Initial Implementation

Первичная реализация генетического алгоритма

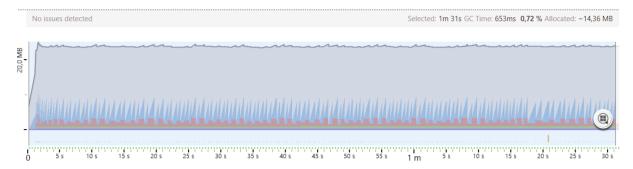
```
▲ 〒100 % All Calls • 118 419 ms
 48,9 % Main • 57 950 ms • GenAlg.Program.Main(String[])
   48,5 % FindSolution • 57 447 ms • GenAlg.Common.GenAlgOptimizer.FindSolution(Random, Int32, TFitnessCalcul.
     32,9 % SelectionTournament • 38 992 ms • GenAlg.Common.Population`2.SelectionTournament()
     7,21 % GetStats • 8 537 ms • GenAlg.Common.Population`2.GetStats()
     ₹5,51% [Unknown] • 6 527 ms
■ 〒100 % All Calls • 880 MB

■ 99,8 % Main • 879 MB • GenAlg.Program.Main(String[])
   ■ 99,8 % FindSolution • 879 MB • GenAlg.Common.GenAlgOptimizer.FindSolution(Random, Int32, TFitnessCalculate
     4 81,8 % Crossover • 720 MB • GenAlg.Common.Population 2.Crossover(Double)

▲ 81,5 % Cross • 718 MB • GenAlg.PushThePoint.Genome.Cross(Genome)

          41,3 % coreclr.dll • 364 MB
        ₹ 39,6 % CreateRange • 349 MB • System.Collections.Immutable.ImmutableArray.CreateRange(IEnumerable)
        ▼0,56 % TolmmutableArray • 5,0 MB • System.Collections.Immutable.ImmutableArray.TolmmutableArray(IEnu
      ▼0,26 % List`1..ctor • 2,3 MB • System.Collections.Generic.List`1..ctor(Int32)
     ▶ 9,47 % Mutate • 83 MB • GenAlg.Common.Population`2.Mutate(Double)
     ▶ 8,36 % SelectionTournament • 74 MB • GenAlg.Common.Population`2.SelectionTournament()
```

Profiling session has finished



ImmutableArray => array[]

Геном перестал быть неизменяемым объектом и операции мутации и скрещивания делаются без дополнительного выделения памяти.

Выделение памяти снизилось с 880 мб до 479 мб.

```
■ 〒100 % All Calls • 130 148 ms

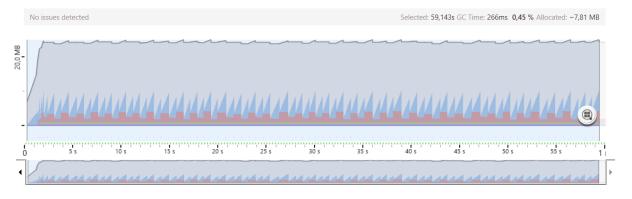
 45,8 % Main • 59 594 ms • GenAlg.Program.Main(String[])
   45,4 % FindSolution • 59 045 ms • GenAlg.Common.GenAlgOptimizer.FindSolution(Random, Int32, TFitnessCalculato
     ▲ 32,8 % SelectionTournament • 42 705 ms • GenAlg.Common.Population`2.SelectionTournament()
       ▶ 32,8 % SelectionTournament • 42 700 ms • GenAlg.Common.Population`2.SelectionTournament(Int32)
       ₹0,00 % corectr.dll • 4,7 ms
         0,00 % [Unknown] • 0,08 ms
     4 6,76 % GetStats • 8 799 ms • GenAlg.Common.Population 2.GetStats()
       ▶ ७ 6,75 % Calculate • 8 791 ms • GenAlg.PushThePoint.FitnessCalculator.Calculate(Genome)
■ $\P$100 % All Calls • 479 MB

■ 99,7 % Main • 478 MB • GenAlg.Program.Main(String[])

■ 99,7 % FindSolution • 478 MB • GenAlg.Common.GenAlgOptimizer.FindSolution(Random, Int32, TFitnessCalculator)

    ▲ 99,4 % SelectionTournament • 477 MB • GenAlg.Common.Population`2.SelectionTournament()
      ▲ 99,4 % SelectionTournament • 477 MB • GenAlg.Common.Population`2.SelectionTournament(Int32)
        ▲ 83,9 % Clone • 402 MB • GenAlg.PushThePoint.Genome.Clone()
         ₹80,4 % ToArray • 385 MB • System.Collections.Generic.EnumerableHelpers.ToArray(IEnumerable)
           2,34 % corectr.dll • 11 MB
         ▼1,16 % ToArray • 5,6 MB • System.Linq.Enumerable.ToArray(IEnumerable)
         12,3 % coreclr.dll • 59 MB
```

Profiling session has finished



Array Pool

Вместо выделения памяти для новых массивов вместо операции new [] был использован ArrayPool<>.Shared.Rent.

Выделение памяти не изменилось.

```
■ ▼100 % All Calls • 130 044 ms

 49,3 % Main • 64 112 ms • GenAlg.Program.Main(String[])
   ■ 49,0 % FindSolution • 63 712 ms • GenAlg.Common.GenAlgOptimizer.FindSolution(Random, Int32, TFitnessCalculator)
     ▶ 34,7 % SelectionTournament • 45 069 ms • GenAlg.Common.Population`2.SelectionTournament()
     ▶ 8,73 % GetStats • 11 354 ms • GenAlg.Common.Population`2.GetStats()
Call Tree
■ 〒100 % All Calls • 482 MB

■ 99,7 % Main • 480 MB • GenAlg.Program.Main(String[])
   ■ 99,7 % FindSolution • 480 MB • GenAlq.Common.GenAlgOptimizer.FindSolution(Random, Int32, TFitnessCalculator)

■ 99,4 % SelectionTournament • 479 MB • GenAlg.Common.Population`2.SelectionTournament()

       ■ 99,4 % SelectionTournament • 479 MB • GenAlg.Common.Population`2.SelectionTournament(Int32)

■ 99,4 % Clone • 479 MB • GenAlg.PushThePoint.Genome.Clone()

          4 97,2 % Genome..ctor • 468 MB • GenAlg.PushThePoint.Genome..ctor(GenomeAction[])
            ₹97,2 % Rent • 468 MB • System.Buffers.TIsOverPerCoreLockedStacksArrayPool`1.Rent(Int32)
Profiling session has finished
 No issues detected
                                                               Selected: 1m 32s GC Time: 239ms 0,26 % Allocated: ~8,01 MB
                                                                                                   Heap generation 0
                                                                                                   Heap generation 1
                                                                                                  Heap generation 2
                                                                                                  M LOH and POH
                                                                                                  M Allocated in LOH since GC
     5s 10s 15s 20s 25s 30s 35s 40s 45s 50s 55s 1m 5s 10s 15s 20s
```

Cache Fitness

Операция определения fitness представляет собой алгоритм, так что она достаточно дорогая. Теперь для неизмененного генома fitness вычисляется 1 раз и кэшируется.

Скорость работы выросла в 3.4 раза.



ArrayPool.Create

Теперь вместо использования общего ArrayPool создаётся отдельный объект методом ArrayPool.Create и в дальнейшем используется для выделения памяти для массивов.

Выделение памяти снизилось с 480 мб до 23 мб.

```
Call Tree

■ 〒100 % All Calls • 23 MB
 ■ 94,1 % Main • 22 MB • GenAlg.Program.Main(String[])
  ■ 92,7 % FindSolution • 21 MB • GenAlg.Common.GenAlgOptimizer.FindSolution(Random, Int32, TFitnessCalculator)
    ■ 87,0 % SelectionTournament • 20 MB • GenAlg.Common.Population`2.SelectionTournament()
      ▲ 87,0 % SelectionTournament • 20 MB • GenAlg.Common.Population`2.SelectionTournament(Int32)
        ▲ 87,0 % Clone • 20 MB • GenAlg.PushThePoint.Genome.Clone()
            82,6 % corectr.dll • 19 MB
          ▶ 4,39 % Genome..ctor • 1,0 MB • GenAlg.PushThePoint.Genome..ctor(GenomeAction[], Double, Boolean)
     ▶ 4,38 % Population`2...ctor • 1,0 MB • GenAlg.Common.Population`2...ctor(Random, Int32, TFitnessCalculator)
     ▶ $\top$0,88 % WriteLine • 0,2 MB • System.Console.WriteLine(Object)
    ₹0,44 % AddWithResize • 0,1 MB • System.Collections.Generic.List`1.AddWithResize(T)
   ▶ 0,48 % corectr.dll • 0,1 MB
   ▼0,45 % Deserialize • 0,1 MB • System.Text.Json.JsonSerializer.Deserialize(JsonNode, JsonSerializerOptions)
   ₹0,44 % Parse • 0,1 MB • System.TextJson.Nodes.JsonNode.Parse(String, Nullable, JsonDocumentOptions)
```

Итог

Удалось снизить время работы программы со 118 секунд до 36 секунд. Удалось снизить выделение памяти с 880 мб до 23 мб.