O Problema do Caminho Mínimo: Algoritmo Label Correcting

Prof. Dr. Daniel Caetano

Objetivo: Capacitar para a identificação de ciclos em problemas de fluxo em rede e apresentar o algoritmo Label Correcting para a resolução destes problemas.

INTRODUÇÃO

Conceitos Chave:
- Problema: reduzir caminho
- Identificar existência de ciclos
- Algoritmos Específicos
    * Label Setting
    * Label Correcting
    * Network Simplex
    * Out-of-Kilter
- Algoritmo Label Correcting

Existem diversas situações o projetista de transportes se depara com algum problema em que é necessários otimizar um trajeto, visando redução de custo de custo, tempo... enfim, problemas que envolvem a redução de algum tipo de "custo" que varia de acordo com o trajeto escolhido.

Este tipo de problema de melhor caminho, ponto a ponto, pode ser tratado com algoritmos mais específicos que o Simplex, como o Label Setting, Label Correcting, Network Simplex, Out-of-Kilter, dentre outros. No caso específico de caminho mínimo entre dois pontos sem mais nenhum tipo de restrição, os algoritmos mais adequados são o Label Correcting e o Label Setting. Estas anotações tratam do primeiro.

1. FORMALIZAÇÃO: "O Problema do Motorista de Taxi"

Conceitos Chave:
- Levar um passageiro de uma Origem a um Destino
- Minimizar distância percorrida
- Não se tem informações do próximo passageiro
- Ignorar trânsito

Um taxista trabalha para uma empresa que lhe paga um valor fixo de acordo com a distância entre os baricentros das sub-regiões da cidade. Obviamente, para este taxista, é um grande negócio reduzir ao máximo a distância percorrida, economizando combustível e tempo.

O problema a ser resolvido trata-se, portanto, de um caso específico dentro dos problemas de fluxo em rede: o problema do caminho mínimo. Este problema tem ainda características mais específicas, dentro de um problema de caminho mínimo: é um problema em que um único passageiro será transportado da origem ao destino. E mais: não há como saber qual será a origem ou destino do próximo passageiro a ser transportado, o que restringe a otimização tão somente ao caminho mínimo entre dois pontos.

Como apenas uma unidade (o passageiro) deve ser transportada, também não é necessária uma preocupação com a limitação de capacidade de fluxo nas vias e, por simplicidade, será desconsiderada a possibilidade de uma via congestionada.

Uma representação física de um possível problema a ser resolvido para o taxista é mostrado na figura 1.:

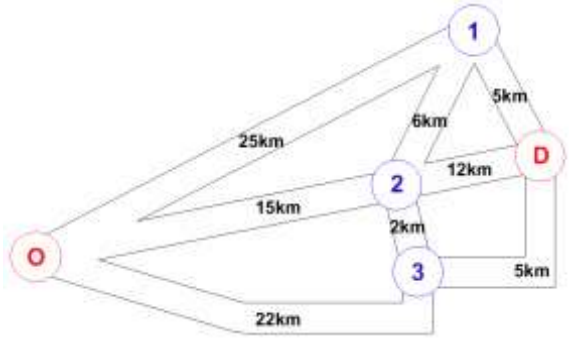


Figura 1: Exemplo de um problema a ser resolvido para o taxista

Em um problema deste nível, com este número reduzido de vias, possibilidades e restrições... É até possível pensarmos em um cálculo manual. Entretanto, a situação se torna extremamente mais complexa para um problema maior, como o apresentado na figura 2.

Figura 2: Um problema real, mais complexo

Apesar de provavelmente ser possível utilizar o mesmo processo de solução, fazer todas as iterações necessárias manualmente seria uma tarefa bastante custosa e ineficiente - para não dizer aborrecida.

Sabe-se que é possível resolver automaticamente problemas deste tipo, já que existem soluções que resolvem este tipo de problema em tempo real, em sistemas como o Google Maps e o Waze, por exemplo.

2. MODELAGEM MATEMÁTICA

Conceitos Chave:
- Representação como Grafo
  * Tamanho dos arcos = Distância de um nó a outro
- Entrada e Saída do Sistema
  * Solução possível => rede balanceada
- Equilíbrio no nó
- Equações
  * Genérica
  * Nó de Origem / Nó de Destino
  * Geral
- Economia
  * Custo de um arco
  * Minimizar custo total
- Número de Equações + Número de Incógnitas => Não pelo Simplex!

O primeiro passo para uma automatização é a modelagem, onde serão desprezadas todas as características irrelevantes para a resolução, além de serem explicitados todos os dados que possam ser necessários à solução.

Como a ideia apresentada desde o início é resolver o problema por um algoritmo de fluxo em rede, modelar o problema como uma rede é praticamente uma necessidade. Para tanto, será utilizada a representação em grafo, com nós e arcos, sendo que os nós representaram as interseções e os arcos, as vias (interligações entre os nós).

É possível associar as distâncias aos comprimentos dos arcos, possibilitando o cálculo das distâncias pela soma dos comprimentos dos arcos. Neste instante, a rede do taxista pode ser representada como na figura 3.

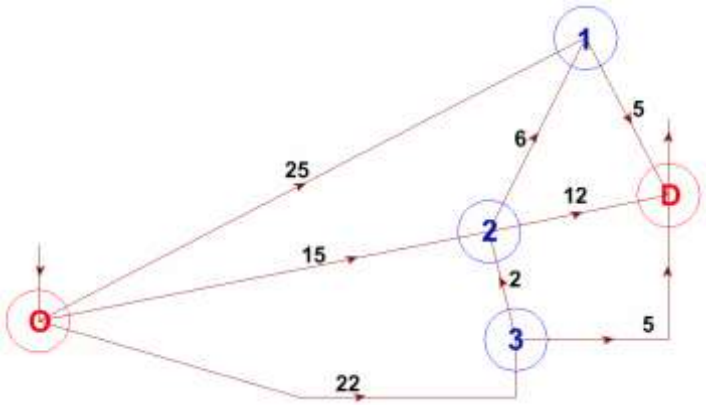


Figura 3: "Modelagem gráfica" da rede

A indicação dos pontos de partida e chegada pode ser feita com duas seta, uma mostrando que entrou 1 unidade no ponto de partida e uma indicando que saiu uma unidade no ponto de chegada. Entretanto, do ponto de vista do equilíbrio dos nós (tudo que entra = tudo que sai), isso cria um problema sério: há um indivíduo entrando em um nó (nó inicial) e ele simplesmente desaparece (já que a representação dele saindo do nó simplesmente não foi feita). A mesma incoerência ocorre no nó final, onde um indivíduo sai do sistema sem nunca ter chegado àquele nó.

É necessário, então, indicar variáveis em cada um dos arcos, especificando se o indivíduo está passando por aquele arco ou não, de forma que todos os nós estejam equilibrados: se chegou um indivíduo a um dado nó, este indivíduo também precisa sair daquele nó, por algum outro arco. Nesta situação, pode-se dizer que a rede estará resolvida, ou seja, balanceada, como pode ser visto na figura 4.

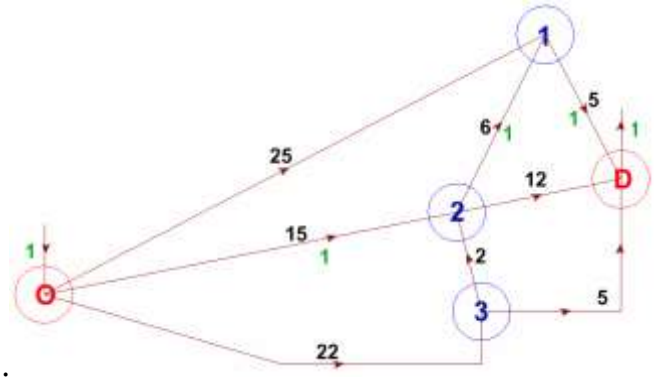


Figura 4: *Rede balanceada*, representando uma solução possível para o problema

Para representar isso matematicamente, será definida uma variável para cada arco, indicando se o indivíduo passa ou não por aquele arco. Por exemplo, a variável X_{ij} pode indicar se o indivíduo está usando o arco que sai do nó *i* para o nó *j*. Se esta variável vale 1, indica que o indivíduo passa por aquele arco; se ela valer 0, o indivíduo não passa por aquele arco.

Em um nó podem chegar vários arcos, e também podem sair vários arcos. Assim, para que um nó esteja em equilíbrio, isto é, tudo que chegar nele tem também que sair, a *soma do fluxo de todos os arcos que chegam* neste nó deve ser **igual** à *soma do fluxo de todos os arcos que saem* deste nó, como pode ser visto na figura 5.

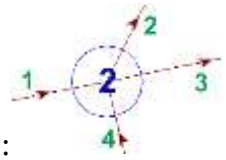


Figura 5: Um nó *balanceado*: tudo que chega, sai.

Uma forma simples de dizer isso é que "um nó em equilíbrio é aquele em que não fica ninguém. Todo mundo que chega nele, sai dele". Considerando que A seja o conjunto de todos os arcos (m,n) de uma rede, pode-se representar matematicamente esta situação para o nó i da seguinte forma:

$$\sum_{(n,i)\in A} x_{ni} - \sum_{(i,m)\in A} x_{im} = 0$$

Esta equação pode ser lida da seguinte forma: "Se for somado o fluxo de todos os arcos que chegam em um nó ***i*** e disso for subtraído o fluxo de todos os nós que saem deste mesmo nó ***i***, o resultado deve ser sempre *zero*. Em outras palavras, todo mundo que chega em no nó ***i***, sai do nó ***i***.

Esta representação é boa para nós genéricos, do meio da rede, mas não é válida para os nós de Origem e de Destino, já que nestes nós há o "aparecimento" e o "desaparecimento" de indivíduos, respectivamente.

Assim, nestes nós, a representação matemática do equilíbrio é ligeiramente diferente. Como no nó de Origem haverá o surgimento de uma unidade e no de Destino haverá o desaparecimento de uma unidade, será feita a seguinte modificação: no nó origem, será adicionado *uma* unidade na equação, e no nó destino será subtraída *uma* unidade. Assim, as equações serão:

Nó Genérico:

$$\sum_{(n,i)\in A} x_{ni} - \sum_{(i,m)\in A} x_{im} = 0$$

Nó de Origem:

$$1 + \sum_{(n,i)\in A} x_{ni} - \sum_{(i,m)\in A} x_{im} = 0 \Rightarrow \sum_{(n,i)\in A} x_{ni} - \sum_{(i,m)\in A} x_{im} = -1$$

:

Nó de Destino:

$$-1 + \sum_{(n,i)\in A} x_{ni} - \sum_{(i,m)\in A} x_{im} = 0 \Rightarrow \sum_{(n,i)\in A} x_{ni} - \sum_{(i,m)\in A} x_{im} = 1$$

Como é possível observar, o que muda é apenas o lado direito da equação: o valor é 0 nos nós genéricos, -1 no nó de origem e 1 no nó de destino. Assim, é possível indicar a equação para todos os nós de forma genérica:

$$\sum_{(n,i)\in A} x_{ni} - \sum_{(i,m)\in A} x_{im} = b$$

Se ***i*** é o nó inicial, *b = -1*. Se ***i*** é o nó final, *b = 1*. Em todas as outras situações, *b = 0*.

Como isso, está completo o conjunto de restrições que garantem o balanceamento dos nós e, portanto, da rede. Entretanto, as restrições garantem uma solução possível; é necessário definir uma função objetivo que garanta o menor comprimento possível para a solução.

Como x_{ij} terá valor 1 se um arco estiver sendo usado e 0 se não estiver sendo usado, pode-se afirmar que o custo de um arco na solução é:

$$\mathbf{c_{ij} * x_{ij}}$$

onde c_{ij} é o comprimento do arco que vai de ***i*** para ***j***. Isso significa que se o arco estiver sendo usado, ele terá um custo $\mathbf{c_{ij} * 1 = c_{ij}}$. Se o arco não estiver sendo usado, seu custo será $\mathbf{c_{ij} * 0 = 0}$. A soma do custo de todos os arcos definidos desta forma, dá o custo da

solução, e a função objetivo - que é minimizar este custo - pode ser representada da seguinte forma:

$$[\min] \sum_{(i,j)\epsilon A} c_{ij} \cdot x_{ij}$$

Em outras palavras, esta equação significa que deve-se minimizar a soma do custo de todos os trechos (arcos) que compõem o caminho usado.

Com essa modelagem completa, é possível observar que há praticamente uma equação de restrição para cada nó do sistema (o equilíbrio de cada nó) e cada uma delas terá tantas variáveis quantos forem os arcos que chegam e saem do nó ao qual esta restrição é referente.

O grande número de equações e variáveis decorrentes do grande número de nós e arcos necessários para representar um "problema real" se torna então uma dificuldade a mais, se for considerada uma solução usando o algoritmo Simplex.

3. O PROBLEMA DOS CICLOS

Conceitos Chave:
- Verificação de Ciclos: Representação em camadas
  * Caminhamento sempre descendo nas camadas
- Remoção de Ciclos
  * Perda de otimalidade

Alguns algoritmos, como o *Label Setting*, são mais eficientes que aquele apresentado aqui. No entanto, esses outros algoritmos, em geral, apresentam uma limitação relevante: não permitem a resolução de redes que possuam ciclos. Analisando a figura 2, é possível verificar que tal rede possui ciclos e, portanto, não poderia ser resolvida por um algoritmo como o *Label Setting*.

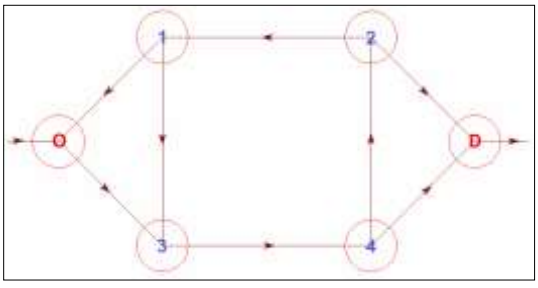


Figura 2: Representação de uma rede com ciclos

Uma forma prática para verificar se uma rede contém ou não ciclos é analisar a possibilidade de representá-la em camadas ordenadas, sendo que a movimentação dos arcos deve sempre ir de uma camada para a seguinte, nunca para a mesma camada ou a camada anterior. Uma rede sem ciclos sempre pode ser descrita assim, como, por exemplo, aquela do problema do taxista original, representada na figura 3.

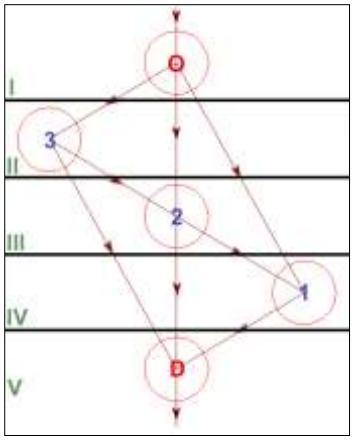


Figura 3: Uma rede sem ciclos representada em camadas

É possível notar que não há como representar a rede da figura 2 num formato como este, sendo que inevitavelmente haverá arcos ligando nós de níveis mais altos em direção a nós de níveis mais baixos, evidenciando assim a existência de ciclos.

Uma alternativa para contornar o problema dos ciclos pode ser a prévia utilização de um algoritmo de remoção de ciclos – como um dos geradores de árvore mínima, já vistos – e, apenas depois disso, lançar mão da utilização do *Label Setting*. Entretanto, esta pode não ser uma forma apropriada, já que a eliminação de ciclos pura e simples pode remover arcos que fariam parte da melhor solução e, sendo assim, deteriorando a qualidade da solução do problema.

Por esta razão, é necessário apresentar um algoritmo que seja capaz de lidar com o problema dos ciclos, garantindo que a solução ótima seja atingida. Esta é a base da proposta do *Label Correcting*.

4. O ALGORITMO LABEL CORRECTING

Conceitos Chave:

- Solução para problemas com ciclos
  * Modificação na ordem de cálculo e critério de parada
- Três dados na etiqueta: *nó pai*, *distância acumulada* e *necessidade de recálculo*.
- Lógica:
  a) Etiquetar
  b) Selecionar nó que deve ter seus descendentes recalculados (com menor distância acumulada)
  c) Calcular todos os seus descendentes (marcando suas etiquetas), trocando eventuais etiquetas existentes, se novo caminho for melhor.
  d) Marcar etiqueta do nó atual como "sem necessidade de recálculo"
  e) Voltar ao passo (b)
- Permite adição de arcos "em tempo de execução".

O algoritmo *Label Correcting* (GALLO; PALOTINO, 1984) foi desenvolvido para lidar com os casos em que o algoritmo *Label Setting* não se comporta bem, ou seja, aqueles que possuem ciclos.

Uma possível sistematização para o *Label Correcting* é:

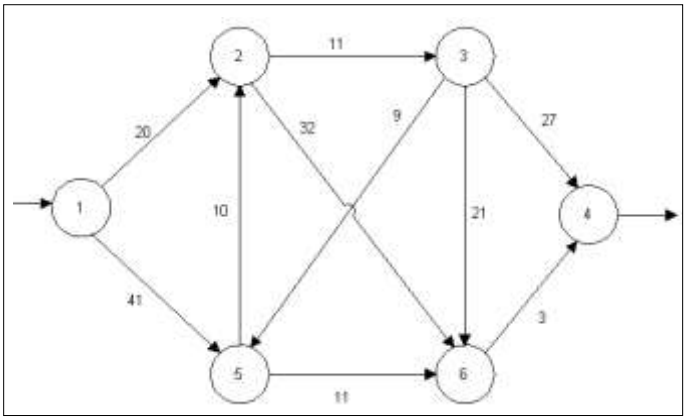
1. Cria-se uma etiqueta em todos os nós, indicando a distância acumulada "0" e "-1" como o nó antecessor. Indica-se também em todas as etiquetas que seus sucessores não precisam ser calculados, com o valor "0".
2. Marca-se o nó 0 (origem) como sendo antecessor de si próprio, indicando "0" em sua etiqueta, distância total acumulada "0" e indicando que este nó precisa ter seus sucessores calculados, indicando "1" na etiqueta.
3. Dentre todos os nós marcados para que seus sucessores sejam calculados, seleciona-se aquele que tem menor distância acumulada. Se não houver qualquer nó com indicação de recálculo de sucessores, fim do processo.
4. Para o nó selecionado, calcula-se a distância total acumulada para todos os nós sucessores deste, sendo esta distância a soma da distância total acumulada até o nó atual com o comprimento do arco que liga este nó ao referido sucessor.
5. Caso o nó sucessor não tenha ainda sido etiquetado com um antecessor ou ainda que a nova distância seja inferior à anteriormente indicada na etiqueta do sucessor, indica-se no nó sucessor a nova distância acumulada, o novo nó antecessor e também se deve indicar que seus descendentes precisam ser recalculados.
6. Voltar ao passo 3.

Este algoritmo resolve casos com ou sem ciclos, embora a existência de ciclos leve a um aumento na quantidade de cálculos necessários para a solução do problema.

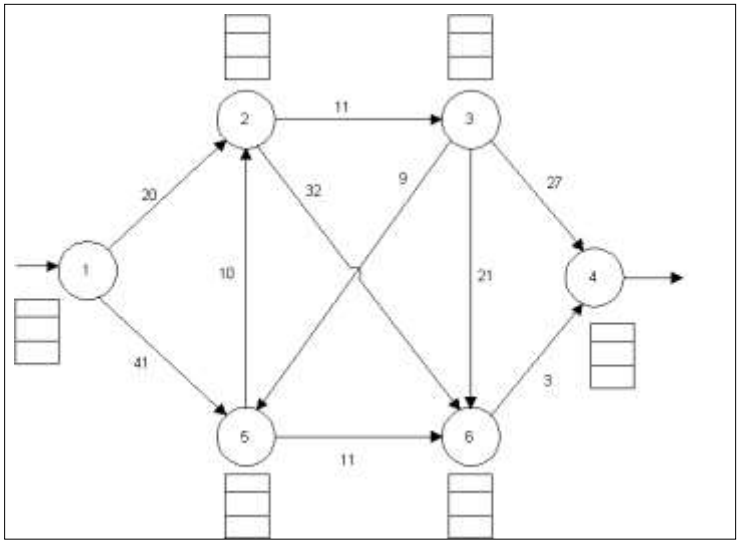
Uma outra característica positiva deste algoritmo é que ele permite que novos nós e arcos sejam adicionados na rede, aproveitando-se o resultado da rede já calculada. Para tanto, basta marcar todos os nós originais aos quais são ligados novos arcos de saída como sendo necessitando recálculo de seus descendentes.

5. EXEMPLO

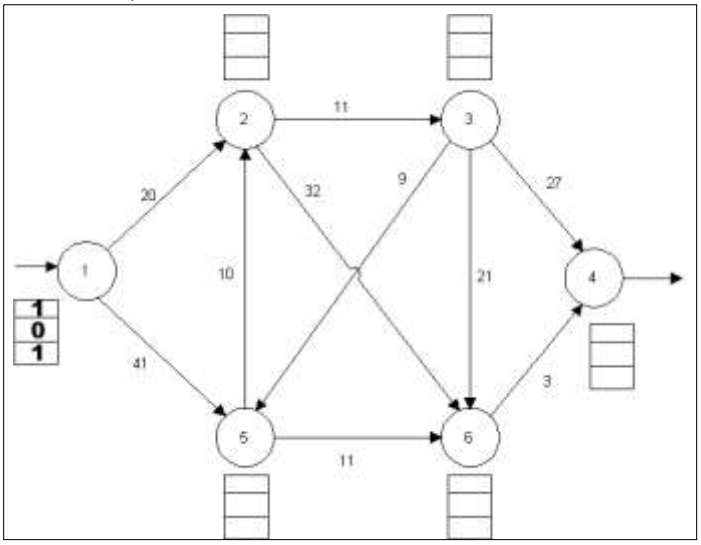
Usando o algoritmo Label Correcting, encontrar o caminho mínimo entre os pontos 1 e 4.



Passo 1: O primeiro passo para a resolução é indicar as etiquetas:



Passo 2: Realizar as marcações iniciais: o nó 1 é o pai de si mesmo (na parte de cima da etiqueta), com distância acumulada 0 (na parte do meio) e é preciso calcular seus descendentes (1 na parte de baixo):

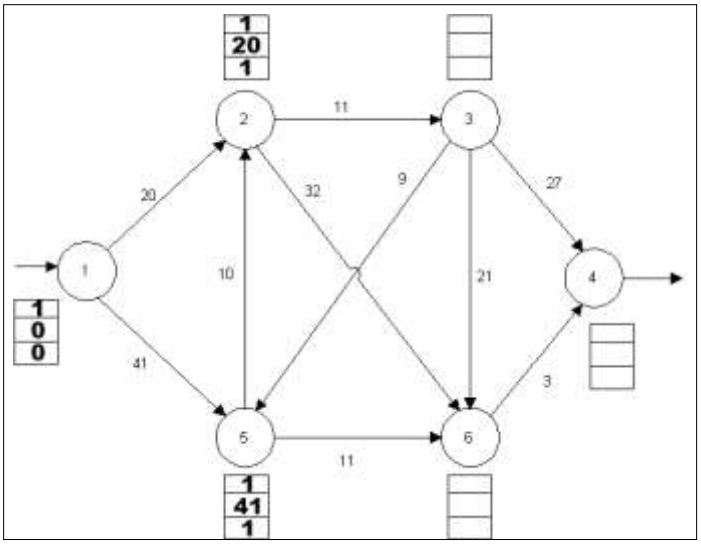


Passo 3: Dos nós que possuem descendentes a calcular (apenas o nó 1, por enquanto), deve-se selecionar aquele que tem a menor distância acumulada: o nó 1, tem distância acumulada 0. Assim, o nó 1 será usado para cálculo no próximo passo.

Passo 4: A partir do nó selecionado (neste caso, o nó 1), calcula-se todos os seus descendentes: 2 e 5:

- Nó 2: 0 + 20 = 20 (distância acumulada até 1 + comprimento do arco de 1 a 2)
- Nó 5: 0 + 41 = 41 (distância acumulada até 1 + comprimento do arco de 1 a 5)

Como os nós descendentes calculados ainda não haviam sido preenchidos, deve-se simplesmente marcar estes valores na etiqueta de cada nó descendente, indicando também seu antecessor ("pai") como sendo o nó 1. Indica-se ainda que estes 2 nós precisam que seus descendentes sejam calculados (indicando 1 na parte inferior da etiqueta), e que o nó 1 não precisa mais ter seus descendentes calculados (indicando 0 na parte inferior da etiqueta):

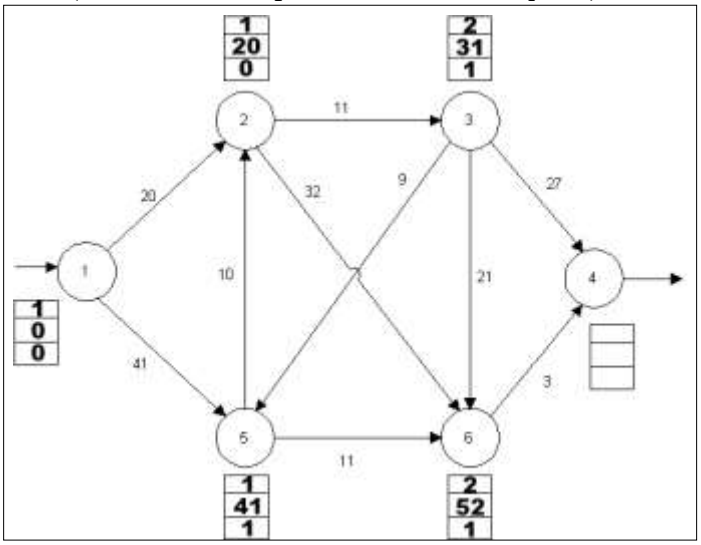


Passo 5: Dos nós que possuem descendentes a calcular (2 e 5), selecionar aquele que tem a menor distância acumulada: o nó 2 tem 20 acumulado e o nó 5 tem 41 acumulado. Assim, o nó 2 será usado para cálculo no próximo passo.

Passo 6: A partir do nó selecionado (neste caso, o nó 2), calcular todos os seus descendentes: 3 e 6:

- Nó 3: 20 + 11 = 31 (distância acumulada até 2 + comprimento do arco de 2 a 3)
- Nó 6: 20 + 32 = 53 (distância acumulada até 2 + comprimento do arco de 2 a 6)

Como os nós descendentes calculados ainda não foram preenchidos, deve-se simplesmente marcar o valor na etiqueta, indicando também seus antecessores ("pais") como sendo o nó 2. Adicionalmente, indica-se que estes nós precisam que seus descendentes sejam calculados (indicando 1 na parte inferior das etiquetas), e que o nó 2 não precisa mais ter seus descendentes calculados (indicando 0 na parte inferior da etiqueta):



Passo 7: Dos nós que possuem descendentes a calcular (3, 5 e 6), selecionar aquele que tem a menor distância acumulada: o nó 3 tem 31 acumulado; o nó 5 tem 41 acumulado; e o nó 6 tem 52 acumulado. Assim, o nó 3 será usado para cálculo no próximo passo.

Passo 8: A partir do nó selecionado (neste caso, o nó 3), calcula-se todos os seus descendentes: 4, 5 e 6:

- Nó 4: 31 + 27 = 58 (distância acumulada até 3 + comprimento do arco de 3 a 4)
- Nó 5: 31 + 9 = 40 (distância acumulada até 3 + comprimento do arco de 3 a 5)

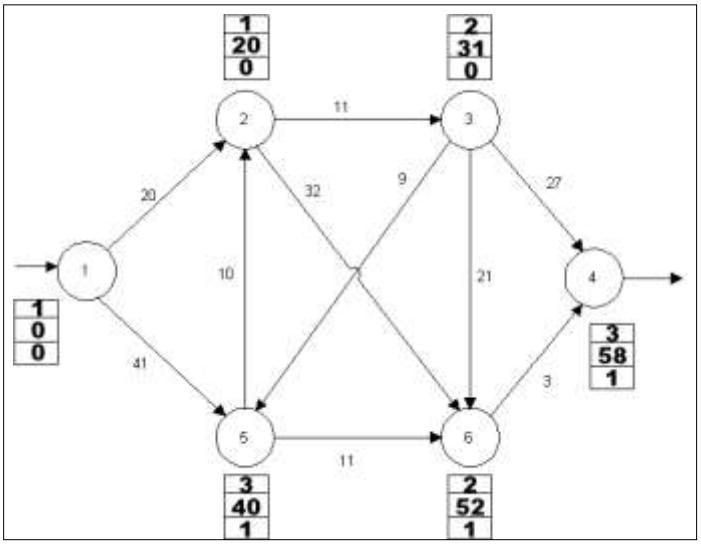
- Nó 6: 31 + 21 = 52 (distância acumulada até 3 + comprimento do arco de 3 a 6)

Como o nó 4 ainda não havia sido calculado, deve-se simplesmente preencher sua etiqueta da mesma maneira que nos passos anteriores: pai (3) na parte de cima, distância acumulada (58) no meio e a necessidade de cálculo dos descendentes (1) na parte inferior.

No caso dos nós 5 e 6, entretanto, já havia etiquetas definidas. Antes de substituí-las, é preciso verificar se o novo caminho até estes nós é melhor do que o anteriormente definido.

No nó 5, o valor anterior de distância acumulada era 41, como é possível ver pela etiqueta. Seguindo o novo caminho, o valor acumulado fica 40. Como 40 é menor que 41, o novo caminho é melhor que o antigo. Por esta razão, a etiqueta do nó 5 será **corrigida** com os valores novos de pai (3), distância acumulada (40) e a necessidade de recálculo de seus descendentes.

No nó 6, o valor anterior de distância acumulada era 52, e seguindo o novo caminho, ele permanece 52. Desta forma, **não se mexe** na etiqueta, já que o caminho anterior era tão bom quanto o novo. Finalmente, o nó 3 deve ser remarcado indicando que seus descendentes não mais precisam ser recalculados:



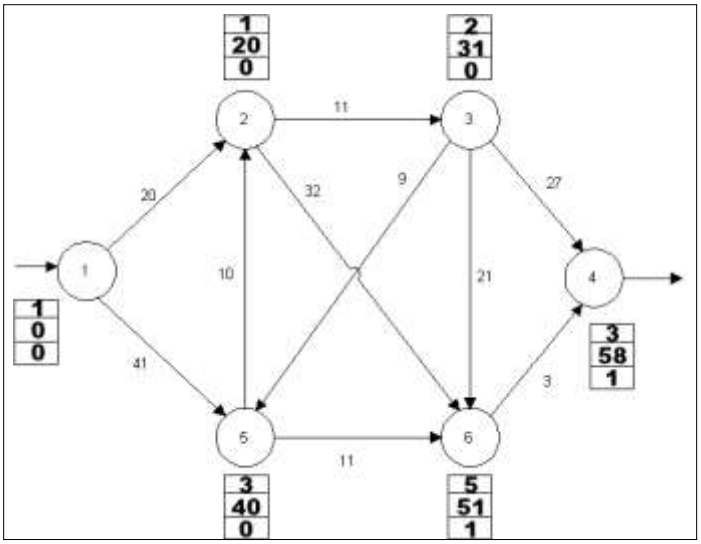
Passo 9: Dos nós que possuem descendentes a calcular (4, 5 e 6), deve-se selecionar aquele que tem a menor distância acumulada: o nó 4 tem 58 acumulado; o nó 5 tem 40 acumulado; e o nó 6 tem 52 acumulado. Assim, o nó 5 será usado no próximo passo.

Passo 10: A partir do nó selecionado (neste caso, o nó 5), calcular todos os seus descendentes: 2 e 6:

- Nó 2: 40 + 10 = 50
- Nó 6: 40 + 11 = 51

Como todos os nós descendentes calculados já foram preenchidos, deve-se verificar se os novos caminhos são melhores que os antigos. No nó 2, a distância acumulada anterior era 20, e pelo novo caminho ela fica 50, sendo claramente uma piora. Este nó fica como estava.

No nó 6, a distância acumulada anterior era 52, e a nova é 51. Assim, há uma melhoria, e este nó deve ter sua etiqueta remarcada para pai como nó 2, distância acumulada 51 e indicar que seus descendentes precisam ser recalculados. Adicionalmente deve-se indicar que o nó 5 não precisa mais ter seus descendentes calculados:

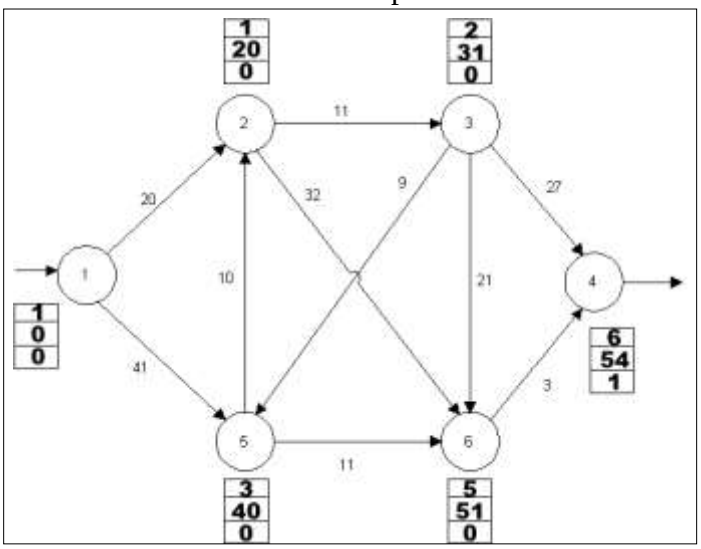


Passo 11: Dos nós que possuem descendentes a calcular (4 e 6), selecionar aquele que tem a menor distância acumulada: o nó 4 tem 58 acumulado; e o nó 6 tem 51 acumulado. Assim, o nó 6 será usado no próximo passo.

Passo 12: A partir do nó selecionado (neste caso, o nó 6), calcular todos os seus descendentes: 4:

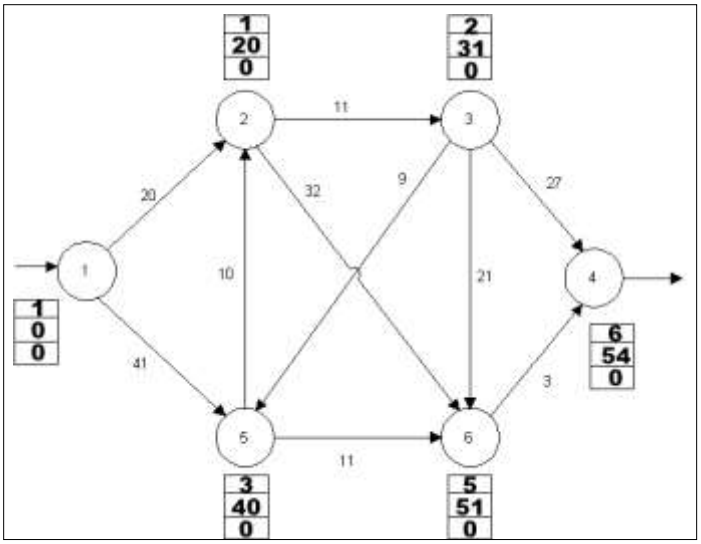
- Nó 4: 51 + 3 = 54

O nó 4 já tinha uma etiqueta, com uma distância acumulada de 58, e pelo novo caminho ela se torna 54. Assim, o novo caminho é melhor que o anterior, e a etiqueta do nó 4 deve ser corrigida: 6 (pai), 54 (distância) e 1 (calcular descendentes). O nó 6 deve ser ajustado para indicar que seus descendentes não mais precisam ser calculados (0):



Passo 13: Sobrou apenas um nó (4) com descendentes a serem calculados, então, o nó 9 será usado no próximo passo.

Passo 14: Ocorre que o nó 4 não tem descendentes, então sua etiqueta pode ser corrigida para indicar que seus descendentes não precisam ser calculados.



Passo 15: Como não há mais nós com descendentes para calcular, esta é a solução do problema. O caminho mínimos é definido da mesma forma que no label setting:

1 a 4: 4-6-5-3-2-1... ou 1-2-3-5-6-4, com distância total 54.

5. BIBLIOGRAFIA

AHUJA, R.K; MAGNANTI, T.L; ORLIN, J.B. **Network Flows**: Theory, Algorithms and Applications. New Jersey: Prentice Hall, 1993.

BRADLEY, S. P.; HAX, A. C.; MAGNANTI, T. L. **Applied mathematical programming**. Reading, Mass.: Addison-Wesley Pub. Co., 1977.

CHVATAL, V. **Linear programming**. New York: W. H. Freeman, 1983.

GALLO, G; PALLOTTINO, S. Shortest path methods in transportation models. (M. Florian, ed.) Transportation Planning Models. North Holland, Elsevier Science Publishers. p. 227-256, 1984.

WINSTON, W. L. **Operations research**: applications and algorithms. S.I.: International Thomson Publishing, 1994.