On the isomorphism of two algorithms : Hu/Tucker and Garsia/Wachs

by

Kurt Mehlhorn

Marcos Tsagarakis

Fachbereich 10 -
Angewandte Mathematik
und Informatik der
Universität des Saarlandes
D-6600 Saarbrücken
W-Germany

January 1979

A 79/01

In 1969 T.C. Hu and K.C. Tucker succeeded in proving the correctness of Hu's algorithm for constructing optimum binary search trees (cf. Knuth, Vol. 3, p. 447 for a historical account). The proof is extremely complicated and until now no simpler proof was found. T.C. Hu published a simplified proof in 1973; however a key lemma in that paper is wrong (see Tsagarakis).

In 1977 Garsia and Wachs published a new algorithm for constructing optimum binary search trees and gave a fairly short correctness proof for it.

Both algorithms use sequences of real numbers as their underlying data structure. We show that the two algorithms manipulate these sequences in an isomorphic way, i.e. the two algorithms, though completely different and hardly related in wording, are the same as much as the manipulation of the underlying data structure is concerned. One immediate consequence of this result is a simple correctness proof for the Hu/Tucker-algorithm.

II. Definitions :

Both algorithms construct the optimal alphabetic tree for a sequence $\alpha_1\ \alpha_2\ \ldots\ \alpha_n$ of positive reals in two phases. In phase I a non-alphabetic tree T is constructed with the following properties:

1) the weighted path length of tree T is equal to the weighted path length of an optimal alphabetic tree.

2) let ℓ_i be the depth of leaf α_i in tree T, $1 \leq i \leq n$. Then there is an optimal alphabetic tree having α_i at depth ℓ_i.

In phase II an optimal alphabetic tree is constructed using the sequence $\ell_1, \ell_2, \ldots, \ell_n$ of depths of the α_i. Phase II is the same in both algorithms; we will only be concerned with phase I of the algorithms. It is time-consuming and difficult to validate. We next describe phase I of both algorithms.

II.1 Garsia/Wachs: Let $\alpha_1, \alpha_2, \ldots, \alpha_n$ be a sequence of positive reals.

Definition:

a) Pair (α_{i-1}, α_i) is right-minimal if $\alpha_{i-2} + \alpha_{i-1} \geq \alpha_{i-1} + \alpha_i$ $< \alpha_{j-1} + \alpha_j$ for all $j > i$.

b) Pair (α_{i-1}, α_i) is right-minimum if $\alpha_{k-1} + \alpha_k \geq \alpha_{i-1} + \alpha_i$ for all $k < i$ and $\alpha_{i-1} + \alpha_i < \alpha_{j-1} + \alpha_j$ for all $j > i$. □

The right-minimum pair is uniquely determined. It is always right-minimal. Phase I of the G.W. algorithm works iteratively: Each iteration works as follows:

Let (α_{i-1}, α_i) be the right-minimum pair and let $k \geq 0$ be such that $\alpha_{i-1} + \alpha_i > \alpha_{i+j}$ for $1 \leq j \leq k$ and $\alpha_{i-1} + \alpha_i \leq \alpha_{i+k+1}$ or $i+k = n$. A new sequence is formed by deleting α_{i-1} and α_i and inserting their sum $\alpha_{i-1} + \alpha_i$ immediately to the left of α_{i+k+1}

(at the right end if i+k = n), i.e. the new sequence is

$\alpha_1\ \alpha_2 \cdots \alpha_{i-2}\quad \alpha_{i+1} \cdots \alpha_{i+k}\ \alpha_{i-1} + \alpha_j\ \alpha_{i+k+1} \cdots \alpha_n$

Figure 1 shows the execution of the G.W.-algorithm on the sequence 5,3,3,2,2,2,3,4 (taken from [GW]). Untangling the lines in Fig. 1 gives the tree of Fig. 2, i.e. both trees have the same solid lines.

II.2 Hu/Tucker: The H.T.-algorithm also works on sequences of reals $\alpha_1,\alpha_2,\ldots,\alpha_n$. In addition, each real is in either one of two states: it is either a circle or a square. Such sequences are called H.T.-sequences. In the initial sequence all reals are squares. We draw such sequences by enclosing each real in the appropiate geometric figure.

Definition:

a) A pair (α_i,α_k) with i < k is tentative-connecting (T.C.) if α_j is a circle for i < j < k.

b) A pair (α_i,α_j) is a minimal T.C. pair, if it is T.C. and $\alpha_i + \alpha_j \leq \alpha_h + \alpha_k$ for all T.C. pairs (α_h,α_k).

c) A pair (α_i,α_j) is the minimum T.C. pair (min. T.C. pair) if it is a minimal T.C. pair and for all other minimal T.C. pairs (α_h,α_k) either h < i or h = i and k > j.

Example: In the sequence

α_1	α_2	α_3	α_4	α_5
(3)	[1]	(2)	[2]	[2]

the pairs (α_2,α_3) and (α_2,α_4) are minimal TC-pairs, the pair (α_2,α_3) is the minimum TC-pair.

Phase I of the H.T.-algorithm works iteratively. Each iteration works as follows:

Let (α_i,α_j) be the minimum T.C. pair. Form a new sequence by deleting α_i and replacing α_j by the circle $\alpha_i + \alpha_j$.

Figure 3 shows the execution of the H.T.-algorithm on the initial sequence $\boxed{5},\boxed{3},\boxed{3},\boxed{2},\boxed{2},\boxed{2},\boxed{3},\boxed{4}$. Untangling this tree also gives the tree of Fig. 2., i.e. <u>both algorithms performed the same sequence of combinations in the same order</u>!!! They manipulate the data structure in an isomorphic way.

III. Proof of Equivalence

In this section we want to show that the two algorithms perform the identical sequence of combinations. The two algorithms differ in two respects: Only neighboring reals can be combined by the G.W.-algorithms, whilst tentative connecting reals can be combined by the H.T.-algorithm. The G.W.-algorithm reorders the sequence after a combination and the H.T.-algorithm does not. We introduce a third algorithm, the M.T.-algorithm, which shows features of both, and show that it is equivalent to the G.W.- and the H.T.-algorithm.

<u>M.T.-algorithm:</u> Let (α_i,α_j) be the minimum T.C. pair and let $k \geq 0$ be such that $\alpha_{j+k+1} > \alpha_i + \alpha_j \geq \alpha_{j+\ell}$ for all $0 < \ell \leq k$. Form a new sequence by deleting α_i and α_j and inserting the circle $\alpha_i + \alpha_j$ immediately to the left of α_{j+k+1}.

Figure 4 shows execution of the M.T.-algorithm on initial sequence 5,3,3,2,2,2,3,4. Note that Figure 4 is the same as Figure 1 except for the fact that the reals are enclosed in circles and squares this time. We will first show that the M.T.- and the G.W.-algorithm are equivalent.

Definition:

A H.T.-sequence $\beta_1,\ldots,\beta_n$ (i.e. a sequence of reals enclosed in squares and circles) is an actual M.T.-sequence if there is some initial sequence $\alpha_1,\ldots,\alpha_m$ consisting of squares such that some number of iterations of M.T. produces $\beta_1,\ldots,\beta_n$.

Lemma 1:

a) Let $\alpha_1,\ldots,\alpha_n$ be an actual M.T.-sequence and let α_i be a circle. Then $\alpha_i \leq \alpha_{i+1}$.

b) Let $\beta_1,\ldots,\beta_n$ be an actual M.T.-sequence. Then the minimum T.C.-pair of that sequence is the right-minimum pair of that sequence.

Proof: We prove a) and b) simultaneously by induction on the number m of iterations. For m = 0 a) is trivial since there are no circles. Then b) follows immediately from the definition of minimum T.C.-pair and right-minimum pair. Suppose now the claim is true after m-1 iterations, and let $\alpha_1,\ldots,\alpha_n$ be an actual M.T.-sequence obtained after m-1 iterations. Since b) is true for that sequence the minimum T.C.-pair is (α_{i-1},α_i) for some i. Let $k \geq 0$ be such that $\alpha_{i+k+1} \geq \alpha_{i-1} + \alpha_i > \alpha_{i+j}$ for $1 \leq j \leq k$. Then the sequence after m iterations is

$$\alpha_1,\ldots,\alpha_{i-2},\alpha_{i+1},\ldots,\alpha_{i+k},\alpha_{i-1} + \alpha_i,\ \alpha_{i+k+1},\ldots,\alpha_n.$$

We only have to show: if α_{i-2} is a circle then $\alpha_{i-2} \leq \alpha_{i+1}$. If α_{i-2} is a circle then $\alpha_{i-2} \leq \alpha_{i-1}$ since a) is true for sequence $\alpha_1,\ldots,\alpha_n$. Since (α_{i-1},α_i) is right-minimum we have $\alpha_{i-1} + \alpha_i \leq \alpha_i + \alpha_{i+1}$ and hence $\alpha_{i-1} < \alpha_{i+1}$. This shows $\alpha_{i-2} < \alpha_{i+1}$ and proves a).

Let $\beta_1, \beta_2, \ldots, \beta_n$ be any M.T.-sequence with:
if β_i is a circle then $\beta_i \leq \beta_{i+1}$. Let (β_h, β_j) be the minimum T.C.-pair. Then β_k is a circle for $h < k < j$ and hence $\beta_k \leq \beta_j$ for $h < k < j$. Hence $\beta_h + \beta_{h+1} \leq \beta_h + \beta_j$ and since (β_h, β_j) is the minimum T.C.-pair we must have $j = h+1$. This proves b). □

Corollary: Algorithms M.T. and G.W. perform exactly the same sequences of combinations.

Proof: By part b) of the lemma above the minimum T.C.-pair of an actual M.T.-sequence is always the right-minimum pair of that sequence. Thus the syntactic difference of the two algorithms disappears in the semantics. □

It remains to show that algorithms M.T. and H.T. are equivalent. To do so we define an equivalence relation on H.T. sequences and then show that the reordering performed by the M.T. algorithm preserves equivalence.

Definition:
Two H.T.-sequences are H.T.-equivalent if phase I of the H.T.-algorithm constructs the same tree from them.

Example: (2) [1] [1] [2] and [1] [1] (2) [2] are H.T.-equivalent since the tree of figure 5 is constructed in both cases

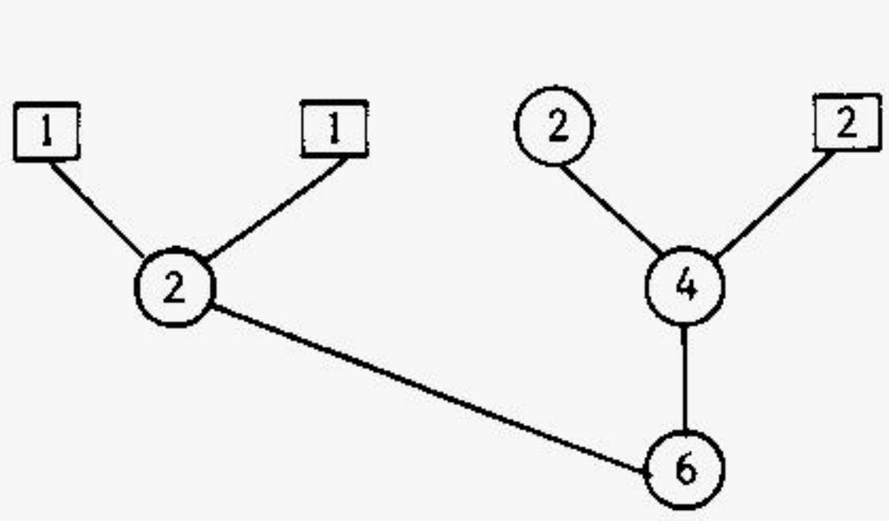


Fig. 5

We need to show that the reordering performed by the M.T.-algorithm preserves equivalence. Therefore we will compare sequences obtained from the same initial sequence by either m M.T.-steps or m H.T.-steps. We show by induction on m that the two sequences are equivalent. This is trivially true for m = 0. For the induction step we need one more fact about H.T.-sequences.

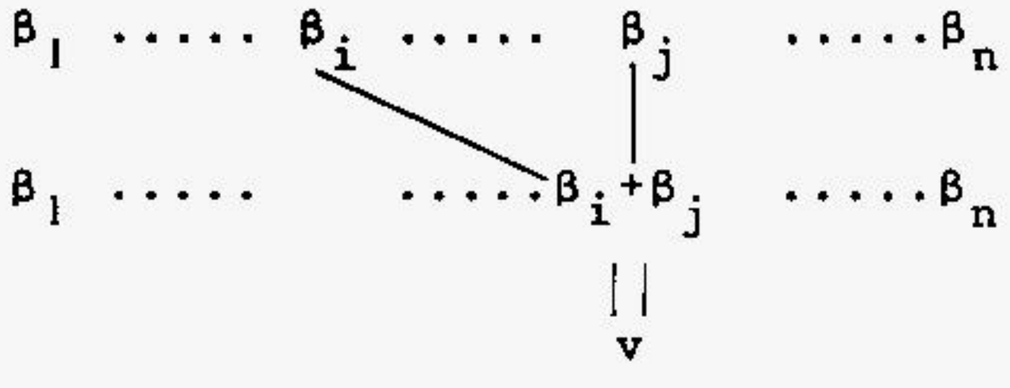
Let $\alpha_1,\alpha_2,\ldots,\alpha_n$ be any H.T.-sequence and let T be the tree constructed by algorithm H.T. on input sequence $\alpha_1,\ldots,\alpha_n$. In our example (Figure 3) interior node v was constructed before interior node w if and only if the real in node v is smaller than the real in node w. A more precise version of this observation appears in Lemma 2. Algorithm H.T. combines in each step the min T.C.-pair, deletes the left constituent of the pair from the actual sequence and replaces the right constituent by the sum. Drawing the execution of algorithm M.T. as in Fig. 3 allows us to introduce the partial ordering "to be to the left of" on the nodes of tree T. In our example (Figure 3) (6) is to the left of (4), (4) is to the left of (5) which in turn is to the left of (24), and so on.

<u>Lemma 2:</u> Let v and w be interior nodes of tree T constructed by algorithm H.T. on H.T. sequence $\alpha_1,\alpha_2,\ldots,\alpha_n$. Let v be ($\alpha$) and w be ($\beta$). Then:

v is constructed before w iff $\alpha < \beta$ or $\alpha = \beta$ and v is to the right of w.

<u>Proof:</u> The clause " $\alpha < \beta$ or $\alpha = \beta$ and v is to the right of w" defines a linear ordering on the set of nodes of T. Note that equal weighted nodes can be not descendants from one another. Hence it suffices to show: If v is immediately constructed before w then $\alpha < \beta$ or $\alpha = \beta$ and v is to the right of w.

Let $\beta_1,\ldots,\beta_m$ be the H.T. sequence immediately before the construction of v. Let β_i,β_j be the min T.C.-pair. Then v represents $\alpha := \beta_i + \beta_j$ at the previous position of β_j



Next w is formed. If v is a son of w then we certainly have $\alpha < \beta$. So suppose v is not a son of w. Then w is formed by combining β_h, β_k with $\{h,k\} \cap \{i,j\} = \emptyset$

Case 1: max $\{h,k\} < \min\{i,j\}$. Then β_h, β_k was T.C. just previous to the construction of v. Hence $\beta := \beta_h + \beta_k \geq \beta_i + \beta_j = \alpha$. Also v is to the right of w.

Case 2: $\min\{h,k\} > \max\{i,j\}$. Then β_h, β_k was T.C. just previous to the construction. Hence $\beta = \beta_h + \beta_k > \beta_i + \beta_j = \alpha$ by the definition of min T.C. pair.

Case 3: $i < h < j < k$ or $i < h < k < j$ or $h < i < j < k$ or $h < i < k < j$. Then β_i, β_h and β_j, β_k were T.C. just prior to the construction of v. Hence $\beta_h + \beta_i \geq \alpha$ and $\beta_j + \beta_k > \alpha$. This shows $\beta_h + \beta_k > \alpha$. □

We are now ready to prove the equivalence of algorithms M.T. and H.T..

Lemma 3: Let S = A w B C be an H.T.-sequence with $A = \beta_1\beta_2\ldots\beta_{i-1}$, $w = \beta_i$, $B = \beta_{i+1},\ldots,\beta_{i+k}$, $C = \beta_{i+k+1}\ldots\beta_n$ and

1) w is a circle
2) $v < w$ for all v in B
3) Execution of algorithm H.T. on S never creates a node of weight exactly w to the right of w.

Then S and $\tilde{S}$ = A B w C are H.T.-equivalent.

Proof: Assume otherwise. Let S be a counterexample of smallest length. We show first that the min T.C.-pairs of S and $\tilde{S}$ are identical.

a) If $u,v \neq w$ then u and v are T.C. in S if and only if they are T.C. in $\tilde{S}$. This follows from the fact that w is a circle.

b) If B contains at least two elements then w is not element of the min T.C.-pair of S (of $\tilde{S}$). This follows from the fact that all elements of B have weight strictly smaller than w.

c) If B contains exactly one element, say β_{i+1}, and $w = \beta_i$ is element of the min T.C. pair of S, then β_i, β_{i+1} is the min T.C. pair of S. It is also the min T.C. pair of $\tilde{S}$. Suppose $w = \beta_i$ is part of the min T.C. pair and β_{i+1} is not. Since w is a circle and $\beta_{i+1} < \beta_i$ this contradicts the definition of min T.C.-pair. So β_i, β_{i+1} is the min T.C. pair of S. It is also T.C. in $\tilde{S}$. Conversely, the min. T.C. pair of $\tilde{S}$ is T.C in S. Hence β_i, β_{i+1} is also the min T.C. pair of $\tilde{S}$.

In any case it follows from a), b) and c) that the min T.C. pairs of S and $\tilde{S}$ are identical, say β_j, β_h with $j < h$. Let S_1 and $\tilde{S}_1$ be the H.T. sequences obtained by one H.T.-step from S and $\tilde{S}$ respectively.

Case 1: $h < i$, then β_j and β_h both lie in A. Then $S_1 = A' w B C$ and $\tilde{S}_1 = A'B w C$ with $A' = \beta_1 \ldots \beta_{j-1} \beta_{j+1} \ldots \beta_{h-1} \beta_j + \beta_h \beta_{h+1} \ldots \beta_{i-1}$.

S_1 satisfies 1), 2) and 3) and hence S_1 and $\tilde{S}_1$ are H.T.-equivalent. So S and $\tilde{S}$ are H.T.-equivalent. Contradiction.

Case 2: $i+1 \leq j < h \leq i+k$, then β_j and β_h both lie in B. Then $S_1 = A w B'C$ and $S_1 = A B'w C$ with $B' = \beta_{i+1} \ldots \beta_{j-1} \beta_{j+1} \ldots \beta_{h-1} \beta_j + \beta_h \beta_{h+1} \ldots \beta_{i+k}$. Let $u = \beta_j + \beta_h$. If $u < w$ then S_1 satisfies 1), 2) and 3) and hence S_1 and $\tilde{S}_1$ are equivalent. If $u \geq w$ then $u > w$ by assumption 3). Hence $S_1 = A w B_1 u B_2 C$ satisfies 1), 2) and 3) with u instead of w and B_2 instead of B [it satisfies 3) because of Lemma 2]. Hence S_1 is H.T.-equivalent to $A w B_1 B_2 u C$ and this sequence is equivalent to $A B_1 B_2 w u C$. The same argument shows

that $\tilde{S}$ is H.T.-equivalent to A B_1 B_2 w u C.
The other cases (β_j in A, β_h in B or β_j in A, β_h in C or ...)
are treated analogously. □

The equivalence of algorithms H.T. and M.T. is a direct consequence of Lemma 3. Let $\alpha_1,\alpha_2,\ldots,\alpha_p$ be an initial sequence consisting of squares and let $S^{(m)}_{H.T.}$ and $S^{(m)}_{M.T.}$ be the sequences obtained from it by m H.T.-steps or M.T.-steps respectively. For m = 0 $S^{(o)}_{H.T.} = S^{(o)}_{M.T.}$. Suppose now that $S^{(m)}_{H.T.}$ and $S^{(m)}_{M.T.}$ are H.T.-equivalent. We want to show that $S^{(m+1)}_{H.T.}$ and $S^{(m+1)}_{M.T.}$ are H.T.-equivalent. Let S be the sequence obtained by one H.T.-step from $S^{(m)}_{M.T.}$, then S = A w B C with w being the node formed by that H.T.-step and v < w for all v in B and v $\geq$ w for the first node of C. S satisfies 1), 2) and 3) of Lemma 3 (Lemma 2 implies 3)) and $\tilde{S}$ = A B w C = $S^{(m+1)}_{M.T.}$. Hence S and $S^{(m+1)}_{M.T.}$ are H.T.-equivalent. Furthermore S and $S^{(m+1)}_{H.T.}$ are H.T.-equivalent by definition of H.T.-equivalence. This shows that $S^{(m+1)}_{H.T.}$ and $S^{(m+1)}_{M.T.}$ are H.T.-equivalent and proves:

Theorem: Algorithms H.T., M.T. and G.W. perform exactly the same sequence of combinations.

In particular, the Hu/Tucker-and Garsia/Wachs-algorithms manipulate their data structure in an isomorphic way. As an immediate consequence we obtain a new correctness proof for the Hu/Tucker-algorithm.

Bibliograhpy

T.C. Hu / K.C. Tucker : Optimum computer search trees,
SIAM J. Appl. Math., 21 (1971), 514-552

T.C. Hu: A new proof of the T.C.-Algorithm, SIAM J. Appl. Math.,
25 (1973), 83-94

D.E. Knuth: The Art of Computer Programming, Vol. 3, 1973

A.M. Garsia / M.L. Wachs: A new algorithm for minimum cost
binary trees, SIAM J. of Computing, 1977

M. Tsagarakis: Korrektheitsbeweis des Hu-Algorithmus, Diplom-
arbeit, FB 10, Universität des Saarlandes, Februar 1978

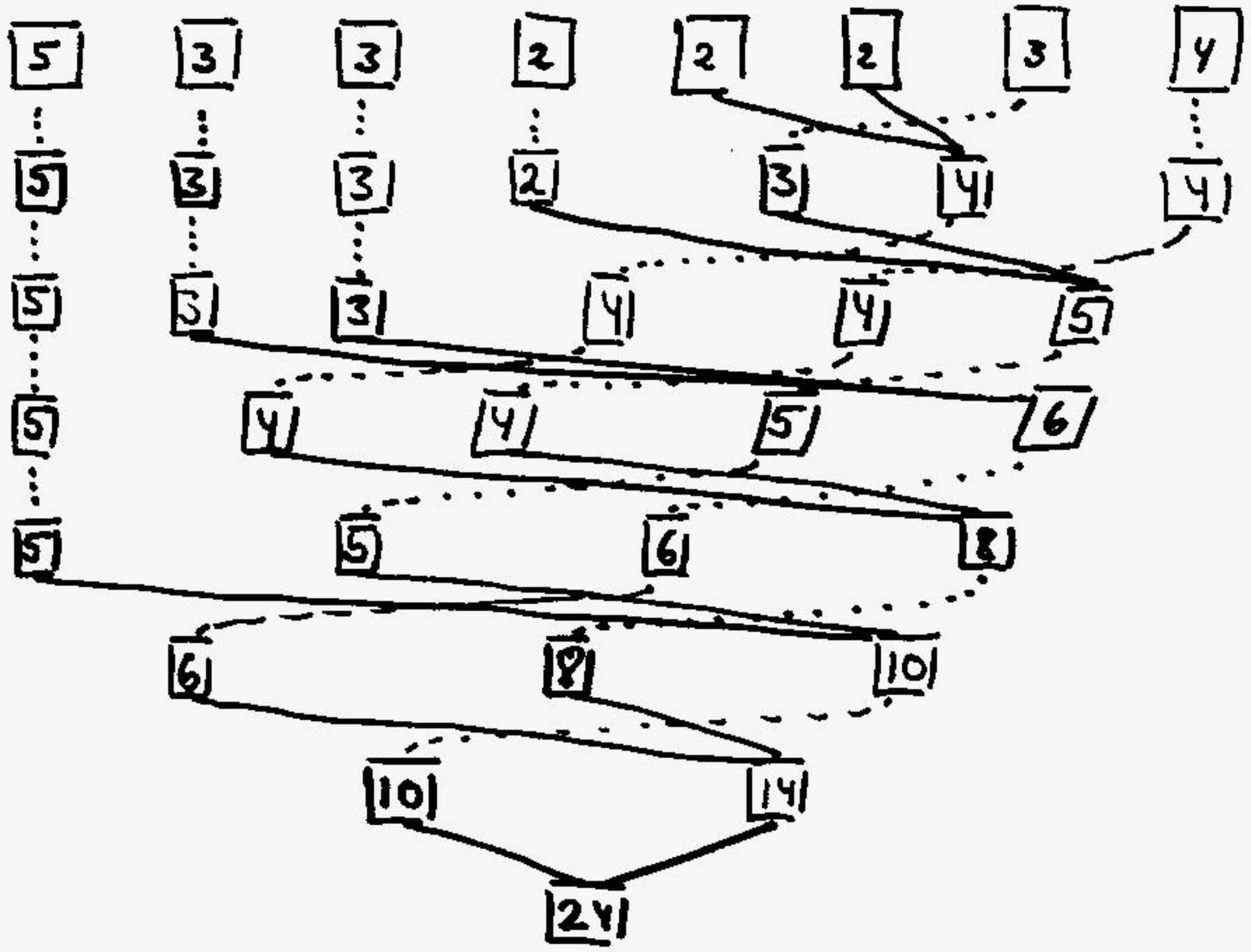


Figure 1 : Algorithm G.W. on sequence 5, 3, 3, 2, 2, 2, 3, 4

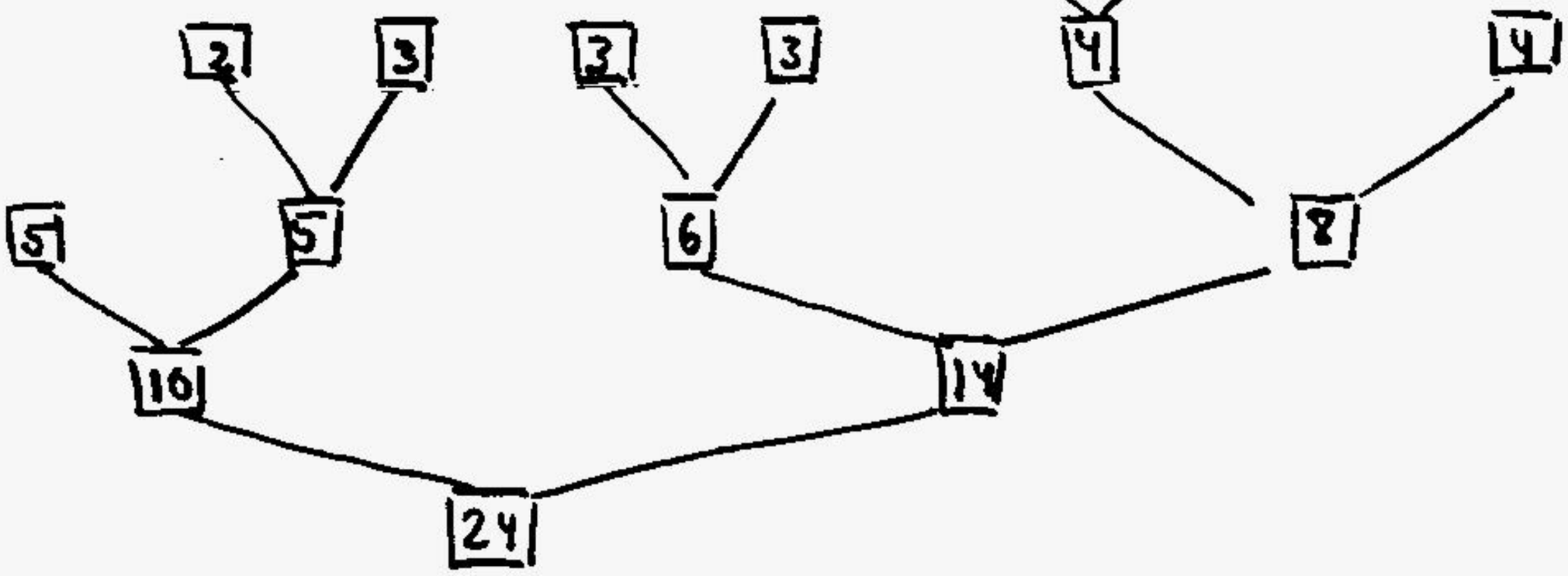


Figure 2 : The untangled version of the trees in Figure 1, 3 and 4.

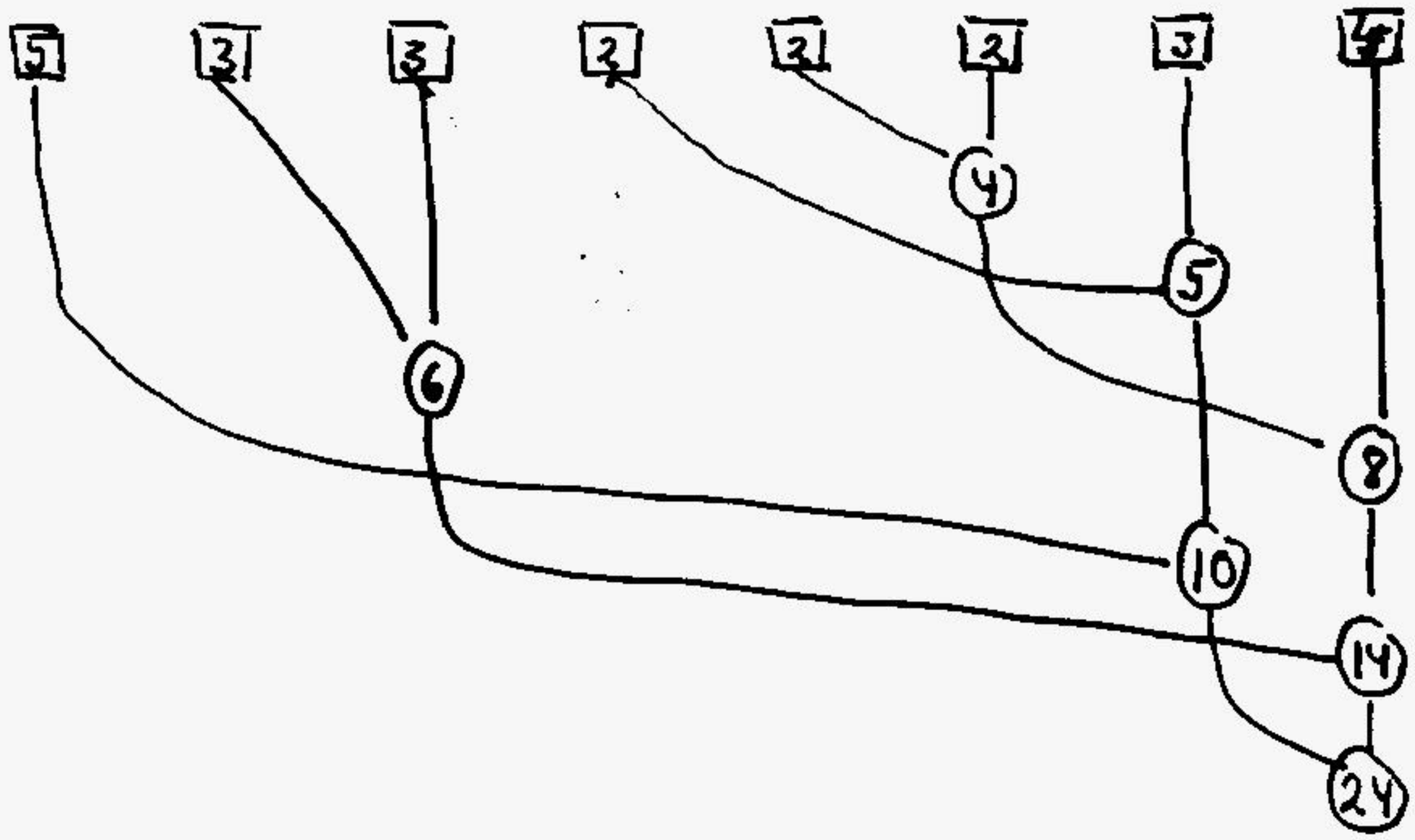


Figure 3 : Algorithm H.T. on sequence 5, 3, 3, 2, 3, 3, 3, 5.

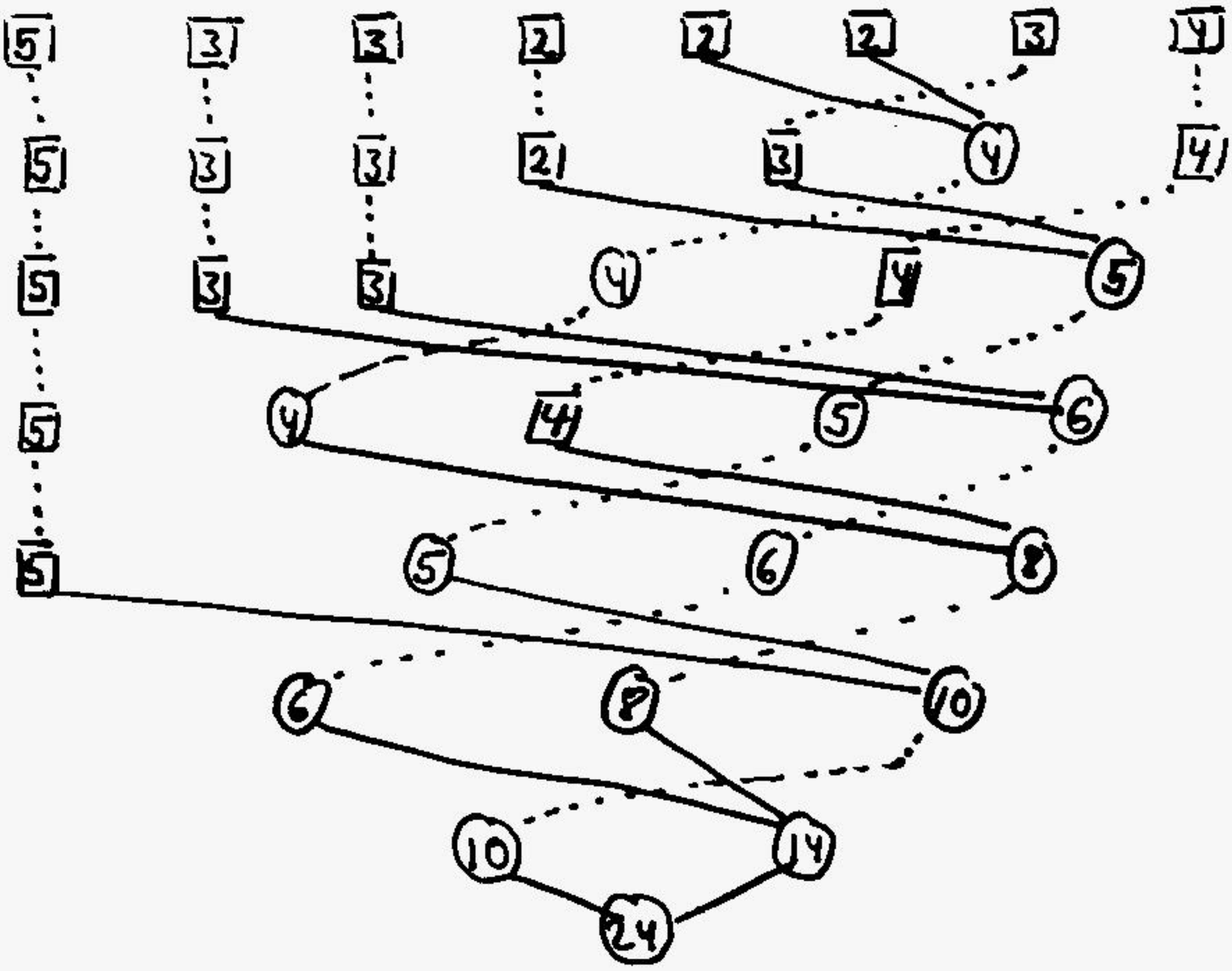


Figure 4 : Algorithm H.T. on sequence 5, 5, 3, 2, 2, 3, 3, 5 .