Complexidade Parametrizada

Matheus Souza D'Andrea Alves

2018.2

Introdução

Complexidade clássica

Suponha um problema Π, se Π possui um algoritmo que o resolve em tempo polinomial dizemos que $\Pi \in P$.

Se dado um certificado de resposta sim para Π se posso validar tal resposta em tempo polinomial então $\Pi \in NP$.

Chamamos de $NP-Completo$ a classe de problemas Π' no qual dado o problema $3-SAT\ 3S\ 3s \propto \Pi'$.

Complexidade Parametrizada

O que traz de diferente?

Une teoria e prática, não ignorando nuances práticas de um problema sabidamente NP-Completo, resolve problemas. Difere de heurísticas e aproximações, pois não perde garantia de tempo ou otimalidade.

O objetivo é desenvolver um algoritmo $\mathcal{O}(f(k).n^{\mathcal{O}(1)})$. Se um problema Π admite uma solução dessa forma, dizemos que $\Pi \in FPT$. Quando isso ocorre, afirmamos que existe um pré-processesamento capaz de reduzir a entrada obtendo uma instância menor, limitado por k, chamamos isso de kernelização; Uma solução para o kernel é uma solução para o problema.

Características - $FPT \subset XP$

Técnicas de FPT

Bound search Tree

Kernelização

Dizemos que se um problema possui um kernel, então por definição o mesmo é FPT.

Pré processamento

chamamos de pré processamento um conjunto de regras que aplicados a uma instancia do problema, pode ou não retirar composições da entrada de forma a podar a instãncia.

Suponha um problema Π, que tem entrada I um parametro k e uma questão q.

Uma kernelização é um algoritmo de pré processamento que recebe I como entrada com o paramêtro k, e retorna uma instância do problema e um inteiro representatne do paramêtro.

$$f(k) \mapsto |I'|$$
$$g(k) \mapsto k'$$

O quão bom pode ser?

Depende de como se planeja abordar o problema, suponha dois algoritmos de kernelização, A e B; Se A chega a um kernel de tamanho $\theta(k)$ em tempo $\theta(n^2)$, e B chega a um tamanho $\theta(k^2)$ em tempo $\theta(n)$ na literatura, diríamos que A é *melhor* pois nos dá um kernel menor.

Teorema 1: Se $(\Pi, k) \in FPT$ então (Π, k) admite um kernel.

Demonstração.
Como $(\Pi, k) \in FPT$ logo o mesmo pode ser resolvido por um algoritmo A em tempo $f(k)n^c$ para c constante.
Considere portanto a seguinte kernelização.

- Execute os n^{c+1} passos de A.
- O problema foi resolvido?
  - Sim:
  - Não: logo $f(k)n^c > n.c \implies f(k) > n$ logo a entrada já era um kernel.

□