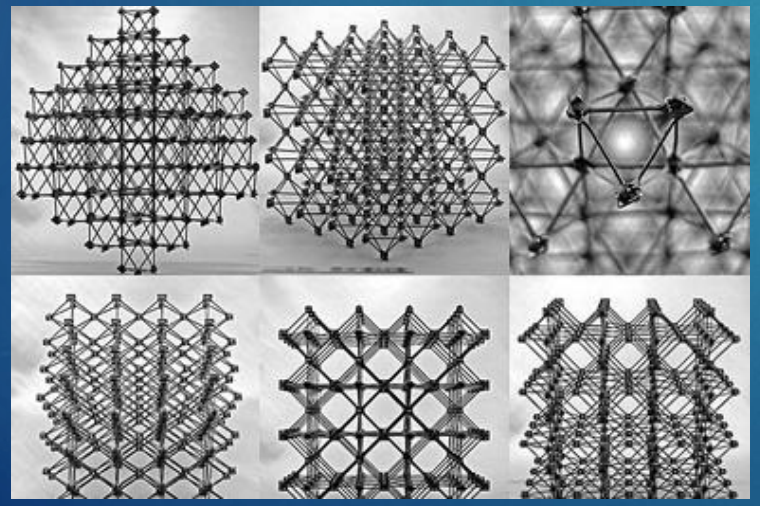
The Structural Power of Reconfigurable Circuits in the Amoebot Model

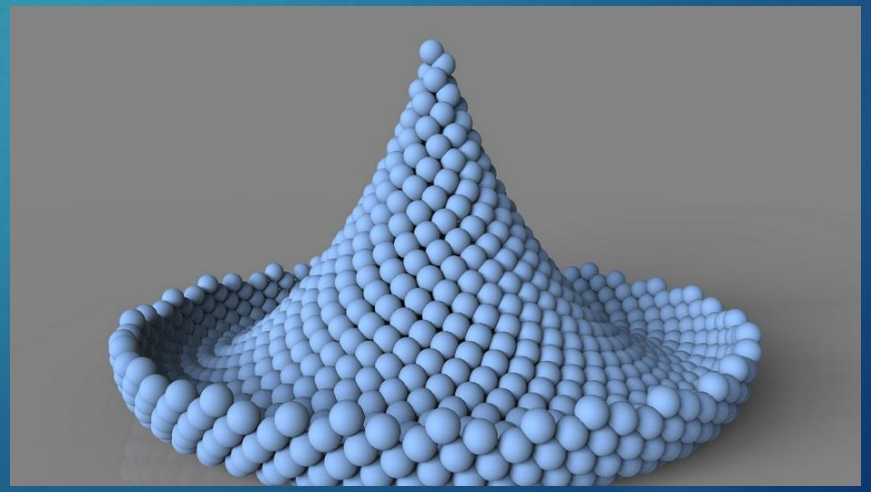
Átprogramozható Áramkörök Szerkezeti Ereje az Amőba Modellben

KÉSZÍTETTE: BÉRES GÁBOR KRISTÓF, TÓTH BOTOND, WERNER BENDEGÚZ

Bevezetés az amőbamodellbe

- Az amőbamodell egy programozható anyagot ír le, amely apró, robotikus egységekből (*amőbotokból*) áll.
- Ezek az egységek egy végtelen háromszög rácsra helyezkednek el, és képesek mozogni tágulás és összehúzódás révén.
- A cél a kollektív viselkedés vizsgálata és optimalizálása.





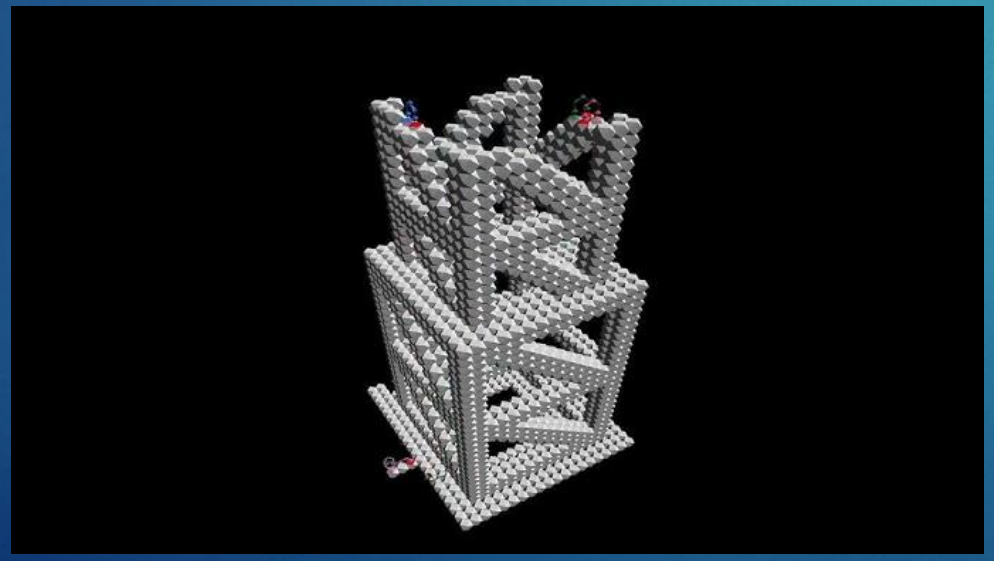
Fő kutatási kérdések a cikkben

- Stripe problem vagy Csík probléma
- Globális maximum probléma
- Csontváz-probléma
- Spanning tree létrehozása
- Szimmetria detektálás

Problem	Required pins	Runtime
Stripe	2	$O(\log n)$
Global maxima	2	$O(\log^2 n)$ w.h.p.
Canonical skeleton	4	$O(\log^2 n)$ w.h.p.
Spanning tree	4	$O(\log^2 n)$ w.h.p.
Symmetry detection	4	$O(\log^5 n)$ w.h.p.

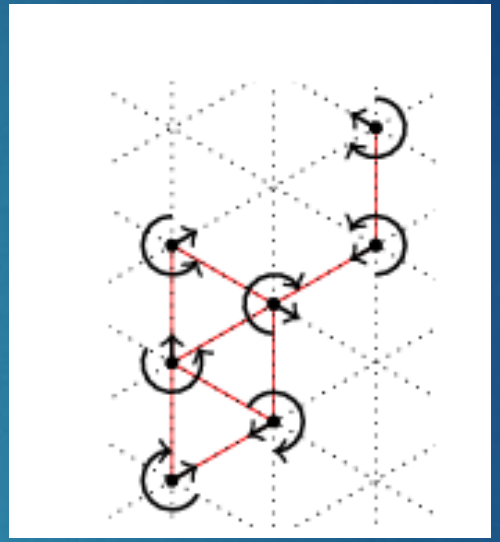
Alkalmazások és jelentőség

- Az amőbamodell és az átprogramozható áramkörök **gyors alakváltásra**, **energiahatékony adatátvitelre**, és **szerkezeti monitorozásra** használhatók.
- Az új megközelítések **polilogaritmikus időbonyolultságot** érnek el, jelentős teljesítményjavulást kínálva.



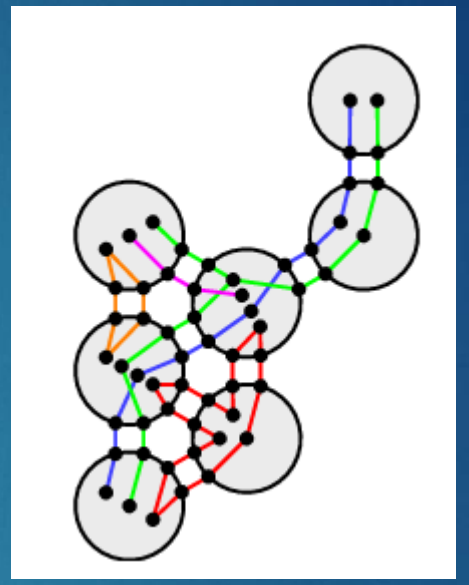
Modell felépülése

- Végtelen háromszög gráf rács
- Helyi mozgások (összehúzódás, tágulás)
- Minden csúcson legfeljebb 1 ameobot
- Közös irány orientáció
- Minden ameobot összehúzodott
- Összefüggőek

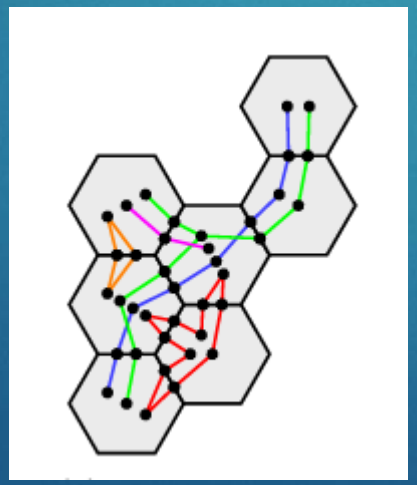


Áramköri kiterjesztés

- Külső kapcsolatok: Minden ameobot között k db él
- Csapok egységesek ($k \geq 1$)
- Csapkészlet felosztása diszjunkt halmazokra
- Áramkörök alakulnak ki
- Primitív jelküldés partíciókon



- Hatszögletű mozaik



PASC algoritmus

- Láncok -> ameobotok rendezett sorozata (u_i és u_{i+1} szomszédok)
- Vezetőválasztás

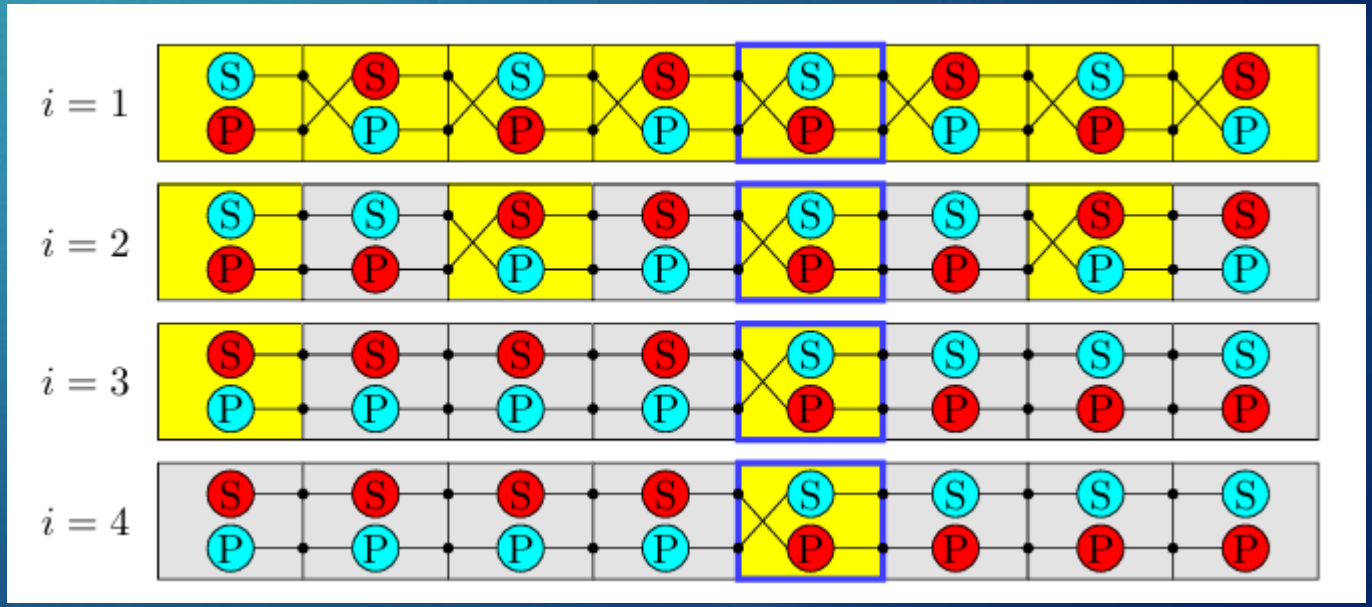
- Lánc meghatározása
- Algoritmus célja: azonosítók kiszámítása

Azonosítók lánc mentén - előkészületek

- Referencia ameobot
- Minden ameobot aktív vagy passzív
- Két áramkör létrehozása
- Elsődleges és másodlagos partíció halmaz
- Összekötés:
  - Aktív: elsődleges -> előd je másodlagos
    másodlagos -> előd je elsődleges
  - Passzív: elsődleges -> előd je elsődleges
    másodlagos -> előd je másodlagos

Azonosítók lánc mentén - folyamat

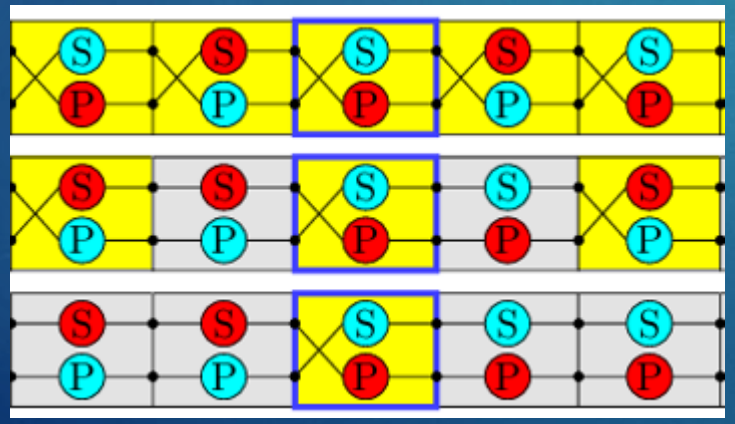
- Referencia ameobot aktíválja az elsődleges áramkört (jelzés)
- Aki a másodlagoson kapott jelzést, a 2.körben a másodlagoson jelez, majd passszív lesz
- Addig tart, amíg a 2.kör csendes



Azonosítók lánc mentén - azonosítók

- Minden körben minden ameobot az egyik áramkörén kap jelzést
- Elsődleges áramkör: 0
- Másodlagos áramkör: 1
- Minden jelzés egy bit, visszafele kell jegyezni: $(x_{k-1},\ldots,x_0)$, x_i körben 1 vagy 0
- k az iterációk száma ($0 \leq i < k$)

- Példa:



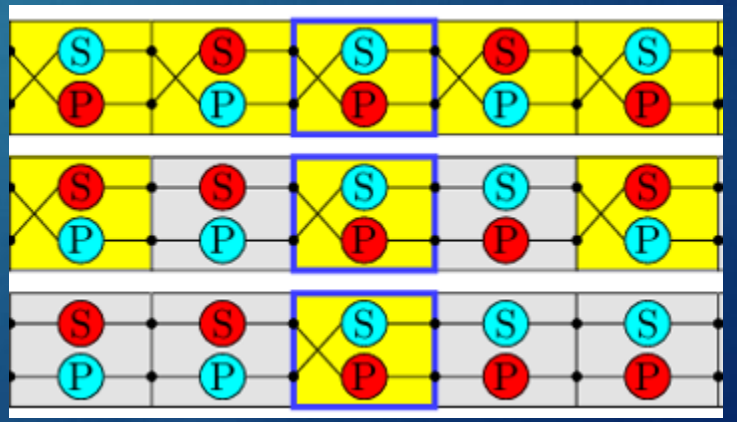
0	1	0	1	0
10	11	00	01	10
110	111	000	001	010

Azonosítók lánc mentén - bitek

- Kettes komplemens ábrázolás
- Pozitív és 0
  - 000: 0
  - 001: 1
  - 010: 2

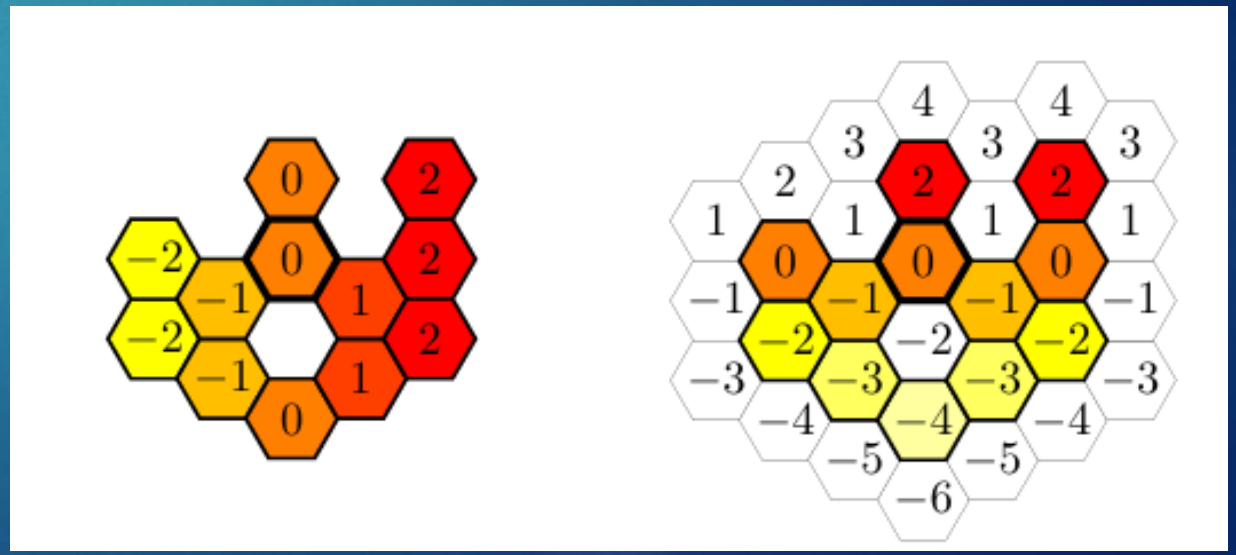
- Negatív
  - bitek invertálása
  - 1 hozzáadása
  - 111 -> 000 -> 001: -1
  - 110 -> 001 -> 010: -2

0	1	0	1	0
10	11	00	01	10
110	111	000	001	010



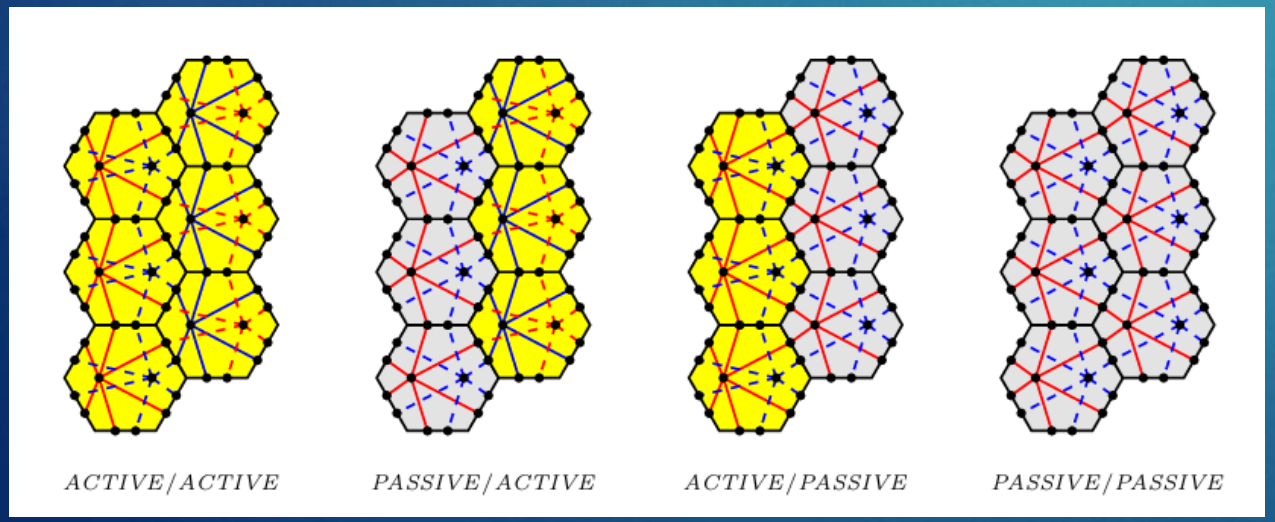
Térbeli azonosítók

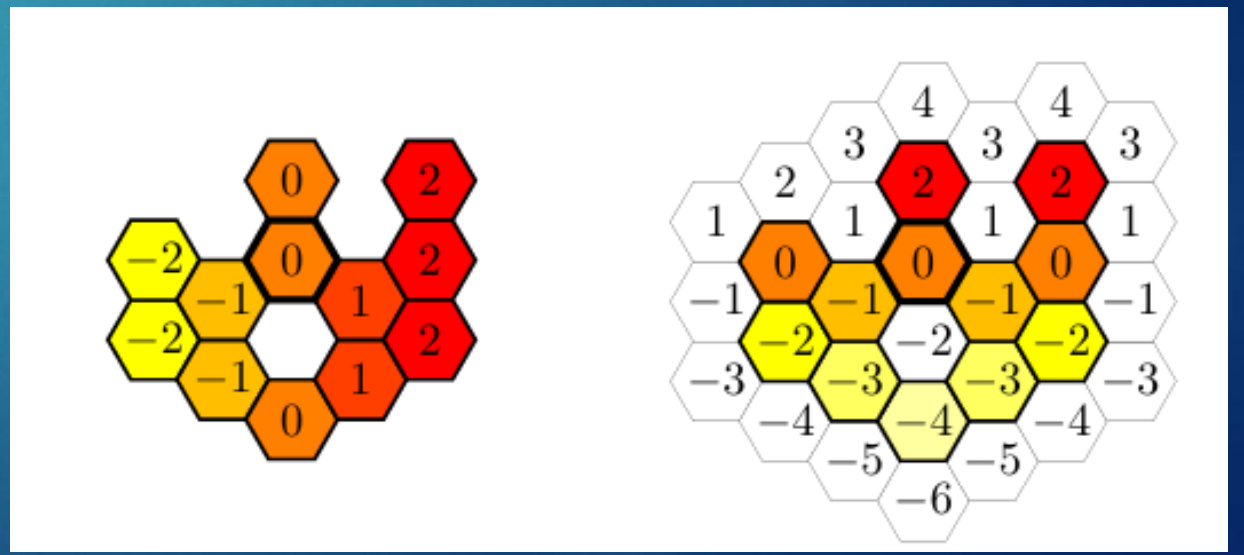
- Adott egy d irány és egy egy 90°-al elforgatott d’ irány
- pl.: d=E -> d’=N
- Diszjunkt csíkokra partícionálás
- Minden csík ismeri az elődjét és utódját
- Csíkok összekötése



Térbeli azonosítók

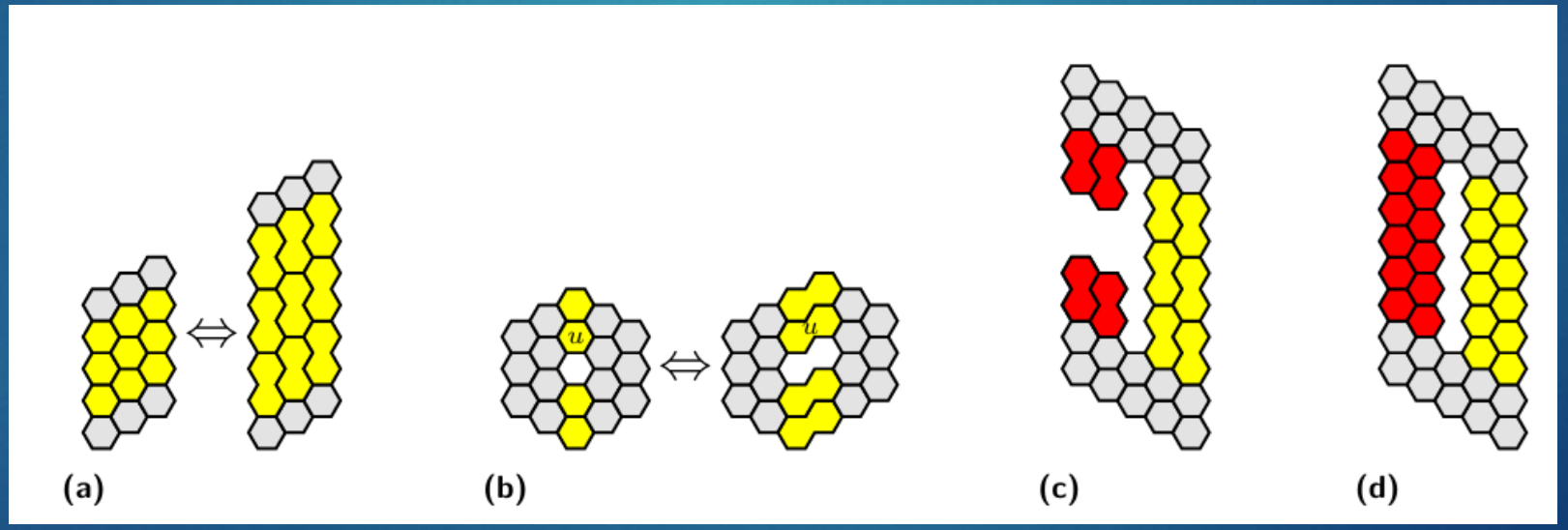
- Adott egy d irány és egy egy 90°-al elforgatott d' irány
- pl.: d=E -> d'=N
- Diszjunkt csíkokra partícionálás
- Minden csík ismeri az elődjét és utódját
- Csíkok összekötése
- PASC algoritmus a csíkok halmazán





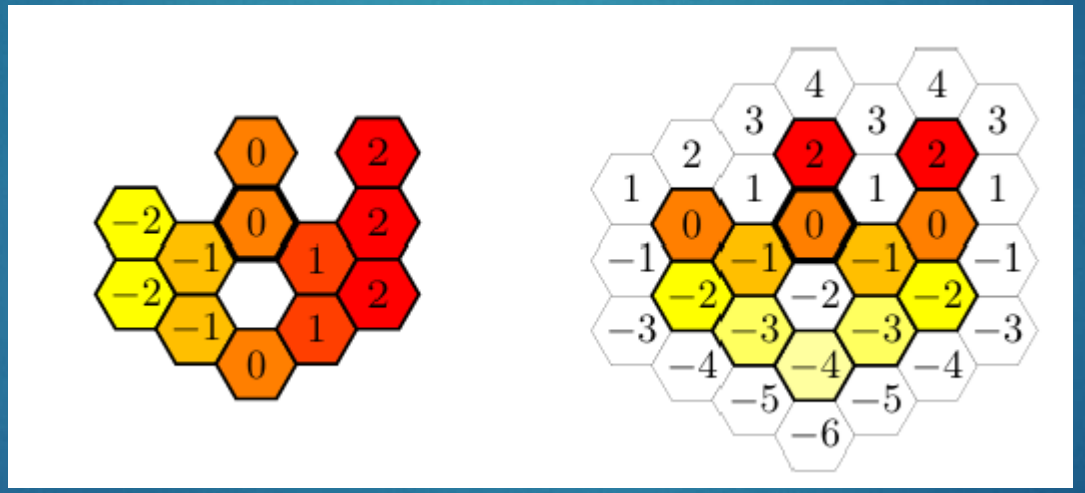
Stripe problem

- Probléma: Egy adott u ameobot és X tengely
  - Összes, a tengelyes áthaladó ameobot meghatározása
- Fontos a konfliktusok elkerüléséhez, gyors alakváltás
- PASC algoritmus megoldja



Globális maximum probléma

- Probléma: Egy adott ameobot halmaz globális maximumának meghatározása
- PASC algoritmus által kiosztott azonosítók meghatározzák a szélsőértékeket egy tengely mentén



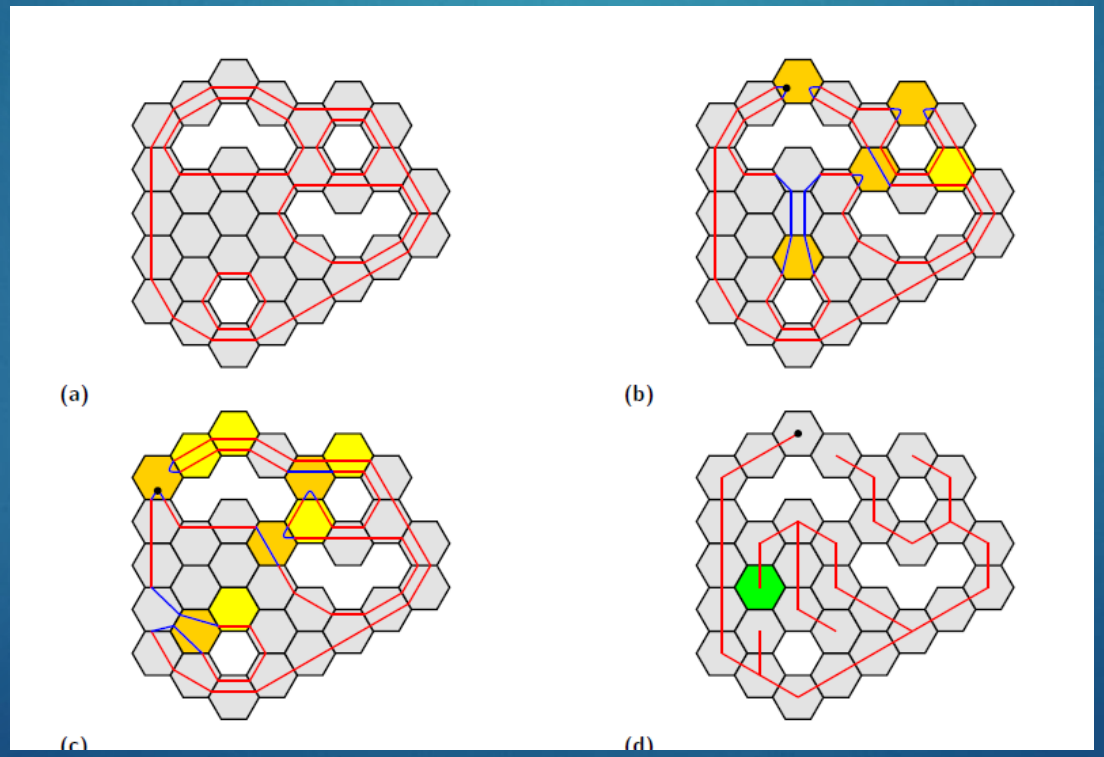
Csontváz-probléma megfogalmazása

- Egy amőbotokból álló struktúrában meg kell találni egy olyan ciklust, amely az összes **határon lévő amőbotot** tartalmazza
- Ezt a csontvázat egy algoritmus segítségével kell kialakítani, amely biztosítja, hogy mindig azonos eredményt kapjunk (ún. **kanonikus csontváz**).
- A cél: egy egyedi, meghatározott módon konstruált csontváz (skeleton) létrehozása.

Kanonikus csontváz megalkotása

- Az algoritmus egy **irányt** és egy **előjelválasztást** használ a csontváz egyedi meghatározására

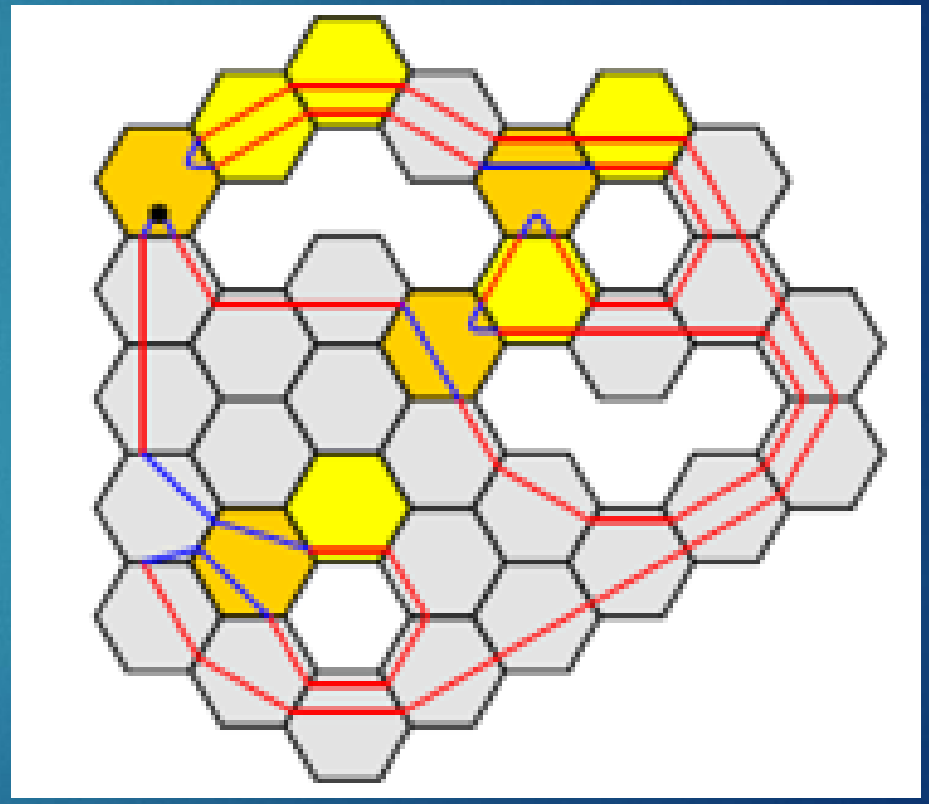
(*N*, +)-



(*NNW*, +)

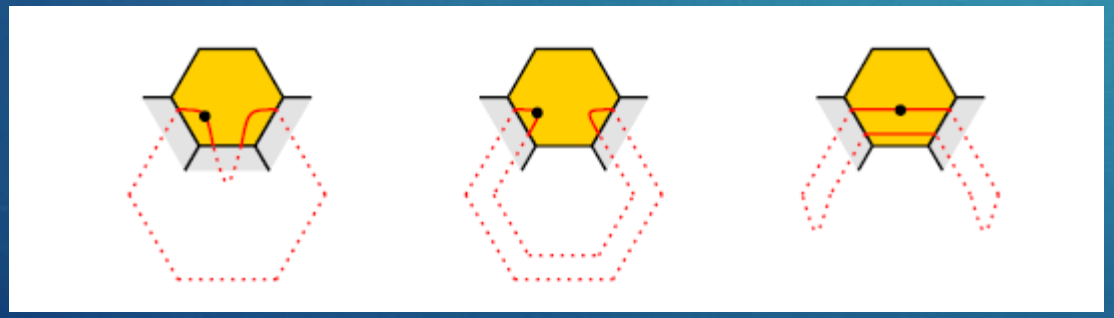
Csontváz létrehozásának lépései

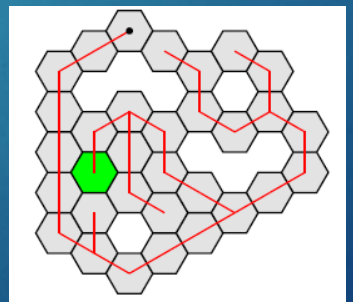
- **Globális maximum keresés**: A szerkezetben a legmagasabban lévő határ-amőbot kiválasztása.
- **Csontváz kiindulási pontjának meghatározása**.
- **Kapcsolódó amőbotok és határok összekötése** az egyedi útvonal létrehozásához.



Csontváz és kapcsolódó határok

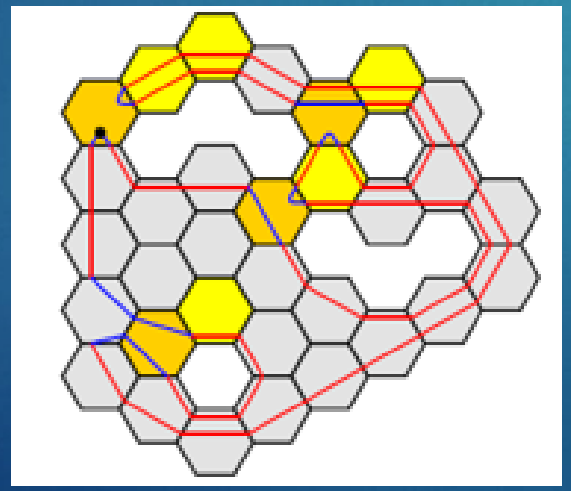
- A határ-amőbotokat tartalmazó **kisebb körökből** indulunk ki.
- Ezeket egyesítjük egy **egyetlen csontvázzá** egy jól meghatározott útvonal segítségével.
- A csontváz mentén található amőbotok egy **fa** struktúrát alkotnak, amely további algoritmusokhoz használható.
- A csontváz nem törik meg és **minden határ-amőbotot tartalmaz**.

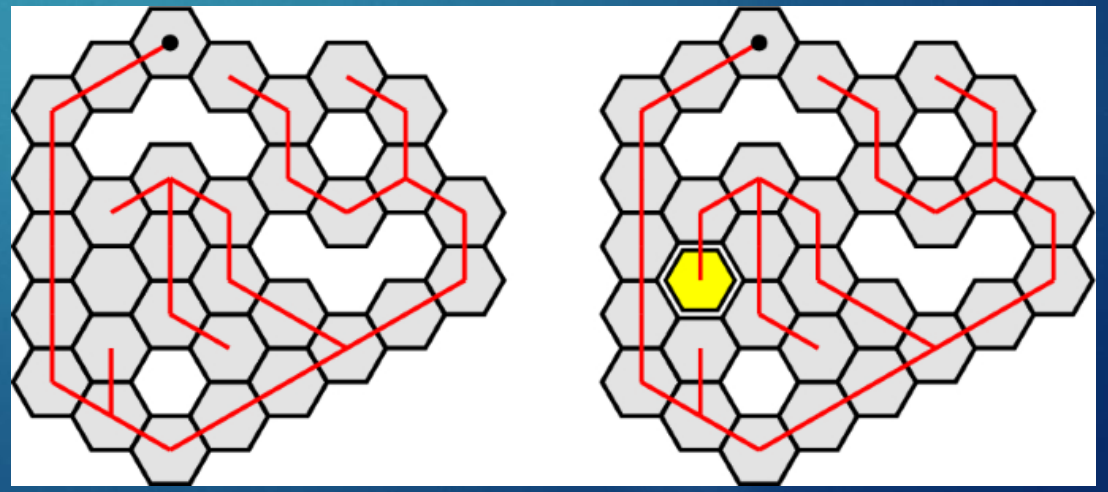




Spanning tree létrehozása

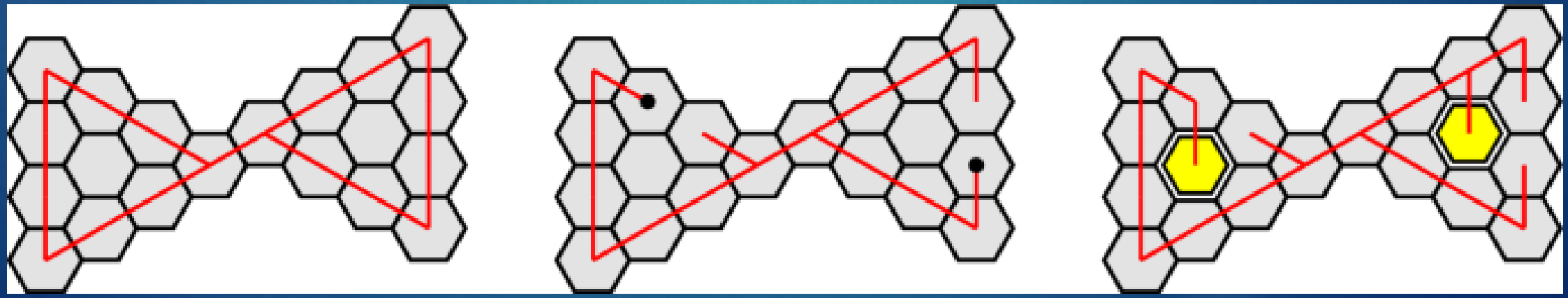
- A csontvázból először létrehozunk egy fát
  - Csontváz út π = (v1, . . . , vm) → 1 csúcs többször előfordulhat
  - PASC algoritmus-al minden csúcs első előfordulását meghatározzuk
  - Minden amőbot első előfordulásakor jelez az elődjének ezzel a fát létrehozva
- Hogy spanning tree-t kapjunk, hozzáadjuk a belső node-okat a fához
  - minden belső node-hoz hozzáadunk egy élt az északi szomszédjától
- O(log n) futási idő





Spanning tree csontváz nélkül

- Ha az amőbotokból álló struktúra nem tartalmaz lyukat, a spanning tree csontváz út nélkül létrehozható
  - Minden szomszédos határ-amőbot pár közé felveszünk egy élt
  - Minden ezzel létrejött ciklusban választunk egy vezetőt, aminek eltávolítjuk az egyik élét ezzel megszüntetve a ciklust
- Hogy spanning tree-t kapjunk, hozzáadjuk a belső node-okat a fához
- O(log n) futási idő

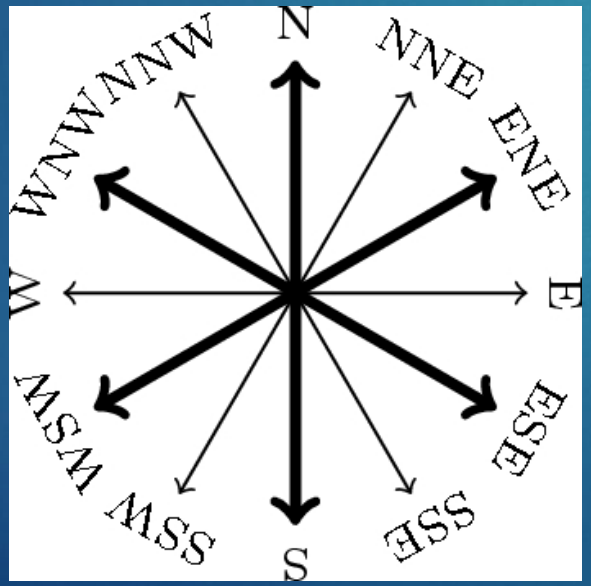


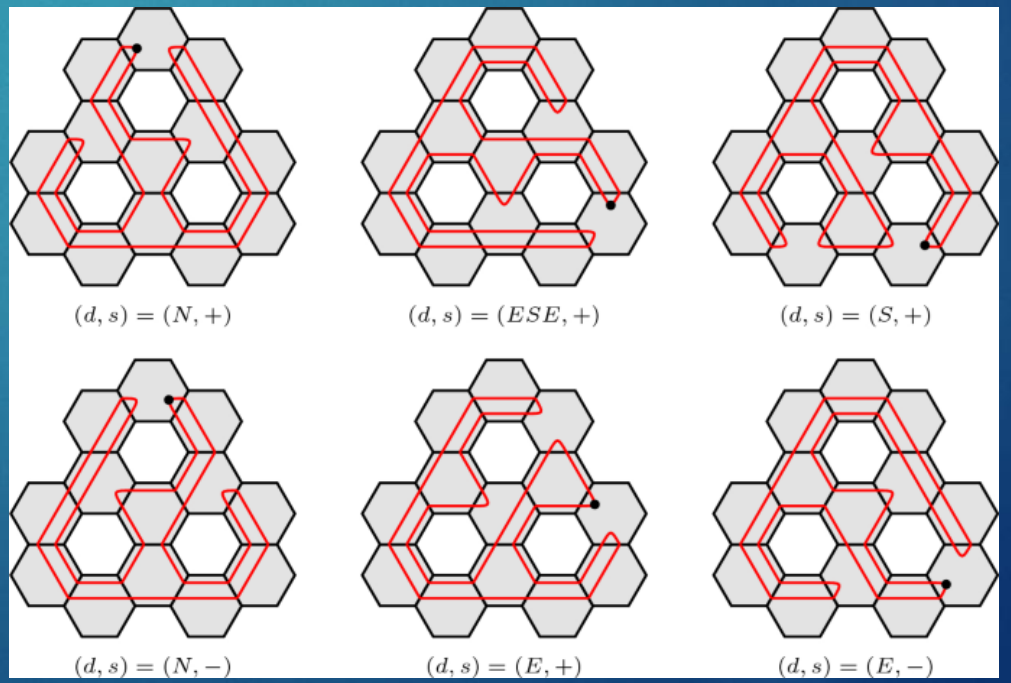
Szimmetria detektálás

- Egy amőbotokból álló struktúra szimmetriái a különböző irányú és előjelű kanonikus csontvázainak összehasonlításával detektálható
  - minden (d, s) kanonikus csontvázat egy egyedi bit sorozattá alakítunk
  - bitek összehasonlítása csontvázak helyett → polilogaritmikus futási idő: O(log^5 n)
- 2 fajta szimmetria értelmezhető
  - lehet valamelyik tengelye mentén szimmetrikus
  - lehet forgásszimmetrikus (2-fold, 3-fold, 6-fold)

Szimmetria detektálás

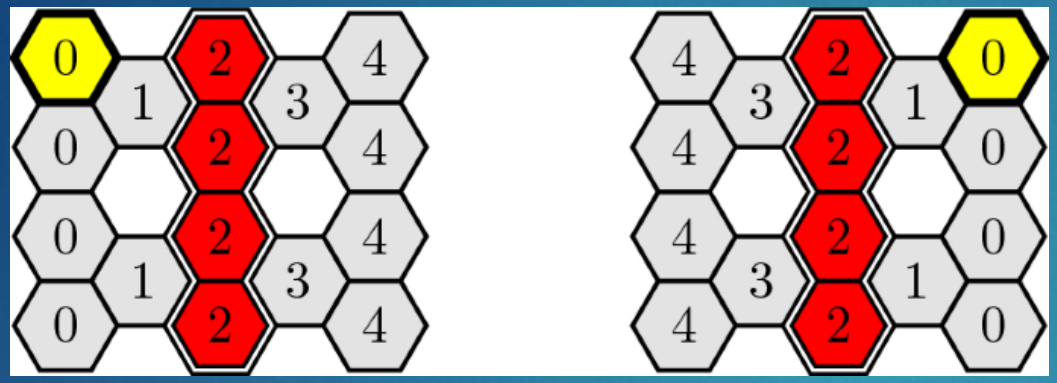
- Északi irányú tengelyre szimmetrikus, ha az (N,+) és (N, -) csontvázak szimmetrikusak
- Keleti irányú tengelyre szimmetrikus, ha az (E, +) és (E, -) csontvázak szimmetrikusak
- 2-fold szimmetrikus (180°), ha az (N, +) és (S, +) csontvázak szimmetrikusak
- 3-fold szimmetrikus (120°), ha az (N, +) és (ESE, +) csontvázak szimmetrikusak
- 6-fold szimmetrikus (60°), ha 2 és 3-fold szimmetrikus



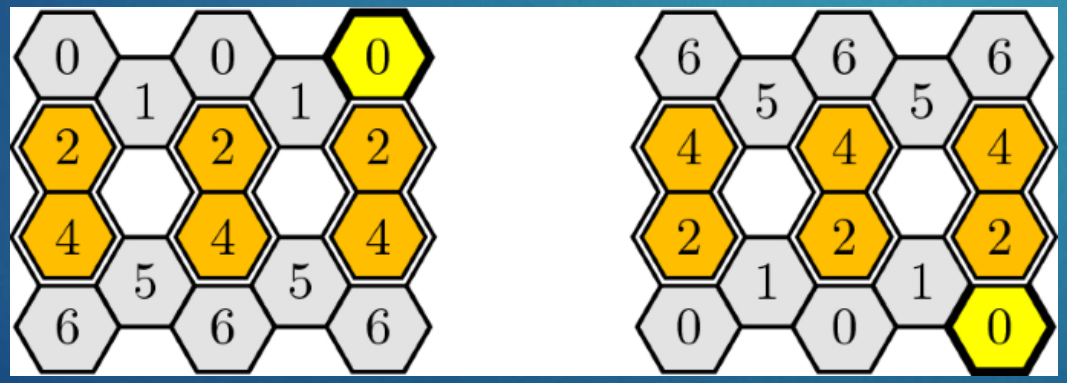


Szimmetria detektálás

- Északi irányú tengelyre szimmetria példa

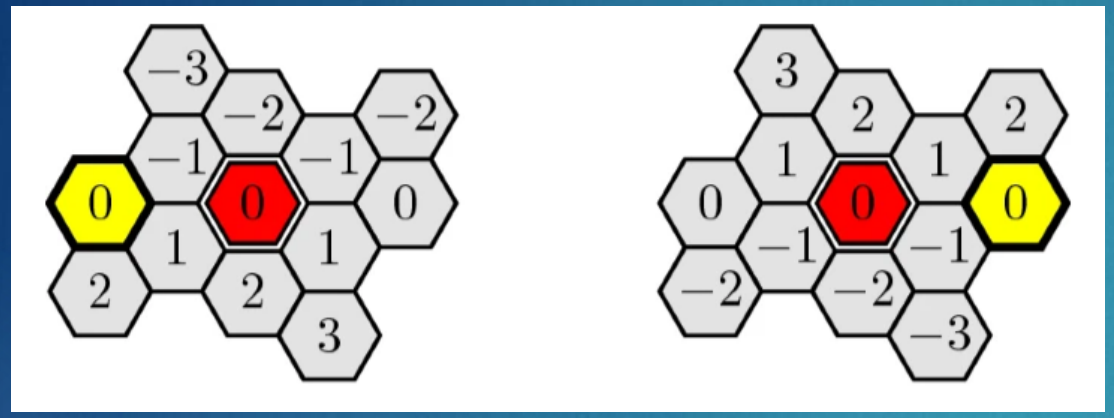


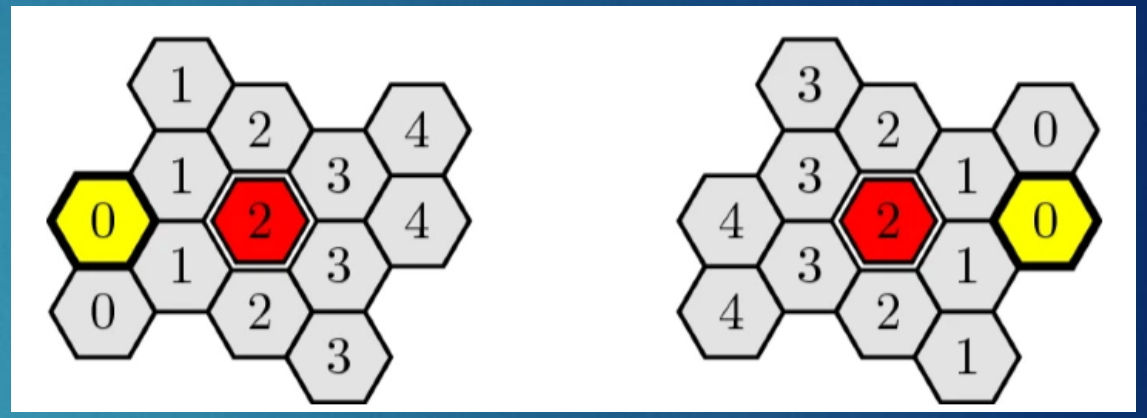
- Keleti irányú tengelyre szimmetria példa



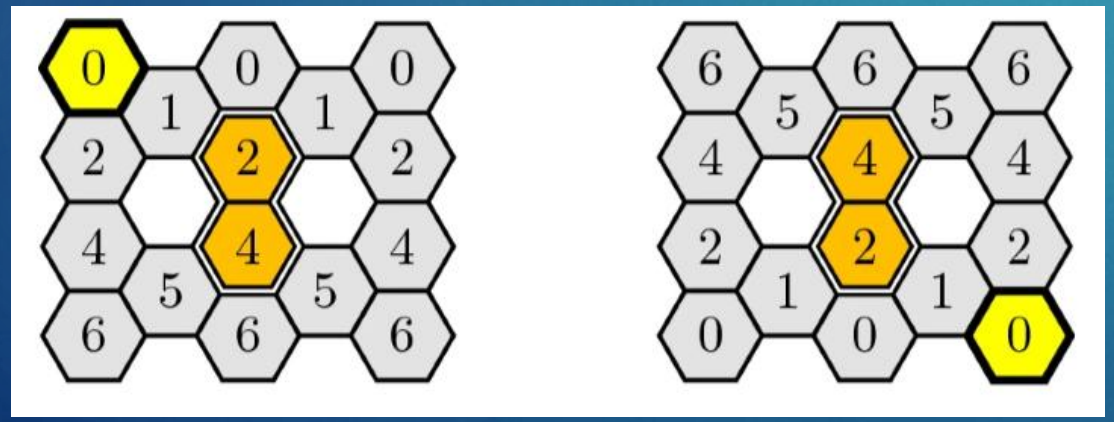
Szimmetria detektálás

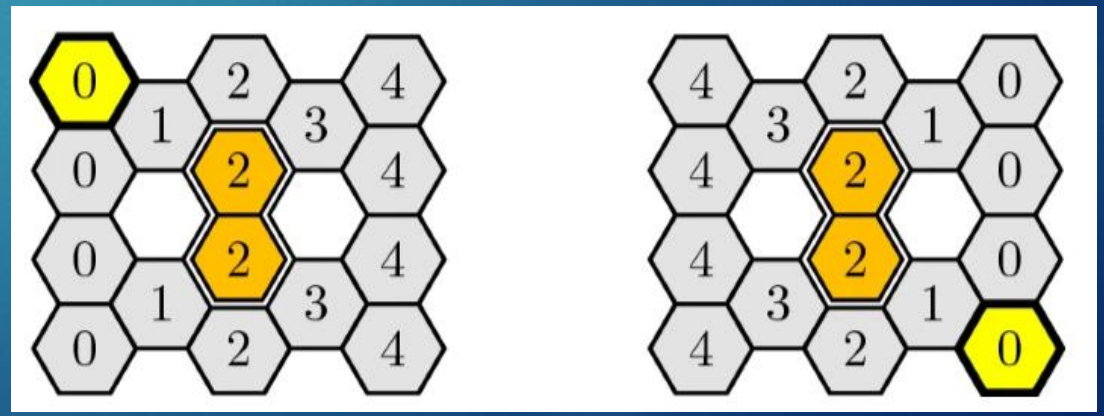
- 2-fold szimmetria egy belső node-ra





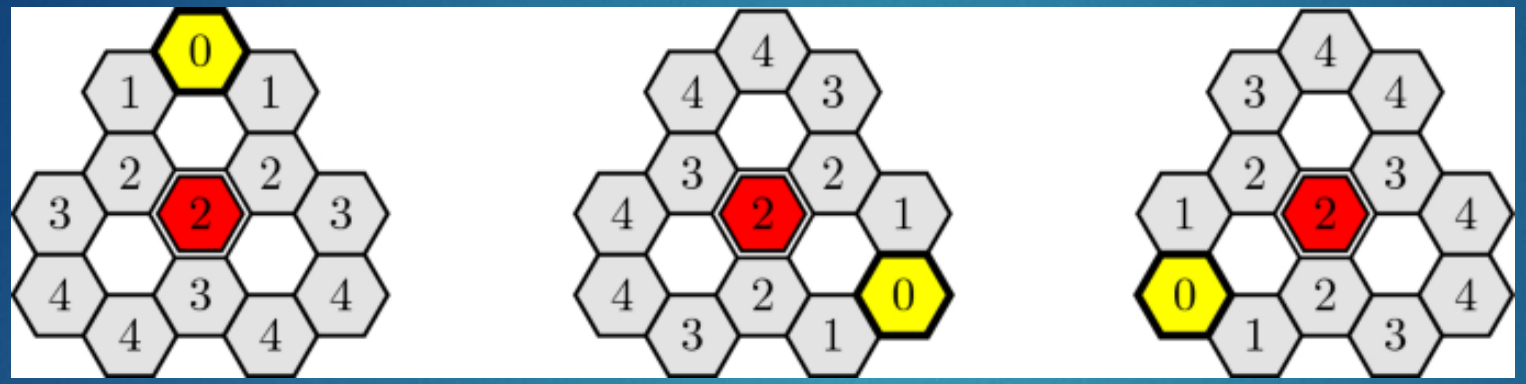
- 2-fold szimmetria 2 belső node közti élre





Szimmetria detektálás

- 3-fold szimmetria egy belső node-ra



- 3-fold szimmetria 3 belső node közötti pontra

Köszönjük a figyelmet!