A New Analysis of the AVL Tree Insertion Algorithm

NINGPING SUN[†],RYOZO NAKAMURA[‡], WENLING SUN[†‡],HONGBING ZHU[‡‡], AKIO TADA[‡‡‡]
[†]Department of Information and Computer Science,
Kumamoto National College of Technology,Japan
[‡]Kumamoto University,Japan
[†‡]Central University for Nationalities,China
[‡‡]Hiroshima Kokusai Gakuin University,Japan
[‡‡‡]Sojo University,Japan

2659-2 Suya,Nishigoshi, Kikuchi,Kumamoto 861-1102, JAPAN
sningping@cs.knct.ac.jp http://www.cs.knct.ac.jp/~sun/sun.html

Abstract: The mathematical analysis of the AVL tree insertion algorithm has never been done successfully. As an important step to this analysis, based on the assumption that each AVL tree with a given number of nodes and height is constructed equally likely, we propose the approaches to count the number of constructed AVL trees and the average number of the patterns associated with imbalance due to an AVL tree insertion, then derive formulae to compute the rebalancing probability. The numerical test results are presented finally.

Key-Words: Analysis of algorithms,AVL tree insertion algorithm, Rebalancing probability

1 Introduction

AVL tree is one of the best known balanced binary search tree algorithms([2],[3]). In this paper the height of a tree is defined to be the length of the longest path from its root to one of its leaf nodes. A binary search tree is called an *AVL tree* if the heights of the left and the right subtree of every node never differ by more than one. Fig.1 illustrates the interesting structure of an AVL tree. If the left subtree is a binary search tree with L nodes and height $h-1$,the right subtree must be a binary search tree with R ($R = n - L - 1$) nodes and height $h-1$ or $h-2$,and vice versa.

The above rule ensures that the tree never becomes substantially imbalance and the expected height of AVL trees is $O(\lg n)$. However, after an insertion we must review the height of the tree and rebalance it with single rotation or double rotation if an imbalance has been introduced.Hence,it is necessary to know the rebalancing probability before the algorithm is actually employed.

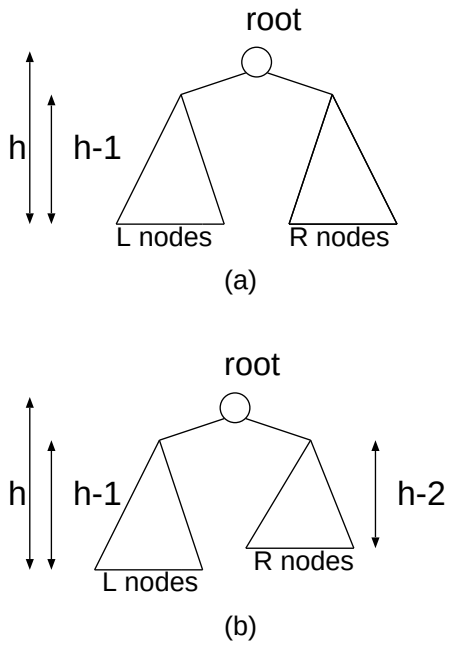


Fig.1 The Structure of the AVL tree with n nodes($n = L + R + 1$) and height h

In order to mathematically analyze the probability,we first present approaches to count the number of AVL trees with a given number of nodes and height and the average number of the patterns related to the imbalance, then derive the formulae to compute the rebalancing probability. Finally numerical test results of the rebalancing probability are provided.

1.1 The Analysis to Count the Number of AVL Trees

Adel'son-Vel'skii and Landis have proved that the height of an AVL tree with n internal nodes,h_n,is guaranteed to lie between the heights of the perfectly balanced tree and the Fibonacci tree([4]),*i.e.*,

$$\lceil \lg(n+1) \rceil - 1 \leq h_n \leq 1.4404 \lg(n+1) - 1.328,$$

where the lower bound is the height of perfectly balanced tree and the upper bound is the Fibonacci tree.

Given a height,h,the Fibonacci tree has the minimum number of nodes.Let f_h be the number of nodes of the Fibonacci tree with height of h,

$$f_h = F_{h+3} - 1 = \lfloor \phi^{h+3}/\sqrt{5} - 1 \rfloor, \qquad (1)$$

for $h \geq 0$.
Here F_{h+3} means the Fibonacci number of order $h+3$,whose closed form approximates to $\phi^{h+3}/\sqrt{5}, \phi = (1+\sqrt{5})/2$. On the other hand,the perfectly balanced tree has the maximum number of nodes, $2^{h+1} - 1$.

First,we examine the number of trees of these two special AVL trees. The structure of a Fibonacci tree is shown in Fig.1 (b), the number of trees can be expressed by the following recurrence,

$$Fib_h = 2Fib_{h-1}Fib_{h-2} \qquad (2)$$

for $h \geq 2$ with $Fib_0 = 1, Fib_1 = 2$.

We get

$$\begin{aligned}
Fib_0 &= 1 = 2^0, \\
Fib_1 &= 2 = 2^0 2^1, \\
Fib_2 &= 4 = 2^0 2^1 2^1, \\
Fib_3 &= 16 = 2^0 2^1 2^1 2^2, \\
Fib_4 &= 128 = 2^0 2^1 2^1 2^2 2^3, \\
Fib_5 &= 4096 = 2^0 2^1 2^1 2^2 2^3 2^5, \\
Fib_6 &= 1048576 = 2^0 2^1 2^1 2^2 2^3 2^5 2^8, ...
\end{aligned}$$

Thus,the number of the Fibonacci trees with height of h becomes

$$\begin{aligned}
Fib_h &= 2^{F_0} 2^{F_1} \cdots 2^{F_h} \\
&= \prod_{0 \leq i \leq h} 2^{F_i} \\
&\doteq 2^{\lfloor \frac{h+1}{\sqrt{5}} \cdot \frac{1-\phi^{h+1}}{1-\phi} \rfloor} \qquad (3)
\end{aligned}$$

for $h \geq 0$.

The perfectly balanced trees have the structure of Fig.1(a), the number of trees can be obtained by

$$Pbt_h = Pbt_{h-1}Pbt_{h-1} \qquad (4)$$

for $h \geq 1$ with $Pbt_0 = 1$.

Obviously,given any height,the number of the perfectly balanced trees is always 1.

To count the number of AVL trees with a given number of nodes and height, we first consider the range of L,the number of nodes of the left subtree. Because a Fibonacci tree is the AVL tree with the minimum number of nodes for a given height, the lower bound of L will be f_{h-1}. Meanwhile, the left subtree of height $h-1$ must have no more than $2^h - 1$ nodes, and the height of the right subtree is no less than $h-2$, *i.e.*,the minimum number of nodes of the right subtree is f_{h-2}. Therefore,the upper bound of L should be $\min\{2^h - 1, n - 1 - f_{h-2}\}$. Thus,

$$f_{h-1} \leq L \leq \min\{2^h - 1, n - 1 - f_{h-2}\},$$

we denote this range of L as $r(L_{h-1})$.

Secondly,we consider the range of R,the number of nodes of the right subtree,$R = n - L - 1$. In the structure of Fig.1 (a), the range of R is the same as the range of L,therefore $r(R_{h-1})$ is denoted for

$$f_{h-1} \leq R \leq \min\{2^h - 1, n - 1 - f_{h-2}\}.$$

On the other hand,in the structure of Fig.1(b),the height of right subtree is $h-2$,the lower bound of R is f_{h-2},the upper bound is $\min\{2^{h-1} - 1, n - 1 - f_{h-1}\}$, $r(R_{h-2})$ stands for

$$f_{h-2} \leq R \leq \min\{2^{h-1} - 1, n - 1 - f_{h-1}\}$$

Under our presumption that each AVL tree with a given number of nodes and height is equally likely,from above arguments, the number of the AVL trees with n nodes and height h,denoted by $Avl_{n,h}$, can be expressed as

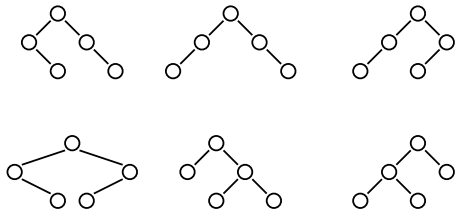
$$\begin{aligned} Avl_{n,h} &= \sum_{r(L_{h-1})\bigcap r(R_{h-1})} Avl_{L,h-1}Avl_{R,h-1} \\ &\quad +2\sum_{r(L_{h-1})\bigcap r(R_{h-2})} Avl_{L,h-1}Avl_{R,h-2} \end{aligned} \tag{5}$$

for $n \geq 3$ and $h \geq 1$ with $Avl_{1,0} = 1, Avl_{2,1} = 2$.

For example,

$$\begin{aligned} Avl_{5,2} &= Avl_{2,1}Avl_{2,1} + 2Avl_{3,1}Avl_{1,0} \\ &= 2 \times 2 + 2 \times 1 \times 1 \\ &= 6, \end{aligned}$$

for the following AVL trees with 5 nodes and height 2.



Eq.(5) is an essential formula in our analysis. Some results from Eq.(5) are shown in Table 1 and Fig.2.

From our results, given a height of tree, n starts from f_h and ends at $2^{h+1} - 1$, the most left values of $Avl_{n,h}$ in Fig.2 are the number of Fibonacci tree, and the most right ones are the number of perfectly balanced tree.When n is increasing and approaching to the middle point of between f_h and $2^{h+1} - 1$,the number of trees increases,the maximum value of $Avl_{n,h}$ is around at this middle point. Separated from this middle point,the larger the n, the smaller the number of AVL trees. The values of $Avl_{n,h}$ approximate bilateral symmetry to this middle point.

2 The Analysis of Rebalancing Probability Due to an Insertion

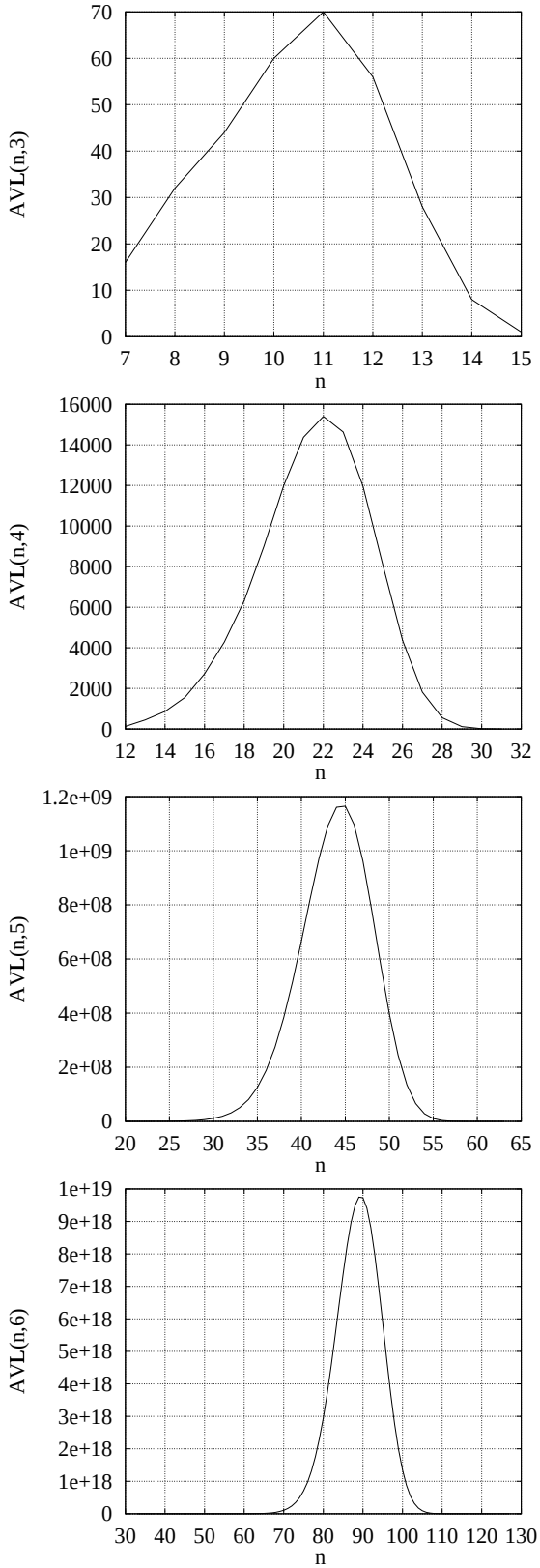


Fig.2 The number of the AVL trees with n node and height h

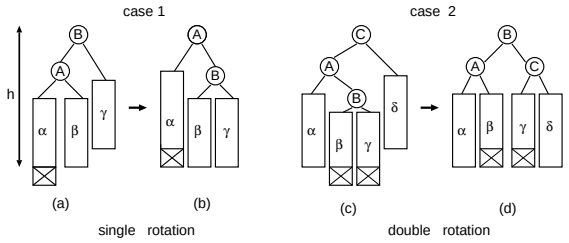


Fig.3 Imbalance resulting from the insertion and restoring the balance

Fig.3(a) and (c) illustrates the two essentially imbalance cases due to an insertion. The other two essentially identical cases will occur if we interchange the left and the right subtrees in Fig.3(a) and (c). In these diagrams the rectangles $\alpha, \beta, \gamma, \delta$ represent the subtrees with the respective heights, and the nodes added by the insertions are indicated by crosses.In case 1, using a single rotation we simply rotate the tree to the right, attaching β to B instead of A. While in case 2,we use a double rotation, first rotating (A, B) left then (B, C) right. These simple transformations restore the desired balance and are shown in Fig.3(b) and (d) respectively.

It is extremely difficult to find the rebalancing probability only based on those two cases shown in Fig.3. We have revealed that there are only two patterns in AVL trees,*pattern-1* and *pattern-2*,associated with the imbalance([1]). They are shown in Fig.4 (a) where the nodes added by the insertions are indicated by crosses. Based on these two proposed patterns, single and double rotations are indicated by two essentially different types of subtrees in Fig.4(b),where S stands for a single rotation and D for a double rotation.

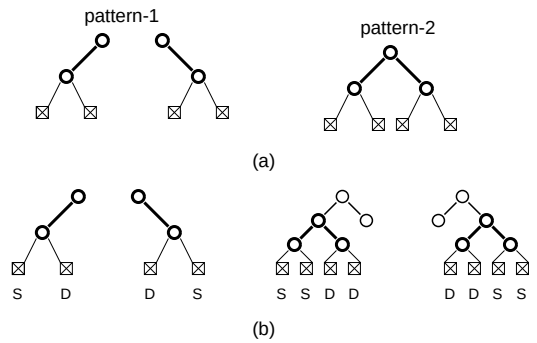


Fig.4 Two essential patterns in an AVL tree

Obviously,an insertion at the external nodes of the leaf of pattern-1 should cause imbalance as shown in Fig.5(a). Differently, inserting a new key into the external nodes of pattern-2 could cause imbalance or not. Only the pattern-2's in the subtrees which are taller than their adjacent subtrees are associated with imbalance,such as those drawn within the dotted square in Fig.5(b).

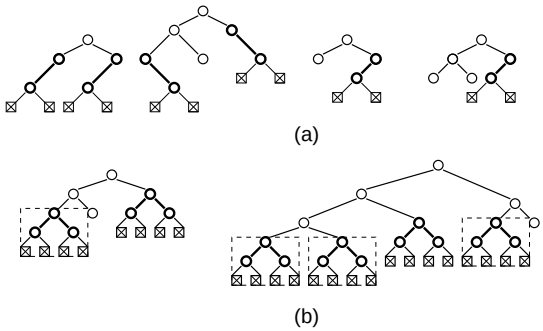


Fig.5 The instances of imbalance related to pattern-1 and pattern-2

Under the presumption that each different AVL tree with a given number of nodes and height is constructed equally likely, we only need to count the average number of all pattern-1 and pattern-2 which causes imbalance. The average number of pattern-1 is denoted as $p_{1_{n,h}}$ and the average number of the pattern-2 that will cause imbalance is denoted as $p_{2_{n,h}}$.

$$p_{1_{n,h}} = \frac{1}{Avl_{n,h}} [\sum_{r(L_{h-1}) \bigcap r(R_{h-1})} Avl_{L,h-1} Avl_{R,h-1} \{p_{1_{L,h-1}} + p_{1_{R,h-1}}\} + 2 \sum_{r(L_{h-1}) \bigcap r(R_{h-2})} Avl_{L,h-1} Avl_{R,h-2} \{p_{1_{L,h-1}} + p_{1_{R,h-2}}\}], \quad (6)$$

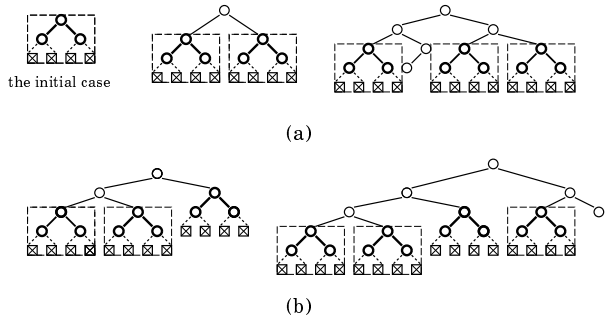
for $n \geq 2$ and $h \geq 1$ with $p_{1_{2,1}} = 1$.

And,

$$p_{2_{n,h}} = \frac{1}{Avl_{n,h}} [\sum_{r(L_{h-1}) \bigcap r(R_{h-1})} Avl_{L,h-1} Avl_{R,h-1} \{p_{2_{L,h-1}} + p_{2_{R,h-1}}\} + 2 \sum_{r(L_{h-1}) \bigcap r(R_{h-2})} Avl_{L,h-1} Avl_{R,h-2} \{p_{3_{L,h-1}} + p_{2_{R,h-2}}\}], \quad (7)$$

for $n \geq 5$ and $h \geq 2$ with $p_{2_{5,2}} = \frac{1}{3}$.

Here, $p3_{n,h}$ is the indirectly recursive function of $p2_{n,h}$, where for every node, if the heights of its two subtrees are the same, all pattern-2 in both of the subtrees are counted, which is shown as the dotted square in the following figure (a).However, if the heights of its two subtrees are different, all pattern-2 in the taller subtree are counted,but in the lower subtree, only the pattern-2 that will cause imbalance are counted,which is exampled in the following figure (b) as well.



$p3_{n,h}$ is derived as follows,

$$p3_{n,h} = \frac{1}{Avl_{n,h}}[\sum_{r(L_{h-1})\bigcap r(R_{h-1})} Avl_{L,h-1}Avl_{R,h-1}\{p3_{L,h-1}+p3_{R,h-1}\} + 2\sum_{r(L_{h-1})\bigcap r(R_{h-2})} Avl_{L,h-1}Avl_{R,h-2}\{p3_{L,h-1}+p2_{R,h-2}\}], \quad (8)$$

for $n \geq 3$ and $h \geq 1$ with $p3_{3,1} = 1$.

As shown in Fig.4(b), a single rotation and a double rotation will happen with the equal probability. For a rotation, pattern-1 and 2 has one and two inserting positions respectively. We use $PR(n,h)$ to express the probability that a single or a double rotation should be used to restore the desired balance when a new node is inserted into the AVL tree with n nodes and height h.

$$PR(n,h) = \frac{1}{n+1}p1_{n,h} + \frac{2}{n+1}p2_{n,h} \quad (9)$$

Moreover,let $P(n)$ be the probability that a single or a double rotation will occur when the $(n+1)$th node is inserted into the AVL tree with n nodes.

$$P(n) = \frac{1}{Avl_n}\sum_h Avl_{n,h}PR(n,h) \quad (10)$$

where $Avl_n = \sum_h Avl_{n,h}$ and $\lceil \lg(n+1) \rceil - 1 \leq h \leq 1.44\lg(n+1) - 1.328$.

3 Numerical Tests

We now provide some test results from above analysis. Table 1 shows some simple results.

Table 1: Some results of the above analysis

n	h	$Avl_{n,h}$	$p1_{n,h}$	$p2_{n,h}$	$PR(n,h)$	$P(n)$
1	0	1	0	0	0	0
2	1	2	1	0	0.3333	0.3333
3	1	1	0	0	0	0
4	2	4	1	0	0.2	0.2
5	2	6	1.3333	0.3333	0.3333	0.3333
6	2	4	1	0	0.1429	0.1429
7	2	1	0	0	0	
7	3	16	2	0	0.25	0.16

From Table 1,we see that the pattern-1's play a larger role than pattern-2's since only some of the latter could cause imbalance.

Fig.6 shows the results of the probability that an insertion requires rebalancing when a new key is inserted into the AVL tree with a given number of nodes and height,obtained from Eq.(9).

In the results of Fig.6, for a given height of tree,h, n starts from f_h and ends at $2^{h+1}-1$. When n is almost equal to f_h, $PR(n,h)$ increases. With the increasing of number of nodes, the number of pattern-1 decreases and the value of $p2_{n,h}$ increases,$PR(n,h)$ becomes gradually smoother and steadier. When the shape of trees turns to the perfectly balanced tree,in which there are neither pattern-1 nor pattern-2,$PR(n,h)$ gets to zero. Moreover,we can see the maximum value of $PR(n,h)$ approximates 0.27 for any given height of tree when $h \geq 3$.

Fig.7 gives the probability that rebalance rotations occur during AVL tree insertion operations. As expected,the value of $P(n)$ will be between 0.26 and 0.27 when $n \geq 20$. Because each configuration of AVL tree has a mirror-image variant in which the out-of-balance subtree is on the other side of the tree, the necessary rebalancing could occur on average within about 0.54 probabilities for a large n. This result says that one rebalance is needed for two insertions on average.

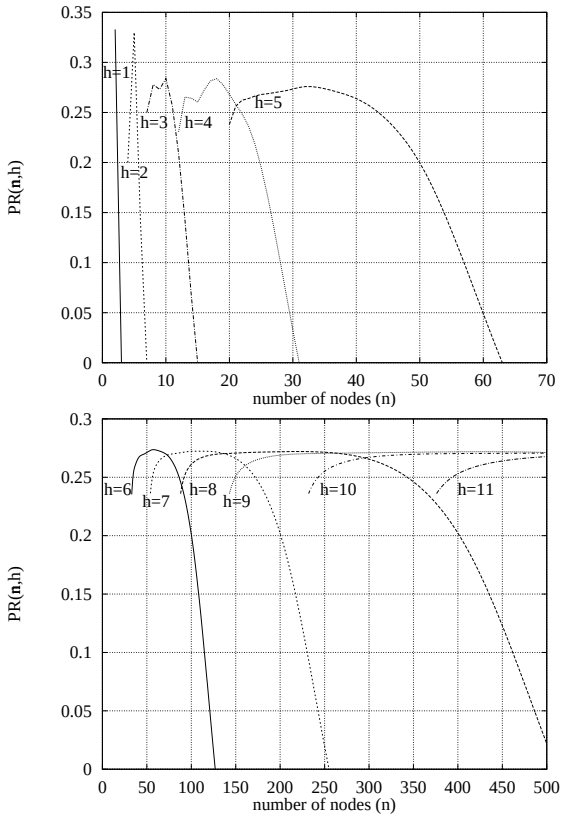


Fig.6 The test results of $PR(n,h)$

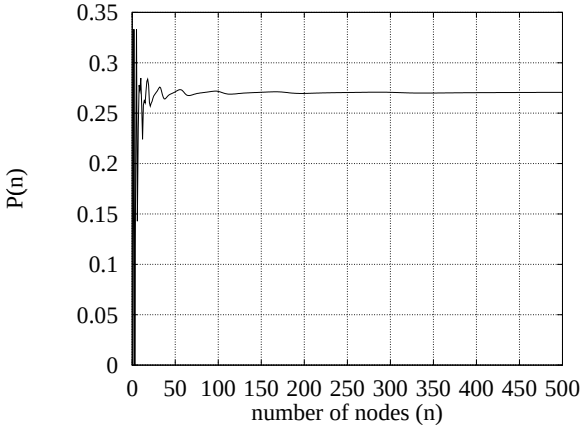


Fig.7 The test results of $P(n)$

4 Conclusions

In above analysis,we have presented an important approach to analyze the AVL tree insertion algorithm. A key to our analysis is the description of the structures of AVL tree shown Fig.1. The other key is the discovery of the two patterns related to imbalance.

The proposed formulae can evaluate the average performance of the AVL tree insertion algorithm exactly. Our future work is to obtain the asymptotic expression of these open recurrence formulae.

References

[1] R. Nakamura, N. Sun, T. Nakashima: An Approximate Analysis of the AVL Balanced Tree Insertion Algorithm,*Trans. IPS. Japan*, Vol.39,No.4,pp.1006-1013,1998.

[2] D. E. Knuth:*The Art of Computer Programming*,Vol.3, Sorting and Searching, Addison-Wesley,pp.458-470,1998.

[3] N. Wirth: *Algorithms + Data Structures = Programs*,Prentice- Hall,pp.216-226,1976.

[4] G.M. Adel'son-Vel'skii, E. M. Landis: An Algorithm for the Organization of Information, *English trans. in Soviet Math*, Vol.3,pp.1259-1263,1962.