PCS 2215

Fundamentos de Engenharia de Computação II

<u>Aula 12</u>

<u>Modelos de Rede e Algoritmo do Fluxo Máximo</u>

Líria Sato

Professor Responsável

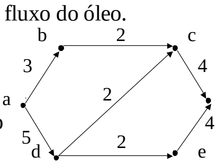
versão: 1.2 (set 2002)

Conteúdo

5.1 Modelos de rede

Considere o grafo orientado apresentado na figura 5.1, que representa uma rede de oleoduto. O óleo é descarregado no porto ***a*** e bombeado através da rede para a refinaria ***z.*** Os vértices ***b, c ,d*** e ***e*** representam estações de bombeamento intermediárias. As arestas orientadas representam subdutos do sistema e a direção do fluxo do óleo.

Problema: encontrar o fluxo máximo



Fluxo(b,c)=2m^3/seg.

Fig. 5.1: rede de transporte

5.1 Modelos de rede

Rede de transporte: é um grafo simples, ponderado e orientado, satisfazendo:

a) um vértice específico, a fonte, que não tem arestas chegando.

b) um vértice, o destino, que não tem arestas saindo.

c) O peso Cij da aresta orientada (i,j), chamada a capacidade de (i,j), é um número não negativo.

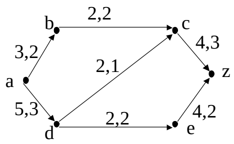
A figura 5.1 é um exemplo de rede de transporte.

5.1 Modelos de rede

Fluxo

O fluxo em uma rede atribui um valor de fluxo em cada aresta orientada que não pode exceder a capacidade daquela aresta.Ex.:$C_{ab}=3m^3/s$, $F_{ab}=2m^3/s$

O fluxo de entrada em um vértice *v*, que não é a fonte e nem o destino, é igual ao fluxo de saída de *v*.



Uma aresta *e* é rotulada *x*,*y*.
x: capacidade
y: o fluxo em e

Fig.5.2: fluxo em uma rede

5. Modelos de rede

O fluxo de saída de *a* é a mesma do fluxo de entrada de *z*.

No exemplo da fig. 5.2, as atribuições:

$F_{ab}=2$, $F_{bc}=2$, $F_{ad}=3$, $F_{cz}=3$, $F_{dc}=1$, $F_{de}=2$, $F_{ez}=2$, definem um fluxo para a rede.

O fluxo de entrada em d é igual ao fluxo de saída de d.

$F_{ad}=F_{dc}+F_{de}=3$

Tem-se que: o fluxo de saída da fonte *a*, é o mesmo que o fluxo entrando no destino *z*.

$$F_{ab}+F_{ad} = F_{cz} + F_{ez}$$

Ambos são 5.

5. Modelos de rede

Definição 8.1.3

Seja *G* uma rede de transporte. Seja C_{ij} a capacidade da aresta orientada *(i,j)*. Um fluxo *F* em *G* atribui a cada aresta orientada *(i,j)* um número não negativo F_{ij} tal que:

a) $F_{ij} \leq C_{ij}$

b) Para cada vértice j, que não é a fonte nem o destino,

$\Sigma_i F_{ij} = \boldsymbol{\Sigma_i F_{ji}}$ (conservação do fluxo)

$\Sigma_i F_{ij}$: fluxo de entrada em j

$\Sigma_i F_{ji}$: fluxo de saída de j

5. Modelos de rede

Teorema 8.1.5

Dado um fluxo *F* em uma rede, o fluxo de saída da fonte *a* é igual ao fluxo de entrada no destino *z*; tal que:

$\Sigma_i F_{ai} = \Sigma_i F_{iz}$

Definição 8.1.6

Seja um fluxo em uma rede, o valor :

$F = \Sigma_i F_{ai} = \Sigma_i F_{iz}$

é chamado o valor do fluxo da rede.

5. Modelos de rede

Problema:

O problema para a rede de transporte pode ser definido como, encontrar o fluxo máximo em G, isto é, encontrar um fluxo F, tal que o valor de F é máximo.

5. Modelos de rede

Rede de bombeamento

O grafo da figura 5.3 representa uma rede de bombeamento que provê água proveniente de três represas, w_1, w_2 e w_3, para duas cidades, *A* e *B*. As capacidades dos sistemas intermediários estão especificadas nas arestas. Os vértices *b*, *c* e *d* representam as estações de bombeamento intermediárias. A modelagem deste sistema como uma rede de transporte permite a obtenção do fluxo máximo.

5. Modelos de rede

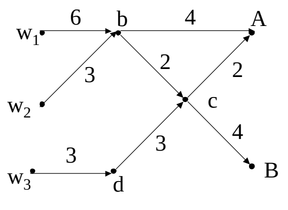


Fig. 5.3: rede de bombeamento

5. Modelos de rede

Podemos obter uma rede de transporte equivalente à rede de bombeamento da figura 5.3, ligando as fontes a uma super fonte e os destinos a um super destino, através de arestas com capacidade infinita.

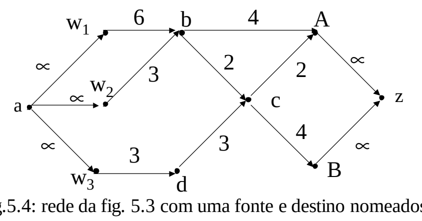


Fig.5.4: rede da fig. 5.3 com uma fonte e destino nomeados

5. Modelos de rede

Uma rede de fluxo de tráfego

É possível ir da cidade A para a cidade C diretamente ou passando pela cidade B. Durante o período das 6:00 as 7:00, a média dos tempos de viagens são:

A para B: 15 minutos
B para C: 30 minutos
A para C: 30 minutos

As capacidades máximas das rotas são:

A para B: 3000 veículos
B para C: 2000 veículos
A para C: 4000 veículos

5. Modelos de rede

Vamos representar o fluxo de tráfego, durante o período das 6:00 as 7:00, como uma rede.

Um vértice representa uma cidade em um tempo particular. Uma aresta conecta 2 vértices se podemos deixar a cidade X em t1 e chegar na cidade Y em t2.

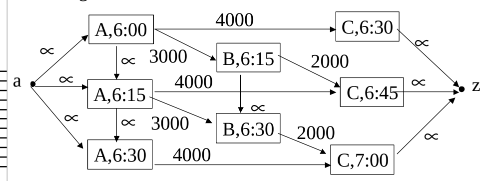


Fig. 5.5: rede de tráfego

5.1 Algoritmo do fluxo máximo

Algoritmo do fluxo máximo

Se G é uma rede de transporte, um fluxo máximo em G é um fluxo com valor máximo.

ideia básica do algoritmo: inicie com algum fluxo inicial mínimo e iterativamente incremente o valor do fluxo até que um aumento não seja mais possível. O fluxo resultante será o fluxo máximo.

5.1 Algoritmo do fluxo máximo

Terminologia que adotaremos nesta seção:

G denota uma rede com fonte *a*, destino *z*, e capacidade *C*.

Consideremos, neste momento, as arestas de *G* como arestas não orientadas e seja

$P=(v_0, v_1,..., v_n)$, $v_0=a$, $v_n=z$,

um caminho de a a z neste grafo não orientado.

Se uma aresta *e* em P é orientada de v_{i-1} para v_i dizemos que *e* é propriamente orientada (com relação a P); caso contrário dizemos que *e* é impropriamente orientada (com relação a P)

5.1 Algoritmo do fluxo máximo

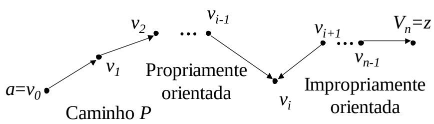


Fig.5.6 :arestas propriamente e impropriamente orientadas

Se podemos encontrar um caminho P da fonte para o destino no qual toda aresta em P é propriamente orientada e o fluxo em cada aresta é menor do que a capacidade da aresta, é possível incrementar o valor do fluxo.

5.1 Algoritmo do fluxo máximo

Exemplo:

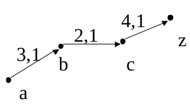
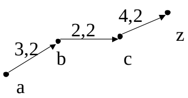


Fig. 5.7: um caminho com apenas arestas propriamente orientadas



O grafo após incremento de 1 no fluxo

5.1 Algoritmo do fluxo máximo

Seja um caminho *P* de *a* para *z* e seja *x* um vértice em *P* que não é *a* nem *z*. Existem 4 possibilidades para a orientação das arestas e_1 e e_2 incidentes em x.

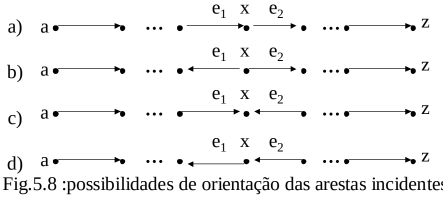


Fig.5.8 :possibilidades de orientação das arestas incidentes em x

5.1 Algoritmo do fluxo máximo

caso a: as duas arestas são propriamente orientadas. Se incrementarmos o fluxo em cada aresta de Δ, o fluxo de entrada em x será ainda igual ao fluxo de saída de x.

caso b: se incrementarmos o fluxo de e_2 de Δ, devemos decrementar o fluxo de e_1 de Δ, de modo que o fluxo de entrada em x será ainda igual ao fluxo de saída de x.

caso c: similar ao caso b, a menos que incrementamos o fluxo de e_1 de Δ e decrementamos o fluxo de e_1 de Δ.

caso d: decrementamos as duas arestas de Δ.

5.1 Algoritmo do fluxo máximo

Teorema 8.2.3

Seja P um caminho de *a* para *z* em uma rede *G* satisfazendo:

a) Para cada aresta propriamente orientada (i,j) em *P*,

$$F_{ij} < C_{ij}$$

b) Para cada aresta impropriamente orientada (i,j) em *P*,

$$0 < F_{ij}$$

Seja Δ=min *X*, onde *X* consiste dos números $C_{ij} - F_{ij}$, para arestas propriamente orientadas (i,j) em *P*, e F_{ij}, para arestas impropriamente orientadas (i,j) em *P*. Define

$$F_{ij}^{*} = \begin{cases} F_{ij} & \text{se (i,j) não está em P} \\ F_{ij} + \Delta & \text{se (i,j) é propriamente orientada em P} \\ F_{ij} - \Delta & \text{se (i,j) não é propriamente orientada em P} \end{cases}$$

Então F* é um fluxo cujo valor é Δ maior do que o valor de F

5.1Algoritmo do fluxo máximo

Algoritmo:

A capacidade de cada aresta deve ser um número inteiro positivo.

Input: Uma rede com fonte *a*, destino *z*, capacidade *C*, vértices *a=v0,..,vn=z* e o número de vértices *n*.

Output: Um fluxo máximo *F*

O algoritmo está no anexo (2215_aula5_anexo.doc)

5.1 Algoritmo do fluxo máximo

Início:
vértices: v
v: a,b,c,d,e,z (a:fonte, z: destino)
rótulo de v é (predecessor(v),val(v))

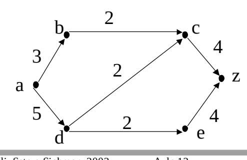


Fig. 5.9: início

5.1 Algoritmo do fluxo máximo

Iteração 1:inicia predecessor(v_i), e val(i)para i de *0* a *n* (todos os vértices
rótulo de v_i :
(predecessor(v_i),val(v_i))
val(v_j):incremento possível
val(a)= ∝
U={a}

predecessor(a)= -	val(a)= ∝
predecessor(b)=null	val(b)=null
predecessor(c)=null	val(c)=null
predecessor(d)=null	val(d)=null
predecessor(e)=null	val(e)=null
predecessor(z)=null	val(z)=null

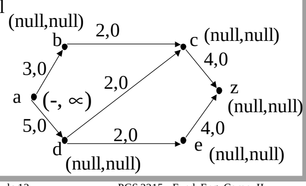


Fig.5.10 :início iteração 1

6

5.1 Algoritmo do fluxo máximo

Iteração 1: próxima escolha de v em U
U={a}
escolhido: v=a
$val(v_j)$:incremento possível
Δ=val(v)
$val(v_j)=\min\{\Delta, C_{vj} - F_{vj}\}$
por exemplo:
val(b)= min{∝,3-0=3}
U={b,d}

predecessor(a)= -
predecessor(b)= a
predecessor(c)=null
predecessor(d)= a
predecessor(e)=null
predecessor(z)=null

val(a)= ∝
val(b)=3
val(c)=null
val(d)=5
val(e)=null
val(z)=null

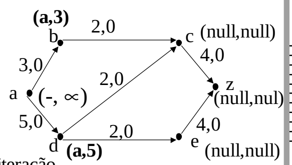


Fig.5.11:Iteração: após o 1a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 1: próxima escolha de v em u
U={b,d}
escolhido: b
Δ=val(b)=3
U={c,d}

predecessor(a)= -
predecessor(b)= a
predecessor(c)=b
predecessor(d)= a
predecessor(e)=null
predecessor(z)=null

val(a)= ∝
val(b)=3
val(c)=2
val(d)=5
val(e)=null
val(z)=null

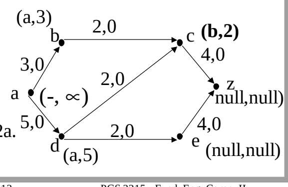


Fig.5.12:Iteração1: após a 2a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 1: próxima escolha de v em u
U={c,d} escolhido: c
Δ=val(c)= 2
U={d,z}

predecessor(a)= -
predecessor(b)= a
predecessor(c)=b
predecessor(d)= a
predecessor(e)=null
predecessor(z)=c

val(a)= ∝
val(b)=3
val(c)=2
val(d)=5
val(e)=null
val(z)=2

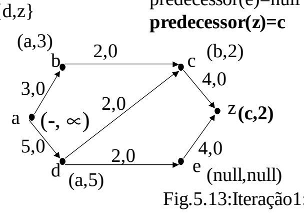
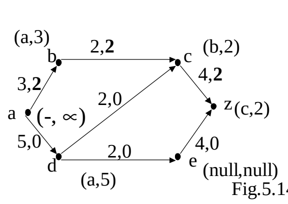


Fig.5.13:Iteração1: após a 3a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 1:z ≠ null
atualiza fluxo (Δ = val(z) =2)
Fluxo=fluxo antigo + Δ : se (i,j) propriamente orientada
Fluxo= fluxo antigo + Δ : se (i,j) não é propriamente orientada

P={a,b,c,z}



F_{cz}=0+2=2
F_{bc}=0+2=2
F_{ab}=0+2=2

Fig.5.14:Iteração1: atualiza fluxo

5.1 Algoritmo do fluxo máximo

Iteração 2:inicia predecessor(vi) e val(vi), para i de 0 a n (todos os vértices)

predecessor(a)= -	val(a)= ∝
predecessor(b)=null	val(a)=null
predecessor(c)=null	val(a)=null
predecessor(d)=null	val(a)=null
predecessor(e)=null	val(a)=null
predecessor(z)=null	val(a)=null

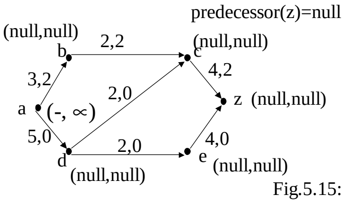


Fig.5.15: Iteração2: início

5.1 Algoritmo do fluxo máximo

Iteração 2: próxima escolha de v em u

U={a}

escolhido:v= a

Δ=val(a)= ∝

val(v_j)=min{Δ,C_{vj}-F_{vj}}

U={b,d}

predecessor(a)= -	val(a)= ∝
predecessor(b)=a	**val(b)=1**
predecessor(c)=null	val(c)=null
predecessor(d)=a	**val(d)=5**
predecessor(e)=null	val(e)=null
predecessor(z)=null	val(z)=null

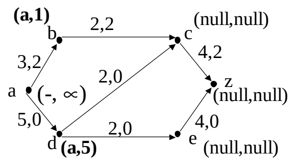


Fig.5.16:Iteração2: após a 1a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 2: próxima escolha de v em u

U={b,d} escolhido:v= b

Δ=val(b)= 1

val(v_j)=min{Δ,C_{vj}-F_{vj}}

C_{bc}=F_{bc} então c não é incluído em U.

U={d}

predecessor(a)= -	val(a)= ∝
predecessor(b)=a	val(b)=1
predecessor(c)=null	val(c)=null
predecessor(d)=b	val(d)=5
predecessor(e)=null	val(e)=null
predecessor(z)=null	val(z)=null

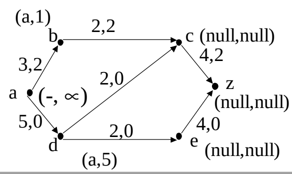


Fig.5.17: Iteração2: após a 2a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 2: próxima escolha de v em u

U={d} escolhido:v= d

Δ=val(d)= 5

val(v_j)=min{Δ,C_{vj}-F_{vj}}

U={c,e}

predecessor(a)= -	val(a)= ∝
predecessor(b)=a	val(b)=1
predecessor(c)=d	**val(c)=2**
predecessor(d)=b	val(d)=5
predecessor(e)=d	**val(e)=2**
predecessor(z)=null	val(z)=null

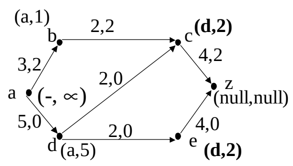


Fig.5.18:Iteração2: após a 3a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 2: próxima escolha de v em u

U={c,e} escolhido:v= c

Δ=val(c)= 2

val(v_j)=min{Δ,C_{vj}-F_{vj}}

U={e,z}

predecessor(a)= -	val(a)= $\propto$
predecessor(b)=a	val(b)=1
predecessor(c)=d	val(c)=2
predecessor(d)=b	val(d)=5
predecessor(e)=d	val(e)=2
predecessor(z)=c	**val(z)=2**

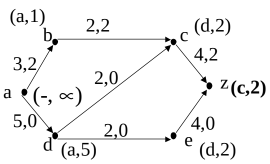


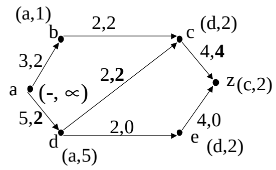
Fig.5.19:Iteração2: após a 4a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 2:z $\neq$ null

atualiza fluxo (Δ = val(z) =2)

P={a,d,c,z}



F_{ad}=0+2=2

F_{dc}=0+2=2

F_{cb}=2+2=4

Fig.5.20: Iteração2: atualiza fluxo

5.1 Algoritmo do fluxo máximo

Iteração 3:inicia predecessor(vi) e val(vi), para i de 0 a n (todos os vértices)

predecessor(a)= -	val(a)= $\propto$
predecessor(b)=null	val(a)=null
predecessor(c)=null	val(a)=null
predecessor(d)=null	val(a)=null
predecessor(e)=null	val(a)=null
predecessor(z)=null	val(a)=null

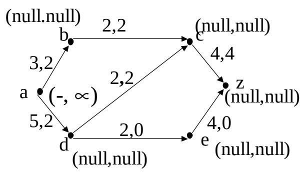


Fig.5.21:Iteração3: início

5.1 Algoritmo do fluxo máximo

Iteração 3: próxima escolha de v em u

U={a}

escolhido:v= a

Δ=val(a)= $\propto$

val(v_j)=min{Δ,C_{vj}-F_{vj}}

U={b,d}

predecessor(a)= -	val(a)= $\propto$
predecessor(b)=a	**val(b)=1**
predecessor(c)=null	val(c)=null
predecessor(d)=a	**val(d)=3**
predecessor(e)=null	val(e)=null
predecessor(z)=null	val(z)=null

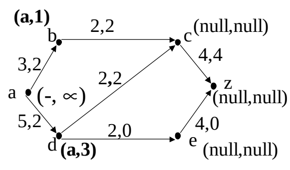


Fig.5.22:Iteração3: após a 1a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 3: próxima escolha de v em u

U={b,d}

escolhido:v= b

Δ=val(b)= 2

val(v_j)=min{Δ,C_{vj}-F_{vj}}

C(b,c)=F(b,c) então c não é incluído em U.

U={d}

predecessor(a)= -	val(a)= ∝
predecessor(b)=a	val(b)=1
predecessor(c)=null	val(c)=null
predecessor(d)=a	val(d)=3
predecessor(e)=null	val(e)=null
predecessor(z)=null	val(z)=null

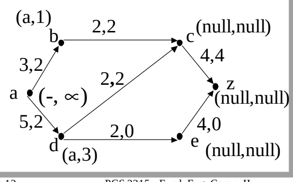


Fig.5.23:Iteração3: após a 2a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 3: próxima escolha de v em U

U={d}

escolhido:v= d

Δ=val(d)= 3

val(v_j)=min{Δ,C_{vj}-F_{vj}}

C_{dc}=F_{dc} então c não é incluído em U.

U={e}

predecessor(a)= -	val(a)= ∝
predecessor(b)=a	val(b)=1
predecessor(c)=null	val(c)=null
predecessor(d)=a	val(d)=3
predecessor(e)=d	**val(e)=2**
predecessor(z)=null	val(z)=null

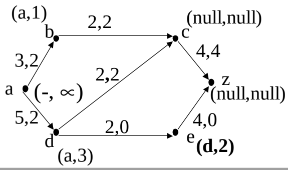


Fig.5.24:Iteração3: após a 3a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 3: próxima escolha de v em u

U={e}

escolhido:v= e

Δ=val(e)= 2

val(v_j)=min{Δ,C_{vj}-F_{vj}}

U={z}

predecessor(a)= -	val(a)= ∝
predecessor(b)=a	val(b)=1
predecessor(c)=null	val(c)=null
predecessor(d)=a	val(d)=3
predecessor(e)=d	val(e)=2
predecessor(z)=e	**val(z)=2**

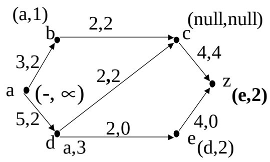
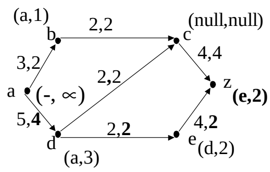


Fig.5.25:Iteração3: após a 4a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 3:z ≠ null

atualiza fluxo (Δ = val(z) =2)



P={a,d,e,z}

F_{ad}=2+2=4

F_{de}=0+2=2

F_{ez}=0+2=4

Fig.5.26:Iteração3: atualiza fluxo

5.1 Algoritmo do fluxo máximo

Iteração 4:inicia predecessor(vi) e val(vi), para i de 0 a n (todos os vértices)

predecessor(a)= -	val(a)= $\propto$
predecessor(b)=null	val(a)=null
predecessor(c)=null	val(a)=null
predecessor(d)=null	val(a)=null
predecessor(e)=null	val(a)=null
predecessor(z)=null	val(a)=null

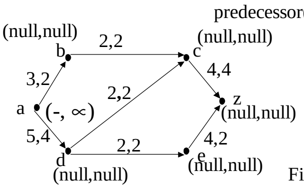


Fig.5.27:Iteração4: início

5.1 Algoritmo do fluxo máximo

Iteração 4: próxima escolha de v em u

U={a}

escolhido:v= a

Δ=val(a)= :

val(v_j)=min{Δ,C_{vj}-F_{vj}}

U={b,d}

predecessor(a)= -	val(a)= $\propto$
predecessor(b)=a	**val(b)=1**
predecessor(c)=null	val(c)=null
predecessor(d)=a	**val(d)=1**
predecessor(e)=null	val(e)=null
predecessor(z)=null	val(z)=null

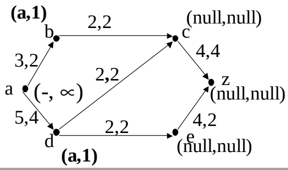


Fig.5.28:Iteração4: após a 1a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 4: próxima escolha de v em u

U={b,d}

escolhido:v= b

Δ=val(b)= 1

val(v_j)=min{Δ,C_{vj}-F_{vj}}

F_{bc}=C_{bc} então c não é incluído em U:

U={d}

predecessor(a)= -	val(a)= $\propto$
predecessor(b)=a	val(b)=1
predecessor(c)=null	val(c)=null
predecessor(d)=a	val(d)=1
predecessor(e)=null	val(e)=null
predecessor(z)=null	val(z)=null

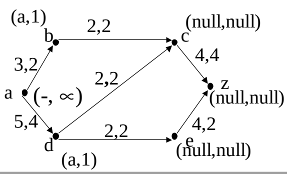


Fig.5.29:Iteração4: após a 2a. iteração

5.1 Algoritmo do fluxo máximo

Iteração 4: próxima escolha de v em u

U={d}

escolhido:v= d

Δ=val(d)= 1

val(v_j)=min{Δ,C_{vj}-F_{vj}}

F_{dc}=C_{dc} e F_{de}=C_{de} então c,e não são incluídos em U.

U=ϕ

Portanto: término

Tem-se o fluxo máximo

predecessor(a)= -	val(a)= $\propto$
predecessor(b)=a	val(b)=1
predecessor(c)=null	val(c)=null
predecessor(d)=a	val(d)=1
predecessor(e)=null	val(e)=null
predecessor(z)=null	val(z)=null

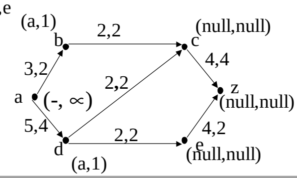


Fig.5.30:Iteração4: após a 3a. iteração e término

Bibliografia

[1] Johnsonbaugh, R. Discrete Mathematics. Prentice Hall International, London, UK, 4th. Ed. 1997. Cap. 8.

12