that the $LS$ and $ES$ computations are done once for each action, and each computation iterates over at most $b$ other actions.) Therefore, finding a minimum-duration schedule, given a partial ordering on the actions and no resource constraints, is quite easy.

Mathematically speaking, critical-path problems are easy to solve because they are defined as a conjunction of linear inequalities on the start and end times. When we introduce resource constraints, the resulting constraints on start and end times become more complicated. For example, the $AddEngine$ actions, which begin at the same time in Figure 11.14, require the same $EngineHoist$ and so cannot overlap. The “cannot overlap” constraint is a disjunction of two linear inequalities, one for each possible ordering. The introduction of disjunctions turns out to make scheduling with resource constraints NP-hard.

Figure 11.15 shows the solution with the fastest completion time, 115 minutes. This is 30 minutes longer than the 85 minutes required for a schedule without resource constraints. Notice that there is no time at which both inspectors are required, so we can immediately move one of our two inspectors to a more productive position.

There is a long history of work on optimal scheduling. A challenge problem posed in 1963—to find the optimal schedule for a problem involving just 10 machines and 10 jobs of 100 actions each—went unsolved for 23 years (Lawler et al., 1993). Many approaches have been tried, including branch-and-bound, simulated annealing, tabu search, and constraint sat-