Does reachability belong to P?

Asked 8 years ago Active 7 years, 4 months ago Viewed 1k times

-2

Reachability is defined as follows: a digraph $G = (V, E)$ and two vertices $v, w \in V$. Is there a directed path from v to w in G?

Is it possible to write a polynomial time algorithm for it?

I asked this question on mathematics and got no answer by far.

complexity-theory graphs time-complexity

Share Cite Improve this question Follow

edited May 28 '13 at 7:11
Raphael ♦ 69.7k 27 160 354

asked May 26 '13 at 5:57
Gigili 2,113 3 20 29

3 This was fully answered by my comment at Mathematics a couple of minutes after you posted it: math.stackexchange.com/questions/401884/... – András Salamon May 26 '13 at 9:57

@AndrásSalamon: Thank you for your comment over there, but I wouldn't say it's fully answered since the answers different are completely different. Also the paper you linked to wasn't really related to what I asked. – Gigili May 26 '13 at 13:25

3 Strange that you ask such a question after receiving 18 upvotes for this answer: cs.stackexchange.com/a/308/6716 . OK, it is quoted, but nevertheless ... – frafl May 27 '13 at 21:53

3 Answers

Active Oldest Votes

6

Although you already know from the other answers that the question is solvable in polynomial time, I thought I would expand on the computational complexity of reachability since you used complexity terminology in your question.

Reachability (or st-connectivity) in digraphs is the prototypical NL-complete problem where NL stands for non-deterministic log space and we use deterministic log-space reductions (although I think it remains complete for NC^1 reductions, too).

- To see why it is in NL, notice that you can guess a next vertex at every step and verify that it is connected to the previous vertex. A series of correct guesses exists if and only if there is a path from s to t.
- To see why is NL-hard, notice that the behavior of a non-deterministic Turing machine can be represented by a configuration graph. The nondeterministic machine accepts only if there exists a path from the start configuration to an accept configuration, and if the machine only [...] is of size $O(|\Gamma|^{S(n)})$ where Γ is the tape alphabet. If $S(n)$ is logarithmic, then the [...]ollows.

[...]o why should I care? Well, we know lots of things about NL; one of those is that is in [...]re are tighter facts that can be useful:

[...]$\mathrm{ACE}((\log n)^2)$: even on a deterministic machine you don't need that much space to

[...]ircuit model, your question can be solved by a polynomial sized circuit of depth [...] that the problem is parallelizable since Nick's Class captures the idea of quick solutions

[...]it is not harder (up to log-space reductions) than membership checking in context-free

[...]e graph is undirected then we believe the question is significantly easier. In particular, [...]ic log space) under first-order reductions (Reingold 2004; pdf).

Share Cite Improve this answer Follow

answered Feb 6 '14 at 7:39
Artem Kaznatcheev
4,702 2 23 52

5

Yes, this problem is solveable in linear time, $O(|V| + |E|)$ to be precise.

The two classic solutions to this are Breadth-First search and Depth-First search.

The algorithms basically look like this:

```
current = v
while (current has an edge to an unmarked vertex)
    if current == w

        return true
    mark current as visited
    for each u where (v,u) is in E
        add u to the Open List
    current = a vertex from the Open List
return false
```

BFS uses a queue as the open list, adding to the back and taking from the front. DFS uses a stack. In any implementation like this, each node and each edge are visited at most once, so the algorithm runs in linear time.

Share Cite Improve this answer Follow

answered May 26 '13 at 6:07
jmite
28.6k 5 58 114

3

Yes, it is in P.

The natural algorithm for is very simple, so simple it doesn't really serve as a learning experience to state it here (it's readily available in almost any text or on the web).

To put you on the right path:

1. What algorithms do you know for exploring a graph?
2. If you were at the start vertex v, what's the obvious way of trying to find w?

Share Cite Improve this answer Follow

answered May 26 '13 at 6:07
Luke Mathieson
17.4k 4 50 83