Proof of an Optimal substructure in Dynammic Programming?

Could someone please explain how exactly the proof of optimal substructure property in dynamic programming problems works?

They usually say:

> Let's say the global optimal solution is A, and B is part of the solution. So B must the optimal solution of the subproblem, because if it weren't, then A wouldn't be the global optimal.

It makes some sense to me, but I think this would work with any non-dynamic problem (the property), or maybe it is because I still don't get how it works.

Can someone please help? How does the contradiction work?

`proof-techniques` `dynamic-programming` `correctness-proof`

edited Sep 1 '15 at 21:02
Gilles ♦
31.7k 7 86 160

asked Jun 30 '14 at 18:05
user3791225
26 1 2

2 Step 1: understand that all you do is prove correctness of a recurrence. Step 2: Use induction. – Raphael ♦ Jun 30 '14 at 21:57

it would be nice to quote it exactly & cite a ref etc – vzn Jul 1 '14 at 23:22

2 Answers

There is no (one) formal definition of "optimal substructure" (or the Bellman optimality criterion) so you can not possibly hope to (formally) prove you have it.

You should do the following:

1. Set up your (candidate) dynamic programming *recurrence*.
2. Prove it correct by induction.
3. Formulate the (iterative, memoizing) algorithm following the recurrence.

answered Jun 30 '14 at 22:16
Raphael ♦
56k 22 137 303

2 That is the best description of dynamic programming I've seen. Most of definitions/descriptions are so vague that almost nonsensical. Because of that I strugled to understand the concept for veeery long time. – Trismegistos Sep 3 '15 at 14:02

Most students are indeed puzzled by this property: it looks obvious when they read it, but yet they can't manage to apply it to solve problems. First note that when you express the objective by recursion on the subproblems, the expression has to be monotonic and the subproblems must be independent. For instance, in the travelling salesman problem, you cannot solve it by a call to find the best path from A to B and another from B to A: the two subproblems have to decide how to divide the nodes between them, since the salesman cannot pass two times in the same node.

answered Jun 30 '14 at 20:05
Pys
134 2