

Trabalho Prático 3

Modelos Determinísticos de Investigação Operacional

Rodrigo Pimentel (a83765)

Angélica Freitas (a83761)

Alexandra Reigada (a84584)

# Parte 0

O número de inscrição do aluno que apresenta maior valor é 84584. De acordo com o modelo ABCDE, as atividades a remover serão o 8 e o 4.

O projeto anterior era descrito pelas atividades e relações de precedência a seguir indicadas:

|  |  |  |
| --- | --- | --- |
| Atividade | Duração | Precedências |
| 0 | 4 | - |
| 1 | 6 | 0 |
| 2 | 7 | 1,4 |
| 3 | 2 | 2,5 |
| 4 | 9 | 0,7 |
| 5 | 4 | 4,8 |
| 6 | 5 | - |
| 7 | 6 | 6 |
| 8 | 4 | 7,10 |
| 9 | 2 | 8,11 |
| 10 | 8 | 6 |
| 11 | 7 | 10 |

Assim, como a atividade 4 foi removida, as atividades 2 e 5 vão ter como novas precedências as atividades 0 e 7 (em vez da atividade 4). Também, removendo a atividade 8, as atividades 5 e 9 passam a ter as atividades 7 e 10 em vez da atividade 8. Isto é, o projeto representa-se agora pelas seguintes atividades e relações de precedência:

|  |  |  |
| --- | --- | --- |
| Atividade | Duração | Precedências |
| 0 | 4 | - |
| 1 | 6 | 0 |
| 2 | 7 | 1,0,7 |
| 3 | 2 | 2,5 |
| 5 | 4 | 0,7,10 |
| 6 | 5 | - |
| 7 | 6 | 6 |
| 9 | 2 | 7,10,11 |
| 10 | 8 | 6 |
| 11 | 7 | 10 |

Uma imagem com mapa, texto, fotografia

Descrição gerada automaticamenteDe seguida apresentamos a rede que representa o projeto depois de eliminar as atividades:

As restrições do problema, relativas a cada um dos arcos do grafo, traduzem as relações de precedência entre as atividades. Para uma dada atividade j, o tempo de início da atividade j deve ser posterior ao tempo de conclusão de cada uma das atividades i que precedem j, originando assim a necessidade de um modelo de programação inteira. Dado que ti designa o tempo de início da atividade i, a função ti + di designa o tempo de conclusão da atividade i. O projeto termina no instante de tempo tf , quando todas as atividades predecessoras imediatas da atividade fictícia fim estiverem concluídas. O objetivo será minimizar o tempo de execução total do projeto.

Assim, obtemos o seguinte modelo:

/\* função objetivo \*/

min: tf ;

/\* restrições \*/

arco\_01: t1 >= t0 + 4 ;

arco\_12: t2 >= t1 + 6 ;

arco\_02: t2 >= t0 + 4 ;

arco\_72: t2 >= t7 + 6 ;

arco\_23: t3 >= t2 + 7 ;

arco\_i0: t0 >= ti + 0 ;

arco\_05: t5 >= t0 + 4 ;

arco\_75: t5 >= t7 + 6 ;

arco\_105: t5 >= t10 + 8 ;

arco\_53: t3 >= t5 + 4 ;

arco\_3f: tf >= t3 + 2 ;

arco\_5f: tf >= t5 + 4 ;

arco\_i6: t6 >= ti + 0 ;

arco\_9f: tf >= t9 + 2 ;

arco\_67: t7 >= t6 + 5 ;

arco\_610: t10 >= t6 + 5 ;

arco\_79: t9 >= t7 + 6 ;

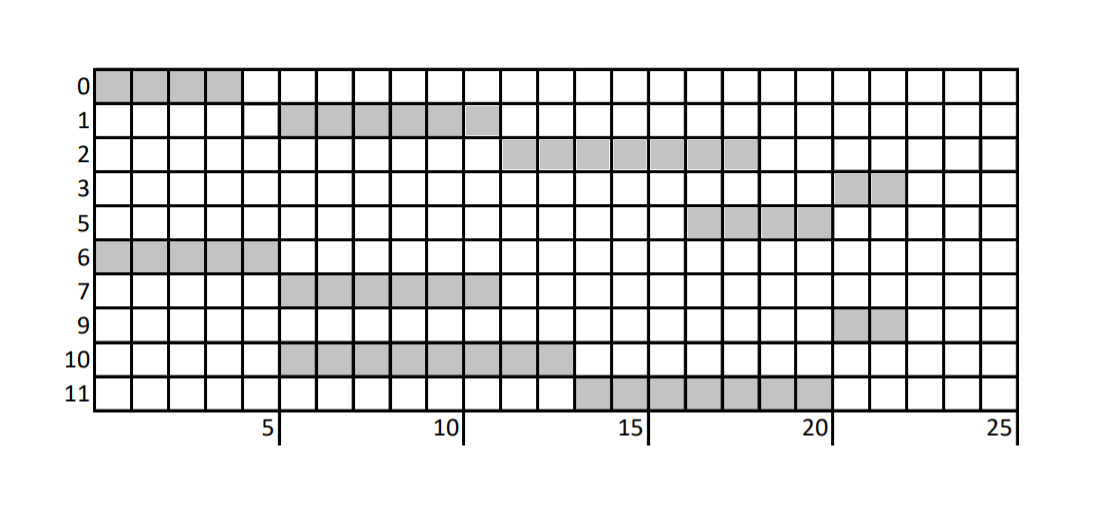
arco\_119: t9 >= t11 + 7 ;

arco\_109: t9 >= t10 + 8 ;

arco\_1011: t11 >= t10 + 8;

É de salientar que algumas restrições são redundantes. Anteriormente à remoção das atividades 4 e 8 tínhamos que atividade 9 tinha como precedência a atividade 11 e a atividade 11 tinha como precedência a atividade 10. Ora como a atividade 8 foi removida, a atividade 10 tornou-se nova precedência da atividade 9 gerando a restrição arco\_109. Esta acaba por ser redundante, pois a atividade 9 apenas começa quando a 7 e 11 acabam, e a atividade 11 começa por sua vez quando a atividade 10 acaba. Logo essa restrição não influencia o tempo necessário para completar a execução do projeto. O mesmo acontece, por exemplo, na restrição arco\_02.

O valor da variável de decisão ti na solução ótima define o tempo de início de execução da atividade i, permitindo construir um plano de execução do projeto (diagrama de Gantt). O projeto é executado num tempo com a duração do valor da solução ótima, que neste caso é de 22. O Diagrama de Gantt é apresentado de seguida:



De seguida, apresentámos o caminho crítico, correspondente a uma duração de 22 unidades de tempo:

Uma imagem com texto, mapa

Descrição gerada automaticamente

# Parte I

Através do diagrama de Gantt da Parte 0, verificamos que as atividades 1, 7 e 10 ocorrem paralelamente. Como anteriormente visto, a atividade 10 pertence ao caminho crítico.

Como apenas há um recurso para realizar as tarefas 1, 7 e 10, uma atividade só pode ser realizada após a que esteja antes tenha terminado. Assim, é necessário implementar programação inteira.

Seja pj o tempo de processamento da tarefa j e igualmente à Parte 0 tj o instante do início de execução da tarefa j.

Como temos 3 tarefas, há 3! Possibilidades para ordenar a realização de cada tarefa. Também iremos atribuir um número a cada ordem:

|  |  |  |  |
| --- | --- | --- | --- |
| Número de ordem | 1ª Tarefa | 2ªTarefa | 3ªTarefa |
| 1 | 1 | 7 | 10 |
| 2 | 1 | 10 | 7 |
| 3 | 7 | 1 | 10 |
| 4 | 7 | 10 | 1 |
| 5 | 10 | 1 | 7 |
| 6 | 10 | 7 | 1 |

De forma a descobrir qual a melhor ordem. É necessária uma variável yo que exprime a dicotomia:

De maneira a ser escolhida apenas uma ordem também teremos de adicionar a restrição:

y1+y2+y3+y4+y5+y6=1;

Assim, se, por exemplo, a variável y3 tiver valor 1, então através da tabela apresentada anteriormente, a 1ª tarefa a ser executada é a 7, que segue a atividade 1 e, por fim a tarefa 10.

Então, para resolvermos este problema temos de adicionar ao modelo anterior as seguintes restrições disjuntivas:

1. ti + pi <= tj e tj + pj <= tz
2. ti + pi <= tz e tz + pz <= tj
3. tj + pj <= ti e ti + pi <= tz
4. tj + pj <= tz e tz + pz <= ti
5. tz + pz <= ti e ti + pi <= tj
6. tz + pz <= tj e tj + pj <= ti

Como num modelo de programação linear não podemos usar disjunção de condições, teremos de utilizar modelo de programação inteira e pode ser escrito da seguinte maneira:

1)

t1 + 6 <= t7 + 100\*(1 – y1) ;

t7 + 6 <= t10 + 100\*(1 – y1);

2)

t1 + 6 <= t10 + 100\*(1 – y2);

t10 + 8 <= t7 + 100\*(1 – y2) ;

3)

t7 + 6 <= t1 + 100\*(1 – y3);

t1 + 6 <= t10 + 100\*(1 – y3);

4)

t7 + 6 <= t10 + 100\*(1 – y4);

t10 + 8 <= t1 + 100\*(1 – y4) ;

5)

t10 + 8 <= t1 + 100\*(1 – y5) ;

t1 + 6 <= t7 + 100\*(1 – y5) ;

6)

t10 + 8 <= t7 + 100\*(1 – y6) ;

t7 + 6 <= t1 + 100\*(1 – y6) ;

Adicionando estas restrições, estamos a acrescentar precedências aos nodos. Por exemplo, a restrição 1) estamos a implicar que a atividade 7 seja realizada depois da atividade 1 e antes da atividade 10. Assim, estaremos a acrescentar arcos adicionais ao grafo anterior. Através da solução no lpsolve, saberemos quais a sequência escolhida e, por sua vez, quais os arcos adicionais a acrescentar no grafo.

/\* função objetivo \*/

min: tf ;

/\* restrições \*/

arco\_01: t1 >= t0 + 4 ;

arco\_12: t2 >= t1 + 6 ;

arco\_02: t2 >= t0 + 4 ;

arco\_72: t2 >= t7 + 6 ;

arco\_23: t3 >= t2 + 7 ;

arco\_i0: t0 >= ti + 0 ;

arco\_05: t5 >= t0 + 4 ;

arco\_75: t5 >= t7 + 6 ;

arco\_105: t5 >= t10 + 8 ;

arco\_53: t3 >= t5 + 4 ;

arco\_3f: tf >= t3 + 2 ;

arco\_5f: tf >= t5 + 4 ;

arco\_i6: t6 >= ti + 0 ;

arco\_9f: tf >= t9 + 2 ;

arco\_67: t7 >= t6 + 5 ;

arco\_610: t10 >= t6 + 5 ;

arco\_79: t9 >= t7 + 6 ;

arco\_119: t9 >= t11 + 7 ;

arco\_109: t9 >= t10 + 8 ;

arco\_1011: t11 >= t10 + 8;

t1 + 6 <= t7 + 100 - 100\*y1 ;

t7 + 6 <= t10 + 100 - 100\*y1 ;

t1 + 6 <= t10 + 100 - 100\*y2 ;

t10 + 8 <= t7 + 100 - 100\*y2 ;

t7 + 6 <= t1 + 100 - 100\*y3 ;

t1 + 6 <= t10 + 100 - 100\*y3 ;

t7 + 6 <= t10 + 100 - 100\*y4 ;

t10 + 8 <= t1 + 100 - 100\*y4 ;

t10 + 8 <= t1 + 100 - 100\*y5 ;

t1 + 6 <= t7 + 100 - 100\*y5 ;

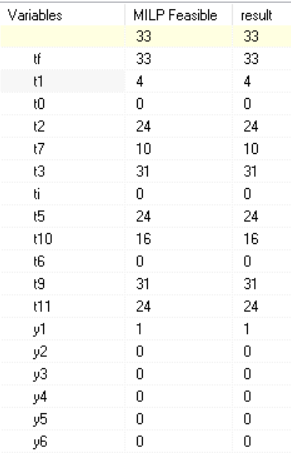
t10 + 8 <= t7 + 100 - 100\*y6 ;

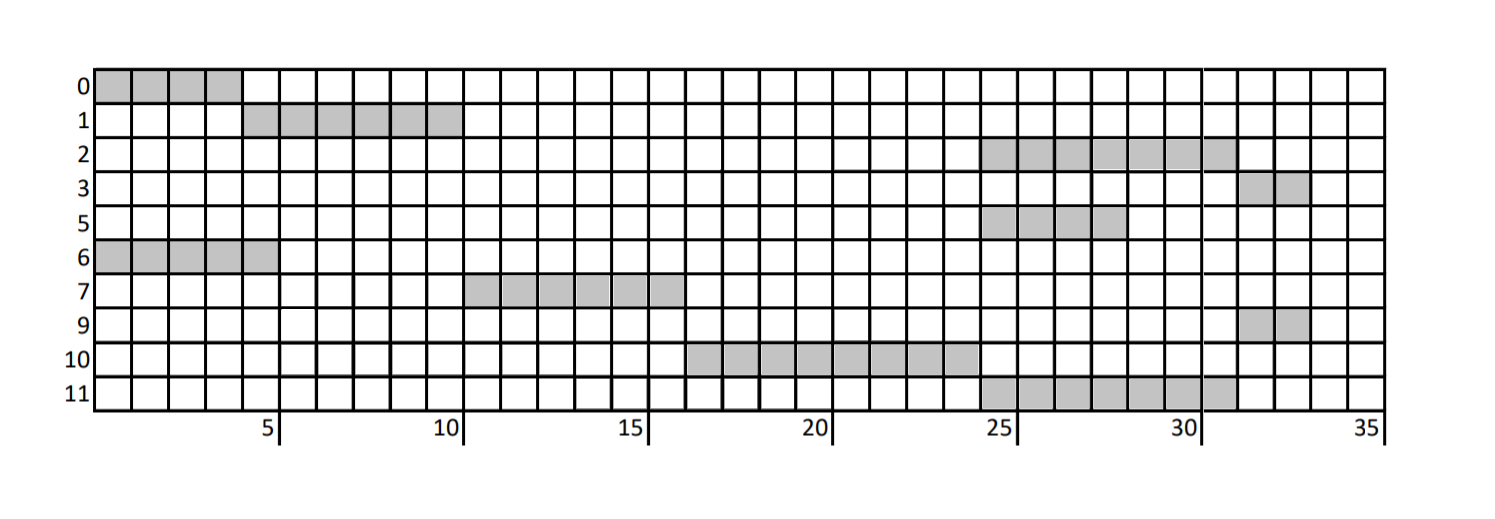
t7 + 6 <= t1 + 100 - 100\*y6 ;

y1+y2+y3+y4+y5+y6=1;

int t1,t2,t3,t4,t5,t6,t7,t8,t9,t10,t11,ti,tf;

bin y1,y2,y3,y4,y5,y6;

1. 



Assim, como a atividade 7 tem como precedência a atividade 1 e a atividade 10 tem como precedência a atividade 7, teremos de adicionar os seguintes arcos a vermelho:

Uma imagem com texto, mapa

Descrição gerada automaticamenteUma imagem com texto, mapa

Descrição gerada automaticamenteE, por sua vez, obtivemos o seguinte caminho crítico:

Sabendo que t1 = 4 , t7 = 10, t10 =16, p1 = 6, p7 = 6, p10 = 8, y1 = 1, y2 = 0, y3 = 0, y4 = 0, y5 = 0, y6 = 0, . Substituindo nas novas restrições temos que:

4 + 6 <= 10 + 100 - 100\*1 ;

10 + 6 <= 16 + 100 - 100\*1 ;

4 + 6 <= 16 + 100 - 100\*0 ;

16 + 8 <= 10 + 100 - 100\*0 ;

10 + 6 <= 4 + 100 - 100\*0 ;

4 + 6 <= 16 + 100 - 100\*0 ;

10 + 6 <= 16 + 100 - 100\*0 ;

16 + 8 <= 4 + 100 - 100\*0 ;

16 + 8 <= 4 + 100 - 100\*0 ;

4 + 6 <= 10 + 100 - 100\*0 ;

16 + 8 <= 10 + 100 - 100\*0 ;

10 + 6 <= 4 + 100 - 100\*0 ;

1+0+0+0+0+0=1;

# Parte II

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Atividade | Custo Normal | C1 | Max red a custo c1 | C2 | Max red a custo c2 |
| 0 | 400 | 200 | 0.5 | 100 | 0.5 |
| 1 | 1000 | 600 | 1 | 300 | 1 |
| 2 | 1400 | 1000 | 3 | 500 | 1 |
| 3 | 300 | 200 | 0.5 | 100 | 0.5 |
| 5 | 1000 | 1600 | 0.5 | 800 | 0.5 |
| 6 | 800 | 180 | 1 | 90 | 1 |
| 7 | 900 | - | 0 | - | 0 |
| 9 | 300 | - | 0 | - | 0 |
| 10 | 1600 | 1000 | 0.5 | 500 | 0.5 |
| 11 | 1400 | 600 | 1 | 300 | 1 |

Inserimos uma nova variável de decisão:

,

Como o valor de c2 corresponde ao custo suplementar de reduzir a duração da atividade de uma unidade de tempo após ter aplicado a máxima redução a um custo c1 é necessário recorrer a um modelo de programação linear inteira. Para tal, fixando uma atividade j, temos que r2j<=r1j. Exemplo: r20 <= r10 ;

Cada restrição rij tem uma redução máxima, dando origem a restrições do género: r10 <= 0.5 ;

As restrições de precedência são as mesmas adicionando as reduções (por exemplo arco\_01: t1 >= t0 -r10 -r20 + 4 ;)

Como queremos que o tempo final seja reduzido em 3 U.T., então iremos subtrair 3 U.T. ao valor obtido na primeira fase (neste caso é 22). Assim, tf <=19.

Temos também um novo dado:

E por sua vez, a função objetivo pretende minimizar o custo suplementar total.

min: 200 r10 + 600 r11 + 1000 r12 + 200 r13 + 1600 r15 + 180 r16 + 0 r17 + 0 r19 + 1000 r110+ 600 r111 + 100 r20 + 300 r21 + 500 r22 + 100 r23 +800 r25 + 90 r26 + 0 r27 + 0 r29 + 500 r210 + 300 r211

tf <= 19;

arco\_01: t1 >= t0 -r10 -r20 + 4 ;

arco\_12: t2 >= t1 -r11 -r21 + 6 ;

arco\_02: t2 >= t0 -r10 -r20 + 4 ;

arco\_72: t2 >= t7 -r17 -r27 + 6 ;

arco\_23: t3 >= t2 -r12 -r22 + 7 ;

arco\_i0: t0 >= ti + 0 ;

arco\_05: t5 >= t0 -r10 -r20 + 4 ;

arco\_75: t5 >= t7 -r17 -r27 + 6 ;

arco\_105: t5 >= t10 -r110 -r210 + 8 ;

arco\_53: t3 >= t5 -r15 -r25 + 4 ;

arco\_3f: tf >= t3 -r13 -r23 + 2 ;

arco\_5f: tf >= t5 -r15 -r25 + 4 ;

arco\_i6: t6 >= ti + 0 ;

arco\_9f: tf >= t9 -r19 -r29 + 2 ;

arco\_67: t7 >= t6 -r16 -r26 + 5 ;

arco\_610: t10 >= t6 -r16 -r26 + 5 ;

arco\_79: t9 >= t7 -r17 -r27 + 6 ;

arco\_119: t9 >= t11 -r111 -r211 + 7 ;

arco\_109: t9 >= t10 -r110 -r210 + 8 ;

arco\_1011: t11 >= t10 -r110 -r210 + 8;

r10 <= 0.5 ;

r11 <= 1 ;

r12 <= 3 ;

r13 <= 0.5 ;

r15 <= 0.5 ;

r16 <= 1 ;

r17 <= 0 ;

r19 <= 0 ;

r110 <= 0.5 ;

r111 <= 1 ;

r20 <= 0.5 ;

r21 <= 1 ;

r22 <= 1 ;

r23 <= 0.5 ;

r25 <= 0.5 ;

r26 <= 1 ;

r27 <= 0 ;

r29 <= 0 ;

r210 <= 0.5 ;

r211 <= 1 ;

r20 <= r10 ;

r21 <= r11 ;

r22 <= r12 ;

r23 <= r13 ;

r25 <= r15 ;

r26 <= r16 ;

r27 <= r17 ;

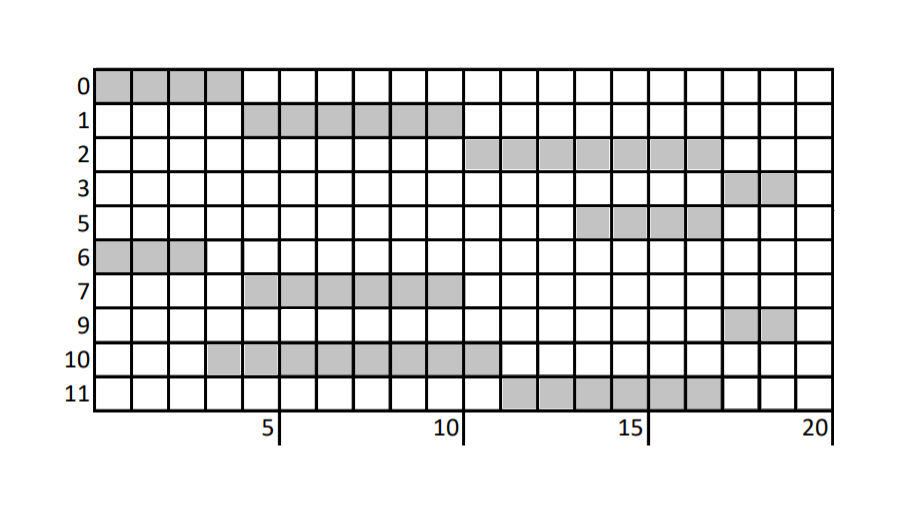
r29 <= r19 ;

r210 <= r110 ;

r211 <= r111 ;

Uma imagem com recibo

Descrição gerada automaticamente

1. 

Uma imagem com texto, mapa

Descrição gerada automaticamenteAtravés do diagrama de Gantt, é notável que temos 2 caminhos críticos alternativos, representados de seguida:

De modo a verificar o custo, iremos substituir os valores do resultado obtido na alínea 3 da parte II na função objetivo:

200\*0 + 600\*0 + 1000\*0 + 200\*0+ 1600\*0+ 180\*1 + 0\*0 + 0\*0 + 1000\*0 + 600\*0.5 + 100\*0 + 300\*0 + 500\*0 + 100\*0 +800\*0 + 90\*1 + 0\*0 + 0\*0 + 500\*0 + 300\*0.5

180 + 300 + 90 + 150 = 720, como queríamos mostrar.