

Национальный Исследовательский Университет
«Московский Энергетический Институт»
Кафедра прикладной математики и искусственного интеллекта

Тема: Мониторинг производительности .NET приложения с помощью JetBrains
dotTrace.

Студент: Ростовых Александра

Москва 2021

Цель работы

Научиться получать точную информацию об узких местах в производительности приложений, созданных на основе платформы .NET Framework. Получить навыки профилирования приложения в нескольких режимах, включая tracing (на основе подсчета числа вызовов), sampling (на основе подсчета времени исполнения) и построчный режим (детальный анализ производительности).

1. Подготовить тестируемое приложение

- Разработать с помощью Visual Studio консольное приложение
- Реализовать в приложении два алгоритма сортировки (быструю сортировку и сортировку пузырьком)

Быстрая сортировка:

```
static int Partition<T>(T[] m, int a, int b) where T : IComparable
{
    int i = a;
    for (int j = a; j <= b; j++)          // просматриваем с a по b
    {
        if (m[j].CompareTo(m[b]) <= 0)  // если элемент m[j] не
        превосходит m[b],
        {
            (m[i], m[j]) = (m[j], m[i]); // меняем местами m[j] и
            m[a], m[a+1], m[a+2] и так далее..
            i++;                          // таким образом последний
            обмен: m[b] и m[i], после чего i++
        }
    }
    return i - 1;                        // в индексе i хранится
    <новая позиция элемента m[b]> + 1
}

static void Quicksort<T>(T[] m, int a, int b) where T : IComparable//
a - начало подмножества, b - конец
{
    // для первого вызова: a =
    0, b = <элементов в массиве> - 1
    if (a >= b) return;
    int c = Partition(m, a, b);
    Quicksort(m, a, c - 1);
    Quicksort(m, c + 1, b);
}
```

Сортировка пузырьком:

```
static void BubbleSort1<T>(T[] mas) where T : IComparable
{
    for (int i = 0; i < mas.Length - 1; i++)
    {
        for (int j = i + 1; j < mas.Length - 1; j++)
        {
            if (mas[i].CompareTo(mas[j]) > 0)
            {

```

```

        (mas[i], mas[j]) = (mas[j], mas[i]);
    }
}
}
}

```

- Сортируемые данные должны загружаться из файла
- Создать несколько наборов несортированных тестовых данных (в т.ч. данные размером 1Кб, 2Кб)

Созданы файлы:

1. Файл с 7000 несортированных числовых данных (объем 40,2 Кб)
2. Файл с 3000 несортированных числовых данных (объем 17,2 Кб)
3. Файл с 1000 строковых несортированных данных (объем 977 Кб)

2. Запустить приложение в режиме профилирования sampling.

Определить наименее производительные функции.

Файл с 7000 числовых данных:

All Calls Overview			
4 functions	Group by: None	Class	Namespace Assembly
<input type="checkbox"/> Show system functions			
Function Name	Time, ms	Own Time, ms	Ovr
RostovkyhAD_Lab9.Program.Main	1 131	0	
RostovkyhAD_Lab9.Program.BubbleSort1	979	732	
RostovkyhAD_Lab9.Program.Quicksort	10	0	
RostovkyhAD_Lab9.Program.Partition	10	10	

Functions called by RostovkyhAD_Lab9.Program.Main			
Function Name	Time, ms	Own Time, ms	Ovr
RostovkyhAD_Lab9.Program.BubbleSort1	979	732	
System.Convert.ToInt32	73	0	
System.IO.StreamReader..ctor	37	0	
System.IO.StreamReader.get_EndOfStream	32	0	
RostovkyhAD_Lab9.Program.Quicksort	10	0	

Файл с 3000 числовых данных:

All Calls Overview			
4 functions	Group by: None	Class	Namespace Assembly
<input type="checkbox"/> Show system functions			
Function Name	Time, ms	Own Time, ms	Ovr
RostovkyhAD_Lab9.Program.Main	348	0	
RostovkyhAD_Lab9.Program.BubbleSort1	213	102	
RostovkyhAD_Lab9.Program.Quicksort	6	0	
RostovkyhAD_Lab9.Program.Partition	6	0	

Functions called by RostovkyhAD_Lab9.Program.Main			
Function Name	Time, ms	Own Time, ms	Ovr
RostovkyhAD_Lab9.Program.BubbleSort1	213	102	
System.Convert.ToInt32	66	0	
System.IO.StreamReader..ctor	30	0	
System.IO.StreamReader.get_EndOfStream	24	0	
System.Collections.Generic.List`1.Add	9	0	
RostovkyhAD_Lab9.Program.Quicksort	6	0	

Файл с 1000 строковых данных:

All Calls Overview			
4 functions	Group by: None Class Namespace Assembly	<input type="checkbox"/> Show system functions	
Function Name	Time, ms	Own Time, ms	Own
RostovykhAD_Lab9.Program.Main	269	6	
RostovykhAD_Lab9.Program.BubbleSort1	176	6	
RostovykhAD_Lab9.Program.Quicksort	45	0	
RostovykhAD_Lab9.Program.Partition	45	0	

Functions called by RostovykhAD_Lab9.Program.Main			
Function Name	Time, ms	Own Time, ms	Own/Tot
RostovykhAD_Lab9.Program.BubbleSort1	176	6	
RostovykhAD_Lab9.Program.Quicksort	45	0	
System.IO.StreamReader..ctor	24	0	
System.IO.StreamReader.get_EndOfStream	12	0	
System.IO.StreamReader.ReadLine	6	6	1

Видим, что сортировка пузырьком работает во много раз медленнее, чем быстрая сортировка во всех трех случаях.

3. Запустить приложение в режиме профилирования tracing. Получить результат. Определить узкие места в реализации программы.

Файл с 7000 числовых данных:

All Calls Overview						
4 functions	Group by: None Class Namespace Assembly	<input type="checkbox"/> Show system functions				
Function Name	Time, ms	Avg Time, ms	Own Time, ms	Own/Total	Calls	
RostovykhAD_Lab9.Program.Main	1 527	1 527	35	2,30%	1	
RostovykhAD_Lab9.Program.BubbleSort1	1 242	1 242	254	20,46%	1	
RostovykhAD_Lab9.Program.Quicksort	101	0	1	0,86%	1 327	
RostovykhAD_Lab9.Program.Partition	100	0	3	3,18%	663	

Functions called by RostovykhAD_Lab9.Program.Main			
Function Name	Time, ms	Avg Time, ms	Own Time, ms
RostovykhAD_Lab9.Program.BubbleSort1	1 242	1 242	25
RostovykhAD_Lab9.Program.Quicksort	101	101	
System.IO.StreamReader..ctor	62	62	
System.IO.StreamReader.ReadLine	46	0	
System.IO.StreamReader.get_EndOfStream	37	0	
System.Collections.Generic.List`1.Add	2	0	
System.Collections.Generic.List`1..ctor	1	1	

Файл с 3000 числовых данных:

All Calls Overview							
4 functions Group by: None Class Namespace Assembly <input type="checkbox"/> Show system functions							
Function Name	Time, ms	Avg Time, ms	Own Time, ms	Own/Total	Calls		
RostovkyhAD_Lab9. Program .Main	700	700	3	0,46%	1		
RostovkyhAD_Lab9. Program .BubbleSort1	568	568	95	16,72%	1		
RostovkyhAD_Lab9. Program .Quicksort	60	0	1	0,89%	1 327		
RostovkyhAD_Lab9. Program .Partition	59	0	2	2,58%	663		

Functions called by RostovkyhAD_Lab9.Program.Main			
Function Name	Time, ms	Avg Time, ms	Own Time, ms
RostovkyhAD_Lab9. Program .BubbleSort1	568	568	95
RostovkyhAD_Lab9. Program .Quicksort	60	60	1
System.IO. StreamReader .ReadLine	25	0	0
System.IO. StreamReader ..ctor	24	24	0
System.IO. StreamReader .get_EndOfStream	19	0	0
System.Collections.Generic. List`1 .Add	1	0	0
System.Collections.Generic. List`1 ..ctor	0	0	0

Файл с 1000 строковых данных:

All Calls Overview							
4 functions Group by: None Class Namespace Assembly <input type="checkbox"/> Show system functions							
Function Name	Time, ms	Avg Time, ms	Own Time, ms	Own/Total	Calls		
RostovkyhAD_Lab9. Program .Main	903	903	3	0,33%	1		
RostovkyhAD_Lab9. Program .BubbleSort1	764	764	129	16,84%	1		
RostovkyhAD_Lab9. Program .Quicksort	65	0	1	0,96%	1 327		
RostovkyhAD_Lab9. Program .Partition	64	0	2	3,41%	663		

Functions called by RostovkyhAD_Lab9.Program.Main			
Function Name	Time, ms	Avg Time, ms	Own Time, ms
RostovkyhAD_Lab9. Program .BubbleSort1	764	764	129
RostovkyhAD_Lab9. Program .Quicksort	65	65	1
System.IO. StreamReader ..ctor	25	25	0
System.IO. StreamReader .ReadLine	23	0	0
System.IO. StreamReader .get_EndOfStream	22	0	0
System.Collections.Generic. List`1 .Add	1	0	0
System.Collections.Generic. List`1 ..ctor	0	0	0

Наблюдаем в последнем столбце количество вызовов каждой функции. Так как в быстрой сортировке мы рекурсивно вызываем эту же функцию, видим, что количество вызовов Quicksort больше всего.

4. Запустить приложение в режиме профилирования line by line.
Получить информацию какие строки кода исполняются чаще всего.

Файл с 7000 числовых данных:
 Сортировка пузырьком:

Overview

4 functions Group by: None Class Namespace As

Function Name Time, ms

- RostovkyhAD_Lab9.Program.Main 2.2
- RostovkyhAD_Lab9.Program.BubbleSort1 1.9**
- RostovkyhAD_Lab9.Program.Quicksort
- RostovkyhAD_Lab9.Program.Partition

Functions called by RostovkyhAD_Lab9.Program.BubbleSort1

Function Name

- System.Globalization.CompareInfo.Compare
- System.String.Compare
- System.String.CompareTo
- System.Runtime.CompilerServices.CastHelpers.StelemRef

Source View - RostovkyhAD_Lab9.Program.BubbleSort1

Program.cs Decompiled source

```

4: namespace RostovkyhAD_Lab9
5: {
6:     class Program
7:     {
8:         static void BubbleSort1<T>(T[] mas) where T : IComparable
9:         {
10:             for (int i = 0; i < mas.Length - 1; i++)
11:             {
12:                 for (int j = i + 1; j < mas.Length - 1; j++)
13:                 {
14:                     if (mas[i].CompareTo(mas[j]) > 0)
15:                     {
16:                         (mas[i], mas[j]) = (mas[j], mas[i]);
17:                     }
18:                 }
19:             }
20:         }
21:     }

```

Функции для быстрой сортировки:

4 functions Group by: None Class Namespace As

Function Name Time, ms

- RostovkyhAD_Lab9.Program.Main 2.2
- RostovkyhAD_Lab9.Program.BubbleSort1 1.9
- RostovkyhAD_Lab9.Program.Quicksort 1.9**
- RostovkyhAD_Lab9.Program.Partition

Functions called by RostovkyhAD_Lab9.Program.Quicksort

Function Name

- System.Globalization.CompareInfo.Compare
- System.String.Compare
- System.String.CompareTo
- System.Runtime.CompilerServices.CastHelpers.StelemRef

Source View - RostovkyhAD_Lab9.Program.Quicksort

Program.cs Decompiled source

```

85: }
86:
87: static void Quicksort<T>(T[] m, int a, int b) where T : IComparable // для первого вызова: a = 0, b = m.Length - 1
88: {
89:     if (a >= b) return;
90:     if (a >= b) return;
91:     if (a >= b) return;
92:     int c = Partition(m, a, b);
93:     Quicksort(m, a, c - 1);
94:     Quicksort(m, c + 1, b);
95: }

```

4 functions Group by: None Class Namespace As

Function Name Time, ms

- RostovkyhAD_Lab9.Program.Main 2.2
- RostovkyhAD_Lab9.Program.BubbleSort1 1.9
- RostovkyhAD_Lab9.Program.Quicksort 1.9
- RostovkyhAD_Lab9.Program.Partition 1.9**

Functions called by RostovkyhAD_Lab9.Program.Partition

Function Name

- System.String.CompareTo
- System.Runtime.CompilerServices.CastHelpers.StelemRef

Source View - RostovkyhAD_Lab9.Program.Partition

Program.cs Decompiled source

```

71: */
72:
73: static int Partition<T>(T[] m, int a, int b) where T : IComparable
74: {
75:     int i = a;
76:     for (int j = a; j <= b; j++) // просматриваем с a по b
77:     {
78:         if (m[j].CompareTo(m[a]) <= 0) // если элемент m[j] не превосходит m[a],
79:         {
80:             (m[j], m[a]) = (m[a], m[j]); // меняем местами m[j] и m[a]
81:             i++; // таким образом последний обмен: m[b] и m[i]
82:         }
83:     }
84:     return i - 1; // в индексе i хранится <новая позиция эле
85: }

```

Файл с 3000 числовых данных:

Сортировка пузырьком:

4 functions Group by: **None** Class Namespace As Program.cs Decompiled source

Function Name

- RostovkyhAD_Lab9.Program.Main
- RostovkyhAD_Lab9.Program.BubbleSort1**
- RostovkyhAD_Lab9.Program.Quicksort
- RostovkyhAD_Lab9.Program.Partition

Functions called by RostovkyhAD_Lab9.Program.BubbleSort1

Function Name

- System.Globalization.CompareInfo.Compare
- System.String.Compare
- System.String.CompareTo
- System.Runtime.CompilerServices.CastHelpers.SteemRef

```

4      namespace RostovkyhAD_Lab9
5      {
6          class Program
7          {
8              static void BubbleSort1<T>(T[] mas) where T : IComparable
9              {
10                 for (int i = 0; i < mas.Length - 1; i++)
11                 {
12                     for (int j = i + 1; j < mas.Length - 1; j++)
13                     {
14                         if (mas[i].CompareTo(mas[j]) > 0)
15                         {
16                             (mas[i], mas[j]) = (mas[j], mas[i]);
17                         }
18                     }
19                 }
20             }
21         }
    
```

Функции для быстрой сортировки:

4 functions Group by: **None** Class Namespace As Program.cs Decompiled source

Function Name

- RostovkyhAD_Lab9.Program.Main
- RostovkyhAD_Lab9.Program.BubbleSort1
- RostovkyhAD_Lab9.Program.Quicksort**
- RostovkyhAD_Lab9.Program.Partition

```

85     }
86
87     static void Quicksort<T>(T[] m, int a, int b) where T : IComparable//
88     { // для первого вызова: a = 0
89         if (a >= b) return;
90         if (a >= b) return;
91         if (a >= b) return;
92         int c = Partition(m, a, b);
93         Quicksort(m, a, c - 1);
94         Quicksort(m, c + 1, b);
95     }
    
```

4 functions Group by: **None** Class Namespace As Program.cs Decompiled source

Function Name

- RostovkyhAD_Lab9.Program.Main
- RostovkyhAD_Lab9.Program.BubbleSort1
- RostovkyhAD_Lab9.Program.Quicksort
- RostovkyhAD_Lab9.Program.Partition**

Functions called by RostovkyhAD_Lab9.Program.Partition

Function Name

- System.String.CompareTo
- System.Runtime.CompilerServices.CastHelpers.SteemRef

```

71     */
72
73     static int Partition<T>(T[] m, int a, int b) where T : IComparable
74     {
75         int i = a;
76         for (int j = a; j <= b; j++) // просматриваем с а по b
77         {
78             if (m[j].CompareTo(m[b]) <= 0) // просматриваем с а по b
79             {
80                 (m[i], m[j]) = (m[j], m[i]); // просматриваем с а по b
81                 i++; // просматриваем с а по b
82             }
83         }
84         return i - 1;
85     }
    
```

Файл с 1000 строковых данных:

Сортировка пузырьком:

4 functions | Group by: None | Class | Namespace | As | Program.cs | Decompiled source

Function Name

- RostovkyhAD_Lab9.Program.Main
- RostovkyhAD_Lab9.Program.BubbleSort1**
- RostovkyhAD_Lab9.Program.QuickSort
- RostovkyhAD_Lab9.Program.Partition

Functions called by RostovkyhAD_Lab9.Program.BubbleSort1

Function Name

- System.Globalization.CompareInfo.Compare
- System.String.Compare
- System.String.CompareTo
- System.Runtime.CompilerServices.CastHelpers.StealRef

```

4 | namespace RostovkyhAD_Lab9
5 | {
6 |     class Program
7 |     {
8 |         static void BubbleSort1<T>(T[] mas) where T : IComparable
9 |         {
10 |             for (int i = 0; i < mas.Length - 1; i++)
11 |             {
12 |                 for (int j = i + 1; j < mas.Length - 1; j++)
13 |                 {
14 |                     if (mas[i].CompareTo(mas[j]) > 0)
15 |                     {
16 |                         (mas[i], mas[j]) = (mas[j], mas[i]);
17 |                     }
18 |                 }
19 |             }
20 |         }
21 |     }
22 | }
  
```

Функции для быстрой сортировки:

4 functions | Group by: None | Class | Namespace | As | Program.cs | Decompiled source

Function Name

- RostovkyhAD_Lab9.Program.Main
- RostovkyhAD_Lab9.Program.BubbleSort1
- RostovkyhAD_Lab9.Program.QuickSort**
- RostovkyhAD_Lab9.Program.Partition

```

85 |     }
86 |
87 |     static void QuickSort<T>(T[] m, int a, int b) where T : IComparable//
88 |     {
89 |         if (a >= b) return; // для первого вызова: a = 0
90 |         if (a >= b) return;
91 |         if (a >= b) return;
92 |         int c = Partition(m, a, b);
93 |         QuickSort(m, a, c - 1);
94 |         QuickSort(m, c + 1, b);
95 |     }
  
```

4 functions | Group by: None | Class | Namespace | As | Program.cs | Decompiled source

Function Name

- RostovkyhAD_Lab9.Program.Main
- RostovkyhAD_Lab9.Program.BubbleSort1
- RostovkyhAD_Lab9.Program.QuickSort
- RostovkyhAD_Lab9.Program.Partition**

Functions called by RostovkyhAD_Lab9.Program.Partition

Function Name

- System.String.CompareTo
- System.Runtime.CompilerServices.CastHelpers.StealRef

```

71 |     */
72 |
73 |     static int Partition<T>(T[] m, int a, int b) where T : IComparable
74 |     {
75 |         int i = a;
76 |         for (int j = a; j <= b; j++) // просматриваем с a по b
77 |         {
78 |             if (m[j].CompareTo(m[b]) <= 0) // если элемент m[j] не превосх
79 |             {
80 |                 (m[i], m[j]) = (m[j], m[i]); // меняем местами m[j] и m[a],
81 |                 i++; // таким образом последний обм
82 |             }
83 |         }
84 |         return i - 1; // в индексе i хранится <новая
85 |     }
  
```

5. Используя полученную информацию улучшить реализацию и проверить это с помощью повтора шагов 2-4

Попробуем изменить наши алгоритмы сортировки

Теперь в быстрой сортировке попробуем взять за опорный элемент середину массива

Общий алгоритм будет такой:

1. вводятся указатели *first* и *last* для обозначения начального и конечного элементов последовательности, а также опорный элемент *mid*;

2. вычисляется значение опорного элемента $(first+last)/2$, и заносится в переменную *mid*;
3. указатель *first* смещается с шагом в 1 элемент к концу массива до тех пор, пока $Mas[first] > mid$. А указатель *last* смещается от конца массива к его началу, пока $Mas[last] < mid$;
4. каждые два найденных элемента меняются местами;
5. пункты 3 и 4 выполняются до тех пор, пока $first < last$.

После разбиения последовательности следует проверить условие на необходимость дальнейшего продолжения сортировки его частей.

```
static void Quicksort<T>(T[] mas, int a, int b) where T : IComparable
{
    if (a >= b)
        return;
    int mid = Partition(mas, a, b);
    Quicksort<T>(mas, a, mid);
    Quicksort<T>(mas, mid + 1, b);
}

static int Partition<T>(T[] mas, int a, int b) where T : IComparable
{
    T mid = mas[(a + b) / 2];
    int i = a;
    int j = b;
    while (i <= j)
    {
        while (mas[i].CompareTo(mid) < 0) i++;
        while (mas[j].CompareTo(mid) > 0) j--;
        if (i >= j) break;
        (mas[i], mas[j]) = (mas[j], mas[i]);
        i++; j--;
    }
    return j;
}
```

И для сортировки пузырьком:

```
static void BubbleSort2<T>(T[] mas) where T : IComparable
{
    int k = 0;
    bool flag = false;
    while (!flag)
    {
        flag = true;
        for (int i = 0; i < mas.Length - 1 - k; i++)
        {
            if (mas[i].CompareTo(mas[i + 1]) > 0)
            {
                flag = false;
                (mas[i], mas[i + 1]) = (mas[i + 1], mas[i]);
            }
        }
        k++;
    }
}
```

```

    }
    k++;
}
}

```

Файл с 7000 числовых данных:

All Calls → Overview				
4 functions	Group by: None Class Namespace Assembly	<input type="checkbox"/> Show system functions		
Function Name	Time, ms	Own Time, ms	Own/Total	
RostovykhAD_Lab9. Program .Main	1 273	0	0,00%	
RostovykhAD_Lab9. Program .BubbleSort2	1 111	686	61,76%	
RostovykhAD_Lab9. Program .Quicksort	20	0	0,00%	
RostovykhAD_Lab9. Program .Partition	20	0	0,00%	

Файл с 3000 числовых данных:

All Calls → Overview				
4 functions	Group by: None Class Namespace Assembly	<input type="checkbox"/> Show system functions		
Function Name	Time, ms	Own Time, ms	Own/Total	
RostovykhAD_Lab9. Program .Main	479	10	2,08%	
RostovykhAD_Lab9. Program .BubbleSort2	343	297	86,63%	
RostovykhAD_Lab9. Program .Quicksort	6	0	0,00%	
RostovykhAD_Lab9. Program .Partition	6	6	100,00%	

Файл с 1000 строковых данных:

All Calls → Overview				
4 functions	Group by: None Class Namespace Assembly	<input type="checkbox"/> Show system functions		
Function Name	Time, ms	Own Time, ms	Own/Total	
RostovykhAD_Lab9. Program .Main	412	10	2,33%	
RostovykhAD_Lab9. Program .BubbleSort2	278	11	4,12%	
RostovykhAD_Lab9. Program .Quicksort	53	0	0,00%	
RostovykhAD_Lab9. Program .Partition	53	0	0,00%	

Видим, что результаты не сильно изменились, возможно, даже стали немного хуже, так что можем сделать вывод, что первая реализация алгоритмов сортировки была достаточно успешной!