CÓDIGOS leilão para a problema de atribuição
Alguns dos códigos são do ANEXOS do livro didático
"Otimização da rede LINEAR: algoritmos e códigos",
MIT Press, 1991
por Dimitri P. Bertsekas

As versões dadas são as mais
recente a partir de

07 de maio de 1992

Todos os códigos podem ser
compilado com o compilador Absoft populares em todos os computadores Macintosh (sujeito a limitações de memória). Ao alterar uma entrada e poucos
instruções de saída, os códigos podem ser facilmente adaptadas para outros computadores e Compiladores Fortran. Os códigos não deve ser modificado, exceto para o mínimo
modificações necessárias para fazê-los funcionar em outros compiladores. Por Favor
indicar claramente as suas modificações ao relatar a investigação.

Os seguintes códigos são incluídos:

1) LEILÃO: algoritmo leilão Avançado para problemas de atribuição simétricas.

2) AUCTION_FLP: Igual ao leilão, mas usa a aritmética de ponto flutuante para lidar
com problemas com a faixa de custo grande e / ou dimensão. Para tais problemas a
preços pode transbordar o intervalo inteiro no decurso do algoritmo.

3) AUCTION_AS: algoritmo Leilão de problemas de atribuição assimétricas.

4) AUCTION_FR: Atacante algoritmo leilão / reversa para atribuição simétrica
problemas.

5) NAUCTION_SP: Combinado leilão ingênuo e sequencial método caminho mais curto
para atribuição.

Os códigos
incluem um gerador problema aleatório embutido. Este gerador pode ser substituído por
outro código que lê um problema de atribuição de um arquivo.

\ Subsect {LEILÃO} Esse código implementa o leilão para a frente
algoritmo com $ \ e $ scaling para o problema de atribuição simétrica; cf. \
Seção 4.1 do livro de otimização de rede do autor.

```
C ********************************************** *****************
C
C programa de exemplo de chamada para o ALGORITHM LEILÃO
C
C ESTE MOTORISTA cria um problema SIMÉTRICA CESSÃO
C com igual número de linhas e colunas,
C e chama o SUBROUTINE LEILÃO para encontrar um
C atribuição do valor máximo.
C
C ********************************************** *****************

        PARÂMETROS (maxNodes = 10000, MAXARCS = 100000)

        NONE IMPLICIT
        INTEGER N, NA, A, ISMALL, BEGEPS, ENDEPS, CICLOS
        INTEGER NUMPHASES, STARTINCR, TEST, o lucro
        INTEGER I, J, IA, ARC, NOASS, ICOST, ABSCOST, CURARC
        INTEGER CURCOL, FSTARC, LSTARC, MINCOST, MAXCOST, MCOST
        INTEGER FOUT (maxNodes), COLASS (maxNodes)
        INTEGER ASSIGN (maxNodes), PCOL (maxNodes)
        INTEGER COST (MAXARCS), END (MAXARCS)
        Real * 8 FACTOR, TT1, TT2, TCOST, AVERAGE
        COMUM / ARRAYC / CUSTO / matrizes / END / ARRAYF / FOUT / BK1 / N, A, ISMALL
     $ / BK2 / ciclos, Médio, NUMPHASES / ARRAYA / ASSIGN / PCOLA / PCOL

C ********************************************** ***************
C
C problema de geração de código começa AQUI
C o usuário pode substituir este código com um código que lê
```

```
C HIS / seu problema de um arquivo
C
C ********************************************** ***************

C Este código inclui UM UNIFORME Random Number Generator
C que retorna um valor na (0,1)

C INITIALIZE Gerador Aleatório

      INTEGER MULT, MODUL, I15, I16, JRAN, ISEED
      RAN REAIS
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN

      ISEED = 13502460
      CHAME SETRAN (ISEED)

      PRINT * ', gerando problema de atribuição de SIMÉTRICA "
      PRINT *, "****************************************** '

C **** LER O número de linhas N & o número de arcos A ****

      PRINT *, 'Digite o número de linhas (e colunas)'
      LEIA *, N
5 PRINT *, 'Digite o número de arcos por ROW (1>) "
      LEIA *, NA
      IF (NA.LT.2) GOTO 5
      PRINT *, 'Digite o mínimo eo custo máximo'
      LEIA *, MINCOST, MAXCOST

C o número de arcos é n * NA

      A = N * NA

C Os arcos incidente a linha i SÃO FOUT (I) PARA FOUT (I + 1) -1
C Além disso, para VIABILIDADE EACH ROW está diretamente ligado
C com a coluna correspondente
```

```
      FAZER 20 I = 1, N
        FOUT (I) = 1 + (i-1) * NA
20 CONTINUAR
      FOUT (N + 1) = A + 1

C gerar a END (ARC) eo custo (ARC), que são os COLUNA
C eo coeficiente custo associado a ARC

      FAZER 25 ARC = 1, A
        END (ARC) = 1 + RAN () * N
        IF ((END (ARC) .GT.N) .OR. (END (ARC) .LT.1)) THEN
          PRINT *, 'Erro no problema de geração de'
          PAUSA
          PARE
        END IF
        COST (ARC) = MINCOST + RAN () * (MAXCOST-MINCOST)
25 CONTINUAR

C MODIFICAR O final da última ARC OUT de cada linha de viabilidade
C e defina seu CUSTO PARA -MAXCOST

      FAZER 30 I = 1, N
        END (FOUT (I + 1 -1)) = I
        CUSTO (FOUT (I + 1) -1) = - MAXCOST
30 CONTINUAR


C
C ********************************************** **************
C
C PROBLEMA geração de código TERMINA AQUI
C
C ********************************************** **************

C ESCALA O CUSTO PARA TRABALHAR COM INTEIRO EPSILON

        MAXCOST = 0
```

```
        FAZER 35 IA = 1, A
           ABSCOST = IABS (COST (IA))
           IF (ABSCOST.GT.MAXCOST) MAXCOST = ABSCOST
           COST (IA) = CUSTO (IA) * (N + 1)
35 CONTINUAR

C *** ISMALL é um inteiro MUITO PEQUENO PARA O SEU APARELHO ***

        ISMALL = -2000000000

        IF (MAXCOST.GT.INT (ABS (ISMALL) / (N + 1))) THEN
          PRINT *, 'A faixa de custo é muito grande para INTEGER ARITHMETIC'
          PAUSA
          PARE
        END IF

        MAXCOST MAXCOST = * (N + 1)


C os seguintes parâmetros BEGEPS, FACTOR, ENDEPS E STARTINCR
C são passados para o LEILÃO algoritmo. Valores entre
C (A) MAXCOST / 5 E MAXCOST / 2 para BEGEPS
C (B) 4 e 6 para FACTOR
C (C) N / 10 e 1 para ENDEPS
C (D) 1 AND BEGEPS PARA STARTINCR
C têm funcionado bem para PROBLEMAS esparsos de grande porte.
C PARA PROBLEMAS DENSOS RECOMENDA-SE QUE
C BEGEPS ser ajustado para um valor menor,
ENDEPS C ser definido como 1,
C STARTINCR ser definido como 1.

        PRINT *, "****************************************** '
        PRINT *, "custo máximo é ', MAXCOST
        PRINT *, 'Digite o EPSILON PARTIDA'
        LER *, BEGEPS
        IF (BEGEPS.LT.1) BEGEPS = 1
        PRINT *, 'Insira o fator EPSILON REDUÇÃO'
```

```
        LER *, FACTOR
        ENDEPS = N / 10
        IF (ENDEPS.LT.1) ENDEPS = 1
        IF (ENDEPS.GT.BEGEPS) ENDEPS = BEGEPS
        STARTINCR = BEGEPS / 10
        IF (STARTINCR.LT.1) STARTINCR = 1

        PRINT *, "****************************************** '
        PRINT *, 'começar EPSILON =', BEGEPS
        PRINT *, 'Epsilon redução de fatores =', FACTOR
        PRINT *, «limiar EPSILON ANTES ela é definida como 1 = ', ENDEPS
        PRINT *, "COMEÇAR MIN DE LICITAÇÃO INCREMENTO = ', STARTINCR
        PRINT *, "CHAMANDO LEILÃO para resolver o problema"
        PRINT *, "****************************************** '

C GET hora de início para o Mac II

      TT1 = longa (362) /60.0

      CHAMADA Leilão (BEGEPS, fator, ENDEPS, STARTINCR)

C GET TÉRMINO TEMPO PARA O MAC II

      TT2 = longa (362) /60.0 - TT1
      PRINT *, "acabado --- tempo de CPU TOTAL ', TT2,' s '
      PRINT *, "****************************************** '

C *** Mostrar resultados ***

X Write (9,2010) CICLOS
X2010 FORMATO ('NO LEILÃO DE CICLOS', I7)
X WRITE (9,2020) MÉDIA
X2020 FORMATO ("número médio de propostas por ciclo", F9.3)
       Write (9,2030) NUMPHASES
2030 FORMATO ('NO DE EPSILON subproblemas RESOLVIDO =', I7)

C VERIFICAÇÃO otimalidade da solução de e calcular o custo
```

```
        DO 40 I = 1, N
          COLASS (I) = 0
40 CONTINUAR
        NOASS = 0
        FAZER 50 J = 1, N
           IF (ASSIGN (J) .GT.0) THEN
             NOASS = + 1 NOASS
             COLASS (ASSIGN (J)) = J
           END IF
50 CONTINUAR
        IF (NOASS.NE.N) THEN
          PRINT *, '# de linhas atribuído não IGUAL # de linhas'
        END IF
        TCOST = 0
        FAZER 60 I = 1, N
          J = COLASS (I)
          IF (J.EQ.0) THEN
            PRINT *, 'ROW', I, 'não é atribuído "
          END IF
          FSTARC = FOUT (I)
          LSTARC FOUT = (I + 1) -1
          MCOST = ISMALL
          FAZER 70 CURARC = FSTARC, LSTARC
            CURCOL = END (CURARC)
            IF (CURCOL.EQ.J) THEN
              IF (MCOST.LT.COST (CURARC)) THEN
                 MCOST = COST (CURARC)
              END IF
            END IF
70 CONTINUAR
          LUCRO = MCOST-PCOL (J)
          FAZER 72 CURARC = FSTARC, LSTARC
             J = END (CURARC)
             TEST = COST (CURARC) -PCOL (J) lucrativos
             IF (TEST.GT.1) THEN
                PRINT *, '1-CS VIOLADOS AT ARC', CURARC
```

```
            END IF
72 CONTINUAR
          TCOST = TCOST MCOST + / (N + 1)
60 CONTINUAR
        ESCREVA (9,2100) TCOST
2100 FORMATO ('atribuição de custo =', F14.2)
        PRINT *, 'programa terminou; <CR> PARA SAIR '
        PAUSA
        FIM



C ********************************************** *****************
C
C CODE LEILÃO PARA N por N problemas de atribuição
C
C ESCRITO POR DIMITRI P. Bertsekas
C (MODIFICAÇÕES por Paul TSENG)
C
C versão 1.1, setembro 1990
C
C ESTE CÓDIGO implementa o algoritmo LEILÃO COM E-escala.
C Ele resolve uma seqüência de subproblemas e diminui
C EPSILON por um fator constante entre subproblemas.
C ESTE versão corresponda a um MODO Gauss-Seidel
C e resolve EPSILON subproblemas inexata.
C
C THE CODE é uma versão melhorada de um antigo (SEPT. 1985)
C CODE LEILÃO COM ESCALA E-ESCRITO POR DIMITRI P. Bertsekas
C
C DO CÓDIGO trata o problema como um problema de maximização.
C para resolver um problema de minimização, inverta o sinal da
C ARC CUSTOS antes de chamar LEILÃO, E RECUO DE NOVO
C O SINAL DO custo ótimo na volta do LEILÃO.
C Este código permite ARCS múltipla entre uma linha e uma coluna.
C
C Esta versão do ALGORITHM LEILÃO é adaptativa. AQUI NO MÍNIMO
```

```
INCREMENTO DE LICITAÇÃO C (armazenado no INCR variável) pode ser menor que
C EPSILON. PARA CADA subproblema, INCR COMEÇA no valor do parâmetro
C STARTINCR (que é passado para a rotina LEILÃO), estando aumentada
C por um fator de 2 no final de cada ciclo, até um valor máximo de
C EPSILON. ESTA FUNÇÃO adaptativo é particularmente eficaz para
C PROBLEMAS densa. TI pode ser derrotado por SELEÇÃO STARTINCR = BEGEPS
C
C ********************************************** ******************
C
C o usuário deve fornecer os seguintes dados problema no FRENTE DA ESTRELA FORMATO
C (ou seja, todos os arcos da linha SAME são numerados consecutivamente):
C N = número de linhas (igual número de colunas)
C A = número de arcos
C FOUT (ROW) = Primeiro ARC QUE SAI DE ROW
C COST (ARC) = CUSTO DE ARC
C END (ARC) = COLUNA correspondente ao arco
C
C E os seguintes parâmetros para o algoritmo LEILÃO:
C BEGEPS = Valor inicial EPSILON (não deve ser menor que 1)
C ENDEPS = valor final da EPSILON ANTES ela é definida como 1
C FACTOR = fator pelo qual EPSILON é diminuída ENTRE subproblemas
C STARTINCR = O valor inicial do incremento LICITAÇÃO
ENDEPS C não devem exceder BEGEPS.
C fator deve ser maior do que 1.
C
C FOUT (.) É uma matriz de comprimento N.
C COST (.), END (.) São matrizes de COMPRIMENTO A.
C
C, a solução está contida no ASSIGN Array (.) ONDE
C ASSIGN (COL) Dá o ROW ATRIBUÍDO COL.
C
C Este algoritmo não verifica a inviabilidade do problema.
C TO garantir que o problema é viável o usuário pode ADD
Adicional C ARCS custo muito pequeno.
C
C Este código pode falhar devido a integer overflow IF o número de nós
C ou o intervalo COST (ou ambos) são grandes. Para corrigir esta situação,
```

```
C OS PREÇOS E outras variáveis relacionadas (Max1, max2, TMAX etc.) devem
C DECLARAR AS REAIS precisão dupla.
C ********************************************** ****************
C ALL problema de dados são integer
C ********************************************** ****************


        SUBROUTINE Leilão (BEGEPS, fator, ENDEPS, STARTINCR)

        PARÂMETROS (maxNodes = 10000, MAXARCS = 100000)

         NONE IMPLICIT
         INTEGER NONEWLIST, THRESH, INCR, STARTINCR, INCRFACTOR
         INTEIRO A, K, N, I, J, CURARC, CURCOL
         ROW INTEIRO, FSTARC, FSTCOL, SNDARC, SNDCOL, TMAX, TMIN, BSTCOL
         INTEGER Max1, MAX2, TRDARC, EPSILON, BEGEPS, ENDEPS
         INTEGER M, ISMALL, ILARGE, LARGEINCR, CICLOS
         INTEGER NUMPHASES, LSTARC, NOLIST, OLDROW
         INTEGER FOUT (maxNodes), PCOL (maxNodes), LIST (maxNodes)
        INTEGER ASSIGN (maxNodes)
         INTEGER COST (MAXARCS), END (MAXARCS)
         Real * 8 AVERAGE, FACTOR
        COMUM / ARRAYC / CUSTO / matrizes / END / ARRAYF / FOUT / BK1 / N, A, ISMALL
     $ / BK2 / ciclos, Médio, NUMPHASES / ARRAYA / ASSIGN / PCOLA / PCOL

C ********************************************** **************

Check C ******* VALIDADE DOS PARÂMETROS ******* PASSOU

         IF (BEGEPS.LT.1) THEN
           PRINT *, 'Valor inicial EPSILON é inferior a 1'
           PRINT *, 'EXECUÇÃO ABORTADO'
           PARE
         END IF
         IF (ENDEPS.GT.BEGEPS) THEN
           PRINT *, 'ENDEPS parâmetro é maior que o parâmetro BEGEPS'
           PRINT *, 'ENDEPS é definido no valor padrão de 1'
```

```
          ENDEPS = 1
        END IF
        IF ((FACTOR.LE.1) .E. (BEGEPS.GT.1)) THEN
          PRINT *, 'Epsilon redução de fatores não é maior que 1'
          PRINT *, 'EXECUÇÃO ABORTADO'
          PARE
        END IF
        IF (STARTINCR.LT.1) THEN
          PRINT *, 'MIN DE LICITAÇÃO incremento é menor que 1'
          PRINT *, 'STARTINCR é definido no valor padrão de 1'
          STARTINCR = 1
        END IF

C ******* INITIALIZATION *******

        EPSILON = BEGEPS
        ILARGE = -ISMALL
        LARGEINCR = INT (ILARGE / 10)
        THRESH = INT (0,2 * N)
        INCRFACTOR = 2
        IF (THRESH.GT.100) THRESH = 100
CICLOS x = 1
X MÉDIA = N
         NUMPHASES = 1
         FAZER 10 J = 1, N
           ASSIGN (J) = 0
           PCOL (J) = ISMALL
10 CONTINUAR

        FOUT (N + 1) = A + 1
        NOLIST = N
        FAZER 20 I = 1, N
            LISTA (I) = I
20 CONTINUAR

C ********************************************** ***************
C
```

```
C Esta implementação do leilão ALGORITHM opera em ciclos.
C Cada ciclo consiste em um lance em cada uma das linhas QUE SEJAM
C não atribuído NO INÍCIO DO CICLO (essas linhas são armazenadas em
C lista de matriz (.)). AS que o ciclo progride NOVO
LINHAS C ficarem não atribuídas; Estes são armazenados em Lista (.)
C e apresentará um lance no próximo ciclo.
C NOTA que apenas uma linha apresenta uma proposta de cada vez; Isto é conhecido como
C o método de Gauss-Seidel versão do algoritmo. A VERSÃO onde todos
C ROWS não atribuído apresentar uma proposta AO MESMO TEMPO, é conhecido como de Jacobi
C versão do algoritmo, e não foi implementada. Geralmente
C tende a correr um pouco mais lento do que a versão de Gauss-Seidel, MAS
C admite um grau mais elevado DE paralelização. A VERSÃO JACOBI
C pode ser preferível ON Uma máquina paralela, mas geralmente é INFERIOR
C PARA A VERSÃO Gauss-Seidel EM UMA MÁQUINA DE SÉRIE.
C
C ********************************************** ***************
C
C INÍCIO subproblema (FASE DE ESCALA) W / NEW EPSILON
C
C ********************************************** ***************

12 CONTINUAR

        IF (EPSILON.EQ.1) debulhar = 0
         INCR = STARTINCR
        IF (INCR.GT.EPSILON) INCR = EPSILON

C ********************************************** ***************
C
C início do ciclo LEILÃO COM LISTA NOVO
C
C ********************************************** ***************

15 CONTINUAR

C REINICIAR CONTAGEM DO PRÓXIMO lista de linhas não atribuídos
```

```
        NONEWLIST = 0

C percorrer a lista atual de linhas não atribuídos

        FAZER 100 I = 1, NOLIST
        ROW = LIST (I)
        FSTARC = FOUT (ROW)
        LSTARC = FOUT (ROW + 1) -1
        FSTCOL = END (FSTARC)

C PRIMEIRA CUIDAR DA caso excepcional ROW tem apenas um ARC

        IF (FSTARC.EQ.LSTARC) THEN
          PCOL (FSTCOL) = PCOL (FSTCOL) + LARGEINCR
            OLDROW = ASSIGN (FSTCOL)
            ASSIGN (FSTCOL) = ROW
            IF (OLDROW.GT.0) THEN
              NONEWLIST = + 1 NONEWLIST
              LIST (NONEWLIST) = OLDROW
            END IF
          Ir para 100
        END IF

C tomar em seguida CUIDADOS DO CASO regular, onde linha tem ARCS MÚLTIPLAS

        SNDARC = + 1 FSTARC
        SNDCOL = END (SNDARC)
        Max1 = COST (FSTARC) -PCOL (FSTCOL)
        MAX2 = COST (SNDARC) -PCOL (SNDCOL)
        IF (MAX1.GE.MAX2) THEN
           BSTCOL = FSTCOL
        MAIS
           TMAX = Max1
           Max1 = MAX2
           MAX2 = TMAX
           BSTCOL = SNDCOL
        END IF
```

```
        IF (SNDARC.LT.LSTARC) THEN
        TRDARC = + 1 SNDARC
        DO 40 CURARC = TRDARC, LSTARC
           CURCOL = END (CURARC)
           TMAX = COST (CURARC) -PCOL (CURCOL)
           IF (TMAX.GT.MAX2) THEN
             IF (TMAX.GT.MAX1) THEN
                  MAX2 = Max1
                  Max1 = TMAX
                  BSTCOL = CURCOL
             MAIS
                  MAX2 = TMAX
             END IF
           END IF
40 CONTINUAR
        END IF


APOSTAS C linha para BSTCOL AUMENTAR SEU PREÇO, E é atribuído
C TO BSTCOL, enquanto qualquer ROW ATRIBUÍDO BSTCOL TORNA não atribuídos

         PCOL (BSTCOL) = PCOL (BSTCOL) + Max1-MAX2 + INCR
         OLDROW = ASSIGN (BSTCOL)
         ASSIGN (BSTCOL) = ROW
         IF (OLDROW.GT.0) THEN
           NONEWLIST = + 1 NONEWLIST
           LIST (NONEWLIST) = OLDROW
         END IF

100 CONTINUAR

C ************* FIM DE UM CICLO DE LEILÃO ***************

C OPTIONALLY coletar estatísticas

X Médio = (ciclos * MÉDIA + NOLIST) / (CICLOS + 1)
CICLOS X = CICLOS + 1
```

```
C Verifique se ainda existem ROWS não atribuído 'muitos', isto é, se o
C número de linhas não atribuído É MAIOR DO QUE
C DO THRESH PARAMETER. NÃO SE, REPLACE lista atual com a lista NEW,
C e ir para outro ciclo. Caso contrário, se EPSILON> 1, REDUZIR EPSILON,
C Redefinir a atribuição ao vazio e REINICIAR LEILÃO;
C IF EPSILON = 1 finalizar.
C também aumentam a LICITAÇÃO MINIMAL INCREMENTO até um máximo
C VALOR DO EPSILON (esta é a característica adaptativa)

        INCR = INCR * INCRFACTOR
        IF (INCR.GT.EPSILON) INCR = EPSILON
         IF (NONEWLIST.GT.THRESH) THEN
             NOLIST = NONEWLIST
             IR PARA 15
         END IF

C ********************************************** **************
C
C FIM DE subproblema (escalonamento FASE)
C
C ********************************************** **************

C ****** SE EPSILON É uma RESCINDIR ******

       IF (EPSILON.EQ.1) THEN
          RETURN
       MAIS

C MAIS REDUZIR EPSILON e redefinir o CESSÃO ESVAZIAR

          NUMPHASES + 1 = NUMPHASES
          EPSILON = INT (EPSILON / FACTOR)
          IF (EPSILON.GT.INCR) EPSILON = INT (EPSILON / FACTOR)
           IF ((EPSILON.LT.1) .OR. (EPSILON.LT.ENDEPS)) EPSILON = 1
           THRESH = INT (THRESH / FACTOR)
```

```
X PRINT *, '*** FIM DE UMA FASE DE ESCALA; NEW EPSILON = ', EPSILON

          TMIN = ILARGE
          FAZER 200 J = 1, N
           IF (PCOL (J) .LT.TMIN) TMIN = PCOL (J)
           IF (ASSIGN (J) .GT.0) THEN
            NONEWLIST = + 1 NONEWLIST
            LIST (NONEWLIST) = ASSIGN (J)
            ASSIGN (J) = 0
           END IF
200 CONTINUAR

C Redefinir preço mínimo a ISMALL

          INCR = TMIN-ISMALL
          DO 210 J = 1, N
            PCOL (J) = PCOL (J) -INCR
210 CONTINUAR


C FINAIS PARÂMETROS atualizações antes de voltar para outra ESCALA FASE

          NOLIST = NONEWLIST
          IF (STARTINCR.LT.EPSILON) STARTINCR = FACTOR * STARTINCR
          IR PARA 12
        END IF


        FIM



      SUBROUTINE SETRAN (ISEED)
      IMPLICIT real * 8 (AH, OZ), Integer * 4 (IN)
C ********************************************** **********************
C PORTABLE congruential (uniforme) RANDOM gerador de números:
NEXT_VALUE C = [(7 ** 5) * PREVIOUS_VALUE] MODULO [(2 ** 31) -1]
```

```
C
C Este gerador é composto por duas ROTINAS:
C (1) SETRAN - inicializa constantes e SEED
C (2) RRAN - gera um número aleatório REAIS
C
C O GERADOR DE RODA uma máquina com pelo menos 32 bits de precisão.
C DA SEED (ISEED) deve estar no intervalo (1, (2 ** 31 -1)).
C ********************************************** **********************
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
      IF (ISEED.LT.1) PARADA 77
      MULT = 16807
      MODUL = 2147483647
      I15 = 2 ** 15
      I16 = 2 ** 16
      JRAN = ISEED
      RETURN
      FIM

      RAN função real ()
      IMPLICIT real * 4 (AH, OZ), Integer * 4 (IN)
C ********************************************** **********************
C RAN gera um número aleatório real entre 0 e 1
C ********************************************** **********************
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
      Ixhi = JRAN / I16
      IXLO = JRAN-Ixhi * I16
      IXALO = IXLO * MULT
      LEFTLO = IXALO / I16
      IXAHI = Ixhi * MULT
      IFULHI = IXAHI + LEFTLO
      IRTLO = IXALO-LEFTLO * I16
      IOVER = IFULHI / I15
      IRTHI = IFULHI-IOVER * I15
      JRAN = ((IRTLO-MODUL) + IRTHI * I16) + IOVER
      IF (JRAN.LT.0) JRAN = JRAN + MODUL
      RAN = FLOAT (JRAN) / FLOAT (MODUL)
      RETURN
```

```
      FIM



\ Subsect {AUCTION_FLP} Este código é o mesmo que o anterior excepto que
ele usa a aritmética de ponto flutuante ao atualizar preços. Isso é útil quando
a gama de custo e a dimensão do problema é grande, em cujo caso
os preços pode transbordar o intervalo inteiro no decurso do algoritmo.
No entanto, a aritmética de ponto flutuante retarda o código um pouco.

C ********************************************** *****************
C
C programa de exemplo de chamada para o ALGORITHM LEILÃO
C
C ESTE MOTORISTA cria um problema SIMÉTRICA CESSÃO
C com igual número de linhas e colunas,
C e chama o SUBROUTINE LEILÃO para encontrar um
C atribuição do valor máximo.
C
C MODIFICADO POR DAVID Castanon (OCT. 1991) PARA TRABALHAR COM arbitrariamente
C GAMA grande custo (até ao intervalo inteiro da máquina);
C Para conseguir isso, os preços de objeto e EPSILON SÃO
VARIÁVEIS C precisão dupla, para que os preços NÃO PODE inundarão o
INTEIRO DO GAMA C MACHINE.
C
C ********************************************** *****************

        PARÂMETROS (maxNodes = 10000, MAXARCS = 100000)

        NONE IMPLICIT
        INTEGER N, NA, A, ISMALL, CICLOS
        INTEGER NUMPHASES
        INTEGER I, J, IA, ARC, NOASS, ICOST, ABSCOST, CURARC
        INTEGER CURCOL, FSTARC, LSTARC, MINCOST, MAXCOST, MCOST
        INTEGER FOUT (maxNodes), COLASS (maxNodes)
        INTEGER ASSIGN (maxNodes)
```

```
        INTEGER COST (MAXARCS), END (MAXARCS)
        Real * 8 TT1, TT2, TCOST
        Real * 8 PCOL (maxNodes)
        Real * 8 MÉDIA
        Real * 8 BEGEPS, ENDEPS, FACTOR, STARTINCR, TEST, o lucro
        COMUM / ARRAYC / CUSTO / matrizes / END / ARRAYF / FOUT / BK1 / N, A, ISMALL
     $ / BK2 / ciclos, Médio, NUMPHASES / ARRAYA / ASSIGN / PCOLA / PCOL

C ********************************************** ***************
C
C problema de geração de código começa AQUI
C o usuário pode substituir este código com um código que lê
C HIS / seu problema de um arquivo
C
C ********************************************** ***************

C Este código inclui UM UNIFORME Random Number Generator
C que retorna um valor na (0,1)

C INITIALIZE Gerador Aleatório

      INTEGER MULT, MODUL, I15, I16, JRAN, ISEED
      RAN REAIS
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN

      ISEED = 13502460
      CHAME SETRAN (ISEED)

      PRINT * ', gerando problema de atribuição de SIMÉTRICA "
      PRINT *, "****************************************** '

C **** LER O número de linhas N & o número de arcos A ****

      PRINT *, 'Digite o número de linhas (e colunas)'
      LEIA *, N
5 PRINT *, 'Digite o número de arcos por ROW (1>) "
      LEIA *, NA
```

```
      IF (NA.LT.2) GOTO 5
      PRINT *, 'Digite o mínimo eo custo máximo'
      LEIA *, MINCOST, MAXCOST

C o número de arcos é n * NA

      A = N * NA

C Os arcos incidente a linha i SÃO FOUT (I) PARA FOUT (I + 1) -1
C Além disso, para VIABILIDADE EACH ROW está diretamente ligado
C com a coluna correspondente

      FAZER 20 I = 1, N
        FOUT (I) = 1 + (i-1) * NA
20 CONTINUAR
      FOUT (N + 1) = A + 1

C gerar a END (ARC) eo custo (ARC), que são os COLUNA
C eo coeficiente custo associado a ARC

      FAZER 25 ARC = 1, A
        END (ARC) = 1 + RAN () * N
        IF ((END (ARC) .GT.N) .OR. (END (ARC) .LT.1)) THEN
          PRINT *, 'Erro no problema de geração de'
          PAUSA
          PARE
        END IF
        COST (ARC) = MINCOST + RAN () * (MAXCOST-MINCOST)
25 CONTINUAR

C MODIFICAR O final da última ARC OUT de cada linha de viabilidade
C e defina seu CUSTO PARA -MAXCOST

      FAZER 30 I = 1, N
        END (FOUT (I + 1 -1)) = I
       CUSTO (FOUT (I + 1) -1) = - MAXCOST
30 CONTINUAR
```

```
C
C ********************************************** ***************
C
C PROBLEMA geração de código TERMINA AQUI
C
C ********************************************** ***************

C ESCALA O CUSTO PARA TRABALHAR COM INTEIRO EPSILON

        MAXCOST = 0
        FAZER 35 IA = 1, A
           ABSCOST = IABS (COST (IA))
           IF (ABSCOST.GT.MAXCOST) MAXCOST = ABSCOST
35 CONTINUAR

C *** ISMALL é um inteiro MUITO PEQUENO PARA O SEU APARELHO ***

        ISMALL = -2000000000

        IF (MAXCOST.GT.ABS (ISMALL)) THEN
          PRINT *, 'A faixa de custo é muito grande para INTEGER ARITHMETIC'
          PAUSA
          PARE
        END IF


C os seguintes parâmetros BEGEPS, FACTOR, ENDEPS E STARTINCR
C são passados para o LEILÃO algoritmo. Valores entre
C (A) MAXCOST / 5 E MAXCOST / 2 para BEGEPS
C (B) 4 e 6 para FACTOR
C (C) 1/10 e 1 / (N + 1) PARA ENDEPS
C (D) 1 AND BEGEPS PARA STARTINCR
C têm funcionado bem para PROBLEMAS esparsos de grande porte.
C PARA PROBLEMAS DENSOS RECOMENDA-SE QUE
C BEGEPS ser ajustado para um valor menor,
```

```
ENDEPS C ser definido como 1 / (N + 1),
C STARTINCR ser definido como 1 / (N + 1).

        PRINT *, "****************************************** '
        PRINT *, "custo máximo é ', MAXCOST
        PRINT *, 'Digite o EPSILON PARTIDA'
        LER *, BEGEPS
        IF (BEGEPS.LT.1.0 / (N + 1)) BEGEPS = 1,0 / (N + 1)
        PRINT *, 'Insira o fator EPSILON REDUÇÃO'
        LER *, FACTOR
       ENDEPS = 1,0 / 10,0
        IF (ENDEPS.LT.1.0 / (N + 1)) ENDEPS = 1,0 / (N + 1)
        IF (ENDEPS.GT.BEGEPS) ENDEPS = BEGEPS
        STARTINCR = BEGEPS / 10.0
        IF (STARTINCR.LT.1.0 / (N + 1)) STARTINCR = 1,0 / (N + 1)

        PRINT *, "****************************************** '
        PRINT *, 'começar EPSILON =', BEGEPS
        PRINT *, 'Epsilon redução de fatores =', FACTOR
        PRINT *, 'Epsilon' Limite 'antes ela é definida como 1 / (N + 1) =', ENDEPS
        PRINT *, "COMEÇAR MIN DE LICITAÇÃO INCREMENTO = ', STARTINCR
        PRINT *, "CHAMANDO LEILÃO para resolver o problema"
        PRINT *, "****************************************** '

C GET hora de início para o Mac II

      TT1 = longa (362) /60.0

      CHAMADA Leilão (BEGEPS, fator, ENDEPS, STARTINCR)

C GET TÉRMINO TEMPO PARA O MAC II

      TT2 = longa (362) /60.0 - TT1
      PRINT *, "acabado --- tempo de CPU TOTAL ', TT2,' s '
      PRINT *, "****************************************** '

C *** Mostrar resultados ***
```

```
X Write (9,2010) CICLOS
X2010 FORMATO ('NO LEILÃO DE CICLOS', I7)
X WRITE (9,2020) MÉDIA
X2020 FORMATO ("número médio de propostas por ciclo", F9.3)
        Write (9,2030) NUMPHASES
2030 FORMATO ('NO DE EPSILON subproblemas RESOLVIDO =', I7)

C verificação de viabilidade de uma solução e calcular o custo

        DO 40 I = 1, N
          COLASS (I) = 0
40 CONTINUAR
        NOASS = 0
        FAZER 50 J = 1, N
           IF (ASSIGN (J) .GT.0) THEN
             NOASS = + 1 NOASS
             COLASS (ASSIGN (J)) = J
           END IF
50 CONTINUAR
        IF (NOASS.NE.N) THEN
          PRINT *, '# de linhas atribuído não IGUAL # de linhas'
        END IF
        TCOST = 0
        FAZER 60 I = 1, N
          J = COLASS (I)
          IF (J.EQ.0) THEN
            PRINT *, 'ROW', I, 'não é atribuído "
          END IF
          FSTARC = FOUT (I)
          LSTARC FOUT = (I + 1) -1
          MCOST = ISMALL
          FAZER 70 CURARC = FSTARC, LSTARC
            CURCOL = END (CURARC)
            IF (CURCOL.EQ.J) THEN
              IF (MCOST.LT.COST (CURARC)) THEN
                 MCOST = COST (CURARC)
```

```
             END IF
           END IF
70 CONTINUAR
          LUCRO = MCOST-PCOL (J)
          FAZER 72 CURARC = FSTARC, LSTARC
             J = END (CURARC)
             TEST = COST (CURARC) -PCOL (J) lucrativos
             IF (TEST.GT.1.0 / N) THEN
                PRINT *, '1 / (N + 1) -cs VIOLADOS AT ARC', CURARC
             END IF
72 CONTINUAR
          TCOST = TCOST + MCOST
60 CONTINUAR
        ESCREVA (9,2100) TCOST
2100 FORMATO ('atribuição de custo =', F22.2)
        PRINT *, 'programa terminou; <CR> PARA SAIR '
        PAUSA
        FIM



C ********************************************** *****************
C
C ponto flutuante LEILÃO CÓDIGO DE N por N problemas de atribuição
C
C ESCRITO POR DIMITRI P. Bertsekas
C (MODIFICAÇÕES por David Castanon E PAULO TSENG)
C
C versão 1.2, outubro 1991
C
C
C MODIFICADO POR DAVID Castanon (OCT. 1991) PARA TRABALHAR COM arbitrariamente
C GAMA grande custo (até ao intervalo inteiro da máquina);
C Para conseguir isso, os preços de objeto e EPSILON SÃO
VARIÁVEIS C precisão dupla, para que os preços NÃO PODE inundarão o
INTEIRO DO GAMA C MACHINE.
C
```

```
C ESTE CÓDIGO implementa o algoritmo LEILÃO COM E-escala.
C Ele resolve uma seqüência de subproblemas e diminui
C EPSILON por um fator constante entre subproblemas.
C ESTE versão corresponda a um MODO Gauss-Seidel
C e resolve EPSILON subproblemas inexata.
C
C THE CODE é uma versão melhorada de um antigo (SEPT. 1985)
C CODE LEILÃO COM ESCALA E-ESCRITO POR DIMITRI P. Bertsekas
C
C DO CÓDIGO trata o problema como um problema de maximização.
C para resolver um problema de minimização, inverta o sinal da
C ARC CUSTOS antes de chamar LEILÃO, E RECUO DE NOVO
C O SINAL DO custo ótimo na volta do LEILÃO.
C Este código permite ARCS múltipla entre uma linha e uma coluna.
C
C Esta versão do ALGORITHM LEILÃO é adaptativa. AQUI NO MÍNIMO
INCREMENTO DE LICITAÇÃO C (armazenado no INCR variável) pode ser menor que
C EPSILON. PARA CADA subproblema, INCR COMEÇA no valor do parâmetro
C STARTINCR (que é passado para a rotina LEILÃO), estando aumentada
C por um fator de 2 no final de cada ciclo, até um valor máximo de
C EPSILON. ESTA FUNÇÃO adaptativo é particularmente eficaz para
C PROBLEMAS densa. TI pode ser derrotado por SELEÇÃO STARTINCR = BEGEPS
C
C ********************************************** ******************
C
C o usuário deve fornecer os seguintes dados problema no FRENTE DA ESTRELA FORMATO
C (ou seja, todos os arcos da linha SAME são numerados consecutivamente):
C N = número de linhas (igual número de colunas)
C A = número de arcos
C FOUT (ROW) = Primeiro ARC QUE SAI DE ROW
C COST (ARC) = CUSTO DE ARC
C END (ARC) = COLUNA correspondente ao arco
C
C E os seguintes parâmetros para o algoritmo LEILÃO:
C BEGEPS = Valor inicial EPSILON (não deve ser menor do que 1 / (N + 1))
C ENDEPS = valor final da EPSILON ANTES ela é definida como 1 / (N + 1)
C FACTOR = fator pelo qual EPSILON é diminuída ENTRE subproblemas
```

```
C STARTINCR = O valor inicial do incremento LICITAÇÃO
ENDEPS C não devem exceder BEGEPS.
C fator deve ser maior do que 1.
C
C FOUT (.) É uma matriz de comprimento N.
C COST (.), END (.) São matrizes de COMPRIMENTO A.
C
C, a solução está contida no ASSIGN Array (.) ONDE
C ASSIGN (COL) Dá o ROW ATRIBUÍDO COL.
C
C Este algoritmo não verifica a inviabilidade do problema.
C TO garantir que o problema é viável o usuário pode ADD
Adicional C ARCS custo muito pequeno.
C
C ********************************************** *****************
C ALL problema de dados são integer
C ********************************************** *****************


        SUBROUTINE Leilão (BEGEPS, fator, ENDEPS, STARTINCR)

        PARÂMETROS (maxNodes = 10000, MAXARCS = 100000)

         NONE IMPLICIT
         INTEGER NONEWLIST, THRESH
         INTEIRO A, K, N, I, J, CURARC, CURCOL
         ROW INTEIRO, FSTARC, FSTCOL, SNDARC, SNDCOL, BSTCOL
         INTEGER TRDARC
         INTEGER M, ISMALL, ILARGE, LARGEINCR, CICLOS
         INTEGER NUMPHASES, LSTARC, NOLIST, OLDROW
         INTEGER FOUT (maxNodes), LIST (maxNodes)
        INTEGER ASSIGN (maxNodes)
         INTEGER COST (MAXARCS), END (MAXARCS)
        Real * 8 BEGEPS, ENDEPS, FACTOR, STARTINCR, INCRFACTOR
        Real * 8 PCOL (maxNodes)
        Real * 8 Max1, MAX2, TMAX, EPSILON, INCR
         Real * 8 MÉDIA
```

```
        COMUM / ARRAYC / CUSTO / matrizes / END / ARRAYF / FOUT / BK1 / N, A, ISMALL
     $ / BK2 / ciclos, Médio, NUMPHASES / ARRAYA / ASSIGN / PCOLA / PCOL

C ********************************************** **************

Check C ******* VALIDADE DOS PARÂMETROS ******* PASSOU

        IF (BEGEPS.LE.1.0 / (N + 2)) THEN
          PRINT *, 'Valor inicial EPSILON é inferior a 1 / (N + 1)'
          PRINT *, 'EXECUÇÃO ABORTADO'
          PARE
        END IF
        IF (ENDEPS.GT.BEGEPS) THEN
          PRINT *, 'ENDEPS parâmetro é maior que o parâmetro BEGEPS'
          PRINT *, 'ENDEPS é definido no valor padrão de 1 / (N + 1)'
          ENDEPS = 1,0 / (N + 1)
        END IF
        IF ((FACTOR.LE.1) .E. (BEGEPS.GE.1.0 / N)) THEN
          PRINT *, 'Epsilon redução de fatores não é maior que 1'
          PRINT *, 'EXECUÇÃO ABORTADO'
          PARE
        END IF
        IF (STARTINCR.LE.1.0 / (N + 2)) THEN
          PRINT *, 'MIN DE LICITAÇÃO incremento é menor que 1 / (N + 1)'
          PRINT *, 'STARTINCR é definido no valor padrão de 1 / (N + 1)'
          STARTINCR = 1,0 / (N + 1)
        END IF

C ******* INITIALIZATION *******

        EPSILON = BEGEPS
        ILARGE = -ISMALL
        LARGEINCR = INT (ILARGE / 10)
        THRESH = INT (0,2 * N)
        INCRFACTOR = 2,0
        IF (THRESH.GT.100) THRESH = 100
CICLOS x = 1
```

```
X MÉDIA = N
         NUMPHASES = 1
         FAZER 10 J = 1, N
           ASSIGN (J) = 0
           PCOL (J) = ISMALL
10 CONTINUAR

        FOUT (N + 1) = A + 1
        NOLIST = N
        FAZER 20 I = 1, N
            LISTA (I) = I
20 CONTINUAR

C ********************************************** ***************
C
C Esta implementação do leilão ALGORITHM opera em ciclos.
C Cada ciclo consiste em um lance em cada uma das linhas QUE SEJAM
C não atribuído NO INÍCIO DO CICLO (essas linhas são armazenadas em
C lista de matriz (.)). AS que o ciclo progride NOVO
LINHAS C ficarem não atribuídas; Estes são armazenados em Lista (.)
C e apresentará um lance no próximo ciclo.
C NOTA que apenas uma linha apresenta uma proposta de cada vez; Isto é conhecido como
C o método de Gauss-Seidel versão do algoritmo. A VERSÃO onde todos
C ROWS não atribuído apresentar uma proposta AO MESMO TEMPO, é conhecido como de Jacobi
C versão do algoritmo, e não foi implementada. Geralmente
C tende a correr um pouco mais lento do que a versão de Gauss-Seidel, MAS
C admite um grau mais elevado DE paralelização. A VERSÃO JACOBI
C pode ser preferível ON Uma máquina paralela, mas geralmente é INFERIOR
C PARA A VERSÃO Gauss-Seidel EM UMA MÁQUINA DE SÉRIE.
C
C ********************************************** ***************
C
C INÍCIO subproblema (FASE DE ESCALA) W / NEW EPSILON
C
C ********************************************** ***************

12 CONTINUAR
```

```
        IF (EPSILON.LE.1.0 / (N + 1)) debulhar = 0
        INCR = STARTINCR
        IF (INCR.GT.EPSILON) INCR = EPSILON

C ********************************************** **************
C
C início do ciclo LEILÃO COM LISTA NOVO
C
C ********************************************** **************

15 CONTINUAR

C REINICIAR CONTAGEM DO PRÓXIMO lista de linhas não atribuídos

        NONEWLIST = 0

C percorrer a lista atual de linhas não atribuídos

        FAZER 100 I = 1, NOLIST
        ROW = LIST (I)
        FSTARC = FOUT (ROW)
        LSTARC = FOUT (ROW + 1) -1
        FSTCOL = END (FSTARC)

C PRIMEIRA CUIDAR DA caso excepcional ROW tem apenas um ARC

        IF (FSTARC.EQ.LSTARC) THEN
          PCOL (FSTCOL) = PCOL (FSTCOL) + LARGEINCR
          OLDROW = ASSIGN (FSTCOL)
          ASSIGN (FSTCOL) = ROW
          IF (OLDROW.GT.0) THEN
            NONEWLIST = + 1 NONEWLIST
            LIST (NONEWLIST) = OLDROW
          END IF
          Ir para 100
        END IF
```

```
C tomar em seguida CUIDADOS DO CASO regular, onde linha tem ARCS MÚLTIPLAS

         SNDARC = + 1 FSTARC
         SNDCOL = END (SNDARC)
         Max1 = COST (FSTARC) -PCOL (FSTCOL)
         MAX2 = COST (SNDARC) -PCOL (SNDCOL)
         IF (MAX1.GE.MAX2) THEN
            BSTCOL = FSTCOL
         MAIS
            TMAX = Max1
            Max1 = MAX2
            MAX2 = TMAX
            BSTCOL = SNDCOL
         END IF
         IF (SNDARC.LT.LSTARC) THEN
         TRDARC = + 1 SNDARC
         DO 40 CURARC = TRDARC, LSTARC
            CURCOL = END (CURARC)
            TMAX = COST (CURARC) -PCOL (CURCOL)
            IF (TMAX.GT.MAX2) THEN
              IF (TMAX.GT.MAX1) THEN
                   MAX2 = Max1
                   Max1 = TMAX
                   BSTCOL = CURCOL
              MAIS
                   MAX2 = TMAX
              END IF
            END IF
40 CONTINUAR
         END IF


APOSTAS C linha para BSTCOL AUMENTAR SEU PREÇO, E é atribuído
C TO BSTCOL, enquanto qualquer ROW ATRIBUÍDO BSTCOL TORNA não atribuídos

          PCOL (BSTCOL) = PCOL (BSTCOL) + Max1-MAX2 + INCR
```

```
          OLDROW = ASSIGN (BSTCOL)
          ASSIGN (BSTCOL) = ROW
          IF (OLDROW.GT.0) THEN
            NONEWLIST = + 1 NONEWLIST
            LIST (NONEWLIST) = OLDROW
          END IF

100 CONTINUAR

C ************* FIM DE UM CICLO DE LEILÃO ***************

C OPTIONALLY coletar estatísticas

X Médio = (ciclos * MÉDIA + NOLIST) / (CICLOS + 1)
CICLOS X = CICLOS + 1

C Verifique se ainda existem ROWS não atribuído 'muitos', isto é, se o
C número de linhas não atribuído É MAIOR DO QUE
C DO THRESH PARAMETER. NÃO SE, REPLACE lista atual com a lista NEW,
C e ir para outro ciclo. Caso contrário, se EPSILON> 1, REDUZIR EPSILON,
C Redefinir a atribuição ao vazio e REINICIAR LEILÃO;
C IF EPSILON = 1 finalizar.
C também aumentam a LICITAÇÃO MINIMAL INCREMENTO até um máximo
C VALOR DO EPSILON (esta é a característica adaptativa)

         INCR = INCR * INCRFACTOR
         IF (INCR.GT.EPSILON) INCR = EPSILON
          IF (NONEWLIST.GT.THRESH) THEN
              NOLIST = NONEWLIST
              IR PARA 15
          END IF

C ********************************************** **************
C
C FIM DE subproblema (escalonamento FASE)
C
C ********************************************** **************
```

```
C ****** SE EPSILON É uma RESCINDIR ******

       IF (EPSILON.LE.1.0 / N) THEN
          RETURN
       MAIS

C MAIS REDUZIR EPSILON e redefinir o CESSÃO ESVAZIAR

          NUMPHASES + 1 = NUMPHASES
          EPSILON = EPSILON / FACTOR
          IF (EPSILON.GT.INCR) EPSILON = EPSILON / FACTOR
          IF ((EPSILON.LT.1.0 / N) .OR. (EPSILON.LT.ENDEPS)) THEN
             EPSILON = 1,0 / (N + 1)
          END IF
          THRESH = INT (THRESH / FACTOR)

X PRINT *, '*** FIM DE UMA FASE DE ESCALA; NEW EPSILON = ', EPSILON

          FAZER 200 J = 1, N
           IF (ASSIGN (J) .GT.0) THEN
            NONEWLIST = + 1 NONEWLIST
            LIST (NONEWLIST) = ASSIGN (J)
            ASSIGN (J) = 0
           END IF
200 CONTINUAR



C FINAIS PARÂMETROS atualizações antes de voltar para outra ESCALA FASE

          NOLIST = NONEWLIST
          IF (STARTINCR.LT.EPSILON) STARTINCR = FACTOR * STARTINCR
          IR PARA 12
       END IF
```

```
        FIM



      SUBROUTINE SETRAN (ISEED)
      IMPLICIT real * 8 (AH, OZ), Integer * 4 (IN)
C ********************************************** **********************
C PORTABLE congruential (uniforme) RANDOM gerador de números:
NEXT_VALUE C = [(7 ** 5) * PREVIOUS_VALUE] MODULO [(2 ** 31) -1]
C
C Este gerador é composto por duas ROTINAS:
C (1) SETRAN - inicializa constantes e SEED
C (2) RRAN - gera um número aleatório REAIS
C
C O GERADOR DE RODA uma máquina com pelo menos 32 bits de precisão.
C DA SEED (ISEED) deve estar no intervalo (1, (2 ** 31 -1)).
C ********************************************** **********************
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
      IF (ISEED.LT.1) PARADA 77
      MULT = 16807
      MODUL = 2147483647
      I15 = 2 ** 15
      I16 = 2 ** 16
      JRAN = ISEED
      RETURN
      FIM

      RAN função real ()
      IMPLICIT real * 4 (AH, OZ), Integer * 4 (IN)
C ********************************************** **********************
C RAN gera um número aleatório real entre 0 e 1
C ********************************************** **********************
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
      Ixhi = JRAN / I16
      IXLO = JRAN-Ixhi * I16
      IXALO = IXLO * MULT
      LEFTLO = IXALO / I16
```

```
      IXAHI = Ixhi * MULT
      IFULHI = IXAHI + LEFTLO
      IRTLO = IXALO-LEFTLO * I16
      IOVER = IFULHI / I15
      IRTHI = IFULHI-IOVER * I15
      JRAN = ((IRTLO-MODUL) + IRTHI * I16) + IOVER
      IF (JRAN.LT.0) JRAN = JRAN + MODUL
      RAN = FLOAT (JRAN) / FLOAT (MODUL)
      RETURN
      FIM
```

\ Subsect {LEILÃO-AS} Esse código implementa o leilão para a frente / verso
algoritmo com $ \ e $ scaling para o problema de atribuição assimétrica; cf. \
Seção 4.2.1 do livro de otimização de rede do autor.
Ele também pode resolver como um caso especial do problema de afectação simétrica.
Neste caso, bem como no caso em que $ \ $ e scaling é contornado ($ \ e = 1 $ em todo o
algoritmo), apenas a parte da frente do algoritmo é utilizado. Note também que este código usa o
`` Terceiro melhor '' implementação do objeto descrito no Exercício 1.7 da Seção 4.1 da rede
livro de otimização.

```
C ********************************************** *****************
C
C programa de amostra CHAMADA PARA LEILÃO ALGORITHM
C para o problema de atribuição ASYMMETRIC
C
C ESTE MOTORISTA cria um problema de atribuição ASYMMETRIC
C COM menor ou igual número de linhas que as colunas,
C e chama o SUBROUTINE AUCTION_AS para encontrar um
C atribuição do valor máximo.
C
C esta versão usa um terceiro melhor VALOR
C
C ********************************************** *****************
```

```
        PARÂMETROS (maxNodes = 10000, MAXARCS = 100000)

        NONE IMPLICIT
        INTEGER M, N, NA, A, IA, K, ILARGE, BEGEPS, ENDEPS, CICLOS
        INTEGER NUMPHASES, STARTINCR, LAMBDA
        INTEGER I, J, ARC, NOASS, ICOST, ABSCOST, CURARC
        INTEGER CURCOL, FSTARC, LSTARC, MINCOST, MAXCOST, MCOST
        INTEGER EXTRA, o restante, INDEX, COUNT
        INTEGER COL_ASSIGNED_TO (maxNodes), ROW_ASSIGNED_TO (maxNodes)
        INTEGER FOUT (maxNodes), FIN (maxNodes), NXTIN (MAXARCS)
        CUSTO INTEIRO (MAXARCS), START (MAXARCS), END (MAXARCS)
        INTEGER PCOL (maxNodes), PROW (maxNodes)
        INTEGER PRDARC (maxNodes)
        Real * 8 FACTOR, TT1, TT2, TCOST, AVERAGE
        COMUM / ARRAYC / CUSTO / matrizes / START / ARRAYE / END
        COMUM / ARRAYFO / FOUT / ARRAYFI / FIN / ARRAYNI / NXTIN
        COMUM / ARRAYRA / ROW_ASSIGNED_TO / ARRAYCA / COL_ASSIGNED_TO
        COMUM / ARRAYPC / PCOL / ARRAYPR / PROW
        COMUM / BK1 / M, N, A, ILARGE
        Common / BK2 / ciclos, Médio, NUMPHASES, LAMBDA

C ********************************************** ***************
C
C problema de geração de código começa AQUI
C o usuário pode substituir este código com um código que lê
C HIS / seu problema de um arquivo
C
C ********************************************** ***************

C Este código inclui UM UNIFORME Random Number Generator
C que retorna um valor na (0,1)

C INITIALIZE Gerador Aleatório

      INTEGER MULT, MODUL, I15, I16, JRAN, ISEED
      RAN REAIS
```

```
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN

      ISEED = 13502460
      CHAME SETRAN (ISEED)

      PRINT *, "gerando um problema de atribuição ASYMMETRIC '
      PRINT *, "****************************************** '

C **** LER O número de linhas M, N colunas e arcos A ****

      PRINT *, 'Digite o número de linhas'
      LEIA *, M

2 PRINT *, 'Digite o número de colunas (> = # de linhas)'
      LEIA *, N
      IF (N.LT.M) ir para 2

5 PRINT *, 'Digite o número de arcos por ROW (1>) "
      LEIA *, NA
      IF (NA.LT.2) GOTO 5

      PRINT *, 'Digite o mínimo eo custo máximo'
      LEIA *, MINCOST, MAXCOST

C o número de arcos é M * NA + NM

      A = H * + NA NM
      PRINT *, "o número de arcos IS ', A

C Os arcos incidente a linha i SÃO FOUT (I) PARA FOUT (I + 1) -1
C Além disso, para VIABILIDADE EACH ROW está diretamente ligado
C com a coluna correspondente;
C TAMBÉM cada coluna tem pelo menos um ARC

      EXTRA = INT ((NM) / M)

      FAZER 20 I = 1, M
```

```
        FOUT (I) = 1 + (i-1) * (EXTRA NA +)
20 CONTINUAR
      FOUT (M + 1) = A + 1

      FAZER 22 I = 1, M
        FAZER 23 ARC = FOUT (I), FOUT (I + 1) -1
          START (ARC) = I
23 CONTINUAR
22 CONTINUAR

C gerar a END (ARC) eo custo (ARC), que são os COLUNA
C eo coeficiente custo associado a ARC

      FAZER 25 ARC = 1, A
        END (ARC) = 1 + RAN () * N
        IF ((END (ARC) .GT.N) .OR. (END (ARC) .LT.1)) THEN
          PRINT *, 'Erro no problema de geração de'
          PAUSA
          PARE
        END IF
        COST (ARC) = MINCOST + RAN () * (MAXCOST-MINCOST)
25 CONTINUAR

C MODIFICAR O final da última ARC OUT de cada linha de viabilidade
C e defina seu CUSTO PARA -MAXCOST

      FAZER 30 I = 1, M
        END (FOUT (I + 1 -1)) = I
        CUSTO (FOUT (I + 1) -1) = - MAXCOST
30 CONTINUAR

C MODIFICAR final de algumas ARCS modo que cada coluna tem pelo menos um ARC

      IF (EXTRA.GE.1) THEN
        FAZER 32 I = 1, M
          FAZER 33 K = 1, EXTRA
            END (FOUT (I) + K-1) = K * M + I
```

```
33 CONTINUAR
32 CONTINUAR
       END IF

       REMAINDER = NM * (1 + EXTRA)
       IF (REMAINDER.GE.1) THEN
         INDEX = FOUT (M) + EXTRA-1
         FAZER 35 J = 1, REMAINDER
          END (INDEX + J) = (1 + EXTRA) * M + J
35 CONTINUAR
       END IF

C construir a FIN () e NXTIN () ARRAYS

      DO 42 J = 1, N
        FIN (J) = 0
        PRDARC (J) = 0
42 CONTINUAR

      FAZER 43 ARC = 1, A
        NXTIN (ARC) = 0
        J = END (ARC)
        IF (FIN (J) .NE.0) THEN
          NXTIN (PRDARC (J)) = ARC
        MAIS
          FIN (J) = ARC
        END IF
        PRDARC (J) = ARC
43 CONTINUAR


C ********************************************** ***************
C
C PROBLEMA geração de código TERMINA AQUI
C
C ********************************************** ***************
```

```
C ESCALA O CUSTO PARA TRABALHAR COM INTEIRO EPSILON

        MAXCOST = 0
        FAZER 45 IA = 1, A
           ABSCOST = IABS (COST (IA))
           IF (ABSCOST.GT.MAXCOST) MAXCOST = ABSCOST
           COST (IA) = CUSTO (IA) * (N + 1)
45 CONTINUAR

C *** ILARGE é um inteiro MUITO GRANDE PARA SUA MÁQUINA ***

        ILARGE = 2000000000

        IF (MAXCOST.GT.INT (ILARGE / (N + 1))) THEN
          PRINT *, 'A faixa de custo é muito grande para INTEGER ARITHMETIC'
          PAUSA
          PARE
        END IF

        MAXCOST MAXCOST = * (N + 1)

        LAMBDA = -ILARGE

C os seguintes parâmetros BEGEPS, FACTOR, ENDEPS E STARTINCR
C são passados para o LEILÃO algoritmo. Valores entre
C (A) MAXCOST / 5 E MAXCOST / 2 para BEGEPS
C (B) 4 e 6 para FACTOR
C (C) M / 10 e 1 para ENDEPS
C (D) 1 AND BEGEPS PARA STARTINCR
C têm funcionado bem para PROBLEMAS esparsos de grande porte.
C PARA PROBLEMAS PARA densa e problemas muito ASSIMÉTRICAS
C RECOMENDA-SE QUE
C BEGEPS ser ajustado para um valor menor (possivelmente 1),
ENDEPS C ser definido como 1,
C STARTINCR ser definido como 1.
```

```
C PARA problemas muito assimétrico, BEGEPS = 1, correspondendo a
C NO E-dimensionamento, pode funcionar melhor e deve ser pelo menos tentei.

        PRINT *, "custo máximo é ', MAXCOST
        PRINT *, 'Digite o EPSILON PARTIDA'
        LER *, BEGEPS
        IF (BEGEPS.LT.1) BEGEPS = 1
        PRINT *, 'Insira o fator EPSILON REDUÇÃO'
        LER *, FACTOR
       ENDEPS = H / 10
        IF (ENDEPS.LT.1) ENDEPS = 1
        IF (ENDEPS.GT.BEGEPS) ENDEPS = BEGEPS
        STARTINCR = BEGEPS / 10
        IF (STARTINCR.LT.1) STARTINCR = 1

        PRINT *, "****************************************** '
        PRINT *, 'começar EPSILON =', BEGEPS
        PRINT *, 'Epsilon redução de fatores =', FACTOR
        PRINT *, «limiar EPSILON ANTES ela é definida como 1 = ', ENDEPS
        PRINT *, "COMEÇAR MIN DE LICITAÇÃO INCREMENTO = ', STARTINCR
        PRINT *, "CHAMANDO ASYMMETRIC LEILÃO CESSÃO '
        PRINT *, "****************************************** '

C GET hora de início para o Mac II

      TT1 = longa (362) /60.0

      CHAMADA AUCTION_AS (BEGEPS, fator, ENDEPS, STARTINCR)

C GET TÉRMINO TEMPO PARA O MAC II

      TT2 = longa (362) /60.0 - TT1
      PRINT *, "acabado --- tempo de CPU TOTAL ', TT2,' s '
      PRINT *, "****************************************** '

C *** Mostrar resultados ***
```

```
X Write (9,2010) CICLOS
X2010 FORMATO ('NO LEILÃO DE CICLOS', I7)
X WRITE (9,2020) MÉDIA
X2020 FORMATO ("número médio de propostas por ciclo", F9.3)
       Write (9,2030) NUMPHASES
2030 FORMATO ('NO DE EPSILON subproblemas RESOLVIDO =', I7)

C VERIFICAÇÃO Optimality & calcular o custo

       FAZER 48 J = 1, N
         I = ROW_ASSIGNED_TO (J)
         IF (N.GT.M) THEN
           ROW_ASSIGNED_TO (J) = 0
         MAIS
           IF (I.GT.0) THEN
             COL_ASSIGNED_TO (I) = J
           END IF
         END IF
48 CONTINUAR


        TCOST = 0
        FAZER 50 I = 1, M
          J = COL_ASSIGNED_TO (I)
          ROW_ASSIGNED_TO (J) = I
          IF (J.EQ.0) THEN
             PRINT *, 'ROW', I, 'não é atribuído "
          END IF
          IF (PCOL (J) .LT.LAMBDA) THEN
             PRINT *, 'CS VIOLAÇÃO EM COLUNA afectadas », J
          END IF
          FSTARC = FOUT (I)
          LSTARC FOUT = (I + 1) -1
          MCOST = -ILARGE
          FAZER 55 ARC = FSTARC, LSTARC
            CURCOL = END (ARC)
```

```
            IF (PROW (I) + PCOL (CURCOL) .LT.COST (ARC -1)) THEN
              PRINT *, '1-CS VIOLADOS AT ARC ", ARC
            END IF
            IF (CURCOL.EQ.J) THEN
              IF (MCOST.LT.COST (ARC)) THEN
                 MCOST = CUSTO (ARC)
              END IF
            END IF
55 CONTINUAR
          TCOST = TCOST MCOST + / (M + 1)
          IF (PROW (I) + PCOL (J) .NE.MCOST) THEN
            PRINT *, '1-CS VIOLADA na linha', I
          END IF
50 CONTINUAR

        COUNT = 0
        DO 60 J = 1, N
           IF (ROW_ASSIGNED_TO (J) .EQ.0) THEN
             IF (PCOL (J) .GT.LAMBDA) THEN
                PRINT *, 'CS VIOLAÇÃO EM COLUNA não atribuído', J
             END IF
           MAIS
             COUNT = COUNT + 1
           END IF
60 CONTINUAR

        IF (COUNT.LT.M) THEN
          PRINT *, "o número de colunas atribuído é errado"
        END IF


        ESCREVA (9,2100) TCOST
2100 FORMATO ('atribuição de custo =', F18.2)
        PRINT *, 'programa terminou; <CR> PARA SAIR '
        PAUSA
        FIM
```

```
C ********************************************** *****************
C
C CODE LEILÃO PARA ASYMMETRIC M por N problemas de atribuição
C
C ESCRITO POR DIMITRI P. Bertsekas
C
C dezembro 1990
C
C ESTE CÓDIGO implementa um avanço / retrocesso ALGORITHM LEILÃO
C COM E-escala para o problema de atribuição assimétrica.
C Ele resolve uma seqüência de subproblemas e diminui
C EPSILON por um fator constante entre subproblemas.
C ESTE versão corresponda a um MODO Gauss-Seidel.
C
C DO CÓDIGO trata o problema como um problema de maximização.
C para resolver um problema de minimização, inverta o sinal da
C ARC CUSTOS antes de chamar LEILÃO, E RECUO DE NOVO
C O SINAL DO custo ótimo na volta do LEILÃO.
C Este código permite ARCS múltipla entre uma linha e uma coluna.
C
C Esta versão do ALGORITHM LEILÃO é adaptativa. AQUI NO MÍNIMO
INCREMENTO DE LICITAÇÃO C (armazenado no INCR variável) pode ser menor que
C EPSILON. PARA CADA subproblema, INCR COMEÇA no valor do parâmetro
C STARTINCR (que é passado para a rotina LEILÃO), estando aumentada
C por um fator de 2 no final de cada ciclo, até um valor máximo de
C EPSILON. ESTA FUNÇÃO adaptativo é particularmente eficaz para
C PROBLEMAS densa. TI pode ser derrotado por SELEÇÃO STARTINCR = BEGEPS
C
C ********************************************** ******************
C
C o usuário deve fornecer os seguintes dados problema no FRENTE DA ESTRELA FORMATO
C (ou seja, todos os arcos da linha SAME são numerados consecutivamente):
C M = número de linhas
C N = número de colunas
```

```
C A = número de arcos
C FOUT (ROW) = Primeiro ARC QUE SAI DE ROW
C FIN (COL) = PRIMEIRO ARC VINDO EM COL
C NXTIN (ARC) = Próxima ARC incidente à COLUNA MESMO AS ARC
C COST (ARC) = CUSTO DE ARC
C START (ARC) = ROW correspondente ao arco
C END (ARC) = COLUNA correspondente ao arco
C
C E os seguintes parâmetros para o algoritmo LEILÃO:
C BEGEPS = Valor inicial EPSILON (não deve ser menor que 1)
C ENDEPS = valor final da EPSILON ANTES ela é definida como 1
C FACTOR = fator pelo qual EPSILON é diminuída ENTRE subproblemas
C STARTINCR = O valor inicial do incremento LICITAÇÃO
ENDEPS C não devem exceder BEGEPS.
C fator deve ser maior do que 1.
C
C FOUT (.) É uma matriz de comprimento N.
C COST (.), END (.) São matrizes de COMPRIMENTO A.
C
C a solução é contidos na matriz ROW_ASSIGNED_TO (.) ONDE
C ROW_ASSIGNED_TO (COL) Dá o ROW ATRIBUÍDO COL.
C TAMBÉM COL_ASSIGNED_TO (ROW) DÁ A COLUNA ATRIBUÍDO linha.
C
C Este algoritmo não verifica a inviabilidade do problema.
C TO garantir que o problema é viável o usuário pode ADD
Adicional C ARCS custo muito pequeno.
C
C Este código pode falhar devido a integer overflow IF o número de nós
C ou o intervalo COST (ou ambos) são grandes. Para corrigir esta situação,
C OS PREÇOS E outras variáveis relacionadas (Max1, max2, TMAX etc.) devem
C DECLARAR AS REAIS precisão dupla.
C ********************************************** *****************
C ALL problema de dados são integer
C ********************************************** *****************
```

```
        SUBROUTINE AUCTION_AS (BEGEPS, fator, ENDEPS, STARTINCR)

        PARÂMETROS (maxNodes = 10000, MAXARCS = 100000)

         NONE IMPLICIT
         INTEIRO A, K, N, I, J, M, CURARC, CURCOL, Currow, LAMBDA, DELTA
         INTEGER THRESH, INCR, STARTINCR, INCRFACTOR, EPS_THRESH
         INTEGER NOLIST, RNOLIST, NONEWLIST, RNONEWLIST
         INTEGER linha, coluna BSTROW, BSTCOL
         INTEGER FSTARC, FSTCOL, NXTARC, NXTROW, LSTARC, SNDARC, SNDCOL
         INTEGER Max1, MAX2, TMAX, TMIN, TRDARC, EPSILON, BEGEPS, ENDEPS
         INTEGER ISMALL, ILARGE, LARGEINCR, CICLOS
         INTEGER NUMPHASES, OLDROW, OLDCOL
        INTEGER MAX3, BSTARC, SBSTARC
        INTEGER CUR_BARC, TRDVAL (maxNodes)
        LÓGICO INIT
        INTEGER BEST_ARC (maxNodes), SECD_ARC (maxNodes)
         INTEGER COL_ASSIGNED_TO (maxNodes), ROW_ASSIGNED_TO (maxNodes)
         INTEGER FOUT (maxNodes), FIN (maxNodes), NXTIN (MAXARCS)
        CUSTO INTEIRO (MAXARCS), START (MAXARCS), END (MAXARCS)
        LISTA INTEIRO (maxNodes), REV_LIST (maxNodes)
        INTEGER PCOL (maxNodes), PROW (maxNodes)
         Real * 8 AVERAGE, FACTOR
        COMUM / ARRAYC / CUSTO / matrizes / START / ARRAYE / END
         COMUM / ARRAYFO / FOUT / ARRAYFI / FIN / ARRAYNI / NXTIN
         COMUM / ARRAYRA / ROW_ASSIGNED_TO / ARRAYCA / COL_ASSIGNED_TO
         COMUM / ARRAYPC / PCOL / ARRAYPR / PROW
         COMUM / BK1 / M, N, A, ILARGE
        Common / BK2 / ciclos, Médio, NUMPHASES, LAMBDA

C ********************************************** **************

Check C ******* VALIDADE DOS PARÂMETROS ******* PASSOU

         IF (BEGEPS.LT.1) THEN
           PRINT *, 'Valor inicial EPSILON é inferior a 1'
           PRINT *, 'EXECUÇÃO ABORTADO'
```

```
          PARE
        END IF
        IF (ENDEPS.GT.BEGEPS) THEN
          PRINT *, 'ENDEPS parâmetro é maior que o parâmetro BEGEPS'
          PRINT *, 'ENDEPS é definido no valor padrão de 1'
          ENDEPS = 1
        END IF
        IF ((FACTOR.LE.1) .E. (BEGEPS.GT.1)) THEN
          PRINT *, 'Epsilon redução de fatores não é maior que 1'
          PRINT *, 'EXECUÇÃO ABORTADO'
          PARE
        END IF
        IF (STARTINCR.LT.1) THEN
          PRINT *, 'MIN DE LICITAÇÃO incremento é menor que 1'
          PRINT *, 'STARTINCR é definido no valor padrão de 1'
          STARTINCR = 1
        END IF

C ******* INITIALIZATION *******

         EPSILON = BEGEPS
         ISMALL = -ILARGE
         LARGEINCR = INT (ILARGE / 10)
         THRESH = INT (0,2 * M)
         INCRFACTOR = 2
         EPS_THRESH = BEGEPS / (FACTOR ** 3)
         IF (EPS_THRESH.LT.2) EPS_THRESH = 2
         INIT = .FALSE.
         IF (THRESH.GT.100) THRESH = 100
CICLOS x = 1
X MÉDIA = N
         NUMPHASES = 1

         FAZER 10 I = 1, N
           PCOL (I) = ISMALL
10 CONTINUAR
```

```
         FOUT (M + 1) = A + 1

C ********************************************** ***************
C
C Esta implementação do leilão ALGORITHM opera em ciclos.
C Cada ciclo consiste em um lance em cada uma das linhas QUE SEJAM
C não atribuído NO INÍCIO DO CICLO (essas linhas são armazenadas em
C lista de matriz (.)). AS que o ciclo progride NOVO
LINHAS C ficarem não atribuídas; Estes são armazenados em Lista (.)
C e apresentará um lance no próximo ciclo.
C
C O primeiro algoritmo encontra uma atribuição possível, onde todos
C forem atribuídos, USANDO UM COMBINADO COM EXPECTATIVA / leilão reverso.
C ENTÃO O algoritmo usa um leilão reverso modificado para PARA SATISFAZER
C DO RESTANTE CONDIÇÕES E-CS.
C
C ********************************************** ***************
C
C INÍCIO subproblema (FASE DE ESCALA) W / NEW EPSILON
C
C ********************************************** ***************

12 CONTINUAR

C INITIALIZE linha lista ATRIBUIÇÃO

        NOLIST = M
        FAZER 20 I = 1, M
           LISTA (I) = I
20 CONTINUAR

        DO 22 J = 1, N
           ROW_ASSIGNED_TO (J) = 0
22 CONTINUAR

        INCR = STARTINCR
        IF (INCR.GT.EPSILON) INCR = EPSILON
```

```
        IF (EPSILON.EQ.1) debulhar = 0


C ********************************************** ***************
C
C partida de avanço LEILÃO CICLO
C
C ********************************************** ***************

      IF (EPSILON.LT.EPS_THRESH) THEN

17 CONTINUAR

C REINICIAR CONTAGEM DO PRÓXIMO lista de linhas não atribuídos

        NONEWLIST = 0

C percorrer a lista atual de linhas não atribuídos

        FAZER 103 I = 1, NOLIST
          ROW = LIST (I)
          FSTARC = FOUT (ROW)
          LSTARC = FOUT (ROW + 1) -1
          FSTCOL = END (FSTARC)

C PRIMEIRA CUIDAR DA caso excepcional ROW tem apenas um ARC

          IF (FSTARC.EQ.LSTARC) THEN
            PCOL (FSTCOL) = PCOL (FSTCOL) + LARGEINCR
            PROW (ROW) = COST (FSTARC) -PCOL (FSTCOL)
            OLDROW = ROW_ASSIGNED_TO (FSTCOL)
            ROW_ASSIGNED_TO (FSTCOL) = ROW
            IF (OLDROW.GT.0) THEN
              NONEWLIST = + 1 NONEWLIST
              LIST (NONEWLIST) = OLDROW
            END IF
            IR PARA 103
```

```
          END IF

C tomar em seguida CUIDADOS DO CASO regular, onde linha tem ARCS MÚLTIPLAS

          IF (((FSTARC + 2) .LE.LSTARC) .E. (INIT)) THEN

            BSTARC = BEST_ARC (ROW)
            SBSTARC = SECD_ARC (ROW)
            BSTCOL = END (BSTARC)
            SNDCOL = END (SECD_ARC (ROW))

            Max1 = COST (BSTARC) -PCOL (BSTCOL)
            MAX2 = COST (SBSTARC) -PCOL (SNDCOL)
            IF ((MAX1.GE.TRDVAL (ROW)). E. (MAX2.GE.TRDVAL (ROW))) THEN

                IF (MAX1.LT.MAX2) THEN
                  TMAX = Max1
                  Max1 = MAX2
                  MAX2 = TMAX
                  BSTCOL = SNDCOL
                END IF
                IR PARA 70

             END IF
          END IF

          SNDARC = + 1 FSTARC
          SNDCOL = END (SNDARC)
          Max1 = COST (FSTARC) -PCOL (FSTCOL)
          MAX2 = COST (SNDARC) -PCOL (SNDCOL)
          IF (MAX1.GE.MAX2) THEN
              BSTARC = FSTARC
              SBSTARC = SNDARC
          MAIS
             TMAX = Max1
             Max1 = MAX2
             MAX2 = TMAX
```

```
            BSTARC = SNDARC
            SBSTARC = FSTARC
          END IF
          IF (SNDARC.LT.LSTARC) THEN
            TRDARC = + 1 SNDARC
            MAX3 = ISMALL
            FAZER 43 CURARC = TRDARC, LSTARC
               CURCOL = END (CURARC)
               TMAX = COST (CURARC) -PCOL (CURCOL)
               IF (TMAX.GT.MAX2) THEN
                IF (TMAX.GT.MAX1) THEN
                      SBSTARC = BSTARC
                      BSTARC = CURARC
                      MAX3 = MAX2
                      MAX2 = Max1
                      Max1 = TMAX
                 MAIS
                      SBSTARC = CURARC
                      MAX3 = MAX2
                      MAX2 = TMAX
                 END IF
                MAIS
                  IF (MAX3.LT.TMAX) MAX3 = TMAX
                END IF
43 CONTINUAR
          END IF

        BEST_ARC (ROW) = BSTARC
        BSTCOL = END (BSTARC)
        SECD_ARC (ROW) = SBSTARC
        TRDVAL (ROW) = MAX3

70 CONTINUAR


APOSTAS C linha para BSTCOL AUMENTAR SEU PREÇO, E é atribuído
C TO BSTCOL, enquanto qualquer ROW ATRIBUÍDO BSTCOL TORNA não atribuídos
```

```
          PCOL (BSTCOL) = PCOL (BSTCOL) + Max1-MAX2 + INCR
          PROW (ROW) = MAX2-INCR
          OLDROW = ROW_ASSIGNED_TO (BSTCOL)
          ROW_ASSIGNED_TO (BSTCOL) = ROW
          IF (OLDROW.GT.0) THEN
            NONEWLIST = + 1 NONEWLIST
            LIST (NONEWLIST) = OLDROW
          END IF

103 CONTINUAR

C ************* fim de um ciclo LEILÃO PARA A FRENTE ***************

C OPTIONALLY coletar estatísticas

X Médio = (ciclos * MÉDIA + NOLIST) / (CICLOS + 1)
CICLOS X = CICLOS + 1

C verificar se a mudança para MODIFICADO leilão reverso
C DEVE SER FEITO (IF NONEWLIST igual a zero).
C também aumentam a LICITAÇÃO MINIMAL INCREMENTO até um máximo
C VALOR DO EPSILON (esta é a característica adaptativa)

         INCR = INCR * INCRFACTOR
         IF (INCR.GT.EPSILON) INCR = EPSILON
         IF (NONEWLIST.GT.THRESH) THEN
             NOLIST = NONEWLIST
             INIT = .TRUE.
             IR PARA 17
         END IF

       MAIS

C ********************************************** ***************
C
C partida de avanço LEILÃO CICLO
```

```
C
C ********************************************** ***************

15 CONTINUAR

C REINICIAR CONTAGEM DO PRÓXIMO lista de linhas não atribuídos

        NONEWLIST = 0

C percorrer a lista atual de linhas não atribuídos

        FAZER 100 I = 1, NOLIST
          ROW = LIST (I)
          FSTARC = FOUT (ROW)
          LSTARC = FOUT (ROW + 1) -1
          FSTCOL = END (FSTARC)

C PRIMEIRA CUIDAR DA caso excepcional ROW tem apenas um ARC

           IF (FSTARC.EQ.LSTARC) THEN
             PCOL (FSTCOL) = PCOL (FSTCOL) + LARGEINCR
            PROW (ROW) = COST (FSTARC) -PCOL (FSTCOL)
             OLDROW = ROW_ASSIGNED_TO (FSTCOL)
             ROW_ASSIGNED_TO (FSTCOL) = ROW
             IF (OLDROW.GT.0) THEN
               NONEWLIST = + 1 NONEWLIST
               LIST (NONEWLIST) = OLDROW
             END IF
             Ir para 100
           END IF

C tomar em seguida CUIDADOS DO CASO regular, onde linha tem ARCS MÚLTIPLAS

           SNDARC = + 1 FSTARC
           SNDCOL = END (SNDARC)
           Max1 = COST (FSTARC) -PCOL (FSTCOL)
           MAX2 = COST (SNDARC) -PCOL (SNDCOL)
```

```
          IF (MAX1.GE.MAX2) THEN
             BSTCOL = FSTCOL
          MAIS
             TMAX = Max1
             Max1 = MAX2
             MAX2 = TMAX
             BSTCOL = SNDCOL
          END IF
          IF (SNDARC.LT.LSTARC) THEN
            TRDARC = + 1 SNDARC
            DO 40 CURARC = TRDARC, LSTARC
               CURCOL = END (CURARC)
               TMAX = COST (CURARC) -PCOL (CURCOL)
               IF (TMAX.GT.MAX2) THEN
                 IF (TMAX.GT.MAX1) THEN
                      MAX2 = Max1
                      Max1 = TMAX
                      BSTCOL = CURCOL
                 MAIS
                      MAX2 = TMAX
                 END IF
               END IF
40 CONTINUAR
           END IF


APOSTAS C linha para BSTCOL AUMENTAR SEU PREÇO, E é atribuído
C TO BSTCOL, enquanto qualquer ROW ATRIBUÍDO BSTCOL TORNA não atribuídos

          PCOL (BSTCOL) = PCOL (BSTCOL) + Max1-MAX2 + INCR
          PROW (ROW) = MAX2-INCR
          OLDROW = ROW_ASSIGNED_TO (BSTCOL)
          ROW_ASSIGNED_TO (BSTCOL) = ROW
          IF (OLDROW.GT.0) THEN
            NONEWLIST = + 1 NONEWLIST
            LIST (NONEWLIST) = OLDROW
          END IF
```

```
100 CONTINUAR

C ************* fim de um ciclo LEILÃO PARA A FRENTE ***************

C OPTIONALLY coletar estatísticas

X Médio = (ciclos * MÉDIA + NOLIST) / (CICLOS + 1)
CICLOS X = CICLOS + 1

C verificar se a mudança para MODIFICADO leilão reverso
C DEVE SER FEITO (IF NONEWLIST HÁ MAIS do que o limite).
C também aumentam a LICITAÇÃO MINIMAL INCREMENTO até um máximo
C VALOR DO EPSILON (esta é a característica adaptativa)

         INCR = INCR * INCRFACTOR
         IF (INCR.GT.EPSILON) INCR = EPSILON
         IF (NONEWLIST.GT.THRESH) THEN
             NOLIST = NONEWLIST
             IR PARA 15
         END IF

      END IF

C ********************************************** **************
C
C INÍCIO DE leilão reverso MODIFICADO
C
C ********************************************** **************

        IF (EPSILON.GT.1) GOTO 400
        IF (N.EQ.M) GOTO 400

        INCR = EPSILON


C COMPUTE LAMBDA que é o mínimo coluna de preços sobre todos
```

```
C COLUNAS atribuído, e compilar a lista de colunas não atribuídos

       LAMBDA = ILARGE
       RNOLIST = 0
       FAZER 310 J = 1, N
         ROW = ROW_ASSIGNED_TO (J)
         IF (ROW.GT.0) THEN
           COL_ASSIGNED_TO (ROW) = J
           IF (LAMBDA.GT.PCOL (J)) LAMBDA = PCOL (J)
         MAIS
           RNOLIST = + 1 RNOLIST
           REV_LIST (RNOLIST) = J
         END IF
310 CONTINUAR


       IF (RNOLIST.NE.NM) THEN
          PRINT *, 'número de linhas não definidos é errado "
          PAUSA
          PARE
       END IF


C início de um novo ciclo reverso MODIFICADO LEILÃO


315 CONTINUAR

C REINICIAR CONTAGEM DO PRÓXIMO lista de colunas não atribuídos

        RNONEWLIST = 0

C percorrer a lista atual de COLUNAS não atribuídos

        FAZER 340 J = 1, RNOLIST
          COLUNA = REV_LIST (J)
          IF (PCOL (coluna) .LE.LAMBDA) ir para 340
```

```
          CURARC = FIN (coluna)
          NXTARC = NXTIN (CURARC)
          Currow = START (CURARC)

C PRIMEIRA CUIDAR DA caso excepcional ROW tem apenas um ARC

          IF (NXTARC.EQ.0) THEN
            Max1 = COST (CURARC) -PROW (Currow)
             IF (LAMBDA.GE.MAX1-INCR) THEN
                PCOL (coluna) = LAMBDA
                IR PARA 340
            END IF
            PCOL (coluna) = LAMBDA
             PROW (Currow) = COST (CURARC) -LAMBDA
             OLDCOL = COL_ASSIGNED_TO (Currow)
             COL_ASSIGNED_TO (Currow) = COLUNA
            IF (PCOL (OLDCOL) .GT.LAMBDA) THEN
              RNONEWLIST = + 1 RNONEWLIST
              REV_LIST (RNONEWLIST) = OLDCOL
            END IF
             IR PARA 340
          END IF

C tomar em seguida CUIDADOS DO CASO regular, onde linha tem ARCS MÚLTIPLAS

          NXTROW = START (NXTARC)
          Max1 = COST (CURARC) -PROW (Currow)
          MAX2 = COST (NXTARC) -PROW (NXTROW)
          IF (MAX1.GE.MAX2) THEN
             BSTROW = Currow
          MAIS
             TMAX = Max1
             Max1 = MAX2
             MAX2 = TMAX
             BSTROW = NXTROW
          END IF
```

```
          CURARC = NXTIN (NXTARC)

330 IF (CURARC.GT.0) THEN
            Currow = START (CURARC)
            TMAX = COST (CURARC) -PROW (Currow)
            IF (TMAX.GT.MAX2) THEN
              IF (TMAX.GT.MAX1) THEN
                   MAX2 = Max1
                   Max1 = TMAX
                   BSTROW = Currow
              MAIS
                   MAX2 = TMAX
              END IF
            END IF
            CURARC = NXTIN (CURARC)
            IR PARA 330
          END IF


APOSTAS C COLUNA BSTROW AUMENTAR SEU PREÇO, E é atribuído
C TO BSTROW, enquanto qualquer COLUNA ATRIBUÍDO BSTROW TORNA não atribuídos


          IF (LAMBDA.GE.MAX1-INCR) THEN
             PCOL (coluna) = LAMBDA
             IR PARA 340
          END IF
          DELTA = Max1-MAX2 + INCR
          IF (DELTA.GT.MAX1-LAMBDA) DELTA = Max1-LAMBDA
          PROW (BSTROW) = PROW (BSTROW) + DELTA
          PCOL (coluna) = Max1-DELTA
          OLDCOL = COL_ASSIGNED_TO (BSTROW)
          COL_ASSIGNED_TO (BSTROW) = COLUNA
          IF (PCOL (OLDCOL) .GT.LAMBDA) THEN
            RNONEWLIST = + 1 RNONEWLIST
            REV_LIST (RNONEWLIST) = OLDCOL
          END IF
```

```
340 CONTINUAR

C ************* fim de um ciclo de leilão reverso MODIFICADO ***************

C OPTIONALLY coletar estatísticas

X Médio = (ciclos * MÉDIA + RNOLIST) / (CICLOS + 1)
CICLOS X = CICLOS + 1

C Verifique se ainda existem COLUNAS ELEGÍVEIS não atribuído 'muitos', isto é,
C Se o número de colunas não atribuído ELEGÍVEIS É MAIOR DO QUE
C DO THRESH PARAMETER. NÃO SE, REPLACE lista atual com a lista NEW,
C e ir para outro ciclo. Caso contrário, se EPSILON> 1, REDUZIR EPSILON,
C Redefinir a atribuição ao vazio e REINICIAR LEILÃO;
C IF EPSILON = 1 finalizar.


        IF (RNONEWLIST.GT.THRESH) THEN
          RNOLIST = RNONEWLIST
          IR PARA 315
        END IF


C ********************************************** **************
C
C FIM DE subproblema (escalonamento FASE)
C
C ********************************************** **************

400 CONTINUAR

C ****** SE EPSILON É uma RESCINDIR ******

       IF (EPSILON.EQ.1) THEN
          RETURN
       MAIS
```

```
C MAIS REDUZIR EPSILON e atualizar os parâmetros para a FASE DE ESCALA PRÓXIMO

          NUMPHASES + 1 = NUMPHASES
          EPSILON = INT (EPSILON / FACTOR)
          IF (EPSILON.GT.INCR) EPSILON = INT (EPSILON / FACTOR)
          IF ((EPSILON.LT.1) .OR. (EPSILON.LT.ENDEPS)) EPSILON = 1
          IF (STARTINCR.LT.EPSILON) STARTINCR = FACTOR * STARTINCR
          THRESH = INT (THRESH / FACTOR)


X PRINT *, '*** FIM DE UMA FASE DE ESCALA; NEW EPSILON = ', EPSILON

          IR PARA 12
        END IF


        FIM


      SUBROUTINE SETRAN (ISEED)
      IMPLICIT real * 8 (AH, OZ), Integer * 4 (IN)
C ********************************************** *********************
C PORTABLE congruential (uniforme) RANDOM gerador de números:
NEXT_VALUE C = [(7 ** 5) * PREVIOUS_VALUE] MODULO [(2 ** 31) -1]
C
C Este gerador é composto por duas ROTINAS:
C (1) SETRAN - inicializa constantes e SEED
C (2) RRAN - gera um número aleatório REAIS
C
C O GERADOR DE RODA uma máquina com pelo menos 32 bits de precisão.
C DA SEED (ISEED) deve estar no intervalo (1, (2 ** 31 -1)).
C ********************************************** *********************
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
      IF (ISEED.LT.1) PARADA 77
      MULT = 16807
      MODUL = 2147483647
```

```
      I15 = 2 ** 15
      I16 = 2 ** 16
      JRAN = ISEED
      RETURN
      FIM
C
      RAN função real ()
      IMPLICIT real * 4 (AH, OZ), Integer * 4 (IN)
C ********************************************** **********************
C RAN gera um número aleatório real entre 0 e 1
C ********************************************** **********************
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
      Ixhi = JRAN / I16
      IXLO = JRAN-Ixhi * I16
      IXALO = IXLO * MULT
      LEFTLO = IXALO / I16
      IXAHI = Ixhi * MULT
      IFULHI = IXAHI + LEFTLO
      IRTLO = IXALO-LEFTLO * I16
      IOVER = IFULHI / I15
      IRTHI = IFULHI-IOVER * I15
      JRAN = ((IRTLO-MODUL) + IRTHI * I16) + IOVER
      IF (JRAN.LT.0) JRAN = JRAN + MODUL
      RAN = FLOAT (JRAN) / FLOAT (MODUL)
      RETURN
      FIM
```

\ Subsect {LEILÃO-FR} Esse código implementa o leilão reverso / combinado para a frente algoritmo com $ \ e $ scaling para o problema de atribuição simétrica; cf. \ Seção 4.2 do livro de otimização de rede do autor.
Para um bom desempenho, ele freqüentemente não é imprescindível a utilização de $ \ e $ scaling em este código. Portanto, é recomendável que o código ser julgado sem $ \ E $ scaling, definindo a partida $ \ e $ 1.

```
C ********************************************** *****************
C
C programa de amostra CHAMADA PARA COMBINADO frente / para trás
C ALGORITHM LEILÃO
C
C ESTE MOTORISTA cria um problema SIMÉTRICA CESSÃO
C com igual número de linhas e colunas,
C e chama o SUBROUTINE AUCTION_FR para encontrar um
C atribuição do valor máximo.
C
C ********************************************** *****************


        PARÂMETROS (maxNodes = 10000, MAXARCS = 100000)

        NONE IMPLICIT
        INTEGER N, NA, A, IA, ILARGE, BEGEPS, ENDEPS, CICLOS
        INTEGER NUMPHASES, STARTINCR
        INTEGER I, J, ARC, NOASS, ICOST, ABSCOST, CURARC
        INTEGER CURCOL, FSTARC, LSTARC, MINCOST, MAXCOST, MCOST
        INTEGER COL_ASSIGNED_TO (maxNodes), ROW_ASSIGNED_TO (maxNodes)
        INTEGER FOUT (maxNodes), FIN (maxNodes), NXTIN (MAXARCS)
        CUSTO INTEIRO (MAXARCS), START (MAXARCS), END (MAXARCS)
        INTEGER PCOL (maxNodes), PROW (maxNodes)
        INTEGER PRDARC (maxNodes)
        Real * 8 FACTOR, TT1, TT2, TCOST, AVERAGE
        COMUM / ARRAYC / CUSTO / matrizes / START / ARRAYE / END
        COMUM / ARRAYFO / FOUT / ARRAYFI / FIN / ARRAYNI / NXTIN
        COMUM / ARRAYRA / ROW_ASSIGNED_TO / ARRAYCA / COL_ASSIGNED_TO
        COMUM / ARRAYPC / PCOL / ARRAYPR / PROW
        Common / BK1 / N, A, ILARGE / BK2 / ciclos, Médio, NUMPHASES

C ********************************************** ***************
C
C problema de geração de código começa AQUI
C o usuário pode substituir este código com um código que lê
C HIS / seu problema de um arquivo
```

```
C
C ********************************************** ***************

C Este código inclui UM UNIFORME Random Number Generator
C que retorna um valor na (0,1)

C INITIALIZE Gerador Aleatório

      INTEGER MULT, MODUL, I15, I16, JRAN, ISEED
      RAN REAIS
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN

      ISEED = 13502460
      CHAME SETRAN (ISEED)

      PRINT * ', gerando problema de atribuição de SIMÉTRICA "
      PRINT *, "****************************************** '

C **** LER O número de linhas N & o número de arcos A ****

      PRINT *, 'Digite o número de linhas (e colunas)'
      LEIA *, N
5 PRINT *, 'Digite o número de arcos por ROW (1>) "
      LEIA *, NA
      IF (NA.LT.2) GOTO 5
      PRINT *, 'Digite o mínimo eo custo máximo'
      LEIA *, MINCOST, MAXCOST

C o número de arcos é n * NA

      A = N * NA

C Os arcos incidente a linha i SÃO FOUT (I) PARA FOUT (I + 1) -1
C Além disso, para VIABILIDADE EACH ROW está diretamente ligado
C com a coluna correspondente

      FAZER 20 I = 1, N
```

```
        FOUT (I) = 1 + (i-1) * NA
20 CONTINUAR
      FOUT (N + 1) = A + 1

      FAZER 22 I = 1, N
        FAZER 23 ARC = FOUT (I), FOUT (I + 1) -1
          START (ARC) = I
23 CONTINUAR
22 CONTINUAR

C gerar a END (ARC) eo custo (ARC), que são os COLUNA
C eo coeficiente custo associado a ARC

      FAZER 25 ARC = 1, A
        END (ARC) = 1 + RAN () * N
        IF ((END (ARC) .GT.N) .OR. (END (ARC) .LT.1)) THEN
          PRINT *, 'Erro no problema de geração de'
          PAUSA
          PARE
        END IF
        COST (ARC) = MINCOST + RAN () * (MAXCOST-MINCOST)
25 CONTINUAR

C MODIFICAR O final da última ARC OUT de cada linha de viabilidade
C e defina seu CUSTO PARA -MAXCOST

      FAZER 30 I = 1, N
        END (FOUT (I + 1 -1)) = I
        CUSTO (FOUT (I + 1) -1) = - MAXCOST
30 CONTINUAR

C construir a FIN () e NXTIN () ARRAYS

      DO 32 J = 1, N
        FIN (J) = 0
        PRDARC (J) = 0
32 CONTINUAR
```

```
      FAZER 33 ARC = 1, A
        NXTIN (ARC) = 0
        J = END (ARC)
        IF (FIN (J) .NE.0) THEN
          NXTIN (PRDARC (J)) = ARC
        MAIS
          FIN (J) = ARC
        END IF
        PRDARC (J) = ARC
33 CONTINUAR


C ********************************************** ***************
C
C PROBLEMA geração de código TERMINA AQUI
C
C ********************************************** ***************


C ESCALA O CUSTO PARA TRABALHAR COM INTEIRO EPSILON

        MAXCOST = 0
        FAZER 35 IA = 1, A
           ABSCOST = IABS (COST (IA))
           IF (ABSCOST.GT.MAXCOST) MAXCOST = ABSCOST
           COST (IA) = CUSTO (IA) * (N + 1)
35 CONTINUAR

C *** ILARGE é um inteiro MUITO GRANDE PARA SUA MÁQUINA ***
C OS CUSTOS escalado deve ser significativamente menor do
C ILARGE E significativamente maior do que -ILARGE


        ILARGE = 2000000000
```

```
        IF (MAXCOST.GT.INT (ILARGE / (N + 1))) THEN
          PRINT *, 'A faixa de custo é muito grande para INTEGER ARITHMETIC'
          PAUSA
          PARE
        END IF

        MAXCOST MAXCOST = * (N + 1)


C os seguintes parâmetros BEGEPS, FACTOR, ENDEPS E STARTINCR
C são passados para o LEILÃO algoritmo. Valores entre
C (A) MAXCOST / 2 e 1 para BEGEPS
C (B) 4 e 10 para FACTOR
C (C) N e 1 para ENDEPS
C (D) 1 AND BEGEPS PARA STARTINCR
C têm funcionado bem para PROBLEMAS esparsos de grande porte.
C PARA PROBLEMAS DENSOS RECOMENDA-SE QUE
C BEGEPS ser ajustado para um valor menor,
ENDEPS C ser definido como 1,
C STARTINCR ser definido como 1.

C Geralmente, o atacante COMBINADO / REVERSE ALGORITHM
C funciona melhor com o menor valor de BEGEPS DO QUE A
C algoritmo Forward.
C Vale a pena tentar BEGEPS = 1, CASO EM QUE HÁ
C só UMA ESCALA DE FASE (ou seja, nenhum E-escala que é utilizada).

        PRINT *, "****************************************** '
        PRINT *, "custo máximo é ', MAXCOST
        PRINT *, 'Digite o EPSILON PARTIDA'
        LER *, BEGEPS
        IF (BEGEPS.LT.1) BEGEPS = 1
        PRINT *, 'Insira o fator EPSILON REDUÇÃO'
        LER *, FACTOR
        ENDEPS = N / 10
        IF (ENDEPS.LT.1) ENDEPS = 1
        IF (ENDEPS.GT.BEGEPS) ENDEPS = BEGEPS
```

```
        STARTINCR = 1
        IF (STARTINCR.LT.1) STARTINCR = 1

        PRINT *, "****************************************** '
        PRINT *, 'começar EPSILON =', BEGEPS
        PRINT *, 'Epsilon redução de fatores =', FACTOR
        PRINT *, «limiar EPSILON ANTES ela é definida como 1 = ', ENDEPS
        PRINT *, "COMEÇAR MIN DE LICITAÇÃO INCREMENTO = ', STARTINCR
        PRINT *, "CHAMANDO A FRENTE / leilão reverso '
        PRINT *, "****************************************** '

C GET hora de início para o Mac II

      TT1 = longa (362) /60.0

      CHAMADA AUCTION_FR (BEGEPS, fator, ENDEPS, STARTINCR)

C GET TÉRMINO TEMPO PARA O MAC II

      TT2 = longa (362) /60.0 - TT1
      PRINT *, "acabado --- tempo de CPU TOTAL ', TT2,' s '
      PRINT *, "****************************************** '

C *** Mostrar resultados ***

X Write (9,2010) CICLOS
X2010 FORMATO ('NO LEILÃO DE CICLOS', I7)
X WRITE (9,2020) MÉDIA
X2020 FORMATO ("número médio de propostas por ciclo", F9.3)
       Write (9,2030) NUMPHASES
2030 FORMATO ('NO DE EPSILON subproblemas RESOLVIDO =', I7)

C VERIFICAÇÃO Optimality & calcular o custo

        TCOST = 0
        DO 40 I = 1, N
          J = COL_ASSIGNED_TO (I)
```

```
          IF (J.EQ.0) THEN
              PRINT *, 'ROW', I, 'não é atribuído "
          END IF
          IF (ROW_ASSIGNED_TO (J) .NE.I) THEN
              PRINT *, 'Atribuição confusão: ROW', I, 'COLUMN', J
          END IF
          FSTARC = FOUT (I)
          LSTARC FOUT = (I + 1) -1
          MCOST = -ILARGE
          FAZER 45 ARC = FSTARC, LSTARC
            CURCOL = END (ARC)
            IF (PROW (I) + PCOL (CURCOL) .LT.COST (ARC -1)) THEN
              PRINT *, '1-CS VIOLADOS AT ARC ", ARC
            END IF
            IF (CURCOL.EQ.J) THEN
              IF (MCOST.LT.COST (ARC)) THEN
                 MCOST = CUSTO (ARC)
              END IF
            END IF
45 CONTINUAR
          TCOST = TCOST MCOST + / (N + 1)
          IF (PROW (I) + PCOL (J) .NE.MCOST) THEN
            PRINT *, '1-CS VIOLADA na linha', I
          END IF
40 CONTINUAR

        FAZER 50 J = 1, N
           IF (ROW_ASSIGNED_TO (J) .EQ.0) THEN
              PRINT *, 'COLUMN', J, "não é atribuído"
           END IF
50 CONTINUAR

        ESCREVA (9,2100) TCOST
2100 FORMATO ('atribuição de custo =', F18.2)
        PRINT *, 'programa terminou; <CR> PARA SAIR '
        PAUSA
        FIM
```

```
C ********************************************** *****************
C
C frente / para trás CODE LEILÃO PARA N por N problemas de atribuição
C
C ESCRITO POR DIMITRI P. Bertsekas
C
C dezembro 1990
C
C ESTE CÓDIGO implementa a frente / para trás ALGORITHM LEILÃO
C COM E-escala para SIMÉTRICA N por N problemas de atribuição.
C Ele resolve uma seqüência de subproblemas e diminui
C EPSILON por um fator constante entre subproblemas.
C ESTE versão corresponda a um MODO Gauss-Seidel
C e resolve EPSILON subproblemas inexata.
C
C THE CODE é uma versão melhorada de um antigo (SEPT. 1985)
C CODE LEILÃO COM ESCALA E-ESCRITO POR DIMITRI P. Bertsekas
C
C DO CÓDIGO trata o problema como um problema de maximização.
C para resolver um problema de minimização, inverta o sinal da
C ARC CUSTOS antes de chamar LEILÃO, E RECUO DE NOVO
C O SINAL DO custo ótimo na volta do LEILÃO.
C Este código permite ARCS múltipla entre uma linha e uma coluna.
C
C Esta versão do ALGORITHM LEILÃO é adaptativa. AQUI NO MÍNIMO
INCREMENTO DE LICITAÇÃO C (armazenado no INCR variável) pode ser menor que
C EPSILON. PARA CADA subproblema, INCR COMEÇA no valor do parâmetro
C STARTINCR (que é passado para a rotina LEILÃO), estando aumentada
C por um fator de 2 no final de cada ciclo, até um valor máximo de
C EPSILON. ESTA FUNÇÃO adaptativo é particularmente eficaz para
C PROBLEMAS densa. TI pode ser derrotado por SELEÇÃO STARTINCR = BEGEPS
C
C Este código pode falhar devido a integer overflow IF o número de nós
```

```
C ou o intervalo COST (ou ambos) são grandes. Para corrigir esta situação,
C OS PREÇOS E outras variáveis relacionadas (Max1, max2, TMAX etc.) devem
C DECLARAR AS REAIS precisão dupla.
C ********************************************** ******************
C
C o usuário deve fornecer os seguintes dados problema no FRENTE DA ESTRELA FORMATO
C (ou seja, todos os arcos da linha SAME são numerados consecutivamente):
C N = número de linhas (igual número de colunas)
C A = número de arcos
C FOUT (ROW) = Primeiro ARC QUE SAI DE ROW
C FIN (COL) = PRIMEIRO ARC VINDO EM COL
C NXTIN (ARC) = Próxima ARC incidente à COLUNA MESMO AS ARC
C COST (ARC) = CUSTO DE ARC
C START (ARC) = ROW correspondente ao arco
C END (ARC) = COLUNA correspondente ao arco
C
C E os seguintes parâmetros para o algoritmo LEILÃO:
C BEGEPS = Valor inicial EPSILON (não deve ser menor que 1)
C ENDEPS = valor final da EPSILON ANTES ela é definida como 1
C FACTOR = fator pelo qual EPSILON é diminuída ENTRE subproblemas
C STARTINCR = O valor inicial do incremento LICITAÇÃO
ENDEPS C não devem exceder BEGEPS.
C fator deve ser superior a 1 (a menos que BEGEPS = 1).
C
C FOUT (.) É uma matriz de comprimento N.
C COST (.), END (.) São matrizes de COMPRIMENTO A.
C
C a solução é contidos na matriz ROW_ASSIGNED_TO (.) ONDE
C ROW_ASSIGNED_TO (COL) Dá o ROW ATRIBUÍDO COL.
C TAMBÉM COL_ASSIGNED_TO (ROW) DÁ A COLUNA ATRIBUÍDO linha.
C
C Este algoritmo não verifica a inviabilidade do problema.
C TO garantir que o problema é viável o usuário pode ADD
Adicional C ARCS custo muito pequeno.
C
C ********************************************** *****************
C ALL problema de dados são integer
```

```
C ********************************************** *****************


        SUBROUTINE AUCTION_FR (BEGEPS, fator, ENDEPS, STARTINCR)

        PARÂMETROS (maxNodes = 10000, MAXARCS = 100000)

        NONE IMPLICIT
        INTEIRO A, K, N, I, J, M, CURARC, CURCOL, Currow
        INTEGER THRESH, INCR, STARTINCR, INCRFACTOR
        INTEGER NOLIST, RNOLIST, NONEWLIST, RNONEWLIST
        INTEGER linha, coluna BSTROW, BSTCOL
        INTEGER FSTARC, FSTCOL, NXTARC, NXTROW, LSTARC, SNDARC, SNDCOL
        INTEGER Max1, MAX2, TMAX, TMIN, TRDARC, EPSILON, BEGEPS, ENDEPS
        INTEGER ISMALL, ILARGE, LARGEINCR, CICLOS
        INTEGER NUMPHASES, OLDROW, OLDCOL
        INTEGER COL_ASSIGNED_TO (maxNodes), ROW_ASSIGNED_TO (maxNodes)
        INTEGER FOUT (maxNodes), FIN (maxNodes), NXTIN (MAXARCS)
       CUSTO INTEIRO (MAXARCS), START (MAXARCS), END (MAXARCS)
       LISTA INTEIRO (maxNodes), REV_LIST (maxNodes)
       INTEGER PCOL (maxNodes), PROW (maxNodes)
        Real * 8 AVERAGE, FACTOR
       READY_SWITCH LÓGICO, Switch
       COMUM / ARRAYC / CUSTO / matrizes / START / ARRAYE / END
        COMUM / ARRAYFO / FOUT / ARRAYFI / FIN / ARRAYNI / NXTIN
        COMUM / ARRAYRA / ROW_ASSIGNED_TO / ARRAYCA / COL_ASSIGNED_TO
        COMUM / ARRAYPC / PCOL / ARRAYPR / PROW
        Common / BK1 / N, A, ILARGE / BK2 / ciclos, Médio, NUMPHASES

C ********************************************** **************

Check C ******* VALIDADE DOS PARÂMETROS ******* PASSOU

        IF (BEGEPS.LT.1) THEN
          PRINT *, 'Valor inicial EPSILON é inferior a 1'
          PRINT *, 'EXECUÇÃO ABORTADO'
          PARE
```

```
        END IF
        IF (ENDEPS.GT.BEGEPS) THEN
          PRINT *, 'ENDEPS parâmetro é maior que o parâmetro BEGEPS'
          PRINT *, 'ENDEPS é definido no valor padrão de 1'
          ENDEPS = 1
        END IF
        IF ((FACTOR.LE.1) .E. (BEGEPS.GT.1)) THEN
          PRINT *, 'Epsilon redução de fatores não é maior que 1'
          PRINT *, 'EXECUÇÃO ABORTADO'
          PARE
        END IF
        IF (STARTINCR.LT.1) THEN
          PRINT *, 'MIN DE LICITAÇÃO incremento é menor que 1'
          PRINT *, 'STARTINCR é definido no valor padrão de 1'
          STARTINCR = 1
        END IF

C ******* INITIALIZATION *******

        EPSILON = BEGEPS
        ISMALL = -ILARGE
        LARGEINCR = INT (ILARGE / 10)
        THRESH = INT (0,2 * N)
        INCRFACTOR = 2
        IF (THRESH.GT.100) THRESH = 100
CICLOS x = 1
X MÉDIA = N
         NUMPHASES = 1

C INITIALIZE FORWARD / PARÂMETROS botão de reversão

         READY_SWITCH = .FALSE.
         CHAVE = .TRUE.


         FAZER 10 J = 1, N
           PCOL (J) = 0
```

```
10 CONTINUAR

         FOUT (N + 1) = A + 1

C ********************************************** ***************
C
C Esta implementação do leilão ALGORITHM opera em ciclos.
C cada ciclo LEILÃO PARA A FRENTE consiste em um lance em cada uma das linhas
C não atribuídos NO INÍCIO DO CICLO (essas linhas SÃO
C armazenado na lista de matriz (.)). AS que o ciclo progride NOVO
LINHAS C ficarem não atribuídas; Estes são armazenados em Lista (.)
C e apresentará um lance no próximo ciclo.
C leilão reverso ciclos são estruturado de forma similar.
C
C ********************************************** ***************
C
C INÍCIO subproblema (FASE DE ESCALA) W / NEW EPSILON
C
C ********************************************** ***************
C
12 CONTINUAR

C REINICIAR linha e coluna lista ATRIBUIÇÃO

        NOLIST = N
        FAZER 20 I = 1, N
           COL_ASSIGNED_TO (I) = 0
           LISTA (I) = I
20 CONTINUAR

        RNOLIST = N
        DO 22 J = 1, N
           ROW_ASSIGNED_TO (J) = 0
           REV_LIST (J) = J
22 CONTINUAR

        INCR = STARTINCR
```

```
        IF (INCR.GT.EPSILON) INCR = EPSILON
        IF (EPSILON.EQ.1) debulhar = 0

C ********************************************** ***************
C
C partida de avanço LEILÃO CICLO
C
C ********************************************** ***************

15 CONTINUAR

C REINICIAR CONTAGEM DO PRÓXIMO lista de linhas não atribuídos

        NONEWLIST = 0

C percorrer a lista atual de linhas não atribuídos

        FAZER 100 I = 1, NOLIST
          ROW = LIST (I)
          IF (COL_ASSIGNED_TO (ROW) .GT.0) ir para 100
          FSTARC = FOUT (ROW)
          LSTARC = FOUT (ROW + 1) -1
          FSTCOL = END (FSTARC)

C PRIMEIRA CUIDAR DA caso excepcional ROW tem apenas um ARC

          IF (FSTARC.EQ.LSTARC) THEN
            PCOL (FSTCOL) = PCOL (FSTCOL) + LARGEINCR
            PROW (ROW) = COST (FSTARC) -PCOL (FSTCOL)
            OLDROW = ROW_ASSIGNED_TO (FSTCOL)
            ROW_ASSIGNED_TO (FSTCOL) = ROW
            COL_ASSIGNED_TO (ROW) = FSTCOL
            IF (OLDROW.GT.0) THEN
              COL_ASSIGNED_TO (OLDROW) = 0
              NONEWLIST = + 1 NONEWLIST
              LIST (NONEWLIST) = OLDROW
            END IF
```

```
            Ir para 100
          END IF

C tomar em seguida CUIDADOS DO CASO regular, onde linha tem ARCS MÚLTIPLAS

          SNDARC = + 1 FSTARC
          SNDCOL = END (SNDARC)
          Max1 = COST (FSTARC) -PCOL (FSTCOL)
          MAX2 = COST (SNDARC) -PCOL (SNDCOL)
          IF (MAX1.GE.MAX2) THEN
             BSTCOL = FSTCOL
          MAIS
             TMAX = Max1
             Max1 = MAX2
             MAX2 = TMAX
             BSTCOL = SNDCOL
          END IF
          IF (SNDARC.LT.LSTARC) THEN
            TRDARC = + 1 SNDARC
            DO 40 CURARC = TRDARC, LSTARC
               CURCOL = END (CURARC)
               TMAX = COST (CURARC) -PCOL (CURCOL)
               IF (TMAX.GT.MAX2) THEN
                 IF (TMAX.GT.MAX1) THEN
                      MAX2 = Max1
                      Max1 = TMAX
                      BSTCOL = CURCOL
                 MAIS
                      MAX2 = TMAX
                 END IF
               END IF
40 CONTINUAR
          END IF


APOSTAS C linha para BSTCOL AUMENTAR SEU PREÇO, E é atribuído
C TO BSTCOL, enquanto qualquer ROW ATRIBUÍDO BSTCOL TORNA não atribuídos
```

```
          PCOL (BSTCOL) = PCOL (BSTCOL) + Max1-MAX2 + INCR
          PROW (ROW) = MAX2-INCR
          COL_ASSIGNED_TO (ROW) = BSTCOL
          OLDROW = ROW_ASSIGNED_TO (BSTCOL)
          ROW_ASSIGNED_TO (BSTCOL) = ROW
          IF (OLDROW.GT.0) THEN
            COL_ASSIGNED_TO (OLDROW) = 0
            NONEWLIST = + 1 NONEWLIST
            LIST (NONEWLIST) = OLDROW
          END IF

100 CONTINUAR

C ************* fim de um ciclo LEILÃO PARA A FRENTE ***************

C OPTIONALLY coletar estatísticas

X Médio = (ciclos * MÉDIA + NOLIST) / (CICLOS + 1)
CICLOS X = CICLOS + 1

C Verifique se ainda existem ROWS não atribuído 'muitos', isto é, se o
C número de linhas não atribuído É MAIOR DO QUE
C DO THRESH PARAMETER. NÃO SE, REPLACE lista atual com a lista NEW,
C e ir para outro ciclo. Caso contrário, se EPSILON> 1, REDUZIR EPSILON,
C Redefinir a atribuição ao vazio e REINICIAR LEILÃO;
C IF EPSILON = 1 finalizar.
C também aumentam a LICITAÇÃO MINIMAL INCREMENTO até um máximo
C VALOR DO EPSILON (esta é a característica adaptativa)


         INCR = INCR * INCRFACTOR
         IF (INCR.GT.EPSILON) INCR = EPSILON
         IF (NONEWLIST.GT.THRESH) THEN
             IF (CHAVE) THEN
               NOLIST = NONEWLIST
               IR PARA 115
```

```
            END IF
            IF (NONEWLIST.LT.RNOLIST) READY_SWITCH = .TRUE.
            IF ((NONEWLIST.EQ.NOLIST) .E. (READY_SWITCH)) THEN
              READY_SWITCH = .FALSE.
              IR PARA 115
            MAIS
              NOLIST = NONEWLIST
              IR PARA 15
            END IF
         MAIS
            IR PARA 300
         END IF


C ********************************************** ***************
C
C Início de ciclo reverso LEILÃO
C
C ********************************************** ***************

115 CONTINUAR

C REINICIAR CONTAGEM DO PRÓXIMO lista de colunas não atribuídos

         RNONEWLIST = 0

C percorrer a lista atual de COLUNAS não atribuídos

         FAZER 200 J = 1, RNOLIST
           COLUNA = REV_LIST (J)
           IF (ROW_ASSIGNED_TO (coluna) .GT.0) ir para 200
           CURARC = FIN (coluna)
           NXTARC = NXTIN (CURARC)
           Currow = START (CURARC)

C PRIMEIRA CUIDAR DA caso excepcional ROW tem apenas um ARC
```

```
          IF (NXTARC.EQ.0) THEN
             PROW (Currow) = PROW (Currow) + LARGEINCR
            PCOL (coluna) = COST (CURARC) -PROW (Currow)
             OLDCOL = COL_ASSIGNED_TO (Currow)
             COL_ASSIGNED_TO (Currow) = COLUNA
             ROW_ASSIGNED_TO (coluna) = Currow
             IF (OLDCOL.GT.0) THEN
               ROW_ASSIGNED_TO (OLDCOL) = 0
                RNONEWLIST = + 1 RNONEWLIST
                REV_LIST (RNONEWLIST) = OLDCOL
             END IF
             IR PARA 200
           END IF

C tomar em seguida CUIDADOS DO CASO regular, onde linha tem ARCS MÚLTIPLAS

          NXTROW = START (NXTARC)
          Max1 = COST (CURARC) -PROW (Currow)
          MAX2 = COST (NXTARC) -PROW (NXTROW)
          IF (MAX1.GE.MAX2) THEN
             BSTROW = Currow
          MAIS
             TMAX = Max1
             Max1 = MAX2
             MAX2 = TMAX
             BSTROW = NXTROW
          END IF

          CURARC = NXTIN (NXTARC)

130 IF (CURARC.GT.0) THEN
              Currow = START (CURARC)
              TMAX = COST (CURARC) -PROW (Currow)
              IF (TMAX.GT.MAX2) THEN
                IF (TMAX.GT.MAX1) THEN
                     MAX2 = Max1
                     Max1 = TMAX
```

```
                BSTROW = Currow
              MAIS
                MAX2 = TMAX
              END IF
            END IF
            CURARC = NXTIN (CURARC)
            IR PARA 130
          END IF


APOSTAS C COLUNA BSTROW AUMENTAR SEU PREÇO, E é atribuído
C TO BSTROW, enquanto qualquer COLUNA ATRIBUÍDO BSTROW TORNA não atribuídos


          PROW (BSTROW) = PROW (BSTROW) + Max1-MAX2 + INCR
          PCOL (coluna) = MAX2-INCR
          ROW_ASSIGNED_TO (coluna) = BSTROW
          OLDCOL = COL_ASSIGNED_TO (BSTROW)
          COL_ASSIGNED_TO (BSTROW) = COLUNA
          IF (OLDCOL.GT.0) THEN
            ROW_ASSIGNED_TO (OLDCOL) = 0
            RNONEWLIST = + 1 RNONEWLIST
            REV_LIST (RNONEWLIST) = OLDCOL
          END IF

200 CONTINUAR

C ************* fim de um ciclo leilão reverso ***************

C OPTIONALLY coletar estatísticas

X Médio = (ciclos * MÉDIA + RNOLIST) / (CICLOS + 1)
CICLOS X = CICLOS + 1

C Verifique se ainda existem COLUNAS não atribuído 'muitos', isto é, se o
C Número de colunas não atribuído É MAIOR DO QUE
C DO THRESH PARAMETER. NÃO SE, REPLACE lista atual com a lista NEW,
```

```
C e ir para outro ciclo. Caso contrário, se EPSILON> 1, REDUZIR EPSILON,
C Redefinir a atribuição ao vazio e REINICIAR LEILÃO;
C IF EPSILON = 1 finalizar.
C também aumentam a LICITAÇÃO MINIMAL INCREMENTO até um máximo
C VALOR DO EPSILON (esta é a característica adaptativa)


        INCR = INCR * INCRFACTOR
        IF (INCR.GT.EPSILON) INCR = EPSILON
        IF (RNONEWLIST.GT.THRESH) THEN
             IF (CHAVE) THEN
               CHAVE = .FALSE.
               RNOLIST = RNONEWLIST
               IR PARA 15
             END IF
             IF (RNONEWLIST.LT.NOLIST) READY_SWITCH = .TRUE.
             IF ((RNONEWLIST.EQ.RNOLIST) .E. (READY_SWITCH)) THEN
               READY_SWITCH = .FALSE.
               IR PARA 15
             MAIS
               RNOLIST = RNONEWLIST
               IR PARA 115
             END IF
        MAIS
            IR PARA 300
        END IF


C ********************************************** **************
C
C FIM DE subproblema (escalonamento FASE)
C
C ********************************************** **************


300 CONTINUAR
```

```
C ****** SE EPSILON É uma RESCINDIR ******

       IF (EPSILON.EQ.1) THEN
          RETURN
       MAIS

C MAIS REDUZIR EPSILON e atualizar os parâmetros para a FASE DE ESCALA PRÓXIMO

          NUMPHASES + 1 = NUMPHASES
          EPSILON = INT (EPSILON / FACTOR)
          IF (EPSILON.GT.INCR) EPSILON = INT (EPSILON / FACTOR)
          IF ((EPSILON.LT.1) .OR. (EPSILON.LT.ENDEPS)) EPSILON = 1
          IF (STARTINCR.LT.EPSILON) STARTINCR = FACTOR * STARTINCR
          THRESH = INT (THRESH / FACTOR)

X PRINT *, '*** FIM DE UMA FASE DE ESCALA; NEW EPSILON = ', EPSILON

          IR PARA 12
        END IF

        FIM




      SUBROUTINE SETRAN (ISEED)
      IMPLICIT real * 8 (AH, OZ), Integer * 4 (IN)
C ********************************************** *********************
C PORTABLE congruential (uniforme) RANDOM gerador de números:
NEXT_VALUE C = [(7 ** 5) * PREVIOUS_VALUE] MODULO [(2 ** 31) -1]
C
C Este gerador é composto por duas ROTINAS:
C (1) SETRAN - inicializa constantes e SEED
C (2) RRAN - gera um número aleatório REAIS
C
C O GERADOR DE RODA uma máquina com pelo menos 32 bits de precisão.
C DA SEED (ISEED) deve estar no intervalo (1, (2 ** 31 -1)).
```

```
C ********************************************** **********************
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
      IF (ISEED.LT.1) PARADA 77
      MULT = 16807
      MODUL = 2147483647
      I15 = 2 ** 15
      I16 = 2 ** 16
      JRAN = ISEED
      RETURN
      FIM
C
      RAN função real ()
      IMPLICIT real * 4 (AH, OZ), Integer * 4 (IN)
C ********************************************** **********************
C RAN gera um número aleatório real entre 0 e 1
C ********************************************** **********************
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
      Ixhi = JRAN / I16
      IXLO = JRAN-Ixhi * I16
      IXALO = IXLO * MULT
      LEFTLO = IXALO / I16
      IXAHI = Ixhi * MULT
      IFULHI = IXAHI + LEFTLO
      IRTLO = IXALO-LEFTLO * I16
      IOVER = IFULHI / I15
      IRTHI = IFULHI-IOVER * I15
      JRAN = ((IRTLO-MODUL) + IRTHI * I16) + IOVER
      IF (JRAN.LT.0) JRAN = JRAN + MODUL
      RAN = FLOAT (JRAN) / FLOAT (MODUL)
      RETURN
      FIM
```

\ Seita {NAUCTION_SP: Combinado
Leilão Naive e Sequential Shortest Path Código} Esse código implementa o

```
método caminho mais curto seqüencial para o problema de atribuição, precedido por um
extensiva a inicialização usando o algoritmo leilão ingênuo (cf. \ Seção
1.2.4 do livro de otimização de rede do autor).

C ********************************************** ******************
C
C COMBINADO LEILÃO NAIVE e seqüencial MENOR método do caminho
C PARA SIMÉTRICA nXn problemas de atribuição
C ENCONTRA uma alocação ótima
C
C ESCRITO POR DIMITRI P. Bertsekas
C
C ********************************************** ******************
C
C usuário deve fornecer os seguintes dados PROBLEMA
CN = número de linhas e número de colunas
CA = número de arcos
C FOUT (ROW) = 1° ARC QUE SAI DE ROW
C COST (ARC) = CUSTO DE ARC
C END (ARC) = COLUNA correspondente ao arco
C
(.) C FOUT É UM N - comprimento da matriz
C COST (.), END (.), NXTEND (.) São matrizes de comprimento A
C DO IDEAL CESSÃO está contido no ASSIGN Array (.) ONDE
C ASSIGN (COL) é a linha ATRIBUÍDO de Col
C Este algoritmo não verifica inviabilidade DO PROBLEMA
C TO garantir que o problema é viável o usuário pode ADD
C ARCS ADICIONAL custo muito elevado
C
C Este código permite ARCS múltipla entre uma linha e uma coluna.
C
C ********************************************** *****************
C ALL problema de dados são integer
C ********************************************** *****************
C
        Implícita INTEGER (AZ)
        INTEGER AUCTNUM, A, STARC, ENDARC, Currow, ARC, COUNT
```

```
        INTEGER HCOUNT, ROW, FSTARC, FSTCOL, SNDARC, SNDCOL, TMAX, BSTCOL
        INTEGER TMARG, TA, ROWPR, OLMARG, PRICE, TPRICE, CURCOL, MANEQUIM
        INTEGER FOUT (6000), PCOL (6000), PROW (6000), MARG (6000)
        INTEGER ASSIGN (6000), COLLAB (6000), ROWLAB (6000)
        LISTA inteiro (6000), SCAN (6000)
        CUSTO INTEIRO (70000), END (70000)
       MAXSET LÓGICO
        THRESH REAL, RAND, TT, temporizador, TCOST

C
C ********************************************** **************
C
C problema de geração de código começa AQUI
C o usuário pode substituir este código com um código que lê
C HIS / seu problema de um arquivo
C
C ********************************************** **************
C
C gerador aleatório STUFF
C
      INTEGER MULT, MODUL, I15, I16, JRAN, ISEED
      RAN REAIS
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
C UNIFORME gerador de números aleatórios que retorna um em valor (0,1)
C
C INITIALIZE Gerador Aleatório
C
      ISEED = 13502460
      CHAME SETRAN (ISEED)

      PRINT * ', gerando problema de atribuição de SIMÉTRICA "
      PRINT *, "****************************************** '

C **** LER O número de linhas N & o número de arcos A ****

      PRINT *, 'Digite o número de linhas (e colunas)'
      LEIA *, N
```

```
5 PRINT *, 'Digite o número de arcos por ROW (1>) "
      LEIA *, NA
      IF (NA.LT.2) GOTO 5
      PRINT *, 'Digite o mínimo eo custo máximo'
      LEIA *, MINCOST, MAXCOST

C o número de arcos é n * NA

      A = N * NA

C Os arcos incidente a linha i SÃO FOUT (I) PARA FOUT (I + 1) -1
C PARA VIABILIDADE EACH ROW está diretamente ligado
C com a coluna correspondente

      FAZER 20 I = 1, N
        FOUT (I) = 1 + (i-1) * NA
20 CONTINUAR

C **** gerar a END (IA) eo custo (IA), que são os COLUNA
C eo coeficiente custo associado a ARC IA ****

      FAZER 25 IA = 1, A
        END (IA) = 1 + RAN () * N
        IF ((END (IA) .GT.N) .OR. (END (IA) .LT.1)) THEN
          PRINT *, 'Erro no problema de geração de'
          PAUSA
          PARE
        END IF
        COST (IA) = MINCOST + RAN () * (MAXCOST-MINCOST)
25 CONTINUAR

C MODIFICAR O final da última ARC OUT de cada linha de viabilidade
C e defina seu CUSTO PARA -MAXCOST

      FAZER 30 I = 1, N
        END (FOUT (I + 1 -1)) = I
        CUSTO (FOUT (I + 1) -1) = - MAXCOST
```

```
30 CONTINUAR

C
C ********************************************** **************
C
C PROBLEMA geração de código TERMINA AQUI
C
C ********************************************** **************
C

C algoritmo principal COMEÇA AQUI

C
C ALGORITMO INTERNAMENTE trata o problema como um
C problema de minimização.
C Para resolver um problema de atribuição de maximização, SET
C o seguinte parâmetro MAXSET para true, defina-o mais para FALSE

        MAXSET = .TRUE.

        IF (MAXSET) THEN
          FAZER 35 IA = 1, A
            COST (IA) = - COST (IA)
35 CONTINUAR
        END IF


        PRINT *, 'NO de linhas (e colunas):', N
        PRINT *, 'NO DE ARCOS:', A
        PRINT *, "COMBINADO LEILÃO ingênuo e SEQ. SH. PATH MÉTODO '
        PRINT *, '******************************************** '
        TIMER = Long (362)

        ISIMPL = 0
        IPRC = 0
        ISMALL = -20000000
        ILARGE = -ISMALL
```

```
        COUNT = 0
        HCOUNT = 0
        FOUT (N + 1) = A + 1

C O SEGUINTE INITIALIZATION ATÉ AO INÍCIO DO
C CICLOS LEILÃO NAIVE foi sugerido por JONKER E Volgenant

C REINICIAR PREÇOS COLUNA

        DO 105 J = 1, N
           PCOL (J) = ILARGE
105 CONTINUAR

C encontrar o melhor linha para cada coluna

          FAZER 120 I = 1, N
            PROW (I) = 0
            ROWLAB (I) = 0
            FSTARC = FOUT (I)
            LSTARC FOUT = (I + 1) -1
            FAZER 110 ARC = FSTARC, LSTARC
              J = END (ARC)
              IF (COST (ARC) .LT.PCOL (J)) THEN
                PCOL (J) = CUSTO (ARC)
               ASSIGN (J) = I
              END IF
110 CONTINUAR
120 CONTINUAR

LINHAS C CESSÃO CONSTRUCT linha e DEASSIGN MULTIPLY ATRIBUIÇÃO

        FAZER 130 J = 1, N
           J0 = N-J + 1
          I = ASSIGN (J0)
          IF (ROWLAB (I) .NE.0) THEN
            ROWLAB (I) = - ABS (ROWLAB (I))
            ASSIGN (J0) = 0
```

```
           MAIS
             ROWLAB (I) = J0
           END IF
130 CONTINUAR

C CONSTRUIR lista candidata E PREÇOS coluna correta


         NEWNOL = 0
         FAZER 150 I = 1, N
           CURCOL = ROWLAB (I)
           IF (CURCOL.EQ.0) THEN
             NEWNOL = + 1 NEWNOL
             LISTA (NEWNOL) = I
             GOTO 150
           END IF
           IF (CURCOL.LT.0) THEN
             ROWLAB (I) = - CURCOL
           MAIS
             MAX2 = ISMALL
             FSTARC = FOUT (I)
             LSTARC FOUT = (I + 1) -1
             FAZER 140 ARC = FSTARC, LSTARC
               J = END (ARC)
               IF (J.NE.CURCOL) THEN
                 IF (PCOL (J) -Custo (ARC) .GT.MAX2) THEN
                    MAX2 = PCOL (J) -Custo (ARC)
                 END IF
               END IF
140 CONTINUAR
             PROW (I) = MAX2
             PCOL (CURCOL) = PCOL (CURCOL) + MAX2
           END IF
150 CONTINUAR


         IF (NEWNOL.EQ.0) IR PARA 2000
```

```
C ********************************************** ***********

C END da inicialização; FAÇA UM NÚMERO DE CICLOS LEILÃO
C que é definido abaixo na AUCTNUM informação com base no
C sparsity DO PROBLEMA

        IF (N * N.LT.10 * A) = 2 AUCTNUM
        IF ((N * N.GE.10 * A) .E. (N * N.LT.25 * A)) AUCTNUM = 3
        IF ((N * N.GE.25 * A) .E. (N * N.LT.50 * A)) AUCTNUM = 4
        IF ((N * N.GE.50 * A) .E. (N * N.LT.100 * A)) AUCTNUM = 5
        IF (N * N.GE.100 * A) AUCTNUM = 6

C para executar um PURE SEQÜENCIAL Shortest Path método set
C = 0 AUCTNUM

        IF (AUCTNUM.EQ.0) GOTO 1000

C INÍCIO DE UM NOVO CICLO

80 NOLIST = NEWNOL
        I = 1
              NEWNOL = 0

C retirar um fila para a iteração

100 ROW = LIST (I)
        I = I + 1

        FSTARC = FOUT (ROW)
        LSTARC = FOUT (ROW + 1) -1
        BSTCOL = END (FSTARC)
        IF (FSTARC.EQ.LSTARC) THEN
           OLDROW = ASSIGN (BSTCOL)
           ASSIGN (BSTCOL) = ROW
           ROWLAB (ROW) = BSTCOL
           PROW (ROW) = ISMALL
```

```
    PCOL (BSTCOL) = ISMALL + COST (FSTARC)
    IF (OLDROW.GT.0) THEN
      I = I-1
      LISTA (I) = OLDROW
      ROWLAB (OLDROW) = 0
    END IF
    ISIMPL = + 1 ISIMPL
    Ir para 100
 END IF
 SNDARC = + 1 FSTARC
 SNDCOL = END (SNDARC)
 Max1 = PCOL (BSTCOL) -Custo (FSTARC)
 MAX2 = PCOL (SNDCOL) -Custo (SNDARC)
 IF (MAX1.LT.MAX2) THEN
    TMAX = Max1
    Max1 = MAX2
    MAX2 = TMAX
    FSTCOL = BSTCOL
    BSTCOL = SNDCOL
    SNDCOL = FSTCOL
 END IF
 IF (SNDARC.LT.LSTARC) THEN
  TRDARC = + 1 SNDARC
  FAZER 108 ARC = TRDARC, LSTARC
    CURCOL = END (ARC)
    TMAX = PCOL (CURCOL) -Custo (ARC)
    IF (TMAX.GT.MAX2) THEN
      IF (TMAX.GT.MAX1) THEN
         MAX2 = Max1
         Max1 = TMAX
         SNDCOL = BSTCOL
         BSTCOL = CURCOL
      MAIS
         MAX2 = TMAX
         SNDCOL = CURCOL
      END IF
    END IF
```

```
108 CONTINUAR
        END IF
        PROW (ROW) = MAX2
        OLDROW = ASSIGN (BSTCOL)
        INCR = Max1-MAX2
        IF (INCR.GT.0) THEN
          PCOL (BSTCOL) = PCOL (BSTCOL) -INCR
          ASSIGN (BSTCOL) = ROW
          ROWLAB (ROW) = BSTCOL
          IF (OLDROW.GT.0) THEN
              I = I-1
              LISTA (I) = OLDROW
              ROWLAB (OLDROW) = 0
              ISIMPL = + 1 ISIMPL
             GOTO 100
          END IF
        MAIS
          IF (OLDROW.GT.0) THEN
            BSTCOL = SNDCOL
            OLDROW = ASSIGN (BSTCOL)
          END IF
          IF (OLDROW.EQ.0) THEN
            ASSIGN (BSTCOL) = ROW
            ROWLAB (ROW) = BSTCOL
          MAIS
            NEWNOL = + 1 NEWNOL
            LIST (NEWNOL) = ROW
          END IF
        END IF
        ISIMPL = + 1 ISIMPL
        IF (I.LE.NOLIST) GOTO 100

C fim de um ciclo LEILÃO NAIVE

        COUNT = COUNT + 1

        IF (NEWNOL.EQ.0) THEN
```

```
          NASSIH = NEWNOL
          GOTO 2000
        END IF
        IF (COUNT.LT.AUCTNUM) GOTO 80



C ********************************************** ***************
C
C END ******* DO LEILÃO NAIVE PARTE *******
C
C
C ******* INÍCIO DE SEQ. SH. PATH MÉTODO *******
C
C ********************************************** ***************



1000 NASSIH = NEWNOL
X = 0 IHPRC
1020 FAZER 180 I = 1, NEWNOL
           SCAN (I) = LIST (I)
180 CONTINUAR
        DO 190 J = 1, N
           MARG (J) = 1
190 CONTINUAR
        NOSCAN = NEWNOL
        IL1 = 1
        IL2 = NOSCAN
1030 DO 1060 I = IL1, IL2
           Currow = SCAN (I)
           ROWPR = PROW (Currow)
           FSTARC = FOUT (Currow)
           LSTARC = FOUT (Currow + 1) -1
           FAZER 1050 ARC = FSTARC, LSTARC
                CURCOL = END (ARC)
                OLMARG = MARG (CURCOL)
```

```
                IF (OLMARG.EQ.0) IR PARA 1050
                TMARG = PCOL (CURCOL) -ROWPR-COST (ARC)
            IF (TMARG.EQ.0) THEN
              IF (ASSIGN (CURCOL) .EQ.0) THEN
                IR PARA 1500

C Faça AUGMENTATION

              MAIS
                MARG (CURCOL) = 0
                NOSCAN = NOSCAN + 1
                SCAN (NOSCAN) = ASSIGN (CURCOL)
                COLLAB (CURCOL) = Currow
              END IF
            MAIS
              IF ((OLMARG.GT.0) .OR. (TMARG.GT.OLMARG)) THEN
                MARG (CURCOL) = TMARG
                COLLAB (CURCOL) = Currow
              END IF
            END IF
1050 CONTINUAR
1060 CONTINUAR

C atual de digitalização completa das fases; Cheque de mais linhas para verificar

          IF (IL2.LT.NOSCAN) THEN
             IL1 IL2 + 1 =
             IL2 = NOSCAN
             IR PARA 1030
          END IF


C Preço Variação


          IPRC = IPRC + 1
X = IHPRC IHPRC + 1
```

```
        INCR = ISMALL
        FAZER 200 J = 1, N
           TMARG = MARG (J)
           IF ((TMARG.LT.0) .E. (TMARG.GT.INCR)) INCR = TMARG
200 CONTINUAR
        FAZER 210 I = 1, NOSCAN
           PROW (SCAN (I)) = PROW (SCAN (I)) + INCR
210 CONTINUAR
        DO 220 J = 1, N
           IF (MARG (J) .EQ.0) PCOL (J) = PCOL (J) + INCR
220 CONTINUAR
        DO 1400 J = 1, N
           IF (MARG (J) .GE.0) IR PARA 1400
            MARG (J) = MARG (J) -INCR
              IF (MARG (J) .EQ.0) THEN
                IF (ASSIGN (J) .EQ.0) THEN
                  CURCOL = J
                  Currow = COLLAB (CURCOL)

C Faça AUGMENTATION

                  IR PARA 1500
                MAIS
                  NOSCAN = NOSCAN + 1
                  SCAN (NOSCAN) = ASSIGN (J)
                END IF
             END IF
1400 CONTINUAR
        IL1 IL2 + 1 =
        IL2 = NOSCAN
        IR PARA 1030

C PREÇO END CHANGE


C AUGMENTATION COMEÇA AQUI
```

```
1500 IF (ROWLAB (Currow) .EQ.0) THEN
          ASSIGN (CURCOL) = Currow
          ROWLAB (Currow) = CURCOL
          IR PARA 1700
        END IF
        TA = ROWLAB (Currow)
        ASSIGN (CURCOL) = Currow
        ROWLAB (Currow) = CURCOL
        CURCOL = TA
        Currow = COLLAB (CURCOL)
        IR PARA 1500
1700 IF (NEWNOL.EQ.1) IR PARA 2000

        FAZER 240 I = 1, NEWNOL-1
           IF (LIST (I) .EQ.CURROW) THEN
             FAZER 250 K = I + 1, NEWNOL
                LIST (KI) = LIST (K)
250 CONTINUAR
             DO 260 K = 1, I-1
                LIST (NEWNOL-I + K) = SCAN (K)
260 CONTINUAR
             NEWNOL = NEWNOL-1
             IR PARA 1020
           END IF

240 CONTINUAR

        NEWNOL = NEWNOL-1
        IR PARA 1020

C AUGMENTATION END


C ********************************************** ******
C
C rotina de saída. Cheques para Optimality DA SOLUÇÃO
C calcula o custo ótimo, e compila SOLUÇÃO
```

```
C ESTATÍSTICAS. O utilizador pode substituir este pelo código que
C ESCREVE AT o dispositivo apropriado a solução ideal
C CONTIDAS NO ASSIGN Array (.).
C
C ********************************************** ******

2000 CONTINUAR

        TT = (LONG (362) - TIMER) / 60

        PRINT *, 'TIME TOTAL =', TT, 's.'
        PRINT *, 'ITERATIONS LEILÃO NO DE ingênuo:', ISIMPL
        PRINT *, 'NO DE SH. ITERATIONS caminho: ', NASSIH
X PRINT *, 'NO das variações de preços:', IHPRC

C verificação de viabilidade de uma solução e calcular o custo

        FAZER 265 I = 1, N
           ROWLAB (I) = 0
265 CONTINUAR

        NOASS = 0
        FAZER 280 J = 1, N
           IF (ASSIGN (J) .GT.0) THEN
             ROWLAB (ASSIGN (J)) = J
             NOASS = + 1 NOASS
           END IF
280 CONTINUAR

        IF (NOASS.LT.N) THEN
          PRINT *, "o número de colunas atribuído é", NOASS, '(muito pequeno)'
        END IF

        TCOST = 0
        FAZER 300 I = 1, N
          J = ROWLAB (I)
          IF (J.EQ.0) THEN
```

```
            PRINT *, 'ROW', I, 'não é atribuído "
          END IF
          FSTARC = FOUT (I)
          LSTARC FOUT = (I + 1) -1
          MINCOST = ILARGE
          FAZER 310 ARC = FSTARC, LSTARC
           CURCOL = END (ARC)
           TMARG = PROW (I) -PCOL (CURCOL) + COST (ARC)
           IF (TMARG.LT.0) THEN
             PRINT *, 'COMPL. Frouxidão VIOLAÇÃO: ROW ', I,' COLUMN ', CURCOL
           END IF
           IF (CURCOL.EQ.J) THEN
             IF (MINCOST.GT.COST (ARC)) THEN
                MINCOST = CUSTO (ARC)
                ROWMARG = TMARG
             END IF
           END IF
310 CONTINUAR
          IF (ROWMARG.NE.0) THEN
            PRINT *, 'COMPL. Folga. VIOLAÇÃO AT ATRIBUÍDO ARC DE ROW ', I
          END IF
          IF (MAXSET) THEN
            TCOST = TCOST-MINCOST
          MAIS
            TCOST = TCOST + MINCOST
          END IF
300 CONTINUAR

        ESCREVA (9,2100) TCOST
2100 FORMATO ('atribuição de custo =', F14.2)

        PRINT *, '****************************************** '
        PRINT *, 'programa terminou; PRESS <CR> '

        PAUSA
        PARE
```

```
        FIM



      SUBROUTINE SETRAN (ISEED)
      IMPLICIT real * 8 (AH, OZ), Integer * 4 (IN)
C ********************************************** **********************
C PORTABLE congruential (uniforme) RANDOM gerador de números:
NEXT_VALUE C = [(7 ** 5) * PREVIOUS_VALUE] MODULO [(2 ** 31) -1]
C
C Este gerador é composto por duas ROTINAS:
C (1) SETRAN - inicializa constantes e SEED
C (2) RRAN - gera um número aleatório REAIS
C
C O GERADOR DE RODA uma máquina com pelo menos 32 bits de precisão.
C DA SEED (ISEED) deve estar no intervalo (1, (2 ** 31 -1)).
C ********************************************** **********************
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
      IF (ISEED.LT.1) PARADA 77
      MULT = 16807
      MODUL = 2147483647
      I15 = 2 ** 15
      I16 = 2 ** 16
      JRAN = ISEED
      RETURN
      FIM
C
      RAN função real ()
      IMPLICIT real * 4 (AH, OZ), Integer * 4 (IN)
C ********************************************** **********************
C RAN gera um número aleatório real entre 0 e 1
C ********************************************** **********************
      COMUM / RANDM / MULT, MODUL, I15, I16, JRAN
      Ixhi = JRAN / I16
      IXLO = JRAN-Ixhi * I16
      IXALO = IXLO * MULT
      LEFTLO = IXALO / I16
```

```
IXAHI = Ixhi * MULT
IFULHI = IXAHI + LEFTLO
IRTLO = IXALO-LEFTLO * I16
IOVER = IFULHI / I15
IRTHI = IFULHI-IOVER * I15
JRAN = ((IRTLO-MODUL) + IRTHI * I16) + IOVER
IF (JRAN.LT.0) JRAN = JRAN + MODUL
RAN = FLOAT (JRAN) / FLOAT (MODUL)
RETURN
FIM
```