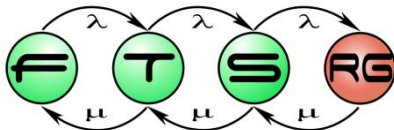


RHadoop

Kocsis Imre

ikocsis@mit.bme.hu



Házi feladat

- Csapatépítés
 - 2 fő, tetszőleges kombinációkban
- <http://goo.gl/M8Yzwq>
- Feladatválasztás
 - Adatsor típusa
 - Adatméret
 - „Jelentkező csapatok maximális száma”
 - Specifikációk feldolgozása beérkezési sorrendben!
 - Honlapon látszik majd, ha valamelyik feladattípus betelt

Specifikáció

- Választott adatsor szöveges leírása
- Dimenziók
 - Fontosabb attribútumok, típussal
 - Sorok száma
 - rmr2 környezet kipróbálása – local backend!
 - Beolvasni, sorokat számolni tudjunk
- Lehetséges kutatási kérdések (~3)
- **Határidő: nov. 10., 23:59**
- **Send to: salanki@mit.bme.hu &&
ikocsis@mit.bme.hu**

RHadoop (rmr2)

Egy/A Big Data probléma

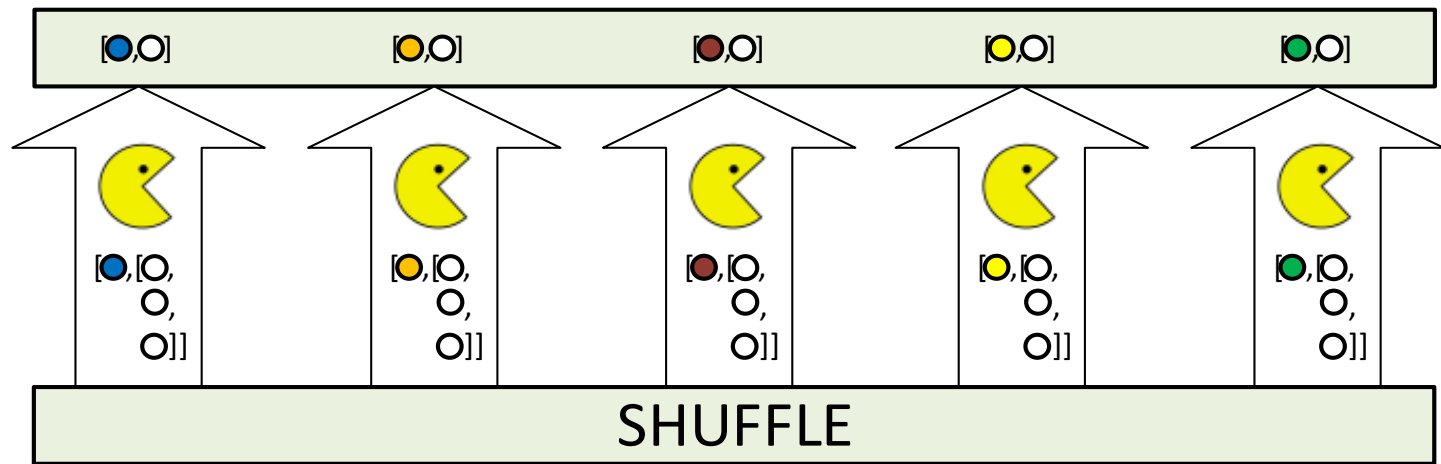
- „At rest Big Data”
 - Nincs update
 - „Mindent” elemzünk
- Elosztott tárolás
- „Computation to data”



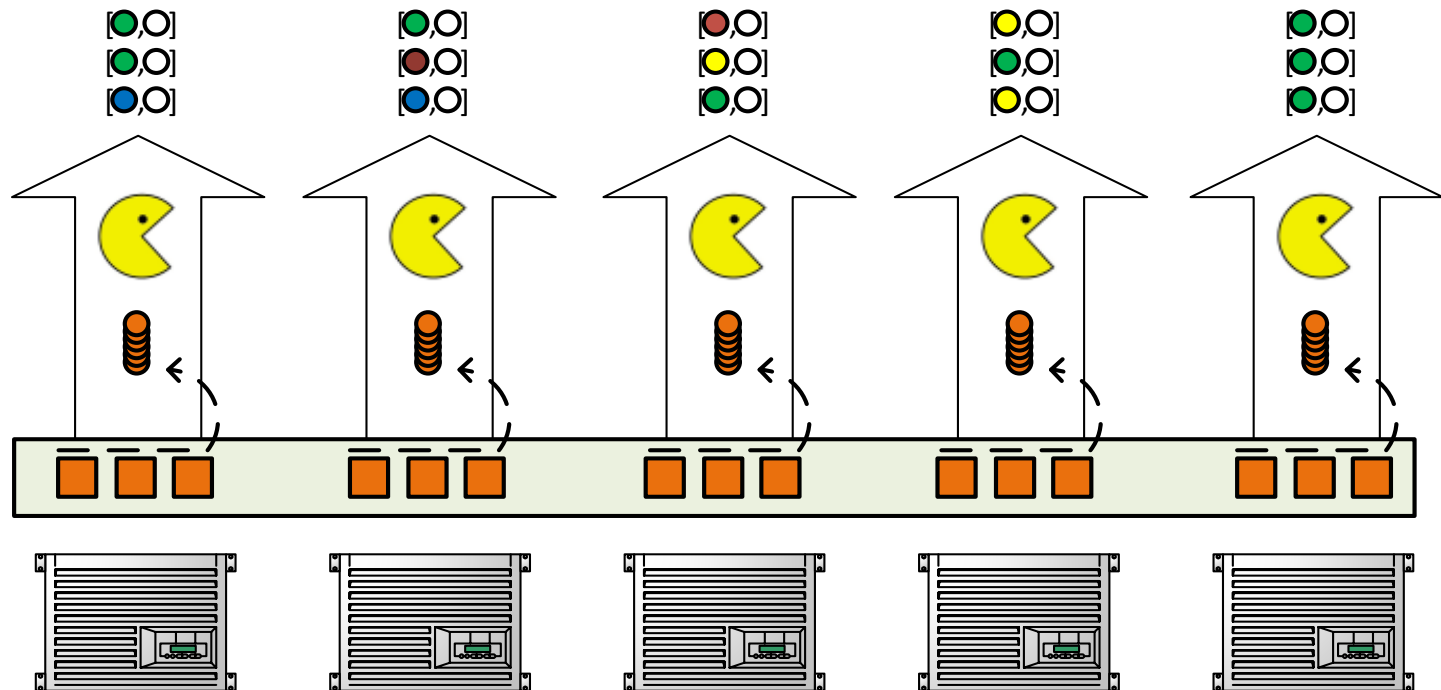
„Not true, but a very, very good lie!”
(T. Pratchett, Nightwatch)

MapReduce

„Reduce”

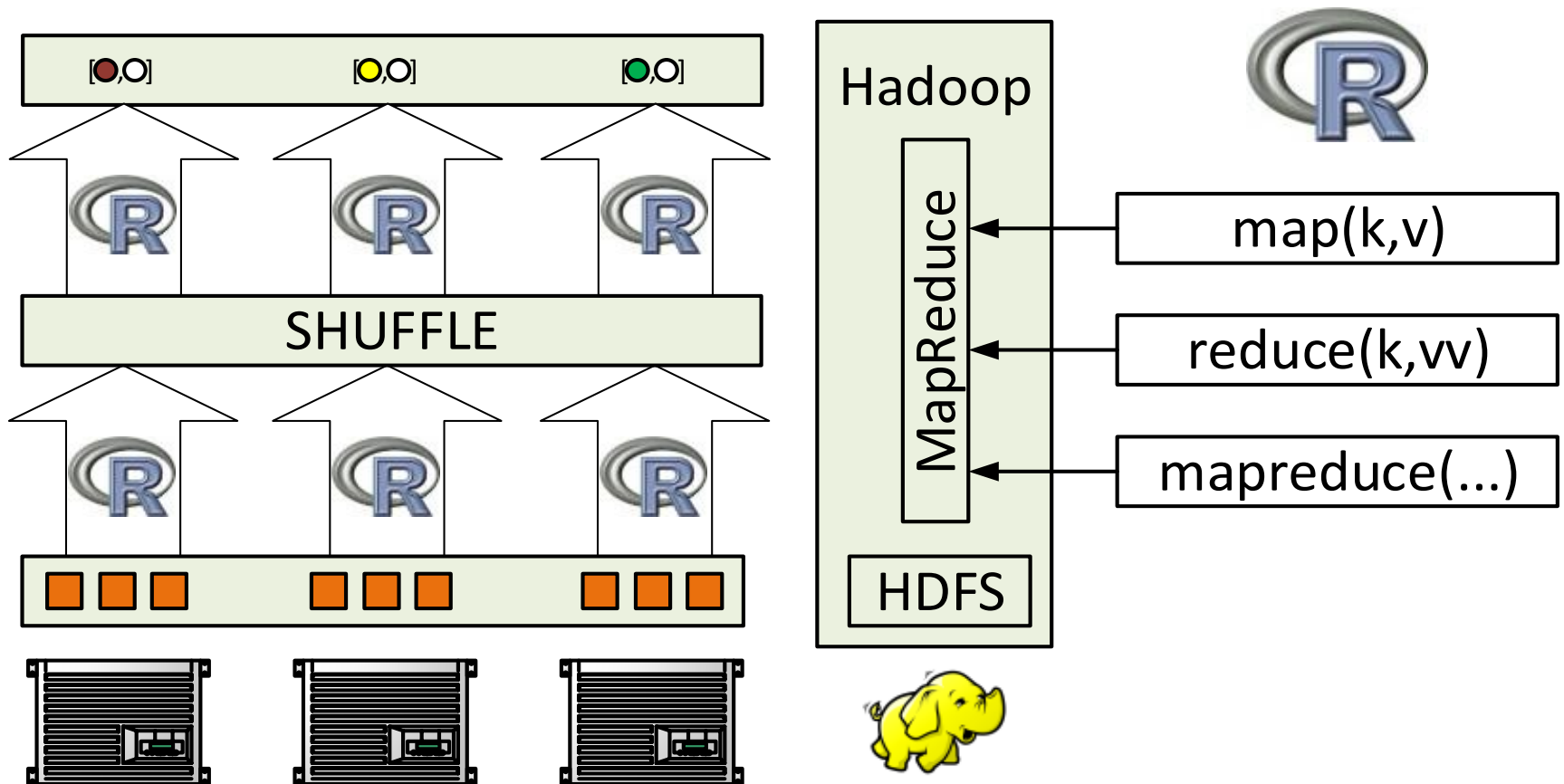


„Map”



Distributed
File System

RHadoop = Hadoop + R



RHadoop

- github.com/RevolutionAnalytics/RHadoop/
- *„The most mature [...] project for R and Hadoop is RHadoop.” (O’Reilly, R In a Nutshell, 2012)*
- `rmr(2)`: mapreduce
- `rhdfs`: HDFS állománykezelés
- `rhbase`, `plyrmr`

Local backend



```
rmr.options(backend="local")
```

- Helyi állományrendszer
- Szekvenciális végrehajtás
- Debug!
- Input/output itt is állományrendszer

RHadoop install (local backend)

- [http://home.mit.bme.hu/~ikocsis/notes/2013/10/23/\(r\)hadoop-sandbox-howto/](http://home.mit.bme.hu/~ikocsis/notes/2013/10/23/(r)hadoop-sandbox-howto/)
- N.B. egy éves tutorial
- De azért megnézzük 😊

Szószámlálás

```
map(String input_key, String input_value):
```

```
    // input_key: document name
```

```
    // input_value: document contents
```

```
    for each word w in input_value:
```

```
        EmitIntermediate(w, "1");
```

```
reduce(String output_key, Iterator intermediate_values):
```

```
    // output_key: a word
```

```
    // output_values: a list of counts
```

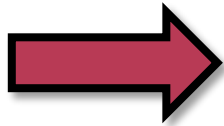
```
    int result = 0;
```

```
    for each v in intermediate_values:
```

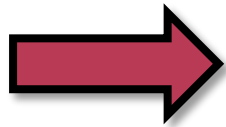
```
        result += ParseInt(v);
```

```
    Emit(AsString(result));
```

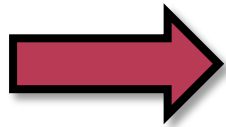
rmr: mapreduce



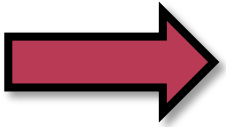
```
wordcount =  
function(  
  input,  
  output = NULL,  
  pattern = " "){
```



```
  wc.map =  
    function(., lines) {  
      keyval(  
        unlist(  
          strsplit(  
            x = lines,  
            split = pattern)),  
        1))}
```

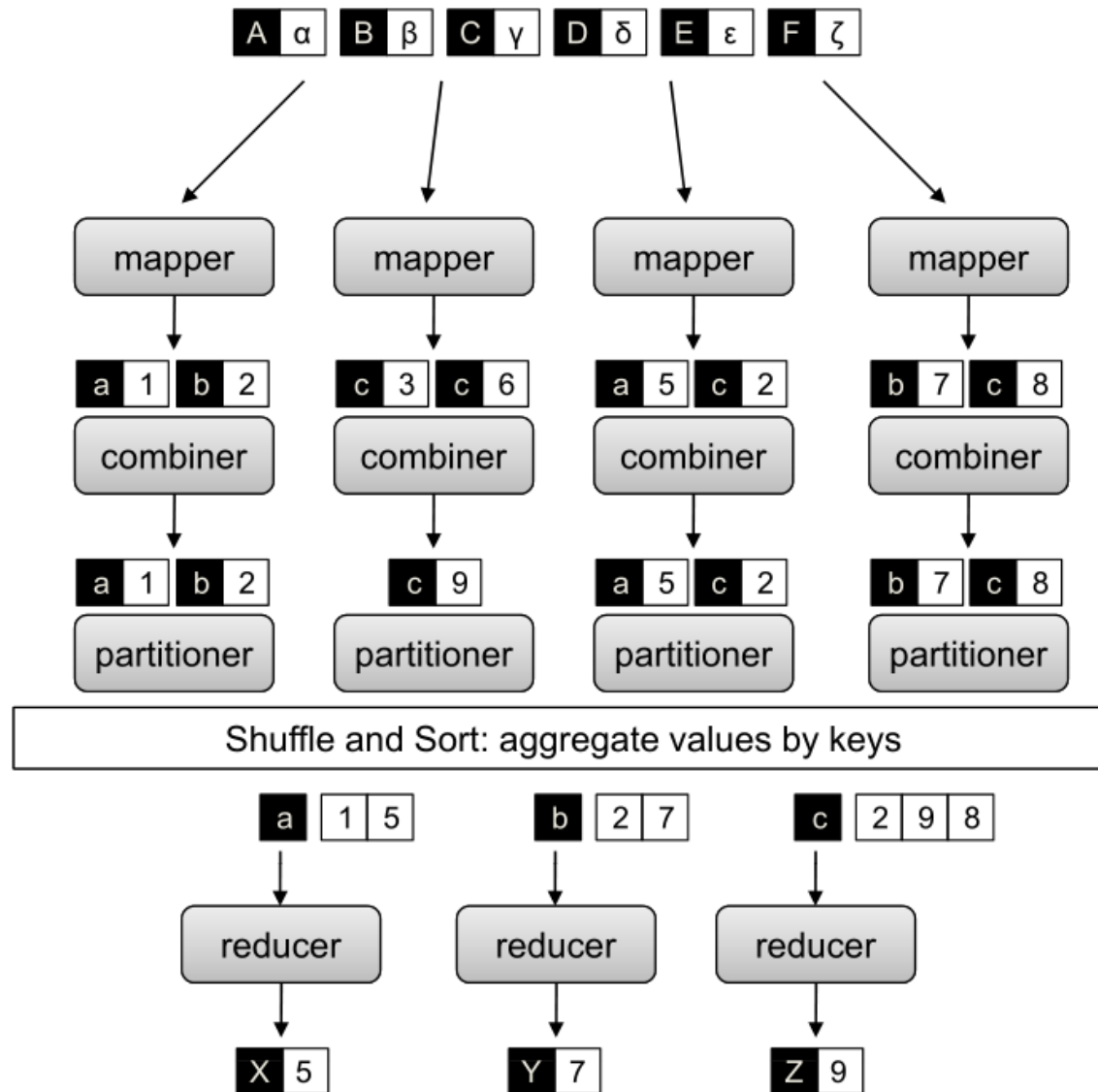


```
  wc.reduce =  
    function(word, counts ) {  
      keyval(word, sum(counts))}
```



```
  mapreduce(  
    input = input ,  
    output = output,  
    input.format = "text",  
    map = wc.map,  
    reduce = wc.reduce,  
    combine = T)}
```

MapReduce: a teljes kép



Forrás: [1], p 30

Input/output format

- text
- json
- csv
- native (R sorosítás)
- sequence.typebytes (Hadoop)
- pig.hive
- hbase

Előnyök

- Map és Reduce: R-ben
 - Csomagok!
 - MR algoritmus-prototipizálás
- + a vezérlés is: kényelem
- Hadoop Job: egy függvényhívás!
 - Pl. iteratív MapReduce teljesen R-ben
 - Map és Reduce: ~a hívó környezetben

Hogyan lehet ilyenem?

- Local backend, sandbox VM-ek
 - Cloudera, Hortonworks
- Saját Hadoop klaszter 😊
- Amazon Elastic MapReduce (EMR)
 - Bérelhető Hadoop klaszter
 - Erősen javasolt kipróbálni
- Saját felhő megoldás

Hátrányok?

- Nehézkes debug
- +1 hangolási réteg
- ~~MAHOUT-klón~~
- ~~Sok Hadoop funkció~~
- Kevés példa



```
14/01/10 15:34:57 INFO streaming.StreamJob: map 100% reduce 0%
14/01/10 15:35:04 INFO streaming.StreamJob: map 100% reduce 100%
14/01/10 15:35:04 INFO streaming.StreamJob: To kill this job, run:
14/01/10 15:35:04 INFO streaming.StreamJob: UNDEF/bin/hadoop job -Dmapred.job.tracker=hdfs://10.6.21.150:54311 -kill job_201401101345_0002
14/01/10 15:35:04 INFO streaming.StreamJob: Tracking URL: http://vm-large-10.vcl.i
ntra:50030/jobdetails.jsp?jobid=job_201401101345_0002
14/01/10 15:35:04 ERROR streaming.StreamJob: Job not successful. Error: NA
14/01/10 15:35:04 INFO streaming.StreamJob: killJob...
Streaming Command Failed!
Error in mr(map = map, reduce = reduce, combine = combine, vectorized.reduce, :
  hadoop streaming failed with error code 1
> |
```

Folytatás: GIM-V

PageRank

- A oldalt linkeljék $T_1, T_2, \dots, T_n$ oldalak.
- $0 < d < 1$.

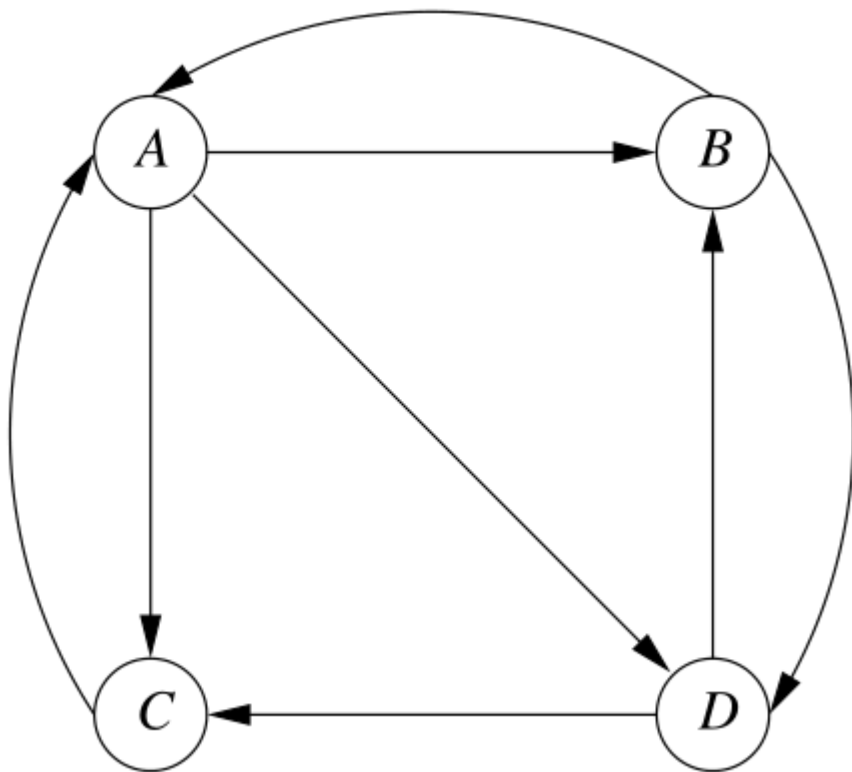
- Eredeti, publikált default: 0.85

- $C(A)$: kimenő fokszám
- Definíció:

$$PR(A) = (1 - d) + d \left(\frac{PR(T_1)}{C(T_1)} + \dots + \frac{PR(T_n)}{C(T_n)} \right)$$

- $\sim$ random látogató helyének val. eloszlása

Tranzíciós mátrix



$$M = \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

Általánosítás: a GIM-V primitív

- „Generalized Iterative Matrix-Vector multiplication”

- A szokásos feladat:

$$M \times v = v', \text{ ahol } v'_i = \sum_{j=1}^n m_{ij} v_j$$

- Valójában három operátor:

- combine2: a szorzás
- combineAll: n szorzás-eredmény szummája i-re
- assign: vektorelemek frissítése

GIM-V

$$v' = M \times_G v$$

$$v'_i = \text{assign}(v_i, \text{combineAll}_i(\{x_j | j = 1 \dots n, x_j = \text{combine2}(m_{ij}, v_j)\}))$$

- Iteratív: $\times_G$ -t konvergenciakritériumig alkalmazzuk
- BTW feltételezve egy $E(\text{sid}, \text{did}, \text{val})$ és $V(\text{id}, \text{val})$ táblát $\times_G$ SQL-ben:

```
SELECT E.sid, combineAllE.sid(combine2(E.val, V.val))  
FROM E, V  
WHERE E.did=V.id  
GROUP BY E.sid
```

GIM-V: PageRank

- M : oszlopnormalizált szomszédossági mátrix
- $\text{combine2}: c \times m_{ij} \times v_j$
- $\text{combineAll}: \frac{1-c}{n} + \sum_{j=1}^n x_j$
- $\text{assign}: v_{\text{new}}$

GIM-V: Random Walk with Restart

- Csomópontok közelségének mérése
- A k -adik csomópontból induló vektor egyenlete (M ua):

$$r_k = cMr_k + (1 - c)e_k$$

- Ahol e_k nullvektor a k -adik, 1 értékű pozíciótól eltekintve
- combine2: $c \times m_{ij} \times v_j$
- combineAll: $(1 - c)I(i \neq k) + \sum_{j=1}^n x_j$
- assign: v_{new}

GIM-V: összefüggő komponensek

- Minden v_i -hez „komponens ID” c_i^h :
- Minimum csomópont-ID h hopen belül
- M : szomszédossági mátrix
- Minden iterációban:
 - Minden csomópont „elküldi” szomszédjainak c_i^h -ját
 - Iteráció: minden csomópontban minimumképzés
- combine2: $m_{ij} \times v_j$
- combineAll: $MIN\{x_j | j = 1 \dots n\}$
- assign: $MIN(v_i, v_{new})$

GIM-V: összefüggő komponensek

- Megállás: nem változik a vektor
- Tétel: legfeljebb gráfátmérő számú lépés kell
 - *Diameter*: pontpárok közötti legrövidebb utak hosszának maximuma

GIM-V: algoritmusok

- Naive multiplication, **block multiplication**, clustered edges, diagonal block iteration, ...
- Blokk szorzás: vektorokat és mátrixot is blokkosítva tárolunk (és csak ha nem nulla)

