Rassemblement d'agents mobiles

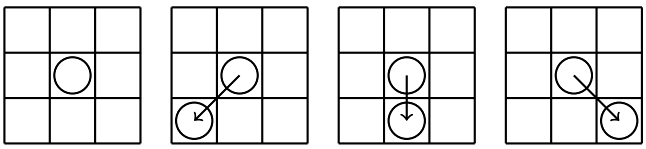
Éloi Perdereau

28 avril 2014

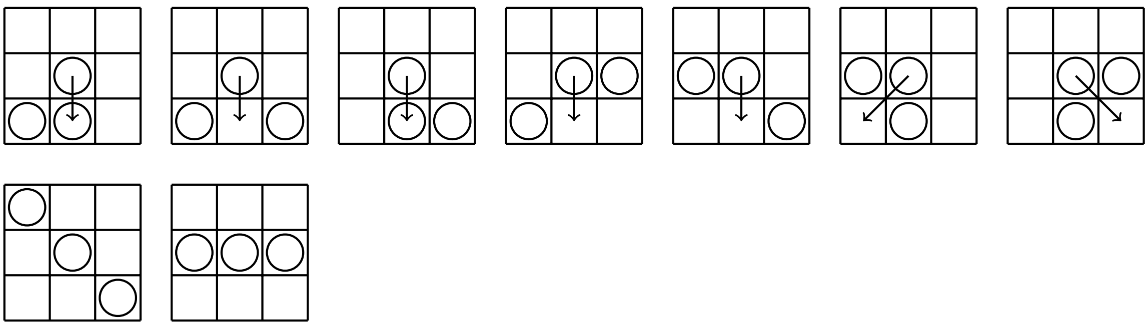
1 Cas

On omet les cas symétriques par rapport au robot du milieu (rotations de 90°, 180° et 270°.)

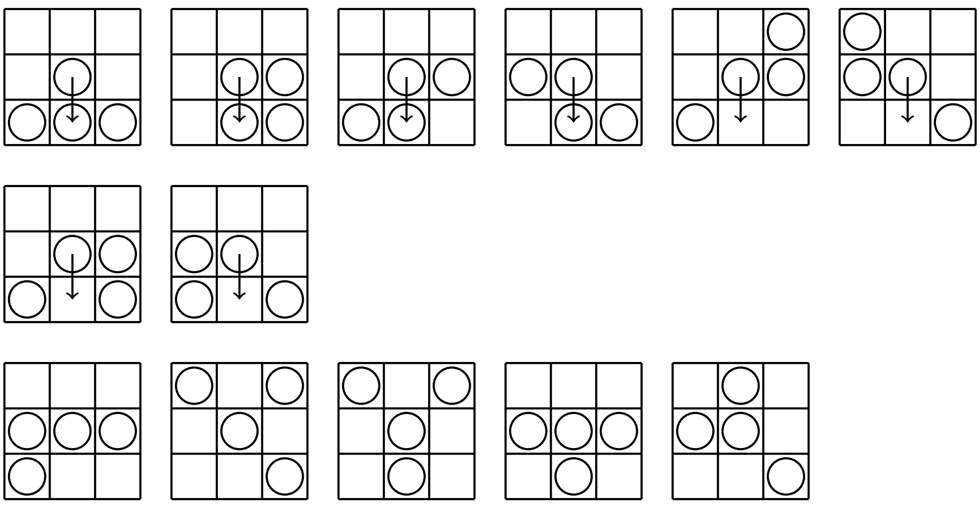
1.1 0 ou 1 voisins



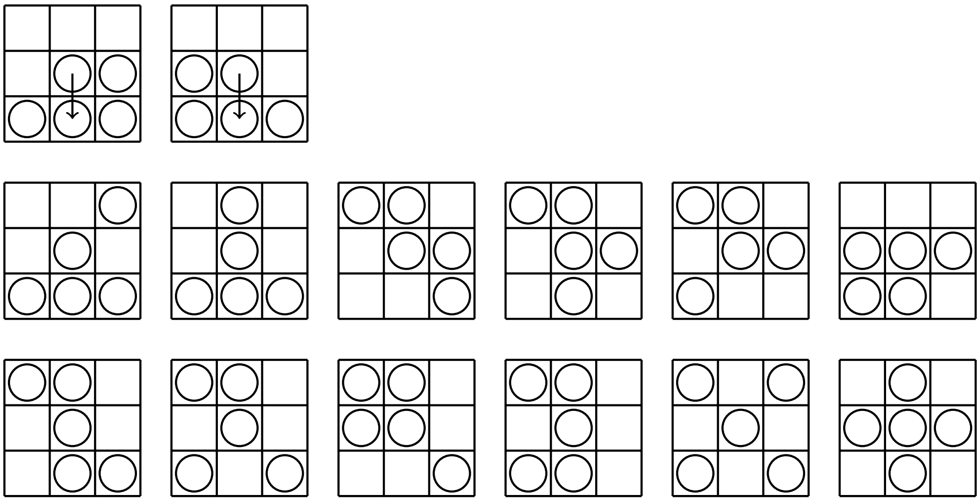
1.2 2 voisins



1.3 3 voisins



1.4 4 voisins



1.5 5 voisins et plus

Pour énumérer les cas avec 5 et 6 robots voisins, il suffit de prendre le complémentaire des cas avec respectivement 3 et 2 robots voisins. Aucun de ces cas n'entraine un mouvement de la part du robot concerné.

1.6 Extension

Tel quels, les cas 6 et 13 peuvent conduire à une déconnexion de l'espace. Pour y remédier, il faut que chaque robot mémorise son entourage d'une ronde sur l'autre (il ne retient que son entourage précédent.) Puis, si à la ronde précédente, il était dans le cas 6 ou 13, il faut qu'il vérifie si au moins une des cases suivante contient un robot : à droite, en bas et en bas à droite. Si ce n'est pas le cas, il revient à sa position précédente. Les cas symétriques sont définis de façon analogues.

Après avoir réglé ces cas de déconnexions, un autre problème survient avec les cas 5 et 6. Il se peut que l'espace alterne entre deux états ce qui rend le rassemblement impossible. Le problème vient du fait que des robots disposés en quinconce soient de nouveau en quinconce à la ronde suivante (avec des positions inversés.) Et ainsi, revenir à la position qu'ils occupaient deux rondes plus tôt. Pour y remédier, on va de nouveau utiliser l'entourage de la ronde précédente : Si à la ronde précédente, un robot était dans le cas 5 ou 6, et qu'il est désormais dans le cas opposé, alors il ne bouge pas pour cette ronde.

1.7 Formalisation

Algorithme 1 :

```
finish ← False;
ok ← True;
k ← 0;
while not finish do
    N_k ← get_neighbors();
    if k%4 = 0 then
        if N_k is case 1.2.{4,5} or 1.3.{5,6} then
            move to (i, j − 1);
            get_neighbors();
            if (i, j − 2) or (i + 1, j − 2) is not empty then
                ok ← False;
            end
        end
    else if k%4 = 1 then
        if N_{k−1} is case 1.2.4 or 1.3.5 then
            move to (i, j + 1);
        end
    else if k%4 = 2 then
        if N_k is case 1.1.1 or
        (N_k is case 1.1.{2,3,4} and N_{k−2} = rotate180(N_k)) or
        (N_k is case 1.3.2 and N_{k−2} = rotate90(N_k)) then
            finish ← True;
        else if ok = True then
            move according to N_k;
        end
        ok ← True;
    else
        if (N_{k−1} is case 1.2.{4,5} or 1.3.{5,6}) and ((i, j + 1), (i − 1, j), (i − 1, j + 1) are all
        empty) then
            move to (i − 1, j);
        end
    end
    k ← k + 1;
end
```

---

2 Calcul du temps de rassemblement dans le cas d'un bloc (draft)

2.1 Cas paire

Bloc de taille n par n avec $n \geq 3$.
On ne considère que un côté (les trois autres sont similaires)
R : le plus petit rectangle englobant tous les robots

Première étape De bloc à disque

Les deux cases prises par les robots sur les bords du côtés se libèrent à chaque ronde. Donc à chaque ronde, le nombre de robots sur le côté diminue de 2. Le temps pour qu'il ne reste plus que 4 robots est

$$\frac{n-4}{2}$$

Deuxième étape Effondrement du disque (ou du cercle)

Le côté ne contient plus que 4 robots adjacents. Donc au bout de deux étapes, il s'effondre. On se rend compte que tant que $n \geq 3$, après chaque effondrement, le côté contient de nouveau 4 robots adjacents. Donc, pour que l'espace devienne un bloc de taille 2 par 2 (cas terminal), il faut que chaque côté "descendent" de $n/2 - 1$, soit un nombre d'étapes de

$$(\frac{n}{2} - 1) * 2$$

Puis il faut une dernière ronde pour que les robots se rendent compte qu'ils ont terminés.

Totaux Le temps total T pour que un bloc se rassemble est donc :

$$T = (\frac{n}{2} - 1) * 2 + \frac{n-4}{2} + 1$$

$$T = n + \frac{n}{2} - 3$$

2.2 Cas impaire

Le cas impaire est assez similaire : On utilise des côtés qui ont 3 robots adjacents, et "descendent" en deux étapes.

Première étape

$$\frac{n-3}{2}$$

Deuxième étape

$$(\frac{n-1}{2}) * 2$$

Pas de ronde en plus après la deuxième étape.

2.3 Cas général (paire et impaire)

$$n + \left\lceil \frac{n}{2} \right\rceil - 3$$

Formule vérifiée pour des blocs de côtés de taille 3 à 50 de côtés de taille 3 à 100.

2.4 Généralisation aux rectangles

On considère des rectangles de taille m par n avec $m \leq n$.

$$\left\lceil \frac{n-4}{2} \right\rceil + 2 * \left\lfloor \frac{m}{2} \right\rfloor - 2 \; si \; m \; pair + 1 \; si \; m \; ou \; n \; pair$$

2.5 Généralisation aux carrés (blocs vides)

Tout ce qui a été dis sur les blocs peut s'appliquer (et a été vérifié) aux carrés et aux rectangles. On peut associer les robots qui forment le contour du bloc aux robots qui forment le cercle. Puis, on se rend compte que à chaque étape, les robots correspondant deux à deux prendront la même décision.