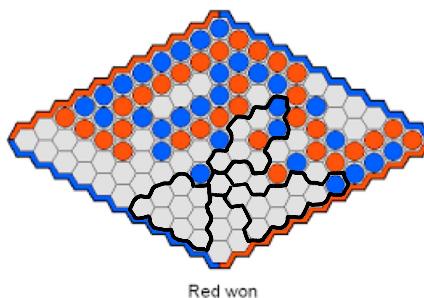


2.19. (RHEX). El joc de RHEX és una variació del joc de l'HEX inspirada en el Reversi. En aquest joc, hi han  $2n$  fitxes, les fitxes tenen dos cares, una de color blanc i l'altra de color negre. El tauler es una xarxa  $n \times n$  de cel·les hexagonals, com la de l'HEX. Hi han dos jugadors, un amb color negre i l'altre amb color blanc, que fan torns. A cada torn el jugador que li toca col·loca una fitxa mostrant el seu color a una posició encara buida. Un cop col·locada una fitxa al tauler, no es pot moure.

Quan es col·loca una fitxa al tauler es considera la *zona dominada*: totes les cel·les ocupades i connectades amb la nova cella amb un camí continuo de fitxes, normalment hi hauran fitxes de tots dos colors en aquests camins. Dues cel·les es consideren connectades si comparteixen una vora i les dues tenen una fitxa, independentment del color de les fitxes. Si a la zona dominada hi ha menys fitxes del color del jugador que té el torn que del color de l'altre jugador, es giren totes les fitxes a la zona dominada.

La partida acaba quan un dels dos jugadors guanya (aconsegueix tenir més de  $n$  fitxes del seu color) o quan s'hagin col·locat les  $2n$  fitxes.

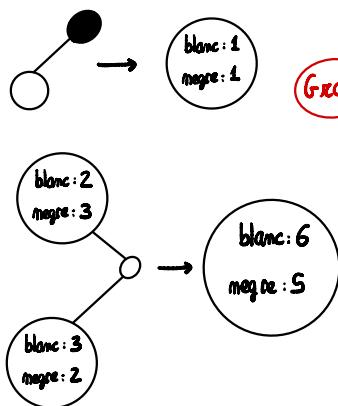
- Descriuviu un algorisme eficient per al seguiment d'una partida. L'algorisme, després de cada jugada, ha de permetre mantenir el nombre total de fitxes de cada color al tauler i a més ha d'indicar si el jugador que acaba de jugar ha guanyat el joc de RHEX.
- Expliqueu com modificar el vostre algorisme per tal de que a més de mantenir la puntuació podeu actualitzar el tauler.



La millor estructura per mantenir les components connexes d'un graf dinàmic.

## UNION-FINDS

- (a)  $\text{fitxes\_blanques} = \text{fitxes\_negres} = 2n$ ;  $\text{blanques\_posades} = \text{negres\_posades} = 0$ ;  
Mentre  $\text{m} \neq 0$  ( $\text{fitxes\_blanques} = 0 \wedge \text{fitxes\_negres} = 0 \wedge \text{blanques\_posades} > n \wedge \text{negres\_posades} > n$ ).  
Torn  $x$ : ( $x$  pot ser blanc o negre)  
place  $x$  a  $(a, b)$ . --  $\text{fitxes\_x}$ ;



**Graf** Algoritme de Prim considerant només les caselles amb fitxes i començant a  $(a, b)$ . comptem quantes caselles blanques i negres hi ha. (blanc/negre + nombre de fitxes en la component connexa on cau)

Si  $\text{num\_x\_arbre} \geq \text{num\_!x\_arbre}$ : (no flip)  
 $\quad \quad \quad ++ x\_posades$ ;

Si  $\text{m} \neq 0$ : (flip)

$$x\_posades = x\_posades + \text{num\_!x\_arbre} - \text{num\_x\_arbre};$$

$$\text{!x\_posades} = \text{!x\_posades} + \text{num\_x\_arbre} - \text{num\_!x\_arbre};$$

Utilitzant union-finds ens podem guardar les zones dominades com a un únic mode, amb el número de fitxes blanques i negres que hi ha en les zones. Quan connectem dues zones, sumem aquests valors; i si form flip, els intercanviem.

b) Per actualitzar el taulell no podem agrupar les zones en únics nodes; ens cal saber el color de la fitxa present en cada casella. Quan es dona una condició de flip, les blanques es tornen negres i viceversa (només les de la zona dominada).

Aquí hauríem de conservar els arbres senyals.