UDC 004.451

E.V. Lytovskiy[1]
O.M. Rysovanyi[2]

STUDY OF MULTI-THREAD HASHING USING INSTRUCTIONS AVX512

[1] Federal Gymnasium and Federal High School Graz, Austria
[2] National Technical University "Kharkiv Polytechnic Institute"

Анотація

Ця робота присвячена дослідженню багатопотокового хешування із застосуванням найсучаснішого набору команд AVX512. Застосування найбільшої розрядності регістрів процесора та їх кількості дозволяє розбивати потік даних на блоки та суттєво зменшити час отримання хеш. При вдосконаленні процесу хешування виконані вимоги дотримання якісних показників оцінювання алгоритмів – часу обробки, кількості колізій та відповідності теоретичних значень хеш з отриманими. Розроблено програму, яка підтверджує відповідність отриманих даних з теоретичними.

Ключові слова: *хеш, masm64, AVX512, багатопотокове перетворення, регістри* zmm0–zmm31.

Abstract

This work is devoted to the study of multi-threaded hashing using the most modern AVX512 instruction set. The use of the largest bit size of the processor registers and their number allows you to divide the data stream into blocks and significantly reduce the time to obtain a hash. When improving the hashing process, the requirements for observing the qualitative indicators of algorithm evaluation - processing time, number of collisions and correspondence of theoretical hash values with the obtained ones - were met. A program was developed that confirms the correspondence of the obtained data with the theoretical ones.

Key words: *hash, masm64, AVX512, multithreaded conversion, registers zmm0–zmm31.*

Introduction

Multi-threaded hashing using AVX-512 instructions is a modern, efficient optimization for data processing performance [1, 2]. This approach combines two different levels of parallelism:

– Data (SIMD/AVX-512): within a single core, the processor processes multiple independent blocks of data in a single cycle (vectorization);

– Tasks (multithreading): multiple processor cores operate simultaneously, each executing its own vectorized task.

Conventional processors process 32-bit or 64-bit words one at a time. AVX-512 provides 512-bit registers (zmm0–zmm31). This enables the single instruction, multiple data (SIMD) concept. For example, when calculating a SHA-256 hash (which operates on 32-bit words), 16 different data streams (512 / 32 = 16) can be packed into a single 512-bit register. The processor performs logical operations (AND, XOR, OR, shifts) on 16 different data blocks simultaneously. This is called inter-block parallelism.

Research results

Research has shown that not all hashing algorithms scale equally well with AVX-512 [2]. The most suitable options for multi-threaded hashing are SHA-256, MD5, and SHA-512. These algorithms have a fixed word size (32 or 64 bits) and consist of simple logical operations. The SHA-NI (hardware-accelerated SHA) version of the algorithm is typically faster for a single thread, but AVX-512 instructions are naturally superior when hashing thousands of small packets simultaneously.

Besides general hashing algorithms, the specialized BLAKE3 algorithm was originally designed for SIMD. It uses a Merkle tree structure, allowing parallel hashing of even a single large file on AVX-512. A hash tree (Merkle tree) is a complete binary tree whose leaf nodes contain hashes of data blocks, and whose internal nodes contain hashes of the sum of the values in the child nodes.

The paper demonstrates that parallelization can be applied to the Bcrypt and Argon2 algorithms in certain modes. These algorithms, designed to protect against ASICs/GPUs with strong data dependencies, are vectorized using additional correction transforms.

The conducted research on hashing algorithms revealed that, in order to utilize parallelization, the application architecture must be divided into two layers:

– Level 1. Distribution across cores (OS threads).

In this case, it is necessary to divide the overall workload (e.g., a million files or a gigantic stream of network packets) between threads (one thread per physical core/logical processor);

– Level 2. Intra-thread vectorization (AVX-512).

Each core takes from the queue not one element at a time, but in blocks of 8 or 16 elements (depending on the algorithm's word size).

AVX-512 instructions require data in memory to be aligned on a 64-byte boundary. Loading unaligned data is slower, while loading aligned data is extremely fast. Example of a processing block:

```
    vmovdqa64 zmm1, zmmword ptr [crc_constants]
align_loop:
    vmovdqa64 zmm0, zmmword ptr [rsi] ;
    ; Parallel multiplication without carry (polynomial addition)
    vpclmulqdq zmm2, zmm0, zmm1, 00h
    vpclmulqdq zmm3, zmm0, zmm1, 11h
    vpxord zmm0, zmm2, zmm3          ; XOR
```

When processing the remainder of a block division, the classic (scalar) method is used.

When using AVX512 instructions, the bottleneck may be disk or network speed. If you're reading files from an old HDD, multi-threaded AVX-512 won't provide any performance gains—the processor will be waiting for the data.

Conclusions

The conducted research resulted in practical recommendations for the implementation of multi-threaded hashing using AVX512 instructions. Splitting the hash into additional blocks using algorithms designed for parallel hashing significantly reduces hash generation time. Furthermore, doubling the number of registers allows the algorithm to compete with disk speeds. Applying parallelization to sequential algorithms requires additional mathematical operations, which are necessary to obtain identical results.

References

1. Рисований О.М. Реверсне програмування. Захист коду. Середовище програмування masm64: навчально-методичний посібник для студентів спеціальності 123 "Комп'ютерна інженерія" всіх форм навчання [електронне видання] / О.М. Рисований. Харків: НТУ "ХПІ". 2024. 214 с. https://repository.kpi.kharkov.ua/handle/KhPI-Press/79627.
2. Рисований О.М. Реверсне програмування. Хеш-функції та їх використання. Середовище програмування masm64: навчальний посібник для студентів спеціальності F7 "Комп'ютерна інженерія" всіх форм навчання [електронне видання] / О.М. Рисований. – Харків: НТУ «ХПІ», 2026. – 342 с. https://repository.kpi.kharkov.ua/handle/KhPI-Press.

Літовський Євгеній Вітальович – федеральна гімназія та федеральна середня школа Граца, Австрія, учень 8 класу, yevgenii.lytovskyi@gmail.com

Рисований Олександр Миколайович – к.т.н., професор, Національний технічний університет «Харківський політехнічний інститут», кафедра комп’ютерної інженерії та програмування, rysov81524@gmail.com

Науковий керівник – ***Рисований Олександр Миколайович*** – к.т.н., професор, Національний технічний університет «Харківський політехнічний інститут», кафедра комп’ютерної інженерії та програмування, rysov81524@gmail.com

Lytovsryi Yevgenii V. – The Federal Gymnasium and Federal Secondary School Graz, 8th grade student, yevgenii.lytovskyi@gmail.com

Rysovanyi Oleksandr M. – Candidate of Technical Sciences, National Technical University "Kharkiv Polytechnic Institute", Department of Computer Engineering and Programming, rysov81524@gmail.com

Supervisor: ***Rysovanyi Oleksandr M.*** – Candidate of Technical Sciences, National Technical University "Kharkiv Polytechnic Institute", Department of Computer Engineering and Programming, rysov81524@gmail.com.