# Agents à base de connaissances

Thi-Bich-Hanh Dao

M1 Informatique - Université d'Orléans

Année 2012-2013



## Aperçu



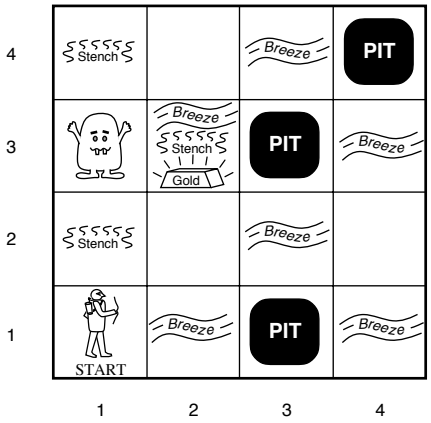


## Motivation

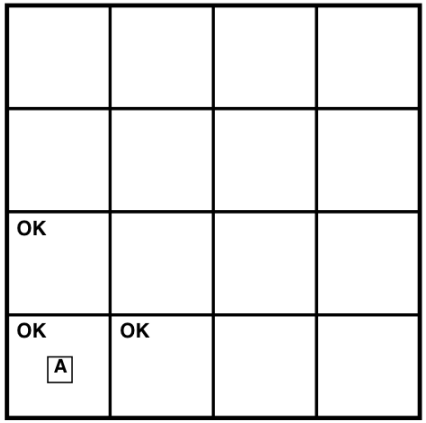
- Jusqu'ici nous avons vu comment manipuler des symboles pour résoudre des problèmes en utilisant des méthodes de recherche
- Caractéristique des méthodes utilisées :
  - très générales
  - ne prennent pas en compte comment la connaissance est représentée
  - la connaissance est "cachée" dans les définitions des états et dans les fonctions heuristiques
- Questions :
  - comment représenter des faits du monde réel ?
  - comment raisonner sur ces fait ?
  - quelles représentations sont appropriées pour traiter le monde réel ?



## La chasse au Wumpus



## Situation de départ



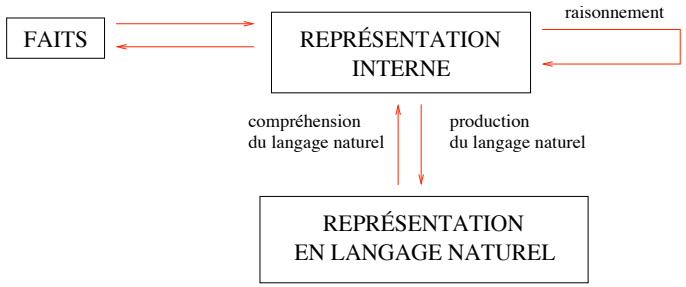



## Description PEAS du monde du Wumpus

- P (mesure de performance)
  - trouver l'or $+1000$
  - mort $-1000$
  - $-1$ par déplacement
  - $-10$ pour décocher la flèche
- E (environnement) *inconnu à l'avance !*
  - odeur désagréable dans les cases adjacentes au Wumpus
  - courant d'air dans les cases adjacentes aux puits
  - éclat dans la case contenant l'or
  - décocher la flèche tue le Wumpus si on lui fait face
- A (actions)
  - tourner à gauche, à droite, avancer
  - saisir le trésor, déposer le trésor
  - décocher la flèche
- S (sensors, capteurs)
  - courant d'air, éclat, odeur



## Agents à base de connaissances

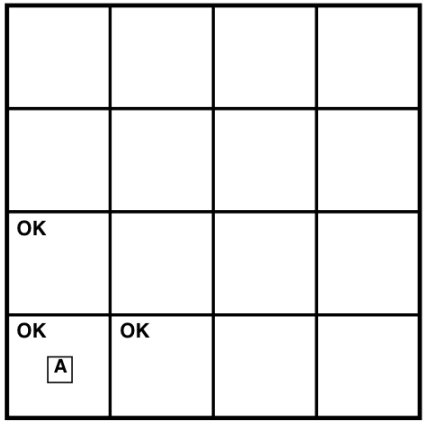



L'agent doit être capable de :

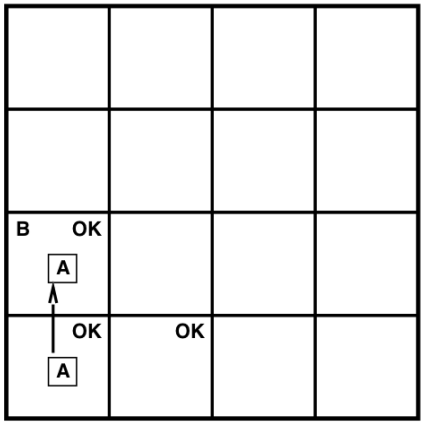
- Représenter des états, des actions, etc.
- Incorporer de nouvelles séquences perceptives
- Mettre à jour ses représentations internes du monde
- Déduire des propriétés cachées du monde
- Déduire les actions appropriées



## Explorer le monde du Wumpus

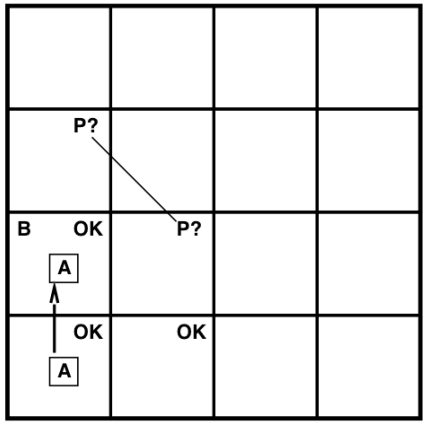


## Explorer le monde du Wumpus



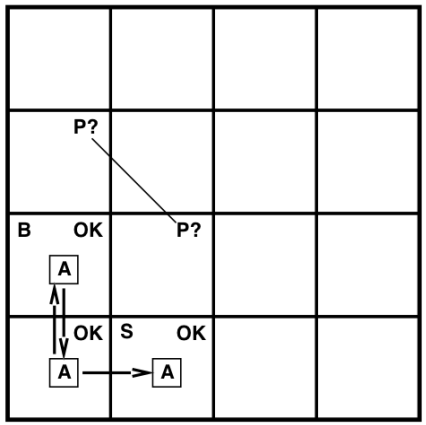



## Explorer le monde du Wumpus



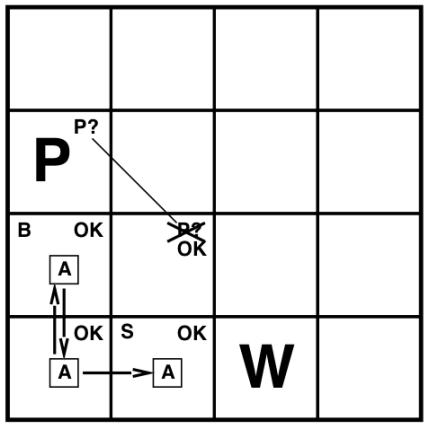



## Explorer le monde du Wumpus

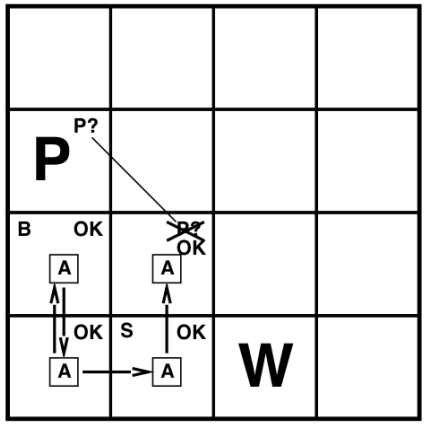





## Explorer le monde du Wumpus

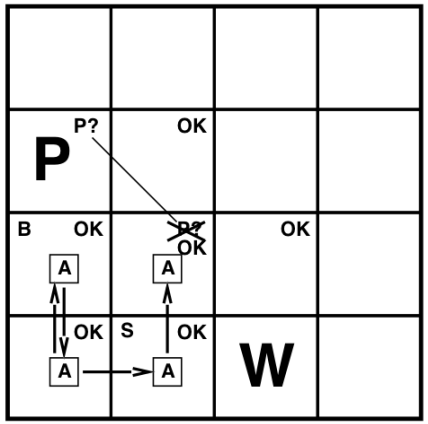


## Explorer le monde du Wumpus



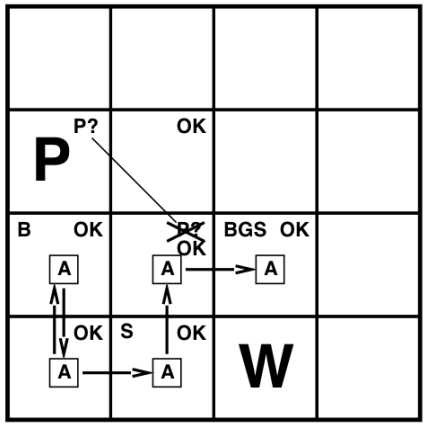



## Explorer le monde du Wumpus





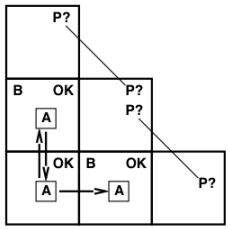

## Explorer le monde du Wumpus





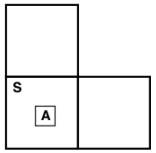

## Quelques situations délicates

- Courant d'air en (1,2) et en (2,1) ⇒ pas d'action sure




faire hypothèse que les puits sont uniformément distribués (2,2) a plus de probabilité d'avoir un puits que (1,2) et (2,1)

- Odeur en (1,1) ⇒ impossible de décider




utiliser la stratégie de force : décocher la flèche droit devant

  - si le Wumpus était devant, il sera mort ⇒ sécurisé
  - si le Wumpus n'y était pas ⇒ sécurisé

## Propriété

- Dans chaque cas
  - l'agent *déduit* une conclusion à partir des informations disponibles
  - la conclusion doit être *correcte* si les informations sont correctes
- La correction est une propriété fondamentale du raisonnement en logique



## Systèmes experts

- Un système expert utilise des connaissances spécifiques à un domaine pour fournir des conseils ou des solutions à des problèmes
- Des connaissances d'un domaine sont présentées dans une base de connaissances (par exemple base de règles)
- Simuler le raisonnement de l'expert humain : moteur d'inférence
- La performance d'un système expert dépend essentiellement de ses connaissances, moins du moteur d'inférence
- Un système expert doit être riche en connaissances : *Knowledge is power.*
- Comme un expert humain, un système expert
  - se spécialise dans un domaine
  - enrichit ses connaissances avec des expériences



## Domaines de systèmes experts

- Interprétation : former des conclusion de haut niveau à partir de données brutes
- Prédiction : trouver des conséquences probables des situations données
- Diagnostique : déterminer la cause du mal fonctionnement dans des situations complexes à partir des symptômes observables
- Configuration : construire une configuration des composants pour atteindre des objectifs de performance tout en satisfaisant des contraintes de configuration
- Planification : déterminer une suite d'actions pour arriver à un ensemble d'objectifs
- Surveillance : comparer des comportements d'un système par rapport au comportement souhaité
- Contrôle : contrôler le comportement d'un environnement complexe
- etc.



## Architecture des systèmes experts

- Base de connaissances :
  - les connaissances générales et spécifiques du domaine
  - connaissance sous forme de règle *Si ... alors ...* pour des systèmes à base de règles
  - des éditeurs à base de connaissances permet de représenter les connaissances sous une forme facile à accéder, modifier ou agrandir
- Moteur d'inférence :
  - effectuer le raisonnement pour tirer les conséquences impliquées par la connaissance incluse dans le système
- Interface d'utilisateur :
  - interface graphique, traitement de question/réponse (ex. en langage naturel), menus, etc.
  - interface pour une consultation du système expert
  - interface pour l'acquisition des connaissances, mettre à jour ou vérifier des connaissances

## Exigences pour un système expert

- Raisonnement correct
- Raisonnement ouvert à l'inspection
- Capacité d'explication des choix et des décisions pris



## Construction d'un système expert

- Des modules pour la création et la gestion d'un système expert peuvent être fournis ou réutilisés
  - CLIPS de la NASA en C,
  - JESS en Java,
  - ILOG rules de IBM,
  - ...
- Construction de la base de connaissances (acquisition, formulation, etc.) est plus importante et en général est plus difficile





## Aperçu

## Logique propositionnelle

- La logique des propositions permet d'exprimer
  - des faits sur le monde : "Jean aime Marie"
  - des négations : "Marie n'aime pas Jean"
  - des conjonctions et des disjonctions
  - des phrases avec "conséquence" logique : "Si Jean n'aime pas Marie, elle ne l'aime pas non plus"
- Une proposition est une expression (phrase) à propos du monde qui est soit vraie soit fausse
- Eléments de base :
  - symboles de propositions : $P, Q, \ldots$ (phrases)
  - phrases spéciales : *Vrai*, *Faux*
  - opérateurs : $\wedge$ (et), $\vee$ (ou), $\neg$ (non), $\rightarrow$ (implique), $\leftrightarrow$ (équivalent)



## Logique propositionnelle : Syntaxe

- Formules (phrases) :
  - les symboles de propositions $P_1, P_2, \ldots$ sont des formules
  - si $P$ et $Q$ sont des formules alors $\neq P$, $P \wedge Q$, $P \vee Q$, $P \rightarrow Q$ et $P \leftrightarrow Q$ sont des formules
- Exemples : Soit $C_{ij}$= "courant d'air dans la case $(i, j)$", $P$ = "un puits en case $(i, j)$". Comment représenter :
  - Si la case $(2, 3)$ a un puits alors il y a un courant d'air dans les cases adjacentes.
  - S'il y a du courant d'air dans la case $(1, 2)$ alors il doit y avoir un puits dans une cases adjacente.



## Logique propositionnelle : Sémantique

- Dans une interprétation, un symbole propositionnelle peut s'évaluer à une valeur vraie ou fausse
- Règles d'évaluation de formules :
  - $\neg P$ est vraie ssi $P$ est fausse
  - $P \wedge Q$ est vraie ssi $P$ est vraie et $Q$ est vraie
  - $P \vee Q$ est vraie ssi $P$ est vraie ou $Q$ est vraie
  - $P \rightarrow Q$ est vraie ssi $P$ est fausse ou $Q$ est vraie
    ou bien $P \rightarrow Q$ est fausse ssi $P$ est vraie et $Q$ est fausse
  - $P \leftrightarrow Q$ est vraie ssi $P$ et $Q$ sont tous deux vraies ou tous les deux fausses



## Base de connaissances du monde de Wumpus

- $KB$ = l'ensemble de toutes les phrases (en logique propositionnelle) décrivant la connaissance actuelle du monde
- Soit $P_{ij}$ vrai s'il y a un puit en $(i, j)$. Soit $B_{ij}$ vrai s'il y a du courant d'air en $(i, j)$.
- Pas de puits en (1,1)

$$R_1 : \neg P_{11}$$

- Un puits crée du courant d'air dans les cases adjacentes

$$R_2 : B_{11} \leftrightarrow (P_{12} \vee P_{21})$$
$$R_3 : B_{21} \leftrightarrow (P_{11} \vee P_{22} \vee P_{31})$$

- Perception : 2 cases visitées

$$R_4 : \neg B_{11}$$
$$R_5 : B_{21}$$

- Base $KB = \{R_1, R_2, R_3, R_4, R_5\}$

## Déductions en logique propositionnelle

- Résolution : connaissances sous forme de clauses
- Chaînages avant et arrière : connaissances sous forme de clauses de Horn (règles)



## Résolution

- Exemple : on sait que
  - il y a un puits en case $(1,3)$ ou en case $(2,2)$ :

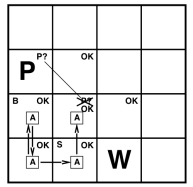
$$P_{1,3} \vee P_{2,2}$$

  - il n'y a pas de puits en case $(2,2)$ :

$$\neg P_{2,2}$$

Le littéral $\neg P_{2,2}$ est résolu avec le littéral $P_{2,2}$ pour donner $P_{1,3}$, on conclut qu'il y a un puits en case $(1,3)$.

$$\frac{P_{1,3} \vee P_{2,2}, \qquad \neg P_{2,2}}{P_{1,3}}$$






## Règle de résolution

- Littéral : variable ou sa négation
- Connaissances sous forme de clauses : disjonctions de littéraux

$$A \vee B, \ B \vee \neg C \vee \neg D$$

- Base de connaissances : conjonction de clauses (forme normale conjonctive - CNF)
- Règle de déduction : règle *résolution*

$$\frac{C_1 \vee A, \qquad C_2 \vee \neg A}{C_1 \vee C_2}$$

où $C_1, C_2$ sont des clauses



## Transformation en forme normale conjonctive

$B_{1,1} \leftrightarrow (P_{1,2} \vee P_{2,1})$

- Eliminer $\leftrightarrow$ : remplacer $\alpha \leftrightarrow \beta$ par $(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$.

$$(B_{1,1} \rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \rightarrow B_{1,1})$$

- Eliminer $\rightarrow$ : remplacer $\alpha \rightarrow \beta$ par $\neg\alpha \vee \beta$

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$$

- Faire entrer $\neg$ en utilisant la loi de de Morgan et double-négation

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$$

- Appliquer la règle de distribution ($\vee$ sur $\wedge$)

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$$

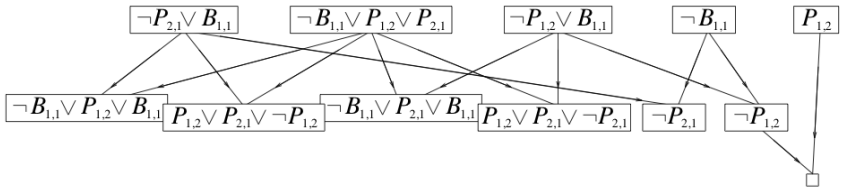
## Algorithme de résolution

- Preuve $KB \models \alpha$ par contradiction : montrer que $KB \wedge \neg\alpha$ est insatisfaisable
  - $KB \wedge \neg\alpha$ est transformé en forme normale conjonctive
  - La règle de déduction s'applique sur chaque pair de clauses qui contiennent des littéraux complémentaires, produit une nouvelle clause, qui est ajoutée dans la base si elle n'y est pas encore
  - Le processus continue jusqu'à ce qu'un des cas suivants se produise :
    - ⋆ plus aucune nouvelle clause soit ajoutée à $KB$, en ce cas $KB \not\models \alpha$
    - ⋆ la résolution de deux clauses produit la clause vide (*faux*), en ce cas $KB \models \alpha$
- La résolution est correcte et complète en logique propositionnelle



## Exemple de résolution

- Base de connaissances $KB$ :
  - si case $(1,1)$ a du courant d'air alors un case adjacente doit avoir un puits : $B_{1,1} \leftrightarrow (P_{1,2} \vee P_{2,1})$
  - pas de courant d'air dans $(1,1)$ : $\neg B_{1,1}$
- On veut prouver qu'il n'y a pas de puits en case $(1,2)$, $\alpha$ : $\neg P_{1,2}$
- $KB$ mis en forme normale conjonctive
  $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1})$, $(\neg P_{1,2} \vee B_{1,1})$, $(\neg P_{2,1} \vee B_{1,1})$, $\neg B_{1,1}$
- On ajoute $\neg\alpha$ dans la base $KB$ et applique la règle de résolution :




- La résolution sur $KB \wedge \neg\alpha$ produit la clause vide, donc $KB \models \neg P_{1,2}$



## Propriété de la résolution

- La résolution est correcte :
  - si la résolution sur $KB \wedge \neg\alpha$ produit la clause vide, alors $\alpha$ est une vraie conséquence (sémantique, signification) des connaissances de la base $KB$
- La résolution en raisonnement sur des propositions est complète :
  - si $\alpha$ est une conséquence logique quelconque des connaissances de la base $KB$, alors la résolution sur $KB \wedge \neg\alpha$ se termine et produit la clause vide