Métodos de Apoio à Decisão

Simulação

João Pedro Pedroso

2024/2025

Aula passada: Noções estudadas

- gestão de projetos: PERT

Hoje:

- simulação

Caraterização de um caminho

- Duração esperada das atividades de um caminho $= \sum_{(i,j) \in \text{caminho}} E[T_{ij}]$
- Variância da duração das atividades no caminho $= \sum_{(i,j) \in \text{caminho}} \text{var}(T_{ij})$
- DT $\rightarrow$ duração total das atividades no caminho crítico
  - (determinado pelo CPM)
  - DT $\rightarrow$ variável aleatória
- Admite-se que o número de atividades é suficientemente grande para se poder aplicar o teorema do limite central:

$$DT = \sum_{(i,j) \in \text{caminho crítico}} T_{ij}$$

tem distribuição normal

Limitações da técnica PERT

- muitas vezes as durações das atividades não são independentes;
- as durações podem não ter distribuição beta;
- pode haver vários caminhos críticos;
- a hipótese de que o caminho crítico é o calculado com as durações médias pode não ser válida;
- uma alternativa a este método é a utilização de simulação de Monte Carlo.

Simulação

Simulação: motivação

- Nos modelos que temos vindo a estudar, encontramos uma formulação analítica do problema, que depois otimizamos.
- Em muitas situações isso não é possível, devido à sua complexidade, a relações estocásticas entre componentes do problema, etc.
- Neste casos, para se conseguir modelos analíticos é necessário fazer tantas simplificações que a soluções ficam inadequadas à realidade.
- Uma alternativa é utilizar simulação como ferramenta de modelação, análise, e suporte à decisão.

Simulação

- **Simulação** é uma técnica em que se imita o funcionamento de operações reais, à medida que elas evoluem com o tempo.
- Um *modelo de simulação* consiste numa série de pressupostos acerca da operação do sistema, expressa em termos de relações *matemáticas* ou *lógicas* entre os objetos de interesse;
- Esse modelo é *executado* ao longo do tempo, em geral com recurso a computadores;
- São assim geradas *amostras* representativas para as medidas de desempenho;
- O resultado da simulação é, portanto, uma série de *observações* (uma *amostra*).

Vantagens e inconvenientes

- Vantagens:
  - a teoria é muito simples;
  - modelos são mais fáceis de aplicar do que os modelos analíticos;
  - modelos mais ajustados à realidade (menos simplificações); maior flexibilidade;
  - é fácil analisar diferentes políticas, parâmetros, e *designs*.
- Inconvenientes:
  - simulação não é uma técnica de otimização;
  - é frequentemente utilizada para análise *what-if*, mas otimização com base em simulação é lenta, e computacionalmente muito dispendiosa.

Terminologia essencial

- Um **sistema** é um conjunto de entidades que agem/interagem com vista à realização de um fim lógico.
- O **estado** de um sistema é uma instanciação do conjunto de variáveis necessárias à sua completa descrição, num determinado instante.
- Num **sistema discreto** as variáveis mudam apenas num conjunto contável de instantes de tempo.
  (Exemplo: a fila de espera de um guichet.)
- Num **sistema contínuo** as variáveis mudam de forma ininterrupta com respeito ao tempo.
  (Exemplo: concentração de reagente num processo químico.)
- Um **modelo de simulação estático** é a representação de um sistema num determinado instante $\rightarrow$ *simulação de Monte Carlo.*
- Um **modelo de simulação dinâmico** representa a evolução de um sistema ao longo do tempo.
- Num sistema *determinístico* não há variáveis aleatórias.
- Um sistema *estocástico* tem pelo menos uma variável aleatória

Simulação de eventos discretos: exemplo

- Sistema: uma fila de espera com um *servidor* (por exemplo, a fila na cantina);
- Clientes chegam ao sistema, e podem
  - ser servidos imediatamente (se o servidor estiver disponível)
  - ir para uma fila de espera (caso contrário).
- Para simularmos o sistema, temos primeiro que o descrever:
  1. assume-se que os clientes chegam de uma população infinita
  2. o espaço de espera é ilimitado
  3. clientes são servidos pela ordem de chegada
  4. chegadas de um cliente de cada vez, com os seguintes intervalos (minutos); todos os clientes são servidos, com os seguintes tempos (minutos):

intervalo chegada	probabilidade
1	.20
2	.30
3	.35
4	.15

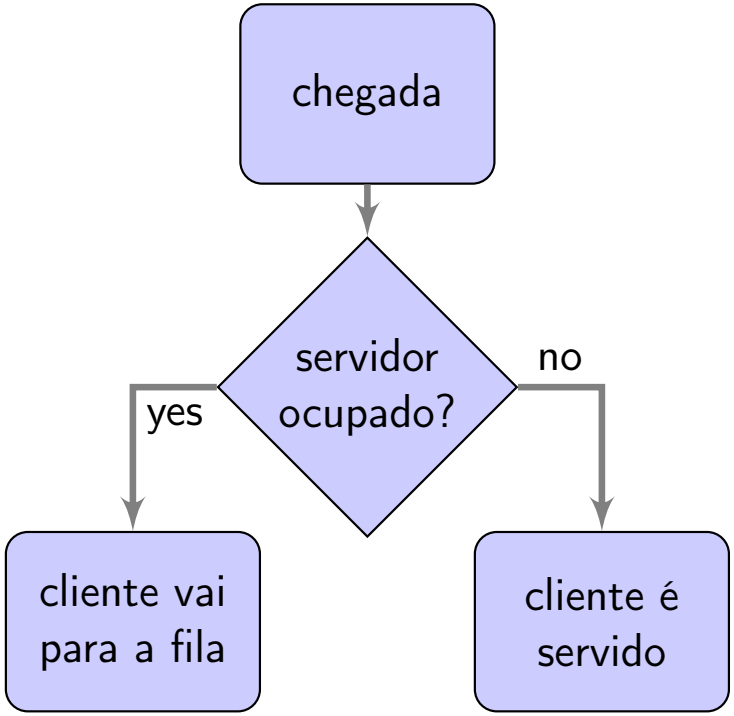
tempo de atendimento	probabilidade
1	.35
2	.40
3	.25

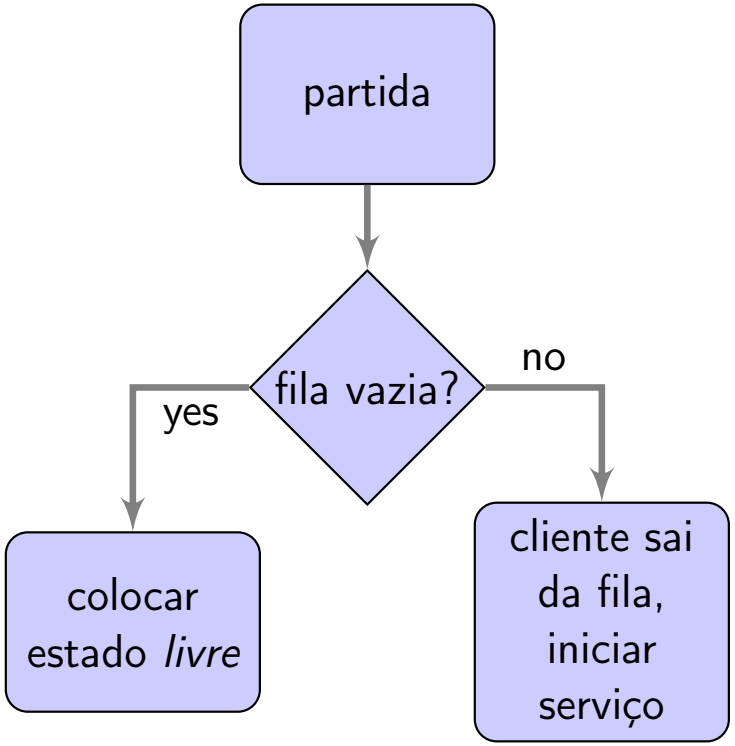
  5. depois de atendidos, os clientes regressam à população.

Definição do estado do sistema

- Variáveis:
  1. número de clientes no sistema
  2. estado do servidor (livre ou ocupado)
  3. instante da próxima chegada
  4. instante da próxima partida

Diagrama de fluxo



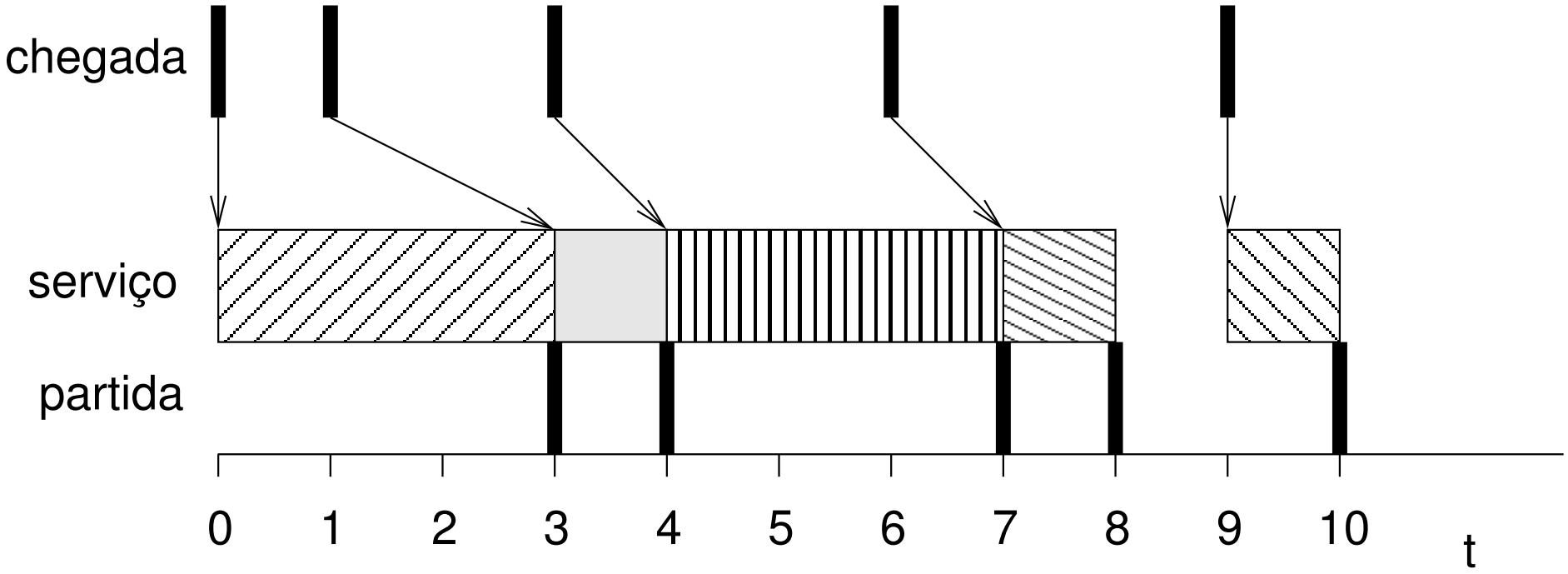


Simulação do modelo

- Um **evento** é uma situação que causa uma alteração instantânea no estado do sistema. Neste caso:
  1. chegada ao sistema
  2. partida do sistema, depois do serviço
- Eventos são sequenciados, ocorrendo a determinados instantes;
- Toda a informação acerca dos eventos é mantida numa *lista de eventos*: tipo de evento, instante a que ocorrem;
- O tempo é mantido numa variável, o *relógio*;
- Começa-se a simulação com um sistema vazio, e assume-se que o primeiro evento (um chegada) ocorre no instante 0.
- Chegadas depois do instante 0 poderão encontrar o sistema;
  - *livre* – cliente é servido imediatamente
  - *ocupado* – cliente entra na fila de espera.
- Partidas: geração aleatória da duração do serviço, que é somada tempo de início do serviço.

Avanço do tempo (temporização)

- *Next-event time advance:* o relógio é avançado para o instante a que ocorrerá o próximo evento relevante.
- *Fixed-increment time advance:* o relógio avança sempre Δt unidades, apropriadas ao problema (habitualmente uma unidade).



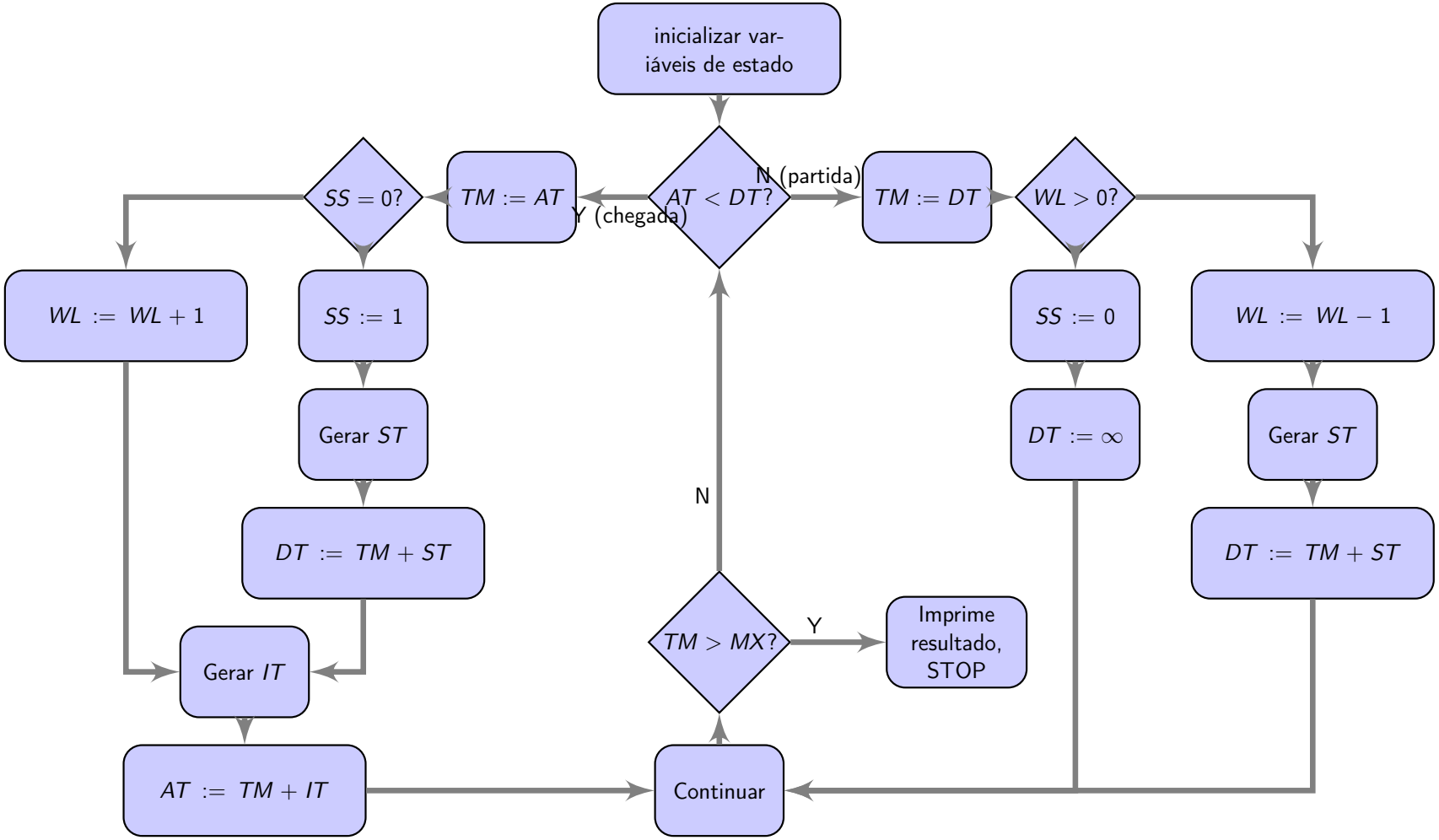
Simulação: modelo para fila com um servidor

Variáveis

- $TM \rightarrow$ *relógio* da simulação
- $AT \rightarrow$ instante da próxima chegada escalonada
- $DT \rightarrow$ instante da próxima partida escalonada
- $SS \rightarrow$ estado do servidor (1—ocupado, 0—livre)
- $WL \rightarrow$ tamanho da fila de espera
- $MX \rightarrow$ tempo de simulação pretendido

Simulação: modelo para fila com um servidor

Diagrama de fluxo



Output da simulação

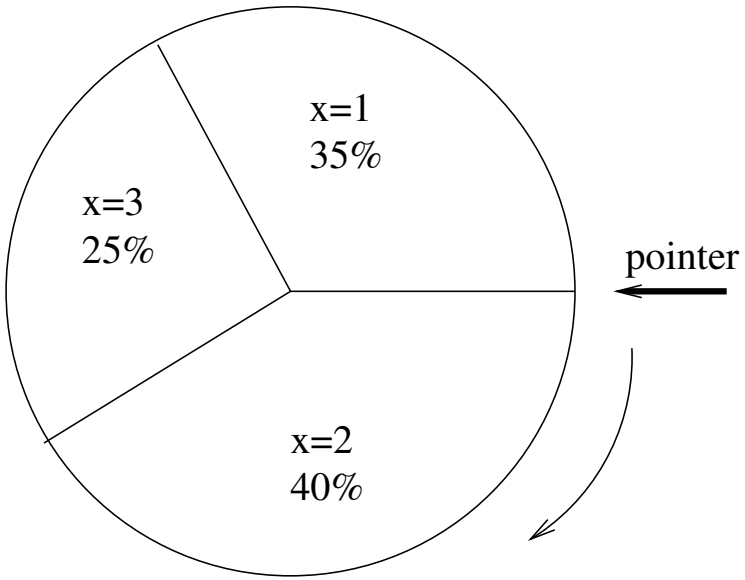
Computer Representation of the Simulation

End of Event	Type of Event	Customer Number	System Variables			Event List	
			TM	SS	WL	AT	DT
0	Initialization	—	0	0	0	0	9,999
1	Arrival	1	0	1	0	2	3
2	Arrival	2	2	1	1	4	3
3	Departure	1	3	1	0	4	6
4	Arrival	3	4	1	1	7	6
5	Departure	2	6	1	0	7	8
6	Arrival	4	7	1	1	11	8
7	Departure	3	8	1	0	11	9
8	Departure	4	9	0	0	11	9,999
9	Arrival	5	11	1	0	13	12
10	Departure	5	12	0	0	13	9,999
11	Arrival	6	13	1	0	14	15
12	Arrival	7	14	1	1	17	15
13	Departure	6	15	1	0	17	16
14	Departure	7	16	0	0	17	9,999
15	Arrival	8	17	1	0	20	19

Geração de números (pseudo) aleatórios

- Está na base da maior parte dos modelos de simulação;
- No exemplo anterior queríamos que o tempo de atendimento fosse:

tempo de atendimento	probabilidade
1	.35
2	.40
3	.25



- Como obter amostras com essa frequência, e independentes?

Método da roleta

- Utiliza-se um gerador com distribuição uniforme (entre 0 e 1).
- Determinar probabilidade acumulada:

tempo de atendimento	probabilidade	probabilidade acumulada	limites para o número aleatório
1	.35	.35	$[0.00, 0.35[$
2	.40	.75	$[0.35, 0.75[$
3	.25	1.00	$[0.75, 1.00[$

- Seja r um numero aleatório, com distribuição uniforme entre 0 e 1
  1. se $0 \leq r < 0.35 \Rightarrow$ tempo de atendimento=1
  2. se $0.35 \leq r < 0.75 \Rightarrow$ tempo de atendimento=2
  3. se $0.75 \leq r < 1. \Rightarrow$ tempo de atendimento=3
- Versão incluída no módulo `random`:

```
import random
random.choices([1,2,3], [35,40,25])
```

Simulação e redes PERT

- Para a simulação, utiliza-se uma *concretização* do projeto:
  1. a cada atividade é atribuída uma duração aleatória, com a distribuição correspondente
  2. com essas durações determina-se a data mais cedo para a conclusão do projeto
  3. ainda com essas durações, determina-se quais as atividades críticas
- repete-se este processo um número elevado de vezes, até se ter confiança na amostra
- com a amostra, determina-se estatísticas para os indicadores desejados (datas de conclusão, ...)

Análise estatística

- Os dados obtidos pelo processo de simulação têm variabilidade aleatória;
- Para os analisar, é necessário recorrer a métodos estatísticos;
- Seja o resultado pretendido um parâmetro θ:
  - determinar, através da simulação, uma estimativa $\hat{\theta}$ de θ;
  - determinar um intervalo de confiança para essa estimativa (para determinado nível de confiança).
- Problema:
  - os resultados da simulação são geralmente interdependentes (em estatística: os dados são autocorrelacionados)
  - por vezes as condições iniciais do sistema influenciam os resultados (períodos transientes, e.g. em filas de espera).

Análise estatística: caso mais simples

- Fazemos n simulações independentes, utilizando em todas elas um determinado critério de paragem;
- Se todas as simulações
  - tiverem as mesmas condições iniciais
  - forem executadas com sequências aleatórias diferentes (i.e., os geradores aleatórios tiverem diferentes *seeds*), cada *simulação* pode ser considerada uma observação independente.
- Seja X_i a estimativa da medida obtida na observação i (cada uma é uma variável aleatória, que se assume independente das outras);
- Na experiência recolhe-se a amostra $X_1, X_2, \ldots, X_n$;

- Suponhamos que se pretende um grau de confiança de $C\%$; seja $f = C/100$ e $\alpha = 1 - f$;
- O intervalo de confiança para $\theta = E(X)$ é $\bar{X} \pm t_{(\alpha/2,n-1)}\sqrt{S^2/n}$ em que
  - $\bar{X} = \sum_{i=1}^{n} X_i/n$
  - $S^2 = \sum_{i=1}^{n}(X_i - \bar{X})^2/(n-1)$
  - $t_{(\alpha,n-1)}$ é o valor para o qual, na distribuição t de student com $n-1$ graus se liberdade, se verifica $P(t_{n-1} \geq t_{(\alpha,n-1)}) = \alpha$

Programa em Python

- A análise pode ser feita através do programa em Python:

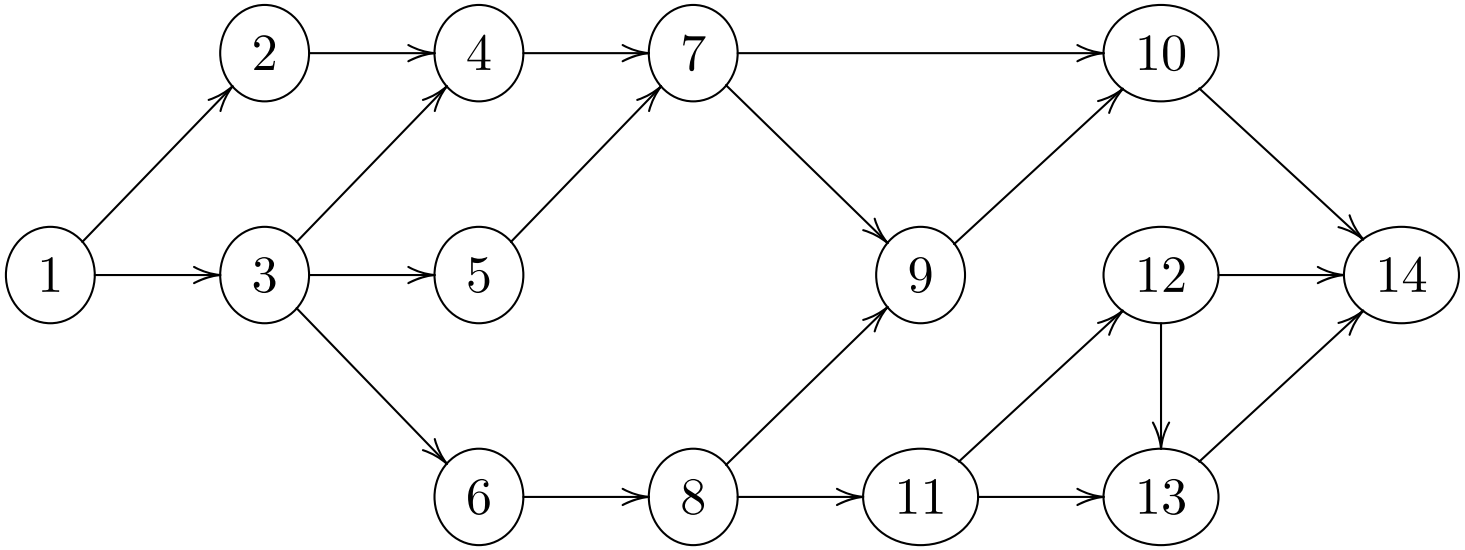
```python
import scipy.stats
import math

def mean_confidence_interval(data, alpha):
    n = len(data)
    m = float(sum(data))/n                                  # sample mean
    var = sum([(x - m)**2 for x in data]) / float(n-1)  # sample variance
    # calls the inverse CDF of the Student's t distribution
    tfact = scipy.stats.t._ppf(1-alpha/2., n-1)
    h = tfact * math.sqrt(var/n)
    print(m, var)
    print(tfact)
    print(tfact * math.sqrt(var/n))
    return m-h, m+h

if __name__ == "__main__":
    confidence = .95  # 95%
    alpha = 1 - confidence
    data = [9.252, 9.273, 9.413, 9.198, 9.532,
            9.355, 9.155, 9.558, 9.310, 9.269]
    print(mean_confidence_interval(data, alpha))
```

Utilização de folhas de cálculo

- A análise pode ser feita em folhas de cálculo, utilizando a função `TINV(2*alpha,dof)`, em que `dof` é o número de graus de liberdade (i.e., $n - 1$).
- Para os dados do exemplo anterior: $t_{(0.025,9)}$ pode ser calculado através de `=TINV(0.05,9)`, que dá o valor 2.26.

Simulação de uma rede PERT

Considere a seguinte rede de actividades, em que se admite que a duração de cada uma das actividades não é conhecida com precisão:



Duração das atividades

As estimativas para a duração mínima, máxima, e mais provável para cada uma das actividades dada na tabela seguinte (em dias).

Actividade	a	b	m	Actividade	a	b	m
(1,2)	4	8	6	(7,10)	5	28	18
(1,3)	2	8	4	(8,9)	6	11	9
(2,4)	1	7	3	(8,11)	7	18	10
(3,4)	6	12	9	(9,10)	2	2	2
(3,5)	5	15	10	(10,14)	10	40	25
(3,6)	7	18	12	(11,12)	5	20	10
(4,7)	5	12	9	(11,13)	4	18	12
(5,7)	1	3	2	(12,13)	1	3	2
(6,8)	2	6	3	(12,14)	8	12	10
(7,9)	10	20	15	(13,14)	7	22	10

Recorrendo à simulação, resolva as seguintes questões.

1. Determine a duração média do projecto, e compare-a com a solução da aproximação analítica utilizada nesta disciplina.
2. Determine a probabilidade de o projecto ser concluído em menos de 50 dias.
3. Determine a probabilidade de a tarefa (1,3) ser crítica.
4. Determine as durações médias do projecto nas seguintes circunstâncias:
   1. Actividades (5,7) e (6,8) têm de começar ao mesmo tempo;
   2. Actividades (5,7) só precisa de ser realizada se (3,5) terminar depois de (3,4).

   Compare-as com o valor anterior.

CPM: código em Python

```python
def cpm(V, A, P, S, D):
    ES, LF = {}, {}
    for i in V:
        ES[i] = 0
        LF[i] = INF
    for i in V:
        for j in S[i]:
            ES[j] = max(ES[j], ES[i]+D[i, j])
    tmin = ES[V[-1]]
    LF[V[-1]] = tmin
    for j in reversed(V):
        for i in P[j]:
            LF[i] = min(LF[i], LF[j]-D[i, j])

    critical = []
    for (i, j) in A:
        F = LF[j]-ES[i]-D[i, j]
        if F < EPS:
            critical.append((i, j))

    return tmin, critical
```

Simulador: código em Python

```python
def simulate(N, V, A, P, S, a, b, m):
    t = 0
    p = {}
    for (i,j) in A:
        p[i,j] = 0
    for k in range(N):
        D = {}  # activity durations
        for (i,j) in A:
            D[i,j] = betarnd(a[i,j],m[i,j],b[i,j])
        tmin, critical = cpm(V, A, P, S, D)
        t += tmin
        for (i,j) in critical:
            p[i,j] += 1
        print(f"{tmin:12g}\t{critical}")
    t = float(t)/N
    for (i,j) in A:
        p[i,j] = float(p[i,j])/N
    return t, p
```

(código completo em https://www.dcc.fc.up.pt/~jpp/mad/pert.py)

Próximas aulas

- Otimização com variáveis inteiras