

# Architektura výpočetních systémů (AVS 2019)

## Počítačové cvičení č. 1: Měření výkonnosti sekvenčního kódu

---

Gabriel Bordovský (ibordovsky@fit.vutbr.cz)

Filip Kuklis (ikuklis@fit.vutbr.cz)

Kristian Kadlubiak (ikadlubiak@fit.vutbr.cz)

### 1 ÚVOD

Cílem tohoto cvičení je vyzkoušet si měření výkonosti sekvenčního kódu pomocí nástroje, Intel Advisor XE<sup>1</sup>. Vaším úkolem je vyzkoušet si práci se zmíněnými nástroji dostupnými v laboratořích.

### 2 INTEL ADVISOR

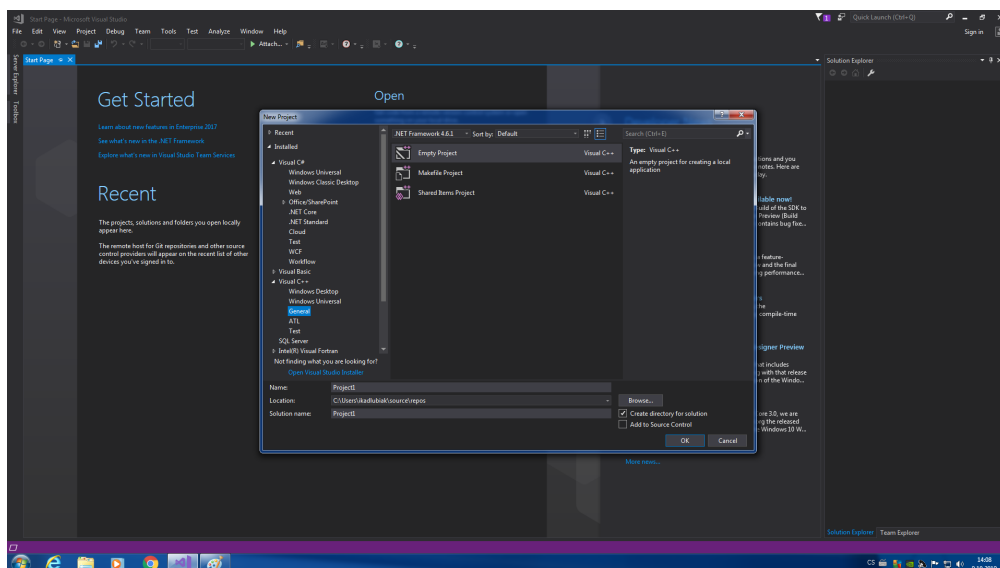
Intel Advisor je vizuální nástroj který dokáže přehledně profilovat kód a je dostupný ve Windows v prostředí Visual Studio. V rámci našeho předmětu nás zajímá část zabývající se vektorizací.

---

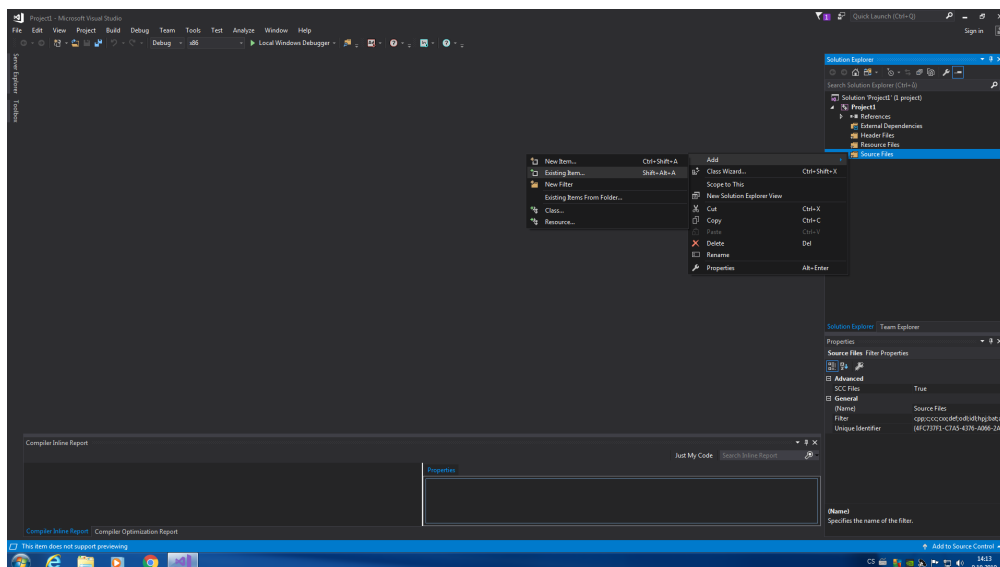
<sup>1</sup><https://docs.it4i.cz/software/intel/intel-suite/intel-advisor/>

## 2.1 NASTAVENÍ

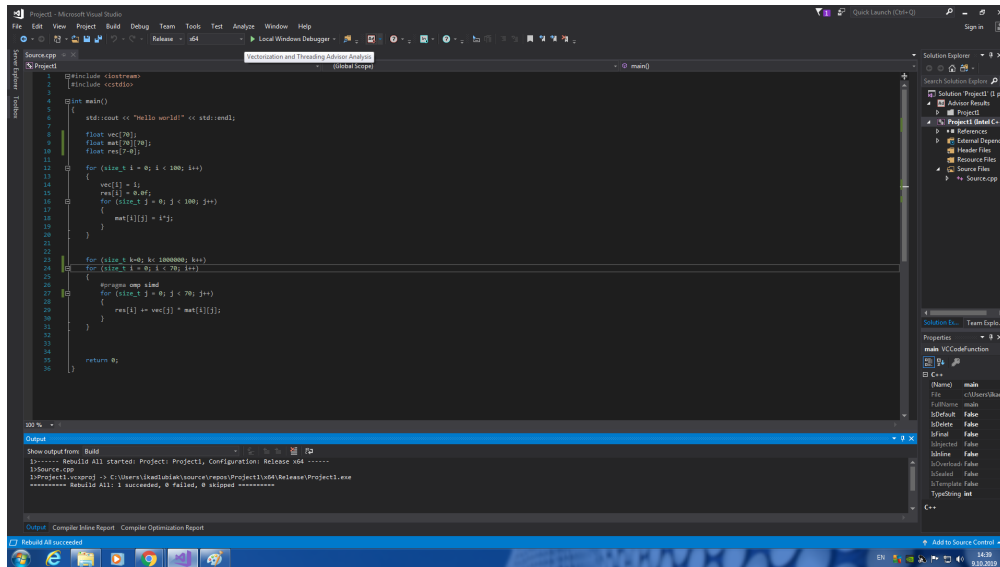
Spust'te prostředí Visual Studio a vytvořte nový prázdný C++ projekt.



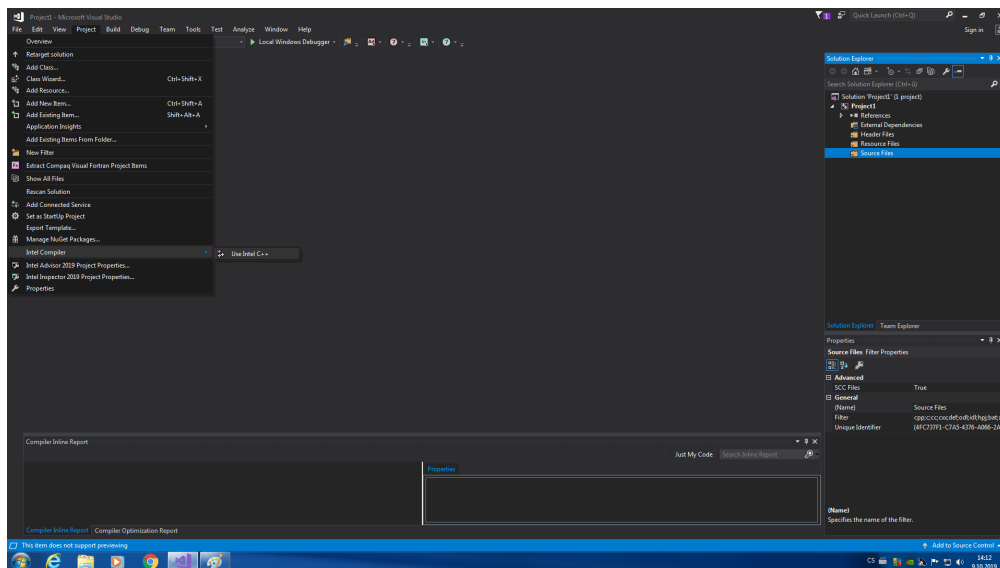
Nyní přidejte do projektu připravený soubor **Source.cpp** pravým klikem na složku Source Files v průzkumníku.



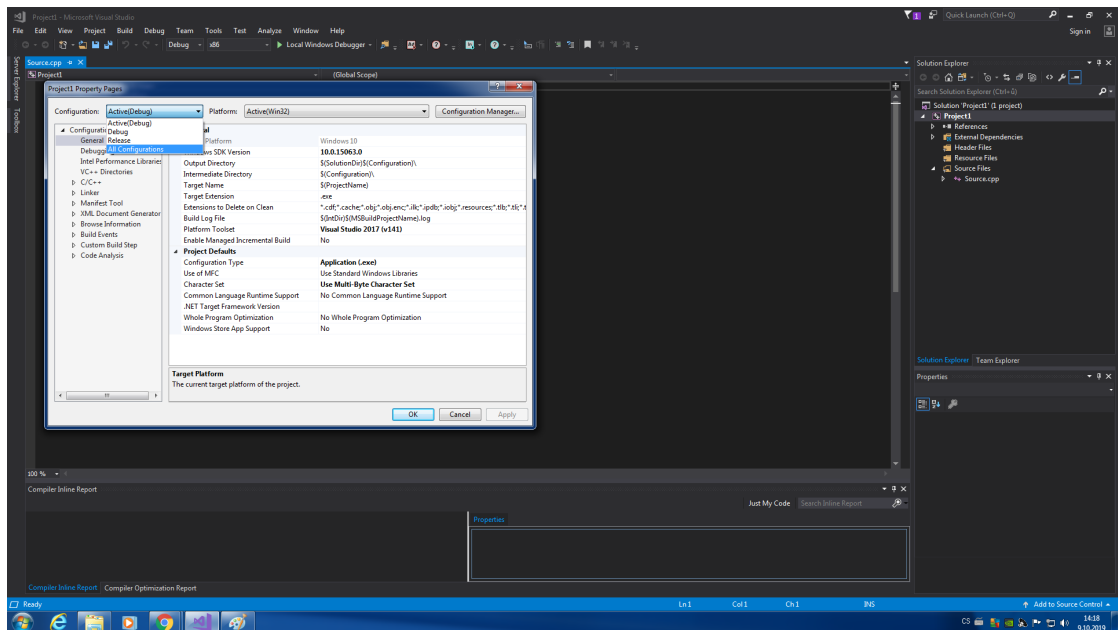
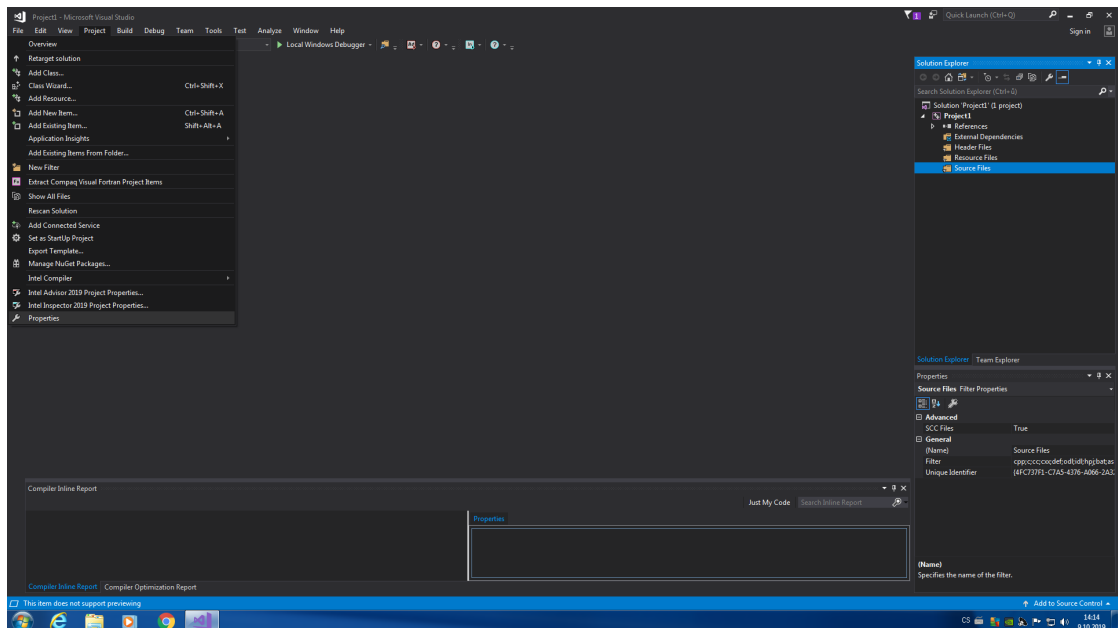
Následně se ujistěte že je správně nastaven mód a cíl kompilace na Release pro x64.



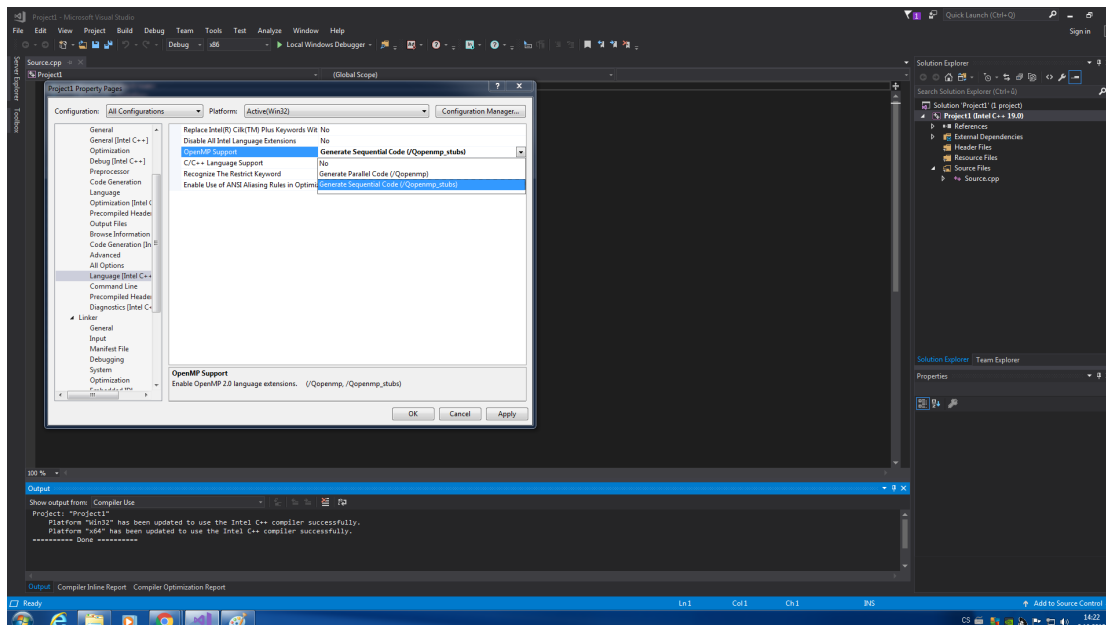
Je nutné explicitně přepnout kompilátor na Intel Visual C++.



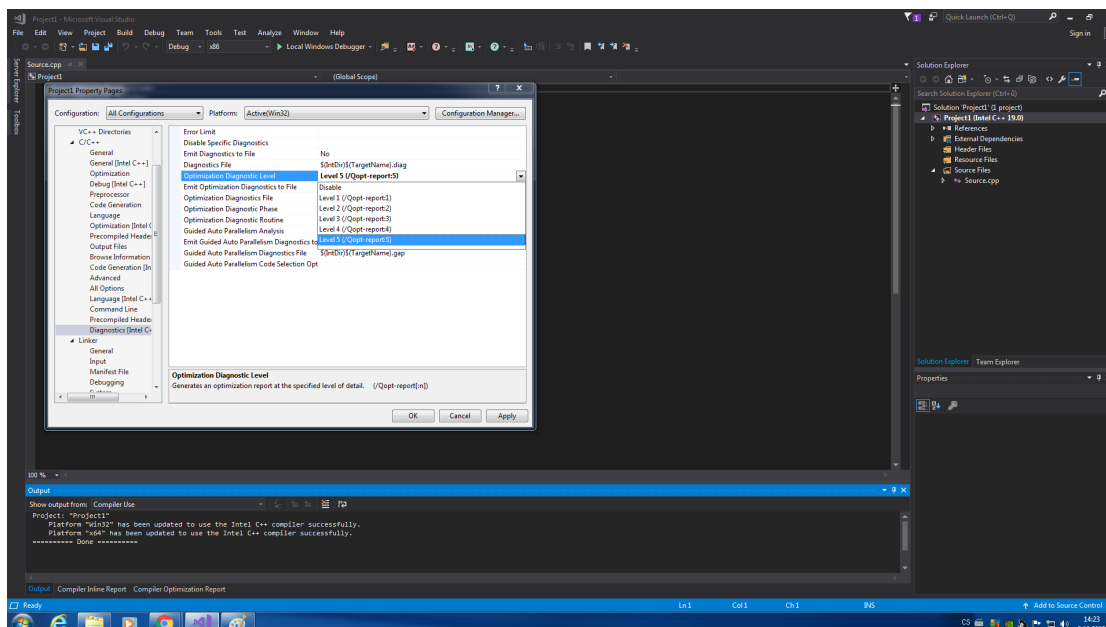
Ted' je ještě nutné upravit několik nastavení projektu. Ujistěte se, že změny jsou aplikovány na správnou konfiguraci a platformu.



Aby bylo možné kód vektorizovat je nezbytně nutné zapnout podporu pro OpenMP. Pro naše účely stačí sekvenční verze.



Intel advisor je schopen vyprodukovat vektorizační report, který je skvělým zdrojem informací při ladení výkonnosti kódu. Nastavíme úroveň diagnostiky na Level 5.



Visual Studio automaticky vkládá diagnostická hlášení přímo do kódu. Pokud znovu zkompilujete aplikaci měli by se nad smyčkami objevit hlášení o vektorizaci.

## 2.2 SURVEY TARGET

Základní profilovací data lze získat v nástroji Intel Advisor, který spustíte pomocí ikony v menu Visual Studio. Dále pomocí tlačítka *collect* v sekci *survey target*. V této sekci jsou zobrazeny jednotlivé smyčky uvnitř programu a doba jejich vykonávání. Taktéž je zde zobrazeno, zda byly smyčky vektorizovány, a pokud ne, proč.

The screenshot shows the Intel Advisor interface within Visual Studio. The 'Survey Target' section is active, displaying a list of loops and functions. The 'Vector Issues' column shows reasons for non-vectorization, such as 'function call cannot be vectorized' or 'inner loop was already executed'. The 'Selected (Total Time)' at the bottom indicates a total time of 0.191s.

Function Call Sites and Loops	Vector Issues	Self Time	Total Time	Type	Why No Vectorization?	Vectorized Loops	Efficiency	Gain	VL (V...)	Traits
loop in main at lab1.cpp:215		1.579s	1.579s	Vectorized (Body)		Avx	~90%	7.19x	8	Float32...
loop in main at lab1.cpp:193		1.540s	1.540s	Vectorized (Body)		Avx	~90%	7.19x	8	Float32...
func0x400b80		0.440s	1.249s	Scalar	function call cannot be...					Type Conversions
loop in main at lab1.cpp:212		0.020s	1.599s	Scalar	inner loop was already...					Extracts; Inserts; Shuffles
loop in main at lab1.cpp:195		0.010s	1.550s	Scalar	inner loop was already...					Extracts; Inserts; Shuffles
start		0.000s	4.438s	Function						
loop in main at lab1.cpp:44		0.000s	0.020s	Scalar	function call cannot be...					Extracts; Inserts; Shuffle...
main		0.000s	4.438s	Function						Type Conversions
loop in main at lab1.cpp:80		0.000s	1.249s	Scalar	outer loop was not au...					Float32...
loop in main at lab1.cpp:185		0.000s	1.560s	Scalar	inner loop was already...					Float32...
loop in main at lab1.cpp:210		0.000s	1.599s	Scalar	inner loop was already...					

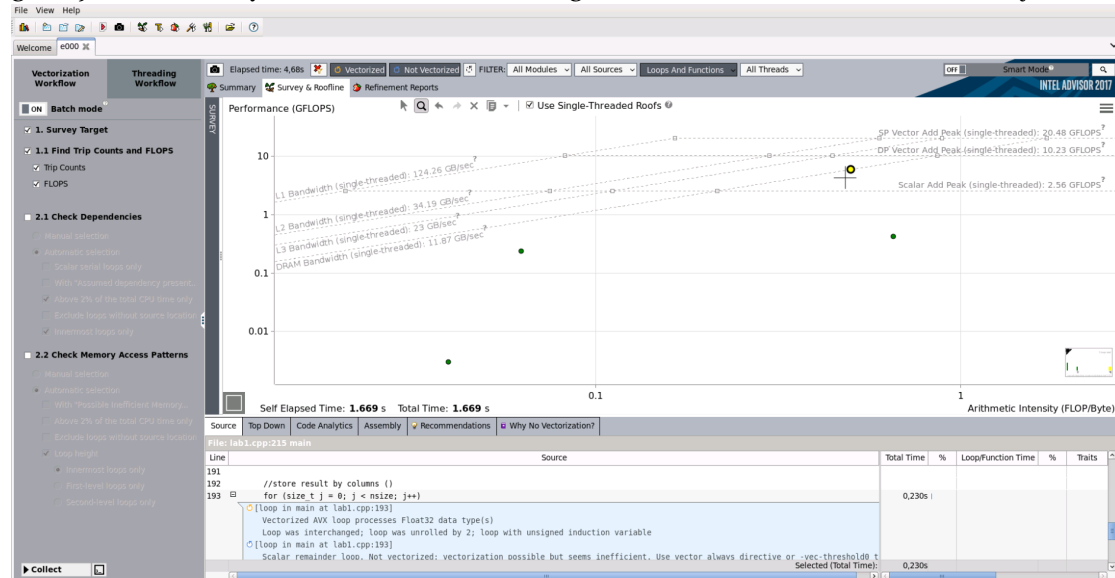
Source: lab1.cpp:215 main

Line 206 #ifdef PAPI  
207 papi\_exampined\_routines["loop\_6: matvec row"].Start();  
209 #endif  
210 for (size\_t i = 0; i < nrpt; i++) {  
211 //  
212 for (size\_t j = 0; j < nsize; j++)  
213 {  
214 a[j] = (float) 0.0f;  
215 for (size\_t k = 0; k < nsize; k++)  
216 {  
217 loop in main at lab1.cpp:215  
218 Vectorized AVX loop processes Float32 data type(s)  
219 Loop was unrolled by 2; loop with unsigned induction variable  
220 Scalar peeled loop [not executed]  
221 Loop with unsigned induction variable  
222 loop in main at lab1.cpp:215  
223 Scalar remainder loop [not executed]. Not vectorized: vectorization possible but seems inefficient. Use vector always directive or -v  
224 }  
225 }  
226 }  
227 }

Selected (Total Time): 0.191s

## 2.3 ROOFLINE

Roofline graf zobrazuje maximální možnou propustnost paměti a aritmetických jednotek. Intel Advisor je schopen zjistit potenciál využívaného procesoru a následně vykreslit do tohoto grafu jednotlivé kusy kódu. Pro získání tohoto grafu stiskněte *collect* v sekci *Run Roofline*.



## 2.4 VYZKOUŠEJTE

Zkuste změnit vektorovou sadu (SSE/AVX/AVX2) a zjistěte jak změna velikosti dat ovlivňuje výkonnost. Následně zkuste zaměnit operace ve výpočtu ze sčítání-násobení na sčítání-sčítání a násobení-násobení. Jaký vliv má tato změna na výkon?

