

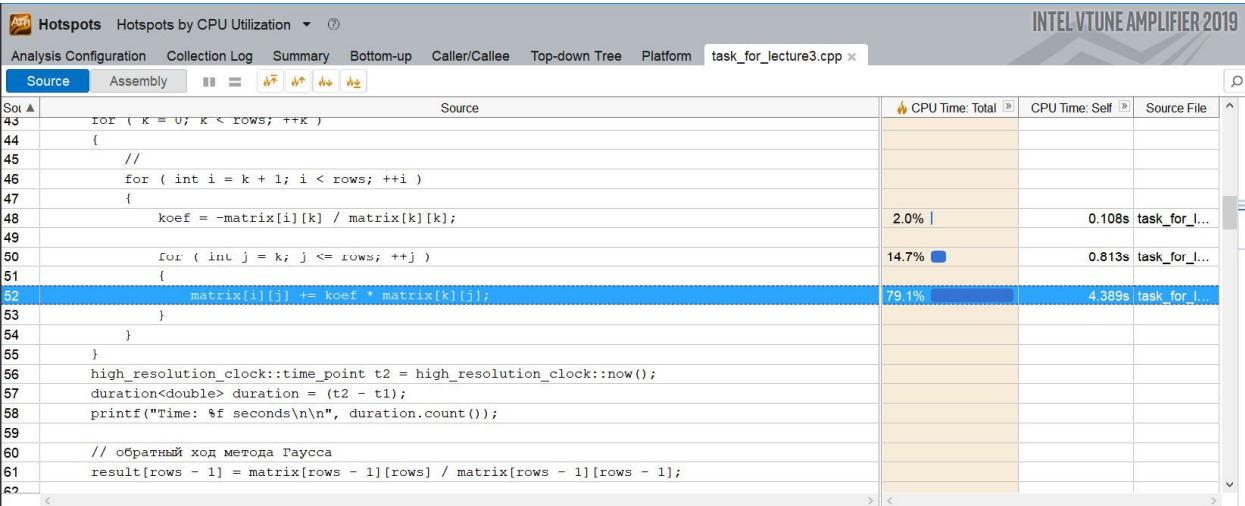
2. Проверка работы последовательного метода Гаусса:

```
Time: 0.000001 seconds

Solution:
x(0) = 1.000000
x(1) = 2.000000
x(2) = 2.000000
x(3) = -0.000000
```

Решение верное (проверено в ручную).

3. Скриншоты Amplifier:



Elapsed Time <sup>?</sup>: 6.427s

<sup>?</sup> CPU Time <sup>?</sup>: 5.551s  
Total Thread Count: 1  
Paused Time <sup>?</sup>: 0s

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions typically results in improving overall application performance.

Function	Module	Module	CPU Time <sup>?</sup>
<a href="#">SerialGaussMethod</a>	IPS.exe	IPS.exe	5.321s
<a href="#">rand</a>	ucrtbased.dll	ucrtbased.dll	0.203s
<a href="#">InitMatrix</a>	IPS.exe	IPS.exe	0.020s
<a href="#">malloc</a>	ucrtbased.dll	ucrtbased.dll	0.006s

\*N/A is applied to non-summable metrics.

4. Скриншоты Inspector:

До:

Locate Deadlocks and Data Races					
<div> Target Analysis Type Collection Log Summary </div>					
Problems					
ID ▲		Type	Sources	Modules	State
⊕ P1	✖	Data race	[Unknown]	ips.exe	New
⊕ P2	✖	Data race	[Unknown]	ips.exe	New
⊕ P3	✖	Data race	[Unknown]	ips.exe	New
⊕ P4	✖	Data race	[Unknown]	ips.exe	New
⊕ P5	✖	Data race	[Unknown]	ips.exe	New

После:

Locate Deadlocks and Data Races					
<div> Target Analysis Type Collection Log Summary </div>					
Problems					
ID ▲		Type	Sources	Modules	State
⊕ P1	✖	Data race	reducer.h; reducer_opadd.h	ips.exe	New

## 5. Проверка работы параллельного метода Гаусса:

```
Time: 0.000001 seconds

Solution:
x(0) = 1.000000
x(1) = 2.000000
x(2) = 2.000000
x(3) = -0.000000
```

Сравнение времени выполнения последовательного и параллельного метода Гаусса, расчёт ускорения:

```
Time serial: 2.804806 seconds
Time parallel: 2.164998 seconds
Compare time: 1.295524 seconds
```

Ссылка на **github**: [https://github.com/u131231/IPS\\_lab2](https://github.com/u131231/IPS_lab2)