2017-2018 Bahar Yarıyılı

Balıkesir Üniversitesi
Endüstri Mühendisliği Bölümü

EMM3208
Optimizasyon Teknikleri

card(), ord(), Küme Elemanlarının Komşulukları, Koşullu İfadeler, $ Operatörü, sameas() ve diag(), Dosyaya Yazdırma ve Dosyadan Okuma, Program Akış Kontrolü, Subsets, Dynamic Sets, Multidimensional Sets

8

Yrd. Doç. Dr. İbrahim Küçükkoç
http://ikucukkoc.baun.edu.tr
ikucukkoc@balikesir.edu.tr

18.04.2018

Kaynak: Dhazn Gillig and Bruce A. McCarl, Introduction to GAMS: Summation Notation with GAMS, Department of Agricultural Economics, Texas A&M University (*from Appendix A in McCarl and Spreen's text book on* *http://ageco.tamu.edu/faculty/mccarl/mccspr/newa1.pdf*)

card() komutu - (Kümedeki Eleman Sayısı)

`Card(x)` komutu x kümesindeki eleman sayısını verir.

- Example:

```
set c      'countries'  / jamaica, haiti, guyana, brazil /;
scalar nc 'number of countries;
  nc = card(c);
```

ord() komutu

Bir kümedeki bir elemanın o küme içindeki sıra numarasına ihtiyaç olursa “ord(x)” fonksiyonu kullanılabilir.

Örneğin;

$$y = \sum_i i * x_i$$

ifadesi GAMS’te aşağıdaki gibi kodlanabilir:

```
y =e= sum(i, ord(i)*X(i))
```

- Example:

```
Set t / 1*24 /;
Parameter hour(t);
hour(t) = ord(t);
```

Küme Elemanlarının Komşulukları (Önceki ve Sonraki Elemanlara Erişim)

Sıralı kümelerde (*ordered sets*), küme elemanının komşuluğundaki öğelere erişmek mümkündür.

Söz dizimi: **`setelement` ± `n`**

Eğer erişilmeye çalışılan öğe için kümede bir karşılık yoksa veya küme boyutlarını aşıyorsa, yani "**setelement + n > card(set)**" ise dönecek değer sıfırdır.

Örneğin;

```
Set t /1*24/;

Variables Stok(t), UretimMik(t), Satis(t);

Equation Stok;

StokHesaplama(t).. Stok(t) =e= Stok(t-1)+UretimMik(t)-Satis(t);
```

Koşullu İfadeler: Boolean (İkili) Operatörler

Sayısal ve mantıksal operatörler aşağıda verildiği gibi kullanılır.

İkili operatörlerin işlenmesi sonucu true dönerse 1'e, false dönerse 0'a karşılık gelmektedir.

Sayısal operatörler

GAMS	Anlamı
lt	<
le	<=
eq	=
ne	<>
ge	>=
gt	>

Mantıksal operatörler

GAMS	Anlamı
not	Değil
and	Ve
or	Veya
xor	Farklı veya

a	*b*	*a and b*	*a or b*	*a xor b*	*not a*
0	0	0	0	0	1
0	non-zero	0	1	1	1
non-zero	0	0	1	1	0
non-zero	non-zero	1	1	0	0

Koşullu İfadeler: $ Operatörü

- **$** operatörü gerekli koşulların uygulanmasını mümkün kılar.
- **$(koşul)** komutu "eğer koşul doğru ise" olarak düşünülebilir.
- Örnek:

  Eğer b>1.5 ise a=2 olsun -> `a$(b>1.5)=2;`

  Eğer b>1.5 ise a=2 olsun, aksi halde a=0 olsun -> `a=2$(b>1.5);`

- **$** eşitliğin solunda ise, koşul sağlanmazsa atama yok
- **$** eşitliğin sağında ise, her zaman atama yapılır. Fakat koşul sağlanmazsa $ ile belirtilen ifade 0 olur.
- **$** ifadesi tanımlamalar için kullanılamaz.

Koşullu İfadeler: $ Operatörü

Kısıtları bazı koşullara göre süzmek için yine $ işareti kullanılabilir.

Örnek:

Aynı iki indise sahip x_{ij} ve y_{ij} değişkenleri olsun. Sadece i'nin j'den küçük olduğu durumlar için, y_{ij}'nin x_{ij}'den büyük olması gerektiğini, diğer durumlar için bir kısıt olmadığını varsayalım. Bu kısıtı matematiksel olarak aşağıdaki gibi yazabiliriz:

$$y_{ij} \geq x_{ij} \quad \forall i < j$$

Bu matematiksel ifadenin GAMS'teki karşılığı ise aşağıdaki gibi olur:

```
Kisit1(i,j)$(ord(i)<ord(j)).. y(i,j) =g= x(i,j);
```

$ Operatörü Uygulamaları

Filtering in indexed operations:

```
parameter sorbetbalance;
set ice; set sorbet(ice);
sorbetbalance = sum(ice$sorbet(ice), price(ice)*purchase.l(ice));
sorbetbalance = sum(sorbet(ice), price(ice) * purchase.l(ice));
```

Conditioned indexed operations:

```
rho = sum(i$(sig(i) ne 0), 1/sig(i) - 1);
```

Conditioned equations:

```
Equation satdemand(ice);
satdemand(ice)$sorbet(ice).. purchase(ice) =g= demand(ice);

balance(t)$(ord(t)>1)..
                level(t) =e= level(t-1) + inflow(t) - outflow(t);
```

Existence of variables in constraints:

```
variables x, y;   parameter A;   equation e;
e.. x + y$(A>2) =e= A;
```

$ Operatörü İçinde Değişken KULLANILAMAZ!

- The following DOES NOT WORK:

```
binary variable x;  variable y;   equation e1, e2;
e1$(x = 1).. y          =l= 100;
e2         .. y$(x = 1) =l= 100;
```

- The value of x is decided by the solver, not by the GAMS.
- However, GAMS has to evaluate $-operators when assembling an instance in the Solve statement.
- Instead, you have to reformulate:

```
e.. y =l= 100 * x + y.up * (1-x);
```

That is: $x=1 \Rightarrow y \leq 100$; $x=0 \Rightarrow y \leq y.up$.

sameas ve diag

`sameas (i1, i2)` komutu, **`i1`** ve **`i2'`** nin birbirine eşit olma durumunu kontrol eder. Eşit olması durumunda true, olmaması durumunda ise false döndürür.

Benzer şekilde, **`i1'`** in verilen "**değer**"e eşit olma durumunu kontrol etmek için de **`sameas (i1, "değer")`** komutu kullanılır.

`diag` komutu da tıpkı **`sameas`** gibi kullanılır, ancak eşitlik durumunda **1** değerini, eşitsizlik durumunda ise **0** değerini döndürür.

Örnek:

```
sum((i,j)$(sameas(i,j)), C(i,j)
```

Burada toplama işlemi sadece i'nin j'ye eşit olduğu **`C(i,j)`** değerleri için yapılır.

sum((i,j)$(NOT sameas(i,j)), C(i,j)

Komutunda ise toplama işlemi sadece i'nin j'den farklı olduğu **C(i,j)** değerleri için yapılır. Aşağıdaki örnek Chicago'dan tekrar Chicago'ya gönderimi engellemektedir.

```
SETS
Source
 / Chicago
   "San Diego"/

Destination
 / "New York"
   Chicago
   Topeka     / ;
```

```
Costsum ..
  TotalCost
  =E=   SUM((Source,Destination)
       $(not sameas(Source,Destination)),
           Trancost(Source,Destination)
            *Transport(Source,Destination));
SupplyBal(Source)..
 SUM(Destination$(not sameas(Source,Destination)),
       Transport(Source,Destination) )
  =L=   Supply(Source) ;
DemandBal(Destination) ..
  SUM(Source$(not sameas(Source,Destination)),
      Transport(Source,Destination) )
  =G=   Need(Destination) ;
```

```
z(k) = sum(i$sameas(i,'water'), r(i,k));
```

şeklindeki bir komut, sadece i'nin water değerini aldığı "**r(i,k)**" değerlerinin toplamını her bir **k** değeri için **z(k)**'ya eşitlemektedir.

Yani i'nin water değerini almadığı diğer "**r(i,k)**" değerleri için bu kısıt aktif değildir.

sameas ve **diag**, iki kümenin birbirine eşit olup olmadığını sorgulamak için de kullanılabilir.

Örnek:

```
sets ice1 / chocolate, strawberry, cherry, vanilla /
     ice2 / strawberry, cherry, banana /;
scalar ncommon;
ncommon = sum((ice1, ice2), diag(ice1,ice2));
ncommon = sum(sameas(ice1, ice2), 1);
```

Text Dosyasına Yazdırma

- $Echo ve $onEcho - $offEcho komutları text dosyasına yazdırmayı sağlar. Örnek:

```
$Echo "merhaba" > dosya1.txt
```

veya

```
$OnEcho >> dosya1.txt
Yeni bir mesaj daha
$OffEcho
```

- "> dosya1.txt" komutu dosya1.txt isminde yeni bir text dosyası oluşturur. Eğer aynı isimde bir dosya varsa üzerine yazar (override).
- ">> dosya1.txt" komutu ise varolan dosya1.txt isimli text dosyasına metin ekler.
- Hatırlatma: Bu komut derleme zamanı komutudur. Çözüm sonuçlarını dosyaya yazdırmak için kullanılmaz. Sonraki derslerde bunun için put özelliğini kullanmayı öğreneceğiz.

Dosyadan Okuma ($Include Komutu)

- $Include komutu GAMS programına ASCII dosyalardan veri eklemek için kulanılır.
- Sonrasında program, dahil edilen veriyi kulanarak çalışmaya devam eder. Örnek:

```
Table d(i,j) şehirler arasındaki mesafe

$include dist.txt
```

Burada dist.txt dosyası aşağıdaki tabloyu içermektedir:

```
             new-york      chicago      topeka
seattle         2.5           1.7          1.8
san-diego       2.5           1.8          1.4  ;
```

Dosyadan Okuma ($OnDelim Komutu)

- $OnDelim komutu, virgülle ayrılmış değerler (comma separated values – .csv) formatında, table ve parameter girişi yapmak için kullanılabilir.
- Örnekler:

```
Table d(i,j) uzaklık
$ondelim
$include uzaklik.csv
$offdelim
;
```

Burada uzaklik.csv dosyasının içeriği:

```
, new-york, chicago, topeka
seattle, 2.5, 1.7, 1.8
san-diego, 2.5, 1.8, 1.4
```

```
Parameter d(i,j) uzaklık/
$ondelim
$include uzaklik.txt
$offdelim
/;
```

Burada uzaklik.txt dosyasının içeriği:

```
seattle, new-york, 2.5
san-diego, new-york, 2.5
Seattle, Chicago, 1.7
san-diego, chicago, 1.8
seattle, topeka, 1.8
san-diego, topeka, 1.4
```

Put Komutu

- Program çalışırken çözüm sırasında (execution time) text dosyasına yasmak için kullanılır.
- Dosya ile birlikte bir de fileid (dosyano) kullanılır: `file fileid /dosya1.txt/;`
- Yazmak için bir veri akım yolu (stream) ve dosya adı seçilir: `put fileid;`
- Text, etiketler, numaralar yazdırmak için: `put item{, item};`
- Boş satır bırakmak için: `put /;`
- Veri akımını kapatmak için: `putclose;`

Örnek:

```
file fx /sonuc.txt/;
put fx 'Fabrikalardan pazarlara sevkedilen miktarlar'/;
put '-------------------------------------' /;
loop ((i,j)$x.l(i,j);
  put 'Fabrika ' i.te(i):10, 'dan ' j.te(j):10, 'a
  sevkedilen: ', x.l(i,j) /;
);
putclose;
```

```
Fabrikalardan pazarlara sevkedilen miktarlar
----------------------------------------
Fabrika seattle'dan new-york'a sevkedilen: 50.00
Fabrika seattle'dan chicago'a sevkedilen: 300.00
Fabrika san-diego'dan new-york'a sevkedilen: 275.00
Fabrika san-diego'dan topeka'a sevkedilen: 275.00
```

Gams Program Akışının Kontrolü

Program akışı, GAMS çözüm yaparken yönlendirilebilir/kontrol edilebilir.

Program akışını yönlendirmek için kullanılan komutlar:

`loop, if-else, while, for`

Bu komut blokları içinde denklem tanımlamalarına veya yazımlarına izin verilmemektedir.

`Solve` komutu kullanılabilir.

loop Komutu*

Loop komutu döngü oluşturmak için kullanılır. Örnek:

- Syntax:

```
loop(controllingset[$(condition)],
     statement {; statement}
    );
```

- Example:

```
set t / 1985*1990 /;
parameter pop(t) / 1985  3456 /
          growth(t) / 1985  25.3,  1986  27.3,  1987  26.2,
                      1988  27.1,  1989  26.6,  1990  26.6 /;
loop(t, pop(t+1) = pop(t) + growth(t));
```

if – elseif-else Komutu

Koşula bağlı olarak akışı yönlendirmek için kullanılır.

If-elseif-else yapısının
söz dizimi aşağıdaki gibidir:

```
if (koşul,
    çalışacak komutlar;
elseif şart,
    çalışacak komutlar;
else
    çalışacak komutlar;
);
```

- Example:

```
if( (ml.modelstat eq 4),
*       model ml was infeasible,
*       relax bound and solve again
  x.up(j) = 2*x.up(j);
  solve ml using lp min z;
elseif ml.modelstat eq 1,
  display x.l;
else
  abort "Error solving the model";
);
```

if – elseif-else Komutu

"$" kullanılarak yapılan koşullandırma ifadeleri, if-elseif-else yapısı ile de ifade edilebilirler.

Örnek:

```
p(i)$(f <= 0) = -1 ;
p(i)$((f > 0) and (f < 1)) = p(i)**2 ;
p(i)$(f > 1) = p(i)**3 ;
q(j)$(f <= 0) = -1 ;
q(j)$((f > 0) and (f < 1)) = q(j)**2 ;
q(j)$(f > 1) = q(j)**3 ;
```

```
if (f <= 0,
  p(i) = -1 ;
  q(j) = -1 ;
elseif ((f > 0) and (f < 1)),
  p(i) = p(i)**2 ;
  q(j) = q(j)**2 ;
else
  p(i) = p(i)**3 ;
  q(j) = q(j)**3 ;
) ;
```

while Komutu

`while` ve **`repeat`** yapıları döngü oluşturmak için kullanılır.

`while` komutu söz dizimi:

```
while(koşul,
    çalışacak komutlar;
);
```

- Example:

```
scalar count / 1 /;
scalar globmin / inf /;
while(count le 1000,
  x.l(j) = uniform(x.lo(j), x.up(j));
  solve ml using nlp min obj;
  if ( obj.l le globmin, globmin = obj.l; );
  count = count + 1;
);
```

while Komutu

`repeat` komutu söz dizimi:

```
repeat(
    çalışacak komutlar;
until koşul);
```

- Example:

```
scalar count / 1 /;  scalar globmin / inf /;
repeat( x.l(j) = uniform(x.lo(j), x.up(j));
        solve ml using nlp min obj;
        if ( obj.l le globmin, globmin = obj.l; );
        count = count + 1;
until count eq 1000 );
```

For Döngüsü

Tıpkı **while** ve **repeat** gibi, döngü oluşturmak için **for** yapısı da kullanılabilir.

Söz dizimi:

```
for (i = başl.değeri to|downto bitişDeğeri [by AdımBüyüklüğü],
      statements;
);
```

Burada **başl.değeri** ve **bitişDeğeri** pozitif veya negatif gerçel sayı olabilir. Fakat **AdımBüyüklüğü** değeri pozitif gerçel sayı olmalıdır.

- Example:

```
scalar count;
scalar globmin / inf /;
for( count = 1 to 1000,
  x.l(j) = uniform(x.lo(j), x.up(j));
  solve ml using nlp min obj;
  if ( obj.l le globmin, globmin = obj.l; );
);
```

Text Özellikleri ve Çıktı Formatı

Label names and explanatory texts can be accessed via attributes:

- `ident.ts`: text associated with identifier
- `element.tl`: label associated with set element
- `set.te(element)`: text associated with element of set

Local Item Formatting:

- Syntax for formatting item output: `item:{<>}width:decimals`
- `{<>}` specifies whether justified left (<), right (>), or centered (<>)
- `width` is the field width
- `decimals` is the number of decimals for numeric output
- each can be omitted, e.g., `x.l::5`

Global Item Formatting:

- change field justification and width for all items of a type
- `.lj`, `.nj`, `.sj`, `.tj`, `.lw`, `.nw`, `.sw`, `.tw` attributes of stream identifier
- see GAMS User's Guide Section 15.10

Cursor Positioning:

- `put @n;` moves cursor to column `n` of current line

Subset (Alt Küme)

Söz dizimi:

set1 ⊆ set2: set set1 (set2);

- Example:

```
set ice / chocolate, strawberry, cherry, vanilla /;
set sorbet(ice) / strawberry, cherry /;
```

- Domain checking: `set sorbet(ice) / strawberry, banana /;` ⇒ error

Dynamic Sets

- dynamic sets allow elements to be added or removed
- dynamic sets are usually domain-checked, i.e., subsets
- Syntax: `setname(othersetelement) = yes | no` (add/remove single element)
- Syntax: `setname(subset) = yes | no` (add/remove another subset)
- Example:

```
set ice / chocolate, strawberry, cherry, vanilla /;
Parameter demand(ice) / chocolate 1000, strawberry 500,
                        cherry 10, vanilla 100 /;

set sorbet(ice);
sorbet(ice) = yes;
sorbet('chocolate') = no;   sorbet('vanilla') = no;

Set highdemand(ice)  ice creams with high demand;
highdemand(ice) = (demand(ice) >= 500);
```

- most often used as controlling index in an assignment or equation definition

```
Scalar sumhighdemand;
sumhighdemand = sum(highdemand, demand(highdemand));
```

Examples: refer to first/last period of discrete-time models

```
Set t / 1*24 /;
Sets tb(t)  base period
     tn(t)  non-base periods
     tt(t)  terminal period;
tb(t) = (ord(t) = 1);
tn(t) = (ord(t) > 1);
tt(t) = (ord(t) = card(t));
Variables level(t), inflow(t), outflow(t);
Equations balance(t)     couple fill levels of reservoir over time
          basebalance(t) define fill level for base period;
* only for time periods > base period
balance(tn(t)).. level(t) =e= level(t-1) + inflow(t) - outflow(t);
* only for base period
basebalance(tb).. level(tb) =e= 100 + inflow(tb) - outflow(tb);
* lower bound on fill level in terminal period
level.lo(tt) = 100;
```

Alternatively (but less readable):

```
equation basebalance;  basebalance..
   sum(tb, level(tb)) =e= 100 + sum(tb, inflow(tb) - outflow(tb));
```

Multidimensional Sets

- describing assignments (relations) between sets
- Example:

```
sets c  'countries'  / jamaica, haiti, guyana, brazil /
     h  'harbors'    / kingston, s-domingo, georgetown, belem /
set  hc(h, c)  harbor to country relation
     / kingston.jamaica,   s-domingo.haiti
       georgetown.guyana,  belem.brazil /;
```