Hexapion

À l'instar d'un enfant qui vient de se brûler en mettant sa main dans le feu à qui l'on interdit de jouer avec le feu, on empêche l'IA de rejouer ce coup qui l'a faite perdre. Petit à petit, tous les coups perdants sont ôtés des possibilités de l'IA. Il ne lui restera que des coups gagnants. Le jeu est ainsi fait qu'en jouant parfaitement, le second joueur ne peut pas perdre.

Punitions et récompenses. L'IA apprend via un système de punitions. On demande à l'IA de jouer une partie, si elle perd, on la punit de façon à ce qu'elle ne reproduise plus l'erreur qu'elle a commise pour perdre. À force elle ne commet plus d'erreur.

Nous aurions également pu procéder par récompenses. Lorsque que l'IA gagne, on rajoute une perle de la couleur du coup gagnant dans la dernière boîte. Ainsi, l'IA a plus de chance de reproduire ce coup gagnant plus tard.

Notez que les perles des coups perdant restent présentes, il y a donc une petite chance pour que ces coups soient joués même après cent parties. On peut donc cumuler les deux approches : récompenser en rajoutant des perles gagnantes et punir en enlevant les perles perdantes. En faisant l'expérience, deux phénomènes devraient se produire : l'IA commence à gagner plus rapidement mais l'apprentissage est globalement plus long (l'IA met plus longtemps avant de devenir 100% imbattable).

Le procédé ainsi mit en valeur se nomme **l'apprentissage par renforcement** (plus connu avec l'anglais *reinforcement learning* ou *RL*).

Exploration et exploitation. Ceci est la manifestation d'un problème fondamental de l'apprentissage par renforcement, qui est **l'équilibre entre exploration et exploitation**. L'exploration représente la capacité de l'IA à tester tous les coups possibles. L'exploitation représente la capacité de l'IA à gagner. L'exploration est suffisante à l'exploitation : si l'on sait à l'avance si chaque coup gagne ou perd, alors il est facile de gagner. Cependant, l'exploration complète nécessite de passer du temps à perdre, pour éliminer les coups perdants. Si l'on souhaite gagner, on doit sacrifier de l'exploration.

Cela ne change pas beaucoup pour Hexapion, car il est rapide de tout explorer. Mais aux échecs, le problème est bien réel. On estime à 10^{120} le nombre de parties possibles (il faudrait donc jouer longtemps pour tout explorer !). Pour créer une intelligence artificielle jouant bien aux échecs, il faut lui faire explorer les possibles de façon intelligente. Par exemple, on peut décider d'arrêter d'explorer dès qu'une position est estimée trop perdante. On gagne du temps en n'explorant pas toutes les fins possibles, mais on passe peut-être à côté d'une configuration où nous aurions pu gagner. Pour les amateurs d'échecs, un bon exemple est le sacrifice : on perd du matériel (désavantage à court terme) pour trouver des compensations sur le long terme.

Définir cette politique d'exploration / exploitation est ainsi déterminant.

Les (désormais moins) récents succès de AlphaGo (pour le jeu de Go) et AlphaZéro (pour le jeu d'échecs) sont ainsi dus (dans une certaine mesure) à l'utilisation d'une technique très performante pour choisir quand explorer et quand exploiter, nommée la **recherche arborescente Monte Carlo**.