INF8225

Prise de décisions

Théorie des décisions et graphes de décision,
prise de décisions complexes,
processus de décision markovien.

Par Christopher Pal, avec quelques diapositives de Michel Gagnon

Dans des systèmes plus autonomes on parle généralement « d'agents » avec

1. Une correspondance formelle entre les conditions sur l 'état courant et les actions possibles
2. Un moyen d'inférer les propriétés pertinentes du monde à partir de la séquence de percepts.
3. Des informations sur la façon dont le monde évolue et sur les résultats des actions que l'agent pourrait effectuer
4. Des informations sur la désirabilité indiquant l'utilité des états du monde
5. Des informations sur la valeur des actions indiquant leur désirabilité
6. Des buts décrivant des classes d'états dont l'atteinte maximise l'utilité de l'agent

Motivation

- Très souvent, un agent ne peut pas déterminer avec certitude quel sera l'effet de ses actions
- Par contre, on peut estimer la probabilité de chaque situation pouvant résulter d'une action
- On peut aussi évaluer dans quelle mesure chaque état résultant est utile ou désirable
- Alors, comment choisir la meilleure action?

Principe du maximum d'utilité espérée (MUE / MEU en anglais)

- Soit :
  - $Résultat_i(A)$ - le *i*ème état résultat possible de l'action A
  - *U(S)* - une fonction qui retourne la valeur d'utilité de l'état *S*
  - *E* - l'ensemble des informations sur l'état actuel dont l'agent dispose
- L'action choisie devrait être celle qui maximise EU(A|E) définie ainsi:

$$EU(A\,|E\,) = \sum_i P(Résultat_i(A)|faire(A),E) \times U(Résultat_i(A))$$

Conditions sur la fonction d'utilité

- Pour que le principe de maximum d'utilité espérée nous assure un comportement rationnel, certaines conditions doivent être respectées par la fonction d'utilité:
  - L'agent doit savoir s'il préfère un état à un autre, ou s'il est indifférent entre les deux
  - Si l'agent préfère A à B et B à C,
    il doit préférer A à C
  - Autres conditions à la page 657 du manuel...
- Idée clé: Les axiomes de la théorie de l'utilité ne disent rien sur l'utilité elle-même -- ils ne parlent que de préférences (relatif)

Exemple: Transitivité

- Un agent dont les préférences intransitives peut être amené à donner tout son argent

Avec des préférences:

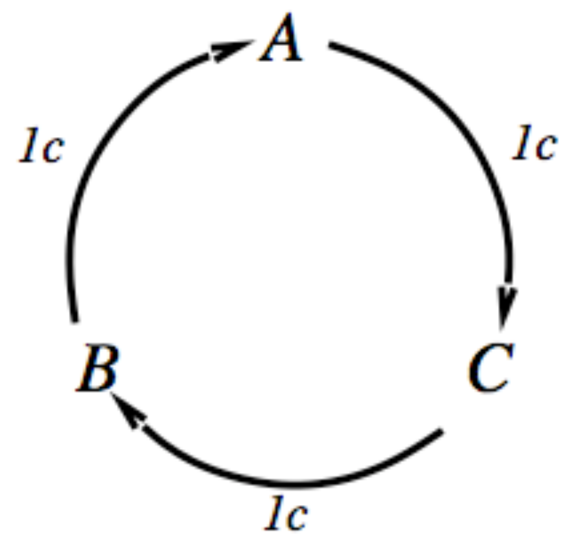
- B > C
- A > B

Il aura des problèmes

si C > A

et nous avons une situation

d'adversité



Puis l'utilité vint…

L'existence d'une fonction d'utilité découle des axiomes :

- **Principe d'utilité***:* Si les préférences d'un agent obéissent aux axiomes de l'utilité, il existe une fonction à valeurs réelles U qui opère sur les états de telle façon que
  - U(A) > U(B) si et seulement si A est préféré à B et
  - U(A) = U(B) si et seulement si l'agent est indifférent entre A à B
- **Principe du maximum de l'utilité espérée :** L'utilité d'une loterie est la somme de la probabilité de chaque résultat multipliée par l'utilité de ce résultat

$$U([p_1, S_1; \ldots, p_n, S_n]) = \sum_i p_i U(S_i)$$

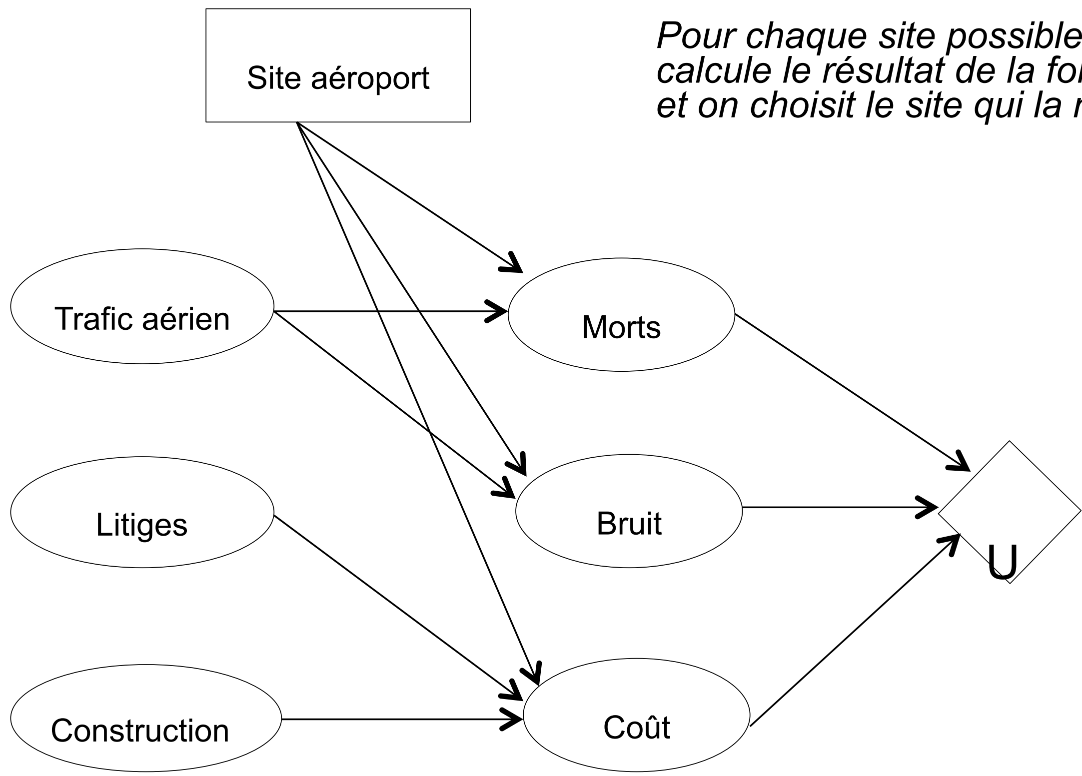
Réseaux de décision

- Il s'agit essentiellement de réseaux bayesiens
- On distingue trois types de noeuds:
  - *noeuds de hasard,* représentant les variables aléatoires (**les cercles**)
  - *noeuds de décision,* représentant des variables dont la valeur doit être sélectionnée parmi toutes celles possibles (**les rectangles**)
  - *noeuds d'utilité*, représentant la valeur d'utilité pour un ensemble de variables (**les diamants**)

Évaluation d'un réseau de décision

1. Établir les valeurs connues parmi les noeuds du réseau
2. Pour chaque valeur possible du noeud de décision :
   a) Fixer le noeud à cette valeur
   b) Calculer les probabilités de tous les noeuds reliés au noeud d'utilité
   c) Calculer la valeur d'utilité
3. Choisir l'action ayant la valeur d'utilité la plus élevée

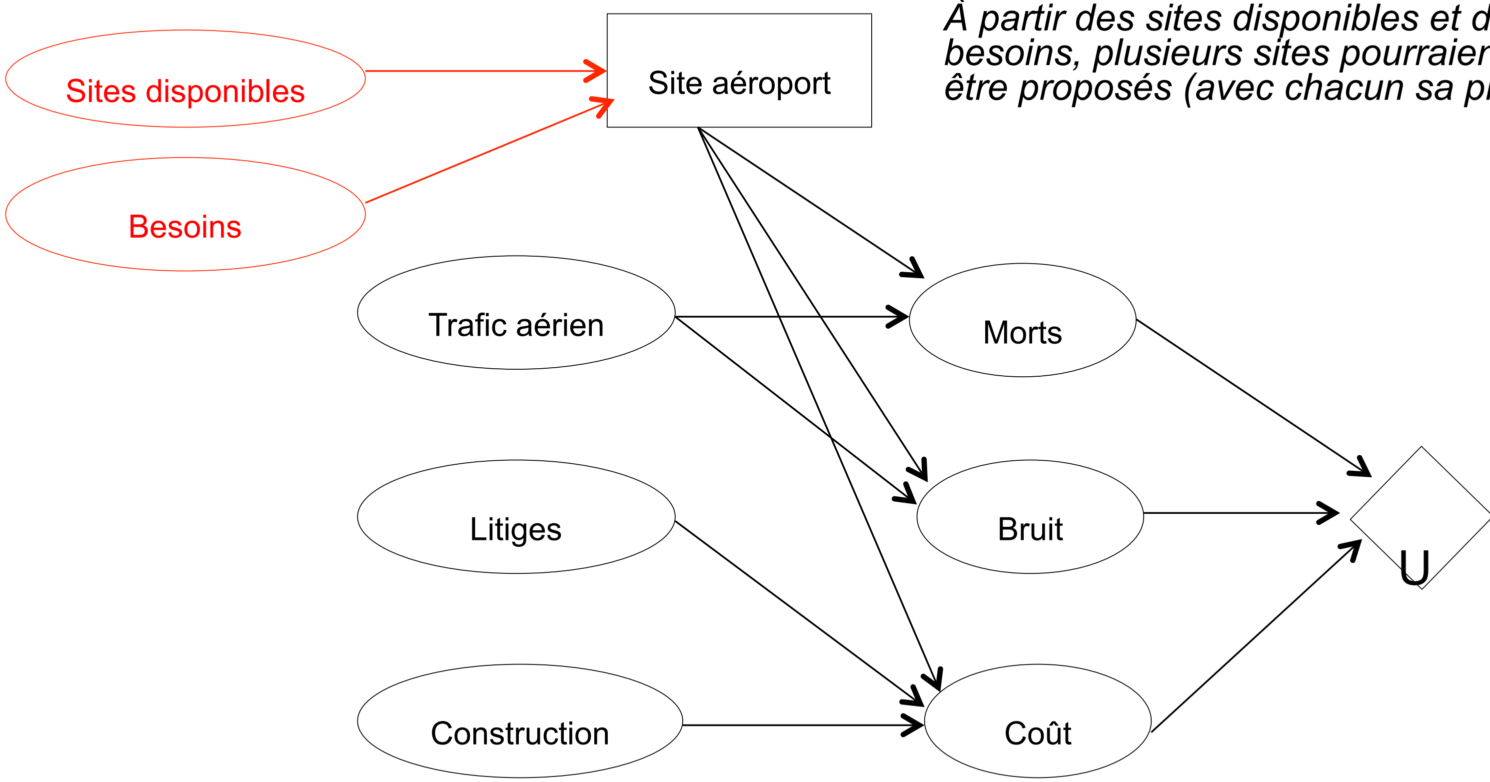
Exemple de réseau de décision



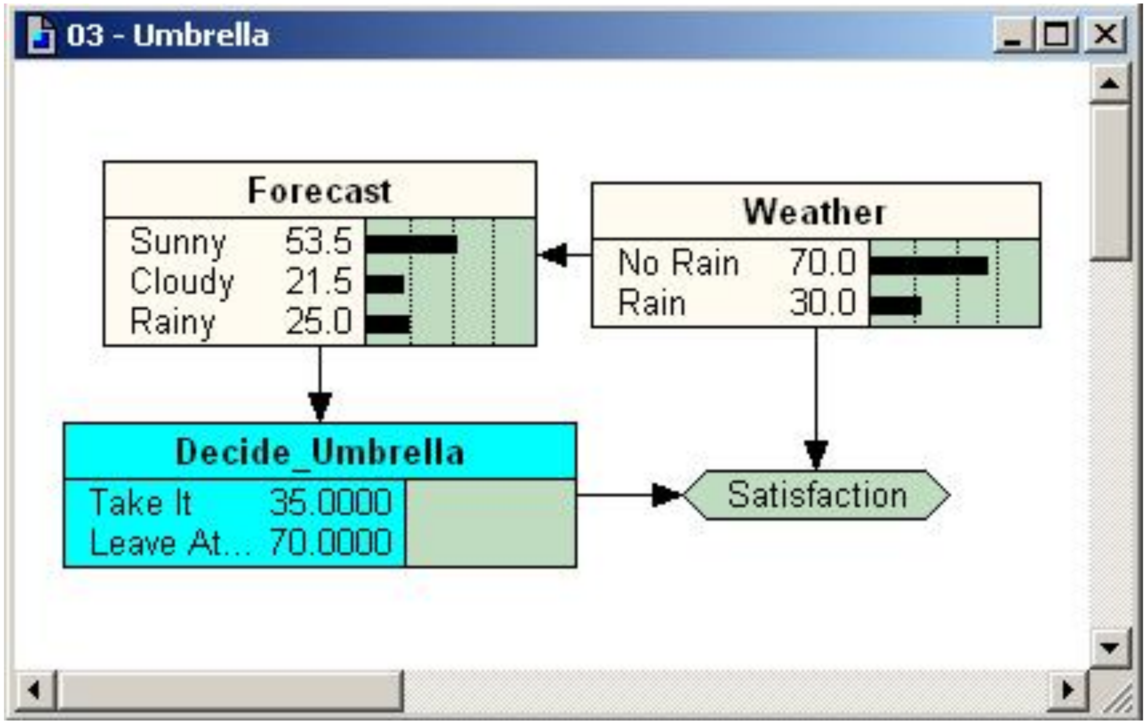
Pour chaque site possible, on calcule le résultat de la fonction d'utilité et on choisit le site qui la maximise.

Exemple de réseau de décision

À partir des sites disponibles et des besoins, plusieurs sites pourraient être proposés (avec chacun sa probabilité).



Un réseau de décision dans Netica



http://www.norsys.com/tutorials/netica/secA/tut_A4.htm

Décisions complexes

- Jusqu'à maintenant, on a supposé que l'agent n'a qu'une seule décision à prendre
- Que fait-on lorsque plusieurs actions consécutives sont nécessaires pour atteindre le but visé?
- Nous verrons que, dans ce cas, l'agent doit avoir <u>une politique </u>pour choisir la meilleure action à chaque état

Processus de décision Markovien

- Modèle de transition: $T(s,a,s') = P(s' \mid s, a)$ représente la probabilité d'obtenir l'état *s'* si on exécute l'action *a* à l'état *s*
- État initial: S_0
- Fonction de récompense: $R(s)$
- La solution est une *politique* $\pi(s)$, qui spécifie l'action recommandée pour chaque état *s*

Politique optimale

- Une politique optimale, notée $\pi^*(s)$, est une politique qui retourne la valeur la plus élevée pour l'utilité espérée
- Si l'horizon n'est pas infini, elle sera non-stationnaire (c'est-à-dire qu'elle ne retournera pas toujours la même action pour le même état)

Comment calculer l'utilité?

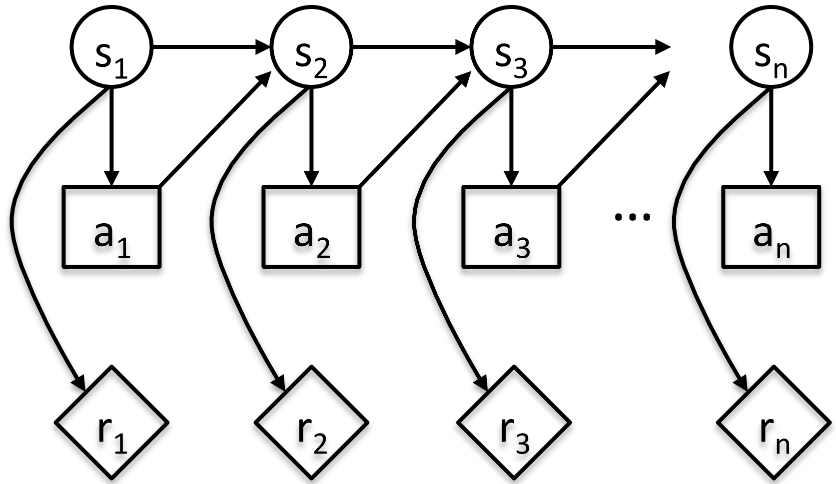
- On supposera que la préférence est stationnaire (si l'agent préfère la séquence $[s_0, s_1, s_2, \ldots]$ à la séquence $[s_0', s_1', s_2', \ldots]$, alors il préfère $[s_1, s_2, \ldots]$ à $[s_1', s_2', \ldots]$)
- Dans ce cas, on a une seule fonction d'utilité raisonnable avec les récompenses <u>escomptées</u>:

$$U([s_0, s_1, s_2, \ldots]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \ldots$$

où le facteur d'escompte γ est compris entre 0 et 1

- Mais, nous cherchons une fonction d'utilité qui tient compte de l'incertitude des états futurs

Un PDM comme un réseau de décision



- But: étant donné les paramètres $R(s)$, $P(s_1)=s_o$, $P(s_{i+1}|s_i, a_i)=T(s',s,a)$, γ
- Calculez la politique $\pi(s)$ qui maximise

$$E\left[\sum_{t=1}^{\infty}\gamma^t R(s_t) \mid \pi(s)\right] = V_\pi(s) = R(s) + \gamma \sum_{s'} P(s' \mid s, \pi(s)) V_\pi(s')$$

La somme infinie des récompenses espérée en forme récursive

$$E\left[\sum_{t=1}^{\infty}\gamma^t R(s_t) \mid \pi(s)\right]$$

$$= \sum_{s_1}\sum_{s_2}\cdots\sum_{s_\infty} P(s_1, s_2, \ldots, s_\infty \mid \pi(s))\left[\sum_{t=1}^{\infty}\gamma^t R(s_t)\right] \in$$

$$= R(s_1) + \gamma\sum_{s_2} P(s_2 \mid s_1, \pi(s_1))R(s_2) + \gamma^2\sum_{s_2} P(s_2 \mid s_1, \pi(s_1))\sum_{s_3} P(s_3 \mid s_2, \pi(s_2))R(s_3) + \gamma^3 \cdots$$

$$= R(s) + \gamma\sum_{s'} P(s' \mid s, \pi(s))\left[R(s') + \gamma\sum_{s''} P(s'' \mid s', \pi(s'))V_\pi(s'')\right]$$

$$= V_\pi(s)$$

$$V_\pi(s) = R(s) + \gamma\sum_{s'} P(s' \mid s, \pi(s))V_\pi(s')$$

Comment choisir entre différentes politiques?

- Donc, une politique optimale π^* est une politique π qui maximise la valeur espérée:

$$E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \mid \pi\right]$$

- Et, étant donné une fonction U, une seule étape, **politique gloutonne** peut être calculé à partir de

$$\pi(s) = \arg\max_a \sum_{s'} T(s,a,s')\, U(s')$$

- Par le principe de l'utilité maximale espérée, on déduit que, <u>si U est une fonction d'utilité d'une politique optimale, alors π est assuré d'être la politique optimale π^*</u>

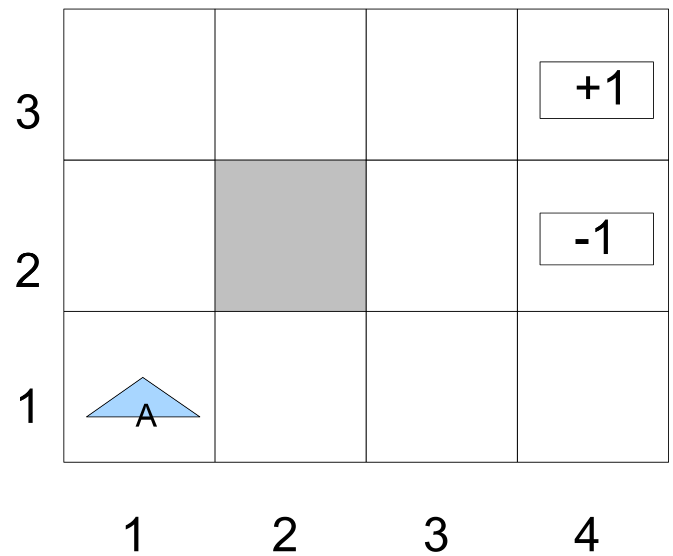
L’équation de Bellman

- On a aussi que: l’utilité d’un état est la récompense immédiate associée à cet état plus l’utilité escomptée espérée de l’état suivant, à supposer que l’agent choisisse l’action optimale

$$U(s) = R(s) + \gamma \max_{a} \sum_{s'} T(s,a,s')\, U(s')$$

- Voici la base de l’algorithme de « value iteration »
- Les itérations sont garanties à converger vers la valeur d'utilité d'une politique optimale,
- **Mais, elle ne nécessite pas l'utilisation d'une politique optimale!**

Exemple



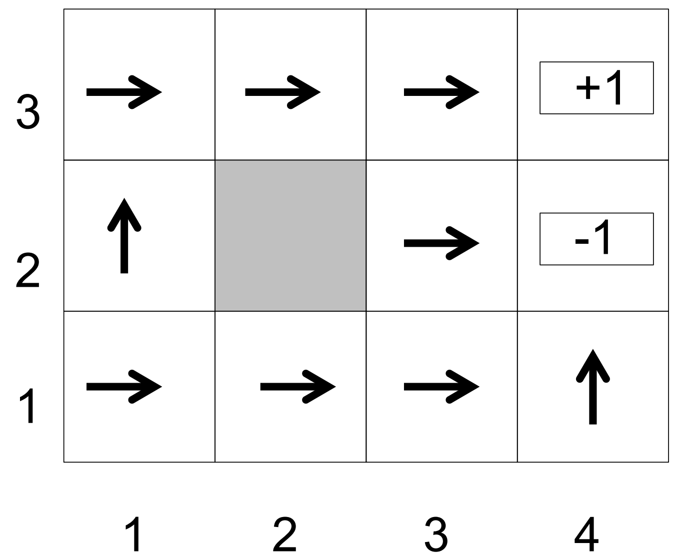
Récompense de -0,04 pour chaque case non finale
Probabilité d'avancer à la case suivante: 0,8
Probabilité de se déplacer sur une case latérale: 0,1

Exemple – politique optimale

3	→	→	→	+1
2	↑		↑	-1
1	↑	←	←	←
	1	2	3	4

Récompense de -0,04 pour chaque case non finale
Probabilité d'avancer à la case suivante: 0,8
Probabilité de se déplacer sur une case latérale: 0,1

Exemple – politique optimale



L'agent se précipite le plus vite possible vers l'état final le plus proche, même s'il est le moins intéressant

Récompense inférieure à -1,6284 pour chaque case non finale
Probabilité d'avancer à la case suivante: 0,8
Probabilité de se déplacer sur une case latérale: 0,1

Exemple – politique optimale

	1	2	3	4
3	→	→	→	+1
2	↑		←	-1
1	↑	←	←	↓

L'agent ne prend aucun risque

Récompense entre -0,0221 et 0 pour chaque case non finale
Probabilité d'avancer à la case suivante: 0,8
Probabilité de se déplacer sur une case latérale: 0,1

Algorithme – itération de la valeur

fonction ITERATION-VALEUR(*pdm*,ε) **retourne** fonction d'utilité
  entrées: *pdm,* un processus de décision markovien
  ε, erreur maximale permise
  U' est une fonction d'utilité initiale telle que U'(x) = 0
  S est l'ensemble des états de *pdm*

répéter:

  $U \leftarrow U'$
  $\delta \leftarrow 0$
  pour chaque $s \in S$ faire:
    $U'[s] \leftarrow R[s] + \gamma \max_a \sum_{s'} T(s,a,s')\, U[s']$
    si $|U'[s] - U[s]| > \delta$ **alors** $\delta \leftarrow |U'[s] - U[s]|$

jusqu'à $\delta < \varepsilon(1-\gamma)/\gamma$

retourner U

Itération de la valeur -simulation

3	0.0	0.0	0.0	0.0
2	0.0		0.0	0.0
1	0.0	0.0	0.0	0.0
	1	2	3	4

$\gamma = 1$

Itération de la valeur -simulation

	1	2	3	4
3	-0.04	-0.04	-0.04	1
2	-0.04		-0.04	-1
1	-0.04	-0.04	-0.04	-0.04

meilleure action: *droite*

U = -0,04 + 0,8*1 -

-0,1*0,04 – 0,1*0,04

= 0,752

meilleure action: *gauche*

U = -0,04 – 0,8*0,04

-0,1*0,04 – 0,1*0,04

= -0,08

$\gamma = 1$

Itération de la valeur -simulation

3	-0.08	-0.08	0,752	1
2	-0.08		-0.08	-1
1	-0.08	-0.08	-0.08	-0.08
	1	2	3	4

meilleure action: *droite*

U = -0,04 + 0,8*1 -

+0,1*0,752 – 0,1*0,08

= 0,827

meilleure action: *haut*

U = -0,04 + 0,8*0,752

-0,1*1 – 0,1*0,08

= 0,454

$\gamma = 1$

Itération de la valeur -simulation

3	-0.12	0,546	0,827	1
2	-0.12		0,454	-1
1	-0.12	-0.12	-0.12	-0.12
	1	2	3	4

$\gamma = 1$

Itération de la valeur -simulation

	1	2	3	4
3	0,372	0,731	0,888	1
2	-0.16		0,567	-1
1	-0.16	-0.16	0,299	-0.16

$\gamma = 1$

Itération de la valeur -simulation

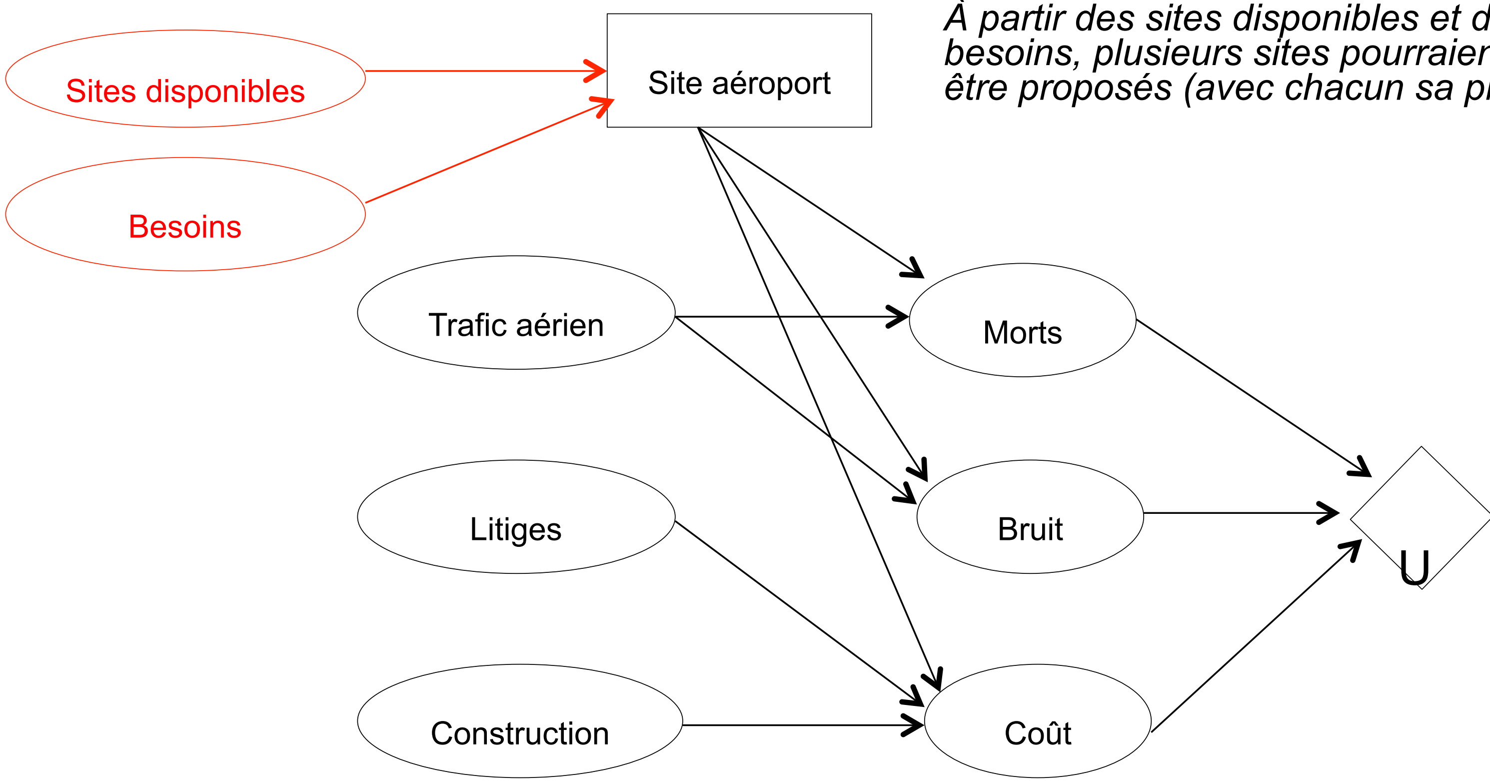
Valeurs finales:

3	0,812	0,868	0,918	1
2	0,762		0,660	-1
1	0,705	0,655	0,611	0,388
	1	2	3	4

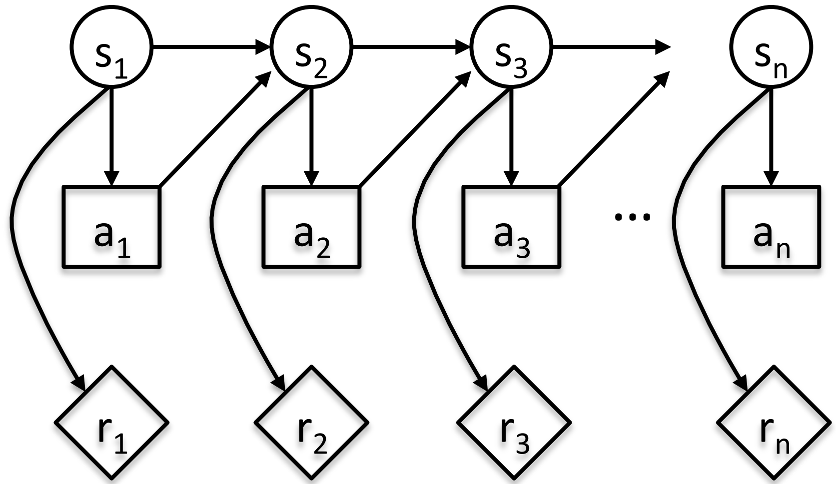
$\gamma = 1$

Exemple de réseau de décision

À partir des sites disponibles et des besoins, plusieurs sites pourraient être proposés (avec chacun sa probabilité).



Un PDM comme un réseau de décision



- But: étant donné paramètres
  $R(s)$, $P(s_1)=s_o$, $P(s_{i+1}|s_i,\ a_i)=T(s',s,a)$, γ
- Calculez la politique $\pi(s)$ qui maximise

$$E\left[\sum_{t=1}^{\infty}\gamma^t R(s_t) \mid \pi(s)\right] = V_\pi(s) = R(s) + \gamma \sum_{s'} P(s' \mid s, \pi(s)) V_\pi(s')$$

Calculs fondamentaux avec des PDM

- **La récupération d'une politique gloutonne** $\pi_g(s)$, étant donné une fonction de la valeur **v**(s) (ex. approximation à la vraie utilité de chaque état) Notez que quand **v** = **v*** est la fonction d'une politique optimale, puis π* est optimale aussi
- <u>**Itération de la valeur**</u> : une procédure itérative pour calculer la fonction de valeur d'une politique optimale, sans la nécessité d'avoir une politique optimale,
- Avec ces deux idées, étant donné un modèle P(s'|s,a) et R(s) on peut calculer une politique optimale

Calculs fondamentaux avec des PDM

- **L'obtention de l'utilité de chaque état, ou « Policy Evaluation » :** étant donné une politique π(s), et un modèle de transition P(s'|s,a) connu, il est possible d'obtenir l'utilité U(s) ou la fonction de valeur **v**(s) de chaque état
- **<u>Itération de la politique</u>** : une (autre) procédure itérative pour calculer une politique optimale étant donné P(s'|s,a) et r(s)
  - Initialiser $\pi_0(s)$ (ex. aléatoirement) et évaluera $\mathbf{v}_0(s)$
  Répéter
  - Calculer *la politique gloutonne* « one step » $\pi_{i+1}(s)$
  - Utiliser « policy evaluation » à obtenir $\mathbf{v}_{i+1}(s)$
  - Arrêter quand la différence entre $\mathbf{v}_{i+1}$ et $\mathbf{v}_i$ est petite

L'évaluation d'une politique

- Étant donné une politique π(s), on peut obtenir la fonction de la valeur **v**(s) « the value function » (ou la fonction d'utilité U(s)) avec la solution d'un système linéaire d'équations

$$\mathbf{v}_\pi(s) = \mathbf{r}(s) + \gamma \sum_{s'} \mathbf{v}_\pi(s') P(s' \mid s, \pi(s))$$

$$\mathbf{v} = \mathbf{r} + \gamma \mathbf{P} \mathbf{v}$$

$$(I - \gamma \mathbf{P})\mathbf{v} = \mathbf{r}$$

$$\mathbf{v} = (I - \gamma \mathbf{P})^{-1} \mathbf{r}$$

- Étant donné les actions a=π(s) et donc une matrice P, composée de P(s'|s,a)

Calculs fondamentaux avec des PDM

- **L'obtention de l'utilité de chaque état, ou « Policy Evaluation » :** étant donné une politique π(s), et un modèle de transition P(s'|s,a) connu, il est possible d'obtenir l'utilité U(s) ou la fonction de valeur **v**(s) de chaque état
- **<u>Itération de la politique</u>** : une (autre) procédure itérative pour calculer une politique optimale étant donné P(s'|s,a) et r(s)
  - Initialiser $\pi_0(s)$ (ex. aléatoirement) et évaluera $\mathbf{v}_0(s)$
  Répéter
  - Calculer *la politique gloutonne* « one step » $\pi_{i+1}(s)$
  - Utiliser « policy evaluation » à obtenir $\mathbf{v}_{i+1}(s)$
  - Arrêter quand la différence entre $\mathbf{v}_{i+1}$ et $\mathbf{v}_i$ est petite

Itération de la valeur vs. politique

- Nous sommes assurés de converger vers la solution optimale avec les deux approches : a) l'itération de la valeur et b) l'itération de la politique
- Toutefois chaque itération est (possiblement) moins chère avec l'itération de la valeur parce qu'on n'a pas besoin d'évaluer la politique (ex. obtenir la solution à un système d'équations linéaire)

Algorithme – itération de la politique

- En principe l'évaluation d'une politique exige la résolution d'une série d'équations linéaires
- On peut simplifier en itérant *k* fois la mise à jour suivante:

$$U_{i+1}(s) \leftarrow R(s) + \gamma \sum_{s'} T(s,\pi_i(s),s')\, U_i[s']$$

Algorithme – itération de la politique

fonction ITERATION-POLITIQUE(*pdm*,ε) **retourne** fonction d'utilité

  entrées: *pdm,* un processus de décision markovien

    ε, erreur maximale permise

  π est une politique initialisée aléatoirement

  S est l'ensemble des états de *pdm*

  répéter:

    U ← EVALUATION-POLITIQUE(π, U, *pdm*)

    inchangé ← *true*

    pour chaque s ∈ S faire:

      si $\max_{a} \sum_{s'} T(s,a,s')\, U[s'] > \sum_{s'} T(s,\pi[s],s')\, U[s']$ **alors**

        $\pi[s]$ **=** $\arg\max_{a} \sum T(s,a,s')\, U[s']$

        inchangé = *false*

    jusqu'à *inchangé*

  retourner π

Apprentissage par renforcement

- Apprentissage passif:
  - <u>On ne connaît pas le modèle de transition</u>
  - On connaît la politique
  - On veut déterminer les valeurs d'utilité
- Apprentissage actif:
  - <u>On ne connaît pas le modèle de transition</u>
  - On **ne connaît pas** la politique
  - On veut déterminer la politique

Apprentissage passif: estimation directe

- On fait un ensemble d'essais, chacun comprenant une séquence de transitions
- Pour chaque occurrence d'une transition, on calcule l'utilité
- Pour chaque transition, on calcule la moyenne des valeurs obtenues
- Cet algorithme converge lentement

Apprentissage passif: programmation dynamique adaptative

fonction AGENT-PDA-PASSIF(s', r') **retourne** une action
entrées: s', état actuellement perçu
r', récompense actuellement perçue
π est une politique fixe
pdm est le processus de décision markovien défini par U, R et T
s, a, état et action précédents, initialement nuls
si s' est nouveau **alors faire** $U[s'] \leftarrow r'$; $R[s'] \leftarrow r'$
si s' non nul **alors faire**
incrémenter $N_{sa}[s,a]$ et $N_{sas'}[s,a,s']$
pour tout t tel que $N_{sas'}[s,a,t] \neq 0$ **faire**
$T[s,a,t] \leftarrow N_{sas'}[s,a,t] \;/\; N_{sa}[s,a]$
$U \leftarrow$ DETERMINER-VALEUR(π, U, *pdm*)
si TERMINAL?[s'] **alors** $s,a \leftarrow$ nul **sinon** $s,a \leftarrow s', \pi[s']$
retourner a

Apprentissage actif

- Rappel, ici :
  - <u>On ne connaît pas le modèle de transition</u>
  - On **ne connaît pas** la politique
  - On veut déterminer la politique
- On peut adapter l'algorithme d'apprentissage passif avec un « model based approach »
- On apprend d'abord le modèle de transition
- Puis on applique un algorithme pour obtenir la politique optimale
- On répète

Apprentissage actif

- Une question: l'agent doit-il choisir la prochaine action selon sa politique optimale courante?
- Réponse: pas toujours.
- L'agent doit aussi faire un peu d'exploration
- Une méthode simple: choisir une action aléatoirement une fois de temps en temps
- Une meilleure approche: donner du poids à des actions rarement choisies, tout en essayant d'éviter des actions qui sont reconnues pour leur peu d'utilité

Apprentissage actif (suite)

- Comment faire?
- On utilise l'équation suivante pour la mise à jour d'une valeur d'utilité :

$$U^+(s) = R(s) + \gamma \max_{a} f\Big(\sum_{s'} T(s,a,s')U^+(s'), N(a,s) \Big)$$

$$f(u,n) = \begin{cases} R^+ & \text{si } n < K \\ u & \text{sinon} \end{cases}$$

K est une constante fixée

R^+ est un estimé de la meilleure récompense possible

Q-Learning

- $U'[s] \leftarrow R[s] + \gamma \max_a \sum_{s'} T(s,a,s')\, U[s']$