13.6
Supplemental: Some experiments with memoization

Edit distance: different memoizations

Input size	Running time in seconds		
n	DP	Partial	Implicit memoization
$1,250$	0.01	0.04	0.20
$2,500$	0.04	0.15	0.84
$5,000$	0.18	0.64	3.73
$10,000$	0.72	2.50	15.05
$20,000$	2.88	9.91	55.35
$40,000$	12.00	40.00	out of memory

For the input n, two random strings of length n were generated, and their distance computed using edit distance.

Note, that edit-distance is simple enough to that DP gets very good performance. For more complicated problems, the advantage of DP would probably be much smaller.

The asymptotic running time here is $\Theta(n^2)$.

Edit distance: different memoizations

More details

1. The implementation was done in C++, using -O9 in compilation.
2. DP = Dynamic Programming = iterative implementation using arrays.
3. Partial memoization = Still uses recursive code, but remembers the results in tables that are managed directly by the code.
4. Implicit memoization = implemented using the standard `unordered_map`.

Edit distance: different memoizations

Conclusions

1. If you are in interview setup, you should probably solve the problem using DP. That what you would be expected to do.
2. Otherwise, I would probably implement partial memoization – it still has the simplicity of the recursive solution, while having a decent performance. If I really care about performance I would implement the DP.
3. Using implicit memoization probably makes sense only if running time is not really an issue.

THE END

...

(for now)