

API alapú tőzsdei adatgyűjtés időzítési kihívásai és megoldásai

Kodolányi János Egyetem

Üzemmérnök-informatikus szak

**Bevezetés:**
Szakdolgozatom során egy olyan rendszert fejlesztettem, amely különböző tőzsdei mutatók alapján próbál előrejelzést adni a bitcoin árfolyamának alakulására. A rendszer lelke a különböző pénzügyi indikátorok, például az RSI, EMA vagy a Supertrend folyamatos lekérdezése és tárolása egy szerveren keresztül. A célom az volt, hogy ezek az adatok perces időközönként frissüljenek, így biztosítva a modellek számára egy pontos és szabályos időalapú adatsort.

**Probléma felvetése:**
A fejlesztés előrehaladtával azonban egy váratlan, de annál fontosabb problémára lettem figyelmes: az adatbázisban szereplő időbélyegek nem pontosan 60 másodpercenként követték egymást, hanem apró elcsúszások jelentkeztek. Ez elsőre jelentéktelennek tűnhet, hosszabb távon azonban komoly problémákhoz vezethet. Az idősor szépen lassan "megcsúszott", és a különböző mutatók már nem pontosan ugyanabban az időpillanatban álltak rendelkezésre.

**Tesztelési környezet beállítása**

* **Szoftver:** Python.
* **API:** Taapi.io API a különböző tőzsdei mutatók (RSI, EMA, Supertrend stb.) lekérdezéséhez
* **Lekérdezések:** 1 perces időintervallumonkénti adatlekérés (BTC/USDT – Binance tőzsde)
* **Cél:** Biztosítani, hogy az adatok pontosan egy perces időközönként frissüljenek

•  **Körülmények:** A teszt stabil internetkapcsolat mellett történt, valós idejű működés közben, a válaszidő és időzítés pontosságának mérésére fókuszálva.



A tesztelési környezet kialakításakor fontos szempont volt a rendszer valós idejű működésének biztosítása, mivel a bitcoin piacának elemzése időérzékeny adatokat igényel. Az adatgyűjtés során alapvető fontosságú, hogy a lekérdezések pontosan, egyenlő időközönként érkezzenek, hiszen bármilyen időzítési hiba torzíthatja az előrejelzések megbízhatóságát. A tesztelés célja az volt, hogy a rendszer valódi működése során mérjük a válaszidők és az időbeli eltérések hatását, és feltérképezzük, hogy miként befolyásolják a tőzsdei mutatók pontos frissítését. Az API-k és az adatbázis kezelésének szinkronizálása érdekében különös figyelmet fordítottam a lekérdezések időzítésére és az adatbázisban tárolt időbélyegek pontosságára.

**Teszt eset:**

| **Sorszám** | **Időbélyeg** | **Eltérés az ideális 60 mp-től (mp)** | **Összesített drift (mp)** |
| --- | --- | --- | --- |
| 1 | 2025-05-03 10:19:19 | – | 0 |
| 2 | 2025-05-03 10:20:19 | 0 | 0 |
| 3 | 2025-05-03 10:21:20 | +1 | 1 |
| 4 | 2025-05-03 10:22:23 | +3 | 4 |
| 5 | 2025-05-03 10:23:25 | +2 | 6 |
| 6 | 2025-05-03 10:24:27 | +2 | 8 |
| 7 | 2025-05-03 10:25:29 | +2 | 10 |
| 8 | 2025-05-03 10:26:31 | +2 | 12 |
| 9 | 2025-05-03 10:27:32 | +1 | 13 |
| 10 | 2025-05-03 10:28:34 | +2 | 15 |



**Tesztelési eredmény értékelése:**

Rövid távú hatások:

A teszt alapján 10 perc alatt 15 másodperc drift keletkezett, amely rövid távon nem jelentős.

Hosszú távú hatások:

A drift hosszú távon problémákat okozhat. Ha napi 36 perc drift keletkezik, akkor egy év alatt akár 9 napnyi (216 órás) drift is kialakulhat.

Ennek hatására az adatgyűjtés és -feldolgozás pontossága csökken, ami hibás előrejelzéseket eredményez.

**API Válaszidő mérése adatfeldolgozás nélkül:**

A válaszidő mérése fontos része az API teljesítményének értékelésében, mivel az adatgyűjtés hatékonyságát jelentősen befolyásolja. Az API válaszidejét egy 100 kérésből álló teszt során mértem, ahol minden kérés között 15 másodperc várakozást alkalmaztam. A mérés során nem történt adatfeldolgozás, adatoptimalizálás vagy CSV-be helyezés, csupán maga az API lekérés válaszidejét mértem.

A teszt során tapasztalt válaszidők 0.13–0.14 másodperc között mozogtak. Ez az eredmény azt mutatja, hogy az API gyors válaszokat biztosít, így a hiba nem az adatgyűjtési folyamatokban keresendő.

**Optimalizációk:**

A problémák kiküszöbölésére több optimalizációt alkalmaztam a rendszerben. Az API-hívások végrehajtása párhuzamos szálakon történik, amely lehetővé teszi, hogy az adatgyűjtés jelentősen gyorsabbá váljon. Ezen felül a dinamikus időzítési rendszer biztosítja, hogy a lekérdezések pontosan 60 másodpercenként történjenek, és ha valamelyik futtatás túl hosszú ideig tartana, akkor az ütemezés kompenzálja azt, biztosítva a pontos időintervallumokat. Mindezek mellett beállítottam időtúllépési korlátot minden API-kéréshez.

**Optimalizáció utáni teszt:**

| **Sorszám** | **Időbélyeg** | **Eltérés az ideális 60 mp-től (mp)** | **Összesített drift (mp)** |
| --- | --- | --- | --- |
| **1** | **2025-05-03 15:41:22** | **–** | **0** |
| **2** | **2025-05-03 15:42:22** | **0** | **0** |
| **3** | **2025-05-03 15:43:22** | **0** | **0** |
| **4** | **2025-05-03 15:44:22** | **0** | **0** |
| **5** | **2025-05-03 15:45:22** | **0** | **0** |
| **6** | **2025-05-03 15:46:22** | **0** | **0** |
| **7** | **2025-05-03 15:47:22** | **0** | **0** |
| **8** | **2025-05-03 15:48:22** | **0** | **0** |
| **9** | **2025-05-03 15:49:22** | **0** | **0** |
| **10** | **2025-05-03 15:50:22** | **0** | **0** |



**Összegzés:**

A tőzsdei mutatók perces lekérdezésénél észlelt másodperces eltérések hosszabb távon komoly időeltolódásokhoz vezettek, ami torzította volna az előrejelzések megbízhatóságát.

A kezdeti tesztelés során megállapítottam, hogy akár 10 perc alatt is 15 másodpercnyi időeltolódás keletkezhet, ami hosszú távon évi több napos drift kialakulásához vezethet. Ez elfogadhatatlan pontatlanságot eredményezne egy időérzékeny pénzügyi elemző rendszerben.

A probléma megoldására több optimalizációt vezettem be:

* Párhuzamos szálak alkalmazása az API-hívások végrehajtására
* Dinamikus időzítési rendszer, amely biztosítja a pontos 60 másodperces frissítési ciklusokat
* Időtúllépési korlátok beállítása minden API-kéréshez

Az optimalizációk után végzett tesztelés egyértelműen igazolta a módosítások sikerességét: a rendszer képes volt tíz egymást követő lekérdezést pontosan egyperces időközönként végrehajtani, nulla másodperces drift mellett.

Ez a gyakorlati tapasztalat rávilágított az időzítéssel kapcsolatos problémák fontosságára és jelentős hatására a valós idejű adatgyűjtési rendszerekben. Az alkalmazott megoldások nemcsak a jelenlegi projektemben hasznosak, hanem hasonló időérzékeny adatgyűjtési és -elemzési rendszerekben is alkalmazhatók, jelentősen növelve az előrejelzések pontosságát és megbízhatóságát.

A tesztelés és optimalizálás eredményeként egy stabilabb, megbízhatóbb rendszert sikerült kialakítanom, amely megfelelő alapot nyújt a pontosabb bitcoin árfolyam-előrejelzésekhez, és szemlélteti a rendszeres, alapos szoftvertesztelés fontosságát.