The Art of Hashing Algorithms

in Cryptography

Sulaimani Polytechnic University
Computer Networks Department

Asst. Prof. Dr. Sarkar H. Ahmed
Sarkar.ahmed@spu.edu.iq
2024

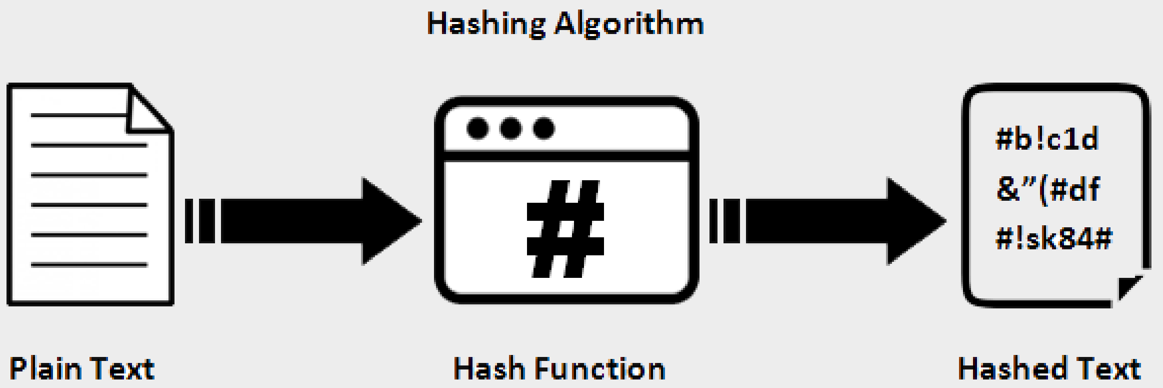
Agenda

- What is Hashing Algorithms
- Key points about hashing algorithms
- Types of hashing Algorithms
- Applications of Sha256
- The way of working Sha256
- The way of working Sha256

What is Hashing Algorithms

- ✓ Hashing algorithms are cryptographic functions that take an input (or 'message') and produce a fixed-size string of bytes.
- ✓ The output, known as a hash value or hash code, is unique to the input data.
- ✓ The hash value typically represented as a hexadecimal number.
- ✓ Input: **This is an example**

  Output: **47FB563CC8F86DC37C86D08BC542968F7986ACD81C97BF76DB7AD744407FE117**



Key points about hashing algorithms

- **Deterministic:** For a given input, a hashing algorithm will always produce the same hash value. This property is crucial for data **integrity** and **verification** purposes.

- **Fixed Output Size:** Regardless of the input size, hashing algorithms produce a fixed-size output. For example, the SHA-256 algorithm always produces a **256-bit hash value**.

- **One-Way Function:** Hash functions are designed to be one-way, meaning it's computationally infeasible to reverse the process and obtain the original input from the hash value. This property is essential for password hashing and digital signatures.

- **Collision Resistance:** A good hashing algorithm should minimize the likelihood of producing the same hash value for different inputs. This property helps maintain the **integrity** and **security** of hashed data.

- **Avalanche Effect:** A small change in the input data should result in a significantly different output hash value. This ensures that even minor modifications to the input will produce drastically different hashes.

Input: In to am attended desirous raptures declared diverted confined at. Collected instantly remaining up certainly to necessary as. Over walk dull into son boy door went new. At or happiness commanded daughters as. Is handsome an declared at received in extended vicinity subjects. Into miss on he over been late pain an. **Only** week bore boy what case left use. Match round scale now style far times. Your me past an much.

Output:
c44dec3110706d7d0edcc8686b9c3ece40f2e18ebfa28c7d061c79d9415ca252

Input: In to am attended desirous raptures declared diverted confined at. Collected instantly remaining up certainly to necessary as. Over walk dull into son boy door went new. At or happiness commanded daughters as. Is handsome an declared at received in extended vicinity subjects. Into miss on he over been late pain an. **only** week bore boy what case left use. Match round scale now style far times. Your me past an much.

Output:
a2aef227bd17c78fae8af28a2bd94d96f36382bf4bbc958ed8935bb9a90c3ba4

Types of hashing Algorithms

1. ✗ **MD5 (Message Digest Algorithm 5):**
   - Description: MD5 produces a 128-bit (16-byte) hash value. Despite being widely used in the past, it is now considered insecure due to vulnerabilities.
   - Output Length: 128 bits, 32 hexadecimal characters
2. ✗ **SHA-1 (Secure Hash Algorithm 1):**
   - Description: SHA-1 produces a 160-bit (20-byte) hash value. Like MD5, it is also considered insecure due to vulnerabilities.
   - Output Length: 160 bits, 40 hexadecimal characters
3. ✓ **SHA-256 (Secure Hash Algorithm 256-bit):**
   - Description: SHA-256 produces a 256-bit (32-byte) hash value. It is widely used and considered secure for various cryptographic applications.
   - Output Length: 256 bits, 64 hexadecimal characters
4. ✓ **SHA-384 (Secure Hash Algorithm 384-bit):**
   - Description: SHA-384 produces a 384-bit (48-byte) hash value. It offers higher security than SHA-256 but with a larger output size.
   - Output Length: 384 bits, 96 hexadecimal characters
5. ✓ **SHA-512 (Secure Hash Algorithm 512-bit):**
   - Description: SHA-512 produces a 512-bit (64-byte) hash value. It provides even stronger security but with a larger output size compared to SHA-256.
   - Output Length: 512 bits, 128 hexadecimal characters
6. ✓ **SHA-3 (Secure Hash Algorithm 3):**
   - Description: SHA-3 produces hash values of variable length, with options for 224, 256, 384, or 512 bits. It is based on the Keccak algorithm and provides high security.
   - Output Length: Varies (e.g., SHA-3-256 produces a 256-bit hash value, SHA-3-512 produces a 512-bit hash value)

Applications of Sha256

1. **Data Integrity:** is commonly used to verify the integrity of data transmitted over a network or stored on a disk. By hashing the data before transmission or storage and comparing the hash value at the receiving end, one can ensure that the data hasn't been altered or corrupted during transit.

2. **Digital Signatures:** is a critical component in digital signature algorithms. It's used to hash the message before signing, providing a unique identifier for the message. This allows recipients to verify both the **integrity** and **authenticity** of the message.

3. **Blockchain Technology (Bitcoin):** the hashing algorithm used in Bitcoin and many other cryptocurrencies. In blockchain technology, it's used to create the cryptographic hash of a block's header, which is essential for **mining** and ensuring the security and immutability of the blockchain.

4. **Password Storage:** When storing passwords, it's crucial to hash them securely to prevent exposure in case of a data breach. SHA-256 (though not ideal on its own for password hashing due to its speed) is often used as part of a more robust password hashing scheme.

5. **File Integrity Checking:** System administrators and developers often use SHA-256 to verify the integrity of files and software distributions. Users can download a file along with its corresponding SHA-256 hash value from a trusted source. After downloading, they can compute the hash of the downloaded file and compare it with the provided hash to ensure the file hasn't been tampered with.

6. **SSL/TLS Certificates:** is used in SSL/TLS certificates for digital signatures. As SHA-1 has been deprecated due to vulnerabilities, SHA-256 is among the recommended hash functions for generating secure SSL/TLS certificates.

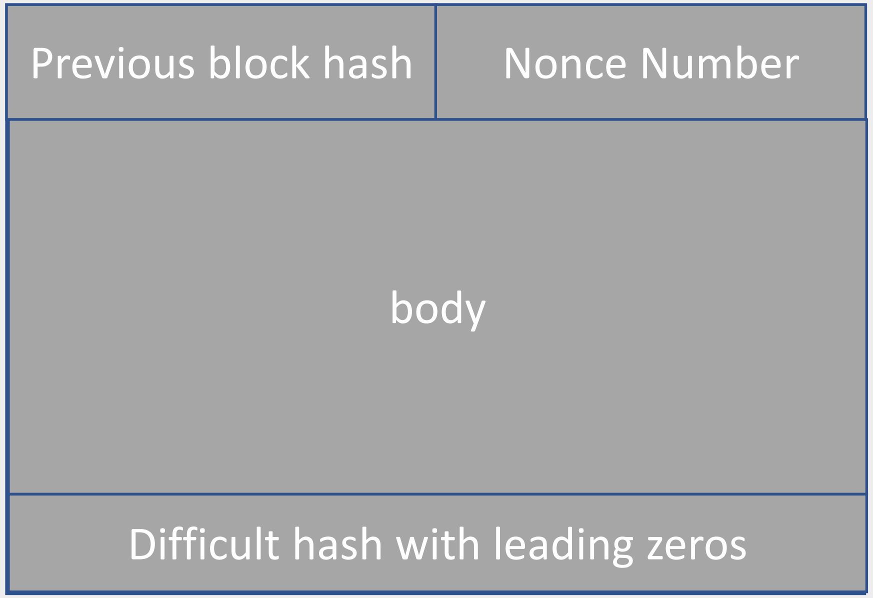
7. **Cryptographic Applications:** is used in various cryptographic protocols and applications, including secure sockets layer (SSL), transport layer security (TLS), and many others where data integrity and authenticity are crucial.

Unsolvable Problem is Worth Billions of Dollars

- **One-Way Function: Sha256** is designed to be one-way, meaning it's computationally infeasible to reverse the process and obtain the original input from the hash value.
- **In Bitcoin**, SHA-256 plays a critical role in several aspects of the protocol, primarily related to **mining** and the **creation of secure, immutable blocks** in the blockchain.
- A Bitcon block consists of (previous hash, nonce (a random number), block body) => very small hash value
- Set Difficulty: Very small hash value means for example leading the hash output with 10 zeros out of 64 characters. (One leading zero= %50, two zeros %25…)
- Change the nonce number and try hash the block to see if you

Be able to get a difficult hash? If not, then try changing the nonce

Again and so on..



Previous block hash

Nonce Number

body

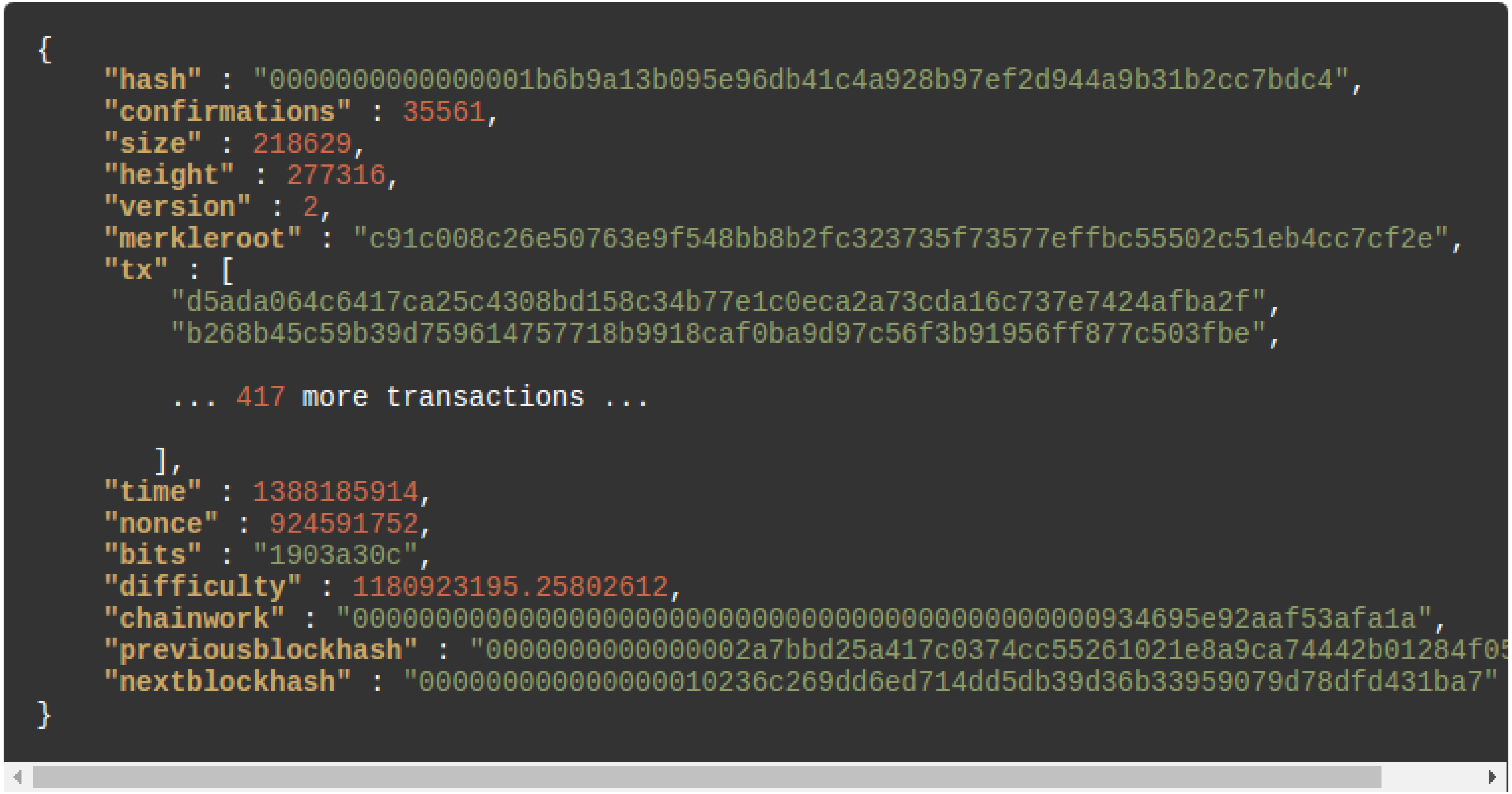
Difficult hash with leading zeros

If you be able to reverse the following hash to plain text (0000000000000000f3a6382849c234c677b886555d5678f)

Then, you can mine as much Bitcon as you like,

where the price of each one is (74,700,000 IQD)

A Real Bitcoin Block Data

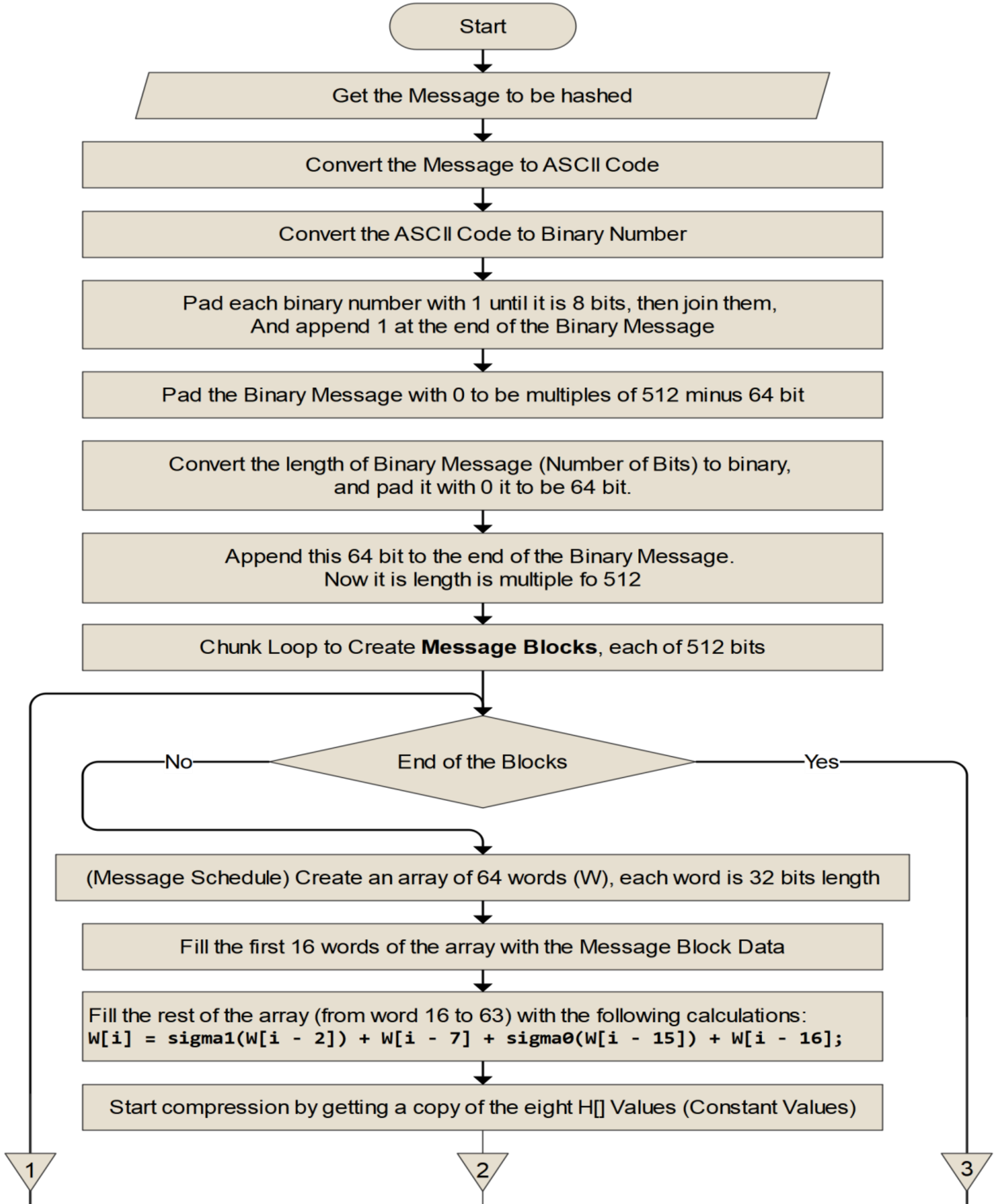


```
{
    "hash" : "0000000000000001b6b9a13b095e96db41c4a928b97ef2d944a9b31b2cc7bdc4",
    "confirmations" : 35561,
    "size" : 218629,
    "height" : 277316,
    "version" : 2,
    "merkleroot" : "c91c008c26e50763e9f548bb8b2fc323735f73577effbc55502c51eb4cc7cf2e",
    "tx" : [
        "d5ada064c6417ca25c4308bd158c34b77e1c0eca2a73cda16c737e7424afba2f",
        "b268b45c59b39d759614757718b9918caf0ba9d97c56f3b91956ff877c503fbe",

        ... 417 more transactions ...

       ],
    "time" : 1388185914,
    "nonce" : 924591752,
    "bits" : "1903a30c",
    "difficulty" : 1180923195.25802612,
    "chainwork" : "000000000000000000000000000000000000000000000934695e92aaf53afa1a",
    "previousblockhash" : "0000000000000002a7bbd25a417c0374cc55261021e8a9ca74442b01284f05
    "nextblockhash" : "00000000000000010236c269dd6ed714dd5db39d36b33959079d78dfd431ba7"
}
```

The way of working Sha256



Start

Get the Message to be hashed

Convert the Message to ASCII Code

Convert the ASCII Code to Binary Number

Pad each binary number with 1 until it is 8 bits, then join them,
And append 1 at the end of the Binary Message

Pad the Binary Message with 0 to be multiples of 512 minus 64 bit

Convert the length of Binary Message (Number of Bits) to binary,
and pad it with 0 it to be 64 bit.

Append this 64 bit to the end of the Binary Message.
Now it is length is multiple fo 512

Chunk Loop to Create **Message Blocks**, each of 512 bits

End of the Blocks

No

Yes

(Message Schedule) Create an array of 64 words (W), each word is 32 bits length

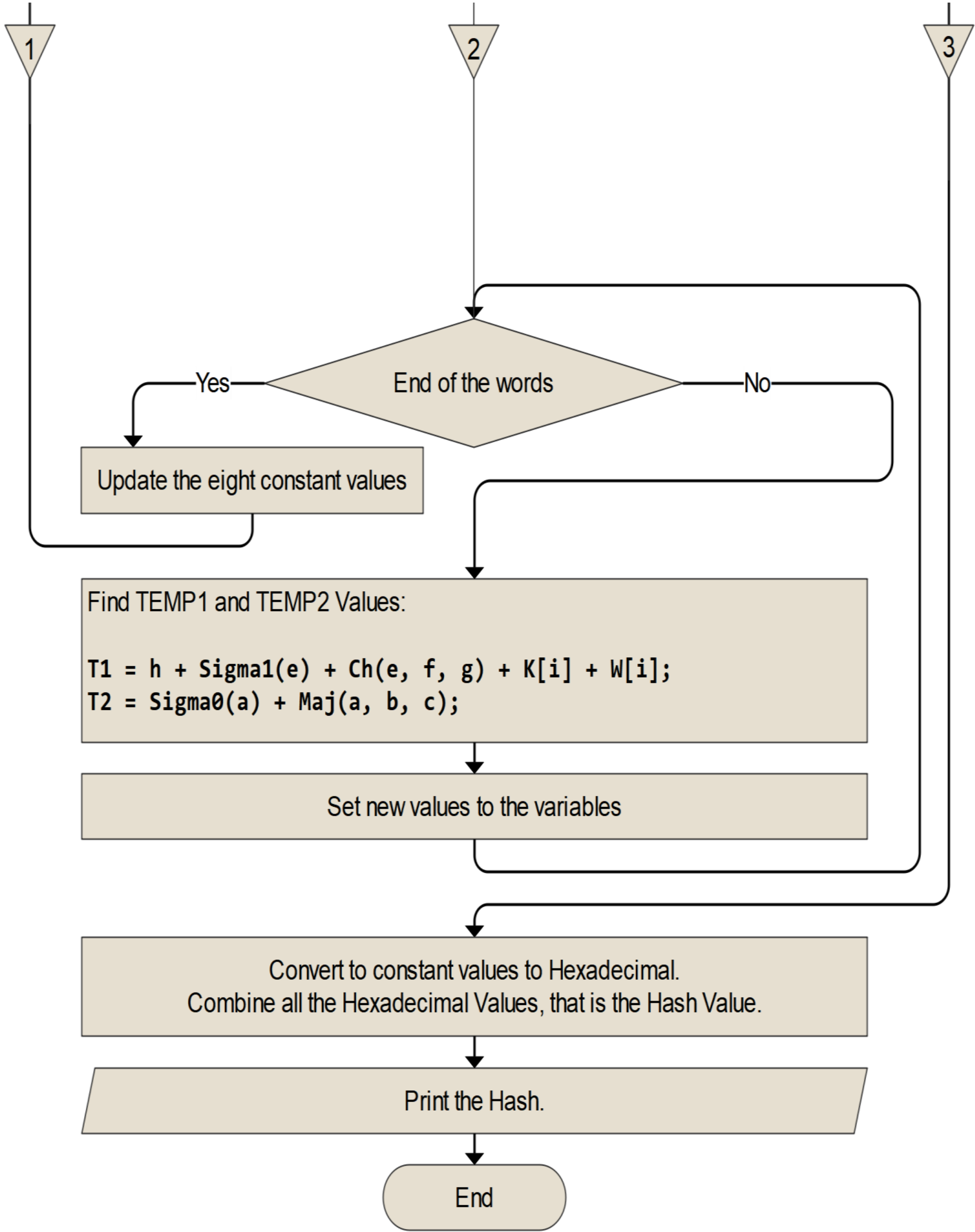
Fill the first 16 words of the array with the Message Block Data

Fill the rest of the array (from word 16 to 63) with the following calculations:

```
W[i] = sigma1(W[i - 2]) + W[i - 7] + sigma0(W[i - 15]) + W[i - 16];
```

Start compression by getting a copy of the eight H[] Values (Constant Values)

The way of working Sha256 (cont)


1
2
3
End of the words
Yes
No
Update the eight constant values
Find TEMP1 and TEMP2 Values:
T1 = h + Sigma1(e) + Ch(e, f, g) + K[i] + W[i];
T2 = Sigma0(a) + Maj(a, b, c);
Set new values to the variables
Convert to constant values to Hexadecimal.
Combine all the Hexadecimal Values, that is the Hash Value.
Print the Hash.
End

Simulation

Source code

زانکۆی پۆلیتەکنیکی سلێمانی

Sulaimani Polytechnic University

Thank You!