Algorithm A.8 Greedy hill-climbing search with random restarts

```
Procedure Search-with-Restarts (
  $\sigma_0$,   // initial candidate solution
  $\mathrm{score}$,   // Score
  $\mathcal{O}$,   // A set of search operators
  Search,   // Search procedure
  $l$,   // random restart length
  $k$   // number of random restarts
)
  $\sigma_{\mathrm{best}} \leftarrow$ Search($\sigma_0, \mathrm{score}, \mathcal{O}$)
  for $i = 1, \ldots, k$
    $\sigma \leftarrow \sigma_{\mathrm{best}}$
      // Perform random walk
    $j \leftarrow 1$
    while $j < l$
      sample $o$ from $\mathcal{O}$
      if $o(\sigma)$ is a legal network then
        $\sigma \leftarrow o(\sigma)$
        $j \leftarrow j + 1$
    $\sigma \leftarrow$ Search($\sigma, \mathrm{score}, \mathcal{O}$)
    if $\mathrm{score}(\sigma) > \mathrm{score}(\sigma_{\mathrm{best}})$ then
      $\sigma_{\mathrm{best}} \leftarrow \sigma$

  return $\sigma_{\mathrm{best}}$
```

to get a network that is fairly similar, yet perturbed. We then restart our search procedure from the new network. If we are lucky, this *random restart* step moves us to a network that belongs to a better "basin of attraction," and thus the search will converge to a better structure. A simple random restart procedure is shown in algorithm A.8; it can be applied as wrapper to plain hill climbing, tabu search, or any other search algorithm.

random restart

This approach can be effective in escaping from fairly local maxima (which can be though of as are small bumps on the slope of a larger hill). However, it is unlikely to move from one wide hill to another. There are different choices in applying random restart, the most important one is how many random "steps" to take. If we take too few, we are unlikely to escape the local maxima. If we take too many, than we move too far off from the region of high scoring network. One possible strategy is to applying random restarts of growing magnitude. That is, each successive random restart applies more random operations.

To make this method concrete, we need a way of determining how to apply random restarts, and how to interleave hill-climbing steps and randomized moves. A general framework for doing is *simulated annealing*. The basic idea of simulated annealing is similar to Metropolis-Hastings MCMC methods that we discuss in section 12.3, and so we only briefly touch it.

simulated annealing

In broad outline, the simulated annealing procedure attempts to mix hill-climbing steps with