are NP Complete languages closed under any regular operations?

I have tried looking online, but I couldn't find any definitive statements. It would make sense to me that Union and Intersection of two NPC languages would produce a language not necessarily in NPC. Is it also true that NPC languages are not closed under the complement, concatenation, and kleene star operations?

complexity-theory np-complete closure-properties

edited Apr 30 '14 at 17:19
Raphael ♦
47.7k 16 115 251

asked Apr 30 '14 at 16:34
user16742
65 2 4

1 just a note: regular operations are union, concatenation and Kleene star and **not** intersection and complement – A.Schulz Apr 30 '14 at 19:14

Why not intersection and complement? I haven't seen any formal definition of regular operations anywhere. – Tushar Apr 30 '14 at 19:18

@Tushar Indeed: union, concatenation and Kleene star are regular operations, whereas union, intersection and complement are Boolean operations. See wikipedia. – Hendrik Jan May 1 '14 at 0:12

@Tushar: Because these operations are used to build **regular** expressions. – A.Schulz May 1 '14 at 16:42

2 Answers

For all of the examples in this answer, I'm taking the alphabet to be $\{0,1\}$. Note that the languages $\emptyset$ and $\{0,1\}^*$ are definitely not **NP**-complete.

- The class of **NP**-complete languages is not closed under intersection. For any **NP**-complete language L, let $L_0 = \{0w \mid w \in L\}$ and $L_1 = \{1w \mid w \in L\}$. L_0 and L_1 are both **NP**-complete but $L_0 \cap L_1 = \emptyset$.
- The class of **NP**-complete languages is not closed under union. Given the **NP**-complete languages L_0 and L_1 from the previous part, let $L_0' = L_0 \cup \{1w \mid w \in \{0,1\}^*\} \cup \{\varepsilon\}$ and $L_1' = L_1 \cup \{0w \mid w \in \{0,1\}^*\} \cup \{\varepsilon\}$. L_0' and L_1' are both **NP**-complete but $L_0' \cup L_1' = \{0,1\}^*$.
- The class of **NP**-complete languages is not closed under concatenation. Consider the **NP**-complete languages L_0' and L_1' from the previous part. Since both languages contain ε, we have $L_0'L_1' \supseteq L_0' \cup L_1' = \{0,1\}^*$.
- The class of **NP**-complete languages is not closed under Kleene star. For any **NP**-complete language L, $L \cup \{0,1\}$ is **NP**-complete but $\left(L \cup \{0,1\}\right)^* = \{0,1\}^*$.
- If the class of **NP**-complete problems is closed under complementation, then **NP** = **coNP**. Whether this is true or not is one of the major open problems in complexity theory.

edited Apr 30 '14 at 22:26

answered Apr 30 '14 at 19:08
David Richerby
43.8k 9 63 121

I get L_0 and L_1 in intersection proof are NP-complete because I can reduce L_0 to L by stripping off the first zero in the language. I don't get why L_0' and L_1' in the union proof are NP-complete. – Tushar Apr 30 '14 at 19:25

@Tushar Your suggested edit was slightly wrong. L_0 and L_1 aren't *any* **NP**-complete problems: they're the two constructed in the previous part (I've now clarified this). The reduction from L to L_i' ($i \in \{0,1\}$) is the same as the reduction from L to L_i: you just add an i to the start of your input. – David Richerby Apr 30 '14 at 19:29

Oh, that was it! I though L_0 and L_1 were fresh NP-complete problems. Thank you! Great answer! – Tushar Apr 30 '14 at 19:43

Take a look at the proofs for union, intersection, concatenation, and kleene star of NP languages, here. Its seems like a similar argument could be made for NP-Complete languages.

For notation let

- A be a oracle that decides a known NP-Complete problem like 3-SAT. See the definition of turing reducible
- L_1 and L_2 are NP-Complete languages
- M_1 and M_2 are Turing machines that decide L_1 and L_2 using A.

- L_3 is $L_1 \cup L_2$
- M_3 is a turing machine that decides L_3

In the case of union from 1, we can create a new machine M_3 that decides L_3 by calling M_1 and M_2 as sub routines. In turn, each time M_1 or M_2 is called, A is also called. So M_3 decides L_3 using A. By the argument from 1, the running time of M_3 is in P and since it uses A as a subroutine, L_3 is NP-Complete. In other words, L_3 is NP-Complete for the same reason that L_1 and L_2 are NP-Complete.

The same argument can be made intersection and it looks like similar arguments could be made for concatenation, and kleene star.

In the case of compliment, it seems likely to be difficult to prove for the same reasons is difficult to prove compliment in NP.

edited Apr 13 at 12:48
Community♦
1

answered Apr 30 '14 at 18:04
joebloggs
30 2

NP-completeness is defined in terms of many-one reductions, not oracle reductions. Further, the NP-complete languages are definitely not closed under union or intersection. If they're closed under complement, then NP=coNP, which is a major open question. – David Richerby Apr 30 '14 at 18:54

In Stephen Cook's 1971 paper[1] which **defines** NP-Completeness he uses a Query machine which is the same concept as an Oracle. You should also check out the link above on turing reducibility. [1] chell.co.uk/media/product/_master/1/files/... – joebloggs Apr 30 '14 at 18:59

@joebloggs: I can see from your argument that union and intersection of two NP-Complete languages is NP. However, it still doesn't prove whether it is NP-complete. You have to reduce the union or intersection of two NP-complete decision problem to a NP-complete decision problem to show that. – Tushar Apr 30 '14 at 19:01

@DavidRicherby: You say that the NP-complete languages are definitely not closed under union or intersection. I am interested in looking at the proof for that. Do you have any references for that proof? – Tushar Apr 30 '14 at 19:04

1 @joebloggs: Your argument works for NP-languages, but NOT for NP-complete languages. To prove that a language L is Np-complete, you need to provide a polynomial reduction from L to a known NP-complete language. As for David's answer, P is closed under intersection, because both empty language and universal language are in P (hence, they are in NP too), but they are not NP-complete. Hope that makes it clear! – Tushar Apr 30 '14 at 19:46