On the *k*-ary Tree Combinatorics

Sung-Hyuk Cha

Computer Science Department, Pace University
1 Pace Plaza, New York, NY 10038 USA
scha@pace.edu

Abstract. This article considers problems of counting the number of possible *k*-ary trees given *n* number of attributes. The simple underlying counting notions for lists are extended to the *k*-ary tree structure in a natural way. Three combinatorial formulae such as *k-ary tree factorial*, *r-permutation*, and *r-sequence* are formally defined in both product notation and recursive forms with their applications in *decision trees*. Moreover, the *k-ary tree r-sequence* has its application in *family trees*.

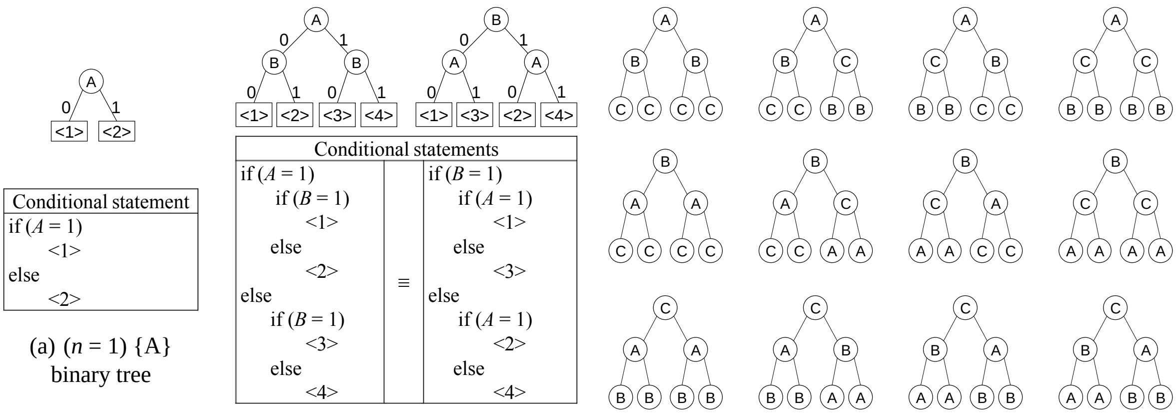
1 Introduction

Decision tree based classification, where the *leaves* represent classifications and the branches represent values of the attributes, have been widely used in *pattern classification* [1] and *machine learning* [2]. Finding the shortest decision tree is an important but hard optimization problem [3, 4]. In attempt to find an optimal decision tree, some combinatorial problems regarding decision trees were raised in [5]. The purpose of this article is to provide formulae for those problems.

Consider *binary decision trees* in Figure 1 where *inner nodes* are *Boolean literals* (attributes with two possible values) and *leaves* are targets. An attribute in an inner node in a *binary decision tree* has two branches where left and right *branches* represent 0 and 1 values, respectively. Decision trees are simply *if-else conditional statements* in programming languages. If there are two Boolean literals, two decision tree representations are possible. If there are three or more attributes, how many binary decision trees are possible? Earlier in [5], the notation Δ was used to denote the number of possible full binary decision trees with *n* attributes and it was formally defined as in eqn (1).

$$n\Delta = \prod_{i=1}^{n} i^{2^{n-i}} \tag{1}$$

For example, 2Δ = (1×1) × 2 = 2 and 3Δ = (1×1×1×1) × (2×2) × 3 = 12, etc.



(a) (*n* = 1) {A} binary tree

(b) (*n* = 2) {A, B} binary trees

(c) (*n* = 3) {A,B,C} binary trees

Figure 1. *Binary decision trees*

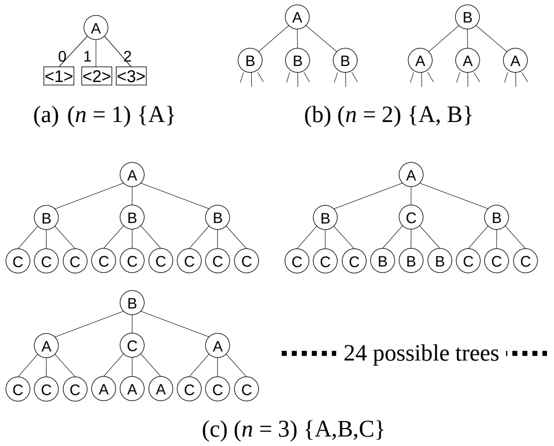


Figure 2. *Ternary decision trees*

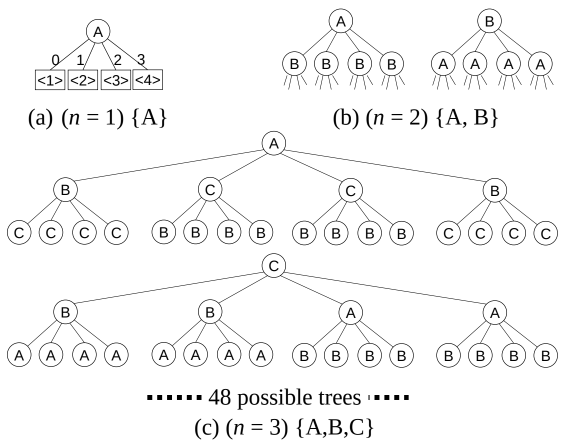


Figure 3. *Quaternary decision trees*

In this paper, this binary tree arrangement problem in [5] is generalized into the k-ary tree where every attribute has exactly k values, e.g., Figures 2 and 3 illustrate the *ternary* and *quaternary* trees where every attribute has exactly three and four values, accordingly. Suppose there are n kinds of Medical examinations which have '*positive*' and '*negative*' as their values for determining a certain disease. Different doctors may follow different binary decision trees in their practices. If the value, '*undecidable*' is included, ternary decision trees are formed. The section 2 shall provide the formula called '*k-ary tree factorial*' defined in both product and recursive forms to count all possible *k-ary* trees with n attributes.

Imagine that the insurance company limits the number of examinations conducted to each patient to be r where $1 \leq r \leq n$. Then decision trees with r height are formed as exemplified in Figure 4. Counting the number of these kinds of decision trees shall be referred to as the '*k-ary tree r-permutation*' problem defined and dealt in the section 3.

Since the result of an examination may be an error, the examination may be repeated to ensure their decisions. In this case, there are more possible k-ary decision trees as illustrated in Figure 5. This problem shall be called the '*k-ary tree r-sequence*' and defined and illustrated with the additional '*family tree*' example in the section 4.

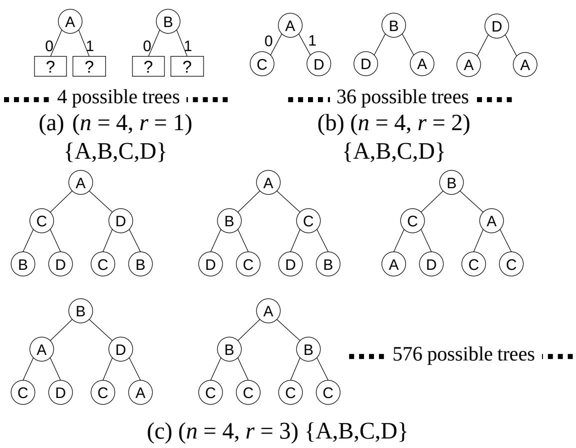


Figure 4. *r*-height binary decision trees without repetition

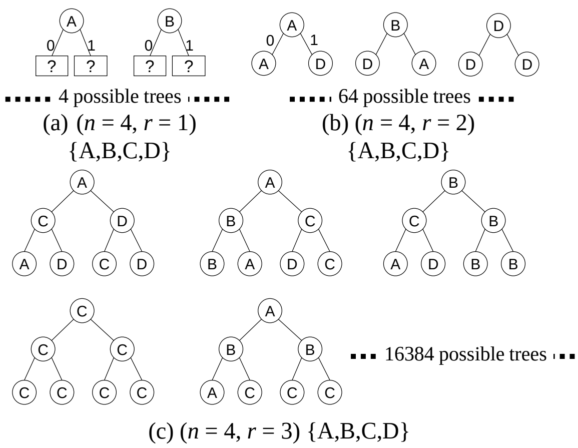
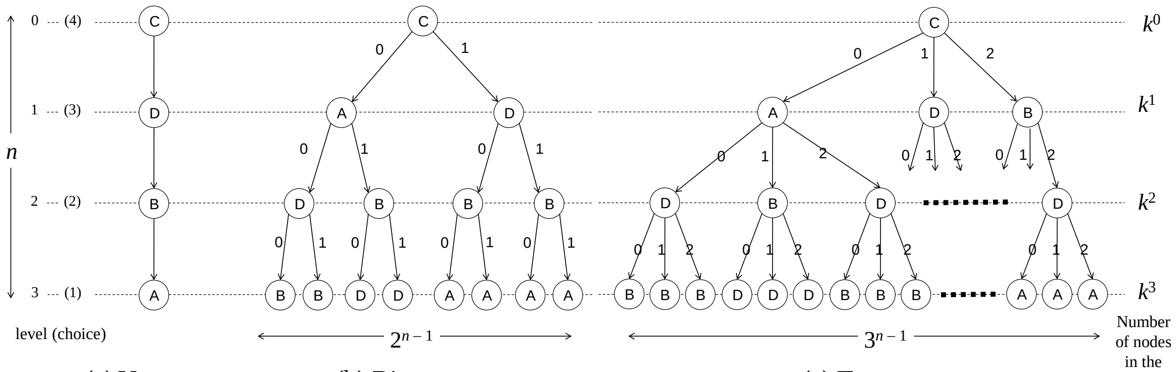


Figure 5. *r*-height binary decision trees with repetition

2 *k*-ary Tree Factorial



(a) Unary tree (k = 1) (b) Binary tree (k = 2) (c) Ternary tree (k = 3)

Figure 6. *k-ary tree factorial* (n = 4)

The *factorial* of a positive integer n is the product of all positive integers less than or equal to n as defined in eqn (2). It merely means the number of ways to arrange n different objects in order. This factorial concept can be generalized into the k-ary tree factorial as shown in Figure 6. From the k-ary tree point of view, the factorial is simply the unary (k = 1) tree.

Table 1. k-ary tree factorial definition and examples.

name	Product def.	Recursive def.	examples	
unary	$n! = \prod_{i=1}^{n} i$	$n! = \begin{cases} n \times (n-1)! & \text{if } n > 1 \\ 1 & \text{if } n = 1 \end{cases}$	3! = 1 × 2 × 3 = 6 4! = 1 × 2 × 3 × 4 = 24 5! = 1 × 2 × 3 × 4 × 5 = 120	(2)
binary	$n!! = \prod_{i=1}^{n} i^{2^{n-i}}$	$n!! = \begin{cases} n \times ((n-1)!!)^2 & \text{if } n > 1 \\ 1 & \text{if } n = 1 \end{cases}$	2!! = (1 × 1) × 2 = 2 3!! = (1×1×1×1) × (2×2)× 3 = 12 4!! = (2×2×2×2) × (3×3)× 4=576	(3)
ternary	$n!!! = \prod_{i=1}^{n} i^{3^{n-i}}$	$n!_{<3>} = \begin{cases} n \times ((n-1)!_{<3>})^3 & \text{if } n > 1 \\ 1 & \text{if } n = 1 \end{cases}$	2!!!= (1 × 1 × 1) × 2 = 2 3!!! = (1×1×1×1×1×1×1×1×1) × (2 × 2 × 2) × 3 = 24	(4)
k-ary	$n!_{<k>} = \prod_{i=1}^{n} i^{k^{n-i}}$	$n!_{<k>} = \begin{cases} n \times ((n-1)!_{<k>})^k & \text{if } n > 1 \\ 1 & \text{if } n = 1 \end{cases}$	$3!_{<k>} = \left(\underbrace{1 \times \cdots \times 1}_{k^2}\right) \times \left(\underbrace{2 \times \cdots \times 2}_{k}\right) \times 3$	(5)

The root of k-ary trees in Figure 6 has n choices but its k children nodes have n – 1 choices. The next level 2 has k^2 number of nodes which have n – 2 choices each. In general, the level l has k^l number of nodes which have n – l choices each. The k-ary tree factorial is the multiplication of all these choices in every node as defined in eqn (5) with examples of unary, binary, and ternary tree factorials in eqns (2), (3), and (4), respectively.

The *exclamation mark* was first used to alert rather alarming rate of growth of the $n!$ function by Christian Kramp in 1808 [6]. The growth rate of $n!$ is often described as 'astronomical' in [7] or 'maddening' and 'unimaginably large' in [8]. Yet, the binary tree factorial grows even faster than $n!$, e.g., while 9! is only 362880, 9!! is greater than *googol* as shown in Table 2. Albeit the symbol Δ was used to denote binary tree factorial earlier in [5], the double exclamation marks shall be used here so that the k-ary tree factorial can be expressed using k exclamation marks to alert their extremely daunting growth rates even more since $n! < n!! < n!!!$ for $n > 2$, as formally stated in the eqn (6).

$$n!_{<a>} < n!_{<b>} \text{ if } a < b \tag{6}$$

Table 2. *k*-ary tree factorial sequences.

	Unary	Binary	Ternary	Quaternary	Quinary	Senary	Septenary
k	1	2	3	4	5	6	7
n	$n!$	$n!!$	$n!!!$	$n!!!!$	$n!!!!!$	$n!!!!!!$	$n!!!!!!!$
1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	6	12	24	48	96	192	384
4	24	576	55296	21233664	32614907904	2.00e+000014	4.93e+0000018
5	120	1658880	8.45e+0014	1.02e+00030	1.85e+000053	3.24e+000086	3.51e+0000131
6	720	1.65e+013	3.63e+0045	6.40e+00120	1.28e+000267	6.91e+000519	3.96e+0000921
7	5040	1.91e+027	3.33e+0137	1.18e+00484	2.44e+001336	7.59e+003119	1.07e+0006452
8	40320	2.91e+055	2.97e+0413	1.54e+01937	6.90e+006682	1.53e+018720	1.25e+0045165
9	362880	7.64e+111	2.35e+1241	5.01e+07749	1.41e+033415	1.16e+112322	4.22e+0316156
10	3628800	5.84e+224	1.29e+3726	6.29e+30999	5.48e+167076	2.38e+673933	2.21e+2213098

The *k*-ary tree factorial functions generate *integer sequences* as first 10 sequences are listed in Table 2. The integer sequences generated by $n!!$ and $n!!!$ appear in the on-line encyclopedia of integer sequence [9 (A052129 and A123851)] and are called *Somo's quadratic* and *cubic recurrence sequences*, respectively. The generating functions were denoted as $g_{n,2}$ and $g_{n,3}$, accordingly in [10] rather than using the exclamation marks. In all, the *k*-ary tree factorial defined in this paper is the Somo's generalized recurrence sequence in [10].

Unfortunately, the *double-factorial* ($n!!$) in [9 (A006882), 11] and the *multi-factorial* ($n!^{(k)}$) in [9, 12] have different meanings in mathematics as defined in the eqns (7) and (9), respectively. The *double-factorial* is sometimes known as the *odd-factorial* which is the product of only odd numbers less than or equals n. According to the *multi-factorial* definition in [12], $n! > n!! > n!!!$ which is opposite to the *k*-ary tree factorial. This is somewhat unexpected from the terms 'double' and 'multi' and thus some revisions in the nomenclature and notations may be necessary for the multi factorial concept in [12].

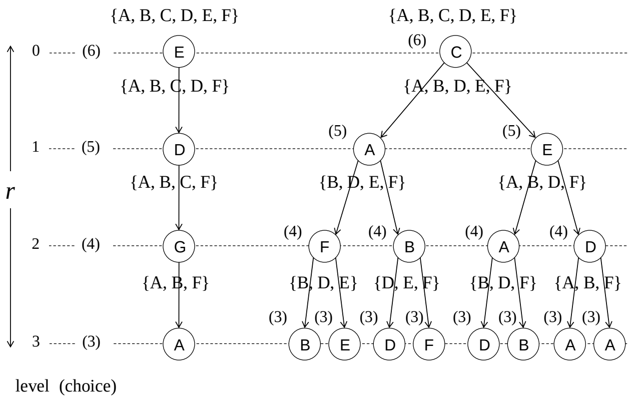
Table 3. *multi factorial* definition coined in [12].

Name	Recursive def.	example	
factorial	$n! = n \times (n-1)!$	7! = 1 × 2 × 3 × 4 × 5 × 6 × 7 = 5040 9! = 1 × 2 × 3 × 4 × 5 × 6 × 7 × 8 × 9 = 362880	
double factorial	$n!! = \begin{cases} 1 & 0 \le n < 2 \\ n \times (n-2)!! & n \ge 2 \end{cases}$	7!! = 1 × 3 × 5 × 7 = 105 9!! = 1 × 3 × 5 × 7 × 9 = 945 11!! = 1 × 3 × 5 × 7 × 9 × 11 = 10395	(7)
triple factorial	$n!!! = \begin{cases} 1 & 0 \le n < 3 \\ n \times (n-3)!!! & n \ge 3 \end{cases}$	7!!! = 1 × 4 × 7 = 28 9!!! = 1 × 3 × 6 × 9 = 162 11!!! = 1 × 2 × 5 × 8 × 11 = 440	(8)
multi factorial	$n!^{(k)} = \begin{cases} 1 & 0 \le n < k \\ n \times (n-k)!^{(k)} & n \ge k \end{cases}$	$51!^{(6)}$=51×45×39×33×27×21×15×9×3=226088287425 $51!^{(7)}$ = 51×44×37×30×23×16×9×2 = 8249662080 $51!^{(8)}$ = 51× 43× 35× 27× 19× 11× 3 = 433128465	(9)

3 *k*-ary Tree *r*-Permutation

Arranging only *r* distinct objects out of *n* different objects in an ordered fashion is known as *r-permutation* or simply *permutation* [13, 14], e.g., a number of ways to choose *r* examinations in order out of *n* examinations. The *r-permutation* is the *r*-th falling factorial power of *n* and has various notations as enumerated in the eqn (10). $P(n,r)$ is used here and its recursive definition is given in the eqn (11).

$$P(n,r) = {}_nP_r = {}^nP_r = P_r^n = n_{(r)} = n^{\underline{r}} = \frac{n!}{(n-r)!} = \prod_{i=n-r+1}^{n} i = \underbrace{(n-r+1)\times(n-r+2)\times\cdots\times(n-1)\times n}_{r} \quad (10)$$



(a) Unary tree (k = 1) (b) Binary tree (k = 2)

Figure 7. *k-ary tree permutation* where $n = 6$, $r = 4$.

Table 4. *k*-ary tree *r*-permutation definition.

Recursive definition	Prod. def.	examples	
$P(n,r) = \begin{cases} n \times P(n-1, r-1) & r > 1 \\ n & r = 1 \end{cases}$	$= \prod_{i=n-r+1}^{n} i$	$P(5,3) = 3 \times 4 \times 5 = 6$ $P(7,4) = 4 \times 5 \times 6 \times 7 = 840$ $P(8,3) = 6 \times 7 \times 8 = 336$	(11)
$P_2(n,r) = \begin{cases} n \times (P_2(n-1, r-1))^2 & r > 1 \\ n & r = 1 \end{cases}$	$= \prod_{i=n-r+1}^{n} i^{2^{n-i}}$	$P_2(5,3) = (3\times3\times3\times3) \times (4\times4) \times 5 = 6480$ $P_2(6,4) = (3\times3\times3\times3\times3\times3\times3\times3) \times (4\times4\times4\times4)\times(5\times5)\times6 = 251942400$	(12)
$P_k(n,r) = \begin{cases} n \times (P_k(n-1, r-1))^k & r > 1 \\ n & r = 1 \end{cases}$	$= \prod_{i=n-r+1}^{n} i^{k^{n-i}}$	$P_k(7,3) = \left(\underbrace{5 \times \cdots \times 5}_{k^2}\right) \times \left(\underbrace{6 \times \cdots \times 6}_{k}\right) \times 7$	(13)

The *k-ary tree r-permutation*, $P_k(n,r)$ is defined in the eqn (13) in a very similar manner to the *k*-ary factorial developed in the previous section. The unary tree *r*-permutation is the conventional *r*-permutation as defined in the eqn (11). Some exemplary values of the *k*-ary tree *r*-permutation are given in Table 5 at the end of this article.

It should be noted that the default bounding condition for r, i.e., $0 \le r \le n$, is omitted for the eqns (10~16). If $r > n$, $P_k(n,r)$ is always 0 and of no interest in this article. When $r = 0$, $P_k(n, 0)$ is always 1. If $r = 1$, $P_k(n, 1)$ is always n. When $r = n$ or $n - 1$, it is the same as $n!_{<k>}$ as stated in the eqn (14).

$$n!_{<k>} = P_k(n, n) = P_k(n, n-1) \tag{14}$$

For example, $\{P_2(3, 3) = 3!! = (1 \times 1 \times 1 \times 1) \times (2 \times 2) \times 3 = 12\} = \{P_2(3, 2) = (2 \times 2) \times 3 = 12\}$.

The *r*-permutation, $P(n,r)$ is occasionally defined only using the factorial function in many text-books [11, 12], i.e., $n!$ divided by $(n - r)!$ as given in the eqn (10). Although it should not be recommended to compute $P(n,r)$ in this way, the definition provides the good relationship between the *r*-permutation and factorial functions. Similarly, the *k*-ary tree *r*-permutation, $P_k(n,r)$ can be defined only using the *k*-ary tree factorial, $n!_{<k>}$ as in the eqn (15).

$$P_k(n,r) = \frac{n!_{<k>}}{\left((n-r)!_{<k>}\right)^{k^r}} \tag{15}$$

The eqn (15) means that branches of a *k*-ary tree factorial are trimmed so that the height of the tree becomes r. Hence, the *k*-ary tree factorial is the upper bound for the *k*-ary tree permutation as stated in the eqn (16). $r!_{<k>}$ is the lower bound.

$$r!_{<k>} \le P_k(n, r) \le n!_{<k>} \tag{16}$$

Table 6. binary tree r-permutation $P_2(18, r)$ sequences.

r	$P_2(18, r)$
2	5202
3	340918272
4	873736243200000000
5	1902956767242460454123156275200000000
6	84258979639877344609084780954599701246245641061904385341254533120000000
7	9.8450050364399438253064418944124484783806387225585235246332091787275633639784661636e+00140
8	1.9564990569385013412632794966199095369393233952682265711841202995960268333030141519e+00274
9	1.9564990569385013412632794966199095369393233952682265711841202995960268333030141519e+00530
10	7.3054079993032925450953755449075779598770192039437567455423404200339388683139604267e+01018
11	4.2441542111376058785045461754898671600136488463118516757027454036600691991448438101e+01943
12	2.4466796217854683088275995595833553640813053043984235283332165198750915602517838625e+03674
13	4.9670516516865948889839912675409718550407253471918135304423936346470409134756361684e+06861
14	4.5538025594386159954684720901082755066808882881861267591503750646712422166428292153e+12587
15	6.4457300659581374113731899549075344536423916237237467314258220425155780789478239226e+22451
16	1.3138529728867728342025683240792759268600456632395695199425018857651700299157910721e+38086
17	2.6323437553326643119570018398980278134510790650024734166115968939618177137281397163e+57814

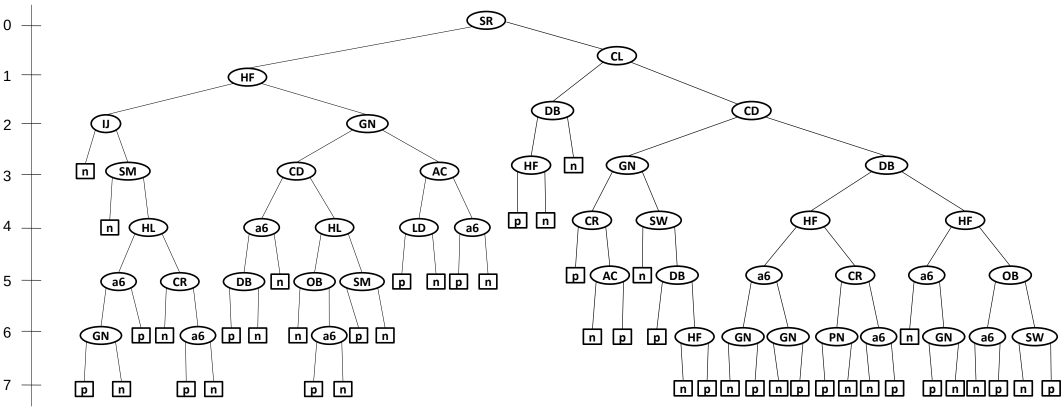


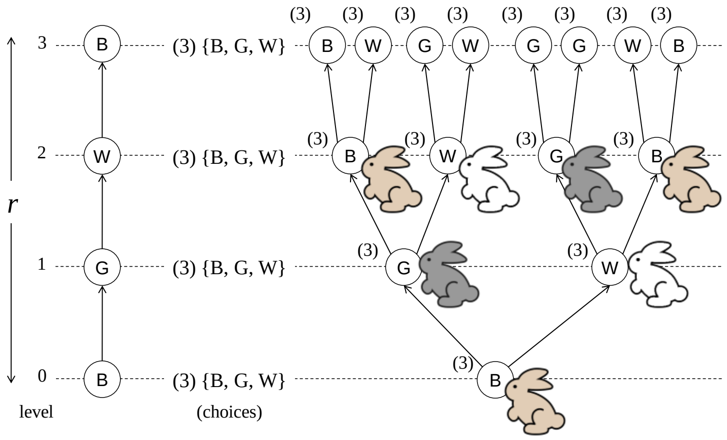
Figure 8. *DVT binary decision trees* where $n = 18$, $r = 7$ (see [10]).

Searching an optimal decision tree using the binary tree factorial and r-permutation concepts as encoding and decoding schemes in a *Genetic Algorithm* was proposed in [5]. It was successfully applied to find better Decision trees to predict the *Deep Venous Thrombosis* (*DVT*) in [15] than ones built by conventional heuristic algorithms found in [1,2]. Figure 8 shows the decision tree found by limiting its height to be 7. The main Motto of this article is to analyze the decision tree search space for [5,15]. Table 6 shows $P_2(18,r)$ values and $P_2(18,7)$ in Figure 8 is greater than the *googol*.

4 *k*-ary Tree *r*-Sequence

In *Combinatorics*, there are two ways to arrange r objects from n objects in order. One way is called 'r-permutation' if repetition is not allowed and the other is called '*r-permutation with repetition*' [14], '*r-string*' [13], or '*r-sequence*', e.g., the 5 digit zip-code ($n = 10$, $r = 5$) has 100000 possible sequences and the 8 bit binary code ($n = 2$, $r = 8$) has 256 possible sequences. The r-sequence denoted as $S(n,r)$ is defined in the eqn (17) and its recursive definition is given in the eqn (18).

$$S(n,r) = n^r = \prod_{i=1}^{r} n = \underbrace{n \times n \times \cdots \times n \times n}_{r} \tag{17}$$



(a) Unary tree (n = 3, r = 4)

(b) Binary tree (n = 3, r = 4)

Figure 9. *k-ary tree sequences* where $n = 3$, $r = 4$.

Table 7. k-ary tree r-sequence definition.

Recursive definition	Prod. def.	Examples	
$S(n,r)=\begin{cases} n\times S(n,r-1) & r>1 \\ n & r=1 \end{cases}$	$=\prod_{i=1}^{r} n$	$S(5,3) = 5 \times 5 \times 5 = 125$ $S(7,4) = 7 \times 7 \times 7 \times 7 = 2401$ $S(8,3) = 8 \times 8\times 8 = 512$	(18)
$S_2(n,r)=\begin{cases} n\times (S_2(n,r-1))^2 & r>1 \\ n & r=1 \end{cases}$	$=\prod_{i=1}^{r} n^{2^{i-1}}$	$S_2(5,3)= (5\times5\times5\times5) \times (5\times5) \times 5=78125$ $S_2(3,4) = (3\times3\times3\times3\times3\times3\times3\times3) \times (3\times3\times3\times3)\times(3\times3)\times3=14348907$	(19)
$S_k(n,r)=\begin{cases} n\times (S_k(n,r-1))^k & r>1 \\ n & r=1 \end{cases}$	$=\prod_{i=1}^{r} n^{k^{i-1}}$	$S_k(7,3)=\left(\underbrace{7\times\cdots\cdots\times 7}_{k^2}\right)\times\left(\underbrace{7\times\cdots\times 7}_{k}\right)\times 7$	(20)

The term 'k-ary tree r-sequence' as defined in the eqn (20) takes account of the fact that repetition is allowed in the k-ary tree structure and the unary and binary tree r-sequences are defined in the eqns (18) and (19), respectively. Suppose that a rabbit can be 'brown', 'gray', or 'white' which means n = 3. Let left and right branches be father and mother rabbits in Figure 7 (b). The family tree up to r ancestors forms a binary tree where repetition is allowed. The total number of this binary family trees is multiplying its possible value n = 3 for the number of nodes times as defined in the eqn (19).

Table 8 shows some binary tree sequence values, $S_2(n,r)$ and Table 9 shows some k-ary tree sequence values, $S_k(n,r)$. Be aware that $S_2(10, 333)$ and $S_3(10, 211)$ are greater than the *googolplex* = $S(10, \textit{googol})$. Note that $P_k(n, r) \leq S_k(n, r)$. Also, while $1 \leq r \leq$ n in $P_k(n, r)$, there is no upper bound, i.e., $r \geq 1$ in $S_k(n, r)$.

Table 8. binary tree r-sequences

r	n = 2	n = 3	n = 4	n = 5	n = 6	n = 7	n = 8	n = 9	n = 10
2	8	27	64	125	216	343	512	729	1000
3	128	2187	16384	78125	279936	823543	2097152	4782969	10000000
4	32728	14348907	1.07e+009	3.05e+010	4.70e+011	4.75e+012	3.52e+013	2.06e+014	1.0e+0015
5	2.15e+009	6.18e+014	4.61e+018	4.66e+021	1.33e+024	1.58e+026	9.90e+027	3.82e+029	1.0e+0031
6	9.22e+018	1.15e+030	8.51e+037	1.08e+044	1.06e+049	1.74e+053	7.85e+056	1.31e+060	1.0e+0063
7	1.70e+038	3.93e+060	2.90e+076	5.88e+088	6.69e+098	2.13e+107	4.93e+114	1.54e+121	1.0e+0127
8	5.79e+076	4.63e+121	3.35e+153	1.73e+178	2.68e+198	3.16e+215	1.94e+230	2.15e+243	1.0e+0255
9	6.70e+153	6.44e+243	4.49e+307	1.49e+357	4.32e+397	7.00e+431	3.01e+461	4.15e+487	1.0e+0511
10	8.99e+307	1.24e+488	8.08e+615	1.11e+715	1.12e+796	3.43e+864	7.26e+923	1.55e+976	1.0e+1023

5 Conclusion

This article has naturally extended the elementary combinatorial concepts for lists to embrace the k-ary tree structure. Three combinatorial formulae, namely *k-ary tree factorial*, *r-permutation*, and *r-sequence* were introduced. Using the exclamation marks and subscription, simple notations and formulae were provided in both product notation and recursive forms. Albeit prohibitive due to their enormously huge values in the past, it is time to make concise formulae and notations easily available since the k-ary tree structure prevails in computer science and pattern recognition.

Dealing with the k-ary tree combinatorics encounters functions that grow extremely fast. Assuming the multiplication between two integers takes constant time, the computational complexity $\Theta(r\log k)$. The complexity of the operation to multiply two big integers, however, grows quadratic to the length of big integer; it not only consumes quite a big memory space, but also takes a while to compute the exact value.

From the k-ary tree structure point of view, the *travelling salesman problem* (TSP) [8,13,14] is only an unary tree search problem. As the search space of possible k-ary tree is enormously big, the exhaustive search for optimal trees is seemingly impossible. Yet, studying their mathematical structures presented in this paper should provide good insights for designing better heuristic algorithms.

References

1. Duda, R. O., Hart, P. E., and Stork, D. G., *Pattern Classification*, 2nd ed. Wiley interscience, 2001
2. Mitchell, T. M., *Machine Learning*, McGraw-hill, 1997
3. Hyafil, L. and Rivest, R. L., "*Constructing optimal binary decision trees is NP-complete*," *Information Processing Letters*, vol. 5, No. 1, 15-17, 1976.
4. Bodlaender, L.H. and Zantema H., "*Finding Small Equivalent Decision Trees is Hard*," *International Journal of Foundations of Computer Science*, vol. 11, n. 2 World Scientific Publishing, 2000, pp 343-354
5. Cha, S.-H. and Tappert, C. C., "*A Genetic Algorithm for Constructing Compact Binary Decision Trees*," *Journal of Pattern Recognition Research*, vol 4 n. 1, 2009, pp 1-13
6. Higgins, P., *Number Story: From Counting to Cryptography*, New York: Copernicus, 2008
7. Mukerjee, R. and Wu, C. F. *A modern theory of factorial designs*, Springer, 2006
8. Michalewicz, Z. and Fogel, D. B. *How to Solve It: Modern Heuristics*, 2nd ed. Springer, 1998
9. Sloane, N. J. A., "The On-Line Encyclopedia of Integer Sequences," http://oeis.org/
10. Sondow, J. and Hadjicostas, P., "*The generalized-Euler-constant function γ(z) and a generalization of Somos's quadratic recurrence constant*," Journal of Mathematical Analysis and Applications, Vol. 332, No. 1, p. 292-314, 2007
11. Meserve, B. E. "*Double Factorials*." Amer. Math. Monthly 55, 425-426, 1948
12. Weisstein, E. W. "*Multifactorial*," From MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/Multifactorial.html
13. Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. *Introduction to Algorithms*, 2nd ed., MIT Press, 2001
14. Rosen, K. H., *Discrete Mathematics and Its Applications*, 5th ed., McGraw-hill, 2003
15. Nwosisi, C., Cha, S.-H., An, Y. J., Tappert, C. C., and Lipsitz, E., "*Constructing Binary Decision Trees for Predicting Deep Venous Thrombosis*," in Proceedings of *IEEE International Conference on Software Technology and Engineering* Oct. 3-5 2010, San Juan, Puerto Rico, vol 1, pp 121-124

Table 5. *k*-ary tree *r*-permutation sequences.

n	r	$P(n, r)$	$P_2(n, r)$	$P_3(n, r)$	$P_4(n, r)$	$P_5(n, r)$
2	1	2	2	2	2	2
3	1	3	3	3	3	3
	2	6	12	24	48	96
4	1	4	4	4	4	4
	2	12	36	108	324	972
	3	24	576	55296	21233664	32614907904
5	1	5	5	5	5	5
	2	20	80	320	1280	5120
	3	60	6480	6298560	55099802880	4338117680348160
	4	120	1658880	845378412871680	1.0164119622e+30	1.845e+53
6	1	6	6	6	6	6
	2	30	150	750	3750	18750
	3	120	38400	196608000	16106127360000	2.111062325330e+019
	4	360	251942400	1.4992534703e+21	5.530334892e+043	9.218449803981e+078
	5	720	16511297126400	3.6249724609e+45	6.403690314e+120	1.283527521277e+267
7	1	7	7	7	7	7
	2	42	252	1512	9072	54432
	3	210	157500	2953125000	1384277343750000	1.62220001221e+0022
	4	840	10321920000	5.319877060e+025	4.7104517101e+053	2.93496763397e+0097
	5	2520	4.443248104e+17	2.358974403e+064	6.5479260221e+175	4.66001716196e+0395
	6	5040	1.908360530e+27	3.334357599e+137	1.1771161658e+484	2.43850677226e+1336
8	1	8	8	8	8	8
	2	56	392	2744	19208	134456
	3	336	508032	27653197824	54187879102414848	3.82262280735e+0024
	4	1680	198450000000	2.060323792e+029	2.937530812e+0061	8.98694303073e+0111
	5	6720	8.523362599e+20	1.204466638e+078	3.938584892e+0215	1.74223433623e+0488
	6	20160	1.579396297e+36	1.050170155e+194	1.470635494e+0704	1.75803520468e+1979
	7	40320	2.913471929e+55	2.965695178e+413	1.535915359e+1937	6.89778760083e+6682
9	1	9	9	9	9	9
	2	72	576	4608	36864	294912
	3	504	1382976	185949421056	1.225098753e+0018	3.9549722876e+00026
	4	3024	2322868617216	1.90317442e+0032	7.759810618e+0067	7.3459800186e+00123
	5	15120	3.54441623e+023	7.87134490e+0088	6.701503245e+0246	5.2759716438e+00560
	6	60480	6.53829390e+042	1.57263101e+0235	2.165725211e+0863	1.4446922818e+02442
	7	181440	2.24504340e+073	1.04236909e+0583	4.209811826e+2817	1.5114032346e+09897
	8	362880	7.63948681e+111	2.34758887e+1241	5.008545719e+7749	1.4053729684e+33415
10	1	10	10	10	10	10
	2	90	810	7290	65610	590490
	3	720	3317760	978447237120	1.84675732e+00019	2.2308081476e+000028
	4	5040	19126226165760	6.42960794e+0034	2.25260161e+00073	9.6764759996e+000133
	5	30240	5.39571861e+025	6.89343639e+0097	3.62580538e+00272	2.1391868171e+000620
	6	151200	1.25628864e+048	4.87693342e+0267	2.01692119e+00988	4.0880129568e+002804
	7	604800	4.27492871e+086	3.88938114e+0706	2.19995303e+03454	6.2932757725e+012211
	8	1814400	5.04021986e+147	1.13256876e+1750	3.14087560e+11271	7.8868165054e+049486
	9	3628800	5.83617587e+224	1.29379696e+3725	6.29283826e+30999	5.4822391828e+167076
11	1	11	11	11	11	11
	2	110	1100	11000	110000	1100000
	3	990	7217100	4261625379000	2.0383222e+000020	2.2308081476e+000029
	4	7920	121082845593600	1.0303978e+00037	1.2794756e+000078	9.6764759996e+000142
	5	55440	4.02393780e+027	2.9237899e+00105	2.8322413e+000294	2.1391868171e+000670
	6	332640	3.20251573e+052	3.6032965e+00294	1.9011249e+001091	4.0880129568e+003102
	7	1663200	1.73608726e+097	1.2759486e+00804	1.8203228e+003954	6.2932757725e+014024
	8	6652800	2.01025170e+174	6.4719358e+02120	2.5765959e+013818	7.8868165054e+061060
	9	19958400	2.79441978e+296	1.5980355e+05251	1.0705221e+045088	5.4822391828e+247436
	10	39916800	3.74670437e+450	2.3822705e+11176	1.7249597e+124001	2.2308081476e+835385

Table 9. *k*-ary tree *r*-sequence sequences

n	r	$S(n, r)$	$S_2(n, r)$	$S_3(n, r)$	$S_4(n, r)$	$S_5(n, r)$
	1	2	2	2	2	2
	2	4	8	16	32	64
	3	8	128	8192	2097152	2147483648
2	4	16	32768	1.09951 e+012	3.86856262277e+0025	9.13438523331814324e+00046
	5	32	2147483648	2.65846e+036	4.47948948436e+0102	1.27182282121274076e+00235
	6	64	9.22337e+018	3.75767e+109	8.05274749371e+0410	6.65522991622385277e+01175
	7	128	1.70141e+038	1.06117e+329	8.41019994726e+1643	2.61123112179775877e+05879
	1	3	3	3	3	3
	2	9	27	81	243	729
	3	27	2187	1594323	10460353203	617673396283947
3	4	81	14348907	1.21577e+019	3.59175455477e+0040	2.69721605590607563e+00074
	5	243	6.17673e+014	5.39103e+057	4.99284241977e+0162	4.28252528734869621e+00372
	6	729	1.14456e+030	4.70042e+173	1.86428666159e+0651	4.32136484944284141e+01863
	7	2187	3.93006e+060	3.11553e+521	3.62386511316e+2605	4.52091071122319240e+09318
	1	4	4	4	4	4
	2	16	64	256	1024	4096
	3	64	16384	67108864	4398046511104	4611686018427387904
4	4	256	1073741824	1.20893e+024	1.49657767663e+0051	8.34369935906605501e+00093
	5	1024	4.61169e+018	7.06739e+072	2.00658260405e+0205	1.61753328855753515e+00470
	6	4096	8.50706e+037	1.41201e+219	6.48467421975e+0821	4.42920852378009504e+02351
	7	16384	2.89480e+076	1.12608e+658	7.07314631529e+3287	6.81852797144518169e+11758
	1	5	5	5	5	5
	2	25	125	625	3125	15625
	3	125	78125	1220703125	476837158203125	4656612873077392578125
5	4	625	30517578125	9.09495e+027	2.58493941423e+0059	1.09476442525376334e+00109
	5	3125	4.65661e+021	3.76158e+084	2.23239724860e+0238	7.86273043163712562e+00545
	6	15625	1.08420e+044	2.66122e+254	1.24181218991e+0954	1.50257769091078293e+02730
	7	78125	5.87747e+088	9.42355e+763	1.18903237292e+3817	3.82961121921497429e+13651
	1	6	6	6	6	6
	2	36	216	1296	7776	46656
	3	216	279936	13060694016	21936950640377856	1326443518324400147398656
6	4	1296	470184984576	1.33675e+031	1.38949274207e+0066	2.46374105121370607e+00121
	5	7776	1.32644e+024	1.43318e+094	2.23653851164e+0265	5.44661339287072210e+00607
	6	46656	1.05567e+049	1.76626e+283	1.50126297417e+1062	2.87596766249301836e+03039
	7	279936	6.68665e+098	3.30611e+850	3.04774301836e+4249	1.18051427480148401e+15198
	1	7	7	7	7	7
	2	49	343	2401	16807	117649
	3	343	823543	96889010407	558545864083284007	1.57775382034845807e+00026
7	4	2401	4.74756e+012	6.36681e+033	6.81292175541e+0071	6.84375166571500149e+00131
	5	16807	1.57775e+026	1.80660e+102	1.50810522587e+0288	1.05091901751082068e+00660
	6	117649	1.74251e+053	4.12750e+307	3.62096743861e+1153	8.97313694147133065e+03300
	7	823543	2.12545e+107	4.92218e+923	1.20336262266e+4615	4.07211016273917031e+16505
	1	8	8	8	8	8
	2	64	512	4096	32768	262144
	3	512	2097152	549755813888	9223372036854775808	9.90352031428304220e+00027
8	4	4096	3.51844e+013	1.32923e+036	5.78960446187e+0076	7.62145642166990291e+00140
	5	32768	9.90352e+027	1.87883e+109	8.98846567431e+0307	2.05721575045876663e+00705
	6	262144	7.84638e+056	5.30585e+328	5.22194440707e+1232	2.94774010726549765e+03527
	7	2097152	4.92525e+114	1.19497e+987	5.94865747679e+4931	1.78047524438861983e+17638
	1	9	9	9	9	9
	2	81	729	6561	59049	531441
	3	729	4782969	2541865828329	1.09418989132e+0020	3.81520424476945832e+00029
9	4	6561	2.05891e+014	1.47809e+0038	1.29007007817e+0081	7.27497445223752651e+00148
	5	59049	3.81520e+029	2.90632e+0115	2.49284754286e+0325	1.83400228367810330e+00745
	6	531441	1.31002e+060	2.20940e+0347	3.47556475658e+1302	1.86741941620001514e+03727
	7	4782969	1.54454e+121	9.70650e+1042	1.31323983584e+5211	2.04386336588525913e+18637