Futási eredmények

A kiválasztott példák különféle módokon történő modellezésének végeztével, az így rendelkezésünkre álló modelleket átfogó teszteknek vetettük alá. Érdeklődésünk középpontjában az állt, hogy a különböző nehézségű példák, illetve az elérő módon modellezett korlátozások miként befolyásolják a megoldók hatékonyságát.

A korábbi fejezetben már vázolt „Gardens” nevű példa elkészülte után, ezzel kezdtük a tesztelést. Ekkor még a MiniZinc IDE, illetve a megoldók korábbi verzióival dolgoztunk. A későbbiekben a szoftverfrissítést követően a teszteket újra elvégezve lehetőségünk nyílt maga a szoftver a két verziója közötti hatékonyságot is ellenőrizni.

A fenti táblázat tartalmazza a futási eredményeket milliszekundumban(ms) mérve. Megoldóként ez esetben a Gecode-ot használtuk. Az 5. verziójú modell, mivel a többihez képest jóval később készült, ezért a régi verziójú szoftverrel teszteredmények nem álltak rendelkezésünkre. Az átlagot mindegyik esetben 6 futtatás eredményéből számoltuk. Az adatokból több következetés is levonható.

Egyrészről, a régebbi szoftver verzió(az utolsó modellt érthető okokból most nem számítva) minden esetben valamivel lassabban oldotta meg példáinkat, így kimondhatjuk, hogy készítői láthatóan javítottak termékükön és optimalizálták a hatékonyságát.

A második szembetűnő jelenség, miszerint az első általunk készített alapmodell, még meglehetősen nyersnek bizonyult megoldhatóság szempontjából is. A futási eredmények jó okot szolgáltatnak a forall függvény where záradékában döntési alapként felhasznált, értékkel még nem feltétlenül rendelkező változók használatának kerülésére.

Harmadrészt pedig jól látható, hogy az olyan speciális, nem „Einstein-típusú” logikai feladvány esetében, mint amilyen a „Gardens” is, nem szerencsés a tömböket használó adatreprezentáció választása. Részben a lassú megoldáskeresés, részben pedig a bonyolult megvalósítás miatt sem.

A bináris mátrixok illetve halmazok használata között jelentős futásidőbeli eltérés nem volt. Főként annak tükrében, hogy az újabb verziónál teljesen azonos volt az átlagos futásidő mindkét modelltípus esetében. Ám a régebbi verzió esetében is csak nagyon kis mértékben bizonyultak hatékonyabbnak a bináris mátrixos adatstruktúrák.

A következőkben a Zebra vagy Einstein-típusú logikai fejtörők hatékonyságának összehasonlítása következik. Jelen esetben 2 példa állt rendelkezésünkre: A „Movies Night” mint 4x5-ös nagyságú könnyebb és a „Fundraising Dinner”, mint 5x6-os nehezebb feladat. Megoldóként továbbra is a Gecode-ot használtuk, és az átlagszámítási módszer is azonos volt az előzőekben ismertetetthez.

Amint az látható a „Fundraising Dinner” feladat esetében hosszabb futásidőket tapasztaltunk, mint a „Movies Night” esetében, amely a példa összetettebb mivoltát tekintve nem meglepő. Az viszont már jóval szembetűnőbb, hogy az eltérés nagyon kicsi, ami jelzi, hogy megoldónk egy nagyobb és nehezebb, több hozzárendeléssel dolgozó feladat esetében is képes a futásidőket abszolút alacsony szinten tartani. Ez komoly érv amellett mikor eszközt választunk a hasonló típusú feladatok megoldásához. Más modellezési technikákhoz képest a korlátprogramozás esetében nem a feladat összetettségének mértékében nőnek a futásidők, hanem annál jelentősen lassabban, és ez a futásidő növekedés is csak a hozzárendelések számának jelentős növelésekor mutatkozik meg igazán.

Emellett azt tapasztaltuk, hogy a jóval gyakoribb Einstein alkotta Zebra Puzzle-ök modellezésekor sokkal célszerűbb egy tömböket használó adatstruktúrát megvalósítani. Mind a modellezés során a könnyebb kezelhetőség, mind pedig a jóval rövidebb futásidők szempontjából sokkal optimálisabb, ha ezt az adatreprezentációt részesítjük előnyben.

Összegzésként, a végső konzekvenciákat levonva: a korlátprogramozás a kifejezetten rövid megoldási és futásidőket figyelembe véve, egy borzasztóan hatékony eszköz az általunk vizsgált logikai fejtörők, mint speciális hozzárendelési feladatok modellezéséhez. A gyakori és sokak által ismert Einstein-féle feladatok esetében válasszuk a felsorol típussal indexelt tömbök használatát. Viszont az olyan nem hétköznapi példáknál – amilyen esetünkben a „Gardens” is volt – ahol nem beszélhetünk a klasszikus egy-az-egyhez hozzárendelésről már más a helyzet. Ez esetben javasolt valamilyen halmazokkal dolgozó adatszerkezet, esetleg bináris mátrixok deklarációja az maximális hatékonyság elérése érdekében.