CSE 491 Presentation

Namespace: Cowboys

Our Contributions

- GP agent Loop
- LGPAgent
- CGPAgent
- Scavenger Queuing for training
- GPAnalyzer
- Profilers
- Unit Tests
- … and more

What the Heck is GP?

Here are some agents

Agent Aman

Agent Jason

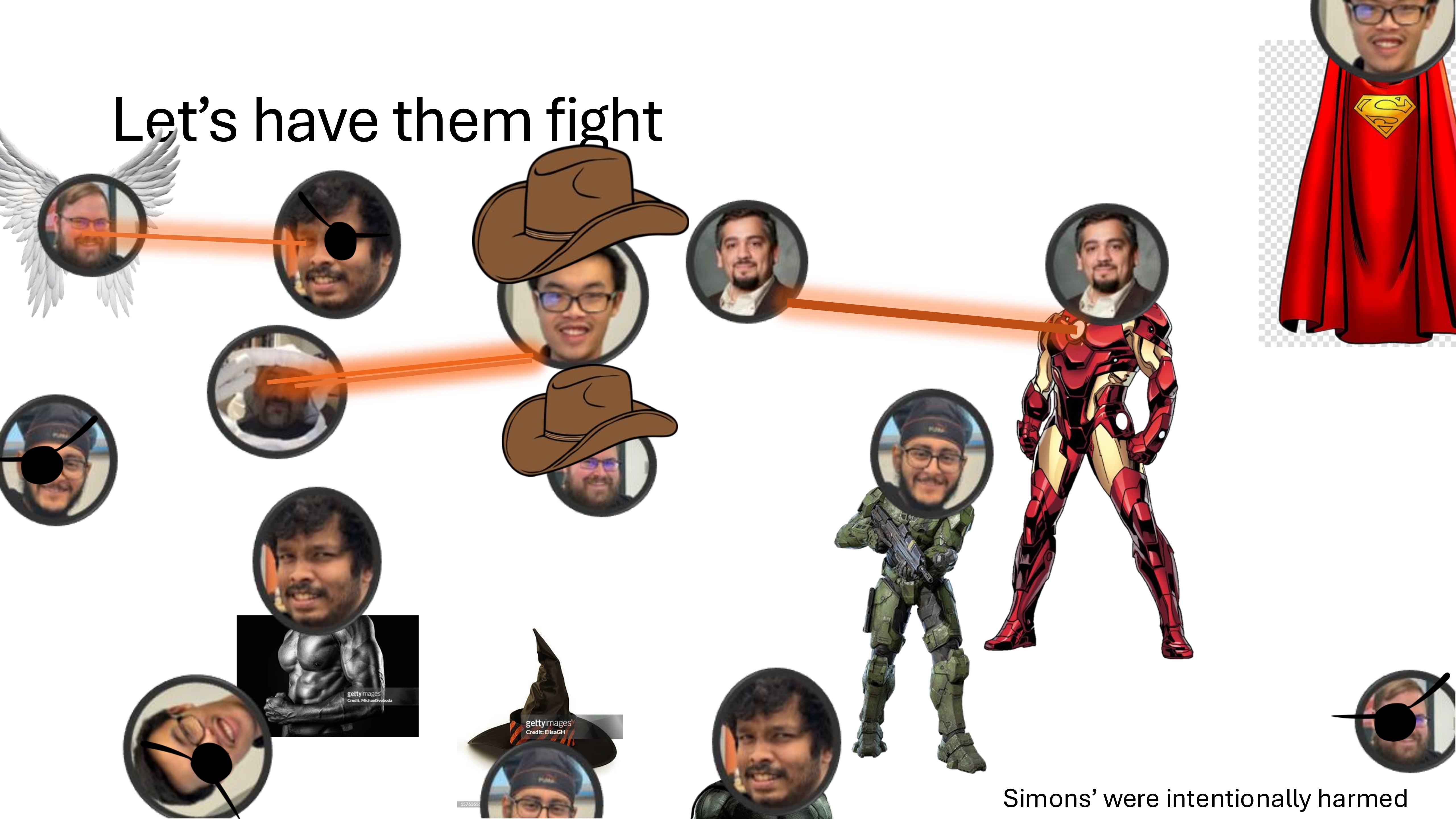
Agent Simon

Agent Rajmeet

Agent 007

Now let's create a lot of these variations

Let’s have them fight



Simons’ were intentionally harmed

Score them

Using a fitness function

Health: 0
Damage done: 0

Score = -10

Health: 0
Damage done: 0

Score = -10

Health: 80
Damage done: 60

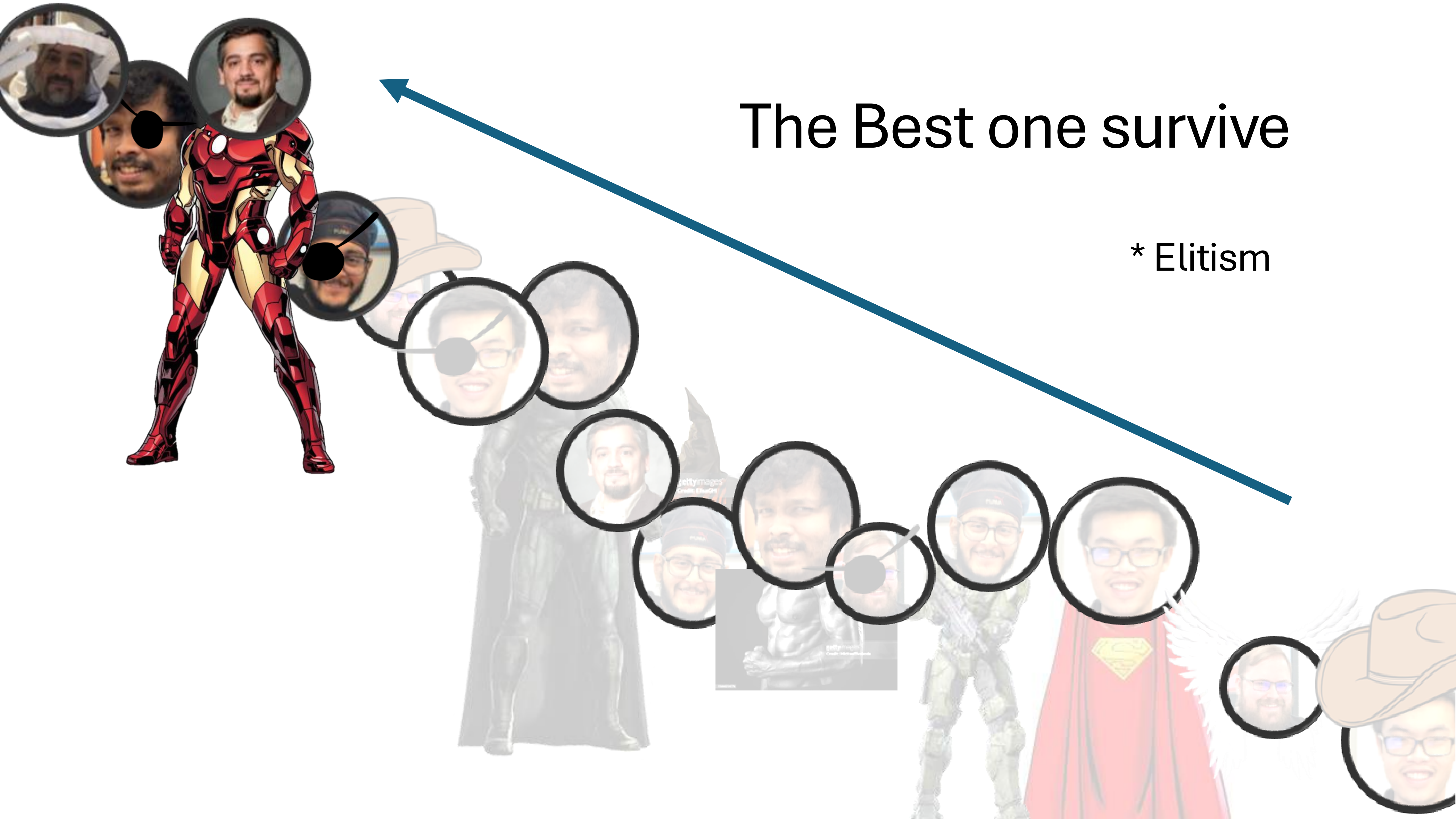
Score = 120

.
.
.

Let's Sort them

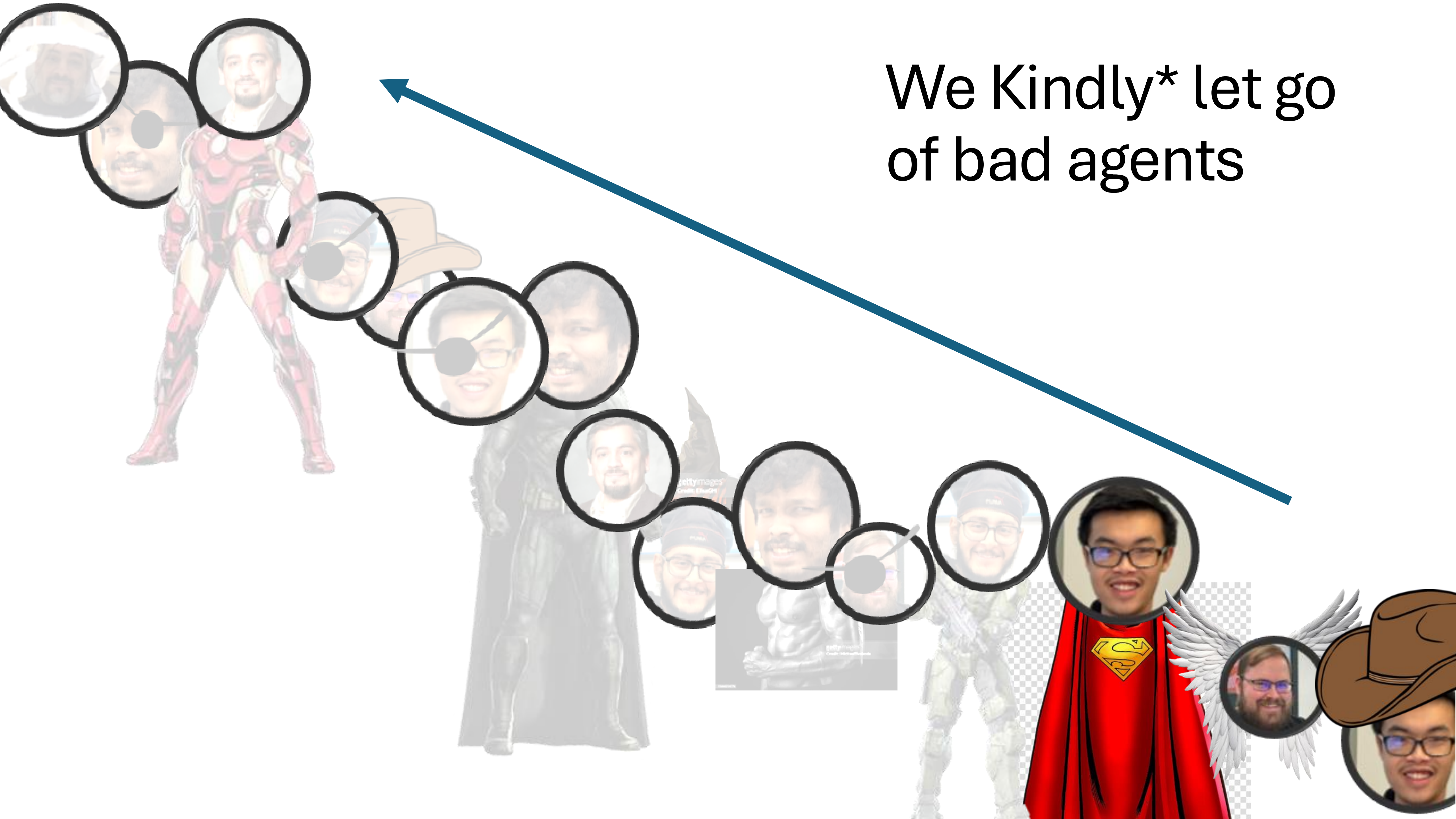
The Best one survive

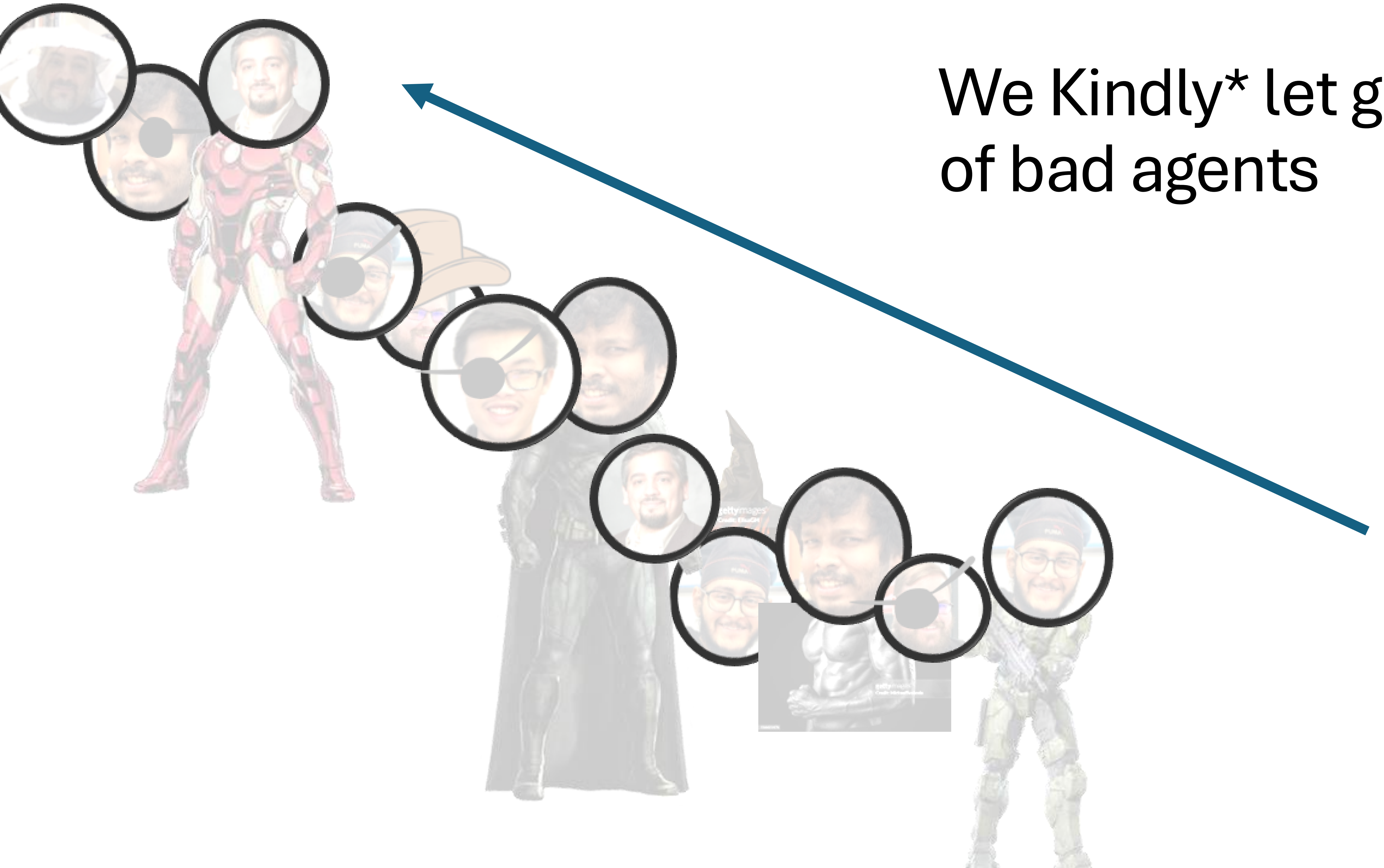
* Elitism



The better ones get mutated

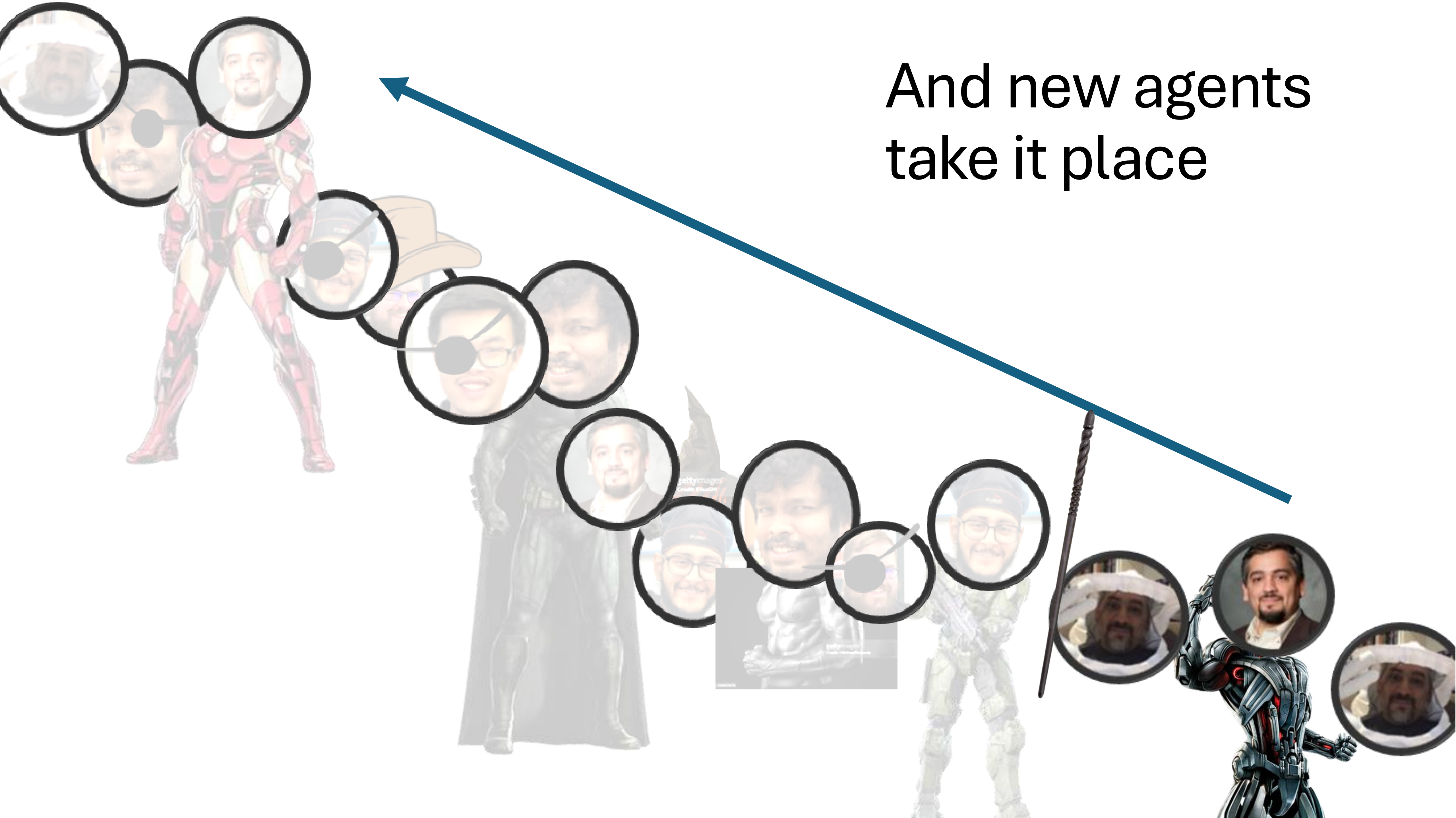
We Kindly* let go of bad agents





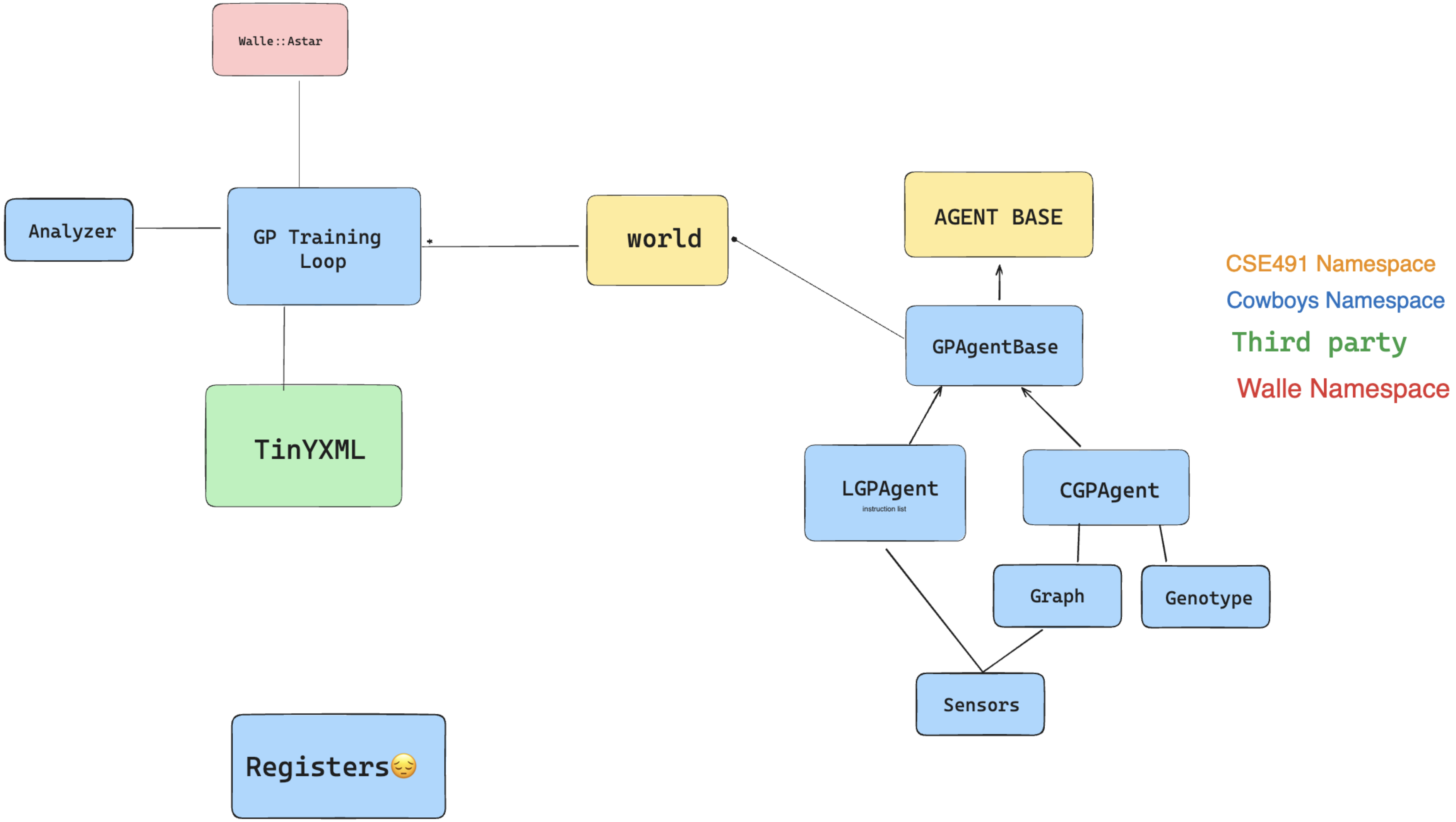
We Kindly* let go of bad agents

And new agents take it place



And then we repeat the same for the next generation

Our Design



CSE491 Namespace

Cowboys Namespace

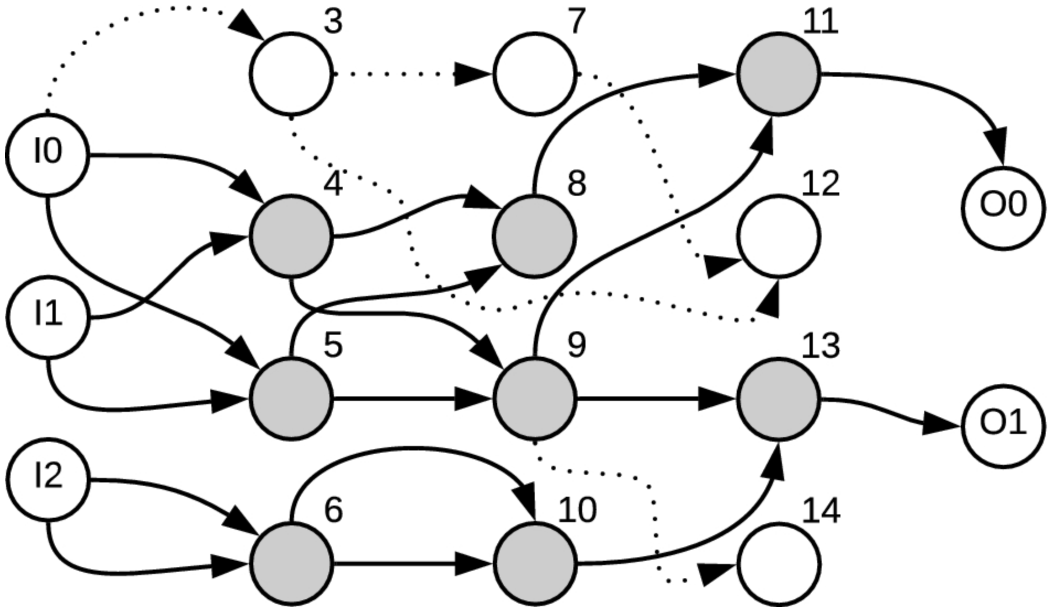
Third party

Walle Namespace

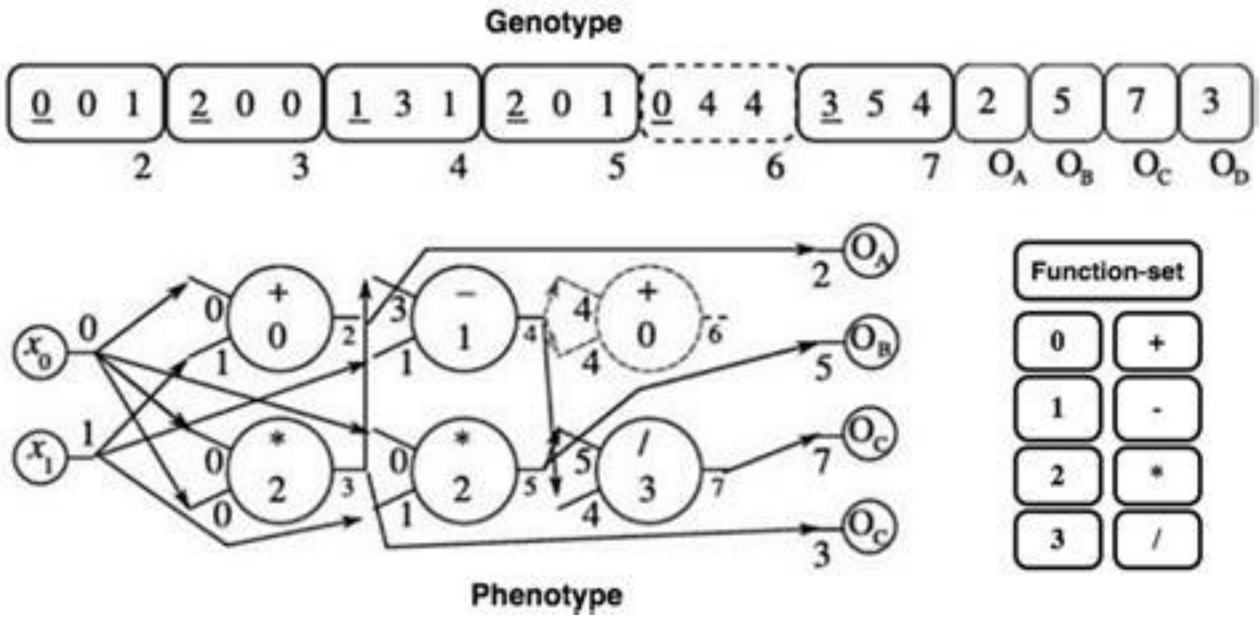
CGPAgent

(Cartesian Genetic Programming)

The idea

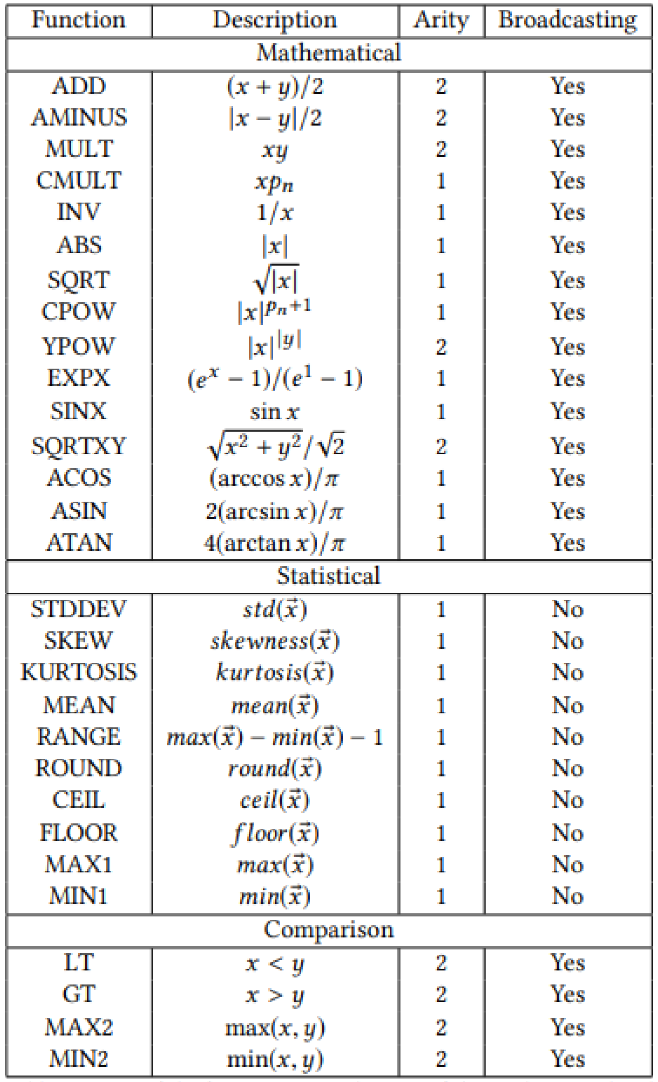


Credit: [Mitchell Spryn]



Credit: [Miragaia et al.]

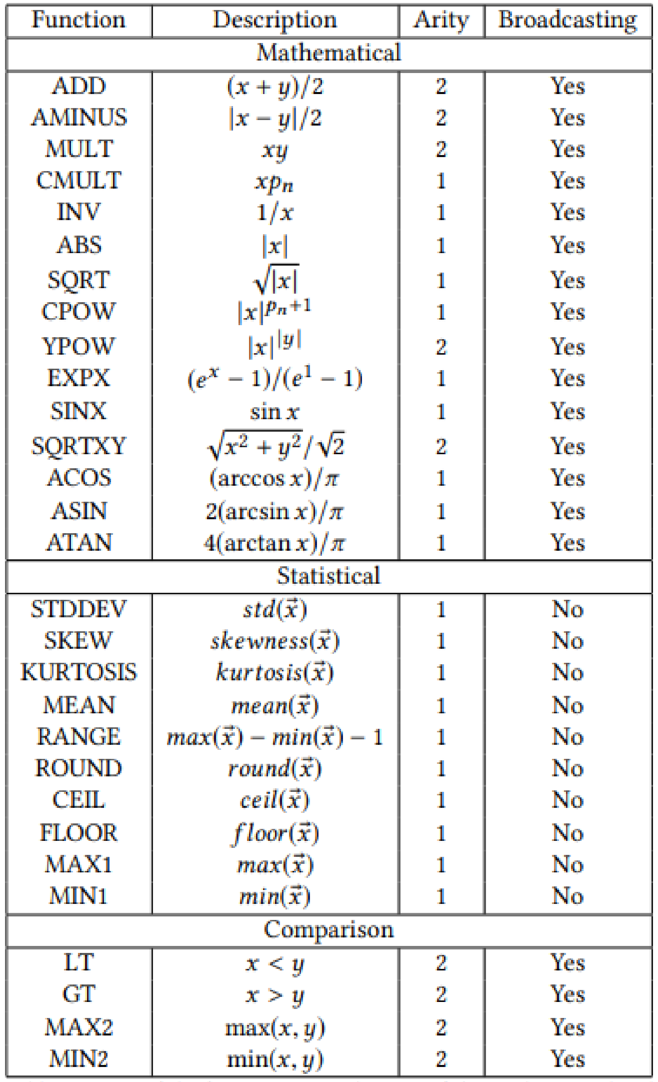
Function Set



Function	Description	Arity	Broadcasting
Mathematical			
ADD	$(x+y)/2$	2	Yes
AMINUS	$\lvert x-y\rvert/2$	2	Yes
MULT	xy	2	Yes
CMULT	xp_n	1	Yes
INV	$1/x$	1	Yes
ABS	$\lvert x\rvert$	1	Yes
SQRT	$\sqrt{\lvert x\rvert}$	1	Yes
CPOW	$\lvert x\rvert^{p_n+1}$	1	Yes
YPOW	$\lvert x\rvert^{\lvert y\rvert}$	2	Yes
EXPX	$(e^x-1)/(e^1-1)$	1	Yes
SINX	$\sin x$	1	Yes
SQRTXY	$\sqrt{x^2+y^2}/\sqrt{2}$	2	Yes
ACOS	$(\arccos x)/\pi$	1	Yes
ASIN	$2(\arcsin x)/\pi$	1	Yes
ATAN	$4(\arctan x)/\pi$	1	Yes
Statistical			
STDDEV	$std(\vec{x})$	1	No
SKEW	$skewness(\vec{x})$	1	No
KURTOSIS	$kurtosis(\vec{x})$	1	No
MEAN	$mean(\vec{x})$	1	No
RANGE	$max(\vec{x})-min(\vec{x})-1$	1	No
ROUND	$round(\vec{x})$	1	No
CEIL	$ceil(\vec{x})$	1	No
FLOOR	$floor(\vec{x})$	1	No
MAX1	$max(\vec{x})$	1	No
MIN1	$min(\vec{x})$	1	No
Comparison			
LT	$x<y$	2	Yes
GT	$x>y$	2	Yes
MAX2	$\max(x,y)$	2	Yes
MIN2	$\min(x,y)$	2	Yes

Credit: *Evolving simple programs for playing Atari games* by Wilson et al.

Function Set



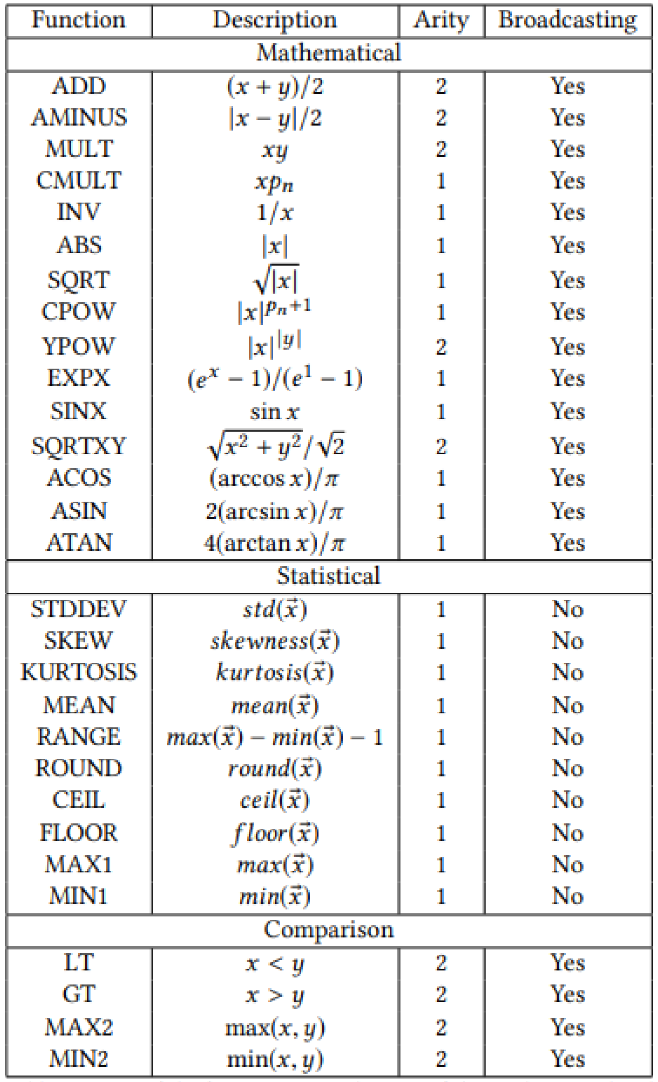
Function	Description	Arity	Broadcasting
Mathematical			
ADD	$(x+y)/2$	2	Yes
AMINUS	$\|x-y\|/2$	2	Yes
MULT	xy	2	Yes
CMULT	xp_n	1	Yes
INV	$1/x$	1	Yes
ABS	$\|x\|$	1	Yes
SQRT	$\sqrt{\|x\|}$	1	Yes
CPOW	$\|x\|^{p_n+1}$	1	Yes
YPOW	$\|x\|^{\|y\|}$	2	Yes
EXPX	$(e^x-1)/(e^1-1)$	1	Yes
SINX	$\sin x$	1	Yes
SQRTXY	$\sqrt{x^2+y^2}/\sqrt{2}$	2	Yes
ACOS	$(\arccos x)/\pi$	1	Yes
ASIN	$2(\arcsin x)/\pi$	1	Yes
ATAN	$4(\arctan x)/\pi$	1	Yes
Statistical			
STDDEV	$std(\vec{x})$	1	No
SKEW	$skewness(\vec{x})$	1	No
KURTOSIS	$kurtosis(\vec{x})$	1	No
MEAN	$mean(\vec{x})$	1	No
RANGE	$max(\vec{x})-min(\vec{x})-1$	1	No
ROUND	$round(\vec{x})$	1	No
CEIL	$ceil(\vec{x})$	1	No
FLOOR	$floor(\vec{x})$	1	No
MAX1	$max(\vec{x})$	1	No
MIN1	$min(\vec{x})$	1	No
Comparison			
LT	$x<y$	2	Yes
GT	$x>y$	2	Yes
MAX2	$\max(x,y)$	2	Yes
MIN2	$\min(x,y)$	2	Yes

What we used:

- Sum
- Product
- Sin
- Cos
- Exp
- Max
- Min
- And many more

Credit: *Evolving simple programs for playing Atari games* by Wilson et al.

Function Set



Function	Description	Arity	Broadcasting
Mathematical			
ADD	$(x+y)/2$	2	Yes
AMINUS	$\|x-y\|/2$	2	Yes
MULT	xy	2	Yes
CMULT	xp_n	1	Yes
INV	$1/x$	1	Yes
ABS	$\|x\|$	1	Yes
SQRT	$\sqrt{\|x\|}$	1	Yes
CPOW	$\|x\|^{p_n+1}$	1	Yes
YPOW	$\|x\|^{\|y\|}$	2	Yes
EXPX	$(e^x-1)/(e^1-1)$	1	Yes
SINX	$\sin x$	1	Yes
SQRTXY	$\sqrt{x^2+y^2}/\sqrt{2}$	2	Yes
ACOS	$(\arccos x)/\pi$	1	Yes
ASIN	$2(\arcsin x)/\pi$	1	Yes
ATAN	$4(\arctan x)/\pi$	1	Yes
Statistical			
STDDEV	$std(\vec{x})$	1	No
SKEW	$skewness(\vec{x})$	1	No
KURTOSIS	$kurtosis(\vec{x})$	1	No
MEAN	$mean(\vec{x})$	1	No
RANGE	$max(\vec{x})-min(\vec{x})-1$	1	No
ROUND	$round(\vec{x})$	1	No
CEIL	$ceil(\vec{x})$	1	No
FLOOR	$floor(\vec{x})$	1	No
MAX1	$max(\vec{x})$	1	No
MIN1	$min(\vec{x})$	1	No
Comparison			
LT	$x<y$	2	Yes
GT	$x>y$	2	Yes
MAX2	$\max(x,y)$	2	Yes
MIN2	$\min(x,y)$	2	Yes

Credit: *Evolving simple programs for playing Atari games by Wilson et al.*

What we used:

- Sum
- Product
- Sin
- Cos
- Exp
- Max
- Min
- And many more

And sensors:

- Distance to nearest walls in each cardinal direction
- A* Distance

LGP Agent

(Linear Genetic Programming)

How it works

Three primary data structures:

1. Instruction list
2. Results list
3. Index of current instruction

How it works

Three primary data structures:

1. Instruction list
2. Results list
3. Index of current instruction

Three types of instructions:

1. Sensors
2. Operations
3. Actions

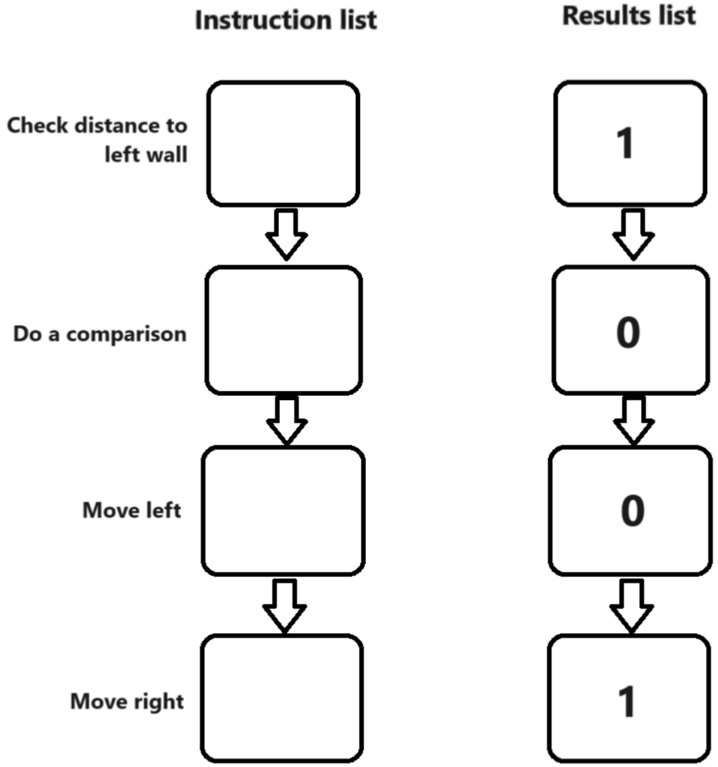
How it works

Three primary data structures:

1. Instruction list
2. Results list
3. Index of current instruction

Three types of instructions:

1. Sensors
2. Operations
3. Actions



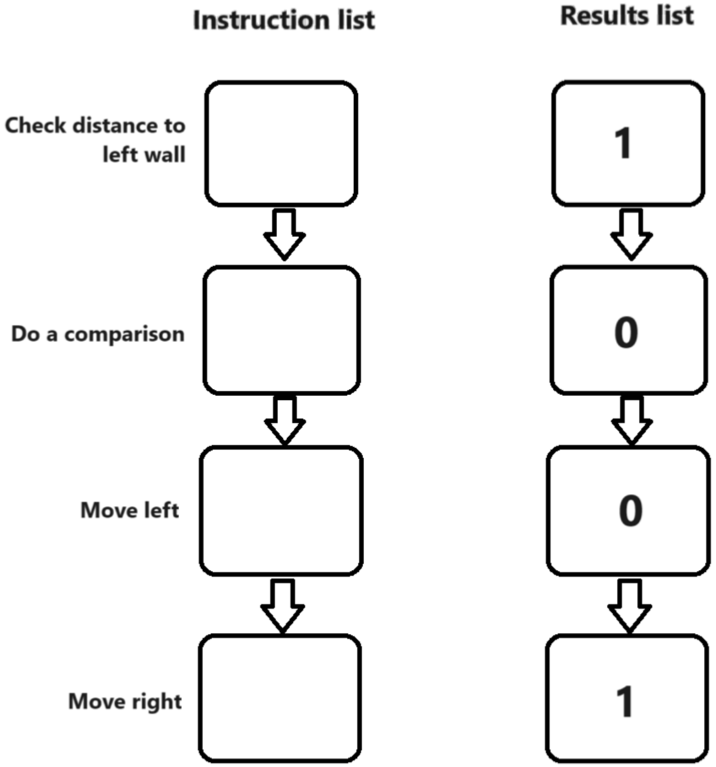
How it works

Three primary data structures:

1. Instruction list
2. Results list
3. Index of current instruction

Three types of instructions:

1. Sensors
2. Operations
3. Actions



Mutating is very easy!
Runs very quickly!

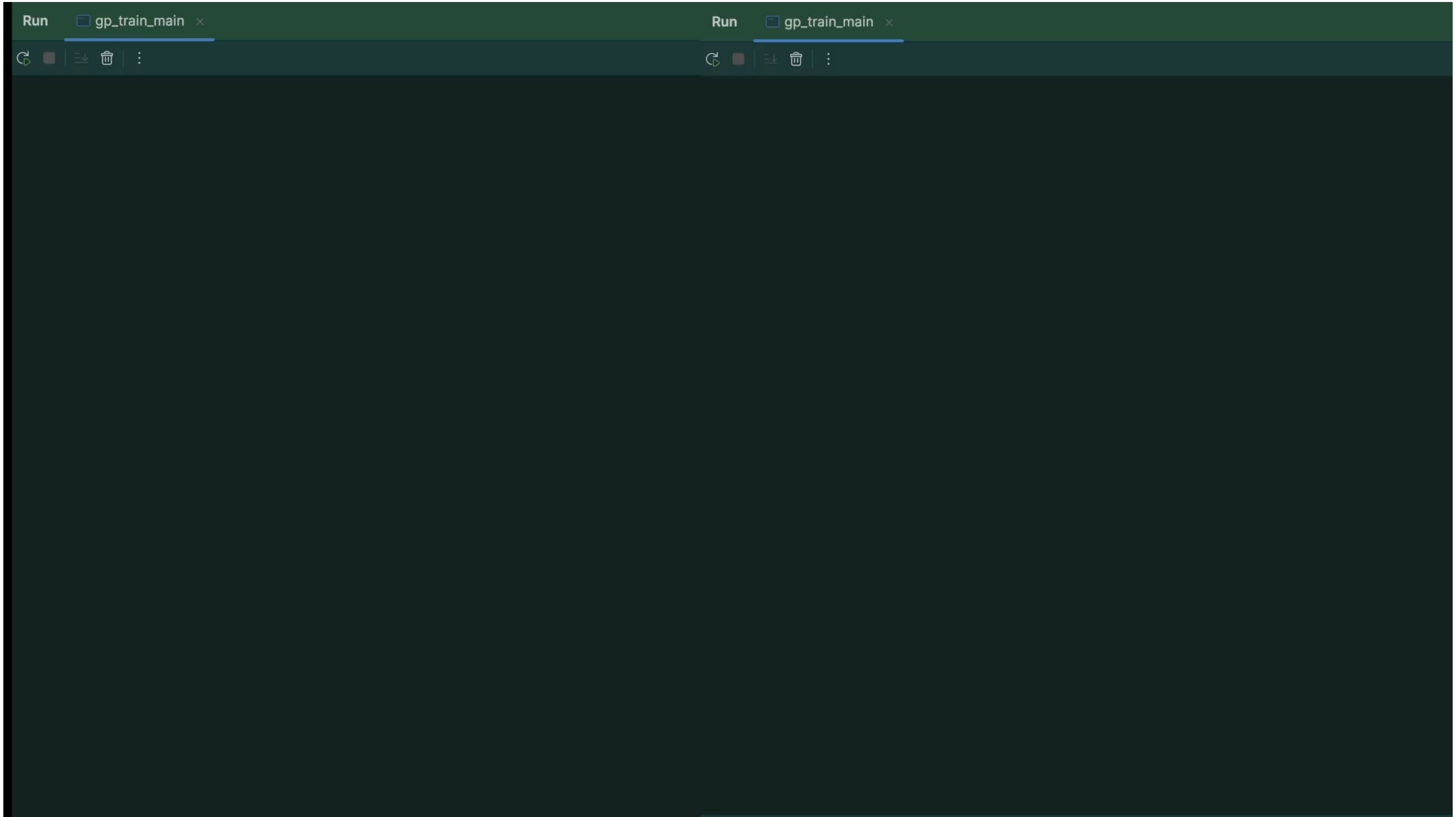
Downside?

Genetic Programing Loop

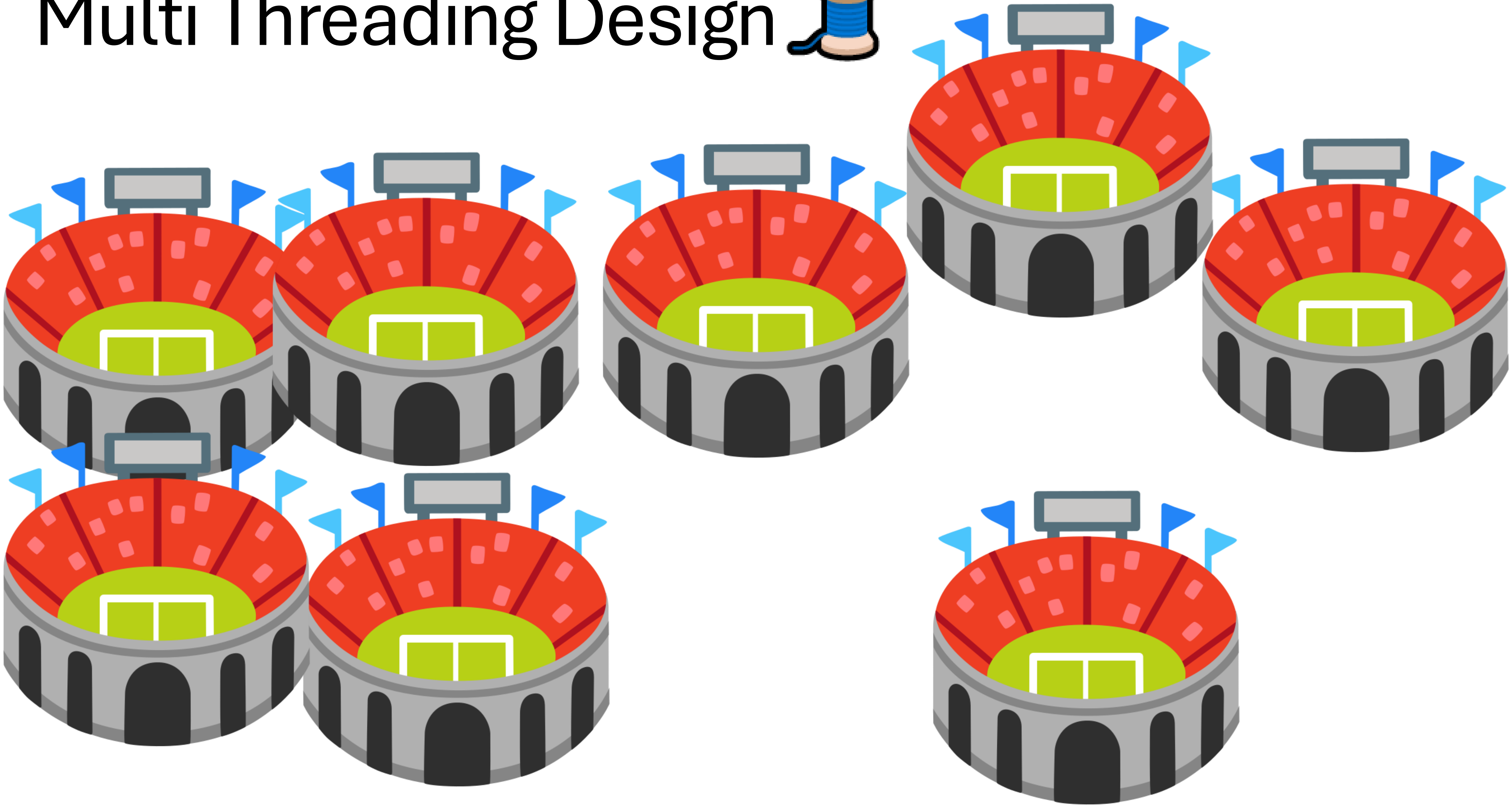
Tasks that GP Loop is responsible for

- Instantiation
- Scoring agents
- Mutating Agents
- Running Worlds

The power of multi threading



Multi Threading Design 🧵



Multi Threading Design 🧵

Thread 1

Thread 2

Thread 3

Scavenger Queuing / Checkpointing

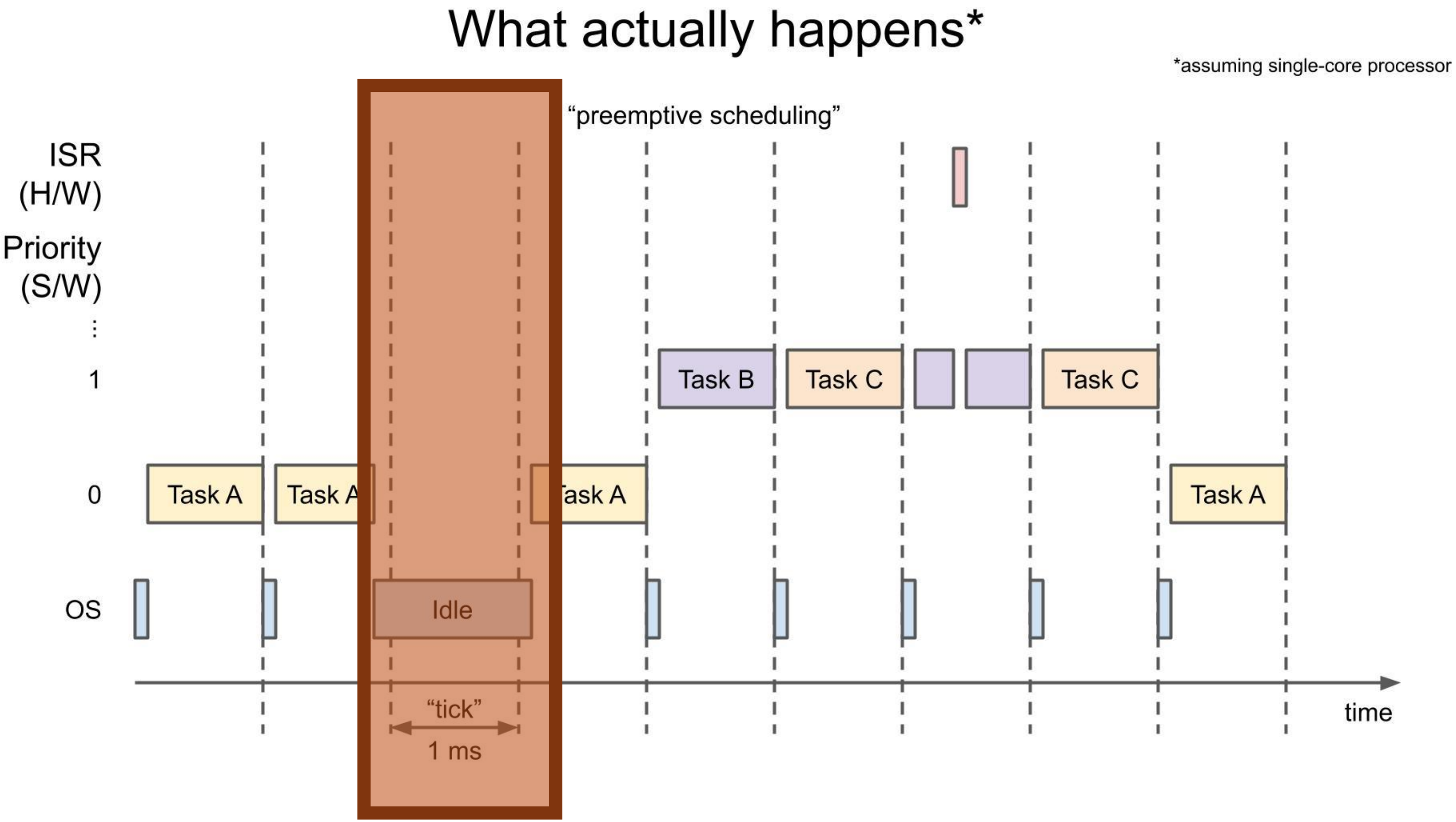


Image from https://www.digikey.com/en/maker/projects/introduction-to-rtos-solution-to-part-3-task-scheduling/8fbb9e0b0eed4279a2dd698f02ce125f

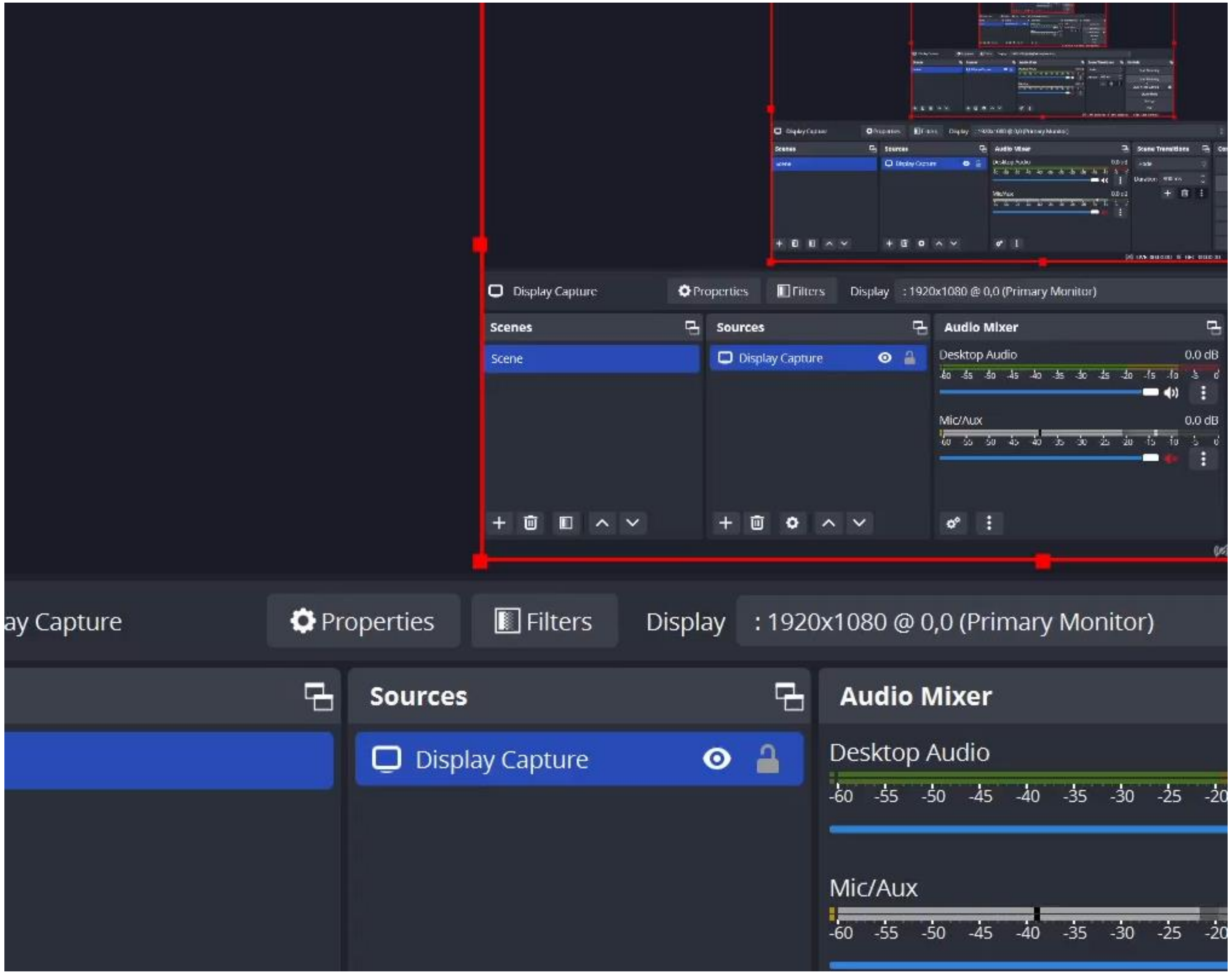
Demo

Is the program really evolving?

Yeah, trust us bro!

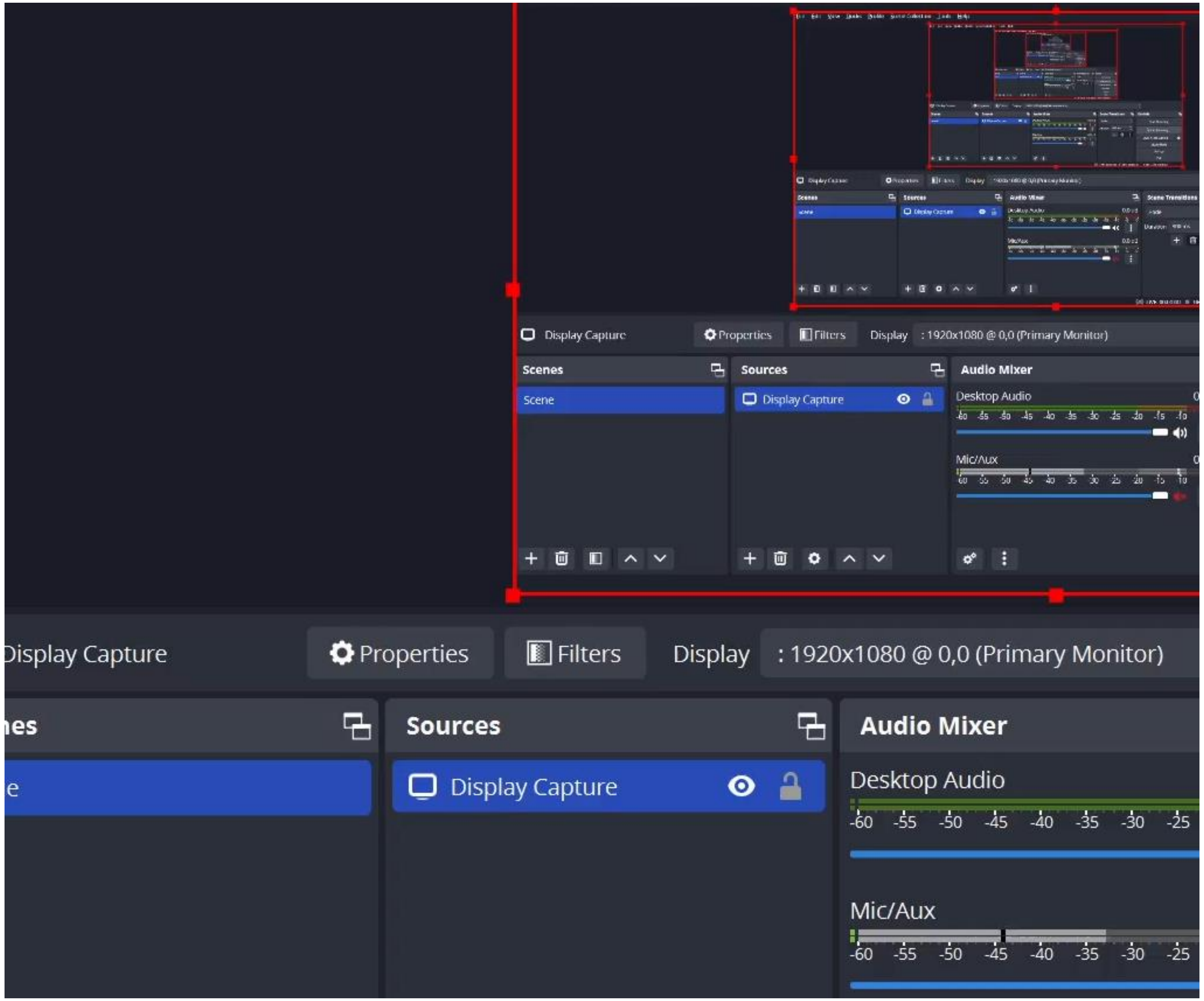
16 arenas
1000 agents
100 generations

Metrics: Average Fitness

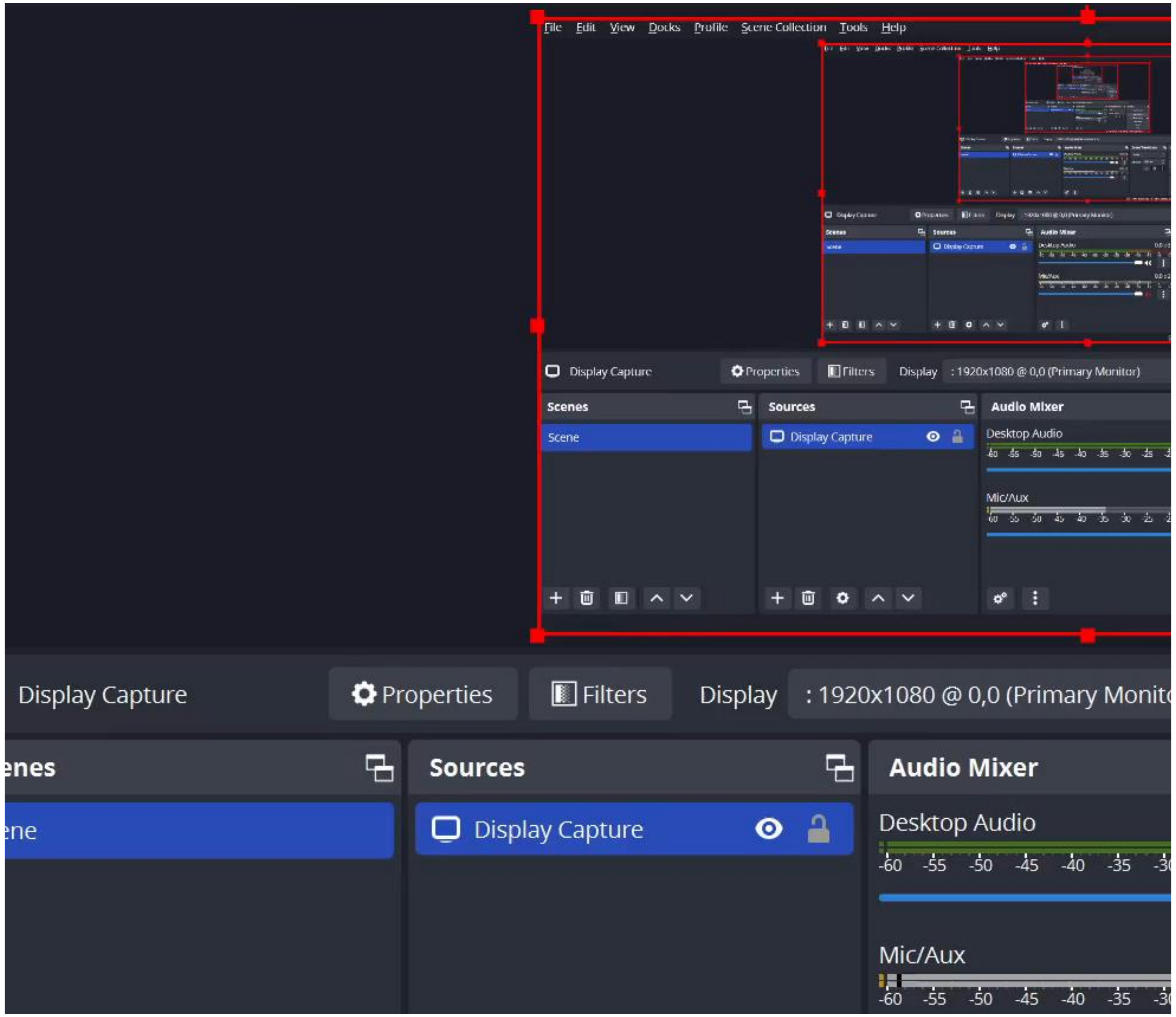


Display : 1920x1080 @ 0,0 (Primary Monitor)

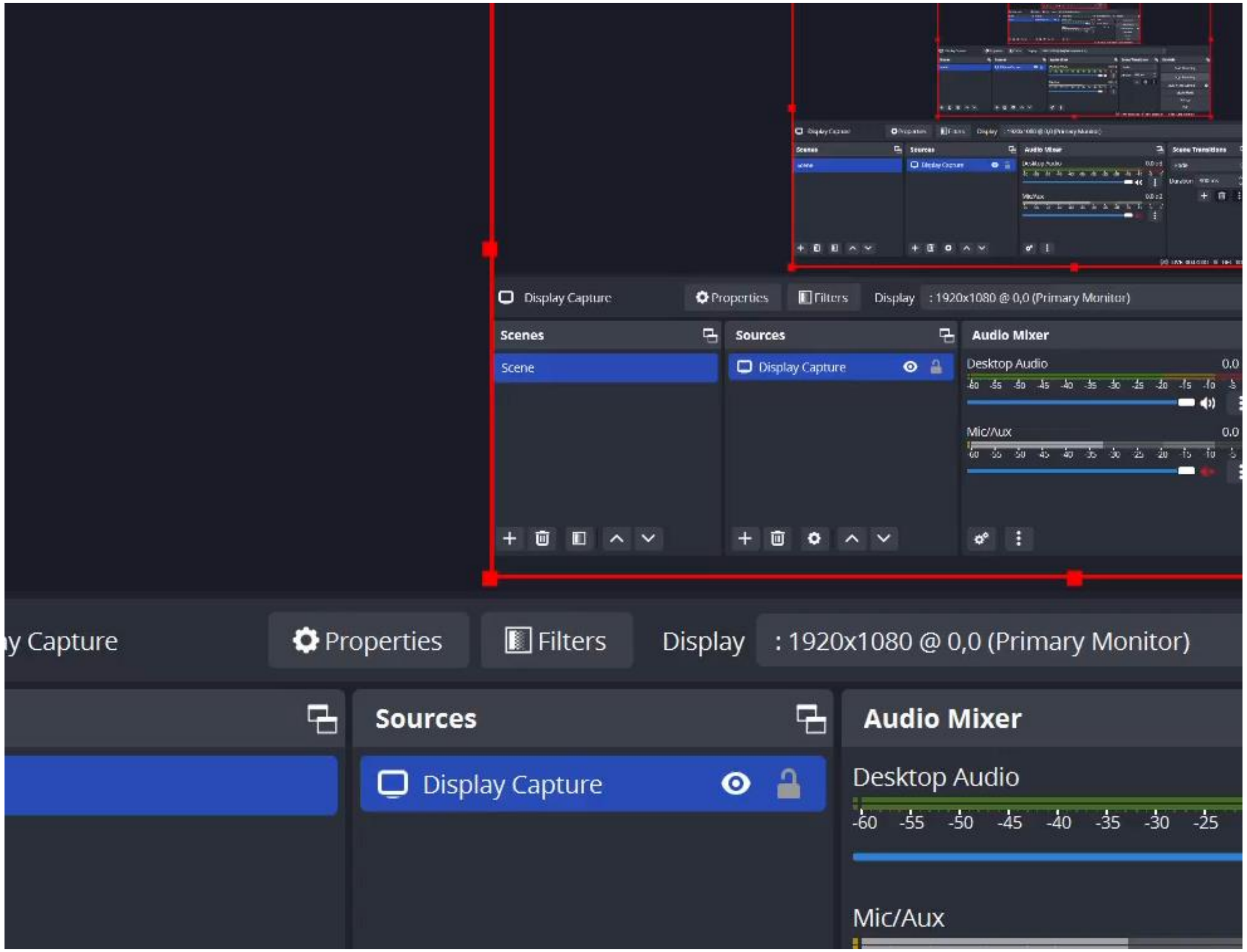
Metrics: Max Fitness



Metrics: Average Elite Score

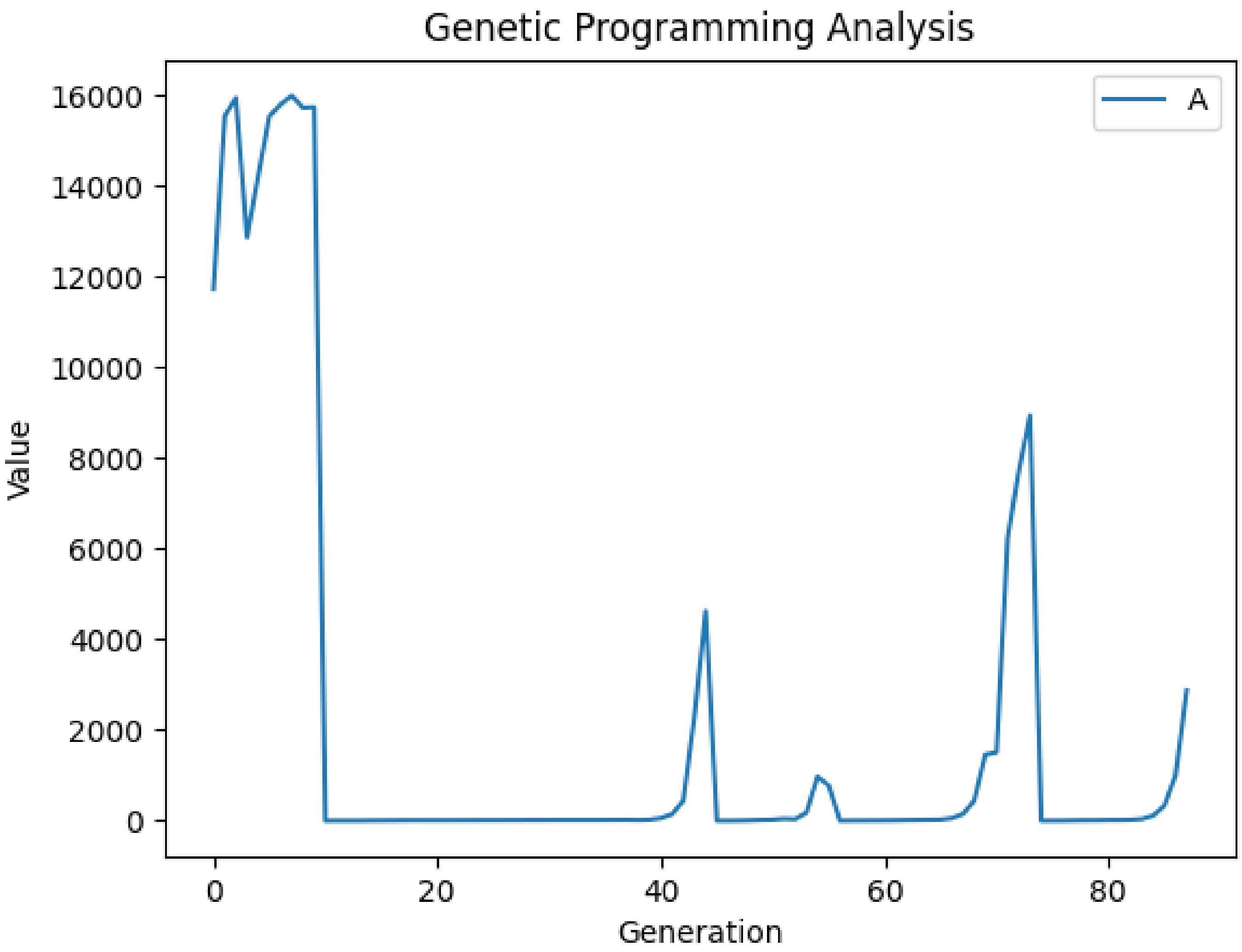


Metrics: Best Agent Weighted Score

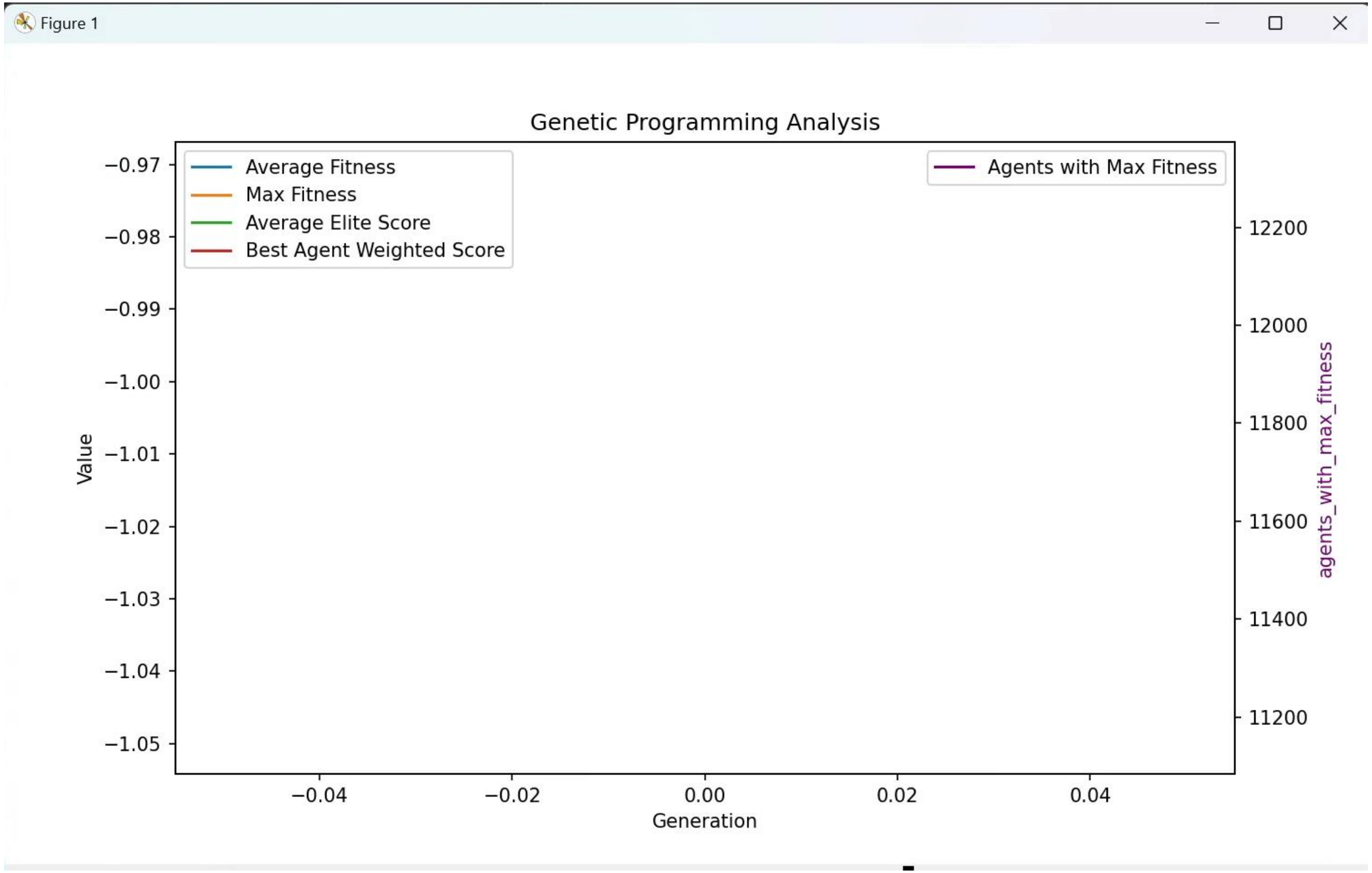


Display : 1920x1080 @ 0,0 (Primary Monitor)

Metrics: Number Best Agents with max fitness



Let’s have fun



Does our code work well?

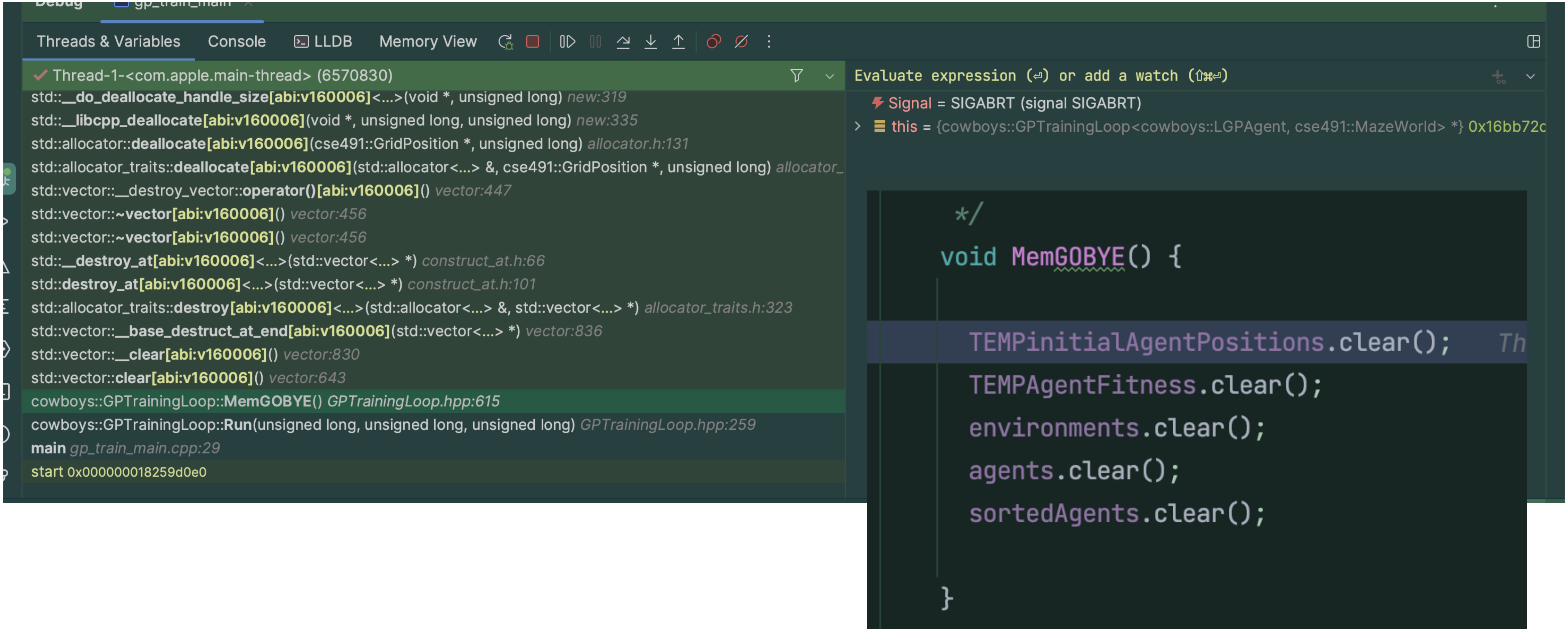
Hell, Yeah it does

(at least on my computer)

Sanitizers

- Google Sanitizers
  - Address Sanitizer (ASan)
  - Leak Sanitizer (LSan)
  - Thread Sanitizer (TSan)
  - Undefined Behavior Sanitizer (UBSsan)
  - Memory Sanitizer (MSan)
- Valgrind

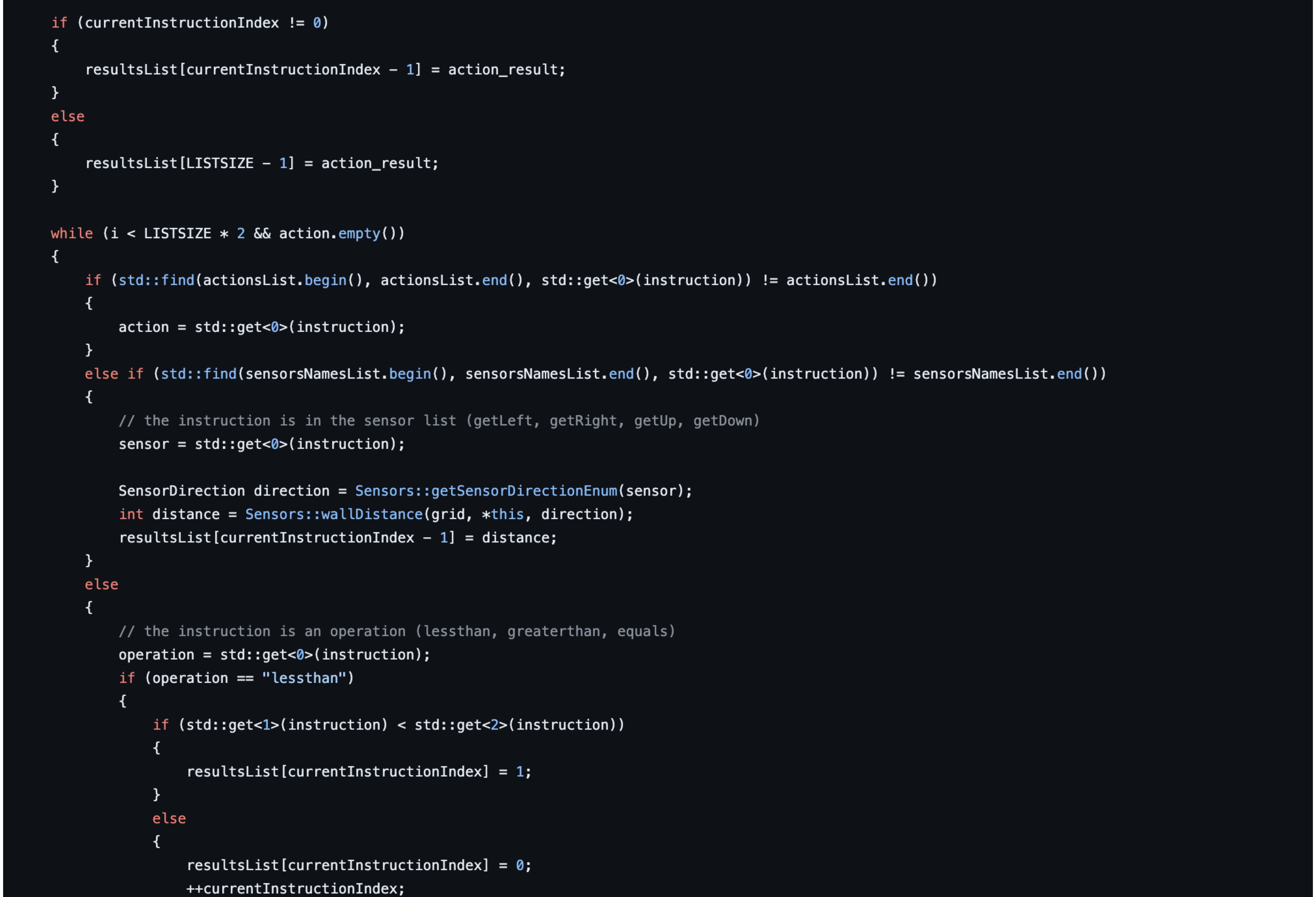
Runtime Errors



```
✓ Thread-1-<com.apple.main-thread> (6570830)
std::__do_deallocate_handle_size[abi:v160006]<...>(void *, unsigned long) new:319
std::__libcpp_deallocate[abi:v160006](void *, unsigned long, unsigned long) new:335
std::allocator::deallocate[abi:v160006](cse491::GridPosition *, unsigned long) allocator.h:131
std::allocator_traits::deallocate[abi:v160006](std::allocator<...> &, cse491::GridPosition *, unsigned long) allocator_
std::vector::__destroy_vector::operator()[abi:v160006]() vector:447
std::vector::~vector[abi:v160006]() vector:456
std::vector::~vector[abi:v160006]() vector:456
std::__destroy_at[abi:v160006]<...>(std::vector<...> *) construct_at.h:66
std::destroy_at[abi:v160006]<...>(std::vector<...> *) construct_at.h:101
std::allocator_traits::destroy[abi:v160006]<...>(std::allocator<...> &, std::vector<...> *) allocator_traits.h:323
std::vector::__base_destruct_at_end[abi:v160006](std::vector<...> *) vector:836
std::vector::__clear[abi:v160006]() vector:830
std::vector::clear[abi:v160006]() vector:643
cowboys::GPTrainingLoop::MemGOBYE() GPTrainingLoop.hpp:615
cowboys::GPTrainingLoop::Run(unsigned long, unsigned long, unsigned long) GPTrainingLoop.hpp:259
main gp_train_main.cpp:29
start 0x000000018259d0e0
```

```
Evaluate expression (⏎) or add a watch (⇧⌘⏎)
Signal = SIGABRT (signal SIGABRT)
> this = {cowboys::GPTrainingLoop<cowboys::LGPAgent, cse491::MazeWorld> *} 0x16bb72c
```

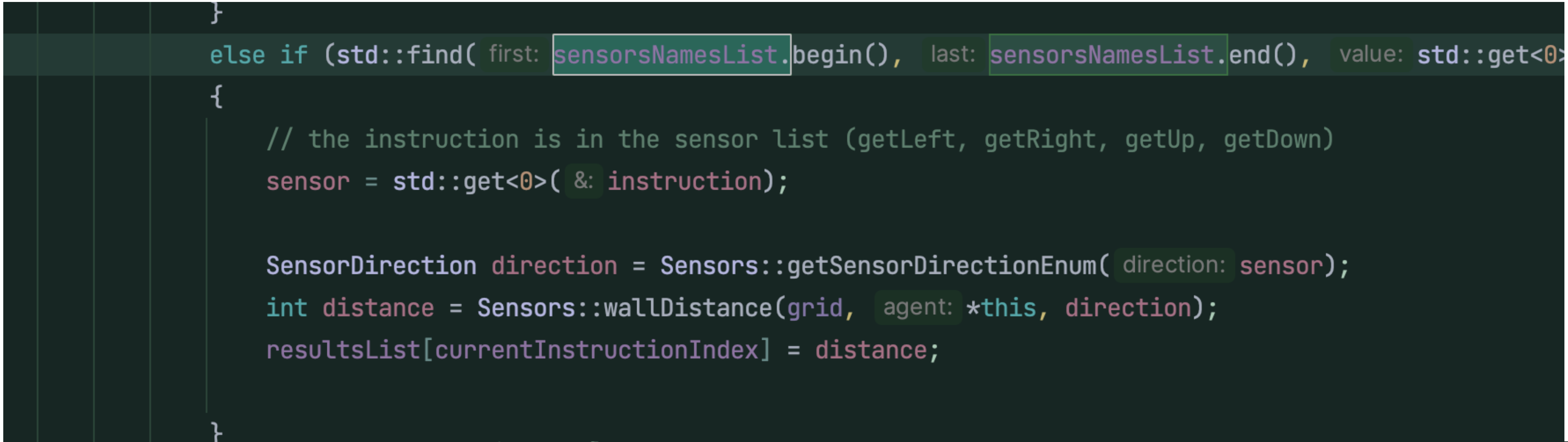
```
    */
  void MemGOBYE() {

      TEMPinitialAgentPositions.clear();    Th
      TEMPAgentFitness.clear();
      environments.clear();
      agents.clear();
      sortedAgents.clear();

  }
```



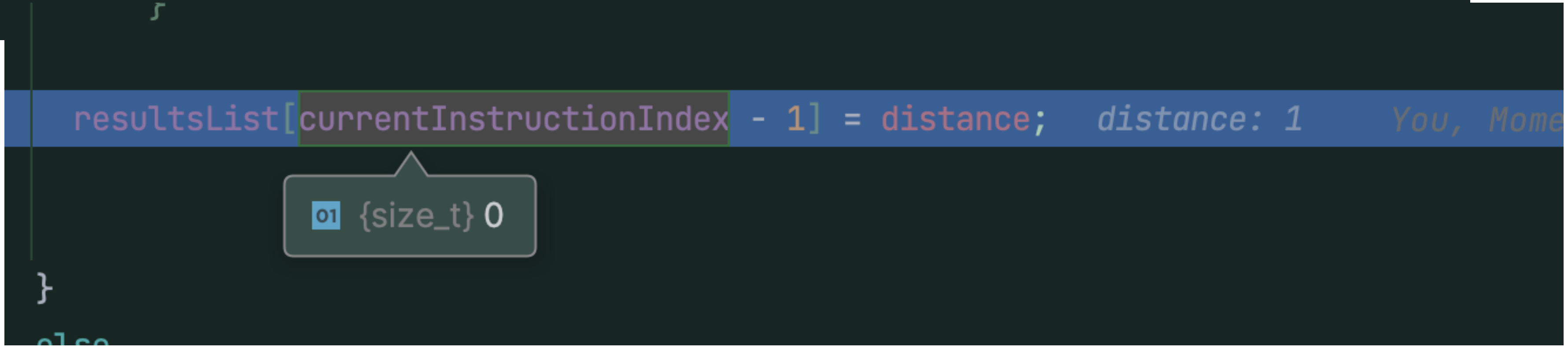
```cpp
    if (currentInstructionIndex != 0)
    {
        resultsList[currentInstructionIndex - 1] = action_result;
    }
    else
    {
        resultsList[LISTSIZE - 1] = action_result;
    }

    while (i < LISTSIZE * 2 && action.empty())
    {
        if (std::find(actionsList.begin(), actionsList.end(), std::get<0>(instruction)) != actionsList.end())
        {
            action = std::get<0>(instruction);
        }
        else if (std::find(sensorsNamesList.begin(), sensorsNamesList.end(), std::get<0>(instruction)) != sensorsNamesList.end())
        {
            // the instruction is in the sensor list (getLeft, getRight, getUp, getDown)
            sensor = std::get<0>(instruction);

            SensorDirection direction = Sensors::getSensorDirectionEnum(sensor);
            int distance = Sensors::wallDistance(grid, *this, direction);
            resultsList[currentInstructionIndex - 1] = distance;
        }
        else
        {
            // the instruction is an operation (lessthan, greaterthan, equals)
            operation = std::get<0>(instruction);
            if (operation == "lessthan")
            {
                if (std::get<1>(instruction) < std::get<2>(instruction))
                {
                    resultsList[currentInstructionIndex] = 1;
                }
                else
                {
                    resultsList[currentInstructionIndex] = 0;
                    ++currentInstructionIndex;
```

Fixed a UB!

G 999 - Debugged and Fixed Undefined Behavior
That took a hot min

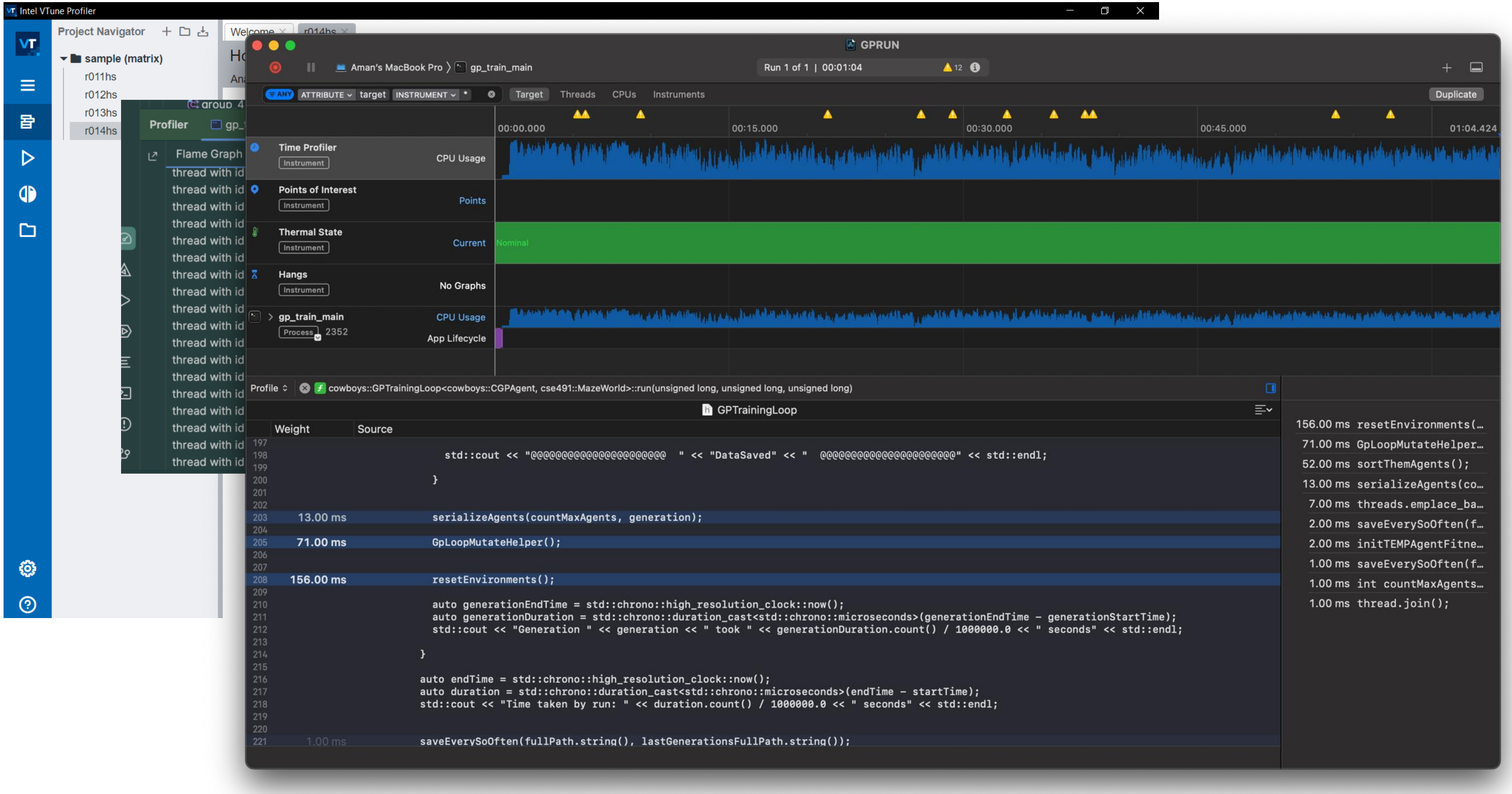


```
}
else if (std::find( first: sensorsNamesList.begin(),  last: sensorsNamesList.end(),  value: std::get<0>
{
    // the instruction is in the sensor list (getLeft, getRight, getUp, getDown)
    sensor = std::get<0>( &: instruction);

    SensorDirection direction = Sensors::getSensorDirectionEnum( direction: sensor);
    int distance = Sensors::wallDistance(grid,  agent: *this, direction);
    resultsList[currentInstructionIndex] = distance;

}
```



```
    resultsList[currentInstructionIndex - 1] = distance;   distance: 1      You, Mome

}
else
```

We profiled to make things efficient

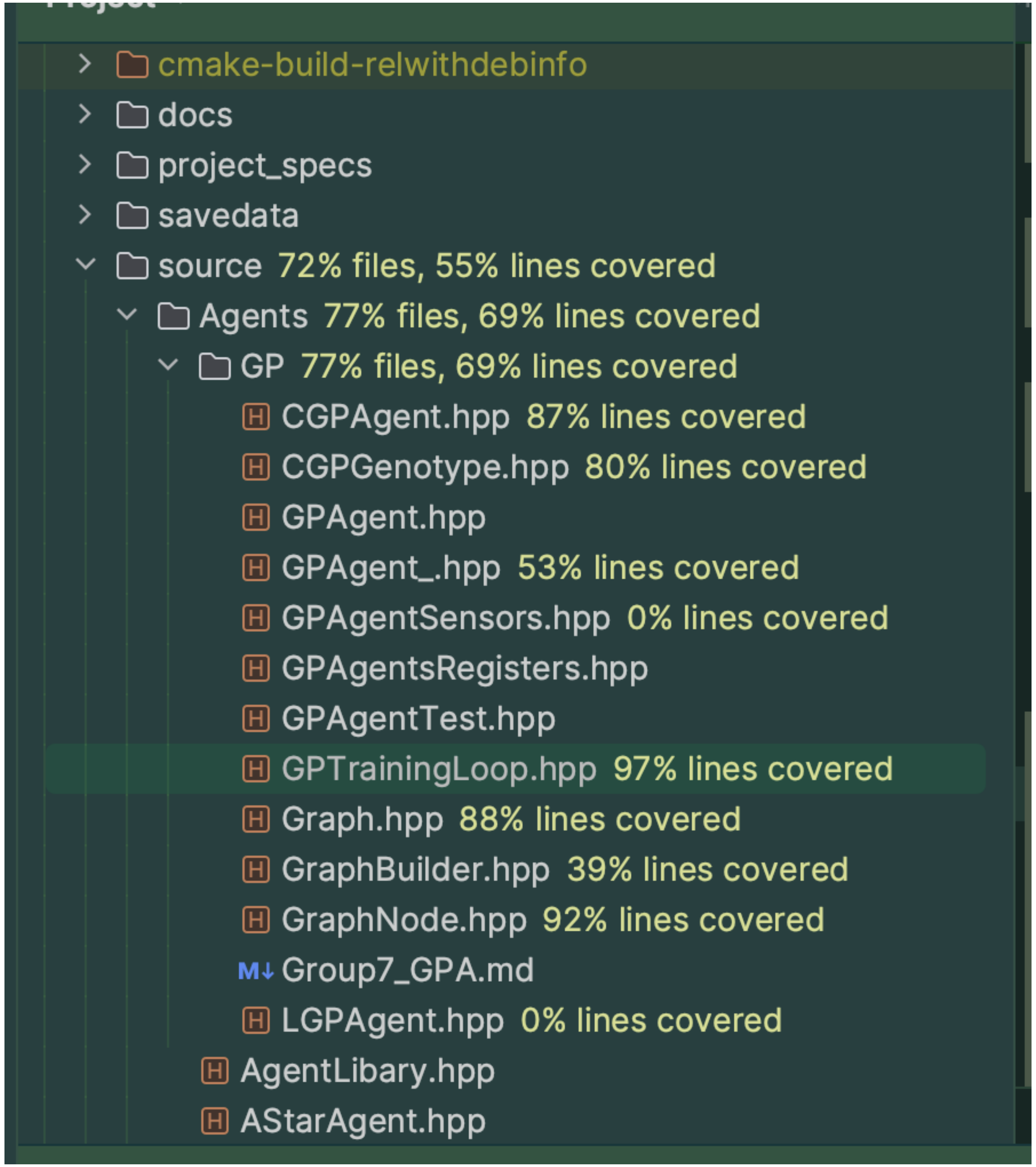


cowboys::GPTrainingLoop<cowboys::CGPAgent, cse491::MazeWorld>::run(unsigned long, unsigned long, unsigned long)

```
                          std::cout << "@@@@@@@@@@@@@@@@@@@@@@  " << "DataSaved" << "  @@@@@@@@@@@@@@@@@@@@@@" << std::endl;

                        }


13.00 ms                serializeAgents(countMaxAgents, generation);

71.00 ms                GpLoopMutateHelper();


156.00 ms               resetEnvironments();

                        auto generationEndTime = std::chrono::high_resolution_clock::now();
                        auto generationDuration = std::chrono::duration_cast<std::chrono::microseconds>(generationEndTime - generationStartTime);
                        std::cout << "Generation " << generation << " took " << generationDuration.count() / 1000000.0 << " seconds" << std::endl;

                      }

                      auto endTime = std::chrono::high_resolution_clock::now();
                      auto duration = std::chrono::duration_cast<std::chrono::microseconds>(endTime - startTime);
                      std::cout << "Time taken by run: " << duration.count() / 1000000.0 << " seconds" << std::endl;


1.00 ms               saveEverySoOften(fullPath.string(), lastGenerationsFullPath.string());
```

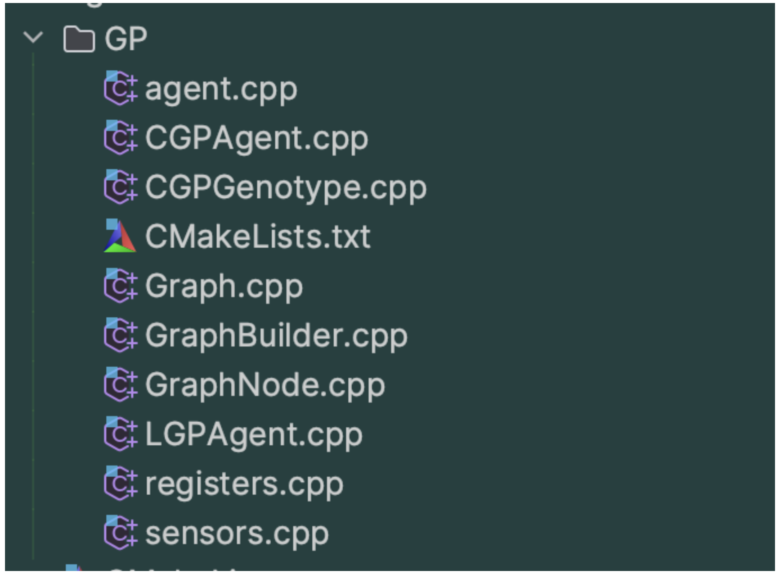
```
156.00 ms resetEnvironments(…
 71.00 ms GpLoopMutateHelper…
 52.00 ms sortThemAgents();
 13.00 ms serializeAgents(co…
  7.00 ms threads.emplace_ba…
  2.00 ms saveEverySoOften(f…
  2.00 ms initTEMPAgentFitne…
  1.00 ms saveEverySoOften(f…
  1.00 ms int countMaxAgents…
  1.00 ms thread.join();
```

Coverage Checks

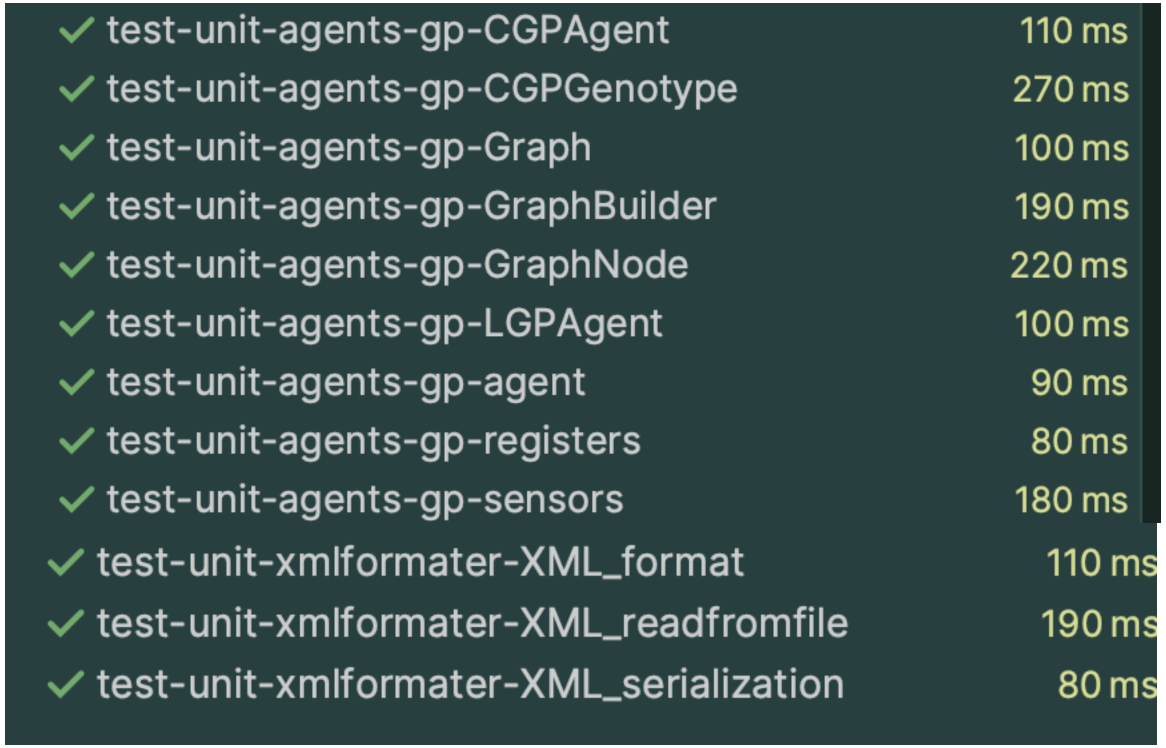


```
Project
> cmake-build-relwithdebinfo
> docs
> project_specs
> savedata
v source  72% files, 55% lines covered
  v Agents  77% files, 69% lines covered
    v GP  77% files, 69% lines covered
        CGPAgent.hpp  87% lines covered
        CGPGenotype.hpp  80% lines covered
        GPAgent.hpp
        GPAgent_.hpp  53% lines covered
        GPAgentSensors.hpp  0% lines covered
        GPAgentsRegisters.hpp
        GPAgentTest.hpp
        GPTrainingLoop.hpp  97% lines covered
        Graph.hpp  88% lines covered
        GraphBuilder.hpp  39% lines covered
        GraphNode.hpp  92% lines covered
        Group7_GPA.md
        LGPAgent.hpp  0% lines covered
      AgentLibary.hpp
      AStarAgent.hpp
```

Unit Tests



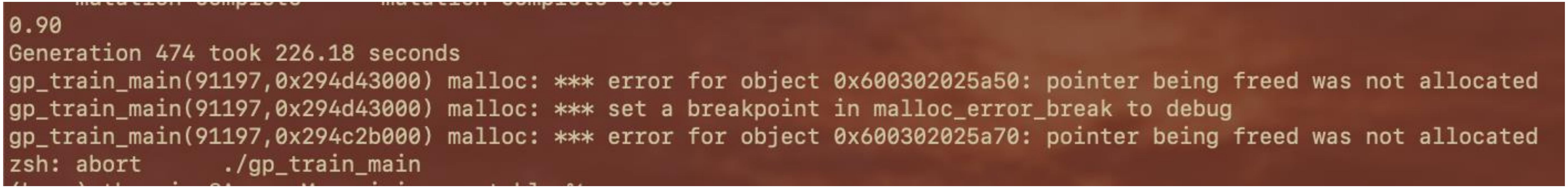
```
⌄ GP
    agent.cpp
    CGPAgent.cpp
    CGPGenotype.cpp
    CMakeLists.txt
    Graph.cpp
    GraphBuilder.cpp
    GraphNode.cpp
    LGPAgent.cpp
    registers.cpp
    sensors.cpp
```



```
✓ test-unit-agents-gp-CGPAgent                 110 ms
✓ test-unit-agents-gp-CGPGenotype              270 ms
✓ test-unit-agents-gp-Graph                    100 ms
✓ test-unit-agents-gp-GraphBuilder             190 ms
✓ test-unit-agents-gp-GraphNode                220 ms
✓ test-unit-agents-gp-LGPAgent                 100 ms
✓ test-unit-agents-gp-agent                     90 ms
✓ test-unit-agents-gp-registers                 80 ms
✓ test-unit-agents-gp-sensors                  180 ms
✓ test-unit-xmlformater-XML_format             110 ms
✓ test-unit-xmlformater-XML_readfromfile       190 ms
✓ test-unit-xmlformater-XML_serialization       80 ms
```

Main Demo

Problems that we might face



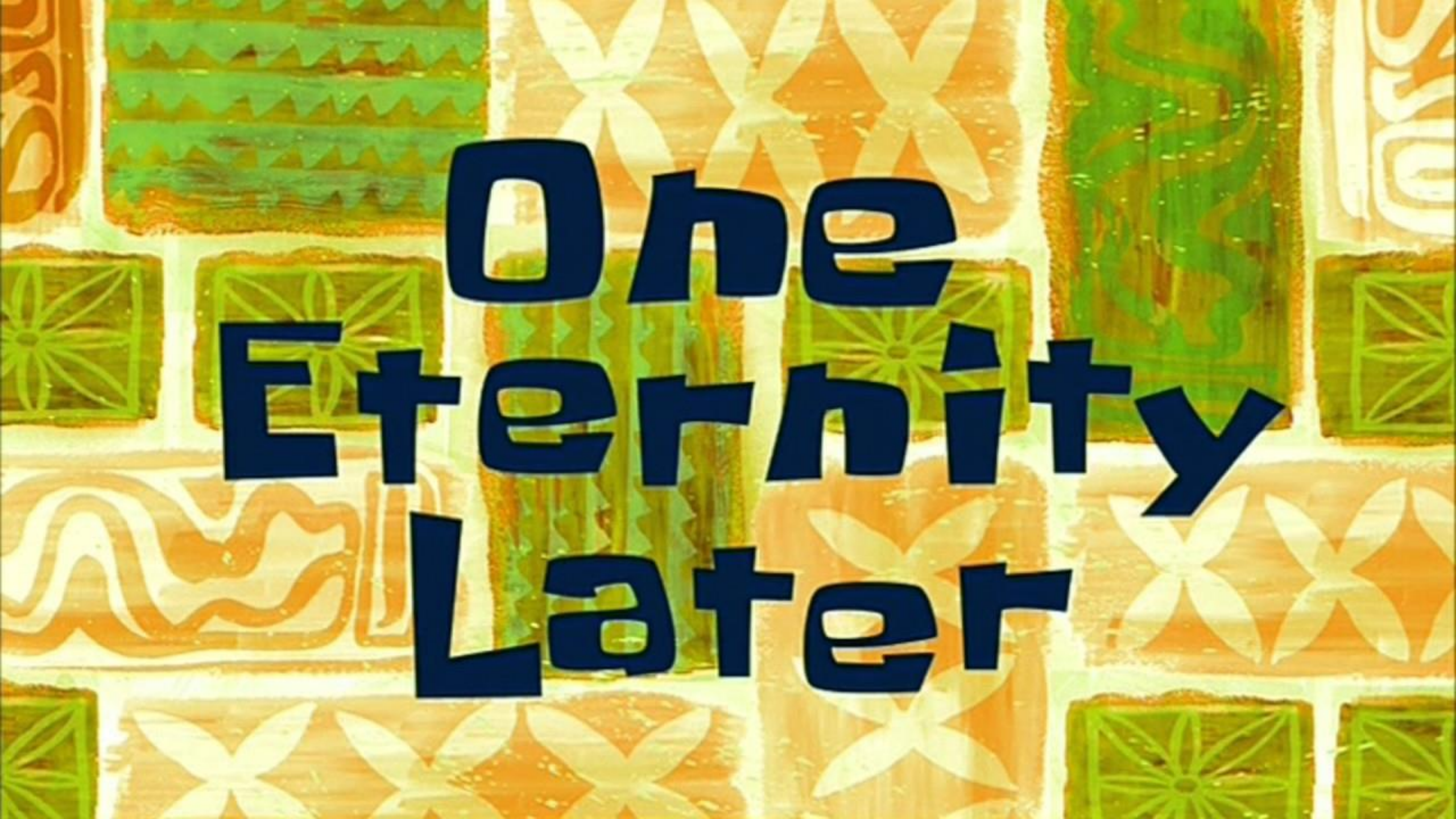
```
0.90
Generation 474 took 226.18 seconds
gp_train_main(91197,0x294d43000) malloc: *** error for object 0x600302025a50: pointer being freed was not allocated
gp_train_main(91197,0x294d43000) malloc: *** set a breakpoint in malloc_error_break to debug
gp_train_main(91197,0x294c2b000) malloc: *** error for object 0x600302025a70: pointer being freed was not allocated
zsh: abort      ./gp_train_main
```

- Restart seeding
    - Seeding Determinism
- Agents Getting stuck at local minimum

Potential Future Improvements

- Better/different fitness functions for more specialized behaviors
- Deploy in HPCC
- Integration Testing?

Who to blame if things don't work



One Eternity Later