Leftrightarrow$ Graph
- Galaxy clustering $\Leftrightarrow$ Trees

A computer science perspective on galaxy clustering: graphs and trees

Reframing of the galaxy clustering problem

- Cosmic web ⇔ Graph
- Galaxy clustering ⇔ Trees

Computer science perspective

Galaxy clustering problem ⇔ Computing Trees on Graphs

A computer science perspective on galaxy clustering: graphs and trees

Reframing of the galaxy clustering problem

- Cosmic web $\Leftrightarrow$ Graph
- Galaxy clustering $\Leftrightarrow$ Trees

Computer science perspective

Galaxy clustering problem $\Leftrightarrow$ Computing Trees on Graphs

Discrete vs continuous mathematics

- Comparison of galaxy clustering algorithms $\Rightarrow$ comparison of trees
- Computer science related problem: tree metric and distances

A computer science perspective on galaxy clustering: graphs and trees

Reframing of the galaxy clustering problem

- Cosmic web ⇔ Graph
- Galaxy clustering ⇔ Trees

Computer science perspective

Galaxy clustering problem ⇔ Computing Trees on Graphs

Discrete vs continuous mathematics

- Comparison of galaxy clustering algorithms ⇒ comparison of trees
- Computer science related problem: tree metric and distances

Research direction

Playing games on the cosmic web: computing galaxy clustering trees on the cosmic web graph

- Well-framed computer science problem
- Bridge between numerical cosmology, computer science, and discrete mathematics
- Computer science tools available

Game trees

Game trees: starting with Tic-Tac-Toe

Game trees

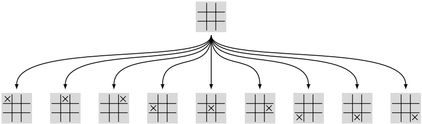
A universal way to represent sequential N-player(s) games with perfect information.

Game trees: starting with Tic-Tac-Toe

Game trees

A universal way to represent sequential N-player(s) games with perfect information.

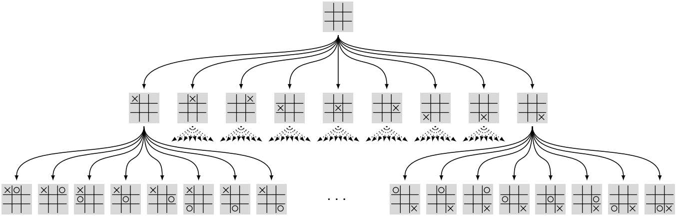
Game trees: starting with Tic-Tac-Toe



Game trees

A universal way to represent sequential N-player(s) games with perfect information.

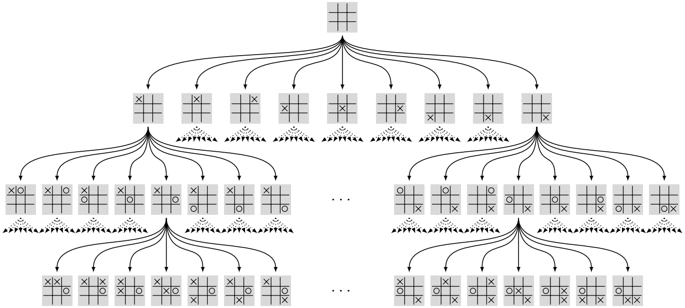
Game trees: starting with Tic-Tac-Toe



Game trees

A universal way to represent sequential N-player(s) games with perfect information.

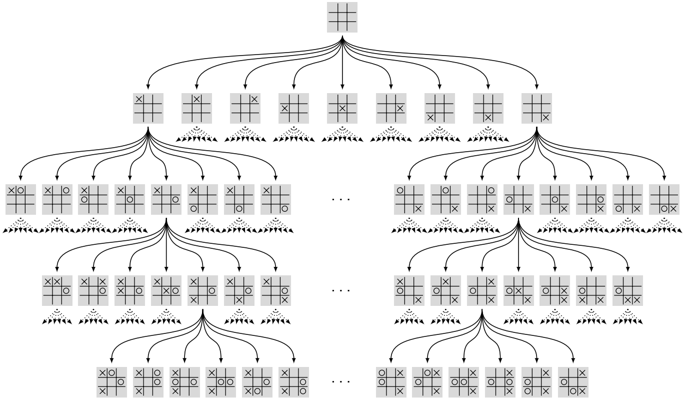
Game trees: starting with Tic-Tac-Toe



Game trees

A universal way to represent sequential N-player(s) games with perfect information.

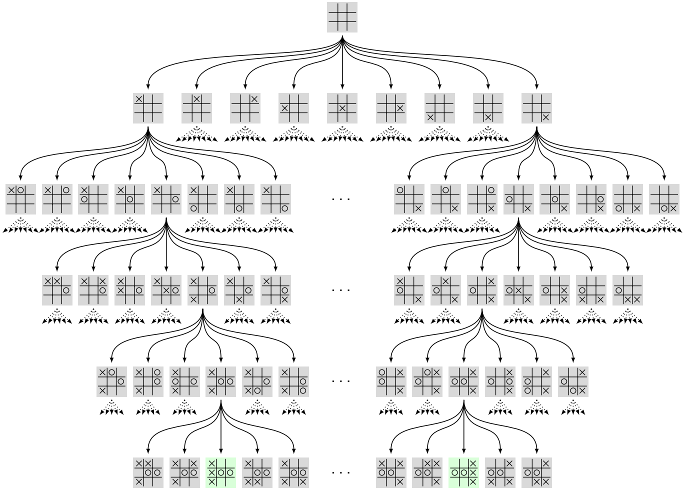
Game trees: starting with Tic-Tac-Toe



Game trees

A universal way to represent sequential N-player(s) games with perfect information.

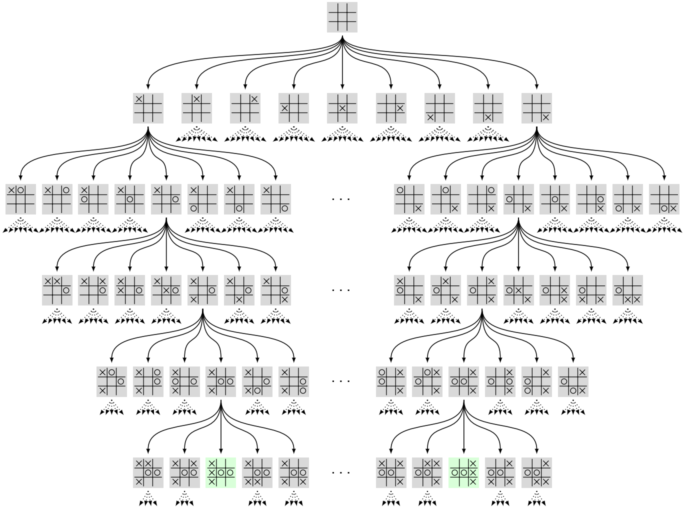
Game trees: starting with Tic-Tac-Toe



Game trees

A universal way to represent sequential N-player(s) games with perfect information.

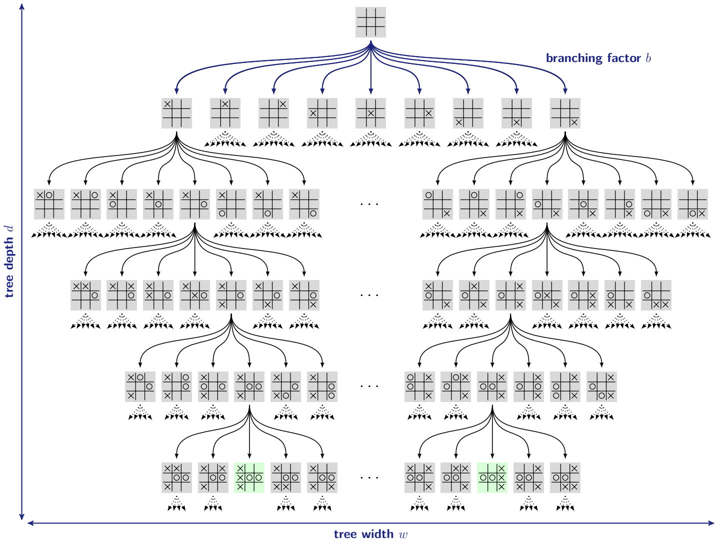
Game trees: starting with Tic-Tac-Toe



Game trees

A universal way to represent sequential N-player(s) games with perfect information.

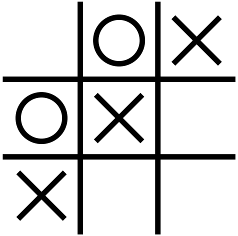
Game trees: starting with Tic-Tac-Toe

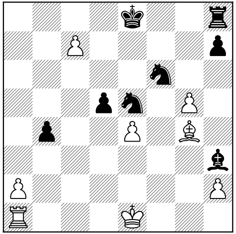


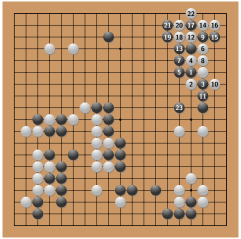
Game trees

A universal way to represent sequential N-player(s) games with perfect information.

Game complexity







Tic-Tac-Toe

- Board size $\Rightarrow 3^2 = 9$
- Average game length $\Rightarrow 9$
- Average branching factor $\Rightarrow 4$
- Number of games $\Rightarrow 255\,168$

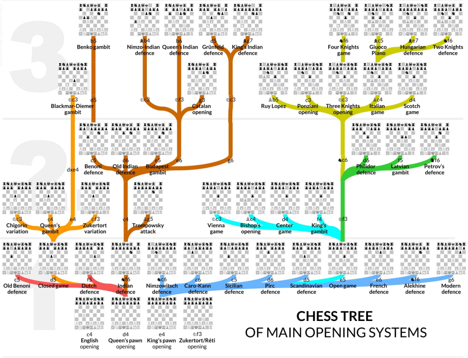
Chess

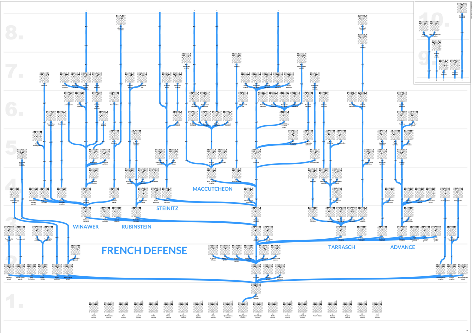
- Board size $\Rightarrow 8^2 = 64$
- Average game length $\Rightarrow 70$
- Average branching factor $\Rightarrow 35$
- Complexity $\Rightarrow 35^{70} \approx 10^{64}$

Go

- Board size $\Rightarrow 19^2 = 361$
- Average game length $\Rightarrow 150$
- Average branching factor $\Rightarrow 250$
- Complexity $\Rightarrow 250^{150} \approx 10^{360}$

Playing chess algorithmically





Brute-forcing

Game tree still tractable: average depth $d = 70$, average branching factor $b = 35$

The problem with the game of Go

Intractability

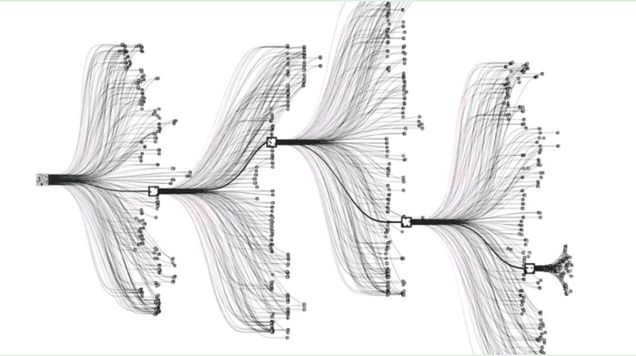
Go's game tree average depth $d = 150$, average branching factor $b = 250$

The problem with the game of Go

Intractability

Go's game tree average depth $d = 150$, average branching factor $b = 250$

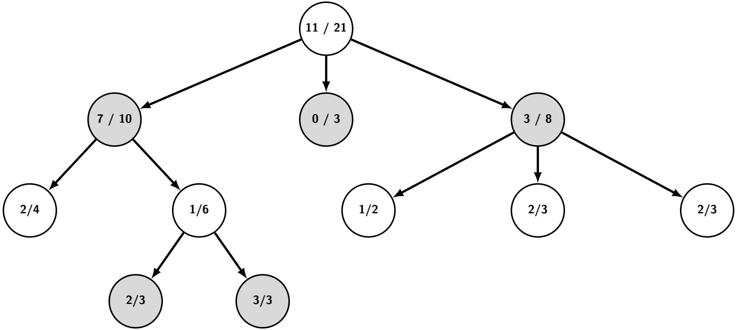
Deepmind's AlphaGo/AlphaZero



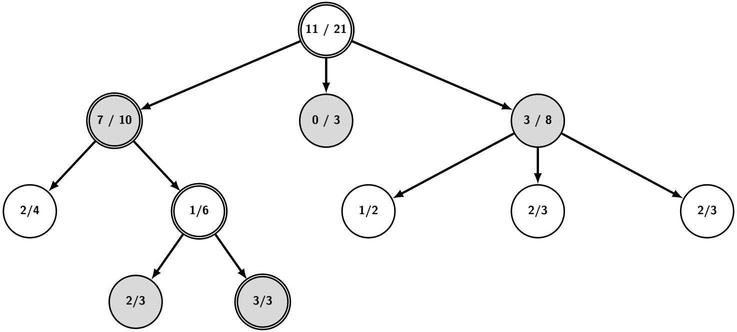
Breakthrough in 2017 (David Silver et al.)

MCTS and AlphaZero

Monte-Carlo-Tree-Search (MCTS) overview



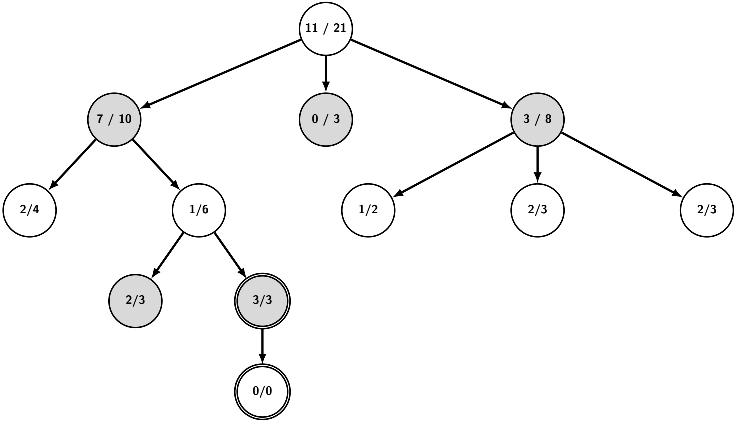
Monte-Carlo-Tree-Search (MCTS) overview



Selection

Select a leaf node.

Monte-Carlo-Tree-Search (MCTS) overview



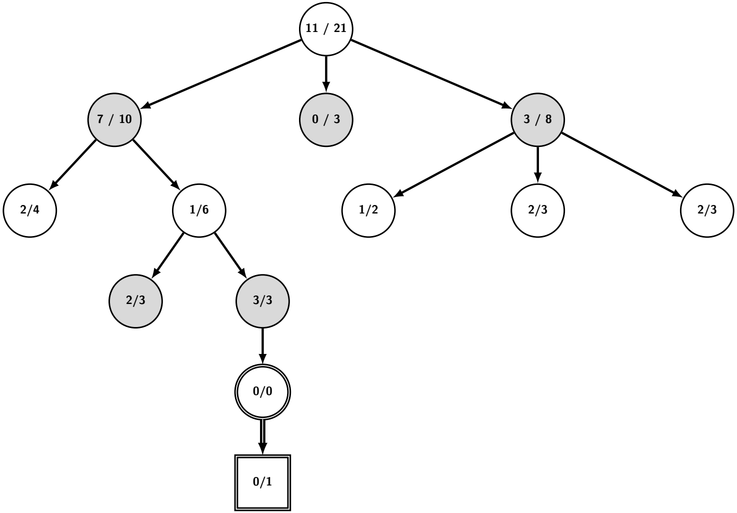
Selection

Select a leaf node.

Expansion

Create one or more children nodes and select one of them.

Monte-Carlo-Tree-Search (MCTS) overview



Selection

Select a leaf node.

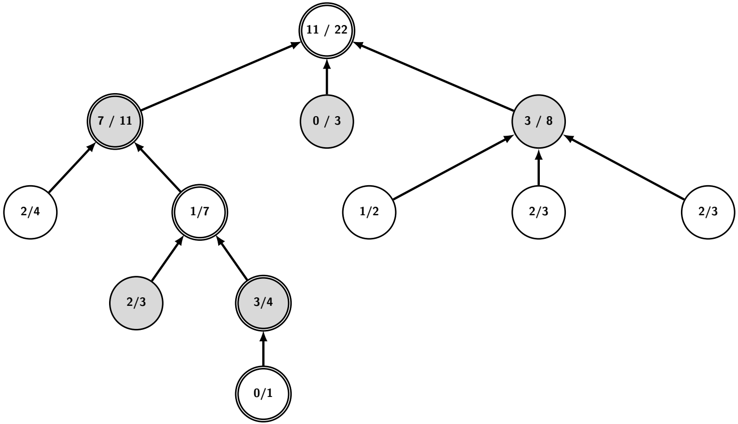
Expansion

Create one or more children nodes and select one of them.

Simulation

Complete a full random playout from the selected child.

Monte-Carlo-Tree-Search (MCTS) overview



Selection

Select a leaf node.

Expansion

Create one or more children nodes and select one of them.

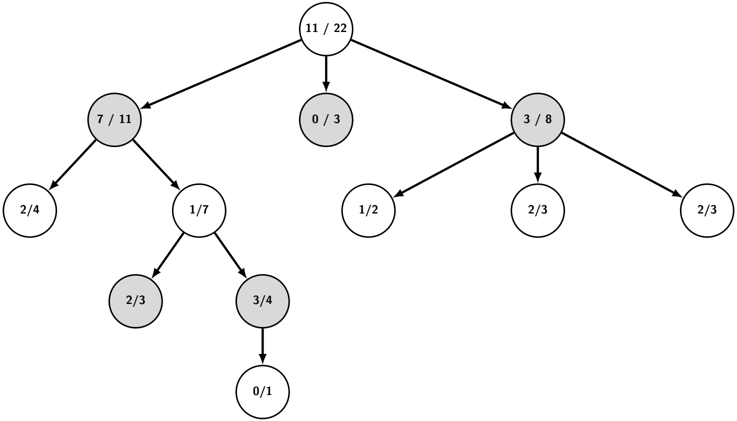
Simulation

Complete a full random playout from the selected child.

Backpropagation

Backpropagate the result of the simulation back to the root.

Monte-Carlo-Tree-Search (MCTS) overview



Selection

Select a leaf node.

Expansion

Create one or more children nodes and select one of them.

Simulation

Complete a full random playout from the selected child.

Backpropagation

Backpropagate the result of the simulation back to the root.

Monte-Carlo-Tree-Search (MCTS) refinements

Simulations

- To get a distribution of results, N_s simulations can be performed at each step
- Simulated playouts can be done in a better way than using random moves

Monte-Carlo-Tree-Search (MCTS) refinements

Simulations

- To get a distribution of results, N_s simulations can be performed at each step
- Simulated playouts can be done in a better way than using random moves

Selection: exploitation vs exploration

Example of the UCT (Upper Confidence Bound 1 applied to Trees) policy:

$$\max_x \left(\frac{w_i(x)}{n_i(x)} + c\sqrt{\frac{\log(N_i(x))}{n_i(x)}} \right)$$

Monte-Carlo-Tree-Search (MCTS) refinements

Simulations

- To get a distribution of results, N_s simulations can be performed at each step
- Simulated playouts can be done in a better way than using random moves

Selection: exploitation vs exploration

Example of the UCT (Upper Confidence Bound 1 applied to Trees) policy:

$$\max_x \left(\frac{w_i(x)}{n_i(x)} + c\sqrt{\frac{\log(N_i(x))}{n_i(x)}} \right)$$

- $w_i \Rightarrow$ number of wins for the node x after the i-th move

Monte-Carlo-Tree-Search (MCTS) refinements

Simulations

- To get a distribution of results, N_s simulations can be performed at each step
- Simulated playouts can be done in a better way than using random moves

Selection: exploitation vs exploration

Example of the UCT (Upper Confidence Bound 1 applied to Trees) policy:

$$\max_x \left(\frac{w_i(x)}{n_i(x)} + c\sqrt{\frac{\log\left(N_i(x)\right)}{n_i(x)}} \right)$$

- $w_i \Rightarrow$ number of wins for the node x after the i-th move
- $n_i \Rightarrow$ number of simulations for the node x after the i-th move

Monte-Carlo-Tree-Search (MCTS) refinements

Simulations

- To get a distribution of results, N_s simulations can be performed at each step
- Simulated playouts can be done in a better way than using random moves

Selection: exploitation vs exploration

Example of the UCT (Upper Confidence Bound 1 applied to Trees) policy:

$$\max_x \left(\frac{w_i(x)}{n_i(x)} + c\sqrt{\frac{\log(N_i(x))}{n_i(x)}} \right)$$

- $w_i \Rightarrow$ number of wins for the node x after the i-th move
- $n_i \Rightarrow$ number of simulations for the node x after the i-th move
- $N_i \Rightarrow$ total number of simulations for the parent node of x after the i-th move

Monte-Carlo-Tree-Search (MCTS) refinements

Simulations

- To get a distribution of results, N_s simulations can be performed at each step
- Simulated playouts can be done in a better way than using random moves

Selection: exploitation vs exploration

Example of the UCT (Upper Confidence Bound 1 applied to Trees) policy:

$$\max_x \left(\frac{w_i(x)}{n_i(x)} + c\sqrt{\frac{\log(N_i(x))}{n_i(x)}} \right)$$

- $w_i \Rightarrow$ number of wins for the node x after the i-th move
- $n_i \Rightarrow$ number of simulations for the node x after the i-th move
- $N_i \Rightarrow$ total number of simulations for the parent node of x after the i-th move
- $c \Rightarrow$ exploration coefficient ($c = \sqrt{2}$)

AlphaZero = Monte-Carlo-Tree-Search + Deep Neural Networks

Monte-Carlo-Tree-Search (MCTS)

- Selection
- Expansion
- Simulation
- Backpropagation

AlphaZero = Monte-Carlo-Tree-Search + Deep Neural Networks

Monte-Carlo-Tree-Search (MCTS)

- Selection
- Expansion
- Simulation
- Backpropagation

Policy deep neural network

- Input: s, the current state of the board (+ optionally previous states)
- Output: p, to select the most probable/promising moves
- General idea: reduce the breadth of the tree search

AlphaZero = Monte-Carlo-Tree-Search + Deep Neural Networks

Monte-Carlo-Tree-Search (MCTS)

- Selection
- Expansion
- Simulation
- Backpropagation

Policy deep neural network

- Input: s, the current state of the board (+ optionally previous states)
- Output: p, to select the most probable/promising moves
- General idea: reduce the breadth of the tree search

Value deep neural network

- Input: s, the current state of the board (+ optionally previous states)
- Output: v, the probability to win
- General idea: reduce the depth of the tree search

AlphaZero = Monte-Carlo-Tree-Search + Deep Neural Networks

Monte-Carlo-Tree-Search (MCTS)

- Selection
- Expansion
- Simulation
- Backpropagation

Policy deep neural network

- Input: s, the current state of the board (+ optionally previous states)
- Output: p, to select the most probable/promising moves
- General idea: reduce the breadth of the tree search

Value deep neural network

- Input: s, the current state of the board (+ optionally previous states)
- Output: v, the probability to win
- General idea: reduce the depth of the tree search

AlphaZero = MCTS + Policy Net + Value Net

Tree search guided by "intuition" and "expert-knowledge".

A clustering game

A clustering game

Game's board

The cosmic-web graph

A clustering game

Game's board

The cosmic-web graph

Game's goal

Extracting a clustering tree from the graph

A clustering game

Game's board

The cosmic-web graph

Game's goal

Extracting a clustering tree from the graph

Game's move

Create a link between a particle and its best neighbouring candidate

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

- Compute the cosmic-web graph $\mathcal{G}$

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

- Compute the cosmic-web graph $\mathcal{G}$
- Compute a reference tree $\mathcal{T}_0$ using a physics-based criteria ϕ (distance, gravitational potential, ...)

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

- Compute the cosmic-web graph $\mathcal{G}$
- Compute a reference tree $\mathcal{T}_0$ using a physics-based criteria ϕ (distance, gravitational potential, ...)
- Extract observational data from simulation: $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$ (RA, DEC, redshift with error bars, simulated images, ...)

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

- Compute the cosmic-web graph $\mathcal{G}$
- Compute a reference tree $\mathcal{T}_0$ using a physics-based criteria ϕ (distance, gravitational potential, ...)
- Extract observational data from simulation: $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$ (RA, DEC, redshift with error bars, simulated images, ...)

Training phase

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

- Compute the cosmic-web graph $\mathcal{G}$
- Compute a reference tree $\mathcal{T}_0$ using a physics-based criteria ϕ (distance, gravitational potential, ...)
- Extract observational data from simulation: $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$ (RA, DEC, redshift with error bars, simulated images, ...)

Training phase

- Play the 1-player game using MCTS $+$ Neural Nets to extract a tree $\mathcal{T}$ from the graph $\mathcal{G}$

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

- Compute the cosmic-web graph $\mathcal{G}$
- Compute a reference tree $\mathcal{T}_0$ using a physics-based criteria ϕ (distance, gravitational potential, ...)
- Extract observational data from simulation: $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$ (RA, DEC, redshift with error bars, simulated images, ...)

Training phase

- Play the 1-player game using MCTS + Neural Nets to extract a tree $\mathcal{T}$ from the graph $\mathcal{G}$
- For each-move use only the information of $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

- Compute the cosmic-web graph $\mathcal{G}$
- Compute a reference tree $\mathcal{T}_0$ using a physics-based criteria ϕ (distance, gravitational potential, ...)
- Extract observational data from simulation: $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$ (RA, DEC, redshift with error bars, simulated images, ...)

Training phase

- Play the 1-player game using MCTS + Neural Nets to extract a tree $\mathcal{T}$ from the graph $\mathcal{G}$
- For each-move use only the information of $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$
- Compute the reward as a tree-distance d between $\mathcal{T}$ and $\mathcal{T}_0$: $d = f(\mathcal{T}, \mathcal{T}_0) \in [0, 1]$

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

- Compute the cosmic-web graph $\mathcal{G}$
- Compute a reference tree $\mathcal{T}_0$ using a physics-based criteria ϕ (distance, gravitational potential, ...)
- Extract observational data from simulation: $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$ (RA, DEC, redshift with error bars, simulated images, ...)

Training phase

- Play the 1-player game using MCTS + Neural Nets to extract a tree $\mathcal{T}$ from the graph $\mathcal{G}$
- For each-move use only the information of $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$
- Compute the reward as a tree-distance d between $\mathcal{T}$ and $\mathcal{T}_0$: $d = f(\mathcal{T}, \mathcal{T}_0) \in [0, 1]$

Playing phase

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

- Compute the cosmic-web graph $\mathcal{G}$
- Compute a reference tree $\mathcal{T}_0$ using a physics-based criteria ϕ (distance, gravitational potential, . . .)
- Extract observational data from simulation: $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$ (RA, DEC, redshift with error bars, simulated images, . . .)

Training phase

- Play the 1-player game using MCTS + Neural Nets to extract a tree $\mathcal{T}$ from the graph $\mathcal{G}$
- For each-move use only the information of $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$
- Compute the reward as a tree-distance d between $\mathcal{T}$ and $\mathcal{T}_0$: $d = f(\mathcal{T}, \mathcal{T}_0) \in [0, 1]$

Playing phase

- Play the 1-player game using MCTS + Neural Nets on real observations using $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$

Using MCTS and Neural Nets to play the clustering game

Preparation using simulations

- Compute the cosmic-web graph $\mathcal{G}$
- Compute a reference tree $\mathcal{T}_0$ using a physics-based criteria ϕ (distance, gravitational potential, ...)
- Extract observational data from simulation: $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$ (RA, DEC, redshift with error bars, simulated images, ...)

Training phase

- Play the 1-player game using MCTS + Neural Nets to extract a tree $\mathcal{T}$ from the graph $\mathcal{G}$
- For each-move use only the information of $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$
- Compute the reward as a tree-distance d between $\mathcal{T}$ and $\mathcal{T}_0$: $d = f(\mathcal{T}, \mathcal{T}_0) \in [0, 1]$

Playing phase

- Play the 1-player game using MCTS + Neural Nets on real observations using $\vec{x} = (x_0, x_1, x_2, \cdots, x_{n-1})$
- No need of observational data on ϕ

Combining the best of both worlds

Physics-guided machine-learning approach

- Works for any physics-based criteria ϕ and observational data $\vec{x}$ (including images and error bars)
- Builds an "intuition" on ϕ using simulations ($\phi \Leftrightarrow$ How does the game board looks like?)
- No black-box effect anymore

Combining the best of both worlds

Physics-guided machine-learning approach

- Works for any physics-based criteria ϕ and observational data $\vec{x}$ (including images and error bars)
- Builds an "intuition" on ϕ using simulations ($\phi \Leftrightarrow$ How does the game board looks like?)
- No black-box effect anymore

Possible cluster definitions based on ϕ

- Physical distance
- Matter density
- Gravitational potential
- Virialization
- Linear combination of all of the above
- ...

Conclusions

Towards the ultimate definition?

Cluster definition collapse in $t = +\infty$

- In $t = +\infty \Rightarrow$ isolated clusters
- All definitions are equivalent

Towards the ultimate definition?

Cluster definition collapse in $t = +\infty$

- In $t = +\infty \Rightarrow$ isolated clusters
- All definitions are equivalent

Simulating the end of time

Towards the ultimate definition?

Cluster definition collapse in $t = +\infty$

- In $t = +\infty \Rightarrow$ isolated clusters
- All definitions are equivalent

Simulating the end of time

- Let simulations run far into the future

Towards the ultimate definition?

Cluster definition collapse in $t = +\infty$

- In $t = +\infty \Rightarrow$ isolated clusters
- All definitions are equivalent

Simulating the end of time

- Let simulations run far into the future
- Detect clusters

Towards the ultimate definition?

Cluster definition collapse in $t = +\infty$

- In $t = +\infty \Rightarrow$ isolated clusters
- All definitions are equivalent

Simulating the end of time

- Let simulations run far into the future
- Detect clusters
- Go back in time to $z = 0$ tracking particles

Towards the ultimate definition?

Cluster definition collapse in $t = +\infty$

- In $t = +\infty \Rightarrow$ isolated clusters
- All definitions are equivalent

Simulating the end of time

- Let simulations run far into the future
- Detect clusters
- Go back in time to $z = 0$ tracking particles
- Use this knowledge to build a reference tree $\mathcal{T}$

Towards the ultimate definition?

Cluster definition collapse in $t = +\infty$

- In $t = +\infty \Rightarrow$ isolated clusters
- All definitions are equivalent

Simulating the end of time

- Let simulations run far into the future
- Detect clusters
- Go back in time to $z = 0$ tracking particles
- Use this knowledge to build a reference tree $\mathcal{T}$
- Train the algorithm on $\mathcal{T}$

Towards the ultimate definition?

Cluster definition collapse in $t = +\infty$

- In $t = +\infty \Rightarrow$ isolated clusters
- All definitions are equivalent

Simulating the end of time

- Let simulations run far into the future
- Detect clusters
- Go back in time to $z = 0$ tracking particles
- Use this knowledge to build a reference tree $\mathcal{T}$
- Train the algorithm on $\mathcal{T}$

Analysis

- Play the game using this criterion
- Use computer science and discrete mathematics to compare $\mathcal{T}_i = f(\phi_i)$

Conclusions

Limitations of machine-learning approaches

- Black-box approaches, hard to interpret and adjust in terms of physics
- Risk of circular definitions

Cluster detection as a computer science problem

- Cosmic-web $\Leftrightarrow$ Graph
- Clusters $\Leftrightarrow$ Trees
- Galaxy clustering problem $\Leftrightarrow$ Computing Trees on Graphs

The game of cluster detection

- MCTS + Deep Neural Networks = physics-guided machine-learning approach
- Work with arbitrary physics criterion ϕ and observational data $\vec{x}$

A starting project!

Reach out if you're interested: vincent.reverdy@lapp.in2p3.fr