TP Optimisation – session 5 – Advent of Code

17 octobre 2025

Objectif d'apprentissage : cette séance vise à modéliser un problème issu de l'*advent of code* en programmation linéaire. Il s'agit du jour 19 de l'année 2022. Le scénario est le suivant.

Vous remarquez une collection de géodes [1] autour d'un étang rempli d'obsidiennes [2]. Peut-être pourriez-vous utiliser l'obsidienne pour créer des robots craqueurs de géodes et les ouvrir (on les appellera **robots géodes**) ?

Pour collecter l'obsidienne au fond de l'étang, vous aurez besoin de robots collecteurs d'obsidienne étanches (appelés **robots obsidiennes**). Heureusement, il y a une grande quantité d'argile à proximité que vous pouvez utiliser pour les rendre imperméables.

Pour récolter l'argile, vous aurez besoin de robots collecteurs d'argile spéciaux (appelés **robots argiles**). Pour fabriquer n'importe quel type de robot, vous aurez besoin de minerai, qui est également abondant, mais dans la direction opposée à celle de l'argile.

Pour collecter du minerai, il faut des robots collecteurs de minerai dotés de grosses foreuses (appelés **robots minerais**). Heureusement, vous avez exactement un robot minerai dans votre pack que vous pouvez utiliser pour lancer toute l'opération.

Chaque robot peut collecter une ressource de son type par minute. Il faut également une minute à l'usine de robots (qui se trouve également dans votre paquetage) pour construire n'importe quel type de robot en consommant les ressources nécessaires disponibles au début de la construction.

L'usine de robots dispose de deux plans parmi lesquels vous pouvez choisir, mais une fois que vous l'avez configurée avec un plan, vous ne pouvez plus le modifier.

Par exemple :

```
Plan 1:
  Chaque robot minerai coûte 4 minerais.
  Chaque robot argile coûte 2 minerais.
  Chaque robot obsidienne coûte 3 minerais et 14 argiles.
  Chaque robot géode coûte 2 minerais et 7 obsidiennes.

Plan 2:
  Chaque robot minerai coûte 2 minerais.
  Chaque robot argile coûte 3 minerais.
  Chaque robot obsidienne coûte 3 minerais et 8 argiles.
  Chaque robot géode coûte 3 minerais et 12 obsidiennes.
```

Vous devez trouver le plan qui maximisera le nombre de géodes ouvertes au bout de 24 minutes en déterminant quels robots construire et à quels moments les construire.

En utilisant le plan 1 dans l'exemple ci-dessus, le plus grand nombre de géodes que vous pouvez ouvrir en 24 minutes est 9. Une façon d'y parvenir est :

```
== Minute 1 ==
1 robot minerai collecte 1 minerai; vous avez maintenant 1 minerai.

== Minute 2 ==
1 robot minerai ramasse 1 minerai; vous avez maintenant 2 minerais.
```

1. Une cavité rocheuse tapissée de cristaux.
2. Une roche volcanique vitreuse et riche en silice.

```
== Minute 3 ==
Dépensez 2 minerais pour commencer à construire un robot argile.
1 robot minerai collecte 1 minerai; vous avez maintenant 1 minerai.
Le nouveau robot argile est prêt; vous en avez maintenant 1.

== Minute 4 ==
1 robot minerai collecte 1 minerai; vous avez maintenant 2 minerais.
1 robot argile ramasse 1 argile; vous avez maintenant 1 argile.

== Minute 5 ==
Dépensez 2 minerais pour commencer à construire un robot argile.
1 robot minerai collecte 1 minerai ; vous avez maintenant 1 minerai.
1 robot argile ramasse 1 argile ; vous avez maintenant 2 argiles.
Le nouveau robot argile est prêt ; vous en avez maintenant 2.

== Minute 6 ==
1 robot minerai collecte 1 minerai ; vous avez maintenant 2 minerais.
2 robots argiles collectent 2 argiles ; vous avez maintenant 4 argiles.

== Minute 7 ==
Dépensez 2 minerais pour commencer à construire un robot argile.
1 robot minerai collecte 1 minerai ; vous avez maintenant 1 minerai.
2 robots argiles collectent 2 argiles ; vous avez maintenant 6 argiles.
Le nouveau robot argile est prêt ; vous en avez maintenant 3.

== Minute 8 ==
1 robot minerai collecte 1 minerai ; vous avez maintenant 2 minerais.
3 robots argiles collectent 3 argiles ; vous avez maintenant 9 argiles.

== Minute 9 ==
1 robot minerai ramasse 1 minerai ; vous avez maintenant 3 minerais.
3 robots argiles collectent 3 argiles ; vous avez maintenant 12 argiles.

== Minute 10 ==
1 robot minerai ramasse 1 minerai ; vous avez maintenant 4 minerais.
3 robots argiles collectent 3 argiles ; vous avez maintenant 15 argiles.

== Minute 11 ==
Dépensez 3 minerais et 14 argiles pour commencer à construire un robot obsidienne.
1 robot minerai collecte 1 minerai ; vous avez maintenant 2 minerais.
3 robots argiles collectent 3 argiles ; vous avez maintenant 4 argiles.
Le nouveau robot obsidienne est prêt ; vous en avez maintenant 1.

...

== Minute 21 ==
Dépensez 2 minerais et 7 obsidiennes pour commencer la construction d'un robot géode.
1 robot minerai collecte 1 minerai ; vous avez maintenant 3 minerais.
4 robots argiles collectent 4 argiles ; vous avez maintenant 29 argiles.
2 robots obsidiennes collectent 2 obsidiennes ; vous avez maintenant 2 obsidiennes.
1 robot géode brise 1 géode ; vous avez maintenant 3 géodes ouvertes.
Le nouveau robot géodes est prêt ; vous en avez maintenant 2.

== Minute 22 ==
1 robot minerai collecte 1 minerai ; vous avez maintenant 4 minerais.
```

```
4 robots argiles collectent 4 argiles ; vous avez maintenant 33 argiles.
2 robots obsidiennes ramassent 2 obsidiennes ; vous avez maintenant 4 obsidiennes.
2 robots géodes cassent 2 géodes ; vous avez maintenant 5 géodes ouvertes.

== Minute 23 ==
1 robot minerai ramasse 1 minerai ; vous avez maintenant 5 minerais.
4 robots argiles collectent 4 argiles ; vous avez maintenant 37 argiles.
2 robots obsidiennes ramassent 2 obsidiennes ; vous avez maintenant 6 obsidiennes.
2 robots géodes cassent 2 géodes ; vous avez maintenant 7 géodes ouvertes.

== Minute 24 ==
1 robot minerai ramasse 1 minerai ; vous avez maintenant 6 minerais.
4 robots argiles collectent 4 argiles ; vous avez maintenant 41 argiles.
2 robots obsidiennes ramassent 2 obsidiennes ; vous avez maintenant 8 obsidiennes.
2 robots géodes cassent 2 géodes ; vous avez maintenant 9 géodes ouvertes.
```

Cependant, en utilisant le plan 2 dans l'exemple ci-dessus, on souhaite savoir si on peut faire encore mieux.

Pour résoudre ce problème, il sera nécessaire d'utiliser des variables entières. Il ne s'agira donc pas d'un programme linéaire, mais d'un programme entier, ou programme linéaire en nombre entier (*Integer Linear Programming*). Dans ce cas, la résolution n'est plus polynomiale en temps mais s'il n'y a pas trop de variables entières, la résolution restera rapide.

Pour modéliser ce problème, il faut 4×24 variables binaires : une variable par type de robots que l'on décide de construire à chaque minute. On aura $x_{rm} \in \{0, 1\}$ pour $1 \leq r \leq 4$ (1 correspondant au robot minéralier, etc.) et $1 \leq m \leq 24$.

L'objectif consiste à maximiser le nombre de géodes ouvertes. Avec le plan 1, les deux robots géodes ont été produits aux minutes 18 et 21. Ils ont donc permis d'ouvrir $(24 - 18) + (24 - 21) = 6 + 3 = 9$ géodes. Pour modéliser avec les variables binaires, on a $x_{4m} = 1$ pour $m \in \{18, 21\}$ et $x_{4m} = 0$ dans les autres cas. La formulation générale de l'objectif est donc la suivante :

$$\sum_{m=1}^{24} (24 - m) \times x_{4m}$$

Il faut maintenant rajouter les contraintes suivants :

— Un seul robot construit à chaque minute.

— Suffisamment de minerai pour construire n'importe quel robots à chaque minute.

— Suffisamment d'argile pour construire un robot obsidienne à chaque minute.

— Suffisamment d'obsidienne pour construire un robot géode à chaque minute.

Pour la première contrainte, il s'agit de contraindre les variables x_{1m}, x_{2m}, x_{3m} et x_{4m}. Par exemple à la minute 1, on aurait $x_{11} + x_{21} + x_{31} + x_{41} \leq 1$. En généralisant, on a $\sum_{r=1}^{4} x_{rm} \leq 1$ pour $1 \leq m \leq 24$.

Pour la contrainte sur l'argile, il s'agit de contrôler les variables x_{3m}. Par exemple à la minute 4, on comptabiliserait d'un coté la quantité d'argile récoltée et on enlèverait l'argile qui aurait déjà été consommé pour construire des robots obsidiennes. Avec le plan 1, il faut que cette quantité dépasse 14 si $x_{34} = 1$. La quantité d'argile récoltée à la minute 4 est $(4-1-1)x_{21} + (4-2-1)x_{22} + (4-3-1)x_{23}$. La quantité d'argile consommée est : $14x_{31}+14x_{32}+14x_{33}$. On obtient donc $(4-1-1)x_{21}+(4-2-1)x_{22}+(4-3-1)x_{23}-(14x_{31}+14x_{32}+14x_{33}) \geq 14x_{34}$. En généralisant, on a $\sum_{m'=1}^{m-1}(m-m'-1)x_{2m'} - bx_{3m'} \geq bx_{3m}$ pour $1 \leq m \leq 24$ avec $b = 14$.

Sans les deux dernières contraintes, le programme entier est donc :

$$\text{maximiser} \quad \sum_{m=1}^{24} (24-m) x_{4m}$$

$$\text{sous les contraintes} \quad \sum_{r=1}^{4} x_{rm} \leq 1 \qquad \text{pour } 1 \leq m \leq 24$$

$$\sum_{m'=1}^{m-1} (m-m'-1) x_{2m'} - 14 x_{3m'} \geq 14 x_{3m} \quad \text{pour } 1 \leq m \leq 24$$

$$x_{rm} \in \{0,1\} \quad \text{pour } 1 \leq r \leq 4 \text{ et } 1 \leq m \leq 24$$

Compléter avec les deux dernières contraintes : la contrainte sur l'obsidienne et celle sur le minerai. Comme l'obsidienne n'est utilisé que pour les robots géodes, la situation est analogue à l'argile qui n'est utilisée que pour les robots obsidiennes. On peut donc simplement l'adapter en changeant les types de machines (de 2 vers 3 et 3 vers 4) et en adaptant les coûts (de 14 vers 7).

Pour la contrainte sur le minerai, il faut étendre la formule pour prendre en compte le minerai consommé pour la construction de n'importe quel robot (plutôt que les robots d'un certain type). Le principe est cependant très semblable. Attention, il ne faut pas oublier d'inclure le minerai produit par le robot initial qui se trouvait déjà dans votre sac.

On va pouvoir désormais coder ce programme entier pour résoudre le problème. Pour indiquer qu'une variable est entière, il faut l'indiquer à `pulp` (ici pour une variable booléenne) :

```
x = LpVariable("Variable booléenne", LpBinary)
```

Certaines contraintes sont déjà codées dans le code suivant. Lesquelles ? Compléter avec l'ensemble des contraintes.

```
from pulp import *

a = [4, 2, 3, 2]
b = 14
c = 7
T = 24

vars = LpVariable.dicts("Robots", (range(4), range(T)), cat = LpBinary)

prob = LpProblem("Géodes", LpMaximize)
prob += lpSum((T - m - 1) * vars[3][m] for m in range(T)), "Géodes ouvertes"
for m in range(T):
    prob += lpSum(vars[r][m] for r in range(4)) <= 1
    prob += m + lpSum((m - mm - 1) * vars[0][mm] -
            lpSum(a[r] * vars[r][mm] for r in range(4))
        for mm in range(m)) >= lpSum(a[r] * vars[r][m] for r in range(4))

prob.solve()
print("Cout total = ", value(prob.objective))
```

Quel est le plus grand nombre de géodes que vous pouvez ouvrir en 24 minutes.