MAC 722 – Complexidade Computacional

Quinta lista de exercícios – Entrega **24 de junho de 1998**

[Seja P um problema de decisão e $\mathcal{I}$ o seu conjunto de instâncias. Para $x \in \mathcal{I}$ denotaremos por $\mathrm{Max}(x)$ a magnitude do maior número ocorrendo em x, se nenhum número ocorre em x então $\mathrm{Max}(x) = 0$. Um algoritmo M (máquina de Turing determinística) que resolve o problema P é chamado de um *algoritmo pseudo-polinomial* para P se M opera em tempo limitada superiormente por um polinômio em $|x|$ e $\mathrm{Max}(x)$, para toda instância $x \in \mathcal{I}$. (Note que M é um algoritmo polinomial se M opera tempo limitada superiormente por um polinomio em $|x|$, para toda instância $x \in \mathcal{I}$.)

Nós dizemos que P é um *number problem* se não existe um polinômio p tal que

$$\mathrm{Max}(x) \leq p(|x|).]$$

Exercício 1
Seja P um problema NP-completo que não é um number problem. Mostre que se P pode ser resolvido por um algoritmo pseudo-polinomial então P = NP.

[Seja p um polínômio sobre os inteiros. Denotaremos por $\mathcal{I}_p$ o conjunto das instâncias de $\mathcal{I}$ que satisfazem

$$\mathrm{Max}(x) \leq p(|x|)$$

e seja P_p o problema P restrito às instâncias em $\mathcal{I}_p$. (Logo P_p não é um number problem.)]

Exercício 2
Mostre que se P pode ser resolvido por um algoritmo pseudo-polinomial então P_p pode ser resolvido por um algoritmo polinomial.

Exercício 3
Seja P um problema NP-completo no sentido forte (*in the strong sense*). Mostre que se P pode ser resolvido por um algoritmo pseudo-polinomial então P = NP.
Observação. Uma conseqüência deste exercício é (por exemplo) que KNAPSACK não é NP-completo no sentido forte.

[Seja M um algoritmo de aproximação para um problema de otimização A. Nós dizemos que o algoritmo M é um *approximation scheme* para A se para cada instância x de A e para cada ϵ $(0 < \epsilon < 1)$ dados temos que

$$\frac{|c(\mathrm{M}(x)) - \mathrm{OPT}(x)|}{\max\{\mathrm{OPT}(x), c(\mathrm{M}(x))\}} \leq \epsilon,$$

onde $c(\mathrm{M}(x))$ é um inteiro positivo representando o custo da solução $M(x)$. O termo "scheme" é usado pois na realidade M fornece vários algoritmo de aproximação para A—um para cada valor fixo de $\epsilon > 0$.

Nós dizemos que M é um *polynomial time approximation scheme* se o algoritmo de aproximação M_ϵ derivado a partir de M fixando-se ϵ opera em tempo polinomial. Nós dizemos que M é um *fully polinomial-time approximation scheme* se M opera em tempo polinomial em $|x|$ e $1/\epsilon$.]

Exercício 4
Assumindo que $\mathrm{P} \neq \mathrm{NP}$ mostre que não existe um algoritmo polinomial M para o KNAPSACK tal que

$$|\mathrm{M}(x) - \mathrm{OPT}(x)| \leq B$$

para uma constante fixa B.

Exercício 5
Seja A um problema de otimização e seja p um polinômio em duas variáveis tal que

$$\mathrm{OPT}(x) < p(|x|, \mathrm{Max}(x))$$

para toda instância x de A. Mostre que se existe um fully polynomial time approximation scheme para A então existe um algoritmo pseudo-polinomial para A.

Exercício 6
Seja A um problema de otimização e seja p um polinômio em duas variáveis tal que

$$\mathrm{OPT}(x) < p(|x|, \mathrm{Max}(x))$$

para toda instância x de A. Suponha que o problema de decisão P associado a A é NP-complete in the strong sense. Mostre que se A pode ser resolvido por um fully polynomial time approximation scheme então P = NP.

Observação. Uma conqüência deste exercício é que (por exemplo) se você encontrar o fully polynomial time approximation scheme para MAX 2-SAT então você terá provado que P = NP (para MAX 2-SAT temos que $\mathrm{OPT}(x) < |x|$) .