高級言語による内部メモリ検索方法

比較検討　検討報告書

第1.0版

|  |  |  |
| --- | --- | --- |
| 所属 | 承認 | 作成 |
| システム技術部  ICTシステム2課 | 山口 | 山口 |

[1. 概要 1](#_Toc100316079)

[2. 検索方法 1](#_Toc100316080)

[2.1.　線形検索 2](#_Toc100316081)

[2.2　改良型線形検索 3](#_Toc100316082)

[2.3　2分探索 4](#_Toc100316083)

[2.4　ハッシュ検索 5](#_Toc100316084)

[3. 前提条件 6](#_Toc100316085)

[4. 試験方法 7](#_Toc100316086)

[5. 試験結果 8](#_Toc100316087)

[6. まとめ 9](#_Toc100316088)

[A) 付録 10](#_Toc100316089)

# 概要

内部メモリに展開したデータを検索する際、C言語であれば、検索データが1000件以下であれば線形検索における検索時間は、大きな差はないと言われていた。

近年は多数の高級言語があり、高級言語の場合にも同様のことがあるのかを確認した。

また、高級言語では簡易に複数の検索方法を扱うことができる。そのため、線形検索を含め、複数の検索方法毎にデータ群の数を変え、それぞれの検索時間の比較を行った。

2章に比較する検索方法の概要、3章に前提条件、4章に試験方法、5章に試験結果、6章にまとめを示す。

# 検索方法

今回、検索時間を測定する検索方法は以下の4つである。次項に各検索方法の説明を示す。

* 線形検索
* 改良型線形検索
* 2分探索
* ハッシュ検索

## 2.1.　線形検索

線形検索とは、配列などに格納されたデータ列の先頭から末尾まで順番に、検索対象データと一致するか比較していく検索方法である。

N個のデータ列の中から線形検索する場合、最良のケースは先頭の要素と一致する場合比較回数は1回、最悪のケースは末尾まで探してもデータが見つからなかった場合で比較はN回となる。平均の比較回数はN/2回となる。比較回数の平均値は検索対象数に正比例して増大する。

検索方法が簡易なため、少ないプログラムコードで記述することができる。

線形検索の動作例を図 2‑1に示す。

|  |  |
| --- | --- |
| 項番 | 検索データ |
| 0 | A |
| 1 | B |
| ： | ： |
| 123 | AABA |
| 124 | AABB |
| 125 | AABC |
| ： |  |
| N | ZZZ |

検索対象

データ：B

検索対象

データ：AABC

比較回数：2

比較回数：126

検索対象

データ：ZZZ

比較回数：N+1

図 2‑1　線形検索の動作例

## 2.2　改良型線形検索

改良型線形検索とは、配列などに格納されたデータ列の初回は、先頭から末尾まで順番に、2回目以降は、前回の比較結果の位置から開始し、末尾に到達した場合には戦闘に戻り、検索対象データと一致するか比較していく検索方法である。

線形検索と同様に、比較回数の平均値は検索対象数に正比例して増大する。

検索する回数が多い場合には、線形検索よりも比較回数が少なくなると考えられる。

改良型線形検索の動作例を図 2‑2に示す。

検索対象

データ：B

検索対象

データ：AABC

比較回数：2

比較回数：124

検索対象

データ：ZZZ

比較回数：

　　（N+1）－124

|  |  |
| --- | --- |
| 項番 | 検索データ |
| 0 | A |
| 1 | B |
| ： | ： |
| 123 | AABA |
| 124 | AABB |
| 125 | AABC |
| ： |  |
| N | ZZZ |

図 2‑2　改良型線形検索の動作例

## 2.3　2分探索

2分探索とは、検索データを昇順または降順に並び替えて格納し、以下の1〜3を繰り返し、検索対象データを比較する方法である。

1. 先頭のデータと末尾のデータの中間値を取得する。
2. 中間値と検索対象データと比較し、一致していれば検索終了
3. 中間値よりも検索対象データが大きい場合、1.で取得した中間値を先頭データに、小さい場合は1.で取得した中間値を末尾データに設定する。

2文探索は、検索対象データを大小比較するため、数値の検索に適しており、文字列の場合には、「a」は「0x41」などの文字コードでデータを昇順または、降順に並び替える必要がある。

なお、本検索方法は、検索対象データ数が2倍となっても、比較回数が1回増えるだけである。

2文探索の動作例を図 2‑3に示す。

|  |  |
| --- | --- |
| 項番 | 検索データ |
| 0 | 101 |
| 1 | 109 |
| 2 | 124 |
| 3 | 210 |
| 4 | 351 |
| 5 | 441 |
| 6 | 492 |
| 7 | 891 |

検索対象データ：492

①項番0と項番7の中間値である項番3の値と比較

②項番4と項番7の中間値である項番6の値と比較

③項番5と項番7の中間値である項番6の値と比較

図 2‑3　2文探索の動作例

## 2.4　ハッシュ検索

ハッシュ検索とは、ハッシュ関数と呼ばれる「一定の計算式」を用いて、データの格納位置を特定する検索方法である。

ハッシュ値の算出は他のデータの存在を考慮せず行われるため、複数の異なるデータから同じハッシュ値が算出される衝突（collision）が起きることがある。同じハッシュ値を持つデータ群をシノニム（synonym）という。

シノニムを処理する方法としては、空いているハッシュ値にデータを振り分けて衝突を解消する「オープンアドレス法」や、同じハッシュ値を持つデータを線形リストとして保存する「直接連鎖法」などがある。

比較数はデータ格納値と検索キーの比較のみで、最小で1回、最大で同一格納位置になったデータ数である。

ハッシュ検索の動作例を図 2‑4に示す。

|  |  |
| --- | --- |
| 検索キー | データ |
| 101 | A |
| 109 | B |
| 124 | CCC |
| 210 | AABA |
| 308 | AABB |
| 481 | AABC |
| 452 | YYYY |
| 841 | ZZZ |

|  |  |  |
| --- | --- | --- |
| ハッシュ値 | 検索キー | データ |
| 0 |  |  |
| 1 | 101 | A |
| ： | ： | ： |
| 8 | 308 | AABB |
| 9 | 109 | B |
| 10 | 210 | AABA |
| ： | ： | ： |
| 24 | 124 |  |
|  |  |  |
| 41 | 841 | ZZZ |
| ： | ： | ： |
| 52 | 452 | YYYY |
| ： | ： | ： |
| 81 | 481 | AABC |

元データ

ハッシュ値を用いた検索データ

一定の計算式を「100で割った余り」とする。

検索対象データ

検索キー：210

一定の計算式からハッシュ値（10）を算出し、当該領域からデータを取得し、検索キーと比較を行う。

図 2‑4　ハッシュ検索の動作例

# 前提条件

検索方法毎の検索時間測定の前提条件を以下に示す。

サーバ

|  |  |  |
| --- | --- | --- |
| OS | ： | OS debian 12 不安定版 |
| CPU | ： | Intel® Core™ i5-2410M CPU @ 2.30GHz × 4 |
| メモリ | ： | 4Gbyte |

言語

**大規模な開発に採用されることが多い、**Javaを使用する。

Javaのバージョンは「openjdk version "11.0.14"」とする。

データ

数値と文字列の対となっている郵政省提供の郵便番号と住所のデータとする。なお、日本郵便株式会社は本データの著作権を主張していない。そのため、自由に改修することが可能である。

# 試験方法

検索方法は2章で示した4つの検索方法とする。ただし、、Javaではハッシュ検索にHashMap、TreeMap、LinkedHashMap（※）の3の実装があるため、それぞれのハッシュ検索も併せて実施する。

検索データ、検索キー、検索対象データ、検索回数については以下に示す。

|  |  |
| --- | --- |
| 検索データ | 郵便番号と住所と対で管理する。また。データ数は、10、100、1000、10000の4パターンとする。 |
| 検索キー | 数値と文字列の2パターンの2パターンとする。  検索キーが文字列の場合、2分探索はソートが必要なため、対象外とする。 |
| 検索対象データ | 検索データとは別で配列で管理し、検索データ数以内で乱数を発生させ、当該乱数をインデックス値として検索対象データを取得する。そのため、検索データなしは発生しない。 |
| 検索回数 | 上記条件毎に10000回検索しを1セットとし、5回繰り返す。 |

※）HashMap、TreeMap、LinkedHashMapの違いを以下に示す。

|  |  |  |
| --- | --- | --- |
| HashMap | ： | キーからハッシュ値を算出して管理するため、順序は不定となる |
| TreeMap | ： | キーの自然順序付けによってソートされる |
| LinkedHashMap | ： | HashMapとLinkedListの両方で管理するため、挿入された順番を保持する |

# 試験結果

Javaにおいてもデータ数が1000件以内であれば、線形検索の時間に大きな差は発生しなかった。

また、検索時間が一番短かったのは、ハッシュ検索であった。Javaのハッシュ値で検索を行う連想配列のクラス（HashMap、TreeMap、LinkedHashMap）では検索時間に大きな差はなかった。

検索キーが数値の場合の結果を表 5‑1に、検索キーが文字列の場合の結果を表 5‑2に示す。なお、表中の数値は10000回検索の5回の平均である。全ての試験結果はA）付録を参照のこと。

表 5‑1　検索キーが数値の場合の10000件の検索時間

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 検索方法 | 検索データ数毎の10000件検索の合計時間（msec） | | | |
| 10件 | 100件 | 1000件 | 10000件 |
| 線形検索 | 11 | 50 | 58 | 270 |
| 改良型線形検索 | 8 | 43 | 45 | 240 |
| 2分探索 | 5 | 12 | 12 | 12 |
| HashMap | 5 | 5 | 4 | 7 |
| TreeMap | 4 | 4 | 4 | 8 |
| LinkedHashMap | 3 | 4 | 3 | 7 |

表 5‑2　検索キーが文字列の場合の10000件あたりの検索時間

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 検索方法 | 検索データ数毎の10000件検索の合計時間（msec） | | | |
| 10件 | 10件 | 10件 | 10件 |
| 線形検索 | 15 | 75 | 76 | 954 |
| 改良型線形検索 | 9 | 72 | 65 | 1146 |
| HashMap | 3 | 3 | 4 | 4 |
| TreeMap | 3 | 3 | 3 | 5 |
| LinkedHashMap | 3 | 3 | 3 | 4 |

# まとめ

線形検索等の6つの方法で、検索データ数毎に10000回の検索を行い、1000回検索にかかった時間の比較を行った。

Javaにおいてもデータ数が1000件以内であれば、線形検索の時間に大きな差は発生しなかった。

検索とは、検索データと検索対象データを比較することである。この比較を行う回数が少ないほど、検索時間が短くなる。そのため、検索データの格納位置を取得するのに、比較を行わず、一定の計算式で算出するハッシュ検索が最も早くデータ検索を行うことができた。

入力順など、順序性を保持しなければならない場合には、以下の2通りの方法がある。

* 性能を求めない場合は、ソースコードが簡易な線形検索で検索を行う。
* 性能が求められる場合は、内部データは配列で管理し、検索時には、ハッシュテーブルに変換して、ハッシュ検索を行う。

なお、本試験で10000データをList形式からHashMapに変換にかかった時間は25msec程度であった。

# 付録

検索キーが数値の場合の検索データ数毎の結果を付録A 1～付録A 4に示し、検索キーが文字列の場合の結果を付録A 5～付録A 8に示す。表中の数値の単位はmsecとする。

付録A 1　数値　検索データ数　10

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 検索データ数  10 | 回数 | | | | | 平均 |
| 1 | 2 | 3 | 4 | 5 |
| 線形検索 | 10 | 11 | 11 | 11 | 10 | 10.6 |
| 改良型線形検索 | 8 | 8 | 8 | 8 | 8 | 8 |
| 2分探索 | 5 | 5 | 5 | 5 | 5 | 5 |
| HashMap | 6 | 5 | 5 | 5 | 4 | 5 |
| TreeMap | 4 | 4 | 4 | 4 | 3 | 3.8 |
| LinkedHashMap | 4 | 3 | 3 | 4 | 3 | 3.4 |

付録A 2　数値　検索データ数　100

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 検索データ数  100 | 回数 | | | | | 平均 |
| 1 | 2 | 3 | 4 | 5 |
| 線形検索 | 50 | 53 | 53 | 53 | 43 | 50.4 |
| 改良型線形検索 | 39 | 39 | 46 | 45 | 46 | 43 |
| 2分探索 | 12 | 12 | 11 | 14 | 10 | 11.8 |
| HashMap | 7 | 7 | 4 | 4 | 4 | 5.2 |
| TreeMap | 4 | 4 | 5 | 5 | 3 | 4.2 |
| LinkedHashMap | 4 | 4 | 3 | 6 | 3 | 4 |

付録A 3　数値　検索データ数　1000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 検索データ数  1000 | 回数 | | | | | 平均 |
| 1 | 2 | 3 | 4 | 5 |
| 線形検索 | 48 | 73 | 60 | 63 | 48 | 58.4 |
| 改良型線形検索 | 42 | 46 | 51 | 51 | 37 | 45.4 |
| 2分探索 | 11 | 12 | 13 | 13 | 11 | 12 |
| HashMap | 5 | 4 | 4 | 4 | 4 | 4.2 |
| TreeMap | 3 | 3 | 5 | 5 | 3 | 3.8 |
| LinkedHashMap | 4 | 3 | 4 | 3 | 2 | 3.2 |

付録A 4　数値　検索データ数　10000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 検索データ数  10000 | 回数 | | | | | 平均 |
| 1 | 2 | 3 | 4 | 5 |
| 線形検索 | 257 | 252 | 259 | 315 | 270 | 270.6 |
| 改良型線形検索 | 248 | 249 | 249 | 206 | 249 | 240.2 |
| 2分探索 | 15 | 14 | 14 | 12 | 12 | 13.4 |
| HashMap | 6 | 7 | 5 | 4 | 7 | 5.8 |
| TreeMap | 7 | 8 | 4 | 5 | 8 | 6.4 |
| LinkedHashMap | 7 | 7 | 5 | 5 | 7 | 6.2 |

付録A 5　文字列　検索データ数　10

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 検索データ数  10 | 回数 | | | | | 平均 |
| 1 | 2 | 3 | 4 | 5 |
| 線形検索 | 16 | 14 | 15 | 16 | 14 | 15 |
| 改良型線形検索 | 9 | 10 | 9 | 9 | 10 | 9.4 |
| HashMap | 3 | 3 | 3 | 3 | 4 | 3.2 |
| TreeMap | 2 | 2 | 3 | 3 | 3 | 2.6 |
| LinkedHashMap | 2 | 2 | 3 | 3 | 3 | 2.6 |

付録A 6　文字列　検索データ数　100

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 検索データ数  100 | 回数 | | | | | 平均 |
| 1 | 2 | 3 | 4 | 5 |
| 線形検索 | 66 | 70 | 78 | 72 | 89 | 75 |
| 改良型線形検索 | 71 | 69 | 71 | 79 | 69 | 71.8 |
| HashMap | 2 | 3 | 4 | 4 | 3 | 3.2 |
| TreeMap | 2 | 3 | 3 | 4 | 3 | 3 |
| LinkedHashMap | 2 | 3 | 3 | 4 | 3 | 3 |

付録A 7　文字列　検索データ数　1000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 検索データ数  1000 | 回数 | | | | | 平均 |
| 1 | 2 | 3 | 4 | 5 |
| 線形検索 | 80 | 70 | 74 | 73 | 82 | 75.8 |
| 改良型線形検索 | 73 | 60 | 63 | 61 | 70 | 68.4 |
| HashMap | 4 | 3 | 5 | 4 | 3 | 3.5 |
| TreeMap | 3 | 4 | 3 | 4 | 3 | 3.4 |
| LinkedHashMap | 3 | 4 | 3 | 3 | 2 | 3 |

付録A 8　文字列　検索データ数　10000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 検索データ数  10000 | 回数 | | | | | 平均 |
| 1 | 2 | 3 | 4 | 5 |
| 線形検索 | 949 | 929 | 986 | 958 | 950 | 954.4 |
| 改良型線形検索 | 1294 | 1106 | 1166 | 1095 | 1069 | 1146 |
| HashMap | 4 | 5 | 4 | 4 | 4 | 4.2 |
| TreeMap | 5 | 4 | 3 | 5 | 6 | 4.6 |
| LinkedHashMap | 4 | 5 | 4 | 3 | 3 | 3.8 |