Becsavarodó csillagkettősök jeleinek keresése GPU-val

GPU nap 2011

Somhegyi Benjámin

KFKI RMKI Virgo csoport

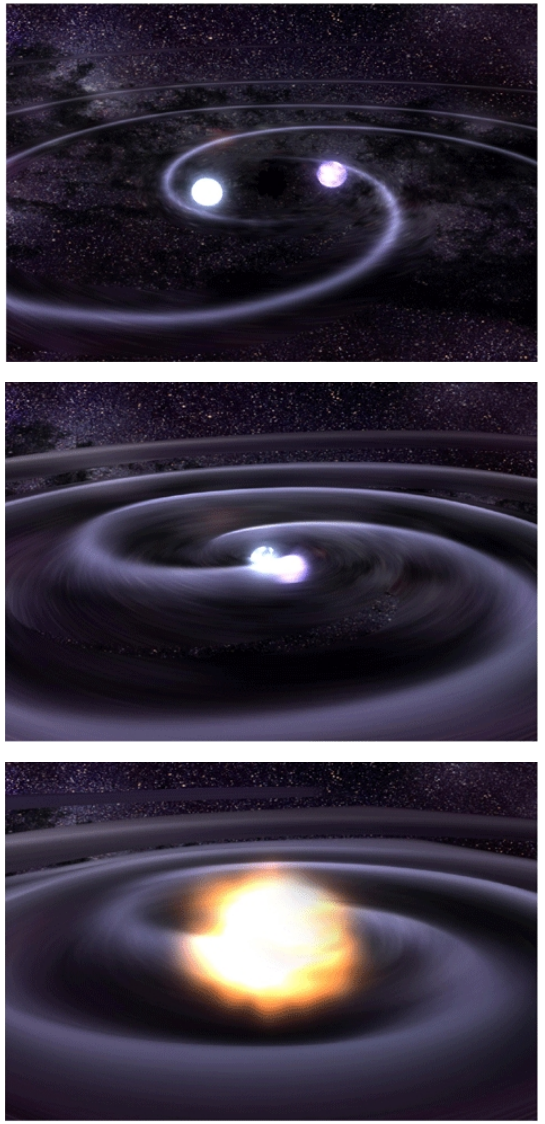
2011. július 8.

Tartalom

1. Becsavarodó csillagkettősök jelei és keresésük
2. A GPU architektúra és programozása
3. Az eredmények ismertetése és értékelése

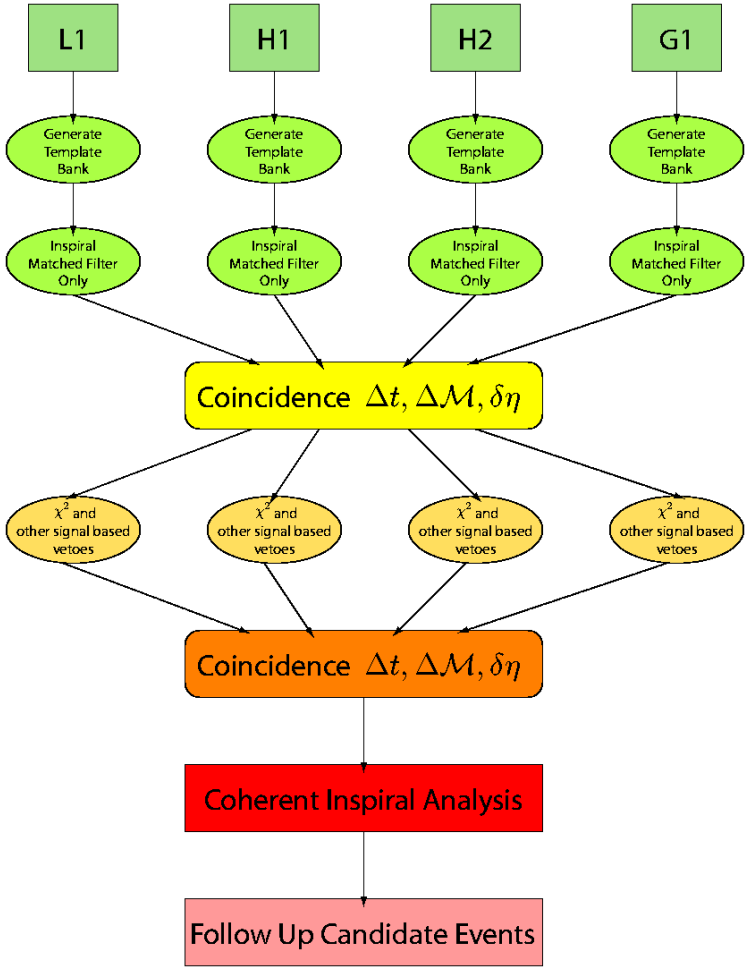
A becsavarodó csillagkettősök

- A legígéretesebb jelforrások egyike
- Több millió évig is eltart a bespirálozódás
- A mai detektorok legfeljebb csak az utolsó néhány percet érzékelik
- A kibocsátott hullám formája, frekvenciája, nagysága függ
  - a csillagok tömegétől, azok eloszlásától
  - forgási sebességétől (spin)
  - a forgástengelyek pályasíkkal bezárt szögeitől
  - a detektor és a kettős távolságától
  - és relatív helyzetétől

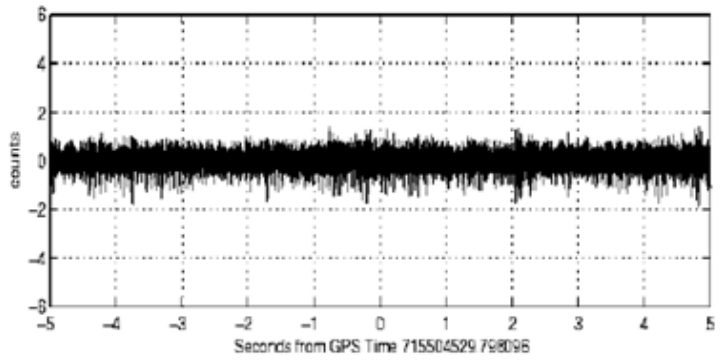


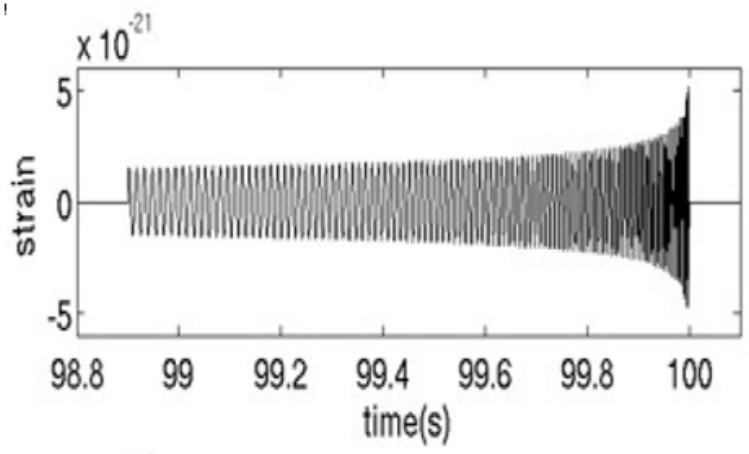
Jelkeresés

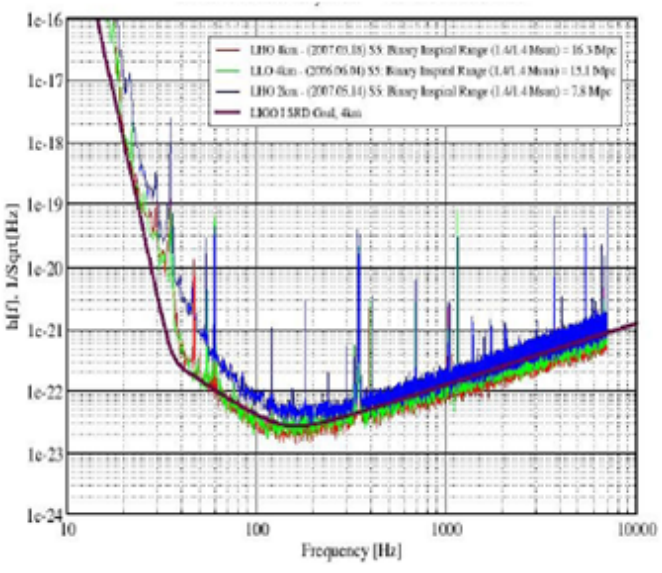
- Konzisztens matematikai-fizikai módszerek a gravitációs hullám formáinak leírására
- Az adatanalízis lényegében: ismert szignálok keresése zajos adatsorban
- A jelenlegi eljárás lépései:
  - Template bank generálás
  - *Matched filtering*
  - Koincidencia vizsgálat
  - Trigbank
  - *Matched filtering vétókkal*
  - Koincidencia, koherencia v.

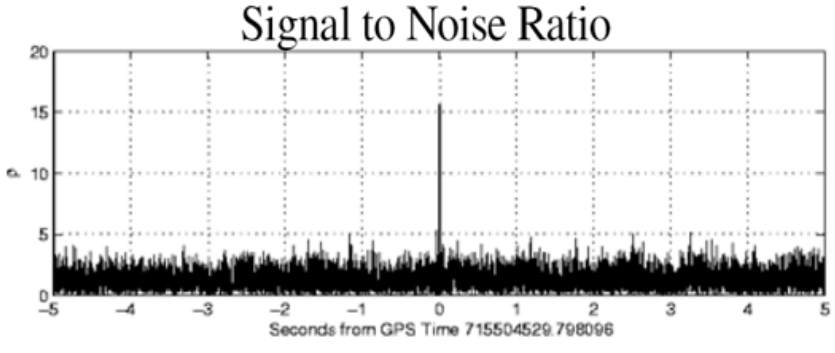


Matched filtering (és χ^2 vétó)









Jelkeresés

- Nagyon sok releváns template konfiguráció vizsgálata szükséges
  - még a fizikailag egyszerűbb eseteknél is
  - Óriási paramétertér (template bank méret)
  - Az analízis folyamat rendkívül számításigényes

Jelkeresés

- Nagyon sok releváns template konfiguráció vizsgálata szükséges
  - még a fizikailag egyszerűbb eseteknél is
  - Óriási paramétertér (template bank méret)
  - Az analízis folyamat rendkívül számításigényes
- Viszont, a folyamat jól párhuzamosítható
  - paramétertér szintjén
  - adatsorok szintjén
  - „csővezeték” (pipelining) feldolgozás

Jelkeresés

- Nagyon sok releváns template konfiguráció vizsgálata szükséges
  - még a fizikailag egyszerűbb eseteknél is
  - Óriási paramétertér (template bank méret)
  - Az analízis folyamat rendkívül számításigényes
- Viszont, a folyamat jól párhuzamosítható
  - paramétertér szintjén
  - adatsorok szintjén
  - „csővezeték” (pipelining) feldolgozás
  - *Műveletek (számítások) párhuzamos elvégzése*

Jelkeresés

Eddigi eredmények

- A sokmagos architektúrák alkalmasak a számításigényes lépcsők párhuzamosítására és gyorsítására
- S. Chung et al.: matched filtering és χ^2 vétó GPU-n történő gyorsítása
  - NVidia CUDA FFT integrációja
  - a jelkeresési lépés $4 - 16\times$ gyorsulása

Jelkeresés

Eddigi eredmények

- A sokmagos architektúrák alkalmasak a számításigényes lépcsők párhuzamosítására és gyorsítására
- S. Chung et al.: matched filtering és χ^2 vétó GPU-n történő gyorsítása
  - NVidia CUDA FFT integrációja
  - a jelkeresési lépés $4 - 16\times$ gyorsulása

Célkitűzések

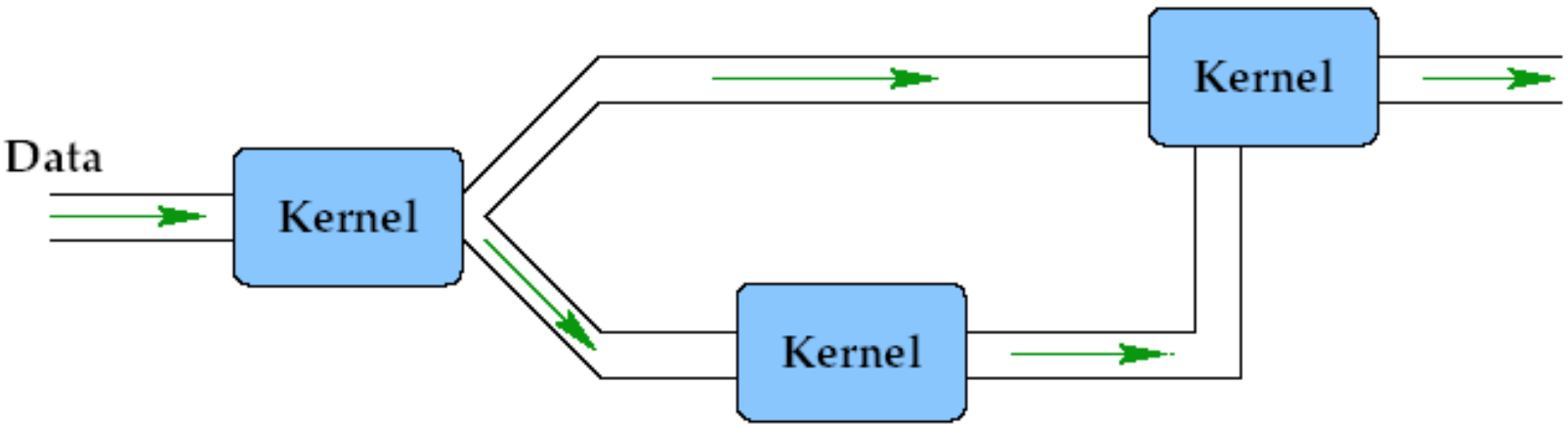
1. A GPU architektúrának, hatékony programozásának tanulmányozása
2. A matched filtering eljárás, valamint a χ^2 vétó olyan GPU párhuzamosítása, amely
   - a számítások összes lépéseire kiterjed,
   - platform független,
   - effektíven támogatja a GPU klasztereket

A GPU architektúra

- Streaming multiprocesszorok (SM), warp-ok
- SIMT (Single Instruction Multiple Thread)
- Memória-modell
- A hatékony végrehajtáshoz sokrétű optimalizáció:
  - Kihasználtság, késedelmi idők elfedése
  - Memória-teljesítmény maximalizálása
  - Warp-on belüli végrehajtási divergencia

GPU-k programozása

- Adatfolyam feldolgozási paradigma (Stream processing)
- A GPU-n hatékonyan elvégezhető adatsor feldolgozási problémákra jellemző:
  - **Adat párhuzamosság**
  - **Aritmetikai sűrűség**
  - **Adat lokalitás**
- OpenCL szabvány

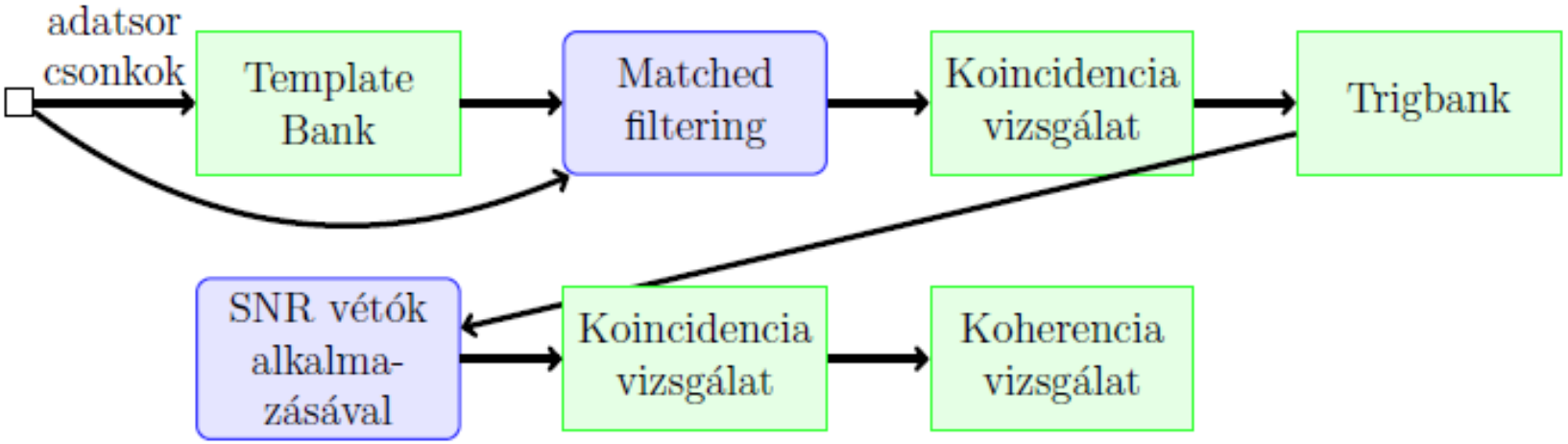


GPU-k programozása

- Alapvető kernel műveleti osztályok
  - Leképezés (map)
  - Redukálás (reduce)
  - Szórás és gyűjtés (scatter and gather)
  - Prefix-összeg képzés (prefix-sum scan)
  - Szűrés (filter)
  - Rendezés (sort)
  - Keresés (search)
- Rengeteg probléma visszavezethető ezekre a számítási típusokra
- Hatékony GPU megvalósításuk kutatása és fejlesztése fontos

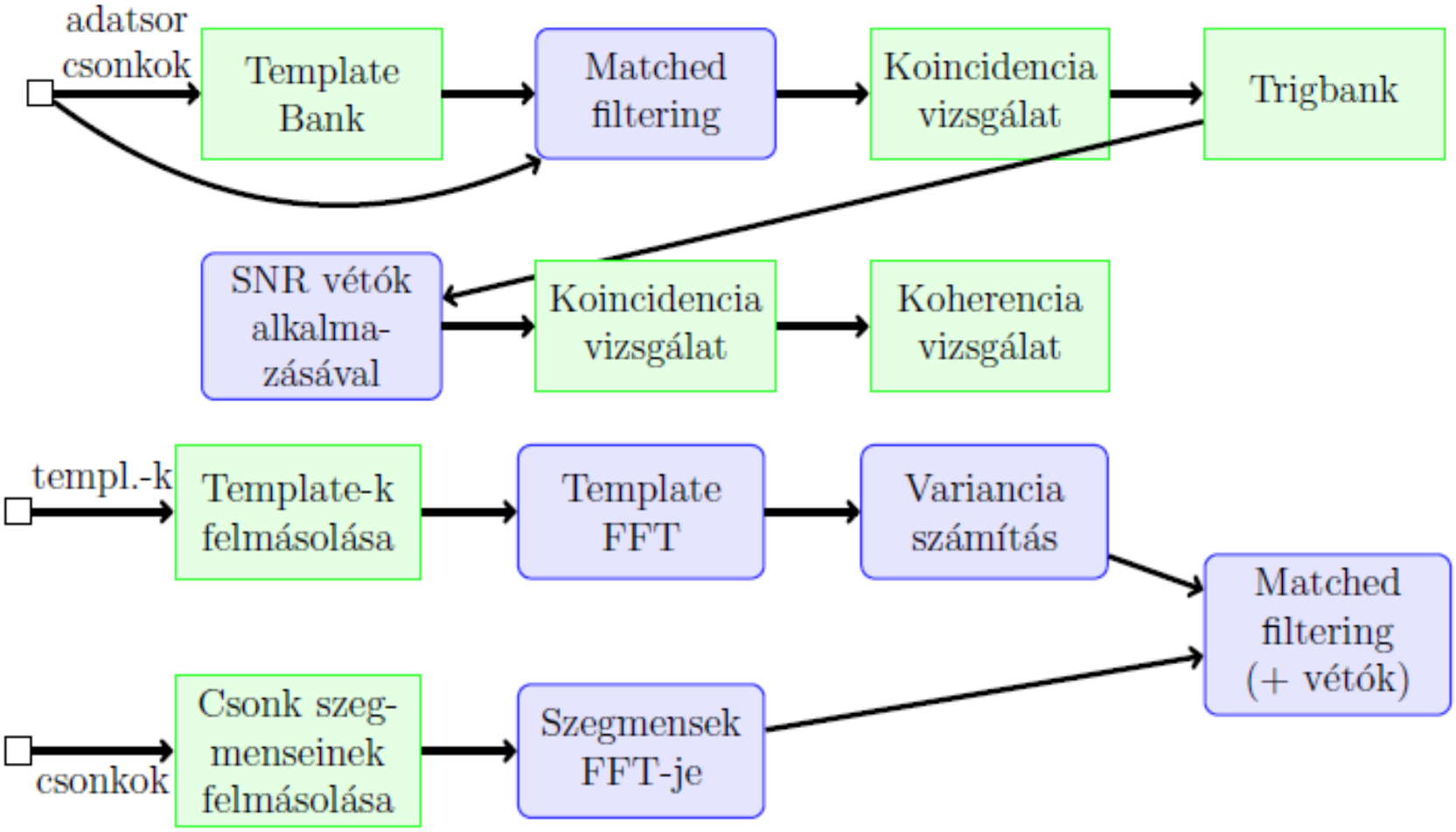
Megvalósítás

- A GPU-n futó lépcsők integrálása az adatanalízis pipeline-ba



Megvalósítás

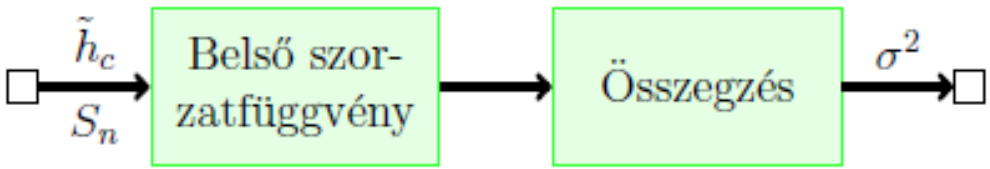
- A GPU-n futó lépcsők integrálása az adatanalízis pipeline-ba



- Gyors Fourier transzformáció (FFT)
  - Volkov et al., Govindaraju et al.
  - Több változat a problémamérettől függően, algoritmikusan eldönthető optimális DFT lebontás!

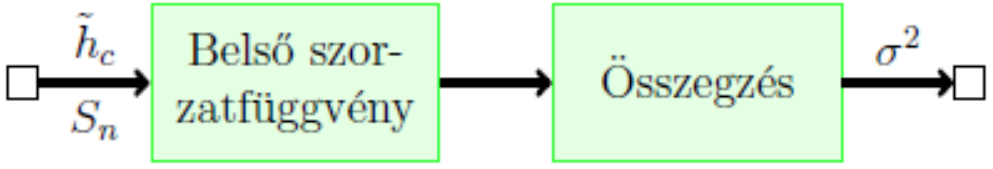
- Gyors Fourier transzformáció (FFT)
  - Volkov et al., Govindaraju et al.
  - Több változat a problémamérettől függően, algoritmikusan eldönthető optimális DFT lebontás!
- Variancia számítás

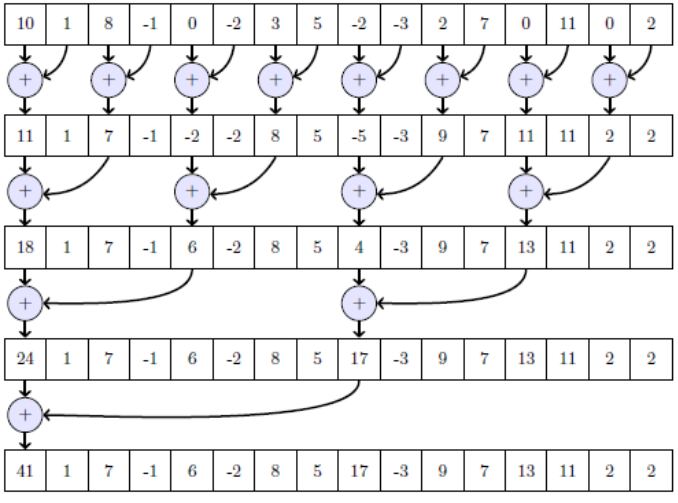
$$\sigma^2 = 4\frac{\Delta t}{N}\sum_{k=0}^{N/2-1}\frac{\tilde{h}_{ck}\tilde{h}^*_{ck}}{S_n(|f_k|)}$$

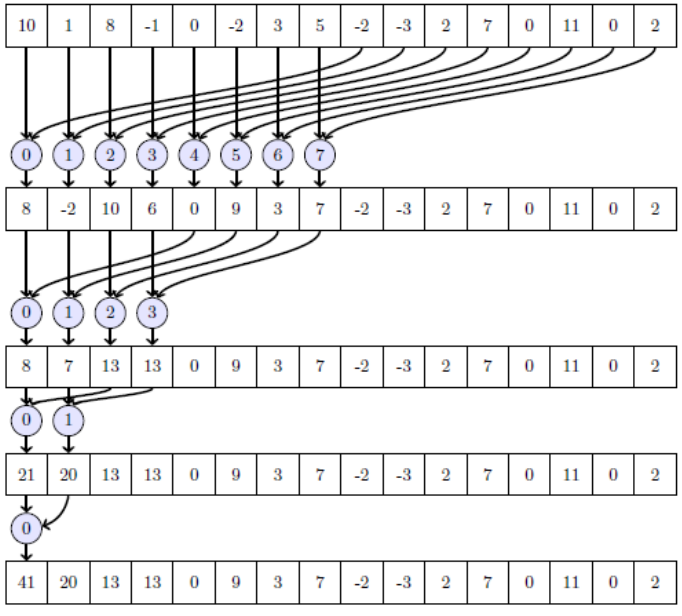


- Gyors Fourier transzformáció (FFT)
  - Volkov et al., Govindaraju et al.
  - Több változat a problémamérettől függően, algoritmikusan eldönthető optimális DFT lebontás!
- Variancia számítás

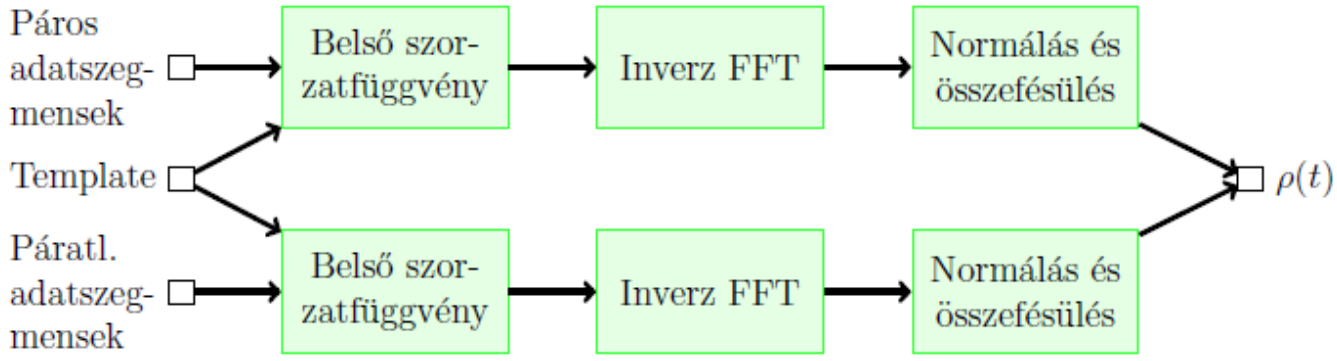
$$\sigma^2 = 4\frac{\Delta t}{N}\sum_{k=0}^{N/2-1}\frac{\tilde{h}_{ck}\tilde{h}^*_{ck}}{S_n(|f_k|)}$$



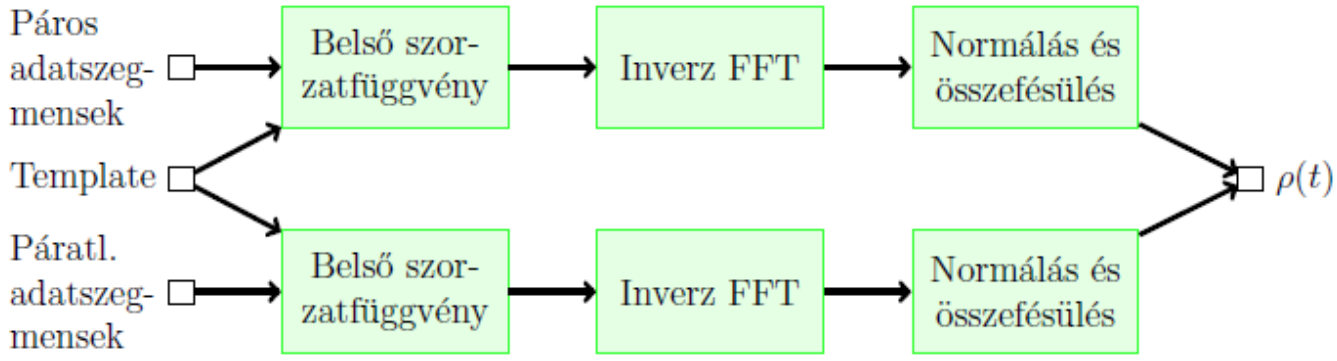




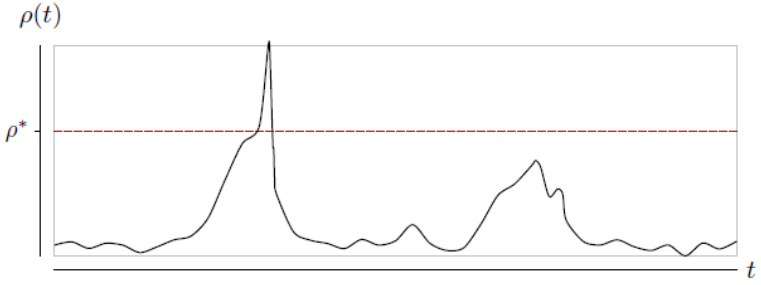
- Matched filtering



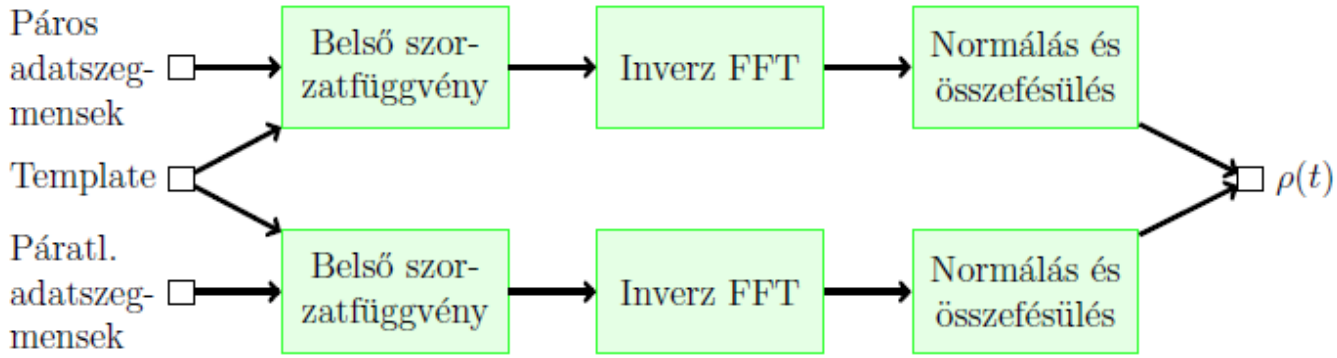
- Matched filtering



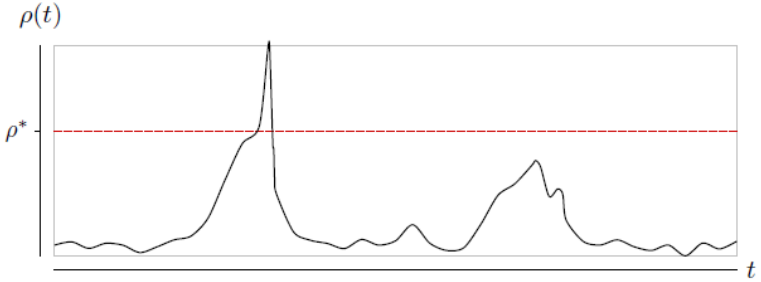
- Küszöbölés



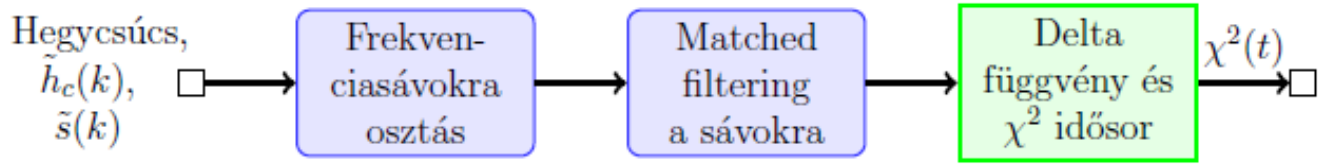
- Matched filtering



- Küszöbölés



- χ^2 vétó



Az eredmények értékelése

Integráció

- Támogatja a paramétertér párhuzamos feldolgozhatóságát (GPU klaszterek)
- Minimalizálja a GPU–CPU kommunikációt

Az eredmények értékelése

Integráció

- Támogatja a paramétertér párhuzamos feldolgozhatóságát (GPU klaszterek)
- Minimalizálja a GPU–CPU kommunikációt

Megvalósítás

- A keresési lépések teljes egészét adat- és művelet-párhuzamosan végzi el sokmagos architektúrán
- A rész-számítások mindegyike effektíven gyorsított
- Platform-független OpenCL-t használó C/C++ programkönyvár

Az eredmények értékelése

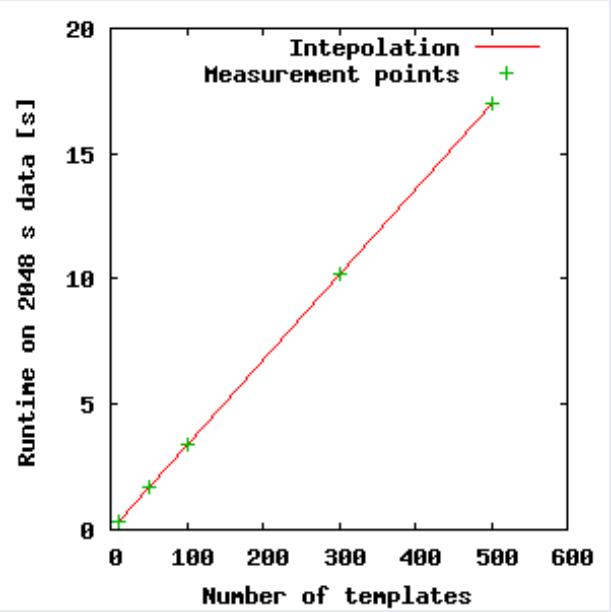
Hatékonyság

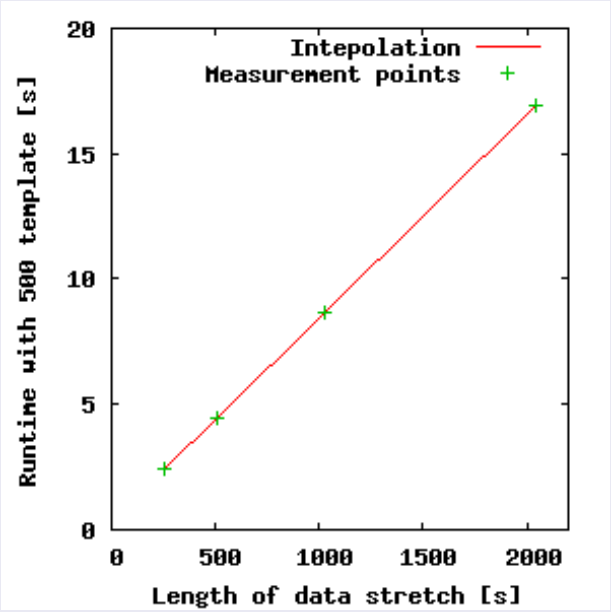
- 2048 sec. hosszúságú adatcsonk (31 db. 128 sec. adatszegmens)
- 500 template, max 64 sec. hossz
- A feldolgozás futásideje CPU-n $\sim$ 33 perc, a GPU-n $\sim$ 18 másodperc
- **két nagyságrendbeli gyorsulás ($\sim 100\times$)!**

Az eredmények értékelése

Hatékonyság

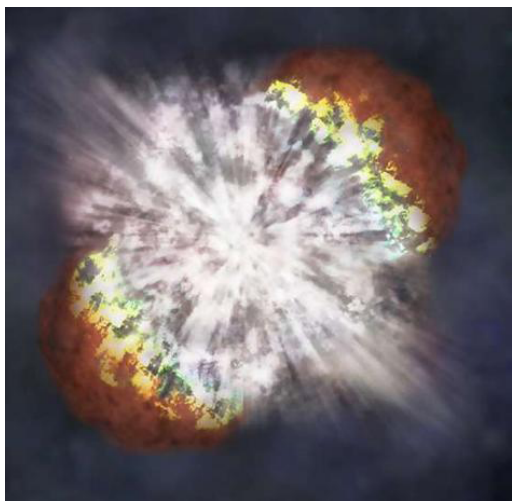
- 2048 sec. hosszúságú adatcsonk (31 db. 128 sec. adatszegmens)
- 500 template, max 64 sec. hossz
- A feldolgozás futásideje CPU-n $\sim$ 33 perc, a GPU-n $\sim$ 18 másodperc
- **két nagyságrendbeli gyorsulás ($\sim 100\times$)!**

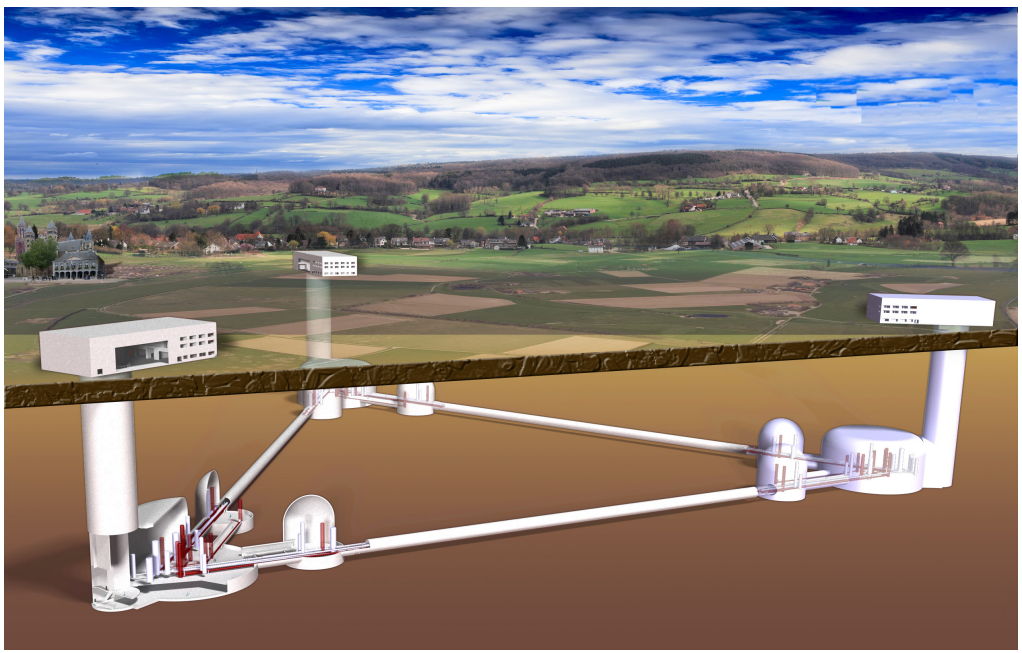




Miért fontos az eredmény?

- Nagyságrendekkel több releváns template-t tudunk vizsgálni
- Lehetővé teszi az alacsony késedelmi idejű feldolgozást
- A GPU-k fejlődési trendje ígéretes: ideális eszközök lesznek a jövő detektorainak adatanalíziséhez





Vége

Köszönöm a figyelmet!