Figure 3.1  Decomposition of dense matrix-vector multiplication into $n$ tasks, where $n$ is the number of rows in the matrix. The portions of the matrix and the input and output vectors accessed by Task 1 are highlighted.
Figure 3.2  The different tables and their dependencies in a query processing operation.
Figure 3.3  An alternate data-dependency graph for the query processing operation.
Figure 3.4  Decomposition of dense matrix-vector multiplication into four tasks. The portions of the matrix and the input and output vectors accessed by Task 1 are highlighted.
Figure 3.5  Abstractions of the task graphs of Figures 3.2 and 3.3, respectively.
Figure 3.6 A decomposition for sparse matrix-vector multiplication and the corresponding task-interaction graph. In the decomposition Task $i$ computes $\sum_{0 \leq j \leq 11, A[i, j] \neq 0} A[i, j] b[j]$. 
Figure 3.7  Mappings of the task graphs of Figure 3.5 onto four processes.
Figure 3.8  The quicksort task-dependency graph based on recursive decomposition for sorting a sequence of 12 numbers.
Figure 3.9 The task-dependency graph for finding the minimum number in the sequence \{4, 9, 1, 7, 8, 11, 2, 12\}. Each node in the tree represents the task of finding the minimum of a pair of numbers.
\[
\begin{pmatrix}
A_{1,1} & A_{1,2} \\
A_{2,1} & A_{2,2}
\end{pmatrix}
\begin{pmatrix}
B_{1,1} & B_{1,2} \\
B_{2,1} & B_{2,2}
\end{pmatrix}
\rightarrow
\begin{pmatrix}
C_{1,1} & C_{1,2} \\
C_{2,1} & C_{2,2}
\end{pmatrix}
\]

(a)

Task 1: \(C_{1,1} = A_{1,1}B_{1,1} + A_{1,2}B_{2,1}\)
Task 2: \(C_{1,2} = A_{1,1}B_{1,2} + A_{1,2}B_{2,2}\)
Task 3: \(C_{2,1} = A_{2,1}B_{1,1} + A_{2,2}B_{2,1}\)
Task 4: \(C_{2,2} = A_{2,1}B_{1,2} + A_{2,2}B_{2,2}\)

(b)

Figure 3.10  (a) Partitioning of input and output matrices into \(2 \times 2\) submatrices. (b) A decomposition of matrix multiplication into four tasks based on the partitioning of the matrices in (a).
<table>
<thead>
<tr>
<th>Decomposition I</th>
<th>Decomposition II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1: $C_{1,1} = A_{1,1}B_{1,1}$</td>
<td>Task 1: $C_{1,1} = A_{1,1}B_{1,1}$</td>
</tr>
<tr>
<td>Task 2: $C_{1,1} = C_{1,1} + A_{1,2}B_{2,1}$</td>
<td>Task 2: $C_{1,1} = C_{1,1} + A_{1,2}B_{2,1}$</td>
</tr>
<tr>
<td>Task 3: $C_{1,2} = A_{1,1}B_{1,2}$</td>
<td>Task 3: $C_{1,2} = A_{1,2}B_{2,2}$</td>
</tr>
<tr>
<td>Task 4: $C_{1,2} = C_{1,2} + A_{1,2}B_{2,2}$</td>
<td>Task 4: $C_{1,2} = C_{1,2} + A_{1,1}B_{1,2}$</td>
</tr>
<tr>
<td>Task 5: $C_{2,1} = A_{2,1}B_{1,1}$</td>
<td>Task 5: $C_{2,1} = A_{2,2}B_{2,1}$</td>
</tr>
<tr>
<td>Task 6: $C_{2,1} = C_{2,1} + A_{2,2}B_{2,1}$</td>
<td>Task 6: $C_{2,1} = C_{2,1} + A_{2,1}B_{1,1}$</td>
</tr>
<tr>
<td>Task 7: $C_{2,2} = A_{2,1}B_{1,2}$</td>
<td>Task 7: $C_{2,2} = A_{2,1}B_{1,2}$</td>
</tr>
<tr>
<td>Task 8: $C_{2,2} = C_{2,2} + A_{2,2}B_{2,2}$</td>
<td>Task 8: $C_{2,2} = C_{2,2} + A_{2,2}B_{2,2}$</td>
</tr>
</tbody>
</table>

Figure 3.11 Two examples of decomposition of matrix multiplication into eight tasks.
Figure 3.12  Computing itemset frequencies in a transaction database.
(a) Partitioning the transactions among the tasks

<table>
<thead>
<tr>
<th>Itemsets</th>
<th>Itemset Frequency</th>
<th>Database Transactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>A, B, C, E, G, H</td>
<td>1</td>
<td>A, B, C</td>
</tr>
<tr>
<td>B, D, E, F, K, L</td>
<td>2</td>
<td>D, E</td>
</tr>
<tr>
<td>A, B, F, H, L</td>
<td>1</td>
<td>C, F, G</td>
</tr>
<tr>
<td>D, E, F, H</td>
<td>1</td>
<td>A, E</td>
</tr>
<tr>
<td>F, G, H, K</td>
<td>0</td>
<td>C, D</td>
</tr>
</tbody>
</table>

(b) Partitioning both transactions and frequencies among the tasks

<table>
<thead>
<tr>
<th>Itemsets</th>
<th>Itemset Frequency</th>
<th>Database Transactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>A, B, C, E, G, H</td>
<td>1</td>
<td>A, B, C</td>
</tr>
<tr>
<td>B, D, E, F, K, L</td>
<td>2</td>
<td>D, E</td>
</tr>
<tr>
<td>A, B, F, H, L</td>
<td>1</td>
<td>C, F, G</td>
</tr>
<tr>
<td>D, E, F, H</td>
<td>1</td>
<td>A, E</td>
</tr>
<tr>
<td>F, G, H, K</td>
<td>0</td>
<td>C, D</td>
</tr>
</tbody>
</table>

Figure 3.13 Some decompositions for computing itemset frequencies in a transaction database.
Figure 3.14  Multiplication of matrices $A$ and $B$ with partitioning of the three-dimensional intermediate matrix $D$. 
Stage I

\[
\begin{pmatrix}
A_{1,1} & A_{1,2} \\
A_{2,1} & A_{2,2}
\end{pmatrix} \cdot \begin{pmatrix}
B_{1,1} & B_{1,2} \\
B_{2,1} & B_{2,2}
\end{pmatrix} \rightarrow \begin{pmatrix}
D_{1,1,1} & D_{1,1,2} \\
D_{1,2,1} & D_{1,2,2} \\
D_{2,1,1} & D_{2,1,2} \\
D_{2,2,1} & D_{2,2,2}
\end{pmatrix}
\]

Stage II

\[
\begin{pmatrix}
D_{1,1,1} & D_{1,1,2} \\
D_{1,2,1} & D_{1,2,2}
\end{pmatrix} + \begin{pmatrix}
D_{2,1,1,1} & D_{2,1,1,2} \\
D_{2,2,1,1} & D_{2,2,1,2}
\end{pmatrix} \rightarrow \begin{pmatrix}
C_{1,1} & C_{1,2} \\
C_{2,1} & C_{2,2}
\end{pmatrix}
\]

A decomposition induced by a partitioning of \( D \)

- Task 01: \( D_{1,1,1} = A_{1,1}B_{1,1} \)
- Task 02: \( D_{2,1,1} = A_{1,2}B_{2,1} \)
- Task 03: \( D_{1,1,2} = A_{1,1}B_{1,2} \)
- Task 04: \( D_{2,1,2} = A_{1,2}B_{2,2} \)
- Task 05: \( D_{1,2,1} = A_{2,1}B_{1,1} \)
- Task 06: \( D_{2,2,1} = A_{2,2}B_{2,1} \)
- Task 07: \( D_{1,2,2} = A_{2,1}B_{1,2} \)
- Task 08: \( D_{2,2,2} = A_{2,2}B_{2,2} \)
- Task 09: \( C_{1,1} = D_{1,1,1} + D_{2,1,1} \)
- Task 10: \( C_{1,2} = D_{1,1,2} + D_{2,1,2} \)
- Task 11: \( C_{2,1} = D_{1,2,1} + D_{2,2,1} \)
- Task 12: \( C_{2,2} = D_{1,2,2} + D_{2,2,2} \)

**Figure 3.15** A decomposition of matrix multiplication based on partitioning the intermediate three-dimensional matrix.
Figure 3.16  The task-dependency graph of the decomposition shown in Figure 3.15.
Figure 3.17  A 15-puzzle problem instance showing the initial configuration (a), the final configuration (d), and a sequence of moves leading from the initial to the final configuration.
Figure 3.18  The states generated by an instance of the 15-puzzle problem.
Figure 3.19  An illustration of anomalous speedups resulting from exploratory decomposition.
Figure 3.20  A simple network for discrete event simulation.
Figure 3.21  Hybrid decomposition for finding the minimum of an array of size 16 using four tasks.
Figure 3.22  The regular two-dimensional task-interaction graph for image dithering. The pixels with dotted outline require color values from the boundary pixels of the neighboring tasks.
Figure 3.23 Two mappings of a hypothetical decomposition with a synchronization.
Figure 3.24 Examples of one-dimensional partitioning of an array among eight processes.
Figure 3.25  Examples of two-dimensional distributions of an array, (a) on a $4 \times 4$ process grid, and (b) on a $2 \times 8$ process grid.
Figure 3.26  Data sharing needed for matrix multiplication with (a) one-dimensional and (b) two-dimensional partitioning of the output matrix. Shaded portions of the input matrices $A$ and $B$ are required by the process that computes the shaded portion of the output matrix $C$. 
\[
\begin{pmatrix}
A_{1,1} & A_{1,2} & A_{1,3} \\
A_{2,1} & A_{2,2} & A_{2,3} \\
A_{3,1} & A_{3,2} & A_{3,3}
\end{pmatrix}
\rightarrow
\begin{pmatrix}
L_{1,1} & 0 & 0 \\
L_{2,1} & L_{2,2} & 0 \\
L_{3,1} & L_{3,2} & L_{3,3}
\end{pmatrix}
\cdot
\begin{pmatrix}
U_{1,1} & U_{1,2} & U_{1,3} \\
0 & U_{2,2} & U_{2,3} \\
0 & 0 & U_{3,3}
\end{pmatrix}
\]

1: \( A_{1,1} \rightarrow L_{1,1}U_{1,1} \)
2: \( L_{2,1} = A_{2,1}U_{1,1}^{-1} \)
3: \( L_{3,1} = A_{3,1}U_{1,1}^{-1} \)
4: \( U_{1,2} = L_{1,1}^{-1}A_{1,2} \)
5: \( U_{1,3} = L_{1,1}^{-1}A_{1,3} \)
6: \( A_{2,2} = A_{2,2} - L_{2,1}U_{1,2} \)
7: \( A_{3,2} = A_{3,2} - L_{3,1}U_{1,2} \)
8: \( A_{2,3} = A_{2,3} - L_{2,1}U_{1,3} \)
9: \( A_{3,3} = A_{3,3} - L_{3,1}U_{1,3} \)
10: \( A_{2,2} \rightarrow L_{2,2}U_{2,2} \)
11: \( L_{3,2} = A_{3,2}U_{2,2}^{-1} \)
12: \( U_{2,3} = L_{2,2}^{-1}A_{2,3} \)
13: \( A_{3,3} = A_{3,3} - L_{3,2}U_{2,3} \)
14: \( A_{3,3} \rightarrow L_{3,3}U_{3,3} \)

**Figure 3.27** A decomposition of LU factorization into 14 tasks.
Figure 3.28 A typical computation in Gaussian elimination and the active part of the coefficient matrix during the $k$th iteration of the outer loop.
Figure 3.29 A naive mapping of LU factorization tasks onto processes based on a two-dimensional block distribution.
Figure 3.30  Examples of one- and two-dimensional block-cyclic distributions among four processes. (a) The rows of the array are grouped into blocks each consisting of two rows, resulting in eight blocks of rows. These blocks are distributed to four processes in a wraparound fashion. (b) The matrix is blocked into 16 blocks each of size $4 \times 4$, and it is mapped onto a $2 \times 2$ grid of processes in a wraparound fashion.
Figure 3.31 Using the block-cyclic distribution shown in (b) to distribute the computations performed in array (a) will lead to load imbalances.
Figure 3.32  A one-dimensional randomized block mapping of 12 blocks onto four process (i.e., \( \alpha = 3 \)).
Figure 3.33 Using a two-dimensional random block distribution shown in (b) to distribute the computations performed in array (a), as shown in (c).
Figure 3.34  A mesh used to model Lake Superior.
Figure 3.35  A random distribution of the mesh elements to eight processes.
Figure 3.36  A distribution of the mesh elements to eight processes, by using a graph-partitioning algorithm.
Figure 3.37  Mapping of a binary tree task-dependency graph onto a hypercube of processes.
Figure 3.38  A mapping for sparse matrix-vector multiplication onto three processes. The list $C_i$ contains the indices of $b$ that Process $i$ needs to access from other processes.
Figure 3.39 Reducing interaction overhead in sparse matrix-vector multiplication by partitioning the task-interaction graph.
Figure 3.40  An example of hierarchical mapping of a task-dependency graph. Each node represented by an array is a supertask. The partitioning of the arrays represents subtasks, which are mapped onto eight processes.
Figure 3.41  Illustration of overlapping interactions in broadcasting data from one to four processes.
Figure 3.42  Task-dependency graphs for Problem ??.