Válasz Dr. Fogarassyné dr. Vathy Ágnes

„Üzleti folyamatok online nyomon követése és javítása folyamatbányászati és mesterséges intelligencia algoritmusok alkalmazásával"

című doktori (PhD) disszertációmhoz megküldött bírálatára

1. kérdés/megjegyzés: A javasolt módszer kétféle gyorsítótárat alkalmaz: a folyamatban lévő és a befejezett esetek gyorsítótárát. A 27. oldalon a folyamatban lévő esetek gyorsítótára 'korlátlan méretűként' kerül említésre, ugyanakkor az azt követő magyarázatban a szerző véleményem szerint már korlátosként kezeli. Kérem, részletezze, hogyan kezeli az algoritmus a két gyorsítótár méretét, és milyen hatással van ez a megoldás teljesítményére, különösen nagy esetszám esetén.

1. kérdésre/megjegyzésre adott válasz: A javasolt módszerben a folyamatban lévő és a befejezett esetek gyorsítótára külön-külön működik, és méretüket három paraméter határozza meg. A folyamatban lévő esetek gyorsítótárának méretét elsősorban az esetfrissítési határidő (t_{max}) és a frissítési ráta (χ) beállításai határozzák meg. Ha ezek az értékek nagyon magasak, akkor a gyorsítótár gyakorlatilag korlátlan méretűként viselkedik. Például, ha a megfigyelési időtartam 8 óra, de a t_{max} értéke 9 óra, akkor a gyorsítótárban gyakorlatilag minden eset megmarad a teljes megfigyelés alatt, vagyis tényleges méretkorlát nem érvényesül. Ilyen esetben a befejezett esetek gyorsítótára üres marad.

Minél nagyobb a t_{max} értéke, annál tovább maradnak az esetek folyamatban lévőként nyilvántartva, még akkor is, ha valójában már befejeződtek. Ezáltal több eset halmozódhat fel a folyamatban lévő esetek gyorsítótárában, ami a tárhasználat növekedését eredményezi. Emellett az új eseményhez tartozó esetazonosító gyorsítótárban történő megtalálása is hosszabb időt vehet igénybe, bár ez nem jelentős teljesítménybeli hátrány, mivel a keresés önmagában nem számításigényes. Ha viszont a t_{max} értéke túl alacsony, akkor előfordulhat, hogy a még folyamatban lévő esetek is befejezettként kerülnek átsorolásra. Ez többszöri, fölösleges igazításszámításhoz, illetve gyakori mozgatáshoz vezethet a két gyorsítótár között, amely már érezhetően ronthatja a teljesítményt.

A frissítési ráta (χ) meghatározza, hogy az algoritmus milyen gyakran vizsgálja meg az esetek frissességét. Alacsony χ érték esetén gyakrabban történik ellenőrzés, így az elavult esetek gyorsabban kerülnek ki a folyamatban lévő esetek gyorsítótárából, ezzel csökkentve annak méretét és növelve a befejezett esetek gyorsítótárának méretét. Ez növelheti a számítási költséget, mivel az összehasonlítások gyakrabban történnek meg. Magasabb χ érték mellett ritkább a vizsgálat, így az esetek hosszabb ideig maradhatnak a folyamatban lévő esetek gyorsítótárban, ami nagyobb memóriahasználathoz vezethet.

A befejezett esetek gyorsítótárának méretét közvetlenül egy külön paraméter, a maximális méret (l_{max}^{cc}) korlátozza. Ez közvetve hatással lehet mind a megoldás teljesítményére, mind a minőségére. Ha az l_{max}^{cc} értéke túl magas, akkor a memóriaterhelés nő,

hiszen sok befejezett esetet kell tárolni. Ha viszont az l_{max}^{cc} értéke túl kicsi, akkor előfordulhat, hogy a még folyamatban lévő esetek is kiszorulnak a gyorsítótárból, különösen alacsony t_{max} érték esetén. Ez hiányzó előtag problémához vezethet: ha egy már törölt esethez új esemény kerül megfigyelésre, akkor az algoritmus (tévesen) új esetként fogja azt kezelni, és így helytelen (előtag-)igazítást számol hozzá.

A megoldás teljesítményét nagymértékben befolyásolja az is, hogy milyen gyakorisággal érkeznek új események, és milyen hosszú ideig tart egy-egy eset lefutása. Ha sok esemény figyelhető meg rövid idő alatt, és az esetek rövidek, akkor gyakori igazításszámításra van szükség. A paraméterek megfelelő beállításával a gyorsítótárak mérete és a számítási igény is optimalizálható, így a megoldás hatékonyabbá tehető. A frissítési rátát (χ) az események érkezési gyakorisága alapján célszerű megválasztani, míg az esetfrissítési határidőt (t_{max}) és a befejezett esetek gyorsítótárának maximálisan megengedett méretét (l_{max}^{cc}) az esetek átlagos időtartama szerint érdemes meghatározni. Amennyiben rendelkezésre áll információ arról, hogy egy eset milyen gyakorisággal frissül, a t_{max} értékét ennek figyelembevételével célszerű beállítani. Az l_{max}^{cc} méretét úgy érdemes meghatározni, hogy elegendő számú lezárt eset maradjon elérhető, miközben a memóriahasználat nem lépi túl a rendelkezésre álló keretet. Az esetek átlagos időtartama mellett az új esetek indulásának gyakorisága is lényeges tényező: minél több eset zárul le minél rövidebb idő alatt, annál nagyobb gyorsítótárra lehet szükség.

2. kérdés/megjegyzés: A javasolt módszert tovább lehetne-e fejleszteni dinamikusan változó folyamatmodellek esetére is, s ha igen, akkor hogyan?

2. kérdésre/megjegyzésre adott válasz: A jelenlegi módszer elvileg alkalmazható dinamikusan változó folyamatmodellek esetén is, feltéve, hogy az éppen aktuális folyamatmodell mindig rendelkezésre áll. A működés szempontjából elsősorban az a kérdés meghatározó, hogy az új modell kizárólag az új esetekre vonatkozik-e, vagy az összes, még folyamatban lévő esetet is érinti.

Ha az új folyamatmodell csak az új esetekre vonatkozik: Ebben az esetben minden egyes folyamatmodell-verziót el kell tárolni, és nyomon kell követni, hogy az egyes esetek melyik verzióhoz tartoznak. Minden modellverzióhoz külön OVAP gyorsítótár tartozna, a többi gyorsítótár (pl. SPN gyorsítótár) kezelése nem változna.

Ha az új folyamatmodell a folyamatban lévő esetekre is vonatkozik: Ebben az esetben az optimális (előtag-)igazítás előállításához használt gyorsítótárakat aktualizálni kell a modellváltozásnak megfelelően. Az OVAP gyorsítótárat csak akkor kell üríteni, ha változik azoknak az átmeneteknek az összetétele vagy a modell szerinti helyes tüzelési sorrendjük, amelyekhez őrkifejezések vagy változóírási műveletek tartoznak. Ide tartozik az is, ha az átmenetekhez tartozó őrkifejezések vagy a változóírási műveletek módosulnak. A többi gyorsítótár (pl. SPN gyorsítótár) tartalma mindenképpen újragenerálandó, mivel a folyamatmodell módosulásával a lehetséges igazítási lépések és a keresési tér is megváltozik. Az optimális (előtag-)igazítást tehát az új SPN alapján újra kell számítani.

3. kérdés/megjegyzés: Hogyan kezeli az RR1 algoritmus a prioritással bíró új feladatokat? Felülírhatja-e a már elfogadott útvonalterveket, ha az optimalitás szempontjából indokolt?

3. kérdésre/megjegyzésre adott válasz: Az RR1 algoritmus korlátozott mértékben módosítja a már elfogadott útvonalterveket új, prioritással bíró feladatok megjelenése esetén. Ha van legalább egy olyan meglévő útvonalterv, amelyhez tartozó járműnek van elegendő szabad kapacitása az új feladat kiszolgálására, akkor az algoritmus csak ezt az egy útvonaltervet módosítja. Amennyiben egyik meglévő útvonal sem képes befogadni az új feladatot, az algoritmus nem módosítja a korábbi útvonalakat, hanem új útvonaltervet hoz létre az új feladat kiszolgálására.

Ezzel szemben más algoritmusok (pl. ABC) nem tartalmaznak ilyen korlátozást. Szükség esetén akár az összes meglévő útvonaltervet módosíthatják annak érdekében, hogy az új feladat beillesztése a lehető legoptimálisabban történjen meg.

4. kérdés/megjegyzés: Elképzelhetőnek tartja-e a két ABC-algoritmus olyan típusú integrációját, amely során az algoritmus kezdetben inkább felfedező, majd a futás vége felé feltáró üzemmódban működik? Milyen problématípusok esetében tartaná célszerűnek egy ilyen adaptív stratégia alkalmazását?

4. kérdésre/megjegyzésre adott válasz: Igen, a két CARP-ABC algoritmus integrációja jól elképzelhető egy olyan időben adaptív keretrendszerben, amely a keresési folyamat során kezdetben inkább a felfedezésre, majd a futás vége felé fokozatosan a feltáró jellegű működésre helyezi a hangsúlyt. Bár mindkét algoritmus változat tartalmaz felfedezésre és feltárásra irányuló keresést, az egyik elsősorban a megoldástér minél szélesebb körű bejárására törekszik, jellemzően véletlenszerű döntések révén, míg a másik a gyors javulásra, a jó minőségű megoldások mielőbbi elérésére koncentrál, inkább tudatos, célorientált döntések segítségével. Ez a különbség jól kihasználható egy olyan adaptív stratégiában, amely a keresés korai szakaszában a megoldástér minél alaposabb bejárását célozza, később pedig a feltárt megoldások finomítására összpontosít.

Egy ilyen adaptív stratégia alkalmazása olyan nagyobb (D)CARP példák esetén tartom célszerűnek, amelyek több lokális optimumot tartalmaznak, és a jó globális megoldás elérése érdekében elengedhetetlen a megfelelő diverzitás biztosítása. Azt viszont kérdésesnek tartom, hogy online környezetben, azaz kisebb időkeret esetén képes lenne-e megfelelő minőségű megoldást találni. Ezt azonban megfelelő paraméterbeállításokkal kezelni lehetne. Például rövidebb időkeret esetén célszerű lehet hamarabb áttérni a felfedező módból a feltáróra, vagy akár kezdettől fogva inkább a feltárásra koncentrálni.

Szeretném megköszönni Dr. Fogarassyné dr. Vathy Ágnes docens asszonynak, hogy elvállalta és elkészítette a bírálatot. A bírálatban megfogalmazott kérdések és észrevételek nagyon hasznosak voltak számomra.

Veszprém, 2025. június 12.

.....Nagy Zsuzsanna.....

Nagy Zsuzsanna
doktorandusz
Pannon Egyetem
Informatikai Tudományok Doktori Iskola