This homework covers developing backtracking algorithms, including pruning and branch-and-bound. It is due in class Friday, April 25.

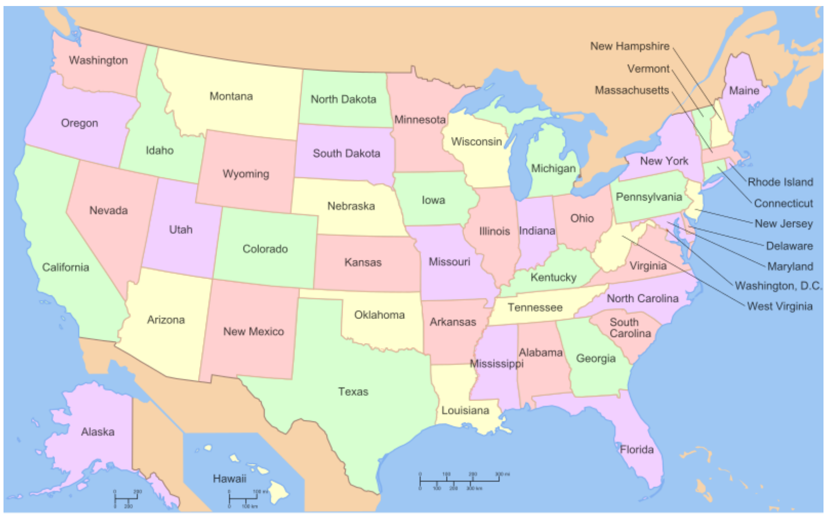
Write your solutions carefully — your work should be neat, readable, organized, and polished. As with writing a paper, you will likely need at least two drafts — the first draft can be rough (similar to what was written in class) as the focus is on going through the steps and finding a solution, while in the final draft the focus is on clarity of explanation and the quality of writing (similar to what was written after class). While it is not required, you are encouraged to type your work so that the revisions for the final draft are easier to make.

See the Policies page on the course website for information about revise-and-resubmit, late work, and academic integrity as it applies to homework.

(Problems start on the next page.)

For the following problems, develop a backtracking algorithm to solve the problem using the process discussed in class. Give each of the steps in the template — don't just give an algorithm. This is not a research task to look up the solution — the point here is to understand and be able to apply the process discussed in class to develop an algorithm yourself.

1. Using just four colors, color a (geographical) map so that no two bordering regions have the same color — or determine that it isn't possible. The input to this problem can be represented as a graph — each region is represented by a vertex and bordering regions are represented by edges between the respective vertices.



2. You are planning a dinner party, with n guests to be seated at two tables. (Each table holds exactly half of the guests.) You also have a list of preferences — some pairs of guests would prefer to be seated at the same table, while other pairs prefer to be at different tables. (There may also be many pairs with no preference about tables.) List all the possible seating arrangements that respect the guests' preferences. (The order of seating around each table doesn't matter, just who is at which table.)

3. You have a set of n jobs to complete, and n contractors have each bid on each of the jobs. Let $c(i, j)$ be the cost of having contractor j complete job i. Find an assignment of jobs to contractors so that each job is assigned to one contractor, each contractor is assigned one job, and the total cost of getting all of the jobs done is minimized.

image credit: https://commons.wikimedia.org/wiki/File:Map_of_USA_with_state_names_2.svg

For the following problems, address how to make the backtracking solution more practical. Consider:

(a) What you can prune beyond just illegal alternatives for the next choice? How effective do you think this will be?

(b) Identify whether the bound function for a branch-and-bound algorithm should be an upper bound or a lower bound. What bound function(s) can you identify? Give at least one possible (safe) bound function — for the tightest bound you can come up with, but list any that you think of even if they are trivial. Explain why your bound(s) are safe. How effective do you think they will be?

(c) Identify whether the initial solution estimate should be an upper bound or a lower bound and give at least one possible (safe) initial solution estimate. Aim for the tightest estimate you can come up with, but list any that you think of even if they are trivial. Explain why your estimate(s) are safe. How effective do you think they will be?

---

4. The contractor assignment problem (#3).

5. The time-o problem (from the programming assignment). For pruning, since there aren't any illegal alternatives as such, consider what can be pruned beyond the basic if-past-the-last-time-window-go-to-finish pruning.