

**KIV/PPR**

**Semestrální Práce**

Jméno a přijmení: Martin Forejt

Osobní číslo: A20N0079P

Datum: 28. 11. 2021

# Zadání

Program semestrální práce dostane, jako jeden z parametrů, zadaný souboru, přístupný pouze pro čtení. Bude ho interpretovat jako čísla v plovoucí čárce - 64-bitový double. Program najde číslo na arbitrárně zadaném percentilu, další z parametrů, a vypíše první a poslední pozici v souboru, na které se toto číslo nachází.

Program se bude spouštět následovně:

pprsolver.exe soubor percentil procesor

* soubor - cesta k souboru, může být relativní k program.exe, ale i absolutní
* percentil - číslo 1 – 100
* procesor - řetězec určujíící, na jakém procesoru a jak výpočet proběhne
  + single - jednovláknový výpočet na CPU
  + SMP - vícevláknový výpočet na CPU
  + anebo název OpenCL zařízení - pozor, v systému může být několik OpenCL platforem
* Součástí programu bude watchdog vlákno, které bude hlídat správnou funkci programu

Další požadavky:

* Součástí programu bude watchdog vlákno, které bude hlídat správnou funkci pro-gramu.
* Testovaný soubor bude velký několik GB, ale paměť bude omezená na 250 MB. Tozařídí validátor.
* Program musí skončit do 15 minut na iCore7 Skylake.
* Program nebude mít povoleno vytvářet soubory na disku.
* Jako čísla budete uvažovat pouze ty 8-bytové sekvence, pro které std::fpclassify vrátí FP\_NORMAL nebo FP\_ZERO. Jiné sekvence budete ignorovat.
* Všechny nuly jsou si stejně rovné.
* Pozice v souboru vypisujte v bytech, tj. indexováno od nuly.
* Nalezené číslo číslo vypište v hexadecimálním formátu, např. pomocí std::hexfloat

# Úvod

Cílem této práce je vytvoření programu, který nalezne hodnotu zadaného percentilu v souboru, který bude interpretovat jako sekvenci 64 bitových čísel v plovoucí čárce (double) a dále nalezne první a poslední výskyt této hodnoty v daném souboru. Prostředí pro běh programu má omezenou paměť na 250 MB a běh výpočtu nesmí běžet déle něž 15 minut. Algoritmus bude implementován třetmi různými způsoby: sériově, paralelně s použitím SMP a paralelně s použitím knihovny OpenCL.

# Algoritmus

Překladač je napsán v programovacím jazyce C++, pro automatizaci překladu je použit CMake.

# Implementace

LLVM podporuje generování strojového kódu pro různé platformy. Seznam podporovaných platforem je možné vypsat příkazem:

# Paralelizace

Složitější rozšíření (2 body):

# Výsledky

Veškeré zdrojové kódy se nachází v adresáři *mlang/src*. Ten obsahuje kromě *.cpp*/*.h* souborů ještě tyto soubory:

# Uživatelská dokumentace

Podařilo se navrhnout gramatiku jazyka a vytvořit překladač pomocí nástrojů flex, bison a LLVM.

# Závěr

sdfasdf