CHAPTER 3: ARRANGEMENTS AND CONFIGURATIONS

3-1. Defining Situations, Usages, Arrangements, and Configurations

In a particular *situation* there will be a certain number of processes to be executed and some number of processing nodes available. We call a situation with N processes and M processing nodes an *NxM situation* (read ‘‘N by M’’). For example, when four processes are to be executed and there are three processing nodes available for use, it is a 4x3 situation. If there are more processing nodes available than there are processes to execute (if M > N), the ‘‘extra’’ nodes cannot be used and we ignore them. Thus, NxN is the ‘‘largest’’ situation, the situation in which each process could be executed on its own processing node.

All the available processors need not necessarily be used. For example, in a 4x4 situation, we might choose to use only two of the processors; we call this a *4@2 usage,* (read ‘‘4 on 2’’). In general, when N process are executed on exactly M processors, we call it an N@M usage.

An *arrangement* specifies how many processes are to be executed on each processing node, but not which specific processes. For example, in a 2x2 situation there are two ways the processes can be arranged on the processing nodes: both processes on one node, or one process on each of the two nodes. We denote the arrangement with both processes on one processor and none on the second processor by 2:0, and the arrangement with one process on each of two processors by 1:1. Another example is a 5:2:1 arrangement, which puts five processes on one node, two on another, and one on a third node; this is an 8@3 usage. This notation for the arrangement does not make it clear how many processors are available: the situation could be 8x3, or 8x4, and so on, up to 8x8; if it were an 8x5 situation, we might choose to write the arrangement as 5:2:1:0:0 to make clear the number of processors available.

A mapping of specific processes to processors is called a *configuration.* For example, there is only one way of mapping three processes onto three processing nodes. Calling the three processes process 0, process 1, and process 2, we represent this configuration as 0/1/2; that is, process 0 is on one processor, process 1 is on a second processor, and process 2 is on a third processor. This configuration is in the 1:1:1 arrangement, it is a 3@3 usage and a 3x3 situation. In a 2x2 situation, the configuration in which both processes are executed on one processing node is written 01/-, that is, processes 0 and 1 on one node and none on the other node. We label the 0/1/2 configuration *APART,* because each process is executed apart from the others. In fact, we will call any configuration which is an N@N usage an APART configuration. The 01/- configuration is called *TOGETHER* because all the processes are together on one node; we will call any configuration which is an N@1 usage TOGETHER. Since we are considering only identical processors, it does not matter which processing node is used for which group of processes, so 0/1/2 is the same as 1/2/0 and the same as 2/0/1; nor does it matter which processing nodes are used and which unused, so 01/- is the same as -/01.

The number of configurations in an arrangement varies from arrangement to arrangement. In 2x2, there are only two arrangements (2:0 and 1:1), each of which has only one configuration (01/- and 0/1, respectively). In 4@2, however, the 3:1 arrangement has four configurations, and the 2:2 arrangement has three (see Figure 8-1).

3-2. Counting Arrangements and Configurations

For small numbers of processes it is feasible to count the number of arrangements and configurations by making lists of all the possible arrangements and configurations. The numbers of arrangements and configurations both grow very rapidly with the number of processes. For example, as the number of processes grows from two to three to four, the number of configurations grows from two to five to 15 (see Sections 6-1.1, 7-1.1, and 8-2). We would like to be able to calculate the numbers of arrangements and configurations that are possible with an arbitrary number of processes, without having to list them all. In this section, we find formulas for calculating the numbers of arrangements and configurations for any number of processes, and for calculating the number of configurations in a given arrangement.†

In combinatoric terms, arrangements are distributions of identical objects to identical recipients (since an arrangement deals only with numbers of processes, the processes are not distinguished from one another), and configurations are distributions of distinct objects to identical recipients. We want to count the number of ways the objects can be distributed to the recipients.

More formally, a distribution is a function between two finite sets. Let N be the set of processes and M the set of processing nodes, with $|N| = n$ and $|M| = m$. (This notation is used in this section only; in the preceding section, and in the rest of the thesis, we use N for the number of processes and M for the number of processing nodes.) We would like to count the number of functions $f : N \rightarrow M$ mapping elements of set N to elements of set M, in our case mapping processes to processing nodes. In dealing with arrangements, the processes are considered indistinguishable; in dealing with configurations, however, the processes are distinguished one from the other. Processing nodes are all identical, and there is no need to distinguish them from one another. The mapping functions f can be arbitrary. In particular, f need not be surjective (or ‘‘onto’’) since the available processing nodes need not all be used, and f need not be injective (or ‘‘one-to-one’’) since processing nodes may execute more than one process ([Liu77, pp. 72-73]).

In the following sections, we discuss combinatoric concepts and techniques that we can use to compute the numbers of arrangements and configurations, and the number of configurations in a given arrangement.

3-2.1. Counting the Number of Arrangements

In an arrangement of n processes on exactly k processing nodes, we wish to divide the number of processes n into k smaller numbers whose sum is n; this is called a partition of the integer n. We wish to count the number of ways in which n can be partitioned into k parts, for all k from 1 to m. In this section we describe partitions of integers and how to count them, we discuss formulas for computing the number of partitions of an integer, and we use the formulas to compute the number of arrangements that are possible for certain numbers of processes and processing nodes.

A *partition* of n, where n is a nonnegative integer, is a sequence of nonnegative integers $l = (l_1, \cdots, l_k)$ such that $\sum l_i = n$ and $l_1 \geq \cdots \geq l_k$. The non-zero terms l_i are called the *parts* of l, and we say that l has k parts, where $k = |\{i : l_i > 0\}|$. We write $p_k(n)$ for the number of partitions of n with exactly k parts, and $p(n)$ for the total number of partitions of n. By convention, $p_0(0) = p(0) = 1$. Note that:

$$p_n(n) = 1$$

$$p_{n-1}(n) = 1 \text{ if } n > 1$$

$$p_1(n) = 1$$

† Much of the material in this section is drawn from [Sta86, pp. 28, 31-34, 38-39], and from [Bog83, pp. 24-25, 41-60].

$$p_2(n) = \left\lfloor \frac{n}{2} \right\rfloor$$

The number of partitions of n with k parts can be computed using the recurrence

$$p_k(n) = p_{k-1}(n-1) + p_k(n-k)$$

To count the number of arrangements that are possible with n processes on exactly k processing nodes, we recall that an arrangement gives the number of processes on each processor. The numbers of processes must be positive integers, and their sum must equal n. Thus, an arrangement is a partition of n into k parts, and the number of arrangements of n processes on k processors is exactly the number of partitions of n into k parts, $p_k(n)$. The total number of arrangements of n processes is the total number of partitions of n, $p(n)$, which is

$$p_1(n) + p_2(n) + \cdots + p_m(n)$$

Using this information we can compute the number of arrangements possible with various numbers of processes and processing nodes. Figure 3-1 shows the number of arrangements possible for various small numbers of processes, using various numbers of processors, as well as the number of arrangements possible with various numbers of processors available. Figure 3-2 shows the numbers of arrangements possible with 32 processes and from one to 32 processing nodes; the pattern seen in this figure is typical: as the number of processors, m, increases, $p_m(n)$ increases rapidly to a peak (in this case at $m=8$), then decreases more slowly down to one. Figure 3-3 shows $p(n)$, the total number of arrangements possible with n processes to be executed and up to n processors available, for some larger numbers of processes.

Number of Processes n	Number of Processing Nodes m	Number of Arrangements Using Exactly m Nodes $p_m(n)$	Number of Arrangements Using Up To m Nodes $p(n) = \sum_{i=1}^{m} p_i(n)$
2	1	1	1
	2	1	2
3	1	1	1
	2	1	2
	3	1	3
4	1	1	1
	2	2	3
	3	1	4
	4	1	5
8	1	1	1
	2	4	5
	3	5	10
	4	5	15
	5	3	18
	6	2	20
	7	1	21
	8	1	22
16	1	1	1
	2	8	9
	3	21	30
	4	34	64
	5	37	101
	6	35	136
	7	28	164
	8	22	186
	9	15	201
	10	11	212
	11	7	219
	12	5	224
	13	3	227
	14	2	229
	15	1	230
	16	1	231

Figure 3-1:
Number of Arrangements Possible
with Various Numbers of Processors and Processing Nodes

Number of Processes n	Number of Processing Nodes m	Number of Arrangements Using Exactly m Nodes $p_m(n)$	Number of Arrangements Using Up To m Nodes $p(n) = \sum_{i=1}^{m} p_i(n)$
32	1	1	1
	2	16	17
	3	85	102
	4	249	351
	5	480	831
	6	709	1,540
	7	860	2,400
	8	919	3,319
	9	887	4,206
	10	807	5,013
	11	695	5,708
	12	582	6,290
	13	471	6,761
	14	378	7,139
	15	295	7,434
	16	231	7,665
	17	176	7,841
	18	135	7,976
	19	101	8,077
	20	77	8,154
	21	56	8,210
	22	42	8,252
	23	30	8,282
	24	22	8,304
	25	15	8,319
	26	11	8,330
	27	7	8,337
	28	5	8,342
	29	3	8,345
	30	2	8,347
	31	1	8,348
	32	1	8,349

Figure 3-2:
Number of Arrangements Possible with 32 Processes

Number of Processes n	Number of Processing Nodes Available m	Number of Arrangements Using Up To m Nodes $p(n) = \sum_{i=1}^{m} p_i(n)$
64	64	$1.74 \cdot 10^{6}$
128	128	$4.35 \cdot 10^{9}$
256	256	$3.66 \cdot 10^{14}$
512	512	$4.45 \cdot 10^{21}$
1,024	1,024	$6.18 \cdot 10^{31}$

Figure 3-3:
Total Numbers of Arrangements Possible
with Larger Numbers of Processes

The numbers of arrangements possible with two, three, and four processes with various numbers of processing nodes match the enumerations presented in later chapters (see Sections 6-1.1, 7-1.1, and 8-2).

3-2.2. Counting the Number of Configurations

A configuration specifies which processes are to be executed on each processing node. The processes are distinguished from each other; the processing nodes, all identical, are not. Thus, for a configuration of n processes on exactly k processing nodes, we have a set N consisting of n processes, and we wish to divide it into k subsets whose union is N; this is called a partition of the set N (as contrasted with a partition of the integer n, which we used in computing the number of arrangements). We wish to count the number of ways in which set N can be partitioned into k subsets, for all k from 1 to m. In this section we describe partitions of sets and how to count them, we discuss formulas for computing the number of partitions of a set, and we use the formulas to compute the number of configurations that are possible for certain numbers of processes and processing nodes.

A finite set can be *partitioned* into non-empty, mutually disjoint subsets that together contain all the elements of the set. More formally, a partition of a finite set N is a collection $\{B_1, B_2, \ldots, B_k\}$ of subsets of N such that: (a) $B_i \neq \varnothing$ for all i, (b) $B_i \cap B_j = \varnothing$ if $i \neq j$, and (c) $B_i \cup B_j \cup \cdots \cup B_k = N$. Define $S(n,k)$ to be the number of partitions of an n-element set into k parts; the numbers $S(n,k)$ are called *Stirling numbers of the second kind.* We have $S(n,k) = 0$ unless n and k are integers and $1 \leq k \leq m$; however, by convention, $S(0,0) = 1$. For $n \geq 1$, Stirling numbers of the second kind have the following properties:

$$S(n,k) = 0 \text{ if } k > n$$

$$S(n,0) = 0$$

$$S(n,1) = 1$$

$$S(n,2) = 2^{n-1} - 1$$

$$S(n,3) = \frac{1}{2}(3^{n-1} - 2^n + 1) \quad \dagger$$

$$S(n,n-1) = \binom{n}{2}$$

$$S(n,n) = 1$$

† The formula for $S(n,3)$ is from [And89, p. 66].

Stirling numbers of the second kind satisfy the following recurrences:

$$S(n,k) = k \cdot S(n-1,k) + S(n-1,k-1)$$

$$S(n,k) = \sum_{i=0}^{n-1} \binom{n-1}{i} S(i,k-1)$$

The total number of partitions of an n-element set, $B(n)$, is

$$B(n) = \sum_{k=1}^{n} S(n,k)$$

$B(n)$ is called a *Bell number*. A recursive formula for $B(n)$ is

$$B(n) = \sum_{i=0}^{n} \binom{n-1}{i} B(i)$$

A configuration of n processes on exactly k processing nodes (an N@k configuration) is a partition of an n-element set into k parts, where the subsets represent the processes to be executed on each processing node. The subsets must be mutually disjoint since no process is executed on more than one node; they must be non-empty since each processor being used must execute at least one process (we are speaking of processors in use; there may be other processors available but not used in the configuration); and all processes must be assigned to a node, that is, the union of the sets of processes on each node must be the complete set of processes. Thus, the number of configurations of n processes on exactly k processing nodes is the number of partitions of an n-element set into exactly k parts, $S(n,k)$.

The total number of configurations possible with n processes and m available processing nodes is the sum of the number of configurations possible with one node, with two nodes, and so on, up to m nodes, $\sum_{k=1}^{m} S(n,k)$. If $n = m$, the total number of configurations possible is the Bell number, $B(n)$.

Using this information we can compute the number of configurations possible with various numbers of processes and processing nodes. Figure 3-4 shows the number of arrangements possible for various small numbers of processes, using various numbers of processors, as well as the number of arrangements possible with various numbers of processors available. Figure 3-5 shows the numbers of arrangements possible with 32 processes and from one to 32 processing nodes. Figure 3-6 shows the Bell number, $B(n)$, which is the total number of configurations possible with n processes to be executed and up to n processors available, for some larger numbers of processes.

Number of Processes n	Number of Processing Nodes m	Number of Configurations Using Exactly m Nodes $S(n,m)$	Number of Configurations Using Up To m Nodes $\sum_{i=1}^{m} S(n,i)$
2	1	1	1
	2	1	2
3	1	1	1
	2	3	4
	3	1	5
4	1	1	1
	2	7	8
	3	6	14
	4	1	15
8	1	1	1
	2	127	128
	3	966	1,094
	4	1,701	2,795
	5	1,050	3,845
	6	266	4,111
	7	28	4,139
	8	1	4,140
16	1	1	1
	2	$3.28 \cdot 10^4$	$3.28 \cdot 10^4$
	3	$7.14 \cdot 10^6$	$7.17 \cdot 10^6$
	4	$1.72 \cdot 10^8$	$1.79 \cdot 10^8$
	5	$1.10 \cdot 10^9$	$1.28 \cdot 10^9$
	6	$2.73 \cdot 10^9$	$4.01 \cdot 10^9$
	7	$3.28 \cdot 10^9$	$7.29 \cdot 10^9$
	8	$2.14 \cdot 10^9$	$9.43 \cdot 10^9$
	9	$8.21 \cdot 10^8$	$1.03 \cdot 10^{10}$
	10	$1.94 \cdot 10^8$	$1.04 \cdot 10^{10}$
	11	$2.89 \cdot 10^7$	$1.05 \cdot 10^{10}$
	12	$2.76 \cdot 10^6$	$1.05 \cdot 10^{10}$
	13	$1.66 \cdot 10^5$	$1.05 \cdot 10^{10}$
	14	$6.02 \cdot 10^3$	$1.05 \cdot 10^{10}$
	15	$1.20 \cdot 10^2$	$1.05 \cdot 10^{10}$
	16	1	$1.05 \cdot 10^{10}$

Figure 3-4:
Number of Configurations Possible
with Various Numbers of Processors and Processing Nodes

Number of Processes n	Number of Processing Nodes m	Number of Configurations Using Exactly m Nodes $S(n,m)$	Number of Configurations Using Up To m Nodes $\sum_{i=1}^{m} S(n,i)$
32	1	1	1
	2	$2.15 \cdot 10^{9}$	$2.15 \cdot 10^{9}$
	3	$3.09 \cdot 10^{14}$	$3.09 \cdot 10^{14}$
	4	$7.68 \cdot 10^{17}$	$7.69 \cdot 10^{17}$
	5	$1.93 \cdot 10^{20}$	$1.94 \cdot 10^{20}$
	6	$1.09 \cdot 10^{22}$	$1.11 \cdot 10^{22}$
	7	$2.08 \cdot 10^{23}$	$2.19 \cdot 10^{23}$
	8	$1.75 \cdot 10^{24}$	$1.97 \cdot 10^{24}$
	9	$7.61 \cdot 10^{24}$	$9.58 \cdot 10^{24}$
	10	$1.90 \cdot 10^{25}$	$2.86 \cdot 10^{25}$
	11	$2.97 \cdot 10^{25}$	$5.84 \cdot 10^{25}$
	12	$3.07 \cdot 10^{25}$	$8.91 \cdot 10^{25}$
	13	$2.20 \cdot 10^{25}$	$1.11 \cdot 10^{26}$
	14	$1.12 \cdot 10^{25}$	$1.22 \cdot 10^{26}$
	15	$4.23 \cdot 10^{24}$	$1.27 \cdot 10^{26}$
	16	$1.19 \cdot 10^{24}$	$1.28 \cdot 10^{26}$
	17	$2.58 \cdot 10^{23}$	$1.28 \cdot 10^{26}$
	18	$4.31 \cdot 10^{22}$	$1.28 \cdot 10^{26}$
	19	$5.63 \cdot 10^{21}$	$1.28 \cdot 10^{26}$
	20	$5.80 \cdot 10^{20}$	$1.28 \cdot 10^{26}$
	21	$4.73 \cdot 10^{19}$	$1.28 \cdot 10^{26}$
	22	$3.07 \cdot 10^{18}$	$1.28 \cdot 10^{26}$
	23	$1.58 \cdot 10^{17}$	$1.28 \cdot 10^{26}$
	24	$6.47 \cdot 10^{15}$	$1.28 \cdot 10^{26}$
	25	$2.08 \cdot 10^{14}$	$1.28 \cdot 10^{26}$
	26	$5.25 \cdot 10^{12}$	$1.28 \cdot 10^{26}$
	27	$1.02 \cdot 10^{11}$	$1.28 \cdot 10^{26}$
	28	$1.48 \cdot 10^{9}$	$1.28 \cdot 10^{26}$
	29	$1.56 \cdot 10^{7}$	$1.28 \cdot 10^{26}$
	30	$1.13 \cdot 10^{5}$	$1.28 \cdot 10^{26}$
	31	$4.96 \cdot 10^{2}$	$1.28 \cdot 10^{26}$
	32	1	$1.28 \cdot 10^{26}$

Figure 3-5:
Number of Configurations Possible with 32 Processes

Number of Processes n	Number of Processing Nodes Available m	Number of Configurations Using Up To m Nodes $B(n) = \sum_{i=1}^{m} S(n,i)$
64	64	$1.72 \cdot 10^{65}$
128	128	$1.12 \cdot 10^{158}$

Figure 3-6:
Total Numbers of Configurations Possible
with Larger Numbers of Processes

The numbers of configurations possible with two, three, and four processes with various numbers of processing nodes match the enumerations presented in later chapters (see Sections 6-1.1, 7-1.1, and 8-2).

3-2.3. Counting the Number of Configurations Per Arrangement

We would like to be able to compute exactly how many configurations there are in each arrangement. For example, we know that there are two 4@2 arrangements because $p_2(4) = 2$, and we know that there are seven 4@2 configurations because $S(4,2) = 7$. We would like to know how many of the seven configurations are in each of the two arrangements.

Another way of stating the problem is that we wish to distribute n identical objects to k identical recipients so that each recipient gets a specified number of objects, and we wish to count the number of ways in which the objects may be so distributed. The number of distributions possible would be just the multinomial coefficient

$$\left(\begin{matrix} n \\ k_1, k_2, \ldots, k_m \end{matrix} \right) = \frac{n!}{k_1!\, k_2!\, \cdots\, k_m!}$$

except that some distributions are equivalent and we must reduce the count accordingly. Two distributions are equivalent if we can get one from the other by interchanging the items distributed to one recipient with those distributed to another. For example, the 2:2 configurations 01/23 and 23/01 are equivalent, and we wish to count the configuration only once.

More formally, we wish to partition a set of n elements into subsets, where the subsets are of specified sizes. Let us specify the sizes of the subsets with a *type vector* $(j_1, j_2, \ldots, j_n)$, where j_i gives the number of subsets of size i for $i = 1$ to n (we may drop trailing zeros from the type vector, keeping only the first k nonzero elements). Then the number of partitions of a set with n elements into j_1 subsets of size 1, j_2 subsets of size 2, and so on, up to j_n subsets of size n is

$$\frac{n!}{\prod_{i=1}^{n} (i!)^{j_i} \cdot j_i!}$$

Let us use this expression to compute the numbers of configurations in our example. In a 4@2 situation, there are two possible arrangements: 3:1 and 2:2. The type vector for the 3:1 arrangement is (1, 0, 1), that is, one subset of size one, none of size two, and one of size three. Thus we have

$$\frac{4!}{(1!^1 \cdot 1!)\,(2!^0 \cdot 0!)\,(3!^1 \cdot 1!)}$$
$$= \frac{4!}{(1)\,(1)\,(3!)}$$
$$= 4$$

So there are four configurations in the 3:1 arrangement.

The type vector for the 2:2 arrangement is (0, 2), that is, there are two subsets of size two. We compute

$$\frac{4!}{(1!^0 \cdot 0!)\,(2!^2 \cdot 2!)}$$
$$= \frac{4!}{(1)\,(2^3)}$$
$$= 3$$

So there are three configurations in the 2:2 arrangement. The total number of 4@2 configurations in the

two arrangements is $4+3=7$, which matches the number of 4@2 configurations computed for the Stirling number $S(4,2)=7$. The numbers of configurations computed for these arrangements are the same as those found through enumeration in Section 8-2 (see Figure 8-1).

3-3. Summary

In this chapter, we have defined situations, usages, arrangements, and configurations. We have shown how to compute the number of arrangements and the number of configurations which are possible with various numbers of processes and processing nodes. We have shown that the numbers grow very large very quickly, so that brute force approaches to making assignments of processes to processing nodes will not work. We have also shown how to compute the number of configurations in a given arrangement.