Oq devemos fazer?

1. A definição do problema;
2. A classe a que pertence;
3. Heristicas utilizadas por algoritmos propostos para sua solução;
4. Algoritmos aproximados propostos para sua solução;
5. Aplicação em problemas reais;

Se o problema for Np-completo

1. Apresente a melhor solução encontrada;
2. Apresente heurísticas utilizadas para resolvê-lo;
3. Apresente aplicações em problemas reais;

Lembrar de:

1. O conceito de problemas de decisão x problemas de otimização;
2. O conceito de máquina de Turing determinística e não determinística;
3. O conceito de heurística e de algoritmos aproximados, ambos utilizados na resolução de problemas Np-completos;

Um pouco de pesquisa...

Na teoria da [complexidade computacional](https://pt.wikipedia.org/wiki/Complexidade_computacional), o **problema de satisfazibilidade booleana** (**SAT**) foi o primeiro problema identificado como pertencente à classe de complexidade [NP-completo](https://pt.wikipedia.org/wiki/NP-completo" \o "NP-completo). O problema de satisfatibilidade booleana é o problema de determinar se existe uma determinada [valoração](https://pt.wikipedia.org/w/index.php?title=Valora%C3%A7%C3%A3o&action=edit&redlink=1) para as variáveis de uma determinada [fórmula](https://pt.wikipedia.org/wiki/F%C3%B3rmula) booleana tal que esta valoração [satisfaça](https://pt.wikipedia.org/w/index.php?title=Satisfazibilidade&action=edit&redlink=1) esta fórmula em questão. Por exemplo, tomando {\displaystyle \ x\_{1},x\_{2},x\_{3},x\_{4}} como as variáveis booleanas e a expressão {\displaystyle (x\_{1}\lor \neg x\_{3}\lor x\_{4})\land (\neg x\_{2}\lor x\_{3}\lor \neg x\_{4})}, caso exista uma atribuição de valores de verdade para as variáveis da fórmula que torne a fórmula avaliada **VERDADEIRA**, esta fórmula é considera satisfazível, em contrapartida se nenhuma atribuição levou a uma avaliação da fórmula como verdadeira, ela é considerada insatisfazível. Para salientar a natureza binária deste problema, ele é referenciado freqüentemente como o problema de satisfazibilidade booleana ou proposicional. A sigla **SAT** também é geralmente utilizada para denotá-lo, com o entendimento implícito de que a função e suas variáveis recebem valores binários.

Referimos-nos a variáveis ou suas negações como [*literais*](https://pt.wikipedia.org/wiki/Literal).

 Se juntarmos um grupo de literais com o símbolo de disjunção (OR), nós formamos uma [*cláusula*](https://pt.wikipedia.org/wiki/Cl%C3%A1usula)

As fórmulas, portanto serão uma conjunção (AND) de cláusulas - esta é chamada [forma normal conjuntiva](https://pt.wikipedia.org/wiki/Forma_normal_conjuntiva) (FNC)

Se cada cláusula for limitada a no máximo três literais. Este último problema é chamado **3SAT**, **3CNFSAT**, ou **3-satisfazibilidade**.

Se nós restringirmos cada cláusula à no máximo dois literais, o problema resultante, 2SAT, é um problema **Polinomial**.

Se cada cláusula for uma [cláusula de Horn](https://pt.wikipedia.org/wiki/Cl%C3%A1usula_de_Horn), contendo no máximo um literal positivo, o problema resultante, satisfazibilidade de Horn, é um problema **P-completo**.

Na [teoria da complexidade computacional](https://pt.wikipedia.org/wiki/Complexidade_computacional), o **teorema de Cook-Levin**, também conhecido como **teorema de Cook**, afirma que o [problema de satisfatibilidade booleana](https://pt.wikipedia.org/wiki/Problema_de_satisfatibilidade_booleana) é [NP-completo](https://pt.wikipedia.org/wiki/NP-completo" \o "NP-completo). Isto é, qualquer problema em [NP](https://pt.wikipedia.org/wiki/NP_(complexidade)) pode ser [reduzido](https://pt.wikipedia.org/wiki/Redu%C3%A7%C3%A3o_(complexidade)) em tempo polinomial por uma [máquina de Turing determinística](https://pt.wikipedia.org/wiki/M%C3%A1quina_de_Turing) para o problema de determinar se uma fórmula booleana é satisfatível.

<https://www.youtube.com/watch?v=wEez6foOdlI>