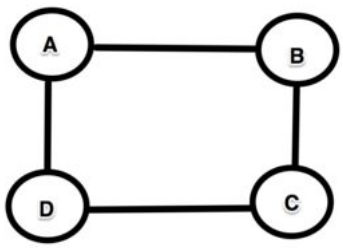
What is the intuition on why the longest path problem does not have optimal substructure?

I was learning about longest paths and came across the fact that longest paths in general graphs is not solvable by dynamic programming because the problem lacked optimal substructure (which I think the statement needs to be corrected to longest **simple** paths on general graphs is **not** solvable by dynamic programming).

If we assume they have to be **simple** (for some reason this is required which I don't appreciate yet) and longest, then its easy to create a counter example. Consider the square graph with 4 nodes A→B→C→D→A.



A longest path from A to D is clearly A→B→C-D. But the longest path from B to C is not part of the longest path to from A to D because the longest path from B to C is B→A→D→C which is clearly not the same as the path B→C (which in this case is in fact a shortest path!).

This seems to work only because we required the paths to be simple. Is it necessary to assume that the paths **must** be simple for us to prove that optimal substructure is **not** present?

I think that the counter example should be good evidence/proof (which I don't deny), I just don't find the counter example very enlightening at all. I see why it proves the point that it doesn't allow optimal substructure but it fails to provide me real understanding or intuition why it should be obvious that there should be **no** longest path optimal substructure. Like for example, why doesn't a cut and paste argument work? If the subpath isn't longest, then just stick in a longer path! It sounds very tempting, I mean, if we put in place something longer then of course it should get longer...though, this is for some reason **wrong**. Does the example actually in fact prove that **DP** can **never** solve longest (simple?) paths efficiently? I don't require for a general proof that its **not** in P (since thats might be asking for a P vs NP solution). I am just after a proof that its **not solvable by DP** (or at least very strong evidence that DP can never solve this longest paths problem or that the problem does **not** have optimal substructure).

I have definitively seen in wikipedia that the problem should be NP-Hard, which means that there is probably no fast algorithm. Not sure if that is the only type of evidence/intuition that exists to provide that optimal substructure should be obviously lacking (or if its not lacking, that it can't be used to make the problem faster). Is that the only way to show that it should **not** be solvable by a fast dynamic program?

Is the reason we require **simple** just because if we don't place that requirement the problem becomes trivial/uninteresting? In other words, if there is any cycle then one has solved the longest path problem to all nodes reachable from that cycle (by finding any path to that cycle). For the nodes that don't have any cycle reachable, we have a acyclic graph, which is solvable by DP (if the weights are positive?). In addition if there are cycles, we have automatically made things have optimal substructure (I think) because any longest path just is made up of longest path which covers two cases, 1 the paths through the cycle or 2 the paths through the DAG, which both contain optimal substructure. So the problem has become trivial without the requirement of simple paths? Or am I missing something or are there better explanations to why simple paths are required? Doesn't the algorithm in this paragraph show that **general** longest paths **are** solvable by DP?

I am also not 100% sure what requirements are needed to guarantee that DP can't be used. Is it necessary to negative edge weights, positive, un-weighted, directed, undirected...what are the requirements?

algorithms graphs optimization dynamic-programming

edited May 27 '16 at 19:41

Gilles ♦
31.7k 7 86 160

asked Apr 29 '16 at 2:58


Charlie Parker
958 7 23

1 "is not solvable by dynamic programming because the problem lacked optimal substructure (which I think the statement needs to be corrected to longest simple paths on general graphs is not solvable by dynamic programming). " -- neither "optimal substructure" nor "dynamic programming" are meaningful terms in a formal sense; there simply are not universally agreed-upon formal definitions for them. See also here and here. That means the proof you request does not exist. – Raphael ♦ May 27 '16 at 12:00

3 Answers

The longest path problem does have optimal substructure, and this can be used to improve the trivial $O(n!)$ algorithm to an $\tilde{O}(2^n)$ algorithm. First we have to generalize the problem:

> Generalized longest path: Given a graph $G = (V, E)$, two vertices $s, t \in V$, and a set of vertices $A \subseteq V \setminus \{s, t\}$, find the longest path in G from s to t avoiding the vertices in A.

Denoting the solution to this problem by $\ell(s, t, A)$ (the graph is understood), we have the recurrence

$$\ell(s,t,A) = 1 + \max_{\substack{x \notin A \cup \{s\}: \\ (s,x) \in E}} \ell(x,t,A \cup \{s\}) \qquad (s \neq t),$$

$$\ell(t,t,A) = 0.$$

If the maximum in the first case is over the empty set (that is, if there is no path from s to t avoiding A), we assign $\ell(s,t,A) = -\infty$.

You can solve this recurrence using dynamic programming in time $\tilde{O}(2^n)$.

While there is optimal substructure here, there are too many parameters to keep track of, and this is what makes the problem intractable.

answered Apr 29 '16 at 10:22

Yuval Filmus
181k 12 171 330

what does the tilde on top of the O mean? I am guessing it mean ur hiding polynomial terms in the big O notation. – Charlie Parker Apr 29 '16 at 22:40

@CharlieParker That's right. – Yuval Filmus Apr 30 '16 at 6:01

How do you enforce *simple* paths in this formulation? – Joost Aug 20 at 8:49

@Joost The recurrence already enforces this. – Yuval Filmus Aug 21 at 5:50

@YuvalFilmus, ah I missed the link that x becomes s in the next recurrence, and that A are the nodes that are already visited. – Joost Aug 21 at 9:13

First of all, the longest simple path is NP-hard and there is no doubt about that (as the Hamiltonian path reduces to it).

Secondly, if you consider the non-simple paths, then the problem does not make much sense, as the non-simple path goes over one edge/vertex more than once, thus it has a loop. As a result of having a loop in the path, the longest non-simple path is **infinite**.

edited May 27 '16 at 13:45

answered Apr 29 '16 at 4:55

orezvani
1,624 8 17

I've been thinking about the same in last 1 hour and came to the following conclusions:

i) Longest path in a graph without cycles has optimal substructure and so does the shortest path.

ii) Longest path without positive cycles has optimal substructure and so does a shortest path without negative cycles.

iii) Longest path with positive cycles and shortest path with negative cycles don't exist as we can loop around the cycle infinitely. So we look at simple paths.

iv) Longest simple path with positive cycles and shortest simple path with negative cycles is NP Hard.

DP solves (i) in O(V+E). Bellman-Ford solves (ii) in O(VE) and Djisktra helps in certain sub-cases of (ii).

Literature is confusing on this topic when longest path is just shortest path with weights negated and vice-versa. I see no reason to give blanket statements like longest path has no optimal sub structure but shortest path has etc.

edited Apr 30 at 15:48

answered Apr 30 at 9:49

Nitesh Bharadwaj
21 5