Towards green computing in Erlang[1]

Mészáros Áron Attila
Nagy Gergely

ELTE IK, Programtervező informatikus BSc

Tudományos Diákköri Konferencia
2018. május 10.
Budapest

[1]A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg (EFOP-3.6.3-VEKOP-16-2017-00002).

Bevezetés – Motiváció

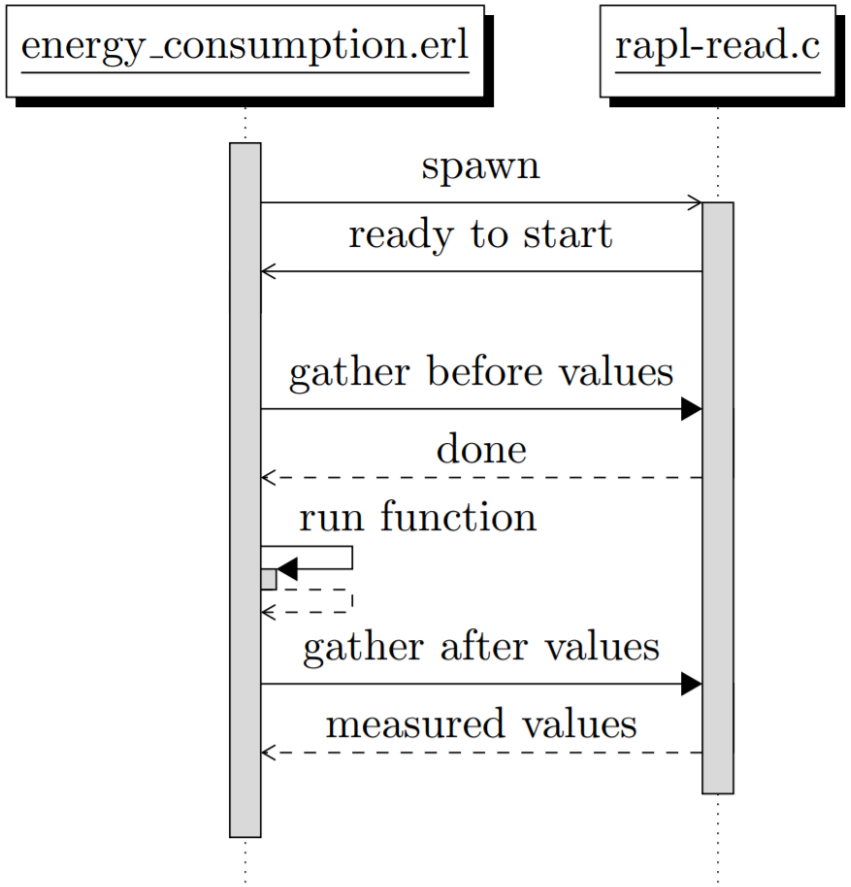
- Környezettudatosság
  - Energiafogyasztás minimalizálása
  - Számítógépes eszközöknél is
  - Green computing
- Miért Erlang?
  - Népszerűbb nyelvek – sok kutatás
    - C++, Java, Haskell
  - Erlang
    - Széleskörűen használt, iparban
    - Még nem volt ilyen kutatás

Bevezetés – Eredmények

- Eszköz energiafogyasztás mérésére
  - RAPL, rapl-read.c
  - Erlang keretrendszer
  - Python megjelenítő
- Magasabb rendű függvények hatása
- Lista vagy tömb
- Párhuzamosítás
  - Process pool
  - Token ring
- Folyóiratcikk és konferencia absztrakt

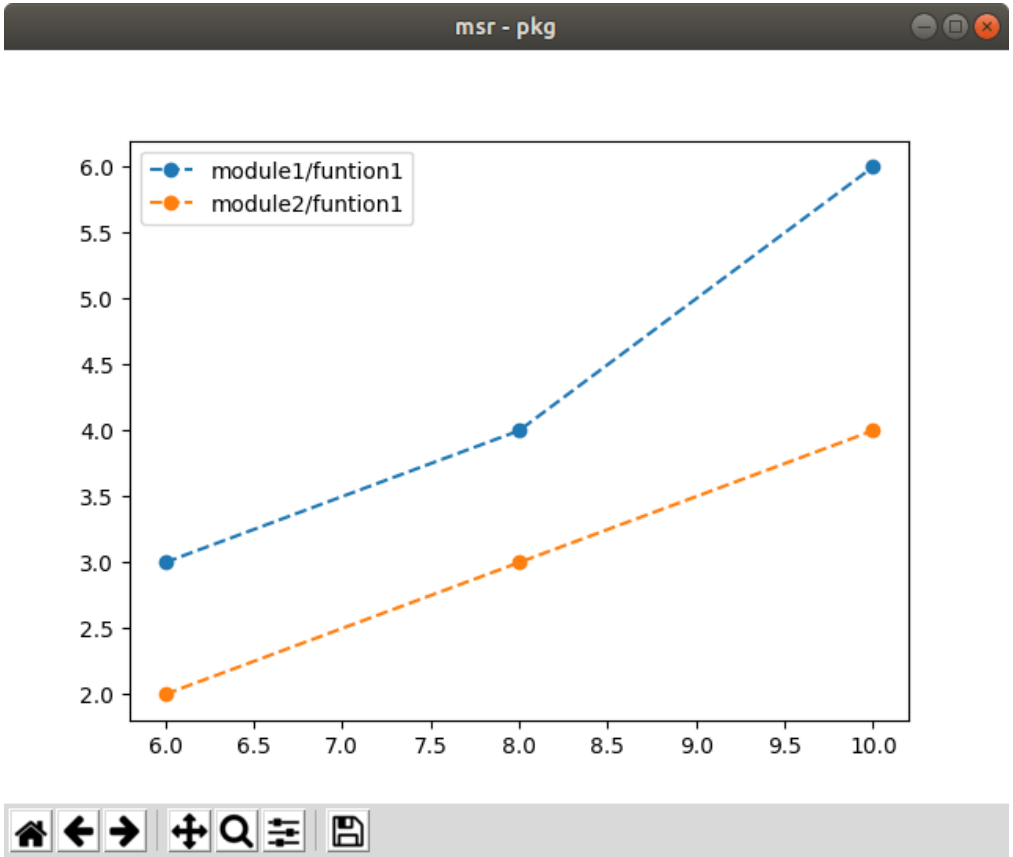
Módszertan – Mérés

- Running Average Power Limit (RAPL)
  - Intel
  - rapl-read.c
  - MSR, perf_event, sysfs módszerek
  - 4 domain
    - PKG – package
    - PP0 – core
    - PP1 – uncore
    - DRAM
- Erlang keretrendszer
  - Kommunikáció
  - Mérni kívánt függvény futtatása
  - Mérési eredmények összegyűjtése

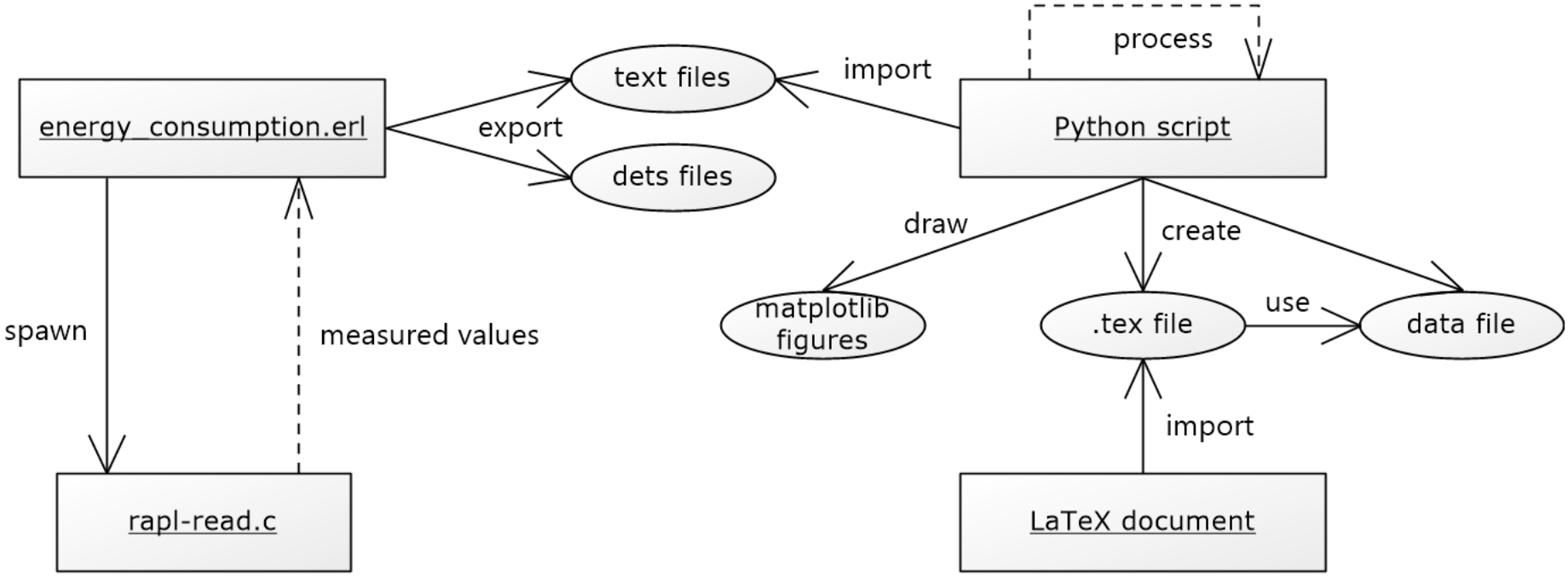


Módszertan – Adatfeldolgozás

- Mérési módszertan
  - 10 db mérés – átlagolás
  - Minimális és maximális értékek eldobása
  - Futási idő mérése
- Vizualizáció
  - Python – matplotlib
- Teljesítmény számítása
- Korreláció számítása energia és idő között



Módszertan – Összefoglalás



Mérési szempontok

- Magasabb rendű függvények
  - map, foldr, foldl, filter stb.
- Különböző adatszerkezetek
  - Lista
  - Tömb
- Párhuzamosítás
  - Brute force
  - Process pool
  - Folyamatok száma
  - Üzenetküldés költsége
  - Magok száma

MÉRT ALGORITMUSOK – N KIRÁLYNŐ

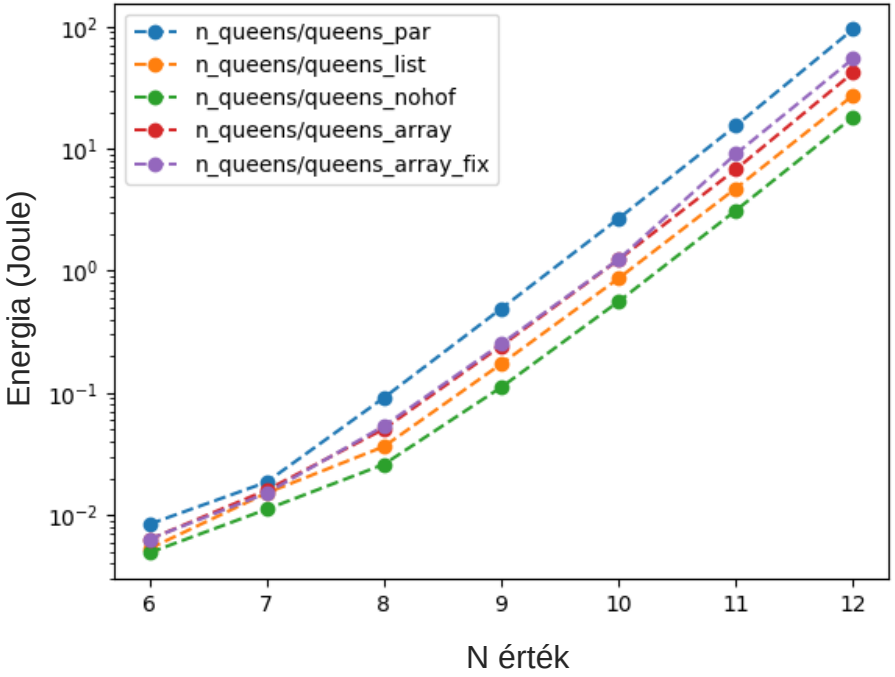
- 5 féle megoldás
  - Lista
    - Magasabb rendű függvények megléte vagy eliminálása
    - Párhuzamos verzió
  - Tömb
    - Fix vagy növelhető méret

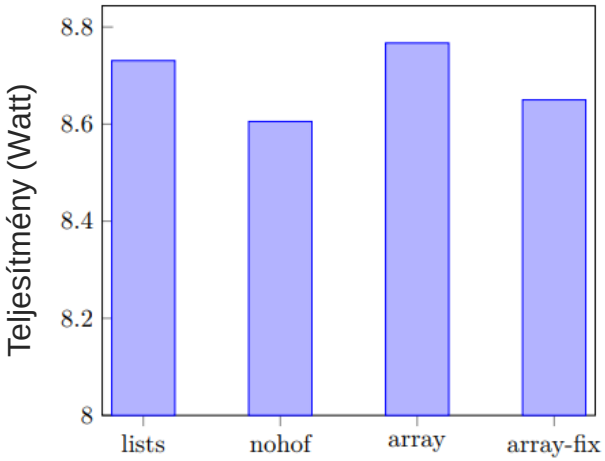
```
par_map(F, Xs) ->
  Me = self(),
  [spawn(fun() -> Me ! F(X) end) || X<-Xs],
  [receive Res -> Res end || _ <- Xs].
```

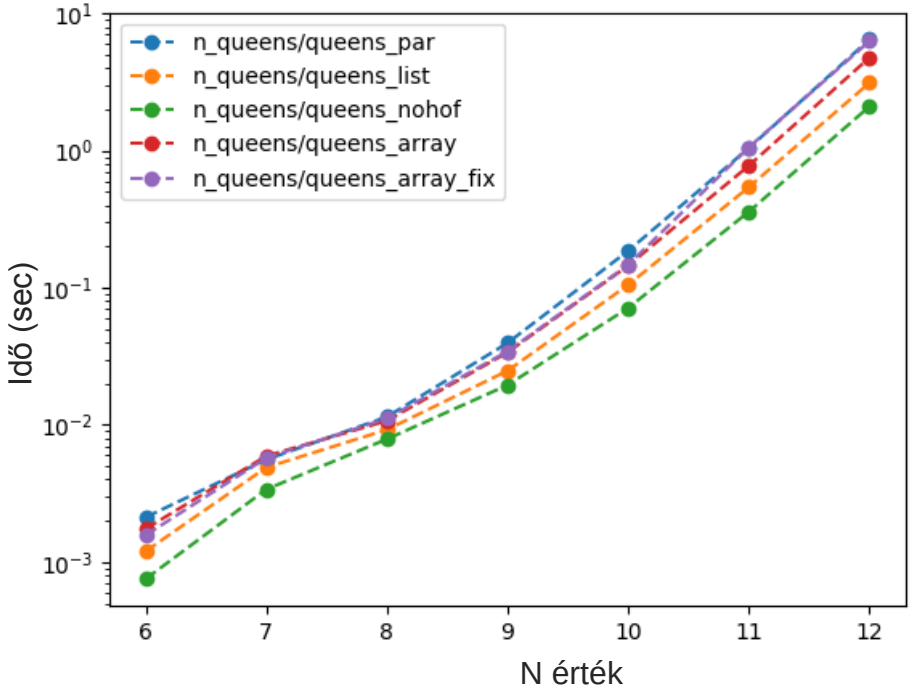
```
lists:flatmap(
fun(Qs) -> solve_list(N,Row+1,Qs) end,
[[{Col,Row}|Queens] || Col <- lists:seq(1,N),legal_list({Col,Row},Queens)]
).
```

```
flatmap_nohof(Queens,N,Row) -> flatmap_nohof(Queens,[],N,Row).
flatmap_nohof([],R,_,_) -> R;
flatmap_nohof([H|T],R,N,Row) ->
  flatmap_nohof(T,solve_nohof(N,Row+1,H) ++ R,N,Row).
```

Eredmények – N királynő







- Magasabb rendű függvény nélkül jobb
- Tömbök rosszabbak
- Brute force párhuzamos rossz

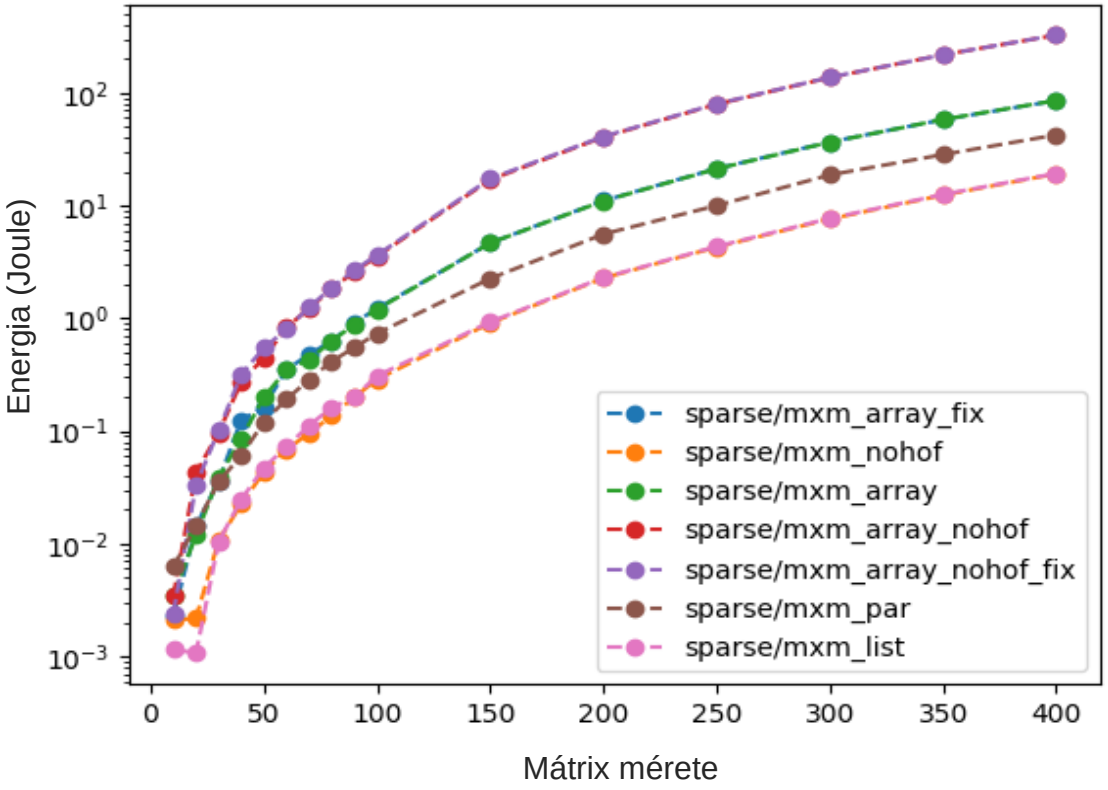
Mért algoritmusok – Ritka mátrix szorzás

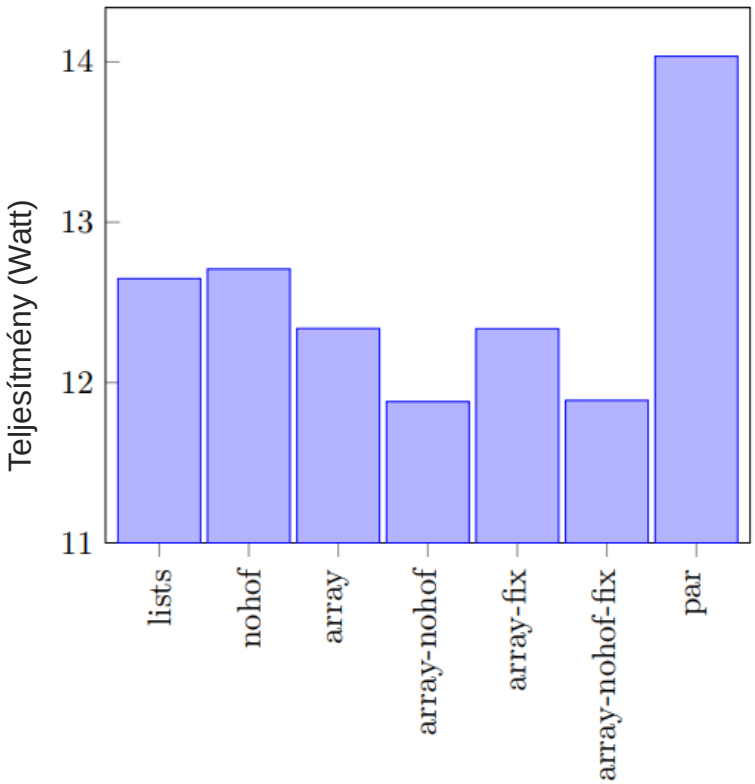
- Hasonló megoldások mint előbb
- Magasabb rendű függvények tömböknél is
- Különböző reprezentáció tömbök és listák esetén

```
mxm_array(Rows,Cols) ->
  array:sparse_map(fun(_,Col) ->
      if  Col == undefined -> undefined;
        true -> mxv_array(Rows,Col) end end,Cols).
```

```
vxv_array_map(Index,Size,_,Row) when Index == Size -> Row;
vxv_array_map(Index,Size,Col,Row) ->
  ElemR = array:get(Index,Row),
  ElemC = array:get(Index,Col),
  if  ElemR == undefined -> vxv_array_map(Index+1,Size,Col,Row);
    ElemC == undefined -> vxv_array_map(Index+1,Size,Col,array:set(Index,
    undefined, Row));
    true -> vxv_array_map(Index+1,Size,Col,array:set(Index, ElemC*ElemR, Row)
    )
  end.
```

Eredmények – Ritka mátrix szorzás



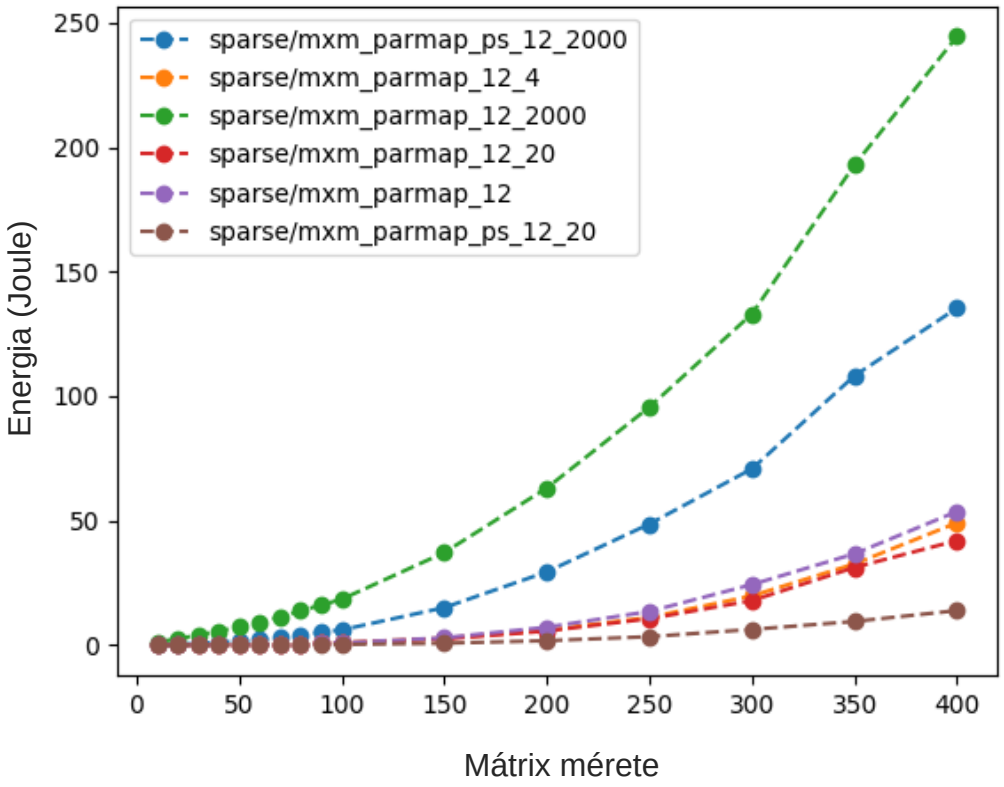


- Magasabb rendű függvény nélkül
  - listánál javulás
  - tömbnél romlás
- Tömbök nem hatékonyak, ha sok az ismeretlen elem

Párhuzamosítási módszerek

- Ritka mátrix szorzás párhuzamosítása
- Cél: folyamatok száma hogyan hat az energiára?
- Brute force módszer
  - Ahol map van párhuzamossá tesszük
- Process poolok
  - worker, dispatcher, collector
- Csak a program külső iterációinak párhuzamosítása
  - Folyamat létrehozások csökkentése
  - Egy folyamat többet számol

Párhuzamosítási módszerek

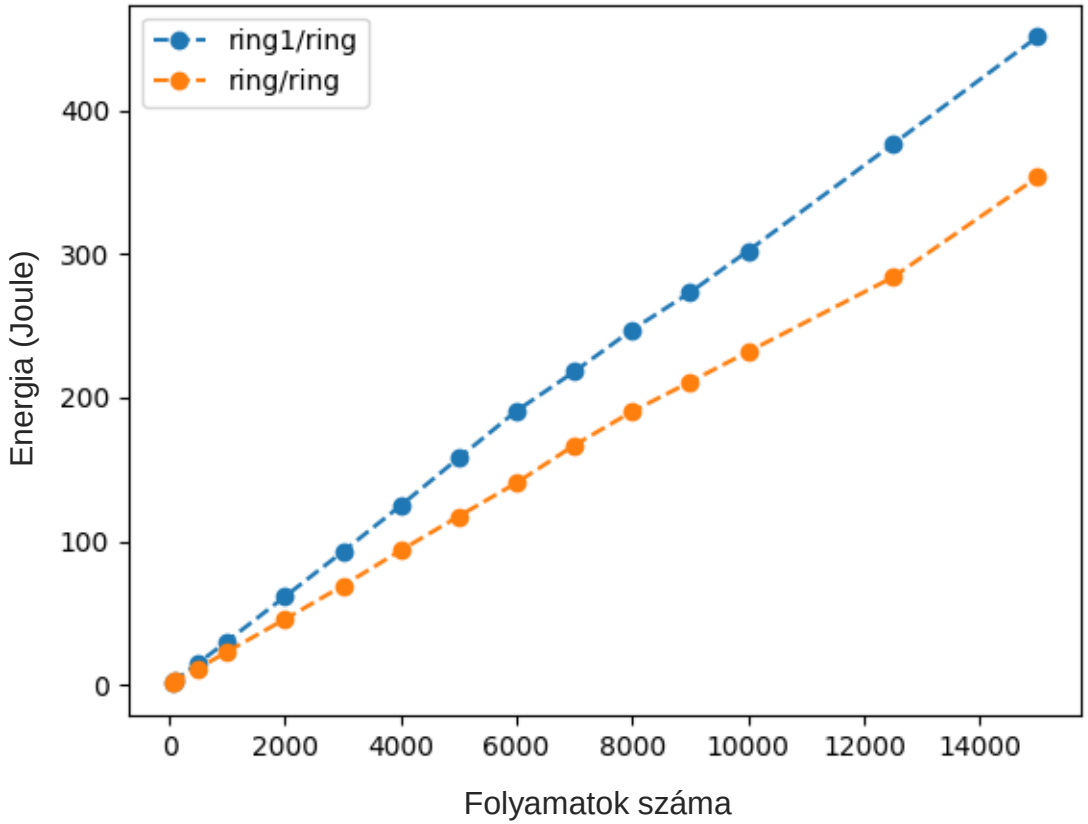


- Legjobb módszer:
  - Egyszerre kevés folyamat
  - Kevés létrehozás
  - Nagyságrendileg a magok számához legyen közel a folyamatok száma
- Ha sok a folyamat sok energiát fogyaszt

Token Ring

- Üzenetküldés költsége?
- Körbe küldjük a folyamatokon
- Minimális a többi számítás

- Több üzenet, több energia



Konklúzió

- Eszköz energia mérésére
- Magasabb rendű függvények
  - Használatuk listák esetén növelte az energiafogyasztást, néha jelentősen, néha csak minimális mértékben.
- Lista vagy tömb
  - A listák hatékonyabbnak bizonyultak
  - Tömböknél nem jó, ha sok definiálatlan elem van
- Párhuzamosítás
  - Brute force párhuzamosítás esetén több energiát fogyaszt, de a folyamatok számának limitálásával nagyobb hatékonyság érhető el
  - Üzenetek száma és mérete

Eredmények és tervek a jövőre

- Egy folyóiratcikk elbírálás alatt (Studia Universitatis Babes-Bolyai)
- Egy konferencia absztrakt elfogadva (MaCS'18)

- További algoritmusok mérése
- Párhuzamosítások további vizsgálata
- RefactorErl
  - Automatizált refaktorálás energiafogyasztás minimalizálása érdekében

KÖSZÖNJÜK A FIGYELMET!