

CS 441/541 Introduction to Parallel Computing July 8, 2004

Data Distribution Functions

(This work builds on previous work by the professors giving the class, and in turn on others' work. The full references will be provided, which are summarized as **Fox 1988, Van de Velde 1992**, with extensions in **Skjellum 1990**, and by others. Furthermore, these mappings are well known techniques in the field by now and appear many places without specific attribution.)

The purpose of a data distribution function, inverse, and cardinality is to map the set of M global indices $\mathcal{I} = \{0, \dots, M-1\}$ onto a set of P partitions, and to have a one to one mapping within those P partitions of those global indices.

The forward function $(p,i) := \mu(I,P,M)$ transforms from the global space to the partition space. This is also referred to as a bijection.

The inverse function $I := \mu^{-1}(p,i,P,M)$ transforms from the partition space to the global space.

The cardinality function defines the number of indices that go into each partition: $m = \mu^{\#}(p,P,M)$.

The transformation is one-to-one (unique partition position for each global index), and onto (complete coverage), and therefore invertible. Notice that the local sets of partitions are also compact, that is, there are no holes, and each local partition is of the form $(0, \dots, n(p)-1)$, where as we see below, $m(p)$ is the cardinality for the given partition p , $0 \leq p < P$.

The first thing we define is a load-balanced cardinality function that we can use with two other distributions:

$$\mu^{\#}(p,P,N) = L \text{ for } P \geq R, \\ L+1 \text{ for } 0 \leq P < R$$

where $L = \text{floor}(N/P)$, and $R = M \bmod P$.

This cardinality function places one extra index from the remainder R in each of the first R partitions.

We refer to this as a *load balanced* distribution because the difference between the largest and smallest partition is minimized. When $R=0$, these are all identical, and they are never worse than 1 different.

Now, we can define the *scatter* or *wrap-mapped distribution*. This is used widely to maximize the average distance between the coefficients of \mathcal{I} . It is like a shuffle.

$$\text{Forward } (p,i) := \sigma(I,P,M) = (I \bmod P, \text{floor}(I/P))$$

where the partition is assigned by "round robin," and the local index is assigned by the number of times the round robin has been completed.

The inverse distribution for $\sigma(I,P,M)$ can be figured out by realizing that the definition of division and modulus for integers is as follows:

$$\text{floor}(a/b)b = a - (a \bmod b), \text{ for } b \neq 0.$$

Thus, the inverse $I := \sigma^{-1}(p,i,P,M) = iP + p$.

One can empirically verify by doing some examples that the cardinality function defined above lays out indices compatibly with this distribution.

The *linear load balanced distribution* can be similarly defined.

First of all, the forward distribution tries to put the $L+1$ indices in the first R processes:

$$(p,i) := \lambda(I,P,M) = \{ \max(\text{floor}(I/(L+1)), \text{floor}((I-(L+1)R)/L) + R), \\ I - \max(p,R)(L+1) - \max(0,p-R)L \}$$

The definition of p can be explained by recognizing that the first R partitions use up $L+1$ indices each, and partitions after that use L indices each. The first term in the maximum adds one to p for each $L+1$ indices in I . This gives an equal or higher count to the second term for small I . They balance at $I=(L+1)R$. After that, the second term in the maximum dominates, and “uses up” L coefficients for the rest of the partitions.

Notice that in this distribution, one computes p first, and then i . Here i depends on p and specifically subtracts off indices along the same logical as described for p . The indices stay in the same order, so that nearby indices in the global space tend to be nearby in the local partitions.

Now, because $i = I - \max(p,R)(L+1) - \max(0,p-R)L$, then this can be easily inverted to provide:

$$I := \lambda^{-1}(p,i,P,M) = i + \max(p,R)(L+1) + \max(0,p-R)L$$

Finally, in order to make the connection between partitions and parallel programming, use P as the number of processes in a decomposition, and consider p to be the logical process number, $0 \leq p < P$. However, these functions represent special permutations of the M integers, and are much more generally useful, and probably have been used in many other ways.

Furthermore, one can apply the distributions to a matrix on a two-dimensional logical topology. For instance, using linear and scatter, we can now distribute a matrix of size $M \times N$ on a logical topology of shape $P \times Q$ with four variations: linear-linear, linear-scatter, scatter-linear, and scatter-scatter. The goal of linear algebra and other computations defined on logical topologies will be to make their correctness independent of these distribution choices, but to allow for this degree of freedom in order to support

- a) how applications want to work with data in parallel before and after some linear algebra operation
- b) to support different performance that may occur as a function of the data layouts.