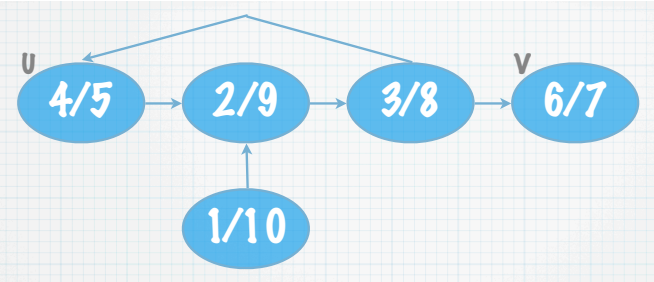
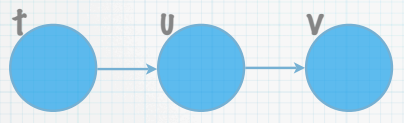
1. Problem 22.3-7.
   This figure shows a graph with discovery times and finishing times marked.



   The depth-first search starts at the bottom vertex. Note that there is a $u - v$ path, $d[u] < d[v]$, but v is not a descendant of u in the DFS tree.

2. Problem 22.3-8.
   The example above works as a counterexample for this too. Note $d[v] = 6 > 5 = f[u]$.

3. Problem 22.3-10.



   The depth-first search may start at v, proceed to u, and then to t. Each vertex will be its own forest and both edges are cross edges. This situation would occur in any graph that has a vertex such as u, where all the nodes reachable from u are visited before u and there are no edges back to vertices that can reach u.

4. Problem 22.4-3.
   We use depth-first search. The basic idea is to run DFS, but halt as soon an edge back to an already-visited vertex is found. This way we never check more than $|V|$ edges. We can fix the DFS-Visit algorithm on page 541 of CLRS by adding an **else** clause right after line 7 that says something to the effect of "**else** immediately halt and output that there is a cycle." If the algorithm completes without finding any back edges, then the graph has no cycle, and there are at most $|V| - 1$ edges, so $O(V + E) = O(V)$. Thus we have $O(V)$ in all cases.

5. The number of scalar multiplications needed to multiply two 4×4 matrices is 7 times the number needed to multiply two 2×2 matrices. The number needed to multiply two 2×2 matrices is 7 times the number needed for the 1×1 case, which is just 1. So the total is $7 \cdot 7 \cdot 1 = 7^{\lg n} = n^{\lg 7} = 49$.

6. We can use Theorem 30.7 to answer this question very easily.

$$V_\omega^{-1} = \frac{1}{4}\begin{bmatrix} \omega^0 & \omega^0 & \omega^0 & \omega^0 \\ \omega^0 & \omega^{-1} & \omega^{-2} & \omega^{-3} \\ \omega^0 & \omega^{-2} & \omega^{-4} & \omega^{-6} \\ \omega^0 & \omega^{-3} & \omega^{-6} & \omega^{-9} \end{bmatrix} = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{-i}{4} & \frac{-1}{4} & \frac{i}{4} \\ \frac{1}{4} & \frac{-1}{4} & \frac{1}{4} & \frac{-1}{4} \\ \frac{1}{4} & \frac{i}{4} & \frac{-1}{4} & \frac{-i}{4} \end{bmatrix}.$$

7. Let C be the result of multiplying the two polynomials A and B. To obtain C using the FFT, we do the following: 1. Convert A and B to point form y^A and y^B, using the divide-and-conquer recursive approach so that the conversion is fast. 2. Multiply the point forms to get y^C. 3. Convert y^C back to standard polynomial form using the inverse FFT (including the multiplication by $1/n$).

$a = (1, 3, 0, 0)$

$a^{[0]} = (1, 0)$ $\qquad$ $a^{[1]} = (3, 0)$

$a^{[0][0]} = (1)$ $\quad a^{[0][1]} = (0)$ $\qquad$ $a^{[1][0]} = (3)$ $\quad a^{[1][1]} = (0)$

$y^{[0][0]} = (1)$ $\quad y^{[0][1]} = (0)$ $\qquad$ $y^{[1][0]} = (3)$ $\quad y^{[1][1]} = (0)$

$y^{[0]} = (1 + 1 \cdot 0, 1 - 1 \cdot 0) = (1, 1)$ $\qquad$ $y^{[1]} = (3 + 1 \cdot 0, 3 - 1 \cdot 0) = (3, 3)$

$y^A = (1 + 1 \cdot 3, 1 + i \cdot 3, 1 - 1 \cdot 3, 1 - i \cdot 3) = (4, 1 + 3i, -2, 1 - 3i)$

$b = (1, -2, 0, 0)$

$b^{[0]} = (1, 0)$ $\qquad$ $b^{[1]} = (-2, 0)$

$b^{[0][0]} = (1)$ $\quad b^{[0][1]} = (0)$ $\qquad$ $b^{[1][0]} = (-2)$ $\quad b^{[1][1]} = (0)$

$y^{[0][0]} = (1)$ $\quad y^{[0][1]} = (0)$ $\qquad$ $y^{[1][0]} = (-2)$ $\quad y^{[1][1]} = (0)$

$y^{[0]} = (1 + 1 \cdot 0, 1 - 1 \cdot 0) = (1, 1)$ $\qquad$ $y^{[1]} = (-2 + 1 \cdot 0, -2 - 1 \cdot 0) = (3, 3)$

$y^B = (1 + 1 \cdot (-2), 1 + i \cdot (-2), 1 - 1 \cdot (-2), 1 - i \cdot (-2)) = (-1, 1 - 2i, 3, 1 + 2i)$

$y^C = y^A \cdot y^B = (-4, 7 + i, -6, 7 - i)$

$y = (-4, 7 + i, -6, 7 - i)$

$y^{[0]} = (-4, -6)$ $\qquad$ $y^{[1]} = (7 + i, 7 - i)$

$y^{[0][0]} = (-4)$ $\quad y^{[0][1]} = (-6)$ $\qquad$ $y^{[1][0]} = (7{+}1)$ $\; y^{[1][1]} = (7 - i)$

$c^{[0][0]} = (-4)$ $\quad c^{[0][1]} = (-6)$ $\qquad$ $c^{[1][0]} = (7{+}1)$ $\; c^{[1][1]} = (7 - i)$

$$\begin{aligned} c^{[0]} &= (-4 + 1 \cdot (-6), -4 - 1 \cdot (-6)) \\ &= (-10, 2) \end{aligned} \qquad \begin{aligned} c^{[1]} &= (7 + i + 1 \cdot (7 - i), 7 + i - 1 \cdot (7 - i)) \\ &= (14, 2i) \end{aligned}$$

$c = (-10 + 1 \cdot 14, 2 + (-i) \cdot (2i), -10 - 1 \cdot 14, 2 - (-i) \cdot (2i))/4 = (4, 4, -24, 0)/4$

$C = 1 + X - 6X^2$

8. The recursive FFT algorithm constructs a vector that represents the input polynomial as a sequence of points. The points are the polynomial evaluated at all the nth roots of unity. Lines 11–13 use the results of the lemmas in the text to fill in this vector correctly and efficiently. Basically, the vector is filled in starting at the beginning and at the middle. It would have worked just as well to eliminate line 12 and run the loop from $k \leftarrow 0$ **to** $n-1$. Both methods are equivalent because $\omega\omega_n^{n/2} = -\omega$ by Corollary 30.4.