



Figure 14.4 The subproblem graph for the rod-cutting problem with $n = 4$. The vertex labels give the sizes of the corresponding subproblems. A directed edge (x, y) indicates that solving subproblem x requires a solution to subproblem y . This graph is a reduced version of the recursion tree of Figure 14.3, in which all nodes with the same label are collapsed into a single vertex and all edges go from parent to child.

loop structure. The number of iterations of its inner **for** loop, in lines 5–6, forms an arithmetic series. The running time of its top-down counterpart, MEMOIZED-CUT-ROD, is also $\Theta(n^2)$, although this running time may be a little harder to see. Because a recursive call to solve a previously solved subproblem returns immediately, MEMOIZED-CUT-ROD solves each subproblem just once. It solves subproblems for sizes $0, 1, \dots, n$. To solve a subproblem of size n , the **for** loop of lines 6–7 iterates n times. Thus, the total number of iterations of this **for** loop, over all recursive calls of MEMOIZED-CUT-ROD, forms an arithmetic series, giving a total of $\Theta(n^2)$ iterations, just like the inner **for** loop of BOTTOM-UP-CUT-ROD. (We actually are using a form of aggregate analysis here. We'll see aggregate analysis in detail in Section 16.1.)

Subproblem graphs

When you think about a dynamic-programming problem, you need to understand the set of subproblems involved and how subproblems depend on one another.

The *subproblem graph* for the problem embodies exactly this information. Figure 14.4 shows the subproblem graph for the rod-cutting problem with $n = 4$. It is a directed graph, containing one vertex for each distinct subproblem. The subproblem graph has a directed edge from the vertex for subproblem x to the vertex for subproblem y if determining an optimal solution for subproblem x involves directly considering an optimal solution for subproblem y . For example, the subproblem graph contains an edge from x to y if a top-down recursive procedure for solving x directly calls itself to solve y . You can think of the subproblem graph as