Partition and Delegate: A Computational Algorithm for Generating Nested Organizational Hierarchies with a Fixed Number of Tasks

Myong-Hun Chang
Department of Economics
Cleveland State University
Cleveland, OH 44115
216-687-4523, -9206 (Fax)
m.chang@csuohio.edu
http://academic.csuohio.edu/changm

June 22, 2007

Abstract

I study the structure of hierarchies that are feasible for an organization with a fixed number of tasks. A computational algorithm for randomly generating the hierarchies is presented. Using the computationally generated hierarchies, I explore various structural measures that charaterize them and study their relationships.

1 Introduction

Consider an organization whose operation can be decomposed into a fixed set of basic tasks. Suppose that each task is carried out by a single worker, but the workers are grouped into "departments," each of which is headed by a manager positioned one level above the workers. The departments may then be grouped into "divisions," with the division managers being still higher up in the hierarchy. Continuing up the hierarchy in this way, one eventually reaches the central HQs (with the CEO) at the top which overlooks the entire set of tasks through several layers of managers. No matter how the basic tasks (and the workers) are allocated to various departments, and departments to divisions, and so forth, most of the managerial organizations are characterized by multi-level hierarchies that consist of multiple layers of middle managers connecting the CEO at the top with the workers at the bottom. My main focus in this paper is on charaterizing the diverse ways in which such connections can be made, given a fixed number of tasks that the organization must carry out.

The underlying issue here is equivalent to that of generating and characterizing a population of single-rooted inverted trees which connect the root to a fixed number of distinct leaves via a group of intermediate nodes. In this paper, I first present the computational algorithm for generating random trees which have a single root and a fixed number of leaves. I then explore the structural properties of the trees thus generated. It should be noted that the present study is strictly limited to exploring the various tree-configurations that are technically feasible for a given number of leaves. It sets aside any consideration of economic forces that may be relevant in the context of organizational hierarchies. That issue is addressed in Chang (2007), using the hierarchies generated with the algorithm presented here.

2 Forests and Trees: The Algorithm

Consider a class of trees which have a single root and N distinct leaves. The leaves are distinct and are indexed by $k \in \{0, ..., N-1\}$. In between the root and the leaves are the intermediate nodes that connect the root and leaves. For any given value of N, there is then a fixed number of ways in which the root can be connected to the leaves. The set of all such trees can be called a *forest* of trees with N leaves. Let us denote by $F(k)$ the forest containing single-rooted trees with k leaves.

My objective is to computationally generate trees which populate this forest. This is done in a recursive manner as follows:

- Level 1 of the hierarchy: The top CEO is in charge of all N activities. He decides how to partition this set of N activities and give them to the next level managers.

- Level 2 and lower: If anyone at the given level is in charge of more than one (1) activity, then he is a manager who further partitions this set of activities and gives them to the next level managers. If anyone at this level is in charge

of exactly one (1) activity, then this person is the worker. The worker does not delegate the activity he is in charge of.

- When the set of activities for one or more managers in that rank (or level) is partitioned, the hierarchy grows by one level.
- This procedure continues until everyone at the given level is a worker. The hierarchy then stops growing.

The above procedure requires two algorithms for partitioning any given set of activities: 1) an algorithm for deciding how many subsets into which a given set of activities will be partitioned and 2) an algorithm for deciding which activity goes into which subset. Let me discuss these in sequence.

1. Given a set of activities, we first decide how many subsets we are going to end up with. For this we use the *Stirling number of the second kind* and the *Bell number* [Stanley, 2000]. The Stirling number of the second kind, $S(x, y)$, is the number of ways in which a set of x elements can be partitioned into y non-empty subsets with non-overlapping elements. The total number of possible partitions of a set containing exactly x elements is then called a *Bell number* and is denoted $B(x)$, where

$$B(x) = \sum_{y=1}^{x} S(x, y), x \geq 1.$$

Since I assume that a manager in the organization does not create a lower level manager who is the exact duplicate of himself, I rule out the possibility of the set of activities being partitioned into itself – i.e., $S(x, 1)$ is ruled out. Let us define a modified Bell number, $B'(x)$, where $B'(x) = B(x) - S(x, 1) = B(x) - 1$. From a given set of x activities, the probability that exactly y blocks (divisions) will be created is then

$$p(x, y) = \frac{S(x, y)}{B'(x)}.$$

Using these probabilities for all $y \in \{2, 3, ..., x\}$, we decide how many lower divisions will be created by the given manager with more than one task.

2. Once the number of divisions is determined, we now need to allocate the activities into these divisions so that each division will have at least one activity and a particular partition of the activities would be realized on the basis of the stochastic process that is consistent with that which determined the number of divisions in the first place. For this, we use the very definition of the *Stirling number of the second kind* which utilizes the following recurrence relationship:

$$S(x, y) = yS(x - 1, y) + S(x - 1, y - 1).$$

The intuition behind this recurrence relationship is simple. To obtain a partition of the set of x elements into y subsets, we can partition the set of the

first $x-1$ elements into y subsets and place the remaining xth element into any of these subsets in $yS(x-1,y)$ ways, or we can put the last element (xth element) of the set into a subset by itself and partition the remaining $x-1$ elements into $y-1$ subsets in $S(x-1,y-1)$ ways. The former happens with the probability of $\frac{yS(x-1,y)}{S(x,y)}$, while the latter happens with the probability of $\frac{S(x-1,y-1)}{S(x,y)}$. Computationally, we can accomplish the allocation task by using these probabilities in a recursive manner. [The pseudo code for this algorithm is provided in the Appendix.]

3 Characterizing the Trees in a Forest

It is straightforward to generate random hierarchies using the above algorithms. Figures 1-A through 1-E show sample sets of trees chosen from various forests. Figure 1-A is a tree from the forest containing trees with only two leaves, $F(2)$. Obviously, this forest consists of only one tree. A forest consisting of trees with three leaves, $F(3)$, is a bit larger. Figure 1-B shows two typical trees from this forest. Note that there are more trees in this forest than these two, since the leaves are distinct from one another – the exact grouping of the business units (workers) will matter. Some sample trees from the forests with $N \in \{4, 5, 12\}$ are captured in Figures 1-C through 1-E.

Even a forest of trees with a moderate number of leaves is likely to have a very large number of trees. What we need is a few structural measures that capture the overall shape of the hierarchies so that we can characterize the trees populating a given forest. Three measures are useful for this purposes:

1. d (depth of the hierarchy): This is the number of levels (ranks) in a given hierarchy. In our model, the minimum depth of any hierarchy is two and the maximum depth of a hierarchy with k activities is k.[1] Figure 1-B(a) shows a hierarchy with depth of 2, while 1-B(b) shows one with depth of 3.[2]

2. m (number of managers in the hierarchy): This is the total number of individuals in the hierarchy who *process* the information. This includes the CEO, but excludes the workers at the bottom level. The hierarchy in Figure 1-C(a) has one manager, the CEO, while the one in Figure 1-C(c) has two managers and the one in Figure 1-C(d) three managers.

[1] The maximum depth of a hierarchy is attained if each manager with n activities handles one activity himself and gives the remaining $(n-1)$ activities to one subordinate manager. If we start out with the CEO partitioning N initial activities and apply this rule recursively, we end up with a hierarchy of depth N.

[2] It is important to note the following peculiarity in my figures representing the trees here. A node in the tree represents a manager, if and only if the node is above the bottom level and has more than one branch below it. If the node is at the bottom level, he is a worker. If the node is above the bottom level but has exactly one subordinate, then this node is a *ghost* which is in place only to identify the level in the hierarchy. One may as well assume that the node does not exist in that case.

3. s (average span of control): Each manager in a given hierarchy has two or more subordinates. What is the average number of subordinates per manager in a given hierarchy? In Figure 1-C(a), the average span of control is four, since the CEO receives information from all four subordinates (workers) simultaneously. In contrast, the hierarchy in Figure 1-C(d) has three managers each with two subordinates and, hence, the average span of control is two. The average span of control in Figure 1-C(e) is 2.5 ($= \frac{2+3}{2}$).

As the hierarchies are selected randomly from a forest, one may be interested in finding out the proportions of the trees in that forest having particular values for these measures. To this end, I have run 100,000 replications of the code that generates random hierarchies with a fixed number of leaves. The exact values of d, m, and s for each generated hierarchy are then recorded. At the end of 100,000 replications, I counted the number of times that any specific value of these measures was observed. The results are reported in Figures 2-A and 2-B in the form of histograms. For four separate forests, $F(6)$, $F(10)$, $F(12)$, and $F(16)$, the first column of Figure 2-A capures the depth of the hierarchy, while the second column captures the number of managers. Likewise, Figure 2-B captures the average span of control.

The next study was done to identify any relationships between the three measures that are either inherent and specific to the trees in a given forest or common across all forests in some general way. For instance, one may ask if there is an inverse relationship between d and m: Do hierarchies with greater depth necessarily have more managers? What about between m and s? Does a manager in a hierarchy with more managers have a narrower span of control on average? Can we say that the average span of control is narrower in hierarchies with greater depths – i.e., Is there an inverse relationship between s and d? These issues are addressed in Figures 3-A and 3-B, where the (d, m, s) measures from the 100,000 replications are plotted one-on-one: $m - s$, $d - m$, and $d - s$. Some of these results are quite obvious and do not require elaboration, but it is, nevertheless, useful at this stage to present them formally since they offer indirect proof that the algorithm generating the trees is properly coded and the resulting outputs are reliable.

Take the case of $N = 10$ in Figure 3-A. We see that there is a clear inverse relationship between s and m, a somewhat positive relationship between m and d, and a negative relationship between s and d. These relationships hold true for all values of N considered here. Interestingly, the relationship between s and m is very clear, while those between m and d as well as s and d tend to be more approximate.

Note that there are many different ways (involving different numbers of managers) in which a hierarchy of depth d can be constructed. Likewise, the span of control can vary for a given depth of hierarchy, depending on how the managers are connected. For the purpose of expositional simplicity, let me define $\underline{m}(d)$ as the minimum number of managers required to generate a hierarchy of given depth d. Similarly, define $\overline{s}(d)$ as the maximum (average) span of control achievable in a hierarchy of depth d. After observing simulation outputs in Figure 3 for all different values of N, one can conclude the following:

Property 1: m and s are inversely related. The average span of control is wider (narrower) in hierarchies with smaller (greater) number of managers.

Property 2: $\underline{m}(d)$ is non-decreasing in d.

Property 3: $\overline{s}(d)$ is non-increasing in d.

To improve the reliability of these "conclusions," I have run one million replications for the case of $N = 16$: $1,000,000$ trees were randomly generated from the forest, $F(16)$. The results are reported in Figure 4 and they confirm all three properties identified above. The exact numerical values of the frequencies giving rise to the histograms in Figure 4 are reported in Table 1. It is interesting to note that 31 hierarchies out of 1 million generated entailed a 2-level organization with one CEO and no middle managers – the CEO connects directly to all 16 workers.

4 Concluding Remarks

A simple partition-and-delegate algorithm was developed to computationally generate the feasible set of hierarchies of varying structures for an organization facing a fixed number of tasks. The generated hierarchies were characterized using three measures – depth, number of managers, and average span of control. The underlying relationships between these measures were identified through an in-depth analysis of the various hierarchies that are feasible for an organization with a given number of tasks.

This algorithm is utilized in another research for generating feasible hierarchies for the purpose of evaluating their effectiveness in processing information and coordinating organizational decisions. The ultimate goal is to identify the optimal hierarchy for an organization. That step is taken in Chang (2007).

References

[1] Chang, Myong-Hun, "Recursively Nested Organizational Hierarchy: Information Processing and Coordination," Working Paper (2007).

[2] Stanley, Richard P., 2000, *Enumerative Combinatorics, Volume 1*, Second Edition, Cambridge University Press.

Appendix

The following pseudo code describes the algorithm with which I allocate a set of x distinct elements into y non-empty blocks:

alloc(x, y)

Step 1: Receive a set of x distinct elements, indexed $\{1, 2, ..., x\}$, which are to be allocated into y non-empty blocks, indexed $\{1, 2, ..., y\}$.

Step 2: Define $\Phi(x, y) \equiv \{0, 1, ..., S(x, y)\}$, where $S(x, y)$ is the Stirling number of the second kind.

Step 3: Take a random draw, ϕ, from $\Phi(x, y)$.

If $\phi \leq y \cdot S(x-1, y)$, then do:

- Assign "x" into any of the y blocks.
- Set $x = x - 1$.

Else do:

- Assign "x" into block "y".
- Set $x = x - 1$ and $y = y - 1$.

Step 4: Recursion.

If $x \geq y$ and $x > 0$, then call alloc(x, y).

Else, terminate.

Table 1: Frequencies for N=16

No. Levels	Frequencies
1	0
2	31
3	22, 945
4	675, 730
5	289, 466
6	11, 720
7	108
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0

No. Managers	Frequencies
1	31
2	0
3	3
4	208
5	3, 404
6	27, 658
7	117, 267
8	265, 136
9	321, 860
10	199, 813
11	57, 837
12	6, 548
13	235
14	0
15	0
16	0

Figure 1 - A : N = 2

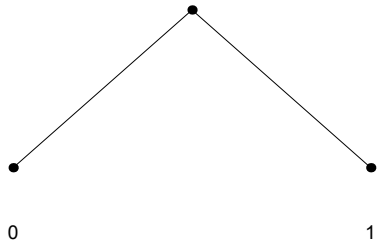


Figure 1 - B : N = 3

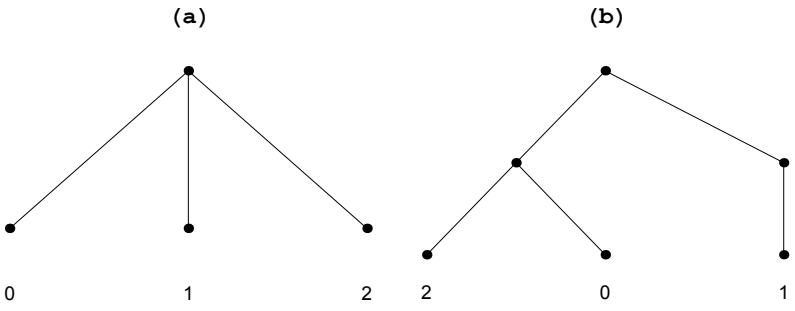


Figure 1 - C : N = 4

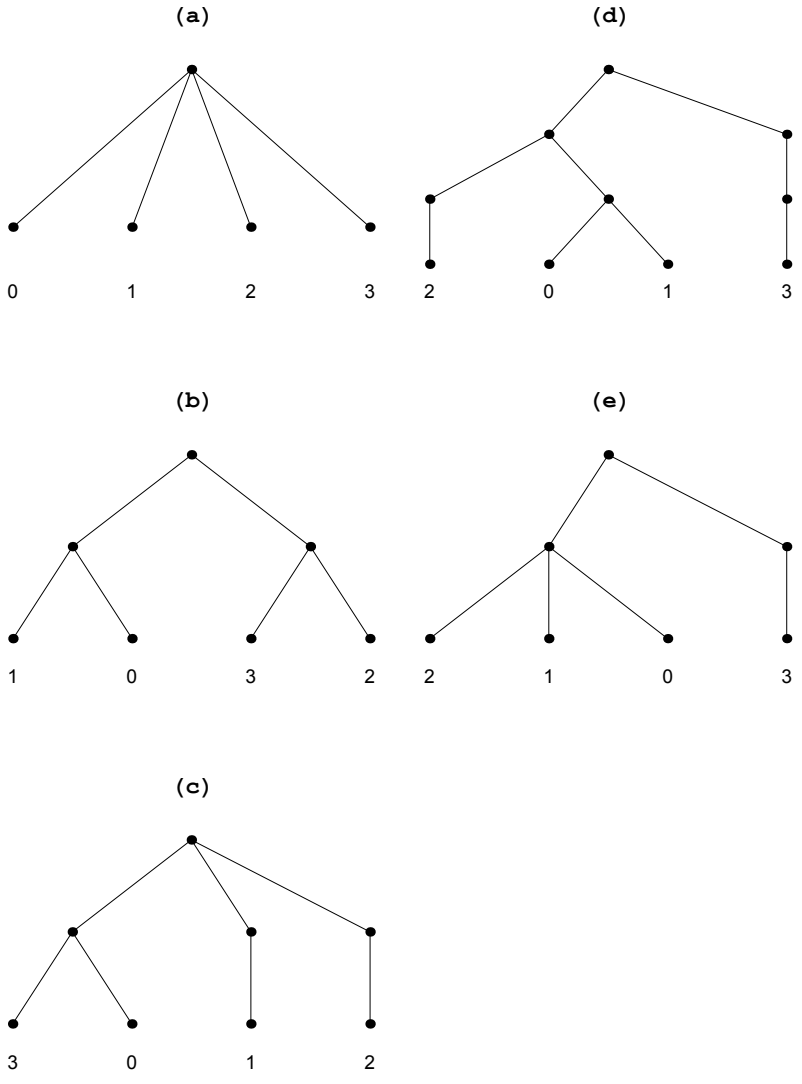


Figure 1 - D : N = 5

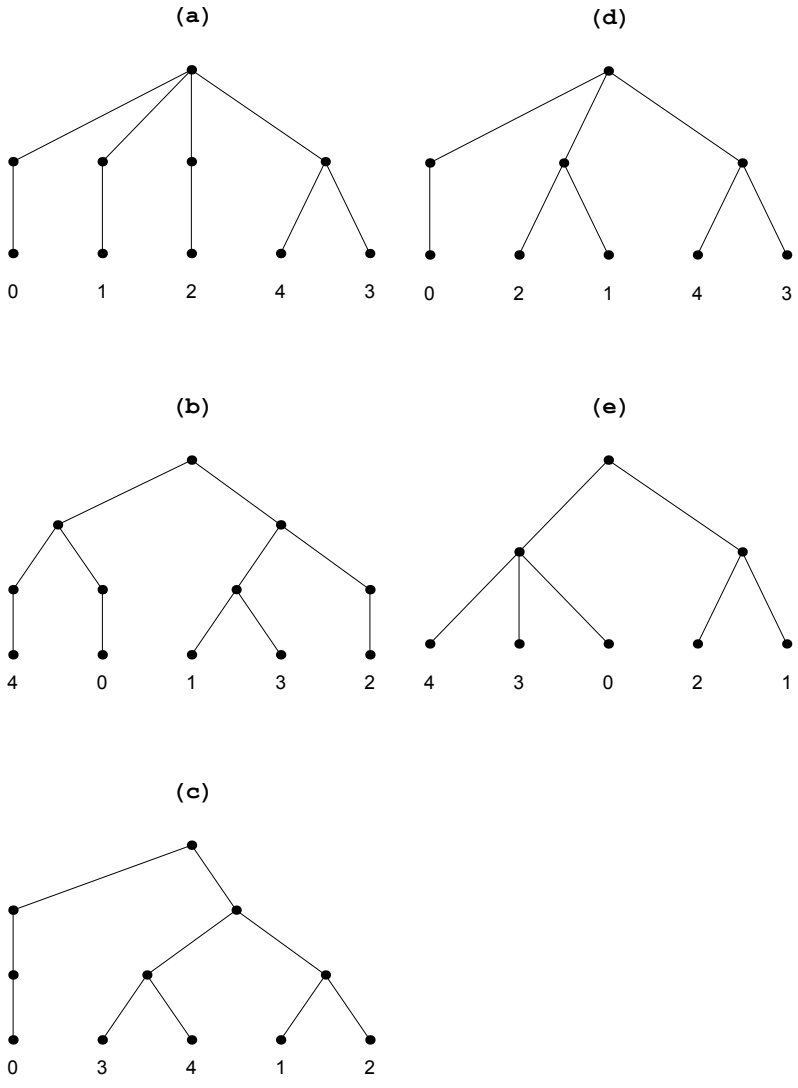


Figure 1 - E : N = 12

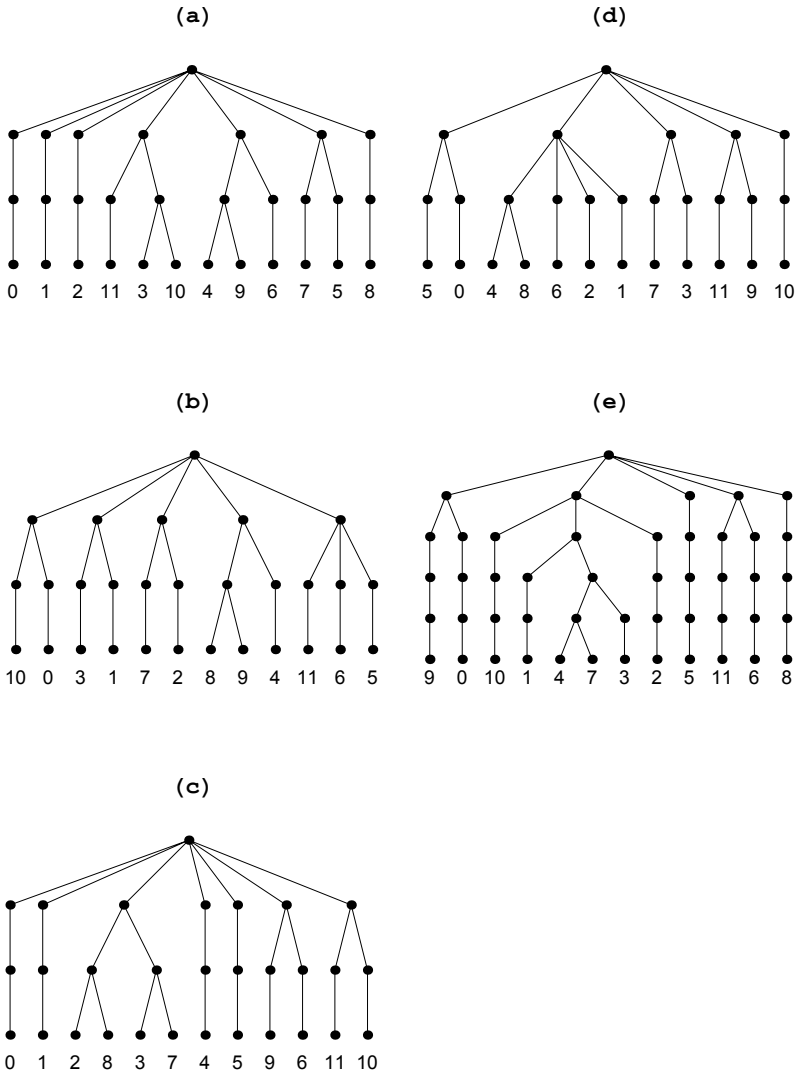
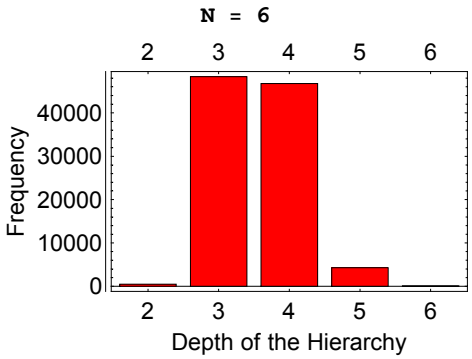
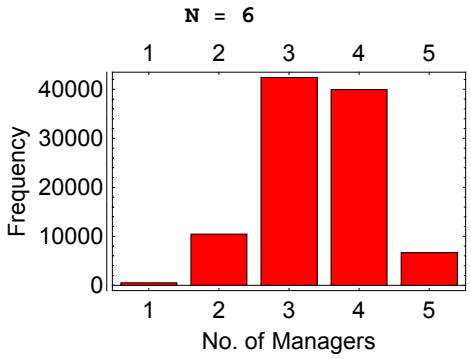
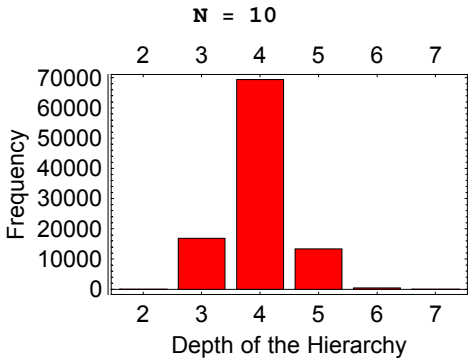
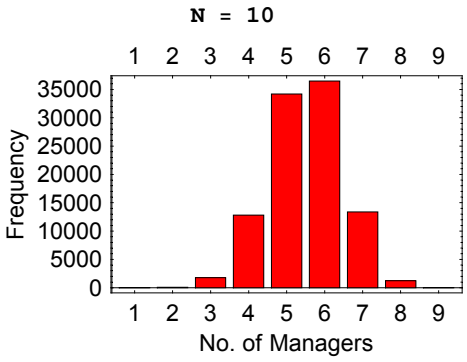


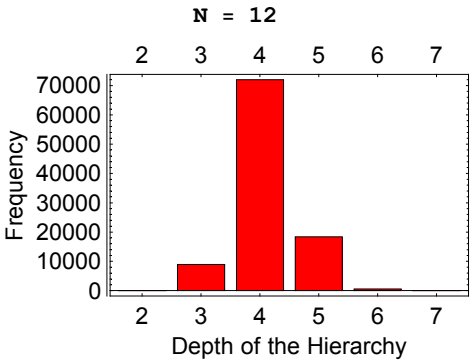
Figure 2 - A

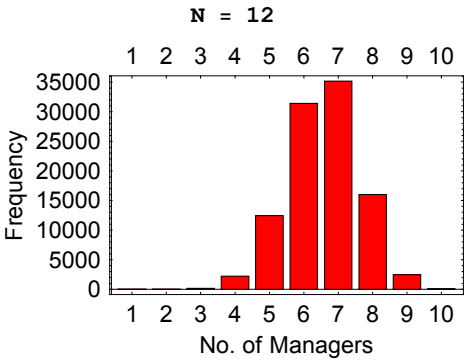

N = 6
Frequency
Depth of the Hierarchy

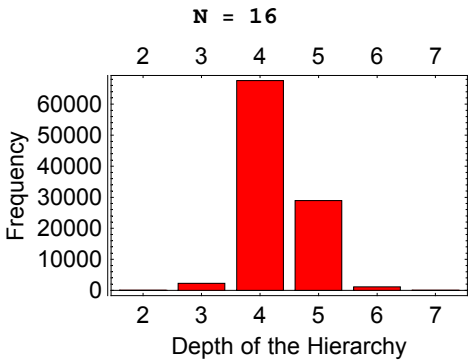

N = 6
Frequency
No. of Managers


N = 10
Frequency
Depth of the Hierarchy


N = 10
Frequency
No. of Managers


N = 12
Frequency
Depth of the Hierarchy


N = 12
Frequency
No. of Managers


N = 16
Frequency
Depth of the Hierarchy

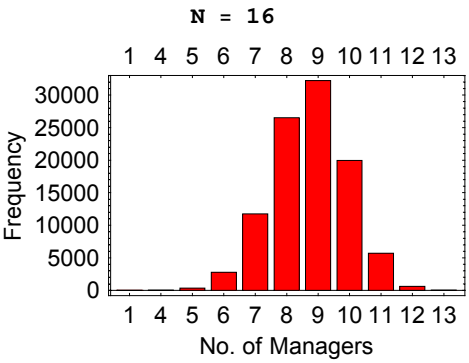
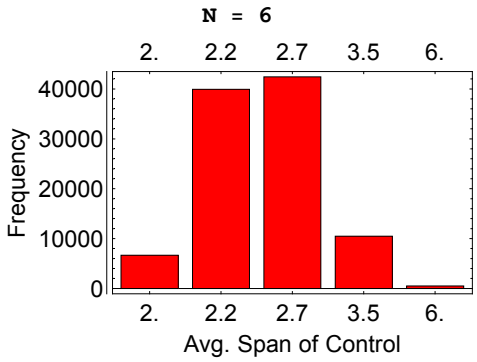
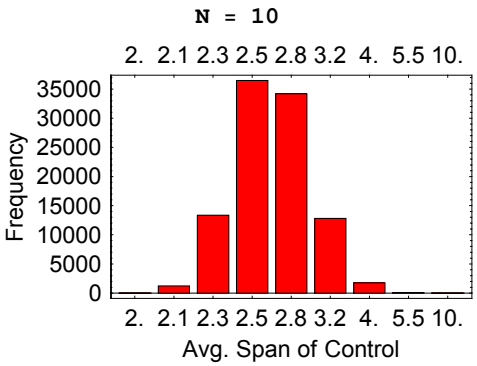
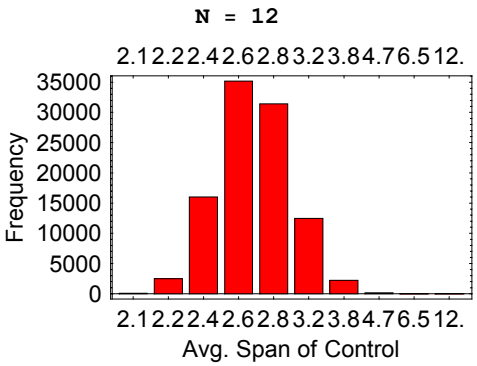

N = 16
Frequency
No. of Managers

Figure 2 - B


N = 6
2. 2.2 2.7 3.5 6.
40000
30000
20000
10000
0
Frequency
2. 2.2 2.7 3.5 6.
Avg. Span of Control


N = 10
2. 2.1 2.3 2.5 2.8 3.2 4. 5.5 10.
35000
30000
25000
20000
15000
10000
5000
0
Frequency
2. 2.1 2.3 2.5 2.8 3.2 4. 5.5 10.
Avg. Span of Control


N = 12
2.1 2.2 2.4 2.6 2.8 3.2 3.8 4.7 6.5 12.
35000
30000
25000
20000
15000
10000
5000
0
Frequency
2.1 2.2 2.4 2.6 2.8 3.2 3.8 4.7 6.5 12.
Avg. Span of Control

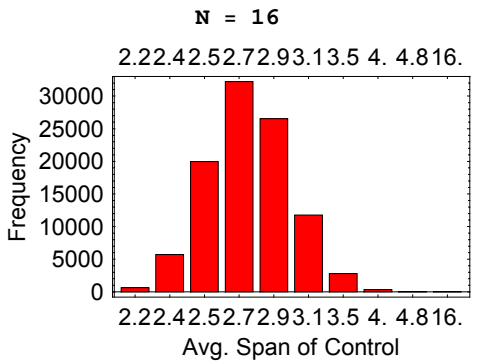

N = 16
2.2 2.4 2.5 2.7 2.9 3.1 3.5 4. 4.8 16.
30000
25000
20000
15000
10000
5000
0
Frequency
2.2 2.4 2.5 2.7 2.9 3.1 3.5 4. 4.8 16.
Avg. Span of Control

Figure 3 - A

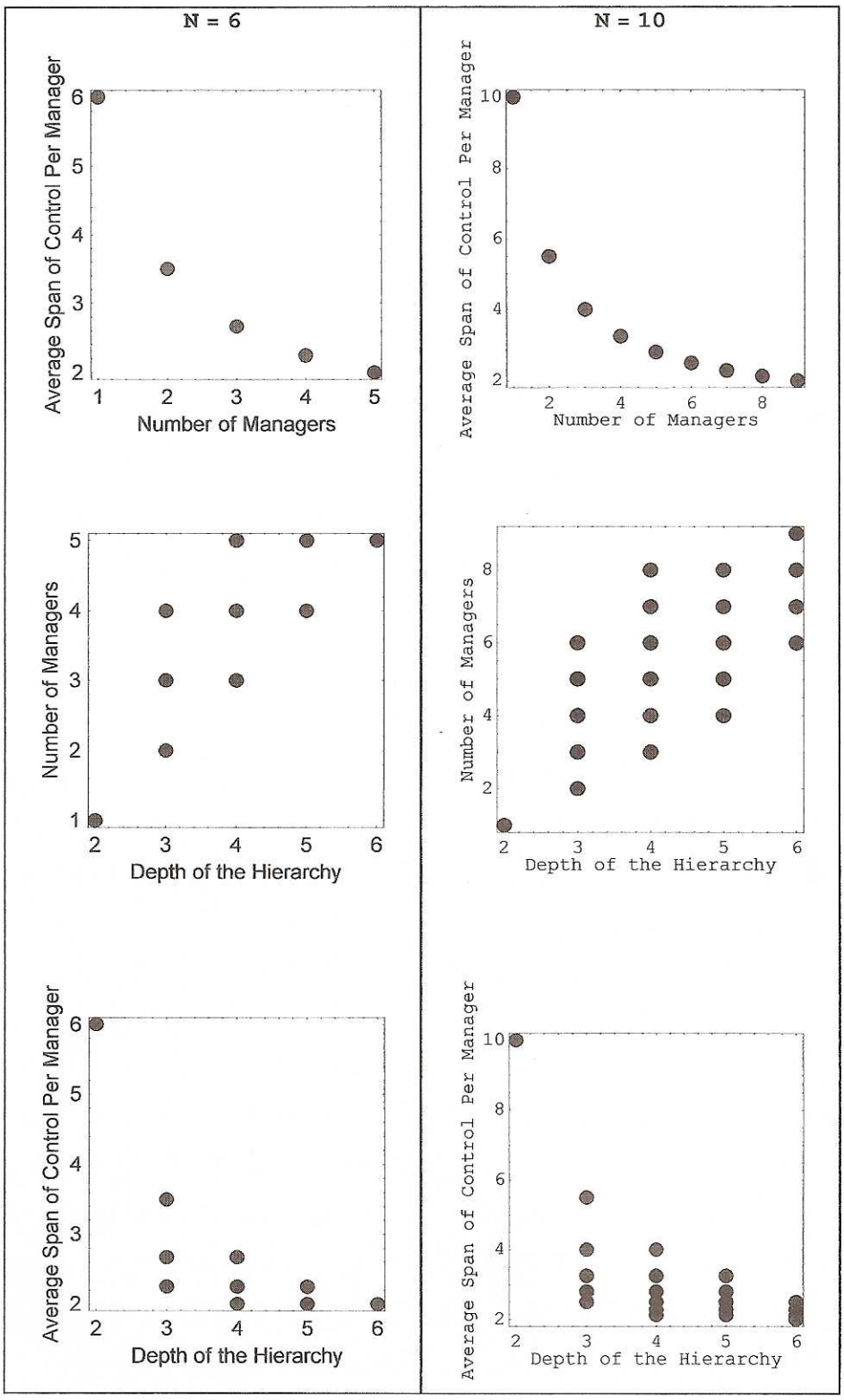

N = 6
N = 10
Average Span of Control Per Manager
Number of Managers
Number of Managers
Depth of the Hierarchy
Average Span of Control Per Manager
Number of Managers
Depth of the Hierarchy
Average Span of Control Per Manager
Depth of the Hierarchy

Figure 3 - B

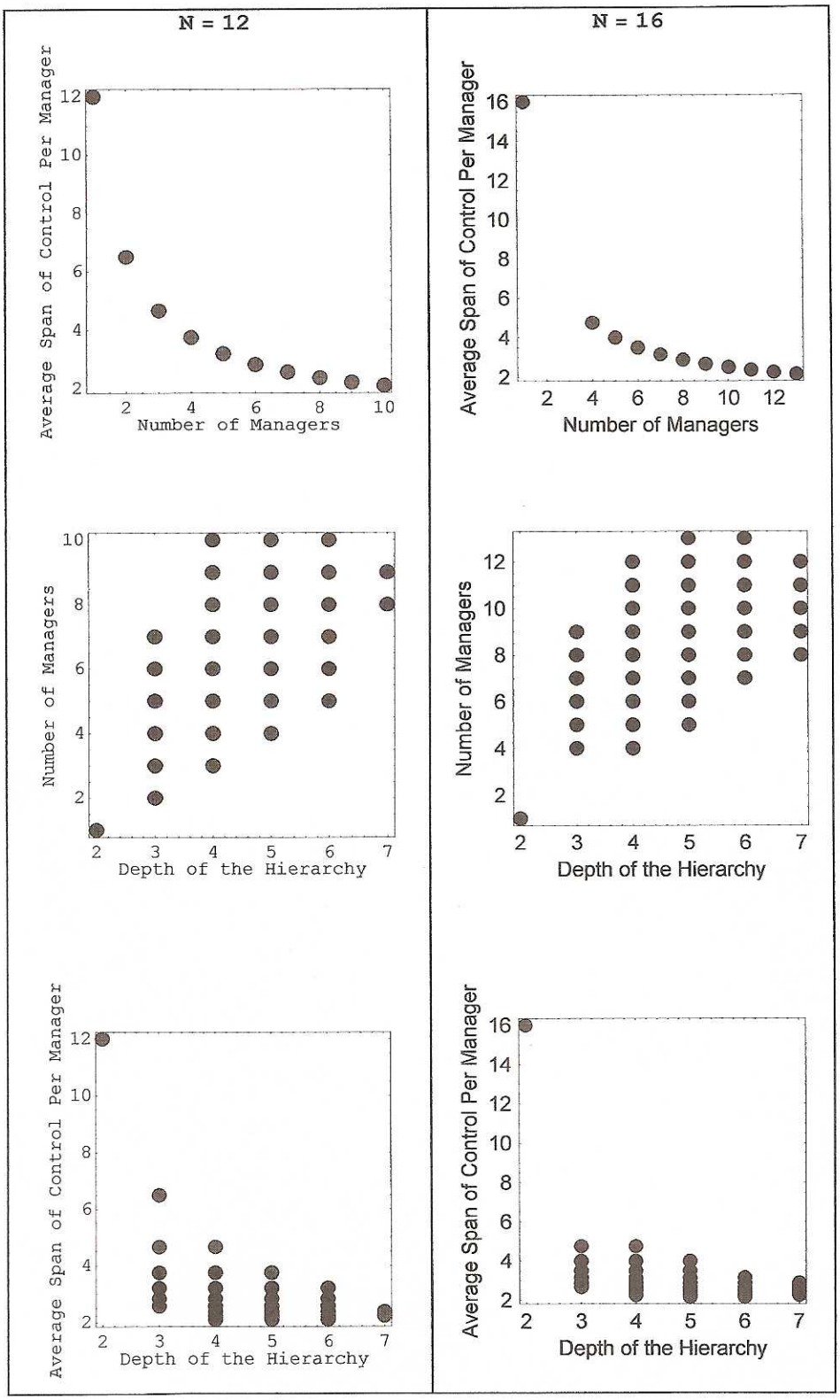

N = 12
N = 16
Average Span of Control Per Manager
Number of Managers
Number of Managers
Depth of the Hierarchy
Number of Managers
Depth of the Hierarchy
Average Span of Control Per Manager
Depth of the Hierarchy
Average Span of Control Per Manager
Depth of the Hierarchy

Figure 4 : N = 16 (1 mil. random hierarchies)