COMPUTABILIDADE

- Kurt Gödel (1906-1978) provou em 1931 the Incompleteness Theorem. Ele mostrou que em qualquer lógica capaz de expressar as propriedades dos números naturais, existem expressões que são indecidíveis--ou seja, a sua verdade não pode ser estabelecida por qualquer algoritmo. O resultado fundamental de Gödel também pode ser interpretado como uma demonstração que certas funções sobre os inteiros não podem ser ***representadas por um algoritmo--ou seja, não podem ser computadas.***

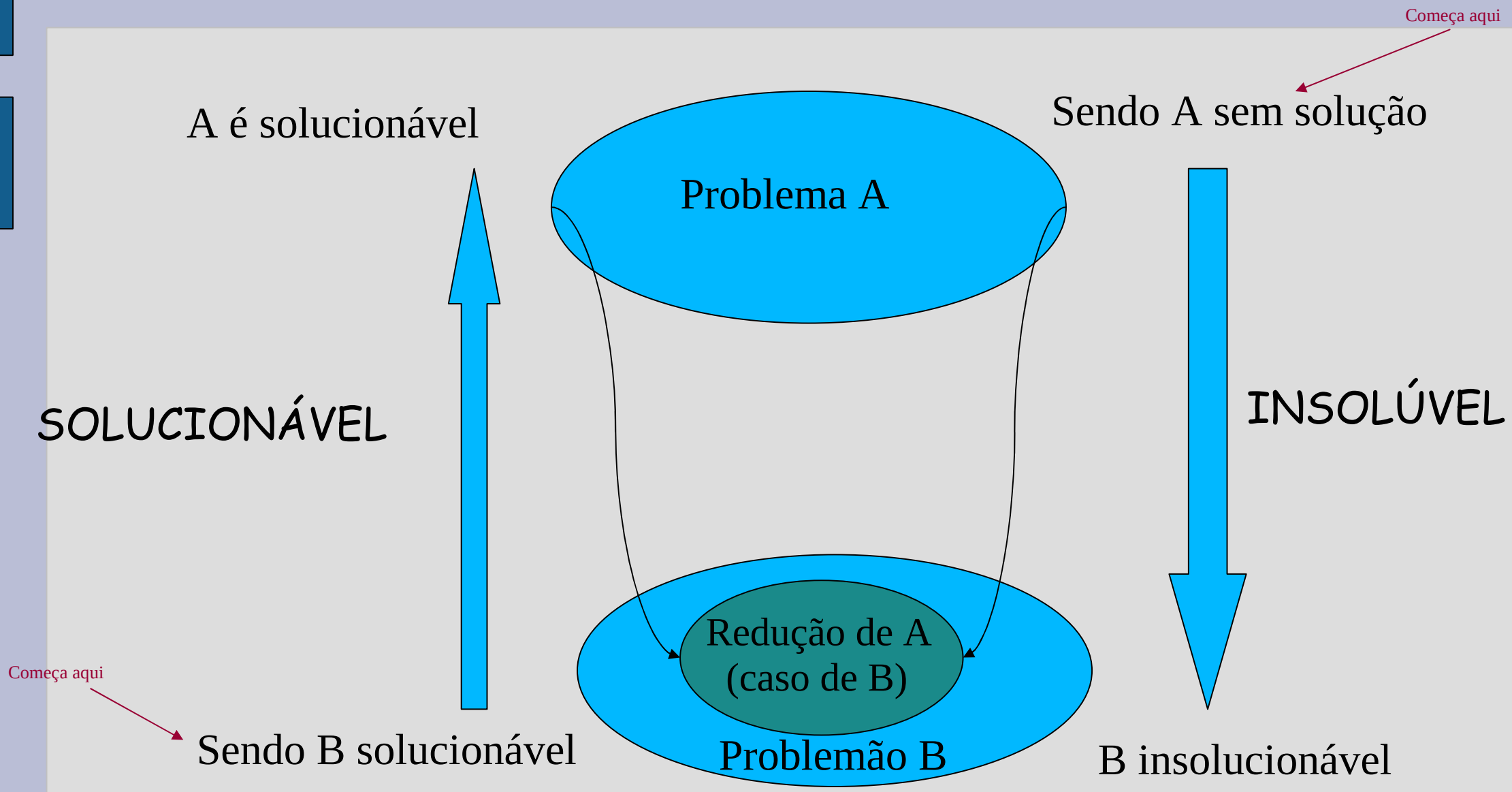
COMPUTABILIDADE

- Motivado pelos resultados de Gödel, Alan Turing (1912-1954) tentou caracterizar precisamente que funções podem ser computadas.
- A tese de Church-Turing estabelece que a máquina de Turing é capaz de computar qualquer função computável
- Embora não-decidibilidade e não-computabilidade sejam conceitos importantes para se entender computação, a noção de intratabilidade teve um impacto muito maior.
- A noção de intratabilidade é muito importante, no sentido de que crescimento exponencial implica na impossibilidade de se resolver certos problemas em um tempo aceitável
- Portanto, deve-se procurar dividir o problema geral de induzir comportamento inteligente em subproblemas tratáveis

COMPUTABILIDADE

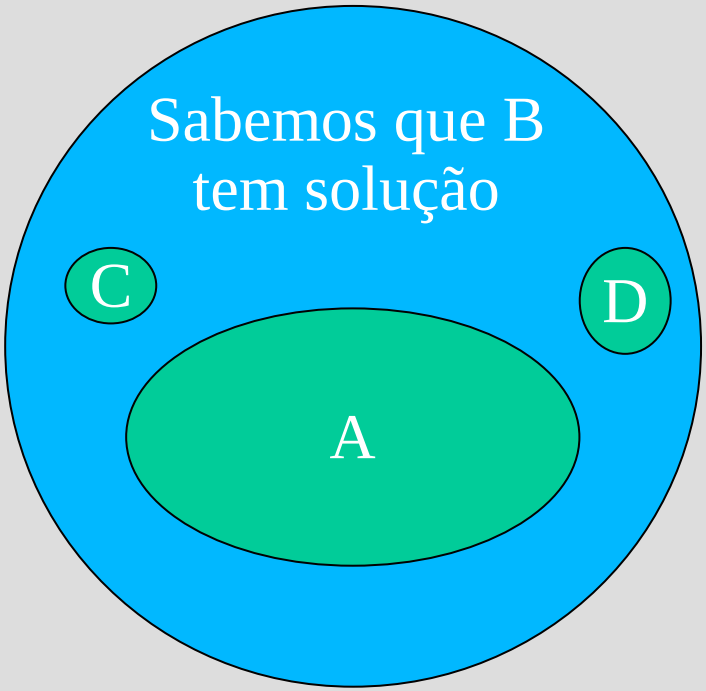
- De uma forma simples, uma classe de problemas é dita intratável se o tempo necessário para resolver instâncias cresce exponencialmente com o tamanho da instância
- Exemplo: (Ordenação de n números)
- Algoritmo 1
- Testar as T(n) = n! possibilidades; n! >= 2n
- Algoritmo 2
- Merge-sort T(n) = n log(n)

PRINCÍPIO DA REDUÇÃO



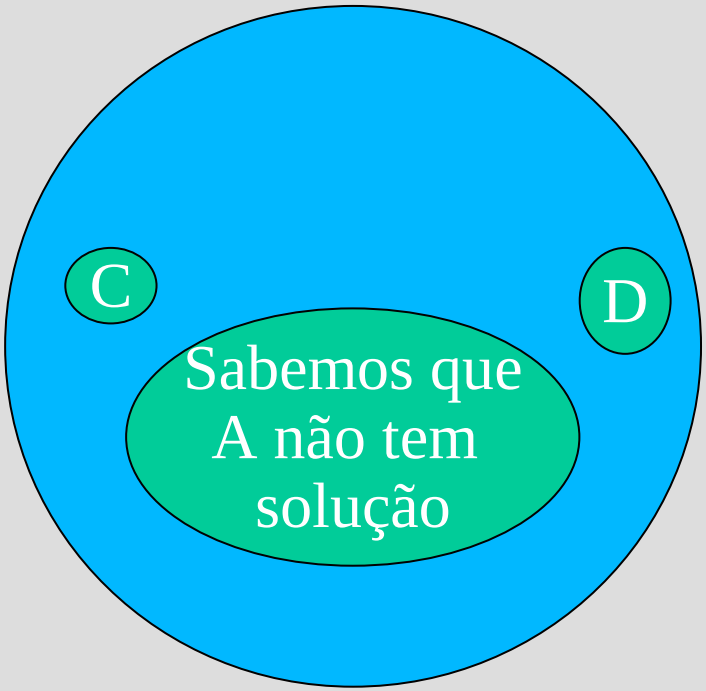
PRINCÍPIO DA REDUÇÃO

A tem solução?



Como A é um caso de B então A tem solução

B tem solução?



Como A é um caso de B então B não tem solução

MAS O QUE É UM PROBLEMA COMPUTÁVEL?

- **Um problema é computável se existe um procedimento que o resolve em um número finito de passos, ou seja se existe um algoritmo que leve à sua solução.**
- Observe que um problema considerado "em princípio" computável pode não ser **tratável** na prática, devido às limitações dos recursos computacionais para executar o algoritmo implementado.

PROBLEMA NP-COMPLETO

- Um problema não-deterministicamente polinomial(NP) é um problema computável cujas soluções até então conhecidas são de ordem exponencial e não se sabe se existe uma solução melhor, de complexidade polinomial.
- **Um problema NP-completo é um problema "representante" de uma classe de problemas NP, de tal sorte que os problemas da classe são redutíveis a ele em tempo polinomial. Por exemplo, o problema de caixeiro-viajante é um poblema redutível ao problema de ciclo hamiltoniano. Dizemos então que o problema de ciclo hamiltoniano é um problema NP-completo.**
- Foi demonstrado que, se um algoritmo de complexidade polinomial puder ser encontrado para qualquer um dos problemas NP-completos, então todos os problemas NP-completos serão na verdade problemas P. Por outro lado, se for provado que um deles requer um algoritmo de solução que apresente complexidade exponencial, então todos irão requerer complexidade exponencial.