# Algoritmos e Sistemas Distribuídos

Aula teórica 1

Ano lectivo 2014/2015

Mestrado Integrado em Engenharia Informática

# Corpo Docente

- Aulas teóricas, aulas práticas, responsável, regente, …
  - Prof. Rodrigo Rodrigues

# Aulas

- Teóricas
  - 1 aula de 2 horas / semana (terça, 14:00)
- Práticas e laboratório
  - 1 aula de 2 horas / semana (terça, 16:00)
  - Começam na próxima semana
  - Focadas em:
    - Apresentação do projecto e introdução ao C++
    - Apoio ao projecto
    - Avaliação do projecto
    - Exercícios práticos semelhantes aos do exame

# Bibliografia Principal

- Distributed Algorithms. Nancy Lynch. Morgan-Kaufmann, 1996.
- Introduction to Reliable and Secure Distributed Programming. C. Cachin, R. Guerraoui, L. Rodrigues. 2nd Edition, Springer, 2011.
  - Alternativamente podem consultar a primeira edição do livro: Introduction to Reliable Distributed Programming. Rachid Guerraoui and Luís Rodrigues. Springer, 2006.
- Slides da cadeira, disponibilizados no CLIP
  - Alguns slides foram gentilmente cedidos pelo Prof. Lorenzo Alvisi, UT Austin (baseado nos slides da cadeira CS380D)
  - Alguns slides baseiam-se na cadeira 6.852 leccionada pela Prof. Nancy Lynch do MIT, disponíveis no OpenCourseWare através de uma licença Creative Commons BY-NC-SA

# Avaliação

- Teórica: 2 testes (25%+25%) ou exame de recurso (50%)
  - Testes do ano passado são testes-tipo
  - Nota mínima de 8,5 valores
- Projecto / frequência (50%)
  - Nota mínima de 8,5 valores para ter acesso ao segundo teste / exame
- Nota final é aproximada ao valor inteiro mais próximo, mínima de 9,5 valores

# Projecto (frequência)

- Nota frequência = nota do projecto * nota do teste prático
- Projecto: grupos de 3
- Programado em C++
- Teste prático: individual, nota de 0 a 1, consiste numa pequena modificação ao projecto, realizado nas aulas práticas
- Ambos componentes divididos em 6 fases (5 checkpoints e uma entrega final)
- Peso das sucessivas fases: 15+15+15+15+20+20

# Flexibilidade na data de entrega das fases do projecto

- São dados a cada grupo 3 dias de folga (adiamento automático da data de entrega)
  - A intenção é precaver quaisquer imprevistos (por exemplo, problemas no acesso à internet na casa dos alunos, greves nos transportes, etc.)
  - Logo, os alunos não devem pedir mais adiamentos devido a qualquer situação imprevista, estes serão recusados
  - Dias de folga podem ser usados ao longo das 6 fases, mas só têm 3 dias NO TOTAL para gerir no somatório de todas as 6 fases
  - Qualquer entrega fora do prazo (e após ter usado os dias de folga) é penalizada automaticamente em 3 valores por dia (até um mínimo de zero valores)

# Datas aproximadas (a confirmar após coordenação entre cadeiras)

- Primeiro teste: 21 de Outubro
- Segundo teste: 2 de Dezembro
- Checkpoints / entrega final do projecto:

| Número | Saída do enunciado | Entrega |
|---|---|---|
| 1 | 16 de Setembro | 25 de Setembro |
| 2 | 26 de Setembro | 2 de Outubro |
| 3 | 3 de Outubro | 9 de Outubro |
| 4 | 10 de Outubro | 16 de Outubro |
| 5 | 22 de Outubro | 6 de Novembro |
| 6 (Final) | 7 de Novembro | 20 de Novembro |

# Programa

| Semana | Data | Tema da aula teórica |
|---|---|---|
| 1 | 9 Set | Introduction and course overview. |
| 2 | 16 Set | Distributed computing model and analysis methods. |
| 3 | 23 Set | Broadcast. Consensus in synchronous systems. |
| 4 | 30 set | Consensus: Efficiency in synchronous systems and impossibility (FLP). |
| 5 | 7 Out | Consensus (cont'd) Paxos. State machine replication. |
| 6 | 14 Out | Read/write replication and atomic semantics. Byzantine replication. |
| 7 | 21 Out | **Midterm quiz** |
| 8 | 28 Out. | Other consistency levels. Eventual consistency. |
| 9 | 4 Nov. | Eventual consistency (cont.) CAP theorem. Consistent hashing. |
| 10 | 11 Nov. | P2P: intro and routing. |
| 11 | 18 Nov. | P2P: systems and discussion. Other forms of scalability: GFS. |
| 12 | 25 Nov. | GFS (cont'd). Distributed transactions: 2PC. Spanner. |
| 13 | 2 Dez. | **Final quiz** |
| 14 | 9 Dez. | TBD |

# Questões sobre a organização da cadeira?

# O problema dos dois generais

- Dois generais Romanos têm de atacar um inimigo de forma coordenada
- Só podem comunicar por mensageiros que atravessam um vale cheio de perigos

# Regras do problema

- Ambos os generais têm uma preferência inicial (atacar ou retirada) e têm de chegar a acordo
- Se ambos tiverem a mesma preferência essa é a única decisão possível
- Os mensageiros podem ser fulminados ou demorar entre $]0,+\infty[$ a fazer o percurso
- Um dos generais pode sucumbir (juntamente com todas as suas tropas) de forma fulminante
  - Nesse caso não emite mais mensagens
- Os generais que não são fulminados têm sempre de decidir (em tempo finito)

# Solução?

- Qual o algoritmo que garantidamente resolve o problema para todos os possíveis padrões de preferências, falhas, e demoras no percurso dos vários mensageiros?

# Solução?

- Não há solução, o problema é impossível de resolver
- Argumento muito informal:
  - Supondo que A não recebe mensagens durante um período muito longo.
  - A a partir de certa altura tem de assumir que B falhou e decide pela sua preferência (e.g., atacar) para o caso de ambos quererem atacar.
  - Mas no entanto B estava vivo e as mensagens estavam lentas
  - Vice-versa: B pensa que A falhou e decide pela sua preferência (e.g., retirar) para o caso de ambos quererem retirar, mas no entanto A estava vivo e as mensagens estavam lentas
  - Ambos decidem coisas diferentes
- Prova rigorosa:
  - Esperem algumas semanas…

# Qual a relevância do problema?

- Veremos que este problema está no cerne do desenho dos sistemas de replicação de dados
- Teremos de contornar esta impossibilidade. Como?
- As diferentes formas de contornar a impossibilidade estão espelhadas nas soluções usadas pela Google, Yahoo, Amazon, Facebook, Microsoft
  - Vamos estudá-las nesta cadeira

# Na próxima semana

- Iremos definir as “regras do jogo”:
  - como modelar um sistema distribuído?
  - que pressupostos podem ser feitos?
  - qual a terminologia que vamos adotar?
  - como analisar o comportamento de um sistema distribuído?
- Começaremos a olhar para alguns problemas relevantes e respectivas soluções