How Deep Should It Be? On the Optimality of Hierarchical Architectures

Amihai Motro[1], Alessandro D'Atri[2], and Eli Gafni[3]

[1] Information and Software Engineering Department
George Mason University, Fairfax, VA 22030, USA
ami@gmu.edu
[2] Centro di Ricerca sui Sistemi Informativi
LUISS "Guido Carli" University, Via Tommasini 1, I-00162 Rome, Italy
datri@luiss.it
[3] Computer Science Department
University of California, Los Angeles, CA 90095, USA
eli@cs.ucla.edu

Abstract. Many areas of information technology implement *hierarchical architectures*. Notable examples are the organization of computer files in folders, the arrangement of program menus, and the distribution of messages from a source to its clients. In each case, one must address the issue of the *optimal configuration* of the hierarchy: Assuming a given number of items, how to choose optimally the number of levels in the hierarchy (and thus the number of items at each level). Without loss of generality, we formalize this problem in the domain of assembly or manufacturing. We consider the process of manufacturing a product from a given number of elementary components. By assembling intermediate products, the target product can be manufactured in a variety of processes, each modeled by a tree. We are interested in *manufacturing turnaround*: the time between receiving an order at the root and its completion. We express the turnaround time of each manufacturing process (tree) with a formula that incorporates three parameters: the time required to *create* elementary components, the time required to *assemble* a product from its components and the time required to *deliver* the product to its procurer (another manufacturer). We show that this turnaround formula is optimized in a manufacturing process that corresponds to a *perfect* (or *nearly perfect*) tree. Somewhat surprisingly, the degree of the optimal tree (i.e., the ideal number of components in each sub-assembly) is shown to be independent of the number of elementary components, suggesting that in each manufacturing environment there is an ideal assembly size, which is optimal for the manufacturing of products of any scale.

1 Introduction

Hierarchy theory suggests that hierarchical organization is a preferred approach to many complex systems [3]. In particular, it is common to model complex

O. Etzion, T. Kuflik, and A. Motro (Eds.): NGITS 2006, LNCS 4032, pp. 260–273, 2006.

manufacturing and assembly processes with trees. In these trees, each node is a manufacturing or assembly step and an edge between two nodes indicates that the product represented by the lower level node is used in the manufacturing of the product represented by the higher level node. A leaf node corresponds to the manufacturing of an *elementary product*, a product that incorporates no other products (a product made from scratch).[1] An internal node corresponds to the manufacturing of a *composite* product, a product that incorporates the products modeled by its children nodes; products modeled by internal nodes are procured by and delivered to other manufacturers. The root node corresponds to the manufacturing of the *final* product, which is composite as well.

Obviously, each manufacturing tree has a prescribed number of leaves (the elementary products required to create the final product), and a single root node where the final assembly occurs. Other than that, the structure is flexible. A shallow manufacturing tree implies few intermediate manufacturing steps, each involving a large number of components. Conversely, a deep manufacturing tree implies many intermediate manufacturing steps, each involving a small number of components.

We are interested in *manufacturing turnaround*: the time between receiving an order at the root and its completion. We identify three major time components in any manufacturing process: (1) The time required to create an elementary product from scratch; it is assumed to be the same for all elementary products; (2) the time required to *assemble* a product from a given set of component products; it is a monotonic function of the number of components used; and (3) the time required to *deliver* a product from one manufacturer to another; it is assumed to be fixed for all products. The last two times are at odds with each other. High assembly times give advantage to assemblies that involve few components (and hence to deep manufacturing trees); high delivery times give advantage to manufacturing processes that involve few intermediate components (and hence to shallow manufacturing trees). Based on these time components we define a turnaround function for manufacturing trees.

The subject of this paper is finding the *optimal manufacturing tree* (from the perspective of turnaround) for a given product in a given manufacturing environment. More specifically, given the times of creation, assembly and delivery, what are the optimal assembly *breadth* (number of components per intermediate products) and *depth* (number of levels in the manufacturing hierarchy) to manufacture a product that incorporates a given number of elementary components.

Although stated here in terms of manufacturing or assembly processes, the problem is more general. It applies in any architecture that incorporates a set of items in a tree-like hierarchy, with an intrinsic conflict between the costs associated with the depth and the breath of the hierarchy. We mention here three different examples.

File and menu organization, taxonomies, hierarchical clustering. The ideal organization of a set of files in a folder hierarchy is easily formulated in

[1] Additionally, in a manufacturing environment that imports some of its components, an imported component is considered elementary as well.

term of the manufacturing trees discussed in this paper. Locating a file in a folder hierarchy involves two times: the time to search a folder for the correct item (equivalent of "assembly time"), and the time to open a subfolder and display its content (equivalent of "delivery time"). High search times reflect the difficulty in searching densely-populated folders; they encourage small folders and deep hierarchies. High traversal times reflect the overhead of changing folders and refreshing the display; they encourage large folders and shallow hierarchies. The overall cost associated with a file hierarchy is measured by the costliest path from the root folder to a file. A well-known issue which is equivalent to file organization is the organization of menus and sub-menus in user interfaces [2]. Similar problems also exist in taxonomies and hierarchical clustering.

Message distribution. Many systems for distribution messages or packets assume a single message source and a hierarchy of "switches" to forward the message to a set of clients. A switch that forwards a message to a large number of recipients incurs higher costs (for example, the message may spend more time in the switch, or costlier hardware may be required); on the other hand, each level in the distribution hierarchy incurs additional delay and possibly decreased reliability. The overall cost to be associated with a distribution system measures the costliest path from the source to a client, and the optimization finds the ideal number of "ports" on every switch. Similar situations often exist in systems that distribute actual goods.

Organizational hierarchy. Large enterprises are often concerned with the efficiency of their organization. The "leaves" of an enterprise are its indivisible operational units, and the issue is the proper level of branching in intermediate organizational units (often referred to as *span of control* [1]) and the proper length of the chain of command from the command center to the basic operational units (often referred to as *levels of management* [5]). Each level in the command incurs delays, bureaucratic overhead, increased inefficiency, and potential loss of control. On the other hand, intermediate units with large numbers of subordinates incur other types of inefficiencies, backlogs, and various managerial problems. The optimal organizational hierarchy states, for a given number of basic operational units, the ideal size of intermediate units and hence the overall depth of the organization. Quantifying the costs of "nodes" and "edges" undoubtedly presents a challenge.

In Section 3 we show that for every product there is an optimal manufacturing tree which is *perfect* or *nearly perfect*. Perfect trees are trees that are both balanced (all leaves are equal distance from the root) and full (all non-leaf nodes have the same degree d). Nearly perfect trees are balanced and full, but their non-leaf nodes are of two consecutive degrees (d and $d + 1$), with nodes at the same level having the same degree.

In a given manufacturing environment, a product with a given number of elementary components may be assembled by different perfect or nearly perfect trees, each with its own degree. The search for an optimal manufacturing process has therefore become a search for the optimal degree (assembly size). In Section 4 we show how to select the optimal tree among these alternatives. We also show

that the degree of the optimal tree depends only on the ratio of the assembly and delivery times; surprisingly, it does not depend on the number of leaves (elementary components).

We begin with a description of the manufacturing model, and a brief review of tree terminology.

2 Modeling Manufacturing Processes with Trees

We assume that manufacturing processes are modeled with trees, as described in the introduction. In practice, a manufacturing step may require only a single component; i.e., it may simply refine and add value to one component. However, we assume here that each manufacturing step requires at least two components. Without this assumption, the depth of the manufacturing tree for a product of n elementary components is not bounded, as it could involve an unlimited number of such single component refinements.

Denoting with n the number of elementary components, at one extreme is a tree of depth 1, that models a manufacturing process in which the final product is manufactured from its n elementary components in a single manufacturing step. At the other extreme is a tree of depth $\lceil \log_2 n \rceil$, that models a process in which each manufacturing step combines exactly two components (the required minimal number) in a new product.

2.1 The Cost Model

In comparing these alternative processes for manufacturing a specific product from its n elementary components, our focus is on the *time* it takes to manufacture (deliver) the final product. This turnaround time is composed of three different times:

1. **Creation time.** This is the time spent at leaf nodes. It models the time required to create an elementary product from scratch.
2. **Assembly time.** This is the time spent at each non-leaf node. It models the time required to assemble a product from its components.
3. **Delivery time.** This is the a time spent at each edge. It models the time required to deliver an intermediate product from one manufacturer to another.

We assume that assembly time increases with the number of components (the number of children nodes). This monotonicity assumption models the indisputable fact that assembling a large number of components requires more time than assembling a small number of components. This is true whether the components are Lego bricks or vegetables to be peeled and chopped for a stew.

Specifically, we assume that assembly time is a *convex* function of the number of components. That is, denote the number of components x and the assembly time $f(x)$, then for any two points x_1, x_2 and $0 \leqslant \alpha \leqslant 1$, $f(\alpha \cdot x_1 + (1-\alpha) \cdot x_2) \leqslant \alpha \cdot f(x_1) + (1-\alpha) \cdot f(x_2)$. Intuitively, a convex function is a continuous function

whose value at the midpoint of every interval does not exceed the average of its values at the ends of the interval. Linear or quadratic functions are convex. We assume that the same function f is associated with every non-leaf node in the manufacturing tree.

An example of linear assembly time is when each of the x components in the assembly requires a single action, and the time for each action is a. Then $f(x) = a \cdot x$. An example of quadratic assembly time is when each *pair* of the x components requires an action and the time for each action is a. Then $f(x) = a \cdot x^2$.

Our manufacturers do not keep any stock of components necessary for manufacturing. If a manufacturer receives a request for a product, it must order the components needed to manufacture that product.[2] Therefore, the use of intermediate products is more time-consuming as it initiates a chain of orders that terminates in the elementary products. We assume that all delivery times are identical; that is, a single time b is associated with every edge in the manufacturing tree.

Whereas assembly times encourage deep manufacturing trees with few components per product, delivery times encourage the opposite: shallow manufacturing trees with many components per product. Finally, we assume that all creation times are identical; that is, the same time c is required to create each of the n elementary components.

Consider now a path in the manufacturing tree, from the root to a leaf node. Denote this path $v_0, v_1, \ldots, v_h$, where v_0 is the root and v_h is a leaf. Let d_i denote the number of children of node v_i $(0 \leqslant i \leqslant h-1)$. We associate an overall time with this manufacturing path: the creation time at the leaf plus the assembly times at each intermediate node, plus the delivery times along the edges of the path, plus the final assembly time at the root.

Definition 1. The *lag of a manufacturing path* $v_0, v_1, \ldots, v_h$ from the root v_0 to a leaf v_h is

$$\sum_{i=0}^{h-1} f(d_i) + h \cdot b + c$$

Since manufacturing begins at the leaves and proceeds along multiple paths simultaneously towards the root, completion time is governed by the slowest manufacturing path.

Definition 2. The *turnaround of a manufacturing tree* is the highest lag of any of its manufacturing paths.

In calculating turnaround, we only measure the time for assembling and delivering products in the upward direction, ignoring the time required to propagate orders in the downward direction. Clearly, this should not impact the search for optimal trees; as ordering time along an edge may be assumed to be embedded in the delivery time b.

[2] Such recursive processes are sometimes referred to as *lazy*.

2.2 Tree Terminology

The terminology we use is standard (for example, [4]). The *length* of a path in a tree is the number of edges in the path. The *height* of a tree is the length of its longest path from the root (thus the height of a tree which is only a single node is 0, and the height of tree which is a root node connected to a set of leaf nodes is 1). The *level* of a node is the length of the path from that node to the root (thus the level of the root is 0).

A tree is *balanced* if all the paths from the root to a leaf have the same length. A tree is *full (of degree d)* if all its non-leaf nodes have d children. Balanced and full binary trees are sometimes referred to as *perfect* trees; hence, we refer to balanced and full trees (of degree d) as *perfect* trees (of degree d). Figure 1 shows a tree which is balanced but not full (a), a tree which is full (of degree 3) but not balanced (b), and a tree which is perfect (of degree 3) (c).

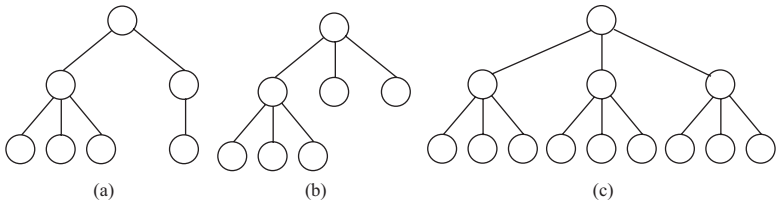

Fig. 1. Balanced, full and perfect trees

When a tree is perfect, its height h, degree d, and number of leaves n maintain: $n = d^h$. Or, alternatively, $h = \log_d n$ or $d = n^{1/h}$.

In general, when n and h are given, there may not be an integer d that satisfies this relationship. In such cases we approximate a perfect tree with a tree of the following properties:

1. The degree of non-leaf nodes is either d or $d + 1$.
2. Nodes at the same level have the same degree.

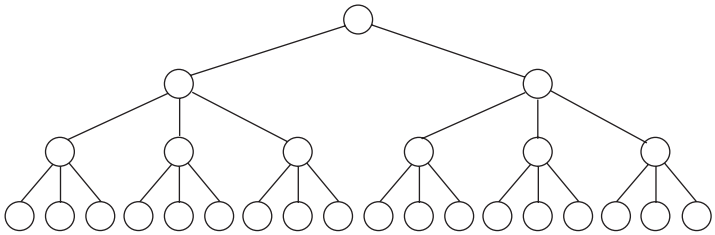

Fig. 2. Nearly perfect tree of degree 2-3

We refer to such trees as *nearly perfect* trees of degree $d-(d+1)$. Figure 2 shows a nearly perfect tree of degree 2-3. Note that the second property implies that the tree is balanced. Also note that there are no conditions on the number of levels of each type, or their order.

3 There Is Always a (Nearly) Perfect Optimal Tree

Let T be a manufacturing tree in the manufacturing environment defined earlier. Let h be its height, let n be its number of leaves, and let d_i be the degree of node v_i. Recall that in this environment, the time spent at a non-leaf node v_i is a convex function $f(d_i)$, the time spent at every edge is b, and the time spent at every leaf node is c. The lag of a path $v_0, v_1, \ldots, v_h$ from the root v_0 to a leaf v_h is $\sum_{i=0}^{h-1} f(d_i) + h \cdot b + c$, and the turnaround of T is the highest lag of any of its root-to-leaf paths.

In a manufacturing environment with costs f, b and c, consider a product that requires n elementary components. Let T be a manufacturing tree that optimizes the turnaround time, and let ϵ denote the optimal turnaround time. That is, the slowest manufacturing path has lag ϵ. Note that there could be several root-to-leaf paths of this lag.

Consider the top-level subtrees of T (the subtrees rooted at level 1 nodes). Let T_s be the top-level subtree with the most leaves (if several subtrees share this distinction, then one of these is chosen at random, and if all subtrees have the same number of leaves, then a subtree is chosen at random).

We observe that T_s includes a root-to-leaf path whose lag is ϵ. Otherwise, we would create a new tree by replacing every top-level subtree of T with T_s. The number of leaves of the new tree would be at least n, but its turnaround would be strictly lower than ϵ, in contradiction with the optimality of T.

We now create a new tree T' by replacing every top-level subtree of T with T_s. Since T_s includes a root-to-leaf path of lag ϵ, the new tree retains the same turnaround ϵ. Since T_s has the most leaves, the number of leaves of the new tree is at least n (the number of leaves is n only if all top-level subtrees of T have the same number of leaves).

Consider now the identical subtrees of T'. Repeating the same construction, we replace each of these subtrees with a tree whose top-level subtrees are identical. Again, the new tree has at least as many leaves as the old tree, but it retains the same turnaround. We work our way down in this way until level $h-1$ subtrees (in these final subtrees the roots are directly connected to the leaves).

Eventually, T is transformed to a new tree T^*, in which all the subtrees that are rooted at the same level are identical. Consequently, the degrees of nodes at the same level are identical, and hence all root-to-leaf paths have the very same length and lag. Note that the turnaround of T^* remains ϵ, and its number of leaves is at least n.

Recall that in T^*, nodes at the same level have the same degree. Let d_i denote the degree of nodes at level i ($0 \leqslant i \leqslant h-1$; we ignore the leaf level, whose

degree is $d_h = 0$). Assume that among these, there are degrees that are different by more than 1; i.e., assume the tree has levels i and j, such as $d_i - d_j > 1$.

We transform T^* to another tree. The new tree is identical to T^*, except that we replace the degrees at level i and j with new degrees $d'_i = d_i - 1$ and $d'_j = j + 1$ (if $d_i - d_j = 2$, then $d'_i = d'_j$). The following inequalities are easy to verify:

1. $f(d'_i) + f(d'_j) \leqslant f(d_i) + f(d_j)$
2. $d'_i \cdot d'_j > d_i \cdot d_j$

The first inequality is due to the convexity of f and it implies that the turnaround of the new tree is at least as good as that of the old tree. The second inequality implies that the new tree has strictly more leaves. If the new tree still has two levels whose degrees are different by more than 1 (for example, d'_i and d'_j themselves may still be different by more than 1), then the same construction is repeated.

Eventually, T^* is transformed to a new tree T^{**}, in which the degrees of nodes at the same level are identical, and the degrees of non-leaf nodes are different by at most 1. The latter fact implies that non-leaf nodes are either of a single degree d or of two degrees d and $d+1$. Recall that T achieved optimal turnaround ϵ for n elementary components (leaves). Throughout its transformation to T^{**}, the number of leaves has been at least n, and the turnaround has been at most ϵ. Because of the optimality of T, it is not possible that ϵ has decreased, but it is possible that n has increased, as small increases in the number of leaves are possible that do not affect the turnaround.

This construction proves the following theorem.

Theorem. For every product, there exists an optimal tree which is either perfect or nearly perfect.

Note that the optimal tree may include extra leaves at no additional cost; excess leaves can be removed arbitrarily from the perfect or nearly perfect tree, without affecting the turnaround, thus obtaining an optimal tree for the given number of elementary components (the resulting tree is no longer perfect or nearly perfect, because its bottom level is no longer full).

4 Finding Optimal Trees

Consider the same manufacturing environment and a product that requires n elementary components. From the theorem we know that there is a perfect or a nearly perfect manufacturing tree that optimizes the manufacturing turnaround. However, multiple such trees are possible, each with a different turnaround. For example, two alternative trees for $n = 6$, one perfect and one nearly perfect, are shown in Figure 3, with their specific turnaround formulas (for simplicity, the example assumes linear assembly time: $f(d) = d \cdot a$). For $a = 10$ and $b = 1$ the right tree would give shorter turnaround, whereas for $a = 1$ and $b = 10$ the left tree would be preferred. Intuitively, when assembly time is low and delivery time is high, a shallow manufacturing tree with a large degree would be preferred;

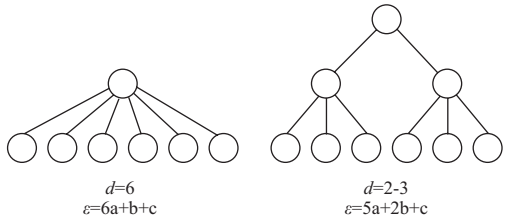

Fig. 3. A perfect and a nearly perfect tree for $n = 6$

conversely, when the assembly time is high and the delivery time is low, a deep tree with a small degree would be preferred. A brute force solution for finding the optimal tree is to enumerate all the possible perfect or nearly perfect trees for a given product and evaluate their turnarounds.

In this section we consider the issue of selecting the optimal tree from the possible alternatives. Our treatment, however, is *approximative*.

We treat the manufacturing tree as a continuous structure, not the discrete structure that it is. In our model, the manufacturing of a given product is modeled by a tree. In the previous treatment, the parameters of such trees (e.g., their height, node degrees and total number of leaves) were integers. In this section, we relax this restriction: We allow node degrees that are not necessarily integers (while maintaining the number of leaves and the height as integers).

In other words, we define a slightly different optimization problem. Assume a convex function f, numbers b and c, and integer n. Assume a sequence of numbers $d_0, \ldots, d_{h-1}$, such that $n = \Pi_{i=0}^{h-1} d_i$, and consider the cost function $t(d_0, \ldots, d_{h-1}) = \sum_{i=0}^{h-1} f(d_i) + h \cdot b + c$. This problem corresponds to the tree T^* from the previous section: a balanced tree in which nodes at the same level have the same degree.

Note that both the length of the sequence and the individual values in the sequence are variables; however, the product of the entire sequence is constant. By convexity of f, we get that for each value h, the cost function t is minimal when the d_i are identical: $d_i = n^{1/h}$. This suggests that the optimal structure is a perfect "tree" of degree $n^{1/h}$, but note that this degree is not necessarily an integer.[3]

Note that by relaxing the requirement that the degree be an integer, a perfect "tree" can be obtained for any number of leaves n (previously, the perfect or nearly perfect tree often had more leaves than the given n).

In the new perfect "tree", the time spent at each node is always $f(d)$ and the length of every root-to-leaf path is h. Hence, the turnaround is simply $f(d) \cdot h + b \cdot h + c$. In this expression, h and d are variables, but since $h = \log_d n$, turnaround can be expressed as a function of d only (with constants n, b, and c and the convex function f):

[3] This suggests why a nearly perfect tree has two consecutive degrees: these are the integers "around" d.

$$t(d) = f(d) \cdot \log_d n + b \cdot \log_d n + c$$
$$= \log_d n(f(d) + b) + c$$

Note that $t(d)$ is defined only for $d \leqslant n$.

The search for an optimal manufacturing tree has now become a search for the optimal degree d. Of course, a non-integer value of d cannot be used in a manufacturing process, but it should serve as a good approximation of the "true" degree of the optimal process.

For specific analyses, we consider two individual convex assembly functions.

4.1 Linear Assembly Time

Assume first that assembly time is a linear function; i.e., $f(x) = a \cdot x$, where x is the number of components and a is a constant per-component assembly time. Such functions model assemblies in which every component requires a single action and the time for each action is a. The turnaround formula becomes

$$t(d) = \log_d n \cdot (a \cdot d + b) + c$$

Figure 4 charts two examples of this turnaround function. In both cases the number of elementary components (leaves) is $n = 500$, and creation time is $c = 5$. In one case assembly time is higher than delivery time ($a = 2$ and $b = 1$), in the other case delivery time is higher than assembly time ($a = 1$ and $b = 10$).

The optimum value of this turnaround function may be found by solving the equation $t'(d) = 0$. We get this expression for the optimal d:

$$d \cdot \ln d - d = b/a$$

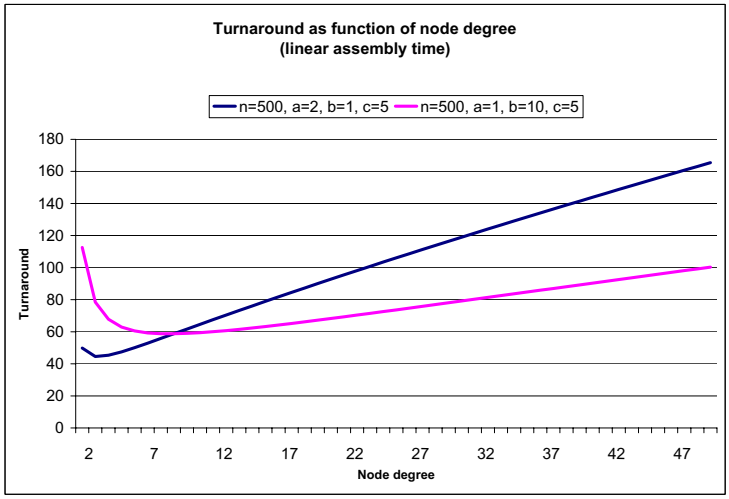

Fig. 4. Turnaround as function of node degree (linear assembly time)

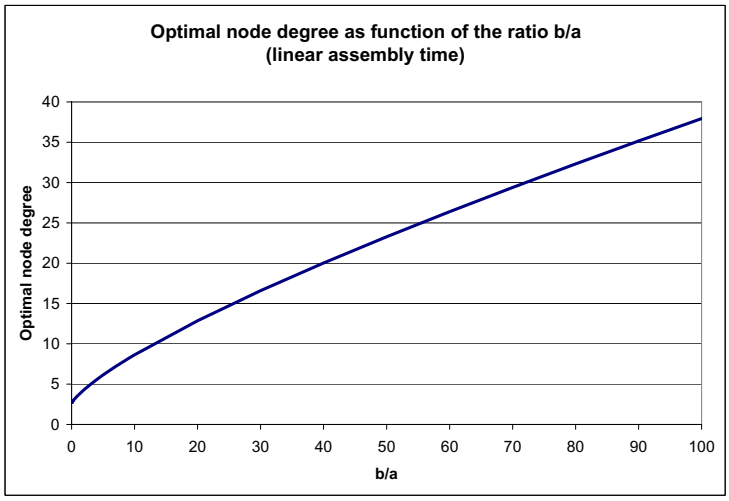

Fig. 5. Optimal node degree as function of the ratio b/a (linear assembly time)

While this equation does not have an open solution for d, it can be approximated with common techniques.

The equation reveals an interesting fact: *the optimal solution is independent of* n. Of the four parameters of the problem, n, a, b, and c, the optimal manufacturing "tree" is of a degree that depends only on the ratio b/a. This somewhat surprising conclusion implies that for each manufacturing environment, there is an *ideal assembly size*, which is optimal for the manufacturing of products of *any* scale (i.e., any number of elementary components).

Figure 5 charts the optimal degree as a function of the ratio b/a. This graph can be used to estimate the optimal assembly size in different manufacturing environments.

4.2 Quadratic Assembly Time

Assume now that assembly time is a quadratic function; i.e., $f(x) = a \cdot x^2$, where x is the number of components and a is a constant per-component assembly time. Such functions model assemblies in which every pair of components requires an action and the time for each action is a. The turnaround function becomes

$$t(d) = \log_d n \cdot (a \cdot d^2 + b) + c$$

Two examples of this turnaround function are charted in Figure 6. In both cases the number of elementary components (leaves) is $n = 500$ and the creation time is $c = 5$. In one case assembly time is higher than delivery time ($a = 2$ and $b = 1$), in the other case delivery time is higher than assembly time ($a = 1$ and $b = 1000$).

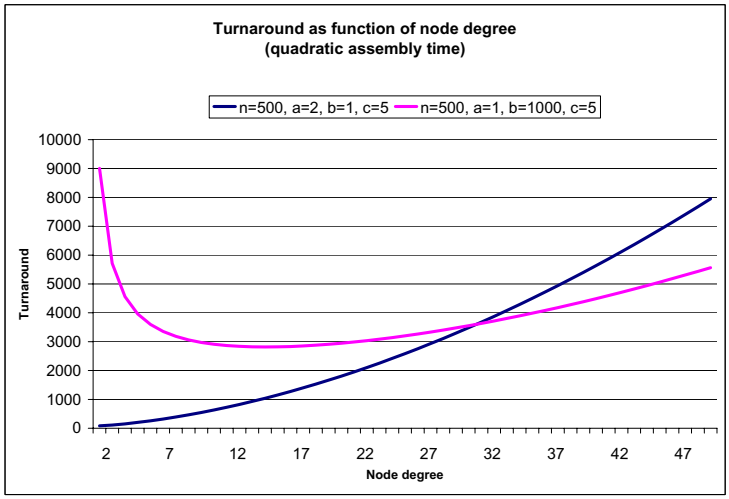

Fig. 6. Turnaround as function of node degree (quadratic assembly time)

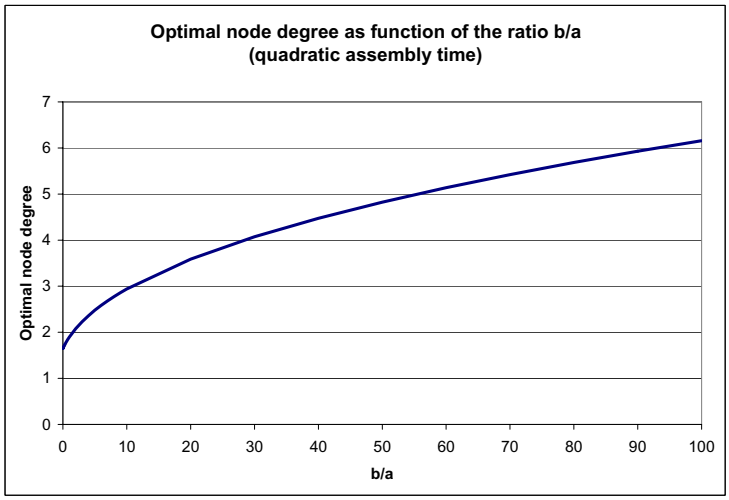

Fig. 7. Optimal node degree as function of the ratio b/a (quadratic assembly time)

The equation $t'(d) = 0$ provides an expression for the optimal d:

$$2d^2 \cdot \ln d - d^2 = b/a$$

Once more, we observe that the degree of the optimal manufacturing "tree" does not depend on the number of leaves (elementary components), but only on the ratio b/a of manufacturing environment parameters. Figure 7 charts the optimum degree as a function of this ratio.

5 Conclusion

Hierarchical organization is a standard approach in a wide variety of complex systems. While the issue of breadth *vs.* depth of hierarchies has been raised in some of these contexts, the authors are not aware of any prior theoretical treatment of this trade-off.

The process of manufacturing (assembly) of complex products from elementary products provides a good context for a theoretical analysis of this trade-off, and in this paper we adopted its terminology and objectives (but recall that in Section 1 we showed other contexts to which the analysis is directly applicable).

We assumed that manufacturing environments are specified with three parameters: the time for creating elementary components, the time for assembling partial products, and the time for delivering partial products to their procurers. Each manufacturing process was then assigned a cost that corresponds to *turnaround*: the time between the receiving of a manufacturing order and its completion. In this model, we addressed the problem of *optimal manufacturing processes*: Given a product with a specified number of elementary components, find a manufacturing process that optimizes the turnaround. This problem translates immediately to finding the optimal *assembly size* for a given product.

We proved that for a given number of elementary components there exists an optimal tree which is perfect or nearly perfect. Since there are several such trees for a given number of elementary components, we showed how to find the optimal tree among these by means of simple differentiation. We observed that the optimal degree (optimal assembly size), for assembly times that are linear or quadratic in the number of components, is a function of the ratio of the assembly and delivery times only (and is independent of the number of components). This somewhat surprising conclusion implies that in each manufacturing environment there is an *ideal assembly size*, which is optimal for the manufacturing of products of *any* scale (i.e., any number of elementary components).

Our model is simple, in that its three cost parameters are uniform across the manufacturing environment. That is, all products have identical creation and delivery times, there is a single function that calculates the cost of assembly, and it depends only on the number of components in the assembly (not on the specific components). We are interested in extending the model to consider manufacturing environments in which different products (or categories of products) incur different costs.

Another underlying assumption is that each manufacturing node is capable of assembling any combination of products. It would be interesting to consider a more general model in which various types of constraints must be enforced; for example, certain products cannot be combined with certain other products in a single sub-assembly, certain products must be combined with certain other products in any sub-assembly, some nodes have limits on the size of their assemblies, and so on.

References

1. L. Gulick. Notes on the theory of organization. In L. Gulick and L. Urwich, editors, *Papers on the Science of Administration*, pages 191–195. Institute of Public Administration, Columbia University, New york, 1937.
2. B. Schneiderman. *Designing the User Interface: Strategies for Effective Computer Interaction.* Addison-Wesley, 1992.
3. H.A. Simon. The organization of complex systems. In H.H. Pattee, editor, *Hierarchy Theory.* George Braziller, New York, 1973.
4. D.B. West. *Introduction to Graph Theory.* Prentice Hall, 2001.
5. J. Woodward. *Industrial Organization: Theory and Practice.* Oxford University Press, 1965.