Bangladesh University of Engineering & Technology

CSE 208 – DSA II

Report – Hashing offline

Student No. 2205083

Sourav Sarker

Section B

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | **Hash1 Function** | | | | | **Hash2 Function** | | | | |
| # of Collisions during insertions | **Before deletion** | | **After deletion** | | # of Collisions during insertions | **Before deletion** | | **After deletion** | |
| Average search time | Average probes | Average search time | Average probes | Average search time | Average probes | Average search time | Average probes |
| **Separate chaining with balanced BST** | 711 | 153.3 | N/A | 162.2 | N/A | 732 | 155.8 | N/A | 157.3 | N/A |
| **Linear probing with step adjustment** | 1335 | 165.0 | 1.3 | 212.2 | 2.3 | 1370 | 165.9 | 1.3 | 196.5 | 2.4 |
| **Double Hashing** | 1113 | 265.2 | 1.3 | 289.0 | 1.9 | 1113 | 263.7 | 1.3 | 288.1 | 2.1 |

For load factor 0.4:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | **Hash1 Function** | | | | | **Hash2 Function** | | | | |
| # of Collisions during insertions | **Before deletion** | | **After deletion** | | # of Collisions during insertions | **Before deletion** | | **After deletion** | |
| Average search time | Average probes | Average search time | Average probes | Average search time | Average probes | Average search time | Average probes |
| **Separate chaining with balanced BST** | 1055 | 159.8 | N/A | 206.9 | N/A | 1066 | 171.1 | N/A | 154.2 | N/A |
| **Linear probing with step adjustment** | 2346 | 169.3 | 1.5 | 204.8 | 2.6 | 2488 | 171.8 | 1.6 | 211.1 | 2.9 |
| **Double Hashing** | 1961 | 268.1 | 1.4 | 290.9 | 2.1 | 1961 | 302.2 | 1.4 | 299.1 | 2.2 |

For load factor 0.5:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | **Hash1 Function** | | | | | **Hash2 Function** | | | | |
| # of Collisions during insertions | **Before deletion** | | **After deletion** | | # of Collisions during insertions | **Before deletion** | | **After deletion** | |
| Average search time | Average probes | Average search time | Average probes | Average search time | Average probes | Average search time | Average probes |
| **Separate chaining with balanced BST** | 1483 | 191.7 | N/A | 191.4 | N/A | 1461 | 172.9 | N/A | 191.1 | N/A |
| **Linear probing with step adjustment** | 4398 | 183.6 | 1.6 | 244.6 | 3.7 | 4404 | 179.4 | 1.7 | 333.3 | 3.3 |
| **Double Hashing** | 3164 | 278.2 | 1.6 | 317.8 | 2.6 | 3164 | 273.6 | 1.5 | 314.8 | 2.6 |

For load factor 0.6:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | **Hash1 Function** | | | | | **Hash2 Function** | | | | |
| # of Collisions during insertions | **Before deletion** | | **After deletion** | | # of Collisions during insertions | **Before deletion** | | **After deletion** | |
| Average search time | Average probes | Average search time | Average probes | Average search time | Average probes | Average search time | Average probes |
| **Separate chaining with balanced BST** | 1970 | 198.2 | N/A | 208.7 | N/A | 1951 | 169.9 | N/A | 171.2 | N/A |
| **Linear probing with step adjustment** | 7875 | 231.1 | 2.0 | 323.5 | 5.3 | 8452 | 219.4 | 2.3 | 310.4 | 5.5 |
| **Double Hashing** | 5277 | 490.5 | 1.9 | 357.9 | 3.0 | 5277 | 281.3 | 1.8 | 351.7 | 3.1 |

For load factor 0.7:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | **Hash1 Function** | | | | | **Hash2 Function** | | | | |
| # of Collisions during insertions | **Before deletion** | | **After deletion** | | # of Collisions during insertions | **Before deletion** | | **After deletion** | |
| Average search time | Average probes | Average search time | Average probes | Average search time | Average probes | Average search time | Average probes |
| **Separate chaining with balanced BST** | 2448 | 239.8 | N/A | 220.6 | N/A | 2441 | 178.0 | N/A | 299.7 | N/A |
| **Linear probing with step adjustment** | 16007 | 207.7 | 2.8 | 364.1 | 9.7 | 16310 | 220.2 | 3.3 | 348.6 | 9.0 |
| **Double Hashing** | 18097 | 287.4 | 2.0 | 336.5 | 4.0 | 18097 | 288.4 | 2.0 | 343.0 | 4.1 |

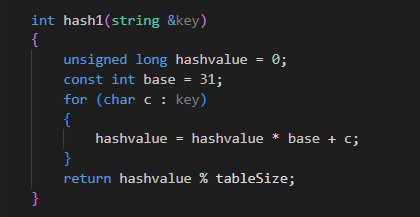
For load factor 0.8:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | **Hash1 Function** | | | | | **Hash2 Function** | | | | |
| # of Collisions during insertions | **Before deletion** | | **After deletion** | | # of Collisions during insertions | **Before deletion** | | **After deletion** | |
| Average search time | Average probes | Average search time | Average probes | Average search time | Average probes | Average search time | Average probes |
| **Separate chaining with balanced BST** | 3089 | 199.4 | N/A | 187.7 | N/A | 3054 | 198.7 | N/A | 180.4 | N/A |
| **Linear probing with step adjustment** | 38878 | 248.0 | 4.7 | 869.5 | 33.8 | 39508 | 236.2 | 4.1 | 804.7 | 31.5 |
| **Double Hashing** | 23650 | 304.1 | 2.3 | 388.5 | 6.4 | 23650 | 298.7 | 2.4 | 419.8 | 7.3 |

For load factor 0.9:

Hash Function Analysis and Load Factor Impact Report

Hash1: Polynomial Rolling Hash



**Description**: This is a polynomial rolling hash function that treats each string as a polynomial with base 31. Each character's ASCII value serves as a coefficient, and the hash is computed as: hash = c₁×31ⁿ⁻¹ + c₂×31ⁿ⁻² + c₃×31ⁿ⁻³ + ... + cₙ×31⁰ mod ***tableSize***

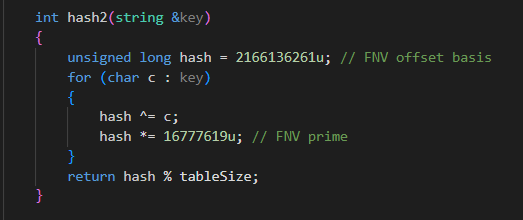
**Why chosen:** Base 31 is prime, which helps reduce collisions

Provides good distribution for string keys

Simple and efficient to compute

**Constants used**: Base = 31 (prime number chosen for good distribution properties)

Hash2: FNV-1a Hash



**Description**: This implements the FNV-1a hash algorithm, which uses XOR and multiplication operations. It starts with an offset basis and for each byte: XOR the byte with the hash, then multiply by the FNV prime.

**Why chosen:**

Different mathematical approach than Hash1, ensuring diverse hash distributions

Well-tested algorithm with good collision resistance

Fast computation with simple operations

**Constants used**: FNV offset basis = 2166136261, FNV prime = 16777619

**Load Factor Impact Analysis**

**Separate Chaining with Balanced BST**

**Observations**:

* **Collisions**: Increase linearly with load factor (711 → 3089 from 0.4 to 0.9)
* **Search Time**: Remains relatively stable (153-239 ns), showing good BST performance
* **Performance**: Most consistent across all load factors due to BST's O(log n) operations

**Analysis**: The balanced BST maintains good performance even at high load factors because tree operations scale logarithmically with chain length.

**Linear Probing with Step Adjustment**

**Observations**:

* **Collisions**: Dramatic increase at higher load factors (1335 → 38878 from 0.4 to 0.9)
* **Search Time**: Significant degradation after 0.7 load factor
* **Probes**: Exponential growth (1.3 → 33.8 average probes from 0.4 to 0.9)

**Analysis**: Linear probing suffers from **primary clustering** at high load factors. As the table fills up, contiguous blocks of occupied slots form, leading to long probe sequences.

**Double Hashing**

**Observations**:

* **Collisions**: Increases substantially but less than linear probing (1113 → 23650)
* **Search Time**: More stable than linear probing but increases with load factor
* **Probes**: Grows more gradually than linear probing (1.3 → 6.4 average probes)

**Analysis**: Double hashing performs better than linear probing because it uses a second hash function to determine step size, reducing clustering effects.

**Key Insights**

**Load Factor Thresholds**

* **0.4-0.6**: All methods perform reasonably well
* **0.7**: Performance starts degrading for open addressing methods
* **0.8-0.9**: Severe performance degradation for linear probing; double hashing still manageable

**Method Comparison**

1. **Separate Chaining**: Most predictable performance, scales well with load factor
2. **Double Hashing**: Good compromise between performance and complexity
3. **Linear Probing**: Fastest at low load factors but degrades quickly at high load factors

**Hash Function Performance**

Both hash functions show similar collision patterns, indicating good independent distribution. The slight differences in collision counts between Hash1 and Hash2 demonstrate that both functions provide adequate dispersion for the given dataset.

**Recommendations**

* **For high load factors (>0.7)**: Use separate chaining with balanced BST
* **For low load factors (<0.6)**: Linear probing offers good cache performance
* **For moderate load factors (0.6-0.8)**: Double hashing provides good balance
* **Critical threshold**: Avoid load factors above 0.8 for open addressing methods