

Programtervezési megközelítések: oszd meg és uralkodj, mohó algoritmusok, dinamikus programozás

„Divide et impera” - Oszd meg és uralkodj

- Felosztjuk a problémát két vagy több (közel egyenlő súlyú/bonyolultságú) részproblémára
- Uralkodunk: amikor a részproblémákat megoldjuk
- Összevonjuk a részproblémák megoldásának eredményeit az eredeti probléma eredményévé

Oszd meg és uralkodj:

Pl. Összefésülő rendezés

1. A listát megfésüljük
2. Az első részlistát összefésülve rendezzük
3. A második részlistát összefésülve rendezzük
4. A 2 és 3 pontok eredményeit összefésüljük.
5. Az összefésülés eredménye a rendezett lista

Mohó algoritmusok: az a problémamegoldó algoritmus, amely helyi optimumok megvalósításával próbálja megtalálni a globális optimumot. Ez például az utazó ügynök problémájaként* ismert feladatban azt jelenti, hogy minden állomásról a legközelebbi, addig még nem látogatott városba fog utazni az ügynök.

(*Adva van n város, illetve az útiköltség bármely két város között, keressük a legolcsóbb utat egy adott városból indulva, amely minden várost pontosan egyszer érint, majd a kiindulási városba ér vissza.)

Általánosan öt pillérre támaszkodik:

- egy halmazból veszi a jelölteket, amelyekkel felállítja a megoldáshalmazt
- egy kiválasztó függvény, amely a legjobb jelöltet választja ki a megoldás reményében
- egy lehetőségvizsgáló függvény, amely megnézi, hogy egy jelölt alkalmas-e a megoldásra
- egy célfüggvény, amely egy értéket megoldásnak, vagy részleges megoldásnak jelöl
- egy megoldásfüggvény, amely jelzi, ha megtaláltuk a teljes megoldást.

- Globális \leftrightarrow lokális optimalizálás: A mohó stratégia csak helyi optimalizálást végez

- Bizonyos problémákra bebizonyíthatóan (globálisan is) optimális eredményt ad, más problémákra csak szuboptimális a mohó eredmény

- **Ciklusos megoldás:** Lineáris szerkezetekre, hagyományos megoldás
- Divide et impera: nem lineáris szerkezetekre, a felosztás egyenlő súly alapján történik
- Rekurzio: mindezek technikai megoldása
- helyettesítheti a ciklusos megoldást is
- megvalósíthatja az „oszd meg és uralkodj” elvet is

Dinamikus programozás

- Oszd meg és uralkodj: független részproblémák

- Dinamikus programozás esetén nem függetlenek. (Az Oszd meg és uralkodj feleslegesen többször dolgozna)

Jellemzése:

A dinamikus programozás éppúgy, mint az oszd meg és uralkodj módszer, a feladatot

részfeladatokra való osztással oldja meg.

A feladatot felosztja részfeladatokra, és a részfeladat eredményét egy táblázatban tárolja el, és ezáltal elkerüli az ismételt számítást, ha a részfeladat megint felmerül.

A dinamikus programozást optimalizálási feladatok megoldására használjuk.

A dinamikus programozás lépései:

1. részproblémákra bontás, optimális megoldás szerkezete
2. a részproblémák közötti összefüggések meghatározása (rekurzív képlettel)
3. a részproblémák optimális értékeinek kiszámítása alulról felfelé haladva (táblázat kitöltése)
4. az optimális megoldás megalkotása