A propos des compétitions de programmes de jeu de Go

Tristan Cazenave
Laboratoire d'Intelligence Artificielle
Département Informatique, Université Paris 8,
2 rue de la Liberté, 93526 Saint Denis, France.

cazenave@ai.univ-paris8.fr

Le jeu de Go est un ancien jeu chinois. Il est très populaire en Corée, au Japon et en Chine. Les générations successives de joueurs de Go ont accumulé une expérience énorme du jeu tout au long de son histoire. Pour donner une idée de la somme des connaissances sur ce jeu, on peut citer les historiens chinois qui écrivent souvent que c'est Yao, un empereur semi-mythique du 23$^{\text{ème}}$ siècle avant J.-C. qui aurait inventé le jeu de Go pour instruire son fils Dan Zhu [Fairbain 1995]. Aujourd'hui on compte des dizaines de millions de joueurs de Go en extrême orient, et des centaines de joueurs professionnels. Le jeu de Go est un jeu à information parfaite, à somme nulle entre deux joueurs : Blanc et Noir. Un damier de Go (Goban) est une grille 19x19, qui est vide au début de la partie et que les joueurs remplissent tour à tour en posant une pierre sur une intersection. Le nombre moyen de coups possibles est de l'ordre de 250, et les parties peuvent facilement durer 200 coups. D'un point de vue programmation, les difficultés viennent à la fois du grand nombre de coups possibles, qui rend une approche combinatoire par force brute de type programme d'Echecs impossible. Mais elles viennent aussi de la complexité de l'évaluation d'une position, et de la définition ambiguë de la fin de la partie (en utilisant des règles non ambiguës, les parties dureraient plus de 350 coups). Les meilleurs programmes de Go utilisent des connaissances stratégiques et des recherches très sélectives orientées vers des sous-buts du jeu (capture de pierres, connexion, vie et mort des groupes…). Les programmes de Go sont encore très faibles par rapport aux meilleurs joueurs humains. Le meilleur programme de Go a un niveau qui est approximativement de 9 kyus coréens (les kyus ont des valeurs différentes suivant les pays, et le kyu coréen est le plus fort, les kyus français sont aussi assez forts). Les meilleurs joueurs de Go sont 9$^{\text{ème}}$ Dan et les débutants sont 20$^{\text{ème}}$ kyu. En théorie il y a donc une différence de 17 pierres entre les meilleurs joueurs et les meilleurs programmes : les programmes peuvent jouer 17 coups de suite au début de la partie et toujours se faire battre. En pratique, la différence est encore plus grande, et un des meilleurs programmes s'est récemment fait battre à 29 pierres par un 6$^{\text{ème}}$ Dan amateur.

Un des nombreux avantages de la programmation du jeu de Go d'un point de vue de l'Intelligence Artificielle est qu'on peut très facilement comparer deux programmes en les faisant jouer l'un contre l'autre. Et les programmeurs de Go ne s'en privent pas. De nombreux tournois de programmes sont régulièrement organisés. En 2001, mon programme Golois a participé à deux tournois internationaux de programmes : la coupe Garosu en Corée du Sud au mois de mars et la 21$^{\text{st}}$ Century Cup aux Etats-Unis au mois de juillet.

Le vainqueur de la 21$^{\text{st}}$ Century cup a été Go4++, de Michael Reiss (Angleterre) avec 5 victoires, qui a gagné de justesse contre le deuxième programme : Haruka, écrit par Ryuichi Kawa (Japon) qui avait aussi 5 victoires. Le troisième programme avec 4 victoires était Wulu, écrit entre autre par Xiuyu Lei (Chine) une joueuse professionnelle de Go, le quatrième avec 4 victoires aussi est The Many Faces of Go, de David Fotland (USA). Suivi de SmartGo avec 4 victoires, Anders Kierulf (USA), GoStar (4 victoires), de Jimmy Lu (Chine), Go Intellect (3 victoires), de Ken Chen (USA), GNU Go (3 victoires), de la Free Software Foundation (USA, Suède, France, Angleterre…), Golois (2 victoires), de Tristan Cazenave (France), Aya (2 victoires), de Hiroshi Yamashita (Japon), Explorer (1 victoire), de Martin Mueller (Canada) et enfin plodder (0), de William Newman (USA).

Comme d'habitude dans les compétitions de programmes de Go, les programmes commerciaux se sont partagé les premières places. Les programmes écrits par des universitaires qui essaient de partager leurs connaissances (Go Intellect, Gnugo, Golois et Explorer) ne sont pas les mieux placés et sont encore trop peu nombreux par rapport aux programmes commerciaux sur lesquels aucune information ne filtre (Go4++, Haruka, Wulu). Certains programmeurs qui ont des intérêts commerciaux sont toutefois plus ouverts et acceptent de parler de leurs programmes, même s'ils rentrent rarement dans les détails (David Fotland, Anders Kierulf). Le prix du meilleur nouveau programme a été attribué à Anders Kierulf pour SmartGo, ce qui peut être mis en perspective quand on sait qu'Anders Kierulf a passé sa thèse sur la programmation du Go en 1990, et qu'il est à l'origine du format informatique standard de notation des parties de Go (SGF). Il est vrai qu'il ne s'est remis que récemment à la programmation du Go à temps plein, après avoir passé 8 ans chez Microsoft à écrire PowerPoint… Les auteurs des trois meilleurs programmes travaillent à temps plein sur leur programme depuis de nombreuses années.

Les tournois de programmes de Go nous renseignent sur la force des programmes les uns par rapport aux autres. A ce titre ils sont utiles car ils permettent de départager des approches différentes. Malheureusement, le résultat d'un tournoi ne nous donne aucune information sur ce qui fait le succès d'une approche ou sur ce que contient réellement un programme dont son auteur ne veut pas révéler le contenu.

Les tournois sont utiles, mais ils donnent une information sur le niveau pas sur le contenu des programmes. Dans une optique de partage des informations élargie, j'ai proposé aux programmeurs de Go d'organiser une compétition de résolution de problèmes [Cazenave 2001], basée sur une librairie de problèmes de Go similaire à TPTP pour les mathématiques [Sutcliffe & Suttner 2001], à SAT pour la satisfaction [Hoos & Stützle 2000] ou à CSPLib pour les contraintes [Gent, Walsh & Selman 2001]. Il existe déjà des bases de problèmes de Go pour tous les niveaux, étant donné la vaste littérature extrême orientale sur le jeu. Je possède ainsi des milliers de problèmes de Go tirés de livres. Toutefois il est impossible de publier ces problèmes sur un site web ou de les donner à d'autres personnes qui n'ont pas acheté les livres qui correspondent parce qu'ils sont protégés par copyright. Ce que je propose c'est donc une librairie de problèmes de Go sur lesquels il n'y aurait pas de copyright.

Une compétition de résolution de problèmes peut être organisée pendant un des tournois de programmes déjà existant (par exemple la 21st century cup, son organisateur est d'accord). Mais organiser une telle compétition soulève de nombreuses questions pratiques. De même écrire une base de problèmes qui ne soit pas trop biaisée et qui rende compte des aptitudes effectives des programmes n'est pas chose aisée.

L'avantage d'une telle compétition d'un point de vue scientifique est que les programmes commerciaux peuvent y participer, permettant ainsi de connaître l'état de l'art pour la résolution de certains problèmes, sans révéler les algorithmes qu'ils utilisent. De plus, les algorithmes déjà connus peuvent aussi être comparés sur cette base. Actuellement, les articles scientifiques sur la programmation du Go font trop peu souvent référence à des tests sur des bases de problèmes reconnues comme pertinentes. On voit trop souvent des chercheurs prétendre que leur algorithme résout beaucoup de problèmes parce qu'il résout deux ou trois exemples qu'ils ont eux-mêmes choisi.

Le test ultime pour un programme de Go est bien sûr de jouer contre des joueurs humains. Pour cela, on peut connecter son programme à un des serveurs Internet de jeu de Go, qui comptent des centaines de joueurs connectés 24 heures sur 24. Ces serveurs ont un système de classement élaboré et très strict. Le niveau d'un programme sur un de ces serveurs est sans doute la meilleure évaluation possible d'un programme. Certains programmes y jouent déjà (Go4++, Many Faces of Go, Golois).

D'un autre coté le niveau global de jeu est intéressant pour les joueurs de Go, mais d'un point de vue résolution de problèmes, il est tout aussi intéressant d'évaluer le niveau d'un programme pour une composante du jeu bien précise. Le jeu de Go est assez complexe pour faire appel à différents types de résolution de problèmes. Certains programmes sont très bons pour certains types de problèmes, mais n'ont aucune connaissance d'autres problèmes. Or si on vise le meilleur niveau global pour un programme, il est préférable d'être moyen sur tous les types de problèmes que d'être très bons sur un ou deux types de problèmes. Des programmes qui sont les meilleurs pour certains types de problèmes comme les problèmes de vie et de mort, les problèmes de semeais, les problèmes de connexion ou de capture ne sont pas forcément les programmes qui ont les meilleurs résultats dans les tournois ou qui ont le meilleur niveau global. Il est pourtant intéressant d'évaluer et de comparer des algorithmes et des programmes pour tous les types de problèmes. Une compétition de résolution de problèmes permettrait d'avoir une vision plus fine des différentes facettes qui compose un programme de Go, voire de leurs importances respectives.

Actuellement, les compétitions de programmes sont dotées de prix conséquents, ce qui a pour effet de renforcer la compétition et de ralentir les velléités de coopération. J'avais conçu au départ une compétition de résolution de problèmes avec des prix conséquents pour attirer les gens qui vivent de la programmation du Go. Je pense maintenant au contraire qu'il ne faut surtout pas avoir de prix en argent pour cette compétition. A mon avis, le prix Ing de 1,6 millions de dollars, et le potentiel commercial des programmes de Go, ont plutôt agi comme des freins à la recherche que comme des accélérateurs. Ils ont incité les programmeurs à ne pas communiquer leurs découvertes, et tout le monde a réinventé la roue dans son coin sans le dire à son voisin. Pour vraiment aider la programmation du Go, il faudrait lancer des prix sur la publication en domaine public d'algorithmes ou de programmes qui permettent de résoudre un type de problèmes sur une base de problèmes standard et reconnue, plutôt que de donner des prix à des programmes fermés et secrets comme cela se fait actuellement.

En général l'algorithme que suit un programmeur de Go pour développer son programme est le suivant :

1. Faire jouer le programme contre un autre programme ou sur Internet,
2. trouver le plus mauvais coup du programme,
3. le corriger pour qu'il ne le joue plus,
4. Goto 1.

Or si l'on ne veut effectuer des cercles dans l'espace des programmes de Go, il faut garder une trace des erreurs que l'on a corrigées pour vérifier que les modifications ultérieures ne reproduisent pas les mauvais comportements. Les programmes de Go sont donc en général associés à une base de tests qui sont chargés de vérifier que le programme ne fait plus les erreurs qui ont été corrigées. Ces préoccupations sont partagées par de nombreux chercheurs qui cherchent à augmenter la performance de leurs algorithmes [Gent & al 1997]. De façon assez étonnante, M. Reiss l'auteur de Go4++ soutient qu'il n'utilise pas de tels tests de régression, et qu'il vérifie la pertinence des ses modifications en faisant jouer son programme. Toutefois d'une façon générale, les programmeurs passent beaucoup de temps à analyser les parties de leur programme et à enrichir leur base de tests de positions intéressantes qui illustrent des points difficiles dans la résolution de certains problèmes. Partager cette information qui est difficile à obtenir par soi-même est un des avantages d'une base de tests publique.

Un problème pratique qui se pose est l'adoption d'un format standard pour les tests, de façons à pouvoir les échanger et à pouvoir facilement tester tous les programmes sur les mêmes problèmes. Le Go Text Protocol [Gnugo 2001] semble être tout indiqué pour cela. J'ai converti ma base de test pour ce format, et j'espère qu'il sera utilisé par un nombre croissant de programmes de Go. J'ai aussi écrit un programme qui permet de connecter un programme de Go connaissant le Go Text Protocol de se connecter aux serveurs Internet de jeu de Go. J'espère ainsi inciter les programmeurs de Go à l'utiliser.

Ensuite viennent les questions sur la comparaison des différents algorithmes [Kaindl & Kainz 1999]. Doit on fixer une limite globale de temps pour la résolution de tous les problèmes ou une limite pour chaque problème, une pénalité graduelle pour le temps de résolution ? Tous les problèmes ont-ils la même importance ? Quelles sont les catégories possibles de problèmes ? Certaines des réponses peuvent être trouvées grâce aux autres compétitions du même type telle que TPTP, pour laquelle G. Sutcliffe montre des histogrammes pour comparer les programmes. Il classe tous les problèmes résolus par un programme par temps croissant et dessine la courbe du temps de résolution cumulé en fonction du nombre de problèmes. Ce pourrait une méthode de comparaison intéressante et instructive pour les programmes de Go, et les algorithmes associés. Il est très facile de modifier un programme de Go pour qu'il réponde parfaitement à une série de problèmes qui sont déjà connues. Pour faire vivre une telle compétition, il serait probablement nécessaire de créer de nouveaux problèmes à chaque compétition. C'est aussi pourquoi il est probablement mauvais de donner des prix pour cette compétition, elle doit plutôt être envisagée comme un outil public d'évaluation d'algorithmes, et d'aide au déboguage que comme un tournoi.

Le bon coté des compétitions actuelles est qu'elles sont amusantes et très motivantes. Elles font travailler dur sur le programme les 2 ou 3 semaines qui précèdent pour les moins sérieux, et toute l'année qui précède pour les gens qui en vivent…En 2001, il y a eu 6 compétitions internationales de programmes de jeu de Go, toutes basées sur le même principe (la coupe Garosu à Séoul, le Congrès Européen à Dublin, la coupe du 21ème siècle a York PA, Les Olympiades des ordinateurs à Maastricht qui n'ont finalement pas eu assez de candidats vu le foisonnement actuel, une compétition en Chine, une au Japon, et un tournoi de programmes sur 9x9 organisé par S. Mertin). Il serait plus approprié d'avoir des compétitions qui permettent plus d'échanges entre les participants. La coupe du 21ème siècle et les Olympiades des ordinateurs semblent aller dans ce sens puisqu'elles sont associées à des workshops.

Bibliographie

[Cazenave 2001] Cazenave T.: A Problem Library for Computer Go. IJCAI-01 Workshop on Empirical methods in Artificial Intelligence, Seattle, 2001 .

[Fairbain 1995] Fairbain J.: Go in ancient China. London, 1995.

[Gent & al. 1997] Gent I.P., Grant S. A., MacIntyre E., Prosser P., Shaw P., Smith B. M., Walsh T.: How Not To Do It. Report 97.27, University of Leeds.

[Gent, Walsh & Selman 2001] Gent I.P., Walsh T., Selman B.: CSPLib: a problem library for constraints. http://www-users.cs.york.ac.uk/~tw/csplib/

[GnuGo 2001] http://www.gnu.org/software/gnugo/

[Hoos & Stützle 2000] Hoos H. H. , Stützle T.: SATLIB - The Satisfiability Library. http://www.satlib.org/.

[Kaindl & Kainz 1999] Kaindl H., Kainz G.: Guidelines for the Experimental Comparison of Search Algorithms. IJCAI-99 Workshop on Empirical AI, Stockholm, 1999.

[Sutcliffe & Suttner 2001] Sutcliffe G., Suttner C.: Evaluating general purpose automated theorem proving systems. Artificial Intelligence 131 (1-2), pp. 39-54.