# Métodos de Apoio à Decisão
# Trabalho 2: *Problema de afetação*

João Pedro PEDROSO

DCC - FCUP, Abril de 2025

*Neste trabalho serão colocadas algumas perguntas com base no enunciado que se segue. No momento de submissão, nas aulas práticas de 24/ABR a 2/MAI, haverá um conjunto de questões no* Codex*, com os computadores configurados como nas aulas anteriores. O livro do AMPL e os slides desta disciplina estarão disponíveis para consulta.*

## Exercício 1

Depois de concluir o seu curso, juntamente com alguns colegas decidiu criar uma empresa chamada UBRbytes, com o objetivo de fazer a ponte entre empresas de software e programadores. A sua plataforma anunciará tarefas que as empresas de software pretendem concluir na semana seguinte, e os programadores anunciarão a sua disponibilidade para programar as componentes necessárias para essas tarefas. A missão da UBRbytes consiste em afetar programadores, com base nas suas competências, às competências exigidas para concluir cada tarefa. A margem entre o que as empresas de software pagam pelas tarefas concluídas e o que os programadores cobram será o seu lucro.

Por exemplo, pode ter disponíveis 3 programadores, capazes de programar C, Python e aplicações Web, conforme indicado na tabela a seguir, onde 1 significa compatível e "."/0 significa incompatível.

| | p1 | p2 | p3 |
|---|---|---|---|
| c | 1 | . | 1 |
| py | 1 | 1 | . |
| web | 1 | 1 | 1 |

Note que esta tabela pode ser lida como um parâmetro em dados AMPL atribuindo um zero a ".", por exemplo, se for declarada como:

```
param a {SKILLS, PROGRAMMERS} default 0;
```

Na próxima semana, cada um deles estará disponível o número de horas, ao custo por hora que se seguem:

| | hours | cost/hour |
|---|---|---|
| p1 | 4 | 5 |
| p2 | 10 | 5 |
| p3 | 7 | 5 |

Há 5 tarefas propostas por empresas de software para a próxima semana, ao seguinte preço:

| | |
|---|---|
| t1 | 250 |
| t2 | 150 |
| t3 | 200 |
| t4 | 100 |
| t5 | 250 |

O número de horas necessárias por competência de programação são:

| | c | py | web |
|---|---|---|---|
| t1 | 5 | . | 2 |
| t2 | . | 3 | . |
| t3 | . | 6 | 3 |
| t4 | 3 | 3 | 3 |
| t5 | 2 | 3 | 2 |

Essas tarefas não podem ser divididas: um programador só pode ser atribuído a uma tarefa se puder concluir todas as suas subtarefas. Por exemplo, o programador `p1` não pode usar as suas 4 horas para programação C na tarefa `t1`, pois a tarefa requer 5 horas de programação C mais 2 horas de programação web. Poderia, no entanto, executar a tarefa 2, que requer apenas programação em Python; nesse caso, teria 1 hora restante, que não poderia ser utilizada em nenhuma outra tarefa, pois todas exigem mais de 1 hora.

A solução ótima neste caso seria `p1` fazendo `t2`, `p2` fazendo a tarefa `t3` (pode programar Python e web, e tem disponível mais do que as 9 horas necessárias) e `p3` fazendo a tarefa `t1`. Tarefas `t4, t5` não são cobertas. O lucro ótimo é de 505 euros.

**Exercício 1.1:** Prepare um modelo, independente dos dados (para ser mais fácil resolver com dados diferentes), para resolver este problema.

## Exercício 2

Como forma de melhorar a satisfação do cliente, decidiu acompanhar a avaliação feita pelos clientes a cada um dos programadores; 5 é a melhor avaliação, 1 é a pior, 0 se não houver informação disponível. Por exemplo, poderia ser:

| | p1 | p2 | p3 |
|---|---|---|---|
| c | 5 | . | 5 |
| py | 5 | 3 | . |
| web | 1 | 5 | 1 |

**Exercício 2.1:** Resolva o problema para o caso em que sua empresa pretende maximizar a avaliação dos programadores atribuídos às tarefas que serão concluídas, somadas para todas as subtarefas e todos os programadores. Considere as mesmas restrições do Exercício 1 e uma restrição que impõe que o lucro não deve ser inferior ao obtido na solução ótima do Exercício 1.

## Exercício 3

Percebeu que há procura para tarefas que exigem mais horas de programação do que qualquer um dos programadores fornece sozinho. Embora idealmente todas as tarefas devam ser feitas pela mesma pessoa, pode propor a alguns programadores que formem uma equipa na semana seguinte e dividam o trabalho necessário para concluir uma tarefa. A sua aplicação deve decidir quantas horas cada um deles trabalha, em cada subtarefa, pois o pagamento a cada programador será nessa proporção.

**Exercício 3.1:** Resolva o problema de maximizar o lucro nesta situação.

**Exercício 3.2:** Resolva o problema de minimizar o número de pessoas na equipa com mais elementos (a maior equipa) sujeito a um lucro pelo menos tão bom como o determinado no Exercício 3.1.