# Sorfolytonos szorzat alakú bázis inverz reprezentáció a primál szimplex módszerben

Smidla József
Témavezető: Dr. Maros István

Rendszer- és Számítástudományi Tanszék
Pannon Egyetem

XXX. Magyar Operációkutatási Konferencia
2013. június 10.

# Lineáris programozás

## Lineáris programozási feladat

$$\min \mathbf{c}^T \mathbf{x}$$

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

$$\mathbf{x}_j \geq 0, j = 1..n$$

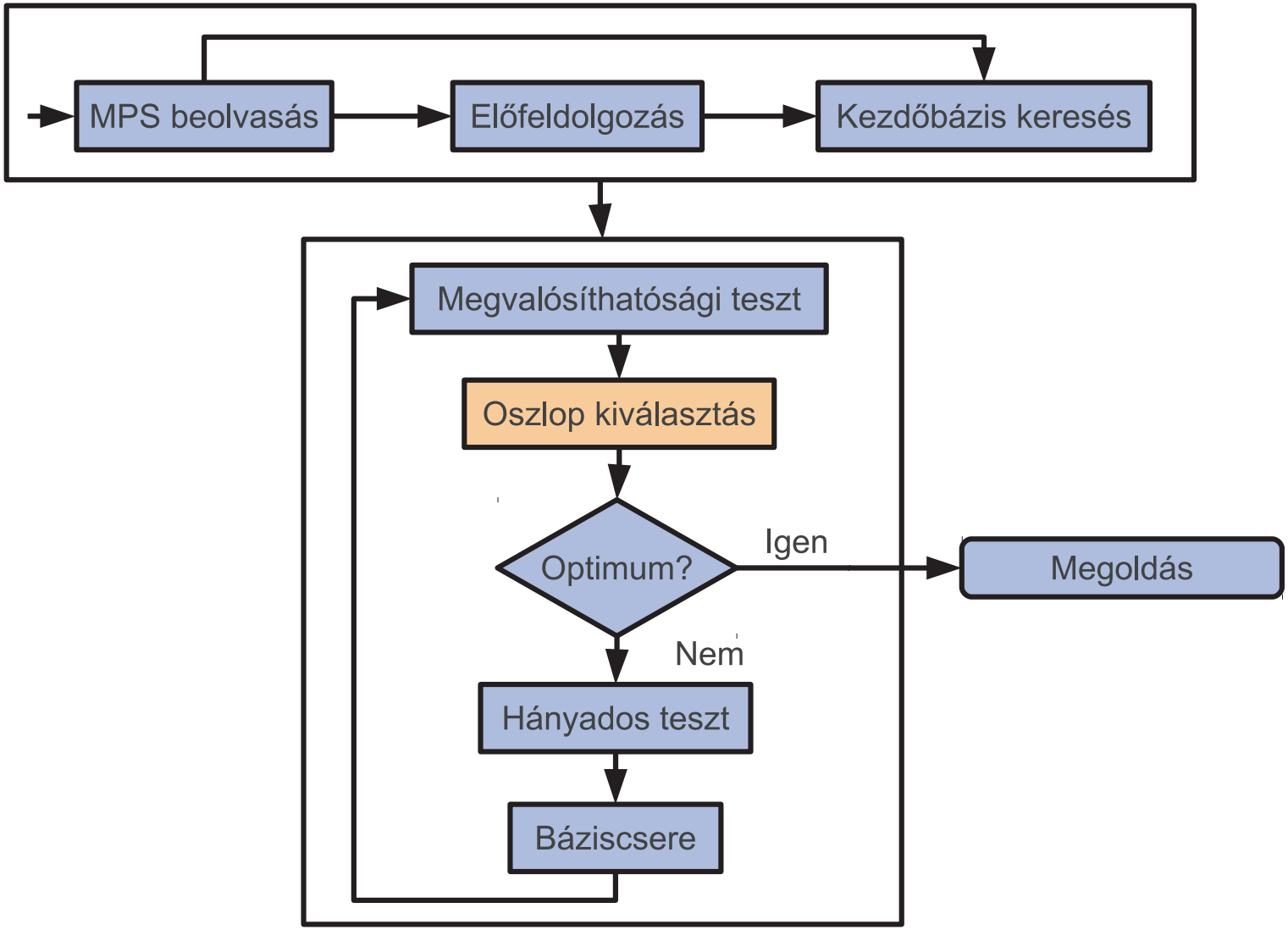
A változók indexeinek halmaza: $\mathcal{N}$
A bázis változók indexeinek halmaza: $\mathcal{B} = \{k_1, \ldots, k_m\}$
A nembázis változók indexeinek halmaza: $\mathcal{R} = \mathcal{N} \setminus \mathcal{B}$

## A mátrix és a vektorok átrendezése

$$\mathbf{A} = [\mathbf{B}|\mathbf{R}], \mathbf{x} = \begin{bmatrix} \mathbf{x}_\mathcal{B} \\ \mathbf{x}_\mathcal{R} \end{bmatrix} \text{ és } \mathbf{c} = \begin{bmatrix} \mathbf{c}_\mathcal{B} \\ \mathbf{c}_\mathcal{R} \end{bmatrix}$$

Egymással szomszédos bázisokon haladva keresünk optimumot

# Az algoritmus



# Megengedettség és optimalitás

## Megengedettségi feltétel

$$\mathbf{x}_j \geq 0, j = 1..n$$

## Optimalitási feltétel

A $j$ változó második fázisú redukált költsége:

$$d_j = c_j - \mathbf{c}_{\mathcal{B}}\mathbf{B}^{-1}\mathbf{a}_j, j \in \mathcal{N}$$

A megoldás optimális, ha:

$$\mathbf{x}_j \geq 0 \text{ és } d_j \geq 0, \forall j \in \mathcal{N}$$

Az első fázisban keresünk egy megvalósítható megoldást $\Longrightarrow$ a második fázisban egy optimális megoldást keresünk

# Oszlop választás az első fázisban

Első fázisú redukált költség:

$$d_j = \sum_{\mathbf{x}_{\mathcal{B}_i} < 0} \alpha_j^i$$

## Az oszlop kiválasztó feladata

Javító változó: Olyan $k \in \mathcal{R}$ nembázis változó, amelyre

$$d_k < 0$$

Keresünk egy javító változót, hogy a bázisba vigyük

- Van javító változó: egy belép a bázisba
- Nincsen javító változó: a feladatnak nincsen megoldása

# Oszlop választás a második fázisban

Második fázisú redukált költség:

$$d_j = c_j - \mathbf{c}_{\mathcal{B}}\mathbf{B}^{-1}\mathbf{a}_j, j \in \mathcal{N}$$

## Az oszlop kiválasztó feladata

Javító változó: Olyan $k \in \mathcal{R}$ nembázis változó, amelyre

$$d_k < 0$$

Keresünk egy javító változót, hogy a bázisba vigyük

- Van javító változó: a megoldás nem optimális
- Nincsen javító változó: optimális megoldást találtunk

# Első fázisú szimplex szorzó

## Számolás

A változóknak megengedett értéken kell lenniük: $\mathbf{x} \geq 0$
Legyen $\mathbf{h}$ egy olyan vektor, ahol

$$h_i = \begin{cases} 1, & \text{ha } \mathbf{x}_{\mathcal{B}_i} < 0 \\ 0, & \text{egyébként} \end{cases}$$

Legyen $\phi$ az első fázisú szimplex szorzó:

$$\phi^T = \mathbf{h}^T \mathbf{B}^{-1}$$

Ez alapján az első fázisú redukált költség:

$$d_j = \phi^T a_j$$

# Első fázisú szimplex szorzó gyorsított számolása

## $\phi$ hagyományos számolása

1. $\mathbf{h}$ meghatározása
2. $\phi^T = \mathbf{h}^T \mathbf{B}^{-1}$ szorzat kiszámítása

## Saját megközelítés

A bázisváltás függvényében határozzuk meg $\phi$ új értékét!

- Az új bázis: $\bar{\mathbf{B}}$
- Bázisváltáskor $\mathbf{h}$-ról $\bar{\mathbf{h}}$-ra váltunk

1. Meghatározzuk $\bar{\mathbf{h}}$-t
2. Legyen $\Delta\mathbf{h} = \bar{\mathbf{h}} - \mathbf{h}$
3. $\Delta\phi^T = (\phi^T(\mathbf{B} - \bar{\mathbf{B}}) + \Delta\mathbf{h}^T)\bar{\mathbf{B}}^{-1}$.
4. $\bar{\phi}^T = \phi^T + \Delta\phi^T$

# Szorzat alakú bázis inverz

- A Pannon Optimizer fejlesztésének kezdetekor: PFI
- Implementációja egyszerűbb, mint az LU dekompozíció
- A fejlesztés kezdete felgyorsítható
- Előzetes várakozásunk: A NETLIB-es feladatoknak csupán kis részét lehet megoldani
- Tapasztalat: A saját implementációnkkal a NETLIB-es feladatok nagy része megoldható úgy, hogy akár több ezer iterációt hajtunk végre az újrainvertálások előtt

Péter Tar and István Maros.
Product Form of the Inverse Revisited.
In Stefan Ravizza and Penny Holborn, editors, *3rd Student Conference on Operational Research*, volume 22 of *OpenAccess Series in Informatics (OASIcs)*, pages 64–74, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

# BTRAN művelet

Bárhogy is számoljuk $\phi$ értékét, szükségünk van az alábbi műveletre:

$$\boldsymbol{\alpha}^T = \mathbf{a}^T \mathbf{B}^{-1}$$

$\mathbf{B}^{-1}$-et Elemi Transzformációs Mátrixok (ETM) szorzataként tároljuk

$$\boldsymbol{\alpha}_0^T = \mathbf{a}^T$$

$$\boldsymbol{\alpha}_i^T = \boldsymbol{\alpha}_{i-1}^T \mathbf{E}_{s-i+1}, i = 1, \ldots, s$$

$$\mathbf{E} = \begin{bmatrix} 1 & & \eta^1 & & \\ & \ddots & \vdots & & \\ & & \eta^p & & \\ & & \vdots & \ddots & \\ & & \eta^m & & 1 \end{bmatrix}$$

# BTRAN művelet

Az $\boldsymbol{\alpha}_i^T = \boldsymbol{\alpha}_{i-1}^T \mathbf{E}_{s-i+1}$ művelet elvégzése szorzat alakkal:

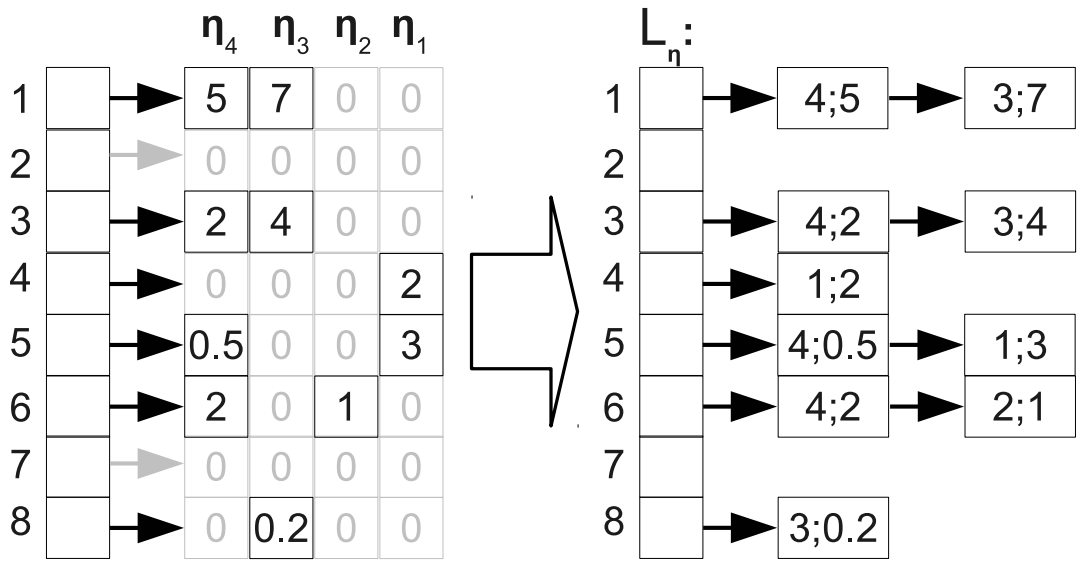
$$\boldsymbol{\alpha}_i^T = \boldsymbol{\alpha}_{i-1}^T \mathbf{E} = \left[ \begin{array}{cccc} \alpha_1, & \ldots, & \alpha_p, & \ldots, \alpha_m \end{array} \right] \left[ \begin{array}{ccccc} 1 & & \eta^1 & & \\ & \ddots & \vdots & & \\ & & \eta^p & & \\ & & \vdots & \ddots & \\ & & \eta^m & & 1 \end{array} \right]$$

$$\boldsymbol{\alpha}_i^T = \boldsymbol{\alpha}_{i-1}^T \mathbf{E} = \left[ \begin{array}{cccc} \alpha_1, & \ldots, & \sum_{i=1}^m \alpha_i \eta^i, & \ldots, \alpha_m \end{array} \right]$$

Hátrány: Ritkás $\boldsymbol{\eta}$ és $\mathbf{a}$ vektorok esetén sok felesleges 0 értékű szorzat keletkezik.

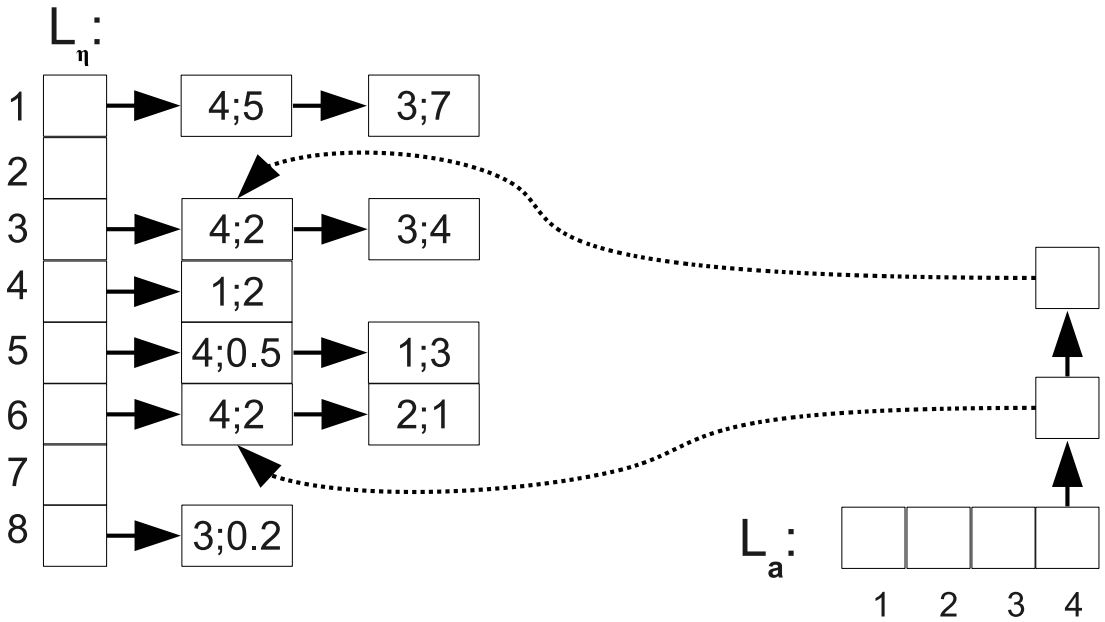
# Sorfolytonos BTRAN implementáció

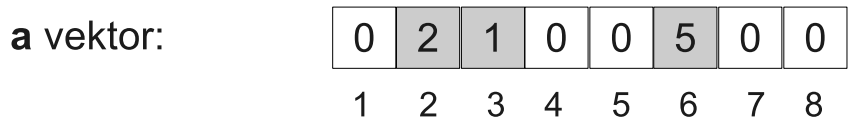
- Előny: Csak olyan szorzásokat hajt végre, amelyek eredménye nem nulla
- Hátrány: Logikailag összetettebb
- Az $\eta$ vektorokat sorfolytonosan tároljuk



# Sorfolytonos BTRAN implantáció

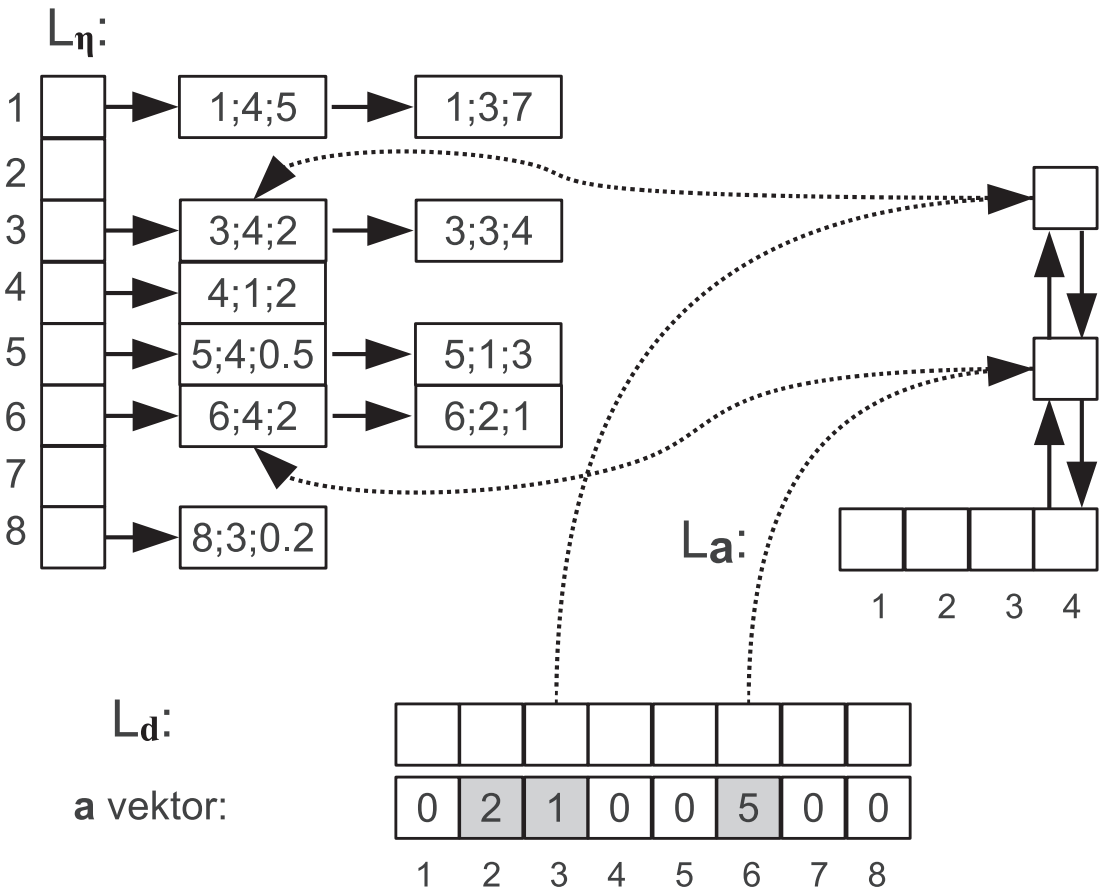
- Példa bemenet vektor: $\mathbf{a} = [0, 2, 1, 0, 0, 5, 0, 0]$
- Először az utolsó, az $\eta_4$ vektorral kell összeszorozni az **a**-t
- Csupán 2 elemet kell összeszorozni: $\eta_4^3 * a_3 + \eta_4^6 * a_6$



# Sorfolytonos BTRAN implementáció

A teljes adatszerkezet:



# Hatások

## Sorfolytonos BTRAN hatása

- Az első fázisban szükségünk van BTRAN-ra
  - Hagyományos szimplex szorzó számoláshoz: $\phi^T = \mathbf{h}^T \mathbf{B}^{-1}$
  - Gyorsított szimplex szorzó számoláshoz: $\Delta\phi^T = (\phi^T(\mathbf{B} - \bar{\mathbf{B}}) + \Delta\mathbf{h}^T)\bar{\mathbf{B}}^{-1}$
- A második fázisban is kell BTRAN
  - Második fázisú szimplex szorzó számolásához: $\bar{\boldsymbol{\pi}}^T = \boldsymbol{\pi}^T + d_q \mathbf{e}_p^T \bar{\mathbf{B}}^{-1}$
- A sorfolytonos BTRAN gyorsít minden szimplex szorzó számolást

## Gyorsított első fázisú szimplex szorzó számolás

- Az oszlopfolytonos BTRAN mellett nem gyorsít
- A sorfolytonos BTRAN-t gyorsítja

# Javító változó választása

## Egyszerű megközelítés

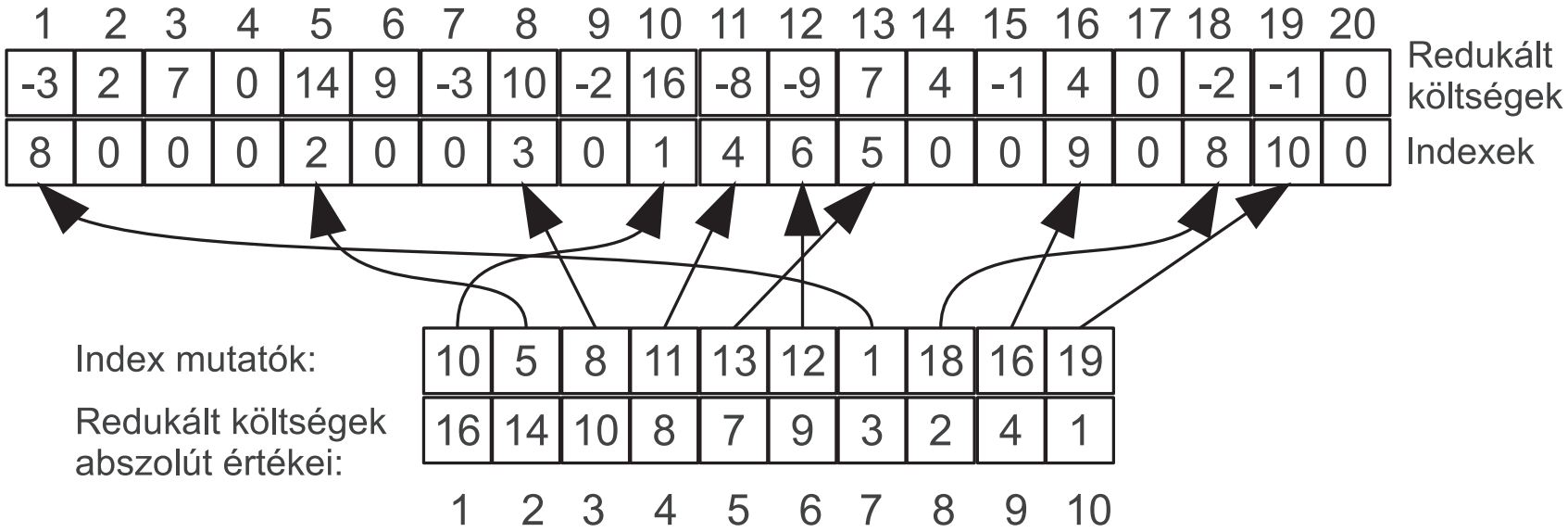
1. Számoljuk ki a szimplex szorzóval a megváltozott redukált költségeket
2. A redukált költségeket egy tömbben tároljuk
3. A redukált költségek között keressük meg a minimálist: $O(n)$

## Előnyök

- Egyszerűen implementálható
- Ha sok redukált költség változik, akkor célszerű ezt alkalmazni

# Gyorsított kiválasztás

- A javító vektorok indexeit és redukált költségeit tároljuk kupacban.
- Megfelelő indexeléssel gyorsan aktualizálhatjuk a kupacot.
- Kupac adminisztrációs lépések komplexitása: $O(\log_2)$
- Kiválasztás komplexitása: $O(1)$
- Kevés redukált költség megváltozásakor érdemes használni



# Néhány mérési eredmény

| Feladat | Eredeti idő [*ms*] | Gyorsítás utáni idő [*ms*] | Gyorsulás |
|---|---|---|---|
| 25FV47 | 3973 | 2463 | 1,61 |
| CAPRI | 53 | 19 | 2,76 |
| CYCLE | 4881 | 690 | 7,08 |
| STOCFOR2 | 203 | 79 | 2,56 |
| STOCFOR3 | 16574 | 3556 | 4,66 |
| GANGES | 123 | 49 | 2,48 |
| GFRD-PNC | 15 | 91 | 0,17 |
| GREENBEA | 11838 | 5507 | 2,15 |
| MAROS-R7 | 6450 | 3215 | 2,01 |
| QAP08 | 21383 | 5895 | 3,63 |
| SCFXM3 | 221 | 70 | 3,16 |
| SCTAP3 | 700 | 74 | 9,46 |
| STAIR | 441 | 65 | 6,80 |

A szoftverünkkel megoldható feladatokat összességében gyorsabban oldottuk meg.

# Összefoglalás

## Elvégzett munka

- Implementáltam az oszlop kiválasztó modult
- Kidolgoztam az alábbi gyorsításokat:
  - Első fázisú szimplex szorzó frissítésének képlete
  - Sorfolytonos BTRAN implementáció
  - Kupac alapú redukált költség választó algoritmus
- Mérésekkel vizsgáltam a gyorsítások hatásait

## További tervek

- Sorfolytonos BTRAN tovább fejlesztése
- A gyorsítások használatának automatikus felismerése

# Köszönöm a figyelmet!