# Iteratív bekezdés- és mondatszinkronizáció

Pohl Gábor

Pázmány Péter Katolikus Egyetem
Információs Technológiai Kar
1083 Budapest, Práter utca 50/A
pohl@morphologic.hu

**Kivonat** A gépi szövegszinkronizáció (*text alignment*) célja egy forrásnyelvi szövegen és fordításán belül az egymás fordításának tekinthető szövegegységek automatikus összerendelése. Korábbi munkáinkban [5][6] bemutattunk egy hibrid szövegegységhosszakon és statisztikai módszerekkel szűrt horgonyokon alapuló megoldást, amellyel a csak szövegegységhosszakat összehasonlító módszernél jobb bekezdés- és mondatszinkronizációs eredmények érhetők el. Most a gépi szinkronizáció emberi javításának kérdéskörét járjuk körül, valamint bemutatjuk, hogy a szinkronizációs folyamat iteratívvá tételével hogyan csökkenthető az az emberi beavatkozások száma. Az újonan bemutatott módszer nagy szinkronizálási feladatok (például kétnyelvű korpuszok párhuzamosítása) során, illetve fordítómemóriák nagy mennyiségű szinkronizálatlan szöveggel való feltöltése esetén válik különösen hasznossá.

## 1. Bevezetés

A gépi szövegszinkronizáció (*text alignment*) célja egy forrásnyelvi szövegen és fordításán belül az egymás fordításának tekinthető szövegegységek automatikus összerendelése. Egy szövegpár mondatszintű szinkronizációja lehetővé teszi a szövegpár fordítómemóriába töltését, a fordítás terminológiai konzisztenciájának ellenőrzését, párhuzamos korpuszok létrehozását. Ugyanakkor csupán gépi eszközökkel – a feladat nehézségből adódóan – az esetek többségében nem érhető el teljesen pontos mondatszintű szinkronizáció[1], azaz a szinkronizáció emberi javítása szükséges. Az automatikus szinkronizációt megvalósító megoldások célja az eddigiekben minél tökéletesebb automatikus szinkronizáció előállítása volt, azonban egy szinkronizáló alkalmazás felhasználójának szemszögéből nézve más a cél: minél kevesebb és egyszerűbb emberi beavatkozással tökéletes szinkronizációt létrehozni.

A következőkben a fordítások szinkronizációt megnehezítő tulajdonságait, illetve az egyes szinkronizációs módszerek hibalehetőségeit járjuk körül, majd bemutatjuk a minimális emberi beavatkozás elvét szem előtt tartó iteratív szövegszinkronizáló megoldásunkat.

[1] Szinkronizációnak nevezzük a szövegegységek összerendelésének folyamatát és a folyamat eredményét is.

## 2. Az automatikus szinkronizáció nehézségei

Az automatikus szinkronizációt megnehezítő tényezők két csoportját különböztethetjük meg. Az egyik csoportba a fordítás, illetve annak szerkesztése során létrejött változtatásokat vehetjük fel, a másik csoportba a különböző nyelvű szövegek automatikus szinkronizációjához szükséges nyelvi előfeldolgozás hibái sorolhatók.

### 2.1. Változtatások a fordítás és a szöveg szerkesztése során

A fordítók (illetve a fordítás szerkesztői) megváltoztathatják a szöveg szegmentációját, elhagyhatnak, beszúrhatnak, összevonhatnak, szétbonthatnak szövegegységeket (mondatokat, bekezdéseket) a fordítás során. A szerkesztés során – ritka esetekben – az egyes szövegegységek sorrendje is változhat (pl. alfabetikusan rendezett felsorolások).

### 2.2. Az automatikus nyelvi előfeldolgozás problémái

A bekezdések határai a különböző dokumentumformátumokban általában pontosan rögzítettek; gondot az okoz, hogy az egyes nyelvekhez készített automatikus mondathatár-meghatározó rendszerek többnyire a szövegpár különböző pontjain hibáznak. Ezeket a hibákat a gépi szinkronizáció során – a fordítói szegmenshatár változtatásokhoz hasonlóan – kompenzálni kell.

Horgonykereső módszereknél (lásd később) a horgonyok keresése során esetlegesen alkalmazott (morfológián alapuló) tövesítésnek a különböző nyelvek esetében összevethetőnek kell lennie.

## 3. Szövegszinkronizációs módszerek és hibalehetőségeik

A szövegszinkronizációs módszerek ismertetésével korábbi munkáinkban [5][6] részletesen foglalkoztunk, most csak hibalehetőségeik szempontjából mutatjuk be röviden az egyes módszertípusokat.

### 3.1. Szövegegységek hosszát összehasonlító módszerek

A szövegegységhosszakon alapuló módszerek a különböző nyelvű szövegek szövegegységeinek hosszait hasonlítják össze, többnyire Gale és Church valószínűségi modelljére [1] alapozva. Az alkalmazott modell azt feltételezi, hogy a szövegegységek sorrendje nem változik a fordítás során, így ilyen esetekben, illetve beszúrások és elhagyások esetén is, többnyire hibás eredmény várható.

A módszer előnye, hogy egy dinamikus programozás típusú algoritmus alkalmazásával globálisan optimális megoldást keres, többszöri (ismételt) futtatás során mindig azonos eredményre jut. A hibák többnyire lokálisak, hibázás esetén nem a teljes eredmény, csak egyes pontjai lesznek hibásak. Ezzel együtt a hibák

csomósodása is megfigyelhető, a hibát csak újabb – általában közeli ponton megtalálható – hibával tudja korrigálni a globális optimumkereső eljárás.

A szinkronizáció során biztos pontot jelentenek a szinkronizálandó szegmensek végpontjai; a szövegpárt kisebb szegmensekre bontva a módszer megbízhatósága nő, ezért célszerű a szinkronizációt először bekezdésszinten, majd a szinkronizált bekezdéseken belül mondatszinten meghatározni, illetve ezért alkalmazható sikeresen a rövidesen bemutatott iteratív javítást lehetővé tevő módszerünk.

### 3.2. Horgonykeresés

A horgonykereső eljárások nem törekednek teljes szinkronizációra, azaz a szinkronizálandó szövegpárnak csak egyes kitüntetett pontjait kívánják horgonyokkal összekapcsolni. Horgonyokkal csak akkor lehetne teljes szinkronizációt megvalósítani, ha minden szövegegységet az összes párjával (és lehetőség szerint csak azokkal) horgonyok kötnének össze. Ilyen magas lefedettség (*recall*) és pontosság (*precision*) elérése azonban nem lehetséges.

A horgonyjelöltek statisztikai szűrésére alkalmazott módszerek [7] is feltételezik, hogy a szövegegységek sorrendje nem változik (sokat) a fordítás során. A statisztikai szűrők többnyire kizárják a megváltoztatott pozíciójú szövegegységek horgonyjelöltjeit, ami a horgonyok számának csökkenésével jár. Beszúrások és elhagyások esetében, a horgonyok számának csökkenésével járhat, ha az érintett szövegegység horgonyjelöltet is tartalmaz, mivel csak azok a horgonyjelöltek választhatók ki horgonyként, amelyekből azonos számú szerepel a szövegpár két oldalán.

### 3.3. Hibrid megoldás

Az általunk korábban bemutatott hibrid, statisztikailag szűrt horgonyokon és szövegegységhosszakon alapuló módszer [6] Gale és Church módszeréhez hasonlóan dinamikus programozásra építve globálisan optimális megoldást keres, amelynek pontosságát a horgonyhasználat növeli[2]. A következőkben azt mutatjuk be, hogyan alkalmazható ez a szinkronizáló motor (vagy más hasonló megoldás), ha a szinkronizáció folyamatát iteratívvá tesszük.

## 4. Iteratív szövegszinkronizáció

A jelenleg elérhető (többnyire fordítómemória alkalmazások részeként vagy kiegészítőjeként értékesített) szövegszinkronizáló eszközökben a szinkronizáció folyamata többnyire egy automatikus szinkronizációt meghatározó lépésből, majd ennek emberi javításából áll. Egyes eszközökben lehetőség van az utolsó kijavított hiba (vagy tetszőleges más szövegpont) utáni szövegrészek gépi újraszinkronizálására. Ezzel a szinkronizáció folyamatát iteratívvá tették, ugyanakkor

[2] Horgonyok hiányában a módszer Gale és Church módszerével azonos.

bizonyos típusú javítások nehézkesek maradtak (pl. a szövegegységek sorrendjének változásait követő összeköttetések felvétele). A szöveg csak lineáris javítását megengedő módszer előnye, hogy viszonylag kevés fejlesztéssel megvalósítható, ugyanakkor az emberi javítások információtartalma a gépi újraszinkronizálás során ennél jobban is kihasználható.

### 4.1. Iteratív javítás szegmentáló és kiemelő összerendelésekkel

Az általunk kidolgozott iteratív szinkronizációt lehetővé tevő rendszerben a gépi szinkronizáló kimenetét a korrektor úgy javítja, hogy szinkronizációs összerendeléseket határoz meg, amelyek a következő szinkronizálási ciklusban a gép által megváltoztathatatlan, a szinkronizáció során támpontként használható összerendelések lesznek. A korrektor kétféle összerendelés típust határozhat meg: szegmentálót és nem szegmentálót (kiemelő).

Szegmentáló összerendelés esetén a szinkronizáló program a szövegpárt az összerendelés előtti és utáni szinkronizálandó szegmenspárra bontja, amelyek szinkronizációja kisebb méretük miatt egyszerűbb, illetve gyorsabban megoldható feladat. (A szegmenshatárok a szövegegységhosszakon alapuló módszernél támpontként szolgálnak a szinkronizáció során.)

Nem szegmentáló, azaz kiemelő összerendelések esetén a program az érintett szövegegységeket a szövegből „kiemelve" végzi el a szinkronizációs folyamatot, azaz ilyenkor nem bontja kisebb szinkronizálandó szegmenspárokra a szövegpárt. Ezzel a fordítás során áthelyezett, beszúrt vagy elhagyott szövegrészek okozta szinkronizációs hibák javíthatók. A kiemelő összeköttetéssel megjelölt szövegrészeket a horgonykeresés során is el lehet távolítani a szövegpárból, így horgonyjelöltek hibás felvételét is kiküszöbölhetjük.

### 4.2. Több hiba javítása egyszerre

Több javító összerendelés is felvehető a szinkronizáló algoritmus újrafuttatása előtt, azonban meg kell tiltani egymást keresztező szegmentáló összerendelések felvételét, illetve fel kell készíteni a rendszert arra az esetre, ha a szövegpár egyik oldalán a szegmentáló összerendelések szövegegységei összeérnek, míg a másik oldalon más (beszúrt vagy elhagyott) szövegegységek ékelődnek közéjük. (Ez utóbbi szövegegységeket ilyenkor beszúrtként vagy elhagyottként kell megjelölni.) Hasonló a helyzet, ha a szinkronizálandó szegmenspár első vagy utolsó szövegegységeit tartalmazza szegmentáló összerendelés.

Az ismertetett javítási módszer előnye, hogy egyes hibák javítása több másik hiba automatikus javítását eredményezheti. Egy hibásan szinkronizált szövegrészben (a hibák csomósodást mutatnak) többnyire egyetlen szegmentáló javítás felvételével helyreállítható a szinkronizáció.

### 4.3. A szinkronizáló motorral szembeni elvárások

Az előzőekben bemutatott módszer egyszerűnek tűnik, azonban a használt szinkronizáló motorral szemben követelményeket támaszt. A felhasználó ugyanis joggal várhatja el, hogy amit egyszer már jól szinkronizált a program, egy későbbi

iteráció során akkor se rontsa el, ha ő nem rögzítette azt külön (költséges emberi munkával). Ez többnyire teljesül, mivel a kisebb szegmenspárok szinkronizálása egyszerűbb feladat, azonban a teljes sikerhez az is szükséges, hogy a szinkronizáló algoritmus azonos bemenet esetén mindig azonos eredményre jusson. Korábban egyes szerzők (több okból is sikertelenül [5]) próbálkoztak véletlent használó, illetve csak lokális optimumot kereső eljárással [2], ilyen szinkronizáló algoritmus azonban nem alkalmazható iteratívan.

A dinamikus programozást alkalmazó algoritmusok előnye, hogy ha a szinkronizálandó szövegpár egy adott pontján az algoritmus által helyesen felvett párt rögzítünk, és az így kapott kisebb szegmenspárokon elvégezzük a gépi szinkronizációt, akkor az előzővel azonos eredményt fogunk kapni. Tehát az iteratív szinkronizáció során helyes összerendelések rögzítésével az eredményt nem lehet elrontani.

Célszerű, ha az alkalmazott szinkronizáló motor lehetővé teszi virtuális (nulla hosszúságú) szegmensek használatát, amelyekkel a több szegmentáló összerendelés felvétele esetén felmerülő kritikus részek egyszerűen, utófeldolgozási lépések nélkül is szinkronizálhatók. (A beszúrt, illetve elhagyott szegmenseket így a szinkronizáló motor is meghatározhatja, nem szükséges ezeket az eseteket külön kezelni.)

### 4.4. Horgonyok keresése az iteratív javítás során

Érdekes kérdés, hogy az iteratív szinkronizáció során a résszegmenseken belül vagy a teljes szövegben (természetesen a kiemelt szövegegységeket kihagyva) érdemesebb-e horgonyokat keresni. Nagyobb szövegrészeket tekintve nagyobb a valószínűsége, hogy egy horgonyjelölt a szövegpár egyik oldalán kevesebbszer fordul elő, mint a másik oldalon. Ilyenkor ezek a horgonyjelöltek nem válhatnak horgonyokká. A szövegpár kisebb szinkronizálandó szegmenseiben viszont lehet, hogy horgonyként jelenhetnek meg ezek a jelöltek is, azaz kisebb szegmenspárokat szinkronizálva a teljes szövegpárt tekintve több horgony található. Ugyanakkor az is elmondható, hogy a rövidebb szegmenspárok esetében a horgonyjelöltek száma is kisebb (az egyes szegmenspárokat tekintve), ami csökkenti a hibás horgonyok szűrésére használt statisztikai módszerek megbízhatóságát.

Jelenleg, mivel csak a lehető legpontosabbnak tekintehő horgonyjelölteket válogatjuk ki a szövegpárból, szövegszinkronizációs megoldásunkban a részszegmensenkénti horgonykiválasztást választottuk. Ezt a módszert azonban a későbbiekben lehet, hogy érdemes lesz kiváltani az egész szövegpárt vizsgáló horgonykereséssel, illetve a két lehetőség kombinációja is szóba jöhet. A teljes szövegpár vizsgálata esetén felmerül, hogy a felhasználó által felvett (vagy helyesnek elfogadott) szinkronizációs összerendelések által lefedett szövegegységeken belüli horgonyjelölteket felhasználjuk-e a statisztikai szűrés során, hiszen bár kicsi a valószínűsége, a felhasználói beavatkozás lehet, hogy pont egy hibásan felismert horgony miatt vált szükségessé.

A részszegmenspárok szinkronizálása során a horgonyjelöltek közül kiválasztott horgonyok eredeti horgonyokhoz képesti megváltozása azt jelenti, hogy a dinamikus programozás bemenete is megváltozik, ezáltal veszélyeztetve az eddig

helyesen meghozott szinkronizációs döntéseket. Azonban azt is feltételezhetjük, hogy a kisebb szegmensben több, a szinkronizációt alapvetően segítő horgony fordul elő, és csak nagyon ritkán találkozunk hibásan felvett horgonnyal.

### 4.5. A szinkronizálási folyamat segítése

Tapasztalataink szerint a korrektor munkáját jelentősen megkönnyíti, ha a szövegpár azon pontjait kiemeljük, amelyeket esetleg helytelenül szinkronizált az automatikus módszer. Ilyen pontok lehetnek a gép által meghatározott (és a korrektor által még el nem fogadott) nem egy-egy típusú összerendelések[3], illetve ezek környezete a szövegpárban. Ezek az összerendelések vagy hibásan jönnek létre, vagy a szövegpár egy kritikus pontjára mutatnak rá.

Hasonlóképpen érdemes kiemelni azokat a szövegegységeket, amelyek gépi szinkronizációja eltér az előző lépésben meghatározottól. Ezek többnyire várható változások, – jobb esetben a hibák eltűnését remélő korrektor pont ezekre a változásokra számít –, ugyanakkor a résszegmenseken belüli eltérő horgonykiválasztás következményei is lehetnek, így célszerű ezekre is felhívni a korrektor figyelmét.

## 5. Eredmények és további célkitűzések

Az iteratív javítás most ismertetett lehetőségét a MorphoLogicnál fejlesztett szövegszinkronizáló rendszerben valósítottuk meg, és teszteltük, egyelőre csak néhány szövegpáron. Az eddigi módszerek (egyszeri javítás, lineáris javítás és újraszinkronizálás) az új megoldással szimulálhatók, így az eddigieknél rosszabb eredményektől nem kellett tartanunk.

A módszer kiértékelésénél gondot okoz, hogy nem mérhető pontosan, hány javítás szükséges a szinkronizáció elvégzéséhez, hiszen a javítások száma attól is függ, hogy a korrektor hova helyez javító összeköttetéseket.

A sorrendcseréket nem tartalmazó szövegpároknál a javítások száma átlagosan kevesebb, mint fele volt a módszert nem használó egyszeri emberi javítás esetében mérhetőnél. Ez annak köszönhető, hogy egy adott hiba javításához az iteratív megoldásnál többnyire elég volt a hibás összerendelés helyett egyetlen jót megadni, míg egyszeri javítás esetén a hibás összerendelés összes szövegegységének helyes összerendelésekbe rendezéséről gondoskodnia kell a korrektornak.

Csomós hibák javításakor még szembetűnőbb a különbség, ugyanakkor a csomós hibákat már a csupán lineáris javítást és újraszinkronizálást lehetővé tevő szinkronizáló eszközökben is könnyen javítani lehetett. A csak lineáris javítást lehetővé tevő módszerekhez képest lényeges javulás olyan esetekben tapasztalható, ahol a szöveg egyes részeit elmozdították a fordítás során.

Szövegszinkronizáló módszerünket a közeljövőben szeretnénk egy komoly szinkronizálási feladat során is kipróbálni: a SZAK Kiadó informatikai témájú,

[3] Automatikusan 1–1, 1–2, 2–1, 2–2, 0–1, 1–0 összeköttetéseket képes meghatározni a rendszer. Természetesen a korrektor manuálisan másmilyen összeköttetéseket (pl. 3–2) is felvehet.

kétnyelvű (angol-magyar), nyelvenként több, mint 1 millió szavas korpuszát [3][4] kívánjuk szinkronizáló alkalmazásunk segítégével párhuzamosítani.

## Hivatkozások


1. Gale, William A.; Kenneth W. Church. A Program for Aligning Sentences in Bilingual Corpora. Computational Linguistics, Volume 19, Number 1, March 1993, Special Issue on Using Large Corpora.
2. Chen, Kuang-hua; Chen, Hsin-Hsi A Part-of-Speech-Based Alignment Algorithm In: *COLING 1994 Volume 1: The 15th International Conference on Computational Linguistics*, 1994.
3. Kis Ádám; Kis Balázs. A Prescriptive Corpus-based Technical Dictionary. In: *Papers in Computational Lexicography: Proceedings of COMPLEX 2003.* Research Institute for Linguistics, Hungarian Academy of Sciences, Budapest, 2003.
4. Kis Balázs; Ugray Gábor. Új korpuszstatisztikai eszköztár kollokációkeresésre. In: *Az I. Magyar Számítógépes Nyelvészeti Konferencia gyűjteményes kötete*, Szegedi Tudományegyetem, Szeged, 2003.
5. Pohl Gábor. Fordítások terminológiai konzisztenciájának vizsgálata. Diplomaterv feladat, Budapesti Műszaki és Gazdaságtudományi Egyetem, Villamosmérnöki és Informatikai Kar, 2003.
6. Pohl Gábor. Szövegszinkronizációs módszerek, hibrid bekezdés- és modatszinkronizációs megoldás. In *Magyar Számítógépes Nyelvészeti Konferencia 2003*, Szegedi Tudományegyetem, Szeged, 2003.
7. Ribeiro, António; Gabriel Lopes; Joao Mexia. Using Confidence Bands for Parallel Texts Alignment In: *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, 2000