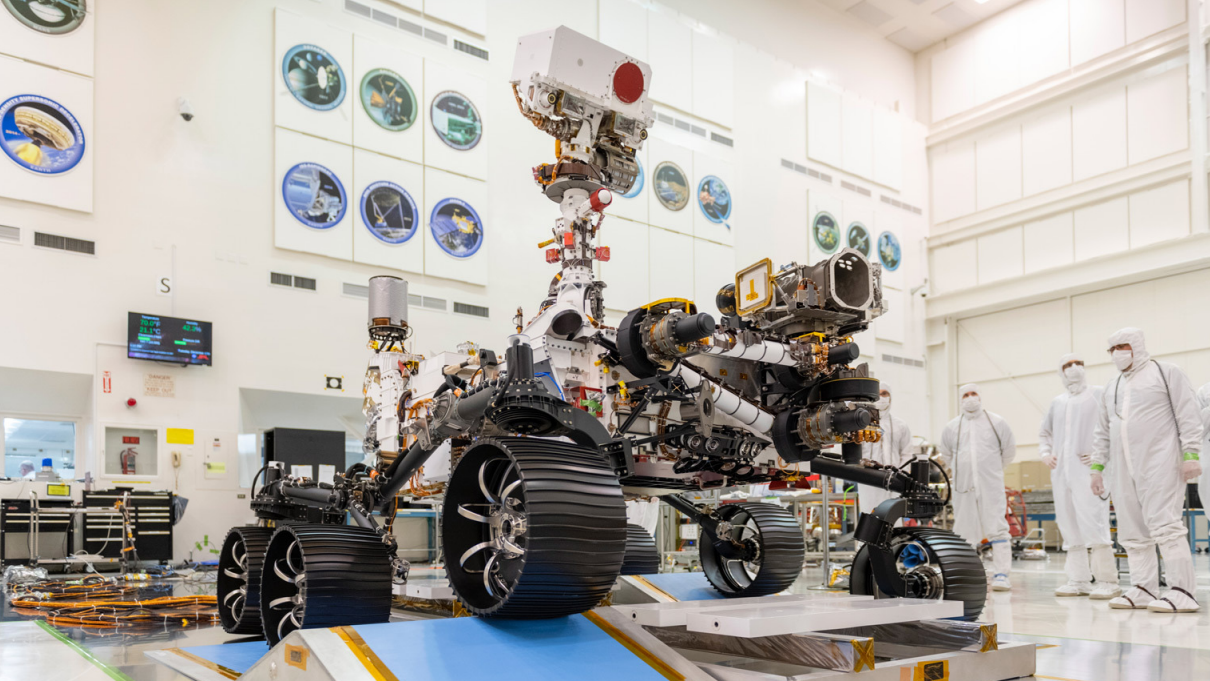
Environment Learning for Robot Path-Planning

Lucas Saldyt Arizona State University



Core Goals & Concepts

Learning

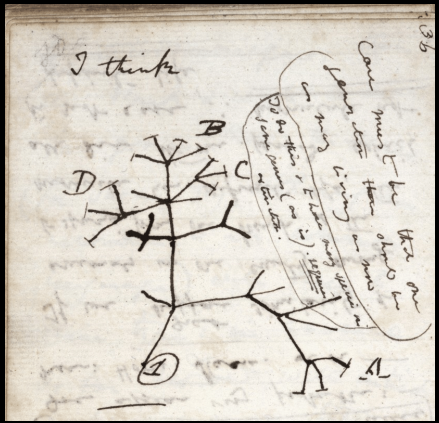
- Flexibility: adapt to new tasks

Generalization

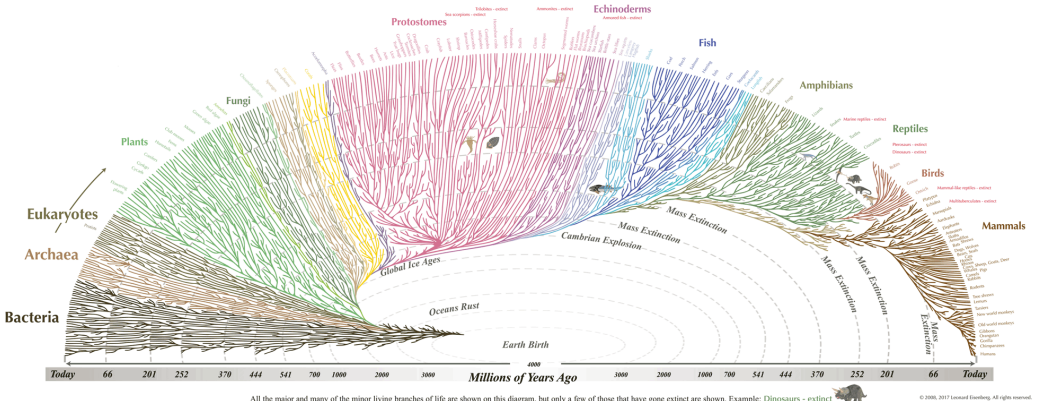
- Learn overall task proficiency?

Specialization

- Learn environment details?



Where is this image from?

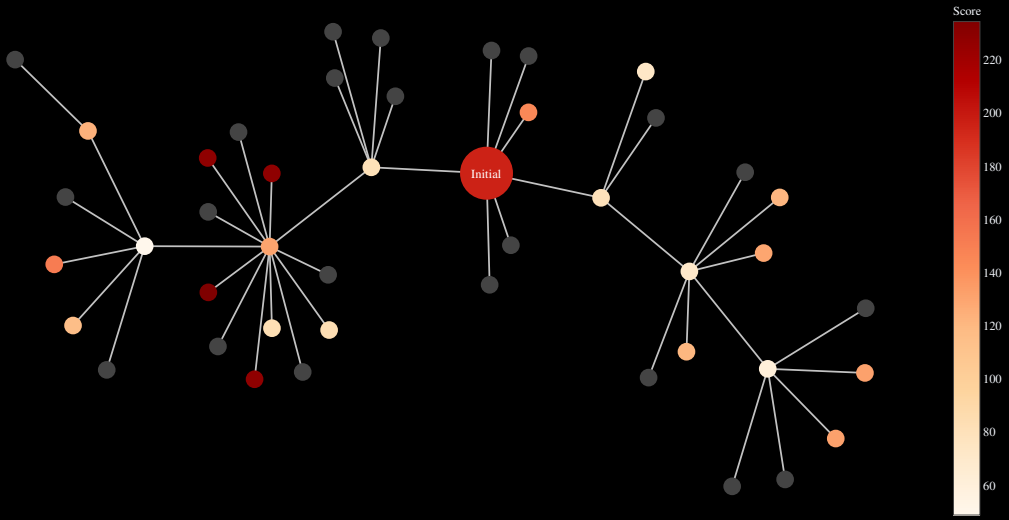


All the major and many of the minor living branches of life are shown on this diagram, but only a few of those that have gone extinct are shown. Example: Dinosaurs - extinct

Natural evolution

Computer Programs instead of species

Evolutionary Programming



Evolved Computer Programs

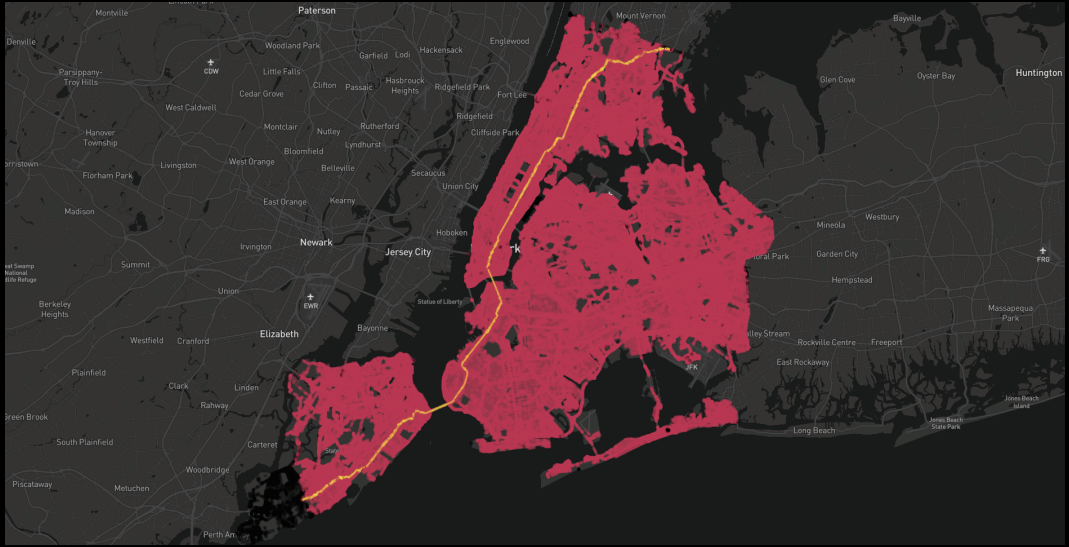
Problems Considered

Path Planning

- Graph-based
- Sample-based (left)
- Dynamic (i.e. Mars)
- Multiagent (i.e. Amazon)

Symbolic Regression
Neural Arch. Search

Graph search w/ Dijkstra (New York)



Graph search w/ A* (New York)

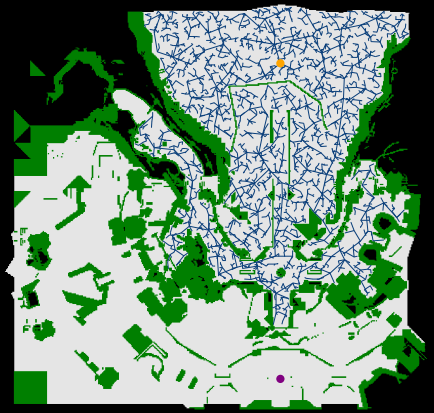
Graph-based Algorithms (Kiev, Ukraine)

- Breadth-first (BFS)
- Depth-first (DFS)
- Dijkstra/A*

- Goal-directed
- Separator-based
- Hierarchical
- Bounded Hop
- Hybrid Techniques

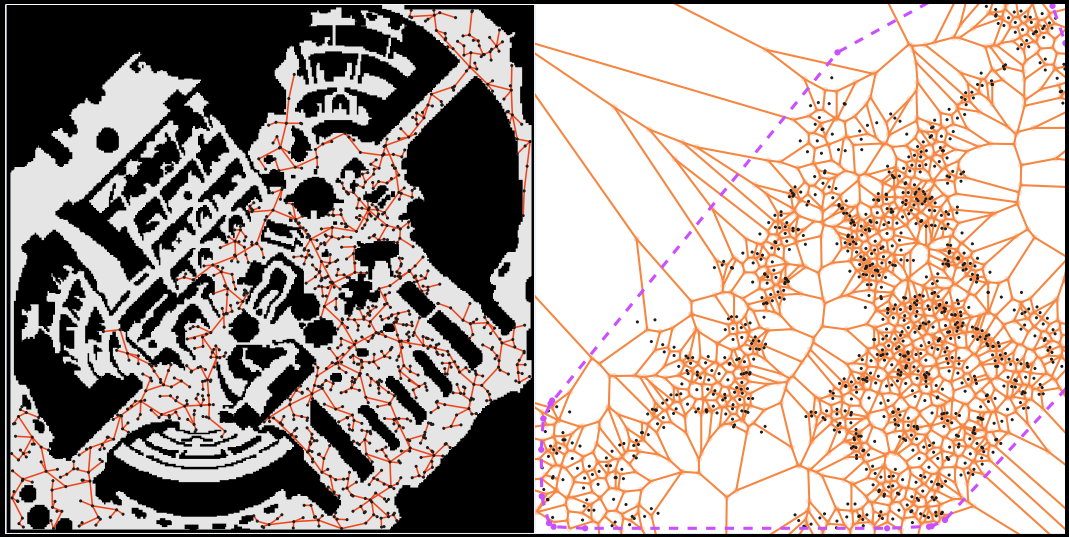
- Bast et al., 2016

Sampling-based search

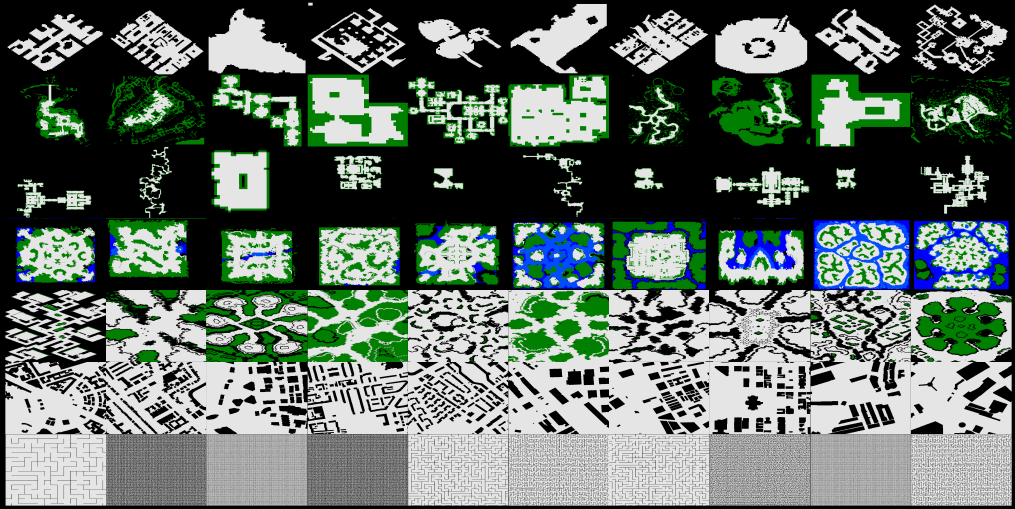


- Rapidly Exploring Random Trees (RRTs)
  - RRT, RRT* , Informed RRT*
  - Constrained (RRT-Blossom)
  - Multi-phase (A*-RRT, Theta*-RRT)
  - Dynamic (RT-RRT*, RRTX, RRT)
  - LaValle and Kuffner Jr, 2001...
- Probabilistic Road Maps (PRMs)
  - Combine sampling & graph-based
  - Kavraki et al., 1996

RRT* (Baldur's Gate)

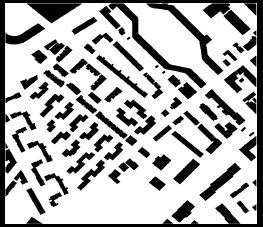


Maps Overview (Sturtevant, 2012)



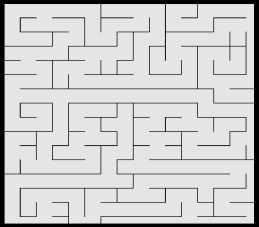
Maps Detail (Sturtevant, 2012)

Streetmaps: Boston_0_1024



Streetmaps: Shanghai_0_1024

Maze: maze512-32-9

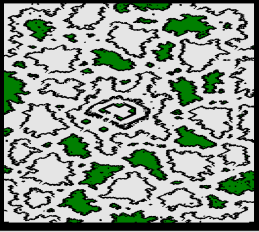


Baldurs Gate: AR0300SR

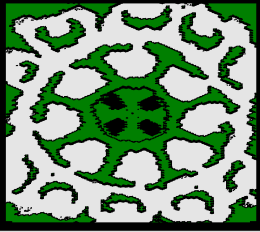
Dragon Age: orz100d

Dragon Age: ht_mansion2b

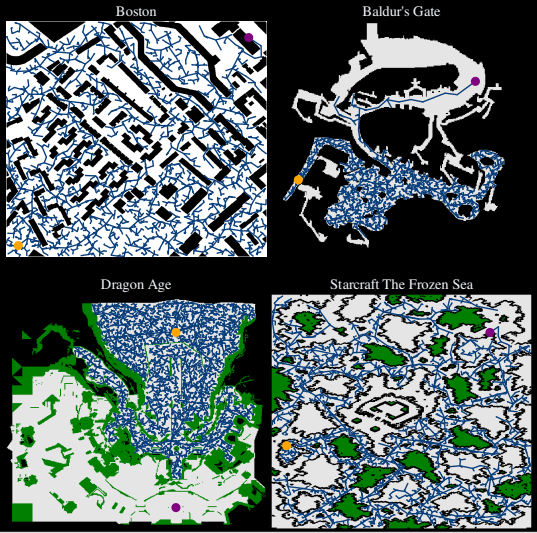
Starcraft: TheFrozenSea



Starcraft: WheelofWar

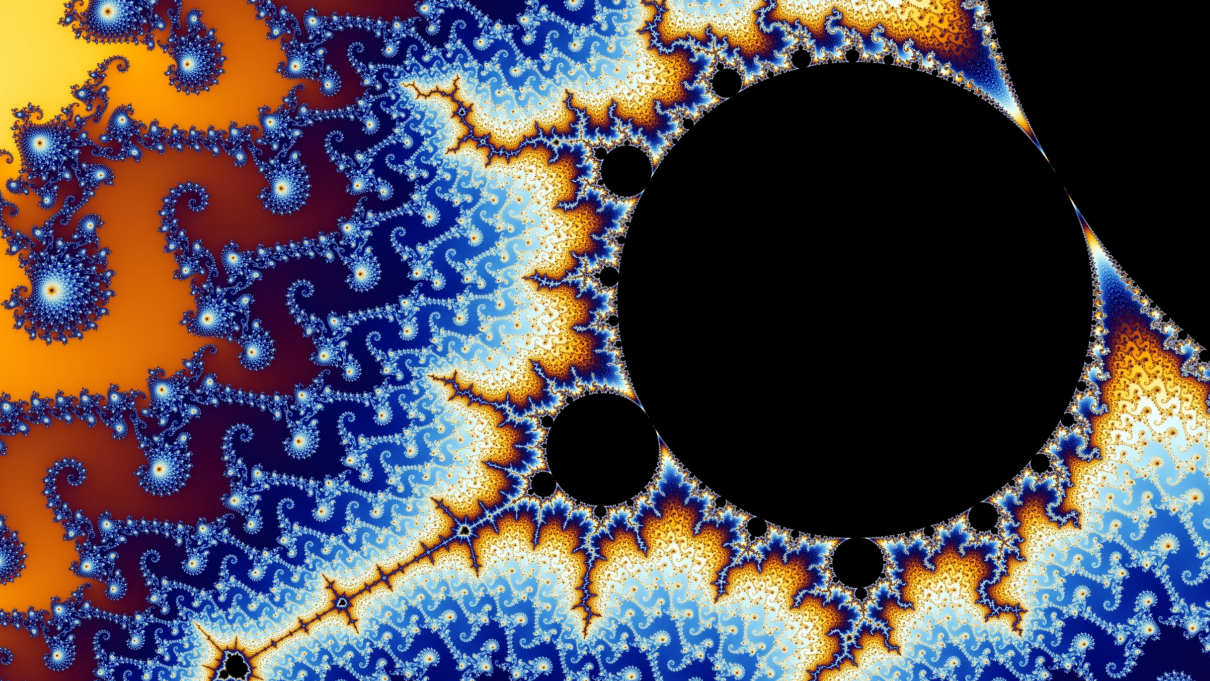


Specialization Teaser



There is no “silver bullet” algorithm for solving all path planning problems. Heuristics that lead to massive speed-up in one scenario might be detrimental in others. Also, algorithmic parameters are mostly ad-hoc and correctly tuning them to a specific environment might drastically increase performance.

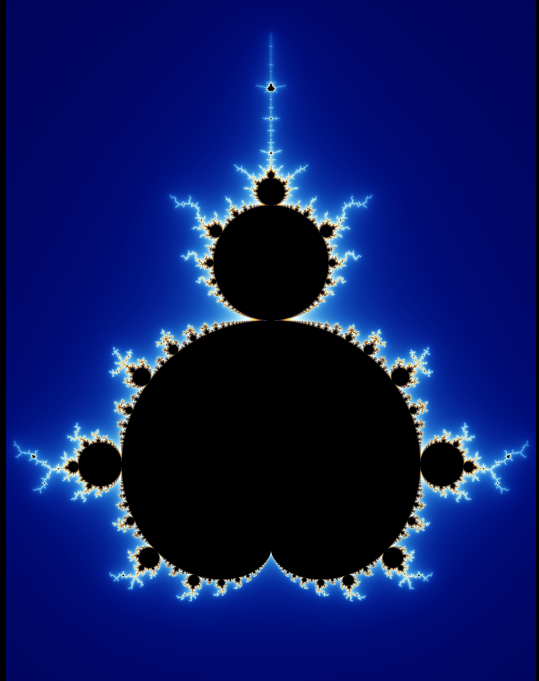
Nikolaus Correll



Program Synthesis

Kolmogorov

- Searching program space *exhaustively* is impossible (Turing: Halting Problem)



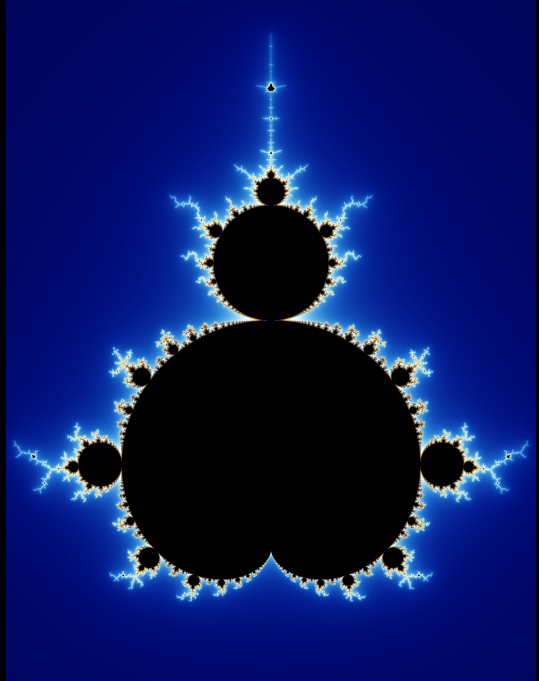
Program Synthesis

Kolmogorov

- Searching program space *exhaustively* is impossible (Turing: Halting Problem)

Schmidhuber

- Optimal Ordered Solvers



Program Synthesis

Kolmogorov

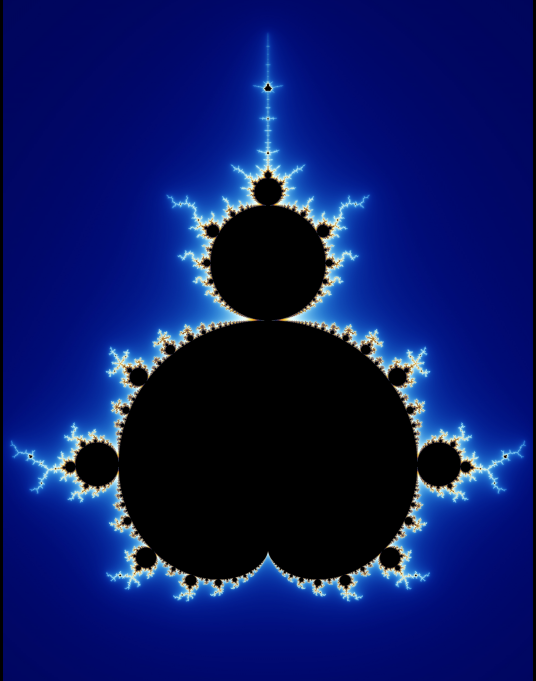
- Searching program space *exhaustively* is impossible (Turing: Halting Problem)

Schmidhuber

- Optimal Ordered Solvers

Lake & Tenenbaum

- Representation Matters



Many Problems Are Program Learning In Disguise

Approach Template Example

Problem

Representation

Optimization

Approach Template Example

Problem — Image Classification

Representation

Optimization

Approach Template Example

Problem	Image Classification
Representation	Network Weights
Optimization	

Approach Template Example

Problem	Image Classification
Representation	Network Weights
Optimization	Gradient Methods

“Meta”

Referring to itself or to the conventions of its genre;
self-referential (Wikipedia Definition)

AutoML-Zero: Evolving Machine Learning Algorithms From Scratch Real et al., 2020

Problem

Representation

Optimization

AutoML-Zero: Evolving Machine Learning Algorithms From Scratch Real et al., 2020

Problem

Meta Image Classifier

Representation

Optimization

AutoML-Zero: Evolving Machine Learning Algorithms From Scratch Real et al., 2020

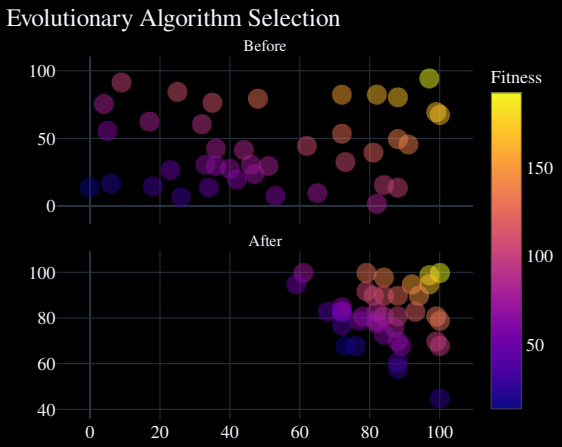
Problem	Meta Image Classifier
Representation	Tensor Machine
Optimization	

AutoML-Zero: Evolving Machine Learning Algorithms From Scratch Real et al., 2020

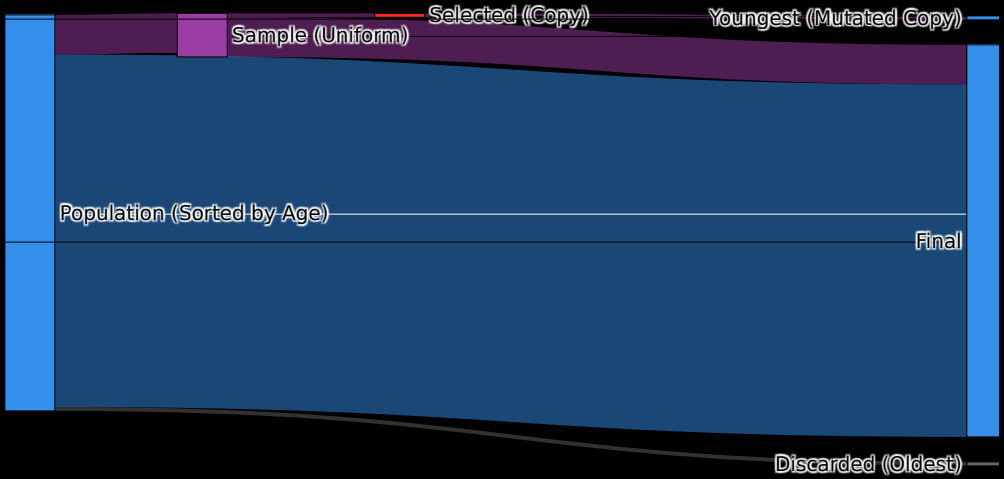
Problem	Meta Image Classifier
Representation	Tensor Machine
Optimization	Reg. Evolution

Population Optimization

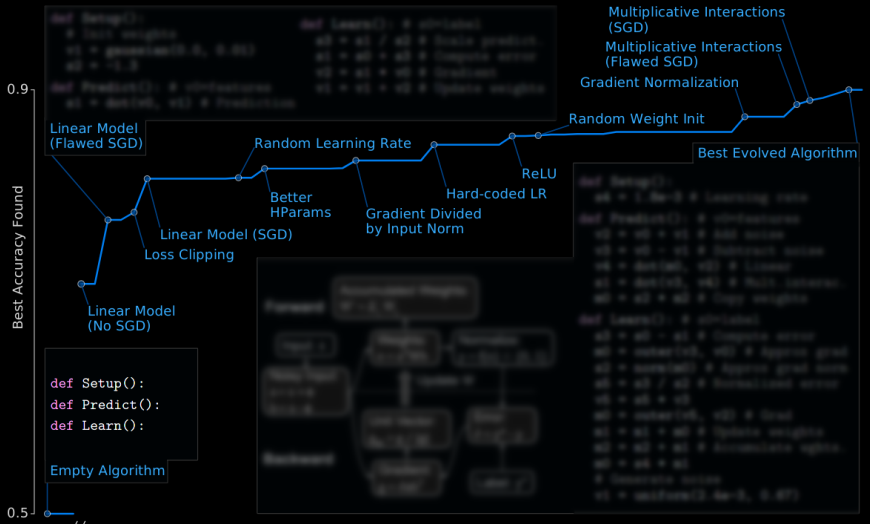
- Selection
- Mutation
- Crossover



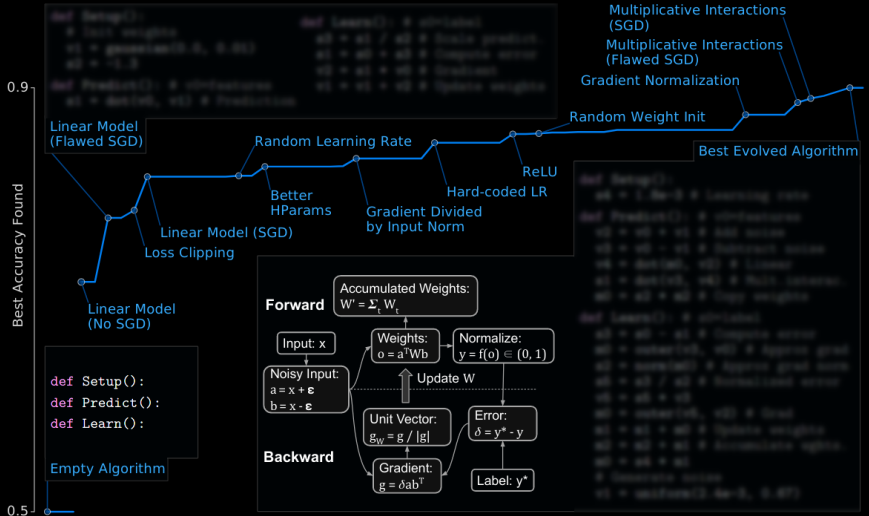
Regularized Evolution Real et al., 2019



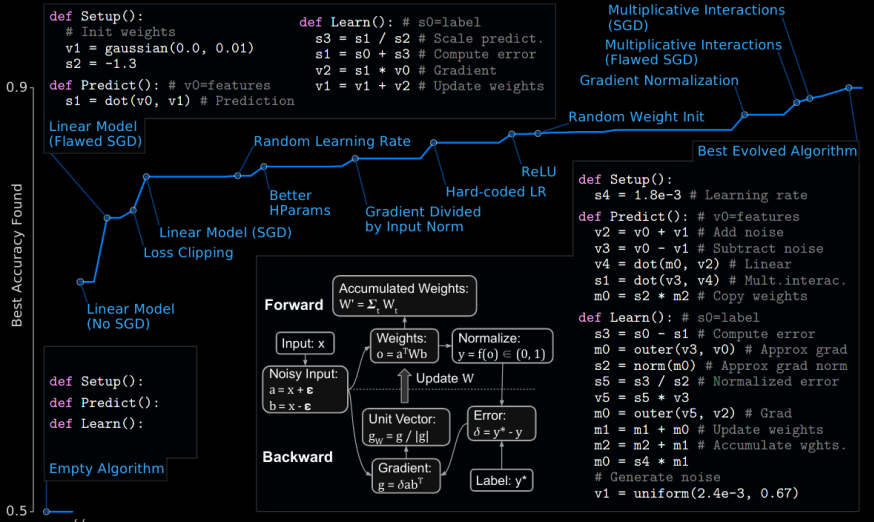
AutoML-Zero Real et al., 2020



AutoML-Zero Real et al., 2020



AutoML-Zero Real et al., 2020



Problem

Representation

Optimization

Problem Representation Optimization

Path Planning

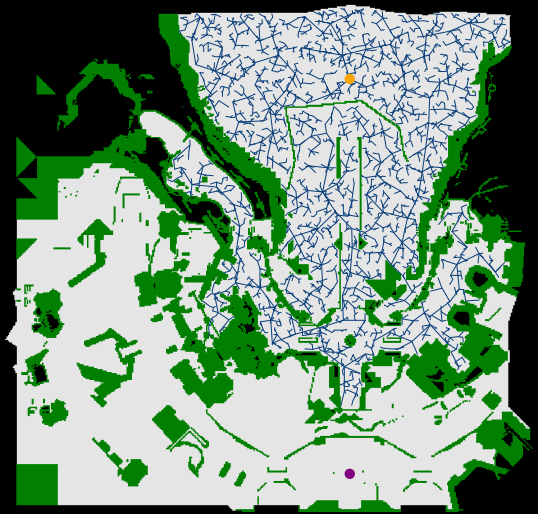
Problem
Representation
Optimization

Path Planning
Python Code

Environment Learning for Path-Planning

Problem	Path Planning
Representation	Python Code
Optimization	Pareto Evolution

Evolvable Python Programs



```python
# Default RRT*
def rrt_star(start, end):
    K = 1000 # Num nodes
    tree = Tree()
    for k in range(K):
        # Uniformly sample
        p = sample(bias, end)
        # Add them to tree
        tree.extend(p, step)
        # Rewire nearby paths
        tree.rewire(p, rad)
    return tree
```


Evolvable Python Programs

```python
def build(K=1000):
  step = 50     # evolved
  rewire = 100 # evolved
  tree = Tree() # const
  for k in range(K):
      p = sample(tree)
      tree.extend(p)
  return tree
```

```python
def sample(tree, goal):
    # all evolved
    a = 1/1 # Code surface!
    if random() > a:
        return goal
    else:
        return (uniform(x),
                uniform(y))
```


Evolvable Python Programs

```python
def build(K=1000):
  step = 50     # evolved
  rewire = 100 # evolved
  tree = Tree() # const
  for k in range(K):
      p = sample(tree)
      tree.extend(p)
  return tree
```

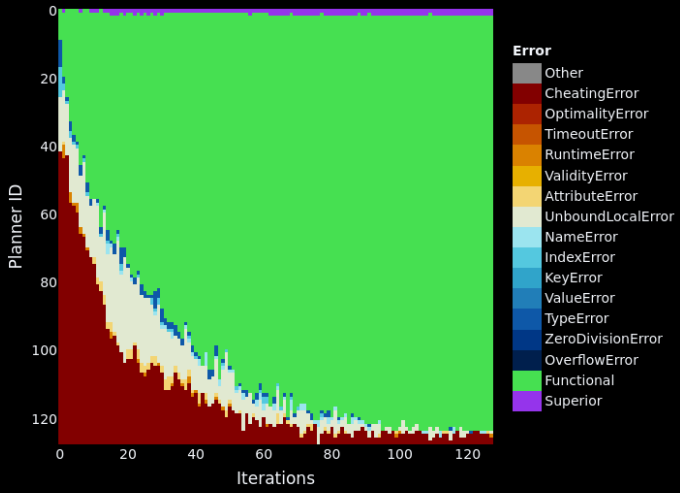
```python
def sample(tree, goal):
    # all evolved
    a = 1/1 # Code surface!
    if random() > a:
        return goal
    else:
        return (uniform(x),
                uniform(y))
```

MD5 Hash	α (bias)	r_0 (step)	r_1 (rewire)	K (nodes)
abcde	1.0	50	100	1000
fghij	0.5	100	200	1000

Evolvable Python Grammar

```
constant   = -10 .. -1 | 0 | 1 .. 10
local      = x      | y      | goal ..
funcname   = sin    | cos    | uniform ..
binop      = Add    | Sub    | Mult | Div | Mod..
compop     = Eq     | NotEq  | Lt   | LtE | Is..
unaryop    = UAdd   | USub   | Not  | Inv
boolop     = And    | Or
expr       = atom   | binary | unary      | compare
stmt       = assign | return | logic_stmt | ast_expr
binary     = [atom binop expr]
unary      = [unaryop expr]
compare    = [atom compops atoms]
if         = [expr body body]
```


Evolvable Python Programs

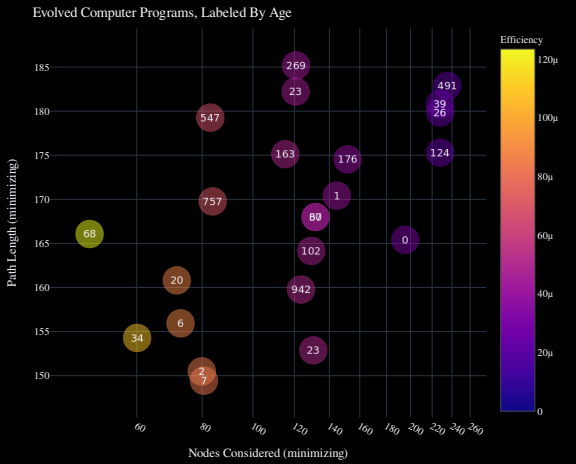


Multiple Objectives

Nodes

Path Length

...



Pareto Fronts

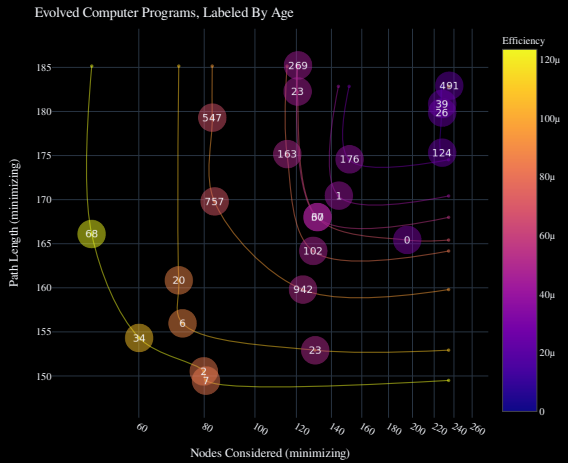
- Vectorized Fitness

$$v_i = \{nodes, path\}$$

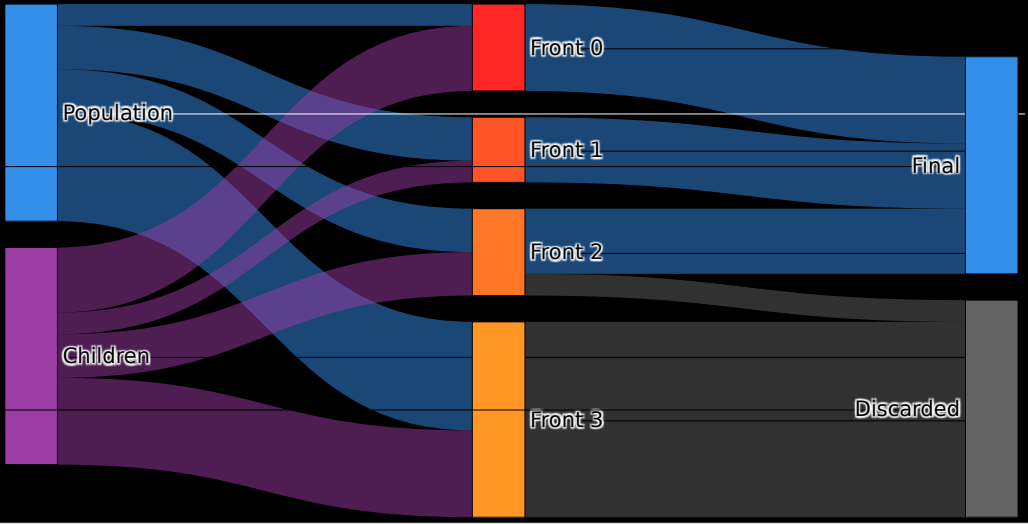
- Pareto Equivalence

$$v_1 = \{80, 151\}$$
$$v_2 = \{60, 155\}$$
$$v_3 = \{50, 165\}$$

$$v_4 = \{130, 153\}$$



Pareto Evolution (Srinivas and Deb, 1994)



Pareto Ev. (NSGA2)

- Multi-objective
- Bandwidth: n

Regularized Ev.

- Single-objective
- Bandwidth: 1

Pareto Ev. (NSGA2)

- Multi-objective
- Bandwidth: n

$$O(g * m * n^2)$$

Regularized Ev.

- Single-objective
- Bandwidth: 1

$$O(g * p)$$

Where g = generations
m = objectives
n = total population size
p = sample size

Pareto Ev. (NSGA2)

- Multi-objective
- Bandwidth: n

$O(g * m * n^2)$
$O(m * n)$ per change

Regularized Ev.

- Single-objective
- Bandwidth: 1

$O(g * p)$
$O(p)$ per change

Where $g =$ generations
$m =$ objectives
$n =$ total population size
$p =$ sample size

Pareto Ev. (NSGA2)

- Multi-objective
- Bandwidth: n

$O(g * m * n^2)$
$O(m * n)$ per change
$O(m * p)$ if sampled

Regularized Ev.

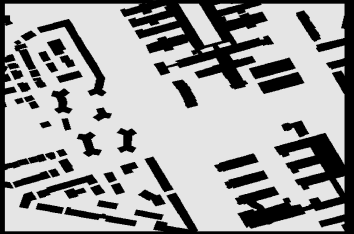
- Single-objective
- Bandwidth: 1

$O(g * p)$
$O(p)$ per change
$O(p)$ (sampled)

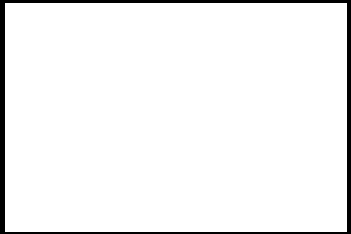
Where $g =$ generations
$m =$ objectives
$n =$ total population size
$p =$ sample size

Specialization Experiment

Streetmaps: Milan_2_512

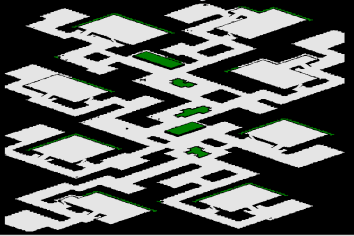


Blank



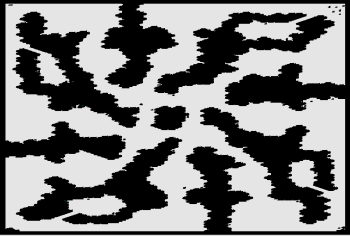
Baldurs Gate: AR0700SR

Starcraft: Enigma



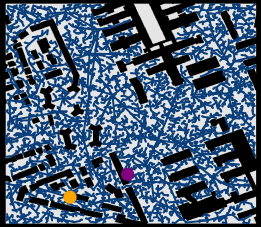
Starcraft: Entanglement

Starcraft: Turbo

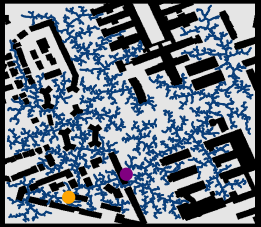


Specialization Results

94e7b (Milan best)



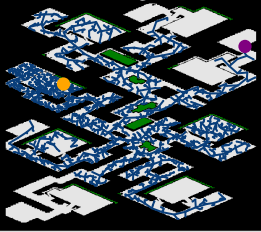
a1be6 (Milan worst)



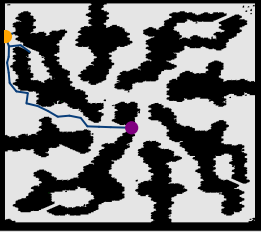
49980 (Validate)

Initial (Validate)

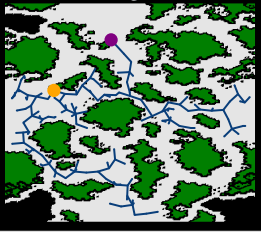
49980 (Enigma best)



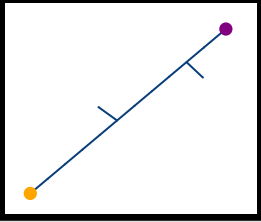
6e799 (Turbo best)



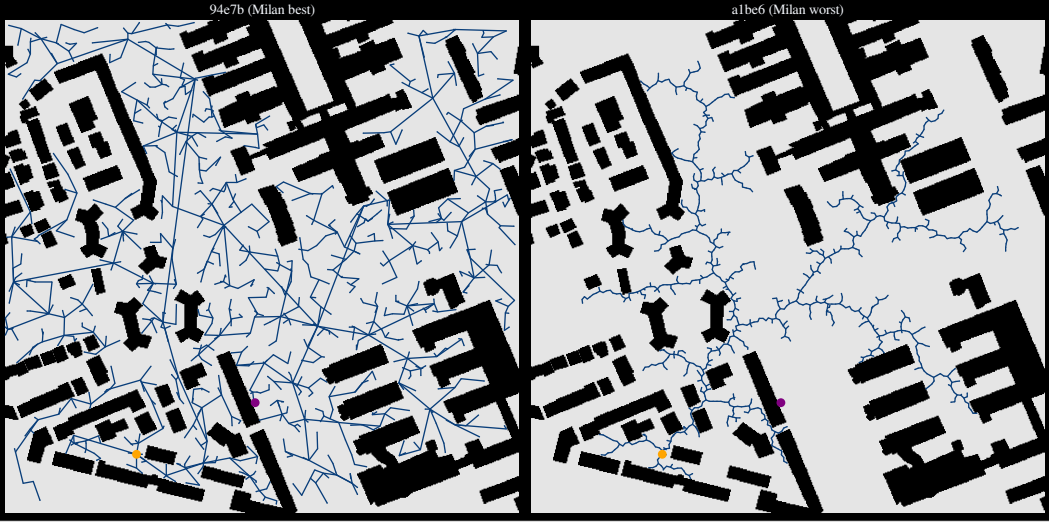
c5b4f (Entanglement best)



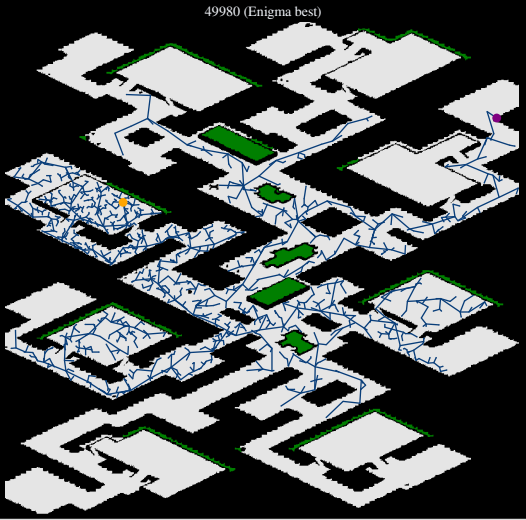
70458 (No Obstacles best)

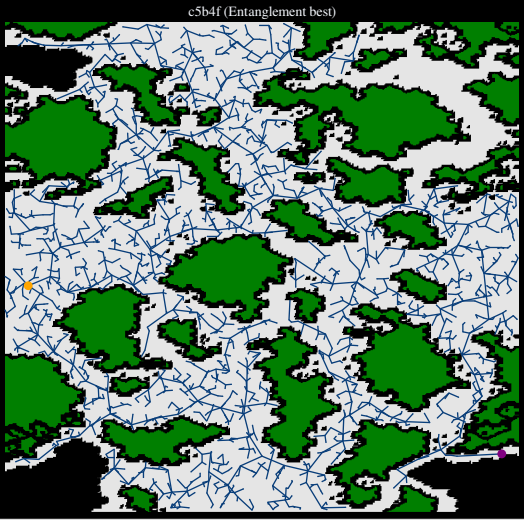


Milan Best/Worst

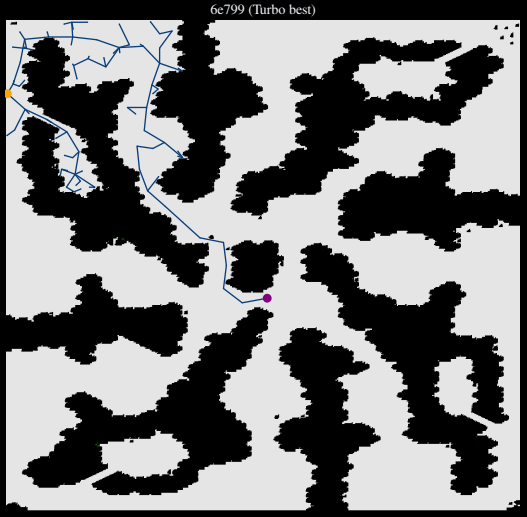


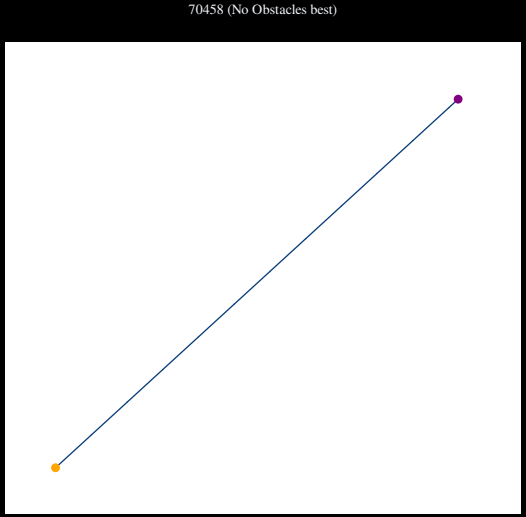
Enigma & Entanglement Best





Turbo and No-Obstacles

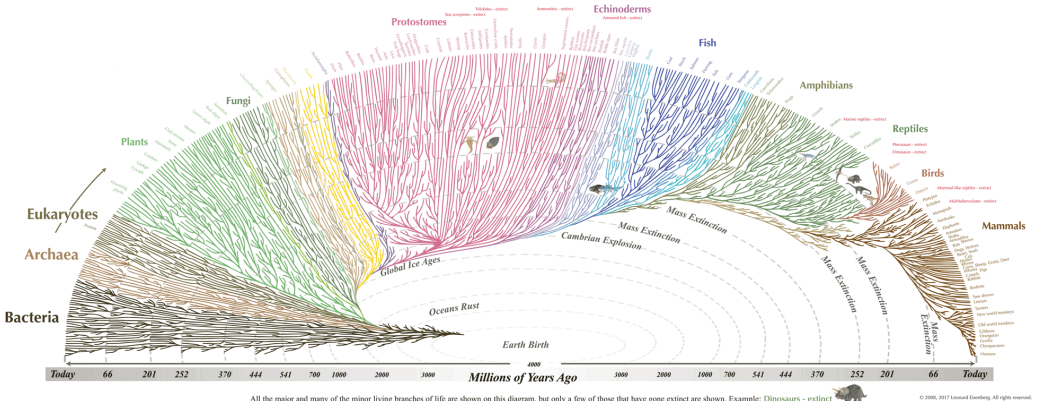




Specialization Results

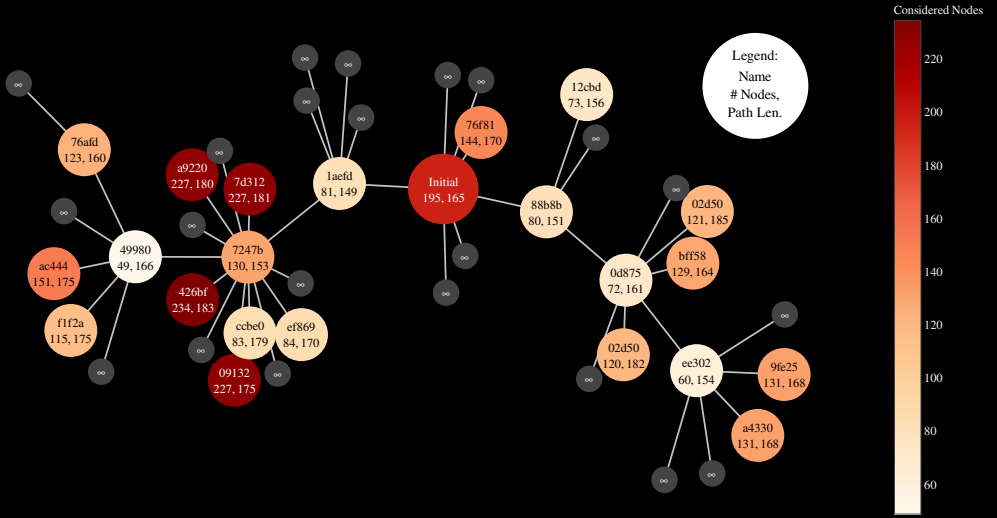
Best-in-class RRT* Statistics (Nodes)

Map	$-\Delta$% Nodes	Edit Depth	Hash	r_0 (step size)	α (bias)
Milan	82.35%	4	*94e7b*	50,000	7/4 (>1)
Enigma	74.87%	3	49980	100	7/8
Entanglement	85.16%	3	*c5b4f*	100	7/8
Turbo	64.29%	3	*6e*799	50	4/9
No Obstacles	89.47%	1	70458	50	1/9
Baseline	0.0%	0	Initial	50	1.0



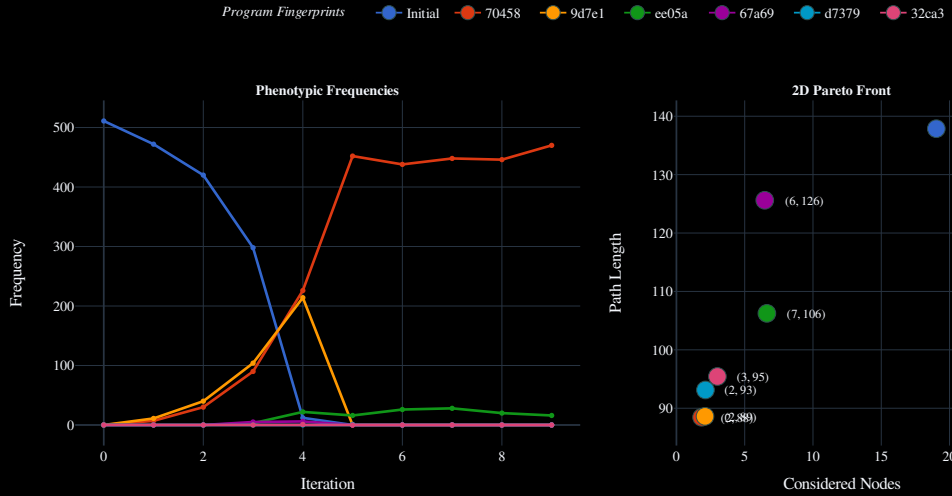
All the major and many of the minor living branches of life are shown on this diagram, but only a few of those that have gone extinct are shown. Example: Dinosaurs - extinct

Natural evolution

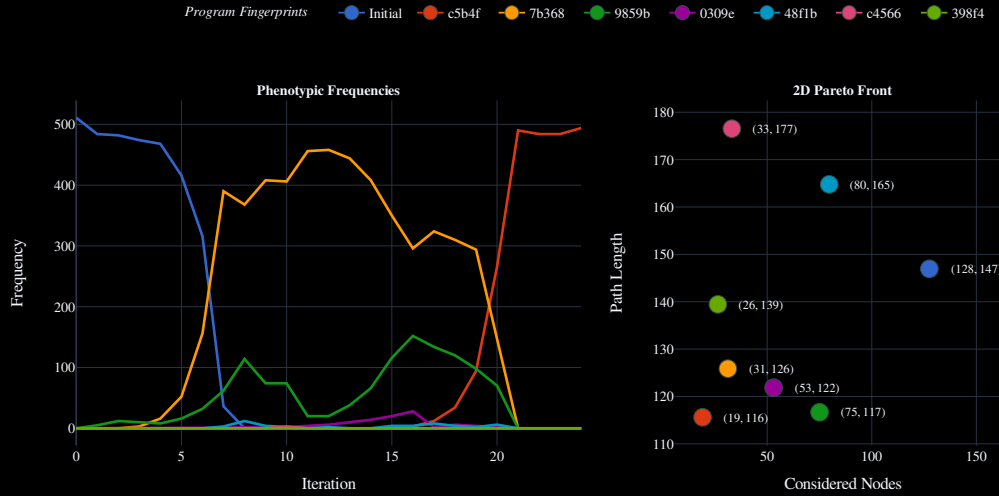


Evolved Computer Programs

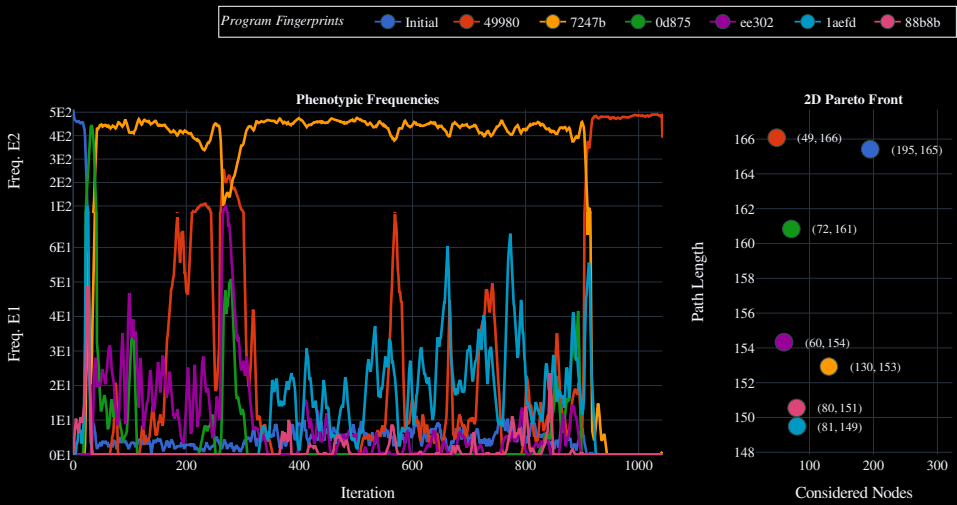
Population Dynamics: Blank Map



Population Dynamics: Entanglement



Population Dynamics: Starcraft Enigma

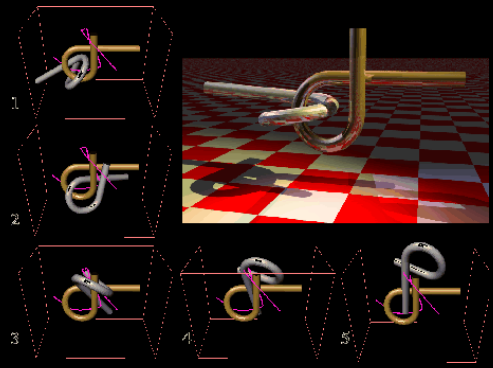


Future Work

Algorithms

Environments

Optimization



LaValle, 2006

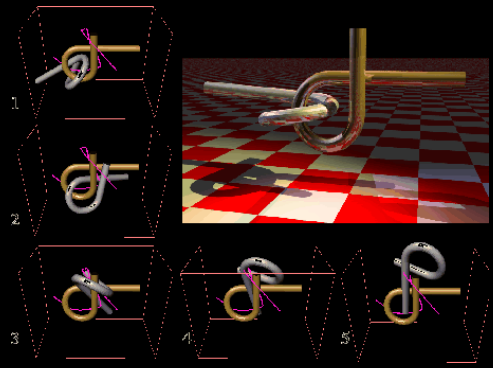
Future Work

Algorithms

- Advanced Route Planning
- Advanced Sampling

Environments

Optimization



LaValle, 2006

Future Work

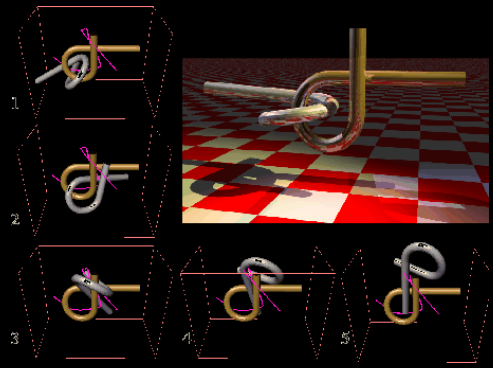
Algorithms

- Advanced Route Planning
- Advanced Sampling

Environments

- High-Dimensions

Optimization



LaValle, 2006

Future Work

Algorithms

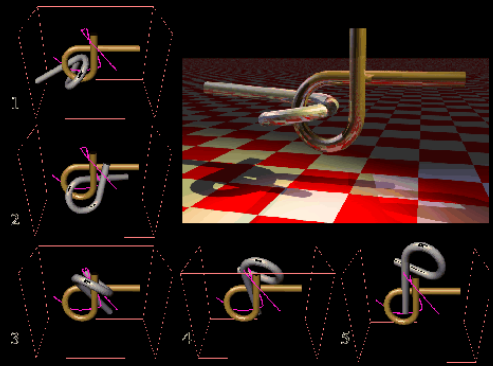
- Advanced Route Planning
- Advanced Sampling

Environments

- High-Dimensions

Optimization

- Alt. Pareto Evolution



LaValle, 2006

Program Learning Template

Program Learning Problems

Problem	Representation	Optimization
Image Classification	NN Weights	Gradient
Char. Recognition	Probabilistic Program	Bayes. Inference
Arch. Search	Tensor Register Machine	Regularized Evolution
Path Planning	Python Program	Pareto Evolution

Wait, it's all program learning?

Always has been.

No Free Lunch

No Free Lunch

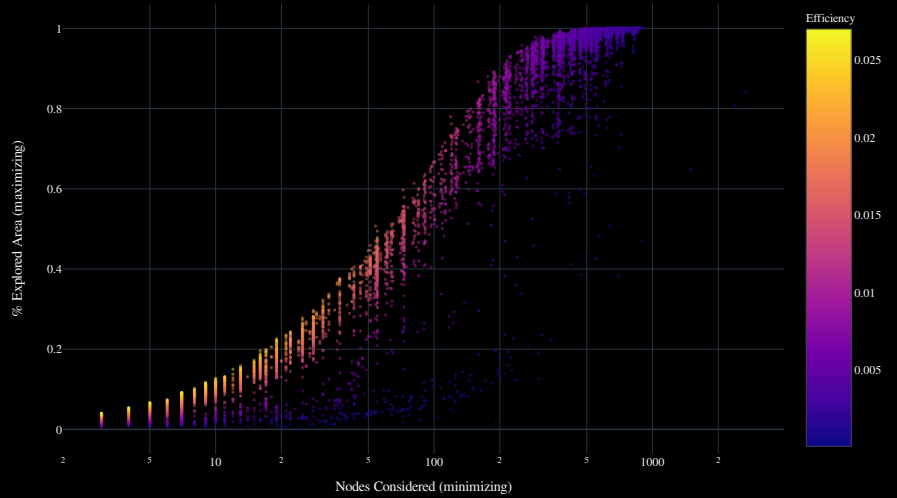
Meta learning and specialization

Questions?

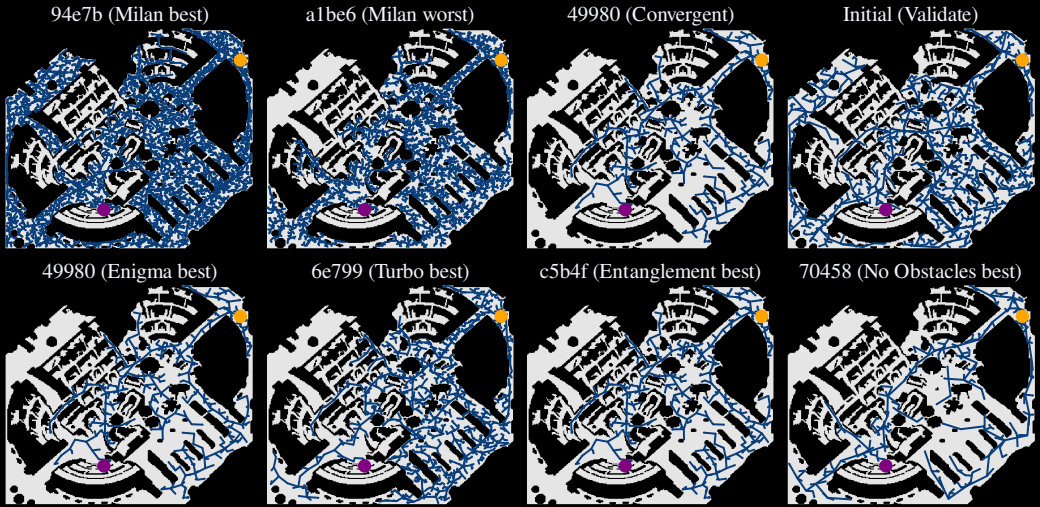
Wait, it's all program learning?

Always has been.

Efficient Exploration

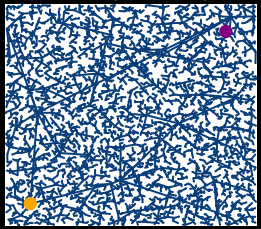


Baldur's Gate Validation

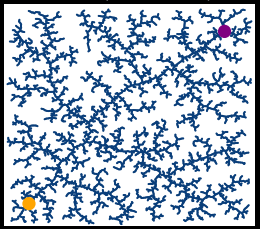


Blank Validation

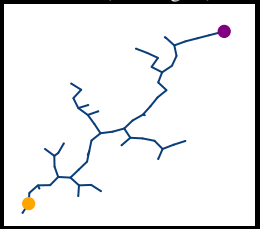
94e7b (Milan best)



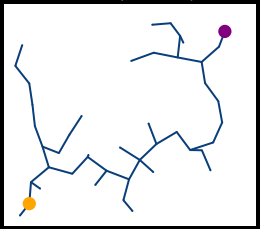
a1be6 (Milan worst)



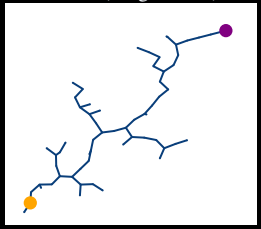
49980 (Convergent)



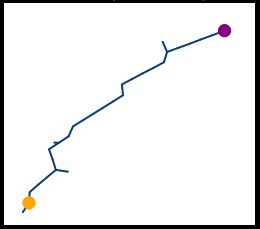
Initial (Validate)



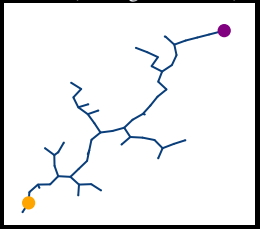
49980 (Enigma best)



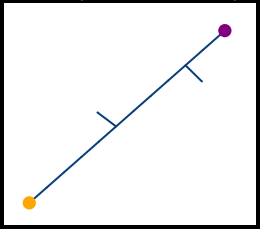
6e799 (Turbo best)



c5b4f (Entanglement best)



70458 (No Obstacles best)



Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., Wagner, D., & Werneck, R. F. (2016). Route planning in transportation networks. *Algorithm engineering* (pp. 19–80). Springer.

Kavraki, L. E., Svestka, P., Latombe, J.-C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE transactions on Robotics and Automation*, *12*(4), 566–580.

LaValle, S. M. (2006). *Planning algorithms*. Cambridge university press.

LaValle, S. M., & Kuffner Jr, J. J. (2001). Randomized kinodynamic planning. *The international journal of robotics research*, *20*(5), 378–400.

Real, E., Aggarwal, A., Huang, Y., & Le, Q. V. (2019). Regularized evolution for image classifier architecture search. *Proceedings of the aaai conference on artificial intelligence*, *33*(01), 4780–4789.

Real, E., Liang, C., So, D., & Le, Q. (2020). Automl-zero: Evolving machine learning algorithms from scratch. *International Conference on Machine Learning*, 8007–8019.

Srinivas, N., & Deb, K. (1994). Muiltiobjective optimization using