Chapter 7 Dynamic Programming

Ağ Üzerinde Akım (Network Flow)

- Minimum Kesme Problemi (Minimum Cut problem)
- Maksimum Akış Problemi (Maximum Flow problem)

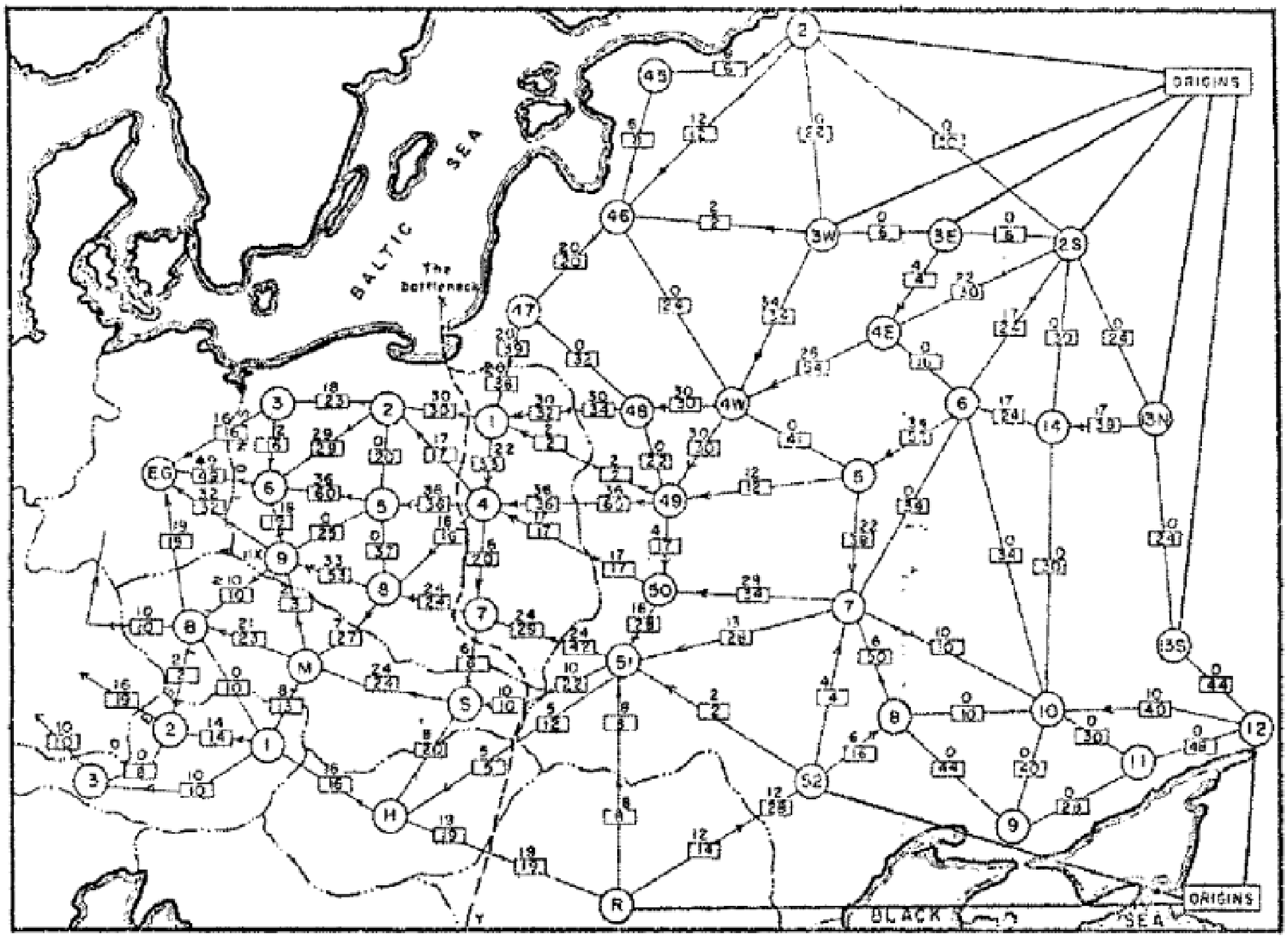
Maksimum Akış ve Minimum Kesim Problemi, graf teorisinin klasik optimizasyon problemleridir ve genellikle ulaşım, iletişim ve kaynakların çeşitli sistemlerdeki akışını modellerler.

ÖZET:

- Minimum Kesim probleminin amacı akış ağında kaynağı ve hedefi ayıran minimum kapasiteli kesimi bulmaktır.

- Maksimum Akış probleminin amacı, belirtilen bir kaynak düğümden belirtilen bir hedef düğüme kadar gönderilebilecek maksimum akış miktarını bulmaktır.

Soviet Rail Network, 1955



Reference: *On the history of the transportation and maximum flow problems.*
Alexander Schrijver in Math Programming, 91: 3, 2002.

Minimum Kesme Problemi (Minimum Cut problem)

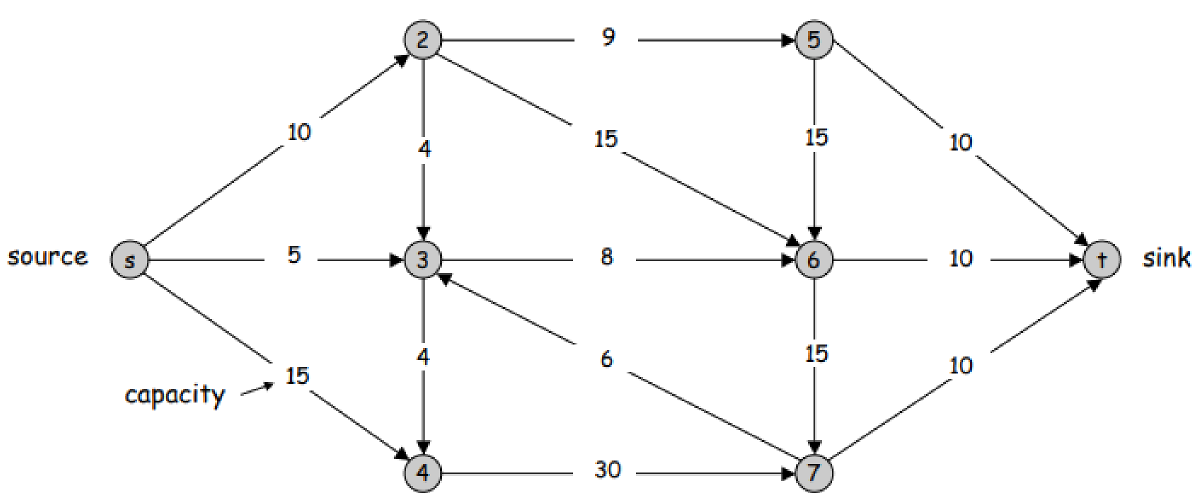
Tanım1) Akış ağı (Network flow)

- Kaynağın bir sistem içindeki hareketinin graf ile görselleştirilmesidir .
- G = (V, E) yönlü graftır.
- İki ayırt edici düğümü vardır : s = kaynak(source), t = hedef(sink).

  s: akışın başladığı tek düğüm

  t: varılması amaçlanan tek bir düğüm
- c(e) : e kenarının kapasitesi - kenarın taşıyabileceği maksimum akış miktarı

➢ s kaynak düğümüne giren hiçbir kenar yoktur
➢ t hedef düğümünden çıkan hiçbir kenar yoktur.
➢ İç düğümlere en az bir gelen kenar ve en az bir giden kenar bulunmaktadır.
➢ Tüm kapasitelerin tamsayıdır.



Tanım2) s-t kesimi (s-t cut)

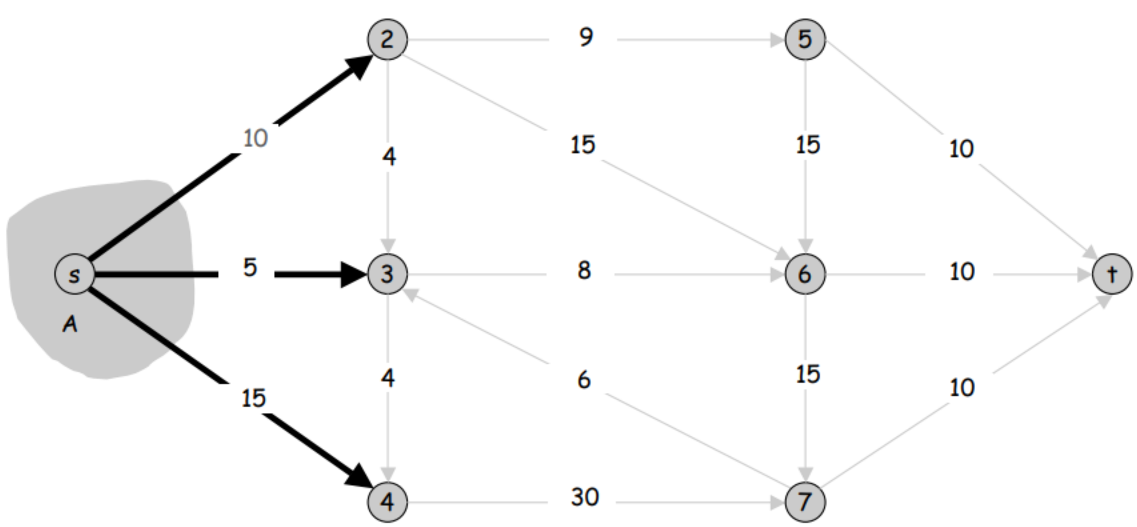
V'nin $s \in A$ ve $t \in B$ ile ayrılan bir (A, B) bölümüdür.

Tanım3) cut (A, B) 'nin kapasitesi

$$cap(A, B) = \sum_{e \text{ out of } A} c(e)$$

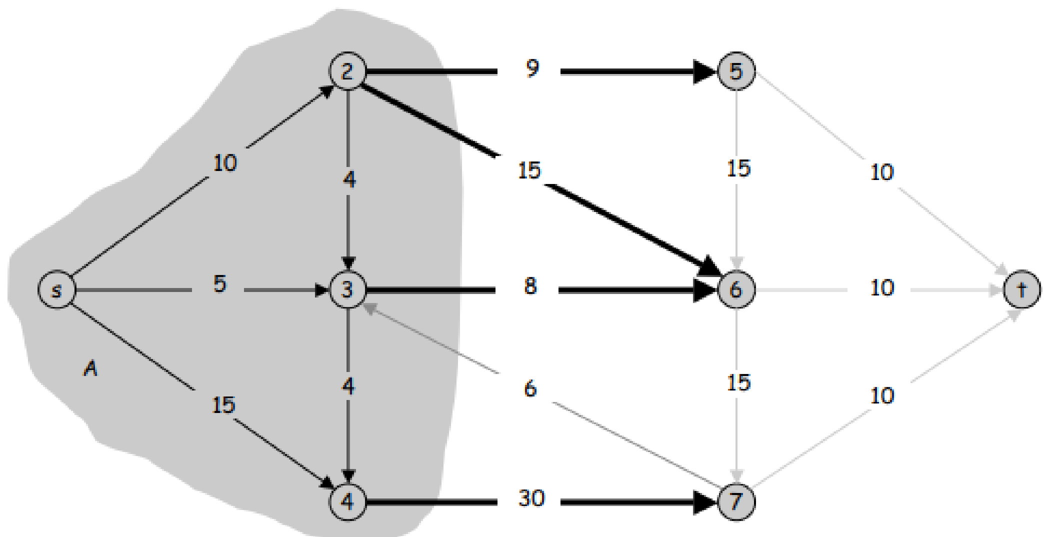
Örnek1

Cut(A,B) → Cap(A,B)= $\boxed{}$



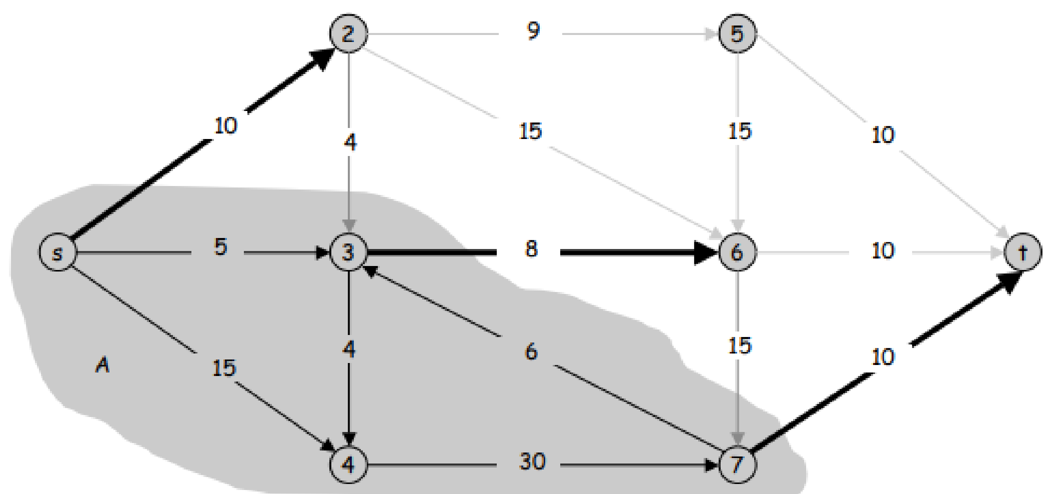
Örnek2

Cut(A,B) → Cap(A,B)=



Örnek3

Cut(A,B) → Cap(A,B)=



Min s-t cut problemi : Minimum kapasiteye sahip s-t kesimi bulunuz.

NOT: Matematiksel optimizasyon teorisinde **dualite** , optimizasyon problemlerinin iki perspektiften, birincil problem veya ikili problemden görülebileceği ilkesidir. Eğer primal bir minimizasyon problemi ise dual bir maksimizasyon problemidir.

Maksimum Akış Problemi (Maximum Flow problem)

Tanım1: s-t Akış (s- t flow)

Bir s-t akışı aşağıdakileri özellikleri olan bir fonksiyondur:

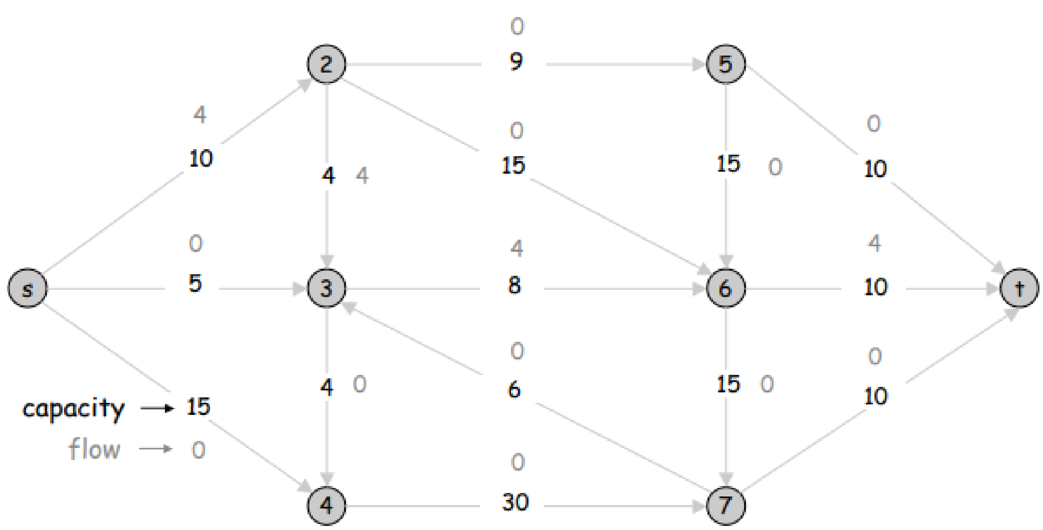
Kapasite: Her $e \in E$ için	$0 \le f(e) \le c(e)$
Korunum: Her $v \in V - \{s, t\}$ için	$\sum_{e \text{ in to } v} f(e) = \sum_{e \text{ out of } v} f(e)$

Tanım2: Akış f 'nin değeri (value of f flow)

$$v(f) = \sum_{e \text{ out of } s} f(e)$$

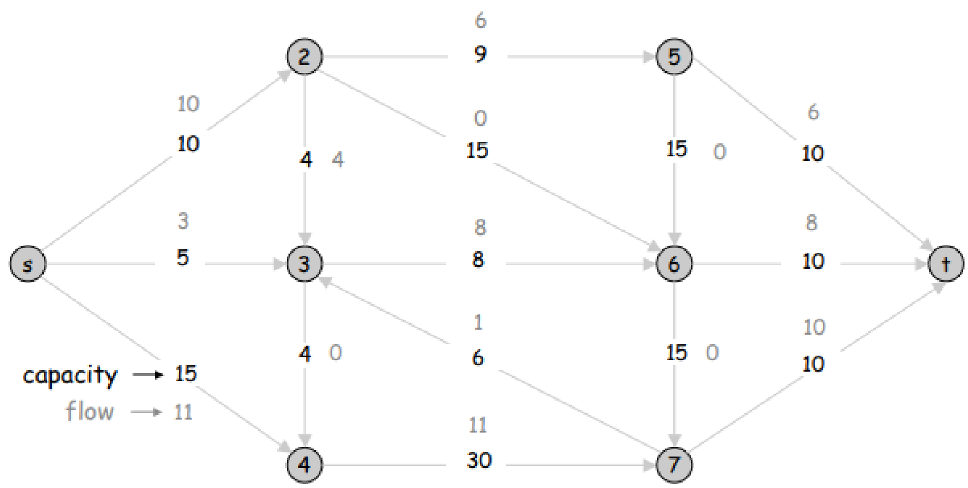
Örnek1

Flow f → v(f) =



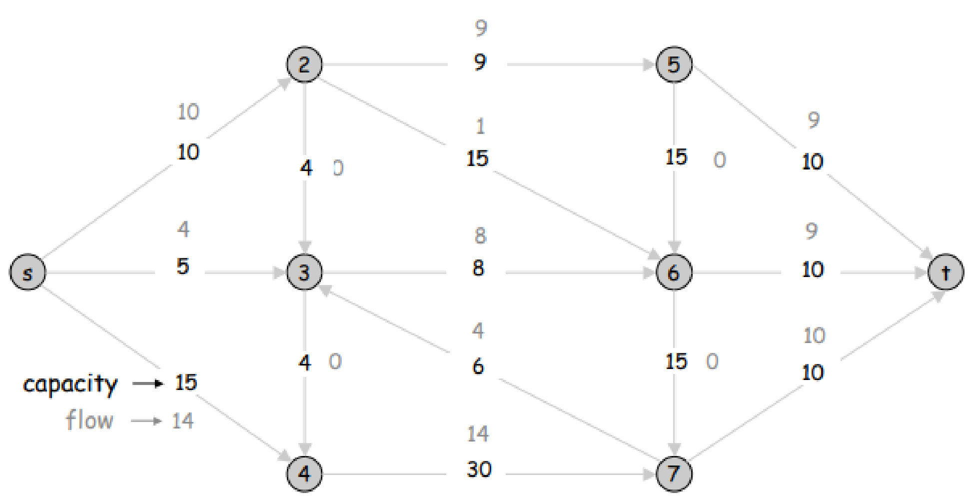
Örnek2

Flow f → v(f) =



Örnek3

Flow f → v(f) =



Maksimum akış problemi: Maksimum değerdeki s-t akışını bulun.

ÇÖZÜM için bir önerme !!!

Önerme : Akış değeri lemması:

f herhangi bir akış olsun ve (A, B) herhangi bir s-t kesimi olsun.

$$\sum_{e \text{ out of } A} f(e) \; - \sum_{e \text{ in to } A} f(e) \; = \; v(f)$$

Kesim boyunca gönderilen net akış, akış f 'nin değerine(s'den ayrılan miktara) eşittir.

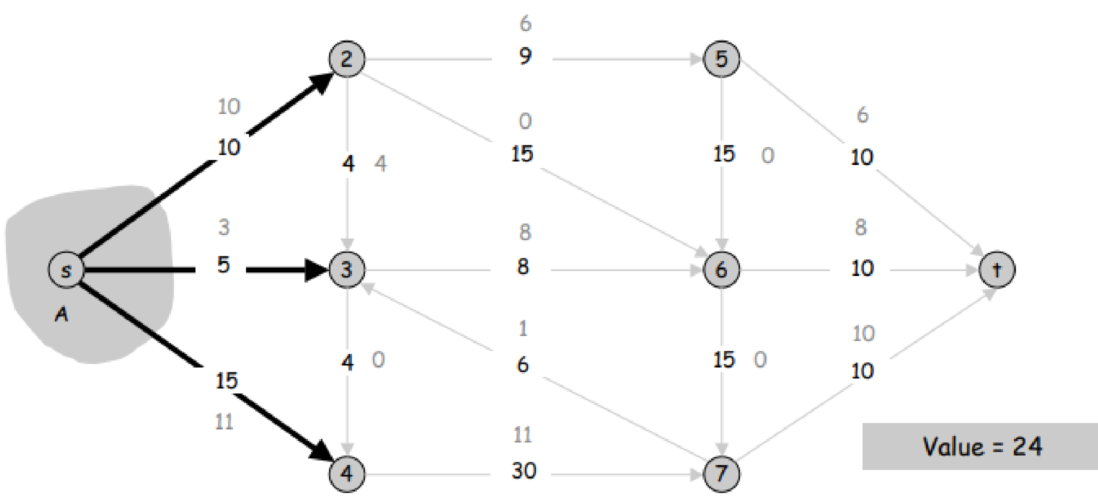
E out of A : A'dan çıkan kenarlar

E in to A : A' ya giren kenarlar

Örnek1

$$\sum_{e \text{ out of } A} f(e) \; - \sum_{e \text{ in to } A} f(e) \; = \; v(f)$$

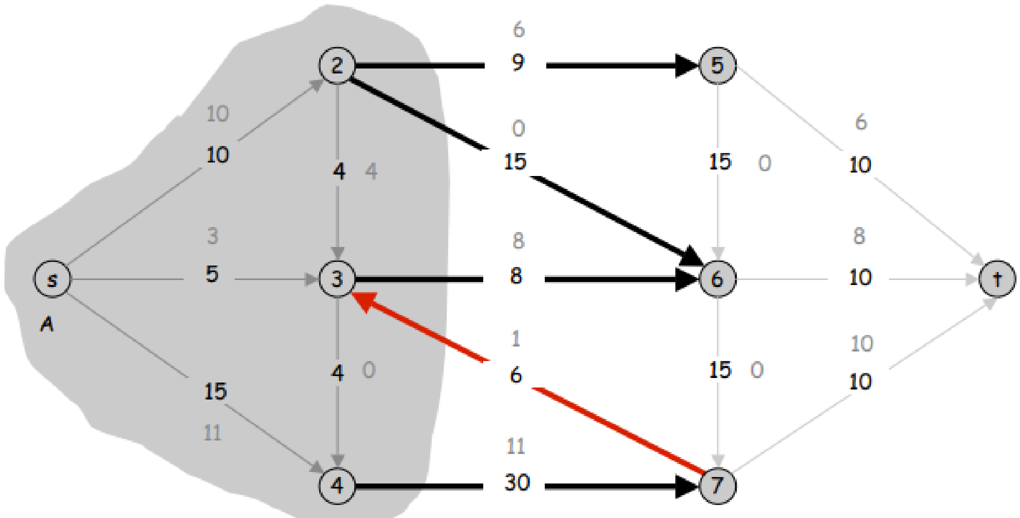
___ - ___ =24



Örnek 2

$$\sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) = v(f)$$

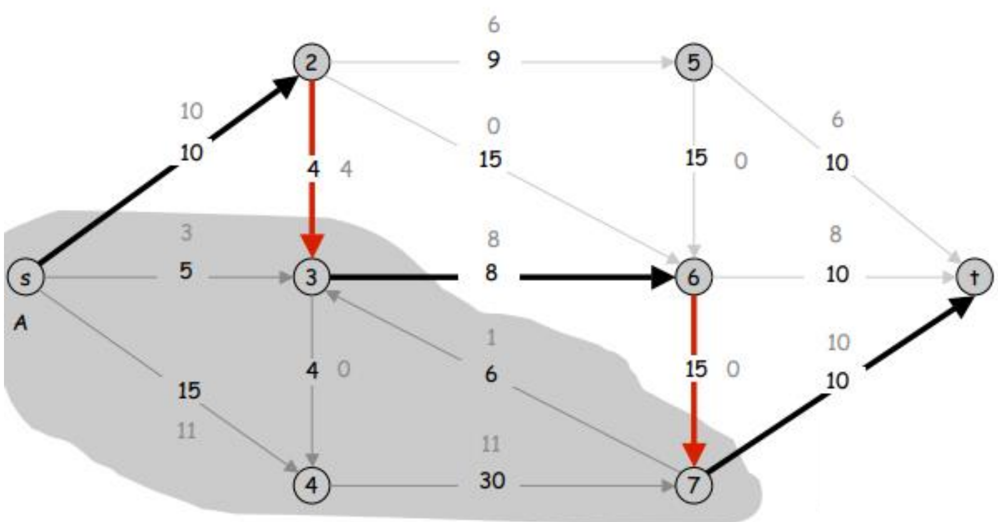
___ - ___ =24



Örnek 3

$$\sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) = v(f)$$

___ - ___ =24



Önerme'nin İspatı : Akış değeri lemması'nın ispatı

Flow value lemma. Let f be any flow, and let (A, B) be any s-t cut. Then

$$\sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) = v(f).$$

Pf.

$$v(f) = \sum_{e \text{ out of } s} f(e) \qquad \boxed{\leftarrow \text{Tanımdan}}$$

by flow conservation, all terms except v = s are 0 →

$$= \sum_{v \in A} \left(\sum_{e \text{ out of } v} f(e) - \sum_{e \text{ in to } v} f(e) \right)$$

$$= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e).$$

CUT ve FLOW arasındaki dualite:

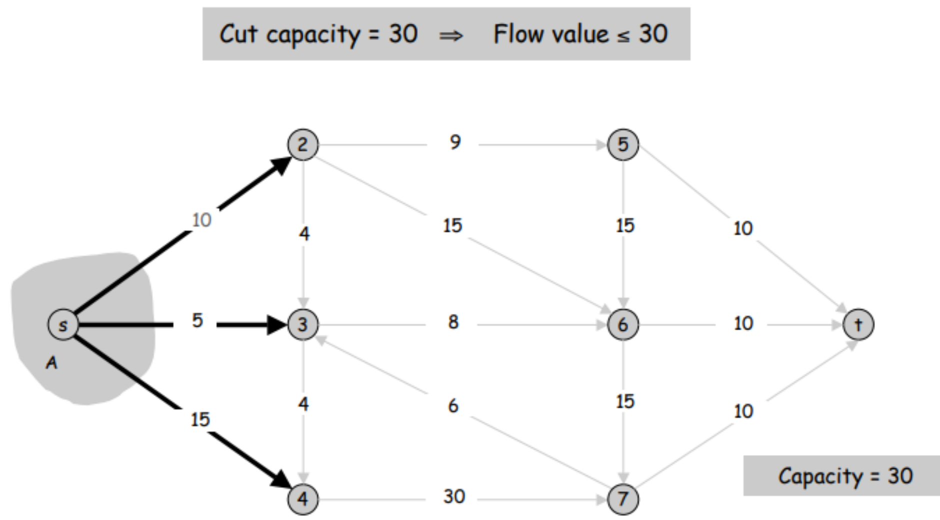
f herhangi bir akış olsun ve (A, B) herhangi bir s-t kesimi olsun.

Bu durumda akışın değeri en fazla kesme kapasitesi kadardır.

Akış Değeri ≤ Kesim Kapasitesi

v(f) ≤ cap(A, B)

Bu ilke, ağdaki akış ve kesimler arasındaki ilişkileri anlamak için temel bir prensiptir. Ağdaki kesimlerin kapasitelerine dayalı olarak akış değerine bir üst sınır belirler.



v(f) ≤ cap(A, B) nin ispatı

Let f be any flow. Then, for any s-t cut (A, B) we have

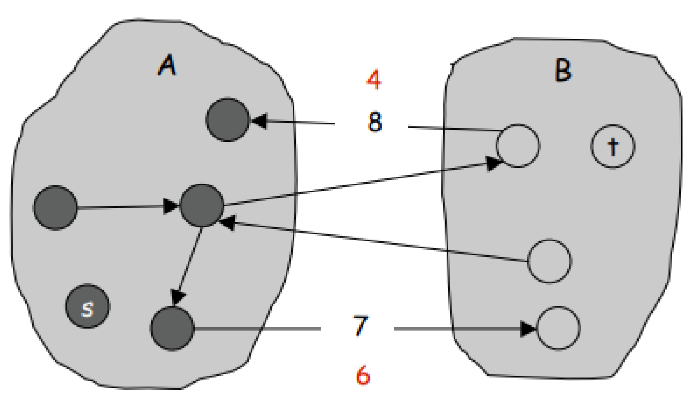
$v(f) \leq cap(A, B)$.

Pf.

$$v(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e)$$

$$\leq \sum_{e \text{ out of } A} f(e)$$

$$\leq \sum_{e \text{ out of } A} c(e) = \operatorname{cap}(A,B)$$



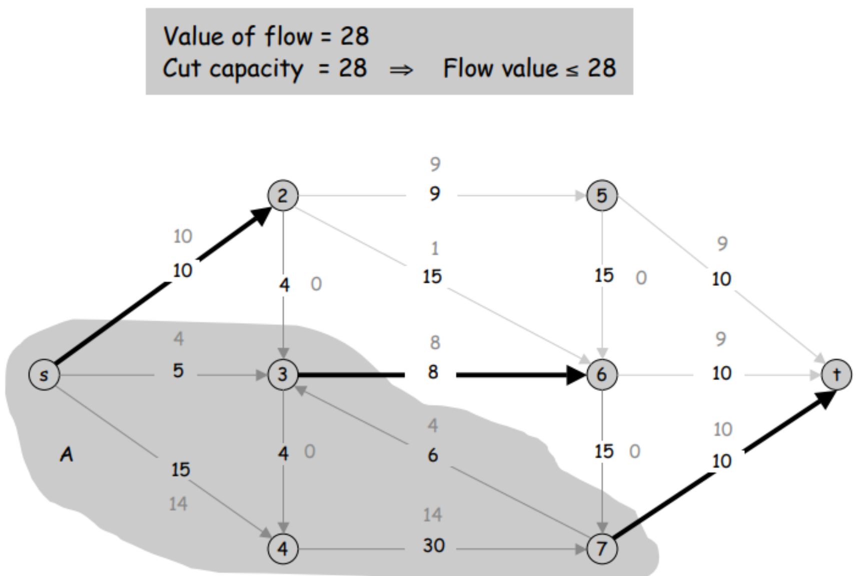
$$v(f) \leq \operatorname{cap}(A,B)$$

Sonuç

f herhangi bir akış ve (A, B) herhangi bir kesim olsun.

v(f) = cap(A, B)

ise **f maksimum akıştır** ve **(A, B) minimum kesimdir.**

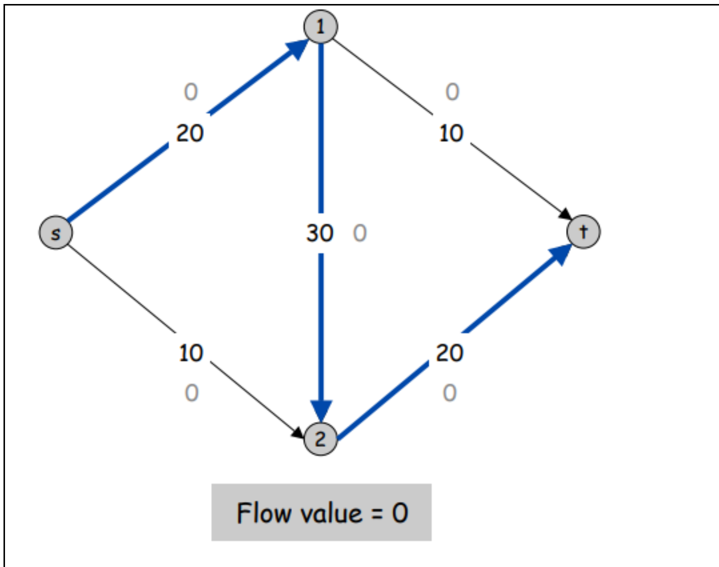


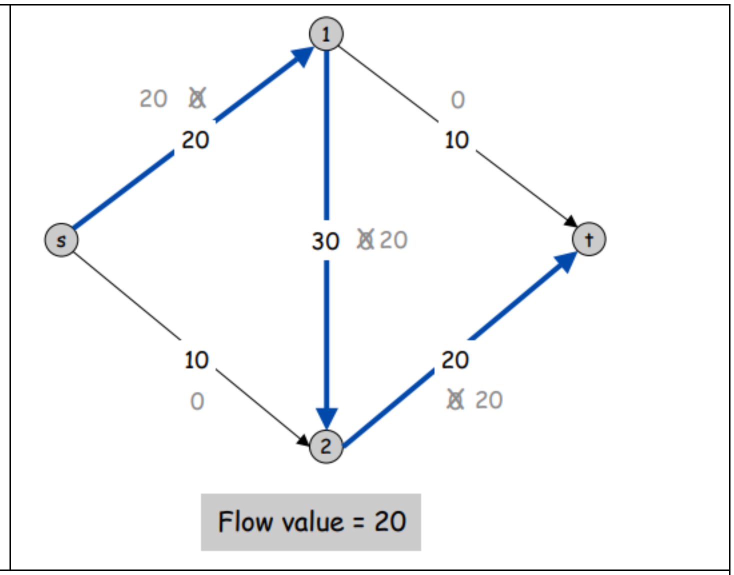
MAKSİMUM AKIŞ PROBLEMİ'NİN ÇÖZÜMÜ İÇİN GREEDY YAKLAŞIMI

Açgözlü algoritma yaklaşımı ve sonuçları

- Tüm $e \in E$ kenarları için $f(e) = 0$ ile başlayın.
- Her bir kenarın $f(e) < c(e)$ olduğu bir "P" s-t yolu bulun.
- P yolu boyunca akışı artırın.
- Adımları kapasiteler uygun olduğu sürece tekrarlayın.

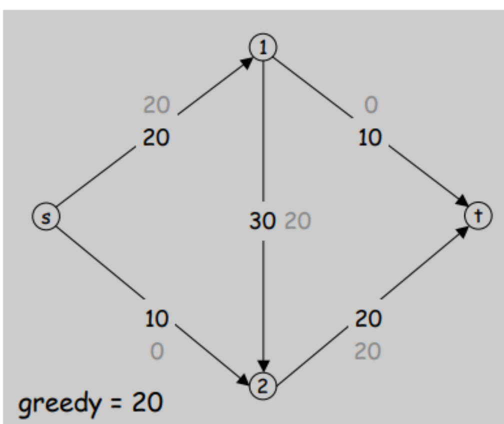
ADIM 1	ADIM2

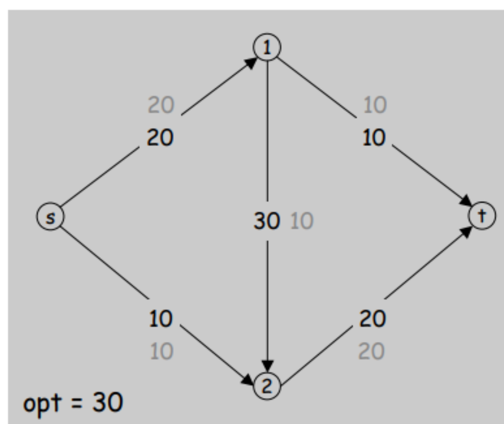




ADIM3: Algoritmaya göre başka akış yok.

Greedy algoritması max akış(optimum) değerini buldu mu? HAYIR !!!





Ford-Fulkerson Algoritması

Bir akış ağındaki maksimum akışı ve minimum cut'ı hesaplamak için kullanılan bir iteratif yöntemdir.

L.R. Ford, Jr. ve D.R. Fulkerson tarafından 1956 yılında tanıtılmıştır.

Algoritma:

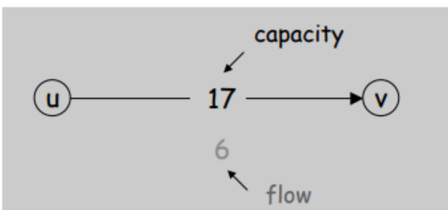
Başlangıçta uygun bir akış ile başlar ve artırma yolları bulunamayana kadar akışı artırmaya devam eder.

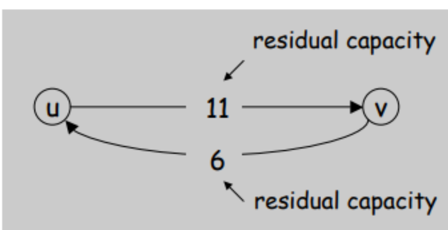
Artırma yollarını arama **residual graf** üzerinde gerçekleştirilir ve akış bu yollar boyunca artırılır.

Algoritma **residual grafı** iteratif olarak günceller ve artık artırma yolları bulunamadığında maksimum akışa ulaşılır.

Tanım : Residual Graf

Ağdaki kenarların belli bir akış miktarından sonra kalan kapasitesini veya "artan kapasiteyi" temsil eden bir grafiği ifade eder.





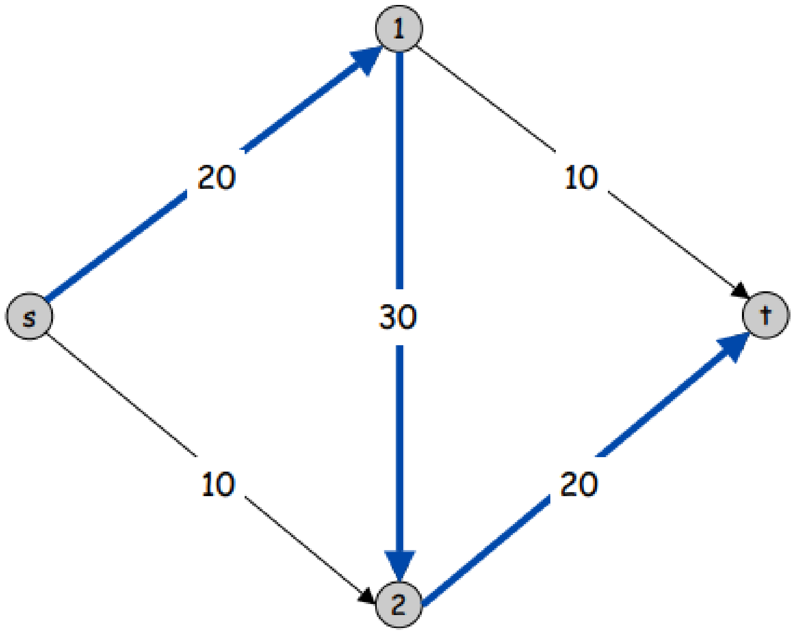
Orijinal kenar için e = (u, v) ∈ E.
Akış: f(e), kapasite c(e) dir.

Residual edge - Artık kenar:
e = (u, v) ise e^R = (v, u) dir.

Residual kapasite tanımı:

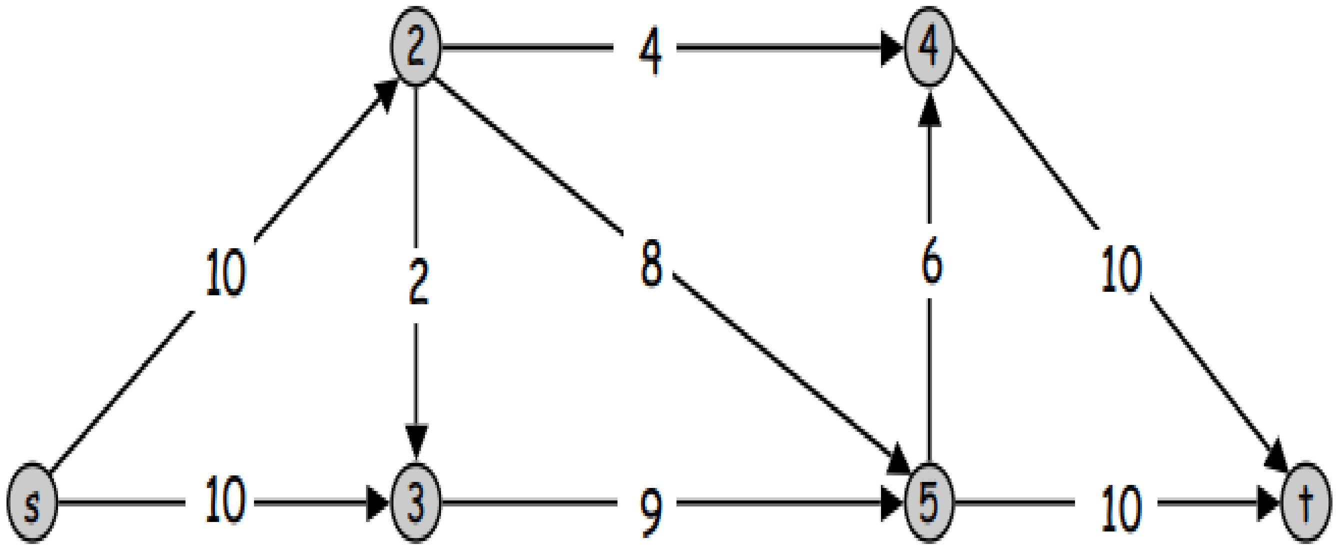
$$c_f(e) = \begin{cases} c(e) - f(e) & \text{if } e \in E \\ f(e) & \text{if } e^R \in E \end{cases}$$

Greedy algoritması ile çözdüğümüz soruya **Ford-Fulkerson Algoritması** ile tekrar bakalım.



Akış	kapasitesi
s-1-2-t	
s-2-1-t	
toplam(max) kapasite	

ÖRNEK: **Ford-Fulkerson Algoritması**



Cevap :

Maksimum akış : 19

Min cut : 19

Ford-Fulkerson algoritmasının toplam çalışma süresi:

O(?)

(kapasite değerlerinin tam sayı olduğu varsayımı ile)