Das Schreiben "intelligenter" Spiele
von Zafer Barutcuoglu, 2002

Überarbeitung und PDF-Konvertierung 5.2003 von Thomas Antoni
www.qbasic.de

Vorwort

Jahrzehnte sind vergangen seit der Ersterscheinung von Schach-Computern bis zu den listigen Strategien von Warcraft II, und es sieht so aus, als ob die künstliche Intelligenz auch in der Zukunft genau so gefragt sein wird. Besonders durch den jüngsten Schach-Wettkampf zwischen Kasparov und IBMs Deep Blue hat dieses Thema erst wieder weltweite Popularität genossen.

Künstliche Intelligenz (abgekürzt KI oder auf englisch AI für Artificial Intelligence) ist eines der ältesten und grössten Forschungsgebiete in der Computer-Wissenschaft. Daher ist es mir nicht möglich, hier ein Lies-das-und-Du-weißt-alles-Tutorial zu bieten. Nun, wie nur wenige Hausfrauen die eine Waschmaschine benutzen, auch wirklich wissen wie sie funktioniert, brauchen auch wir nicht zu tief in unser Thema eintauchen; nur tief genug um es in unseren Spielen zu anzuwenden.

Was ist Intelligenz? Amüsant ist nur, dass sich diese einzige Frage eine ganze Menge der wissenschaftlichen Texte über AI zum Thema gemacht haben. In unserem Fall würde die Antwort lauten: AI ist ein Computergegner, der den Spieler wirklich herausfordert. Aber lassen Sie uns nicht in diese Kontroverse gelangen und unsere Zeit der hier ausschließlich interessierenden Frage widmen, nämlich: Wie können wir AI softwaretechnisch realisieren? Auch dies muss wieder zum aktuellen Fall berücksichtigt werden, und sogar noch spezieller zu dem zu programmierenden Spiel. Es gibt unzählige Varianten, geschaffen für die verschiedensten Aufgaben. Ein Schach-Algorithmus ist keineswegs gleich dem eines Fussballspiels, oder eines "GO", einem anderen Spiel. Dafür sind die Regeln und Problemstellungen zu unterschiedlich. Da ich hier nicht AI für jedes einzelne Spiel beschreiben kann, werde ich versuchen die am häufigsten angewandten Hauptschritte zu beschreiben.

Der einfachste Level

Wenn Sie jemals ein Arcade Spiel mit Computergegnern geschrieben haben, dann wissen Sie, dass diese auf dem Was-ist-wenn-Prinzip basieren. Dies ist dann der einfachste Level der AI. Doch es ist ja noch nicht mal klar, ob dies tatsächlich AI ist. Aber wie ich schon erwähnte, werde ich das hier nicht weiter diskutieren. Denken Sie immer nur daran, dass ein großer Teil der AI- Systeme nur auf solch einem Grundprinzip basiert, auf einem Was-ist-wenn-Frage-Antwort-Spiel. Jedoch für schwerwiegendere Probleme braucht man ingend-etwas besseres.

Spielbäume und Schach

Schach, wie kompliziert es als Strategiespiel zu sein scheint, ist eines der am einfachsten zu analysierenden Spiele. Das kommt daher, dass es ein Brettspiel ist, endlich viele mögliche Positionen (Alles in Allem 64), und auch endlich viele Spielsteine (32) hat. Auf allen Positionen hat man als Spieler nur begrenzte und wirklich nicht viele Möglichkeiten, sich zu bewegen. Auf dieser Grundlage können wir sagen, dass auf der ganzen Welt nur eine endliche Anzahl von Schach-Sequenzen existiert.

Überall auf dem Schachbrett können wir dann alle möglichen Züge auflisten. Darüber hinaus können wir nach jedem dieser Züge bestimmen, wie das Spielbrett aussehen wird. Auf diesem Wege können wir einen so genannten Spielbaum konstruieren. Dessen Wurzel ist die Ausgangsposition, die Äste und Verzweigungspunkte sind Zwischen-Spielstände und dessen Blätter Spielenden bzw. Spielausgänge.

Wenn wir diesen Baum einmal am Anfang konstruieren, gewinnen wir praktisch das Spiel EGAL WAS der Gegner macht. Wirklich verlieren wir nur, wenn wir schwarz spielen und gegen jemanden spielen der ebenfalls den klompletten Spielbaum kennt. Praktisch gesehen ist dieser Baum jedoch sehr groß, sehr sehr sehr groß. Es ist praktisch unmöglich einen vollständigen Spielbaum zu erstellen, auch nicht, wenn man die weltbesten Computer der heutigen Zeit benutzt.

Bevor wir den ersten Schritt gehen, konstruieren wir den Spielbaum, der am aktuellen Fall mit einer örtlich festgelegten Tiefe verwurzelt wird. So sind die Blätter nicht mehr notwendigerweise Spielenden. Jetzt müssen wir uns entscheiden, in welcher Richtung wir die einzelnen Blätter entwickeln wollen. Das heißt, welches Blatt ist das vorteilhaftere für uns? Für dieses Problem konstruieren wir eine zählende Funktion, die eine Brettkombination als Input nimmt, und setzen eine numerische Marke, die anzeigt, welche Seite in einer besseren Position ist und wieviel besser diese Position ist.

Diese Funktion hängt gewöhnlich von der Zahl und Art der Spielfiguren ab, die auf jeder Seite übrig geblieben sind, und ebenso von der Mobilität (Zahl der möglichen Bewegungen), der Rochier-Möglichkeiten und anderen Faktoren. Dies wird Ihrer Erfahrung des Schachspielens überlassen. Wenn wir diese Funktion haben, zählen wir die Blattpositionen. Dann, annehmend, daß unser Konkurrent seinen besten Zug machen wird und daß wir unseren besten machen werden, rücken wir den Baum vor. Dieses wird der Minimal_Maximal-Algorithmus genannt, denn wenn der Konkurrent am Zug ist, werden wir das Minimale nehmen, da er versucht unsere Punktzahl zu minimieren, und wenn wir am Zug sind, werden wir das Maximale nehmen. Jeder Knotenpunkt nimmt das Minimum oder Maximum seiner Kinder, und wenn wir die Stufe unterhalb der Wurzel erreichen, haben wir eine Liste von möglichen Zügen, bei denen jeder Zug eine bestimmte Punkteanzahl hat, die von den Blättern kommt.

Was wir dann nur noch machen müssen, ist, den Zug mit der höchsten Punktanzahl zu ermitteln und durchzuführen. So einfach ist das.

Die sogenannte Tiefenschicht, zu der wir den Baum konstruieren, ist sehr wichtig, da Zeit und Raum exponential ansteigen. Vier Schichten werden als Minimum akzeptiert, aber ich konnte in meinem ersten Schachprogramm kaum bis 3 gehen. Es existieren gewisse Methoden, die die Tiefe bedingt erhöhen, wie zum Beispiel eine Sicherung am Blatt.

Dieses, auch wenn sehr einfach und ursprünglich, ist die allgemeine Grundlage für ein Schachprogramm, auch wenn es einfach zu gewinnen ist, kann man ein Schachprogramm bekommen, das Anfänger herausfordern wird. Ich tat es. Es waren ungefähr 600 Pascal-Zeilen und brachte mir mehr als 600 Seiten von KI bei, weil diese Methode, der Spielbaum, keineswegs auf Schach eingeschränkt wird. In der Tat bildet sie das Skelett von vielen KI-Maschinen, die es so gibt. Und dennoch tun sie sich einen Gefallen!!! Schreiben Sie ein Schachprogramm!!!

Heuristik

Beobachten Sie, daß wir die Suchtiefe verringerten, um die Suche besser anwendbar zu machen. Dieses ist ein Beispiel der häufig angetroffenen Kompromisse in der Technik. In der Tat ist das der Trick bei der Sache, weil jeder weiß, wie man die Arbeit erledigt, aber nur der Schnellste gewinnt. Heuristik ist der Name für das Bilden der klugen Kompromisse. Stellen Sie sich einen heuristischen Reiseführer vor, der sie zu den besten Plätzen in der kürzesten Zeit bringt. Ein anderes Beispiel wäre ein Auto beim Parken. Der Fahrer parkt sein Auto auf den ersten Platz, den er findet, wenn er nahe genug zu seinem Zielpunkt ist, obwohl der andere Plätze viel näher am Ziel sein könnten. Ähnlich können Sie Ihren Spielbaum von wirklich dummen Bewegungen befreien bevor Sie sich in der ganzen Tiefe ausbreiten; und obgleich diese Bewegung eine äußerst gescheite Falle sein könnte, sind die Chancen sehr groß, dass es keine ist, und Ihr Programm läuft wesentlich schneller. Dieses wird "Alpha-Beta-Beschneidung" oder abgekürzt "Beschneidung" genannt. Was auch immer, beachten Sie Ihre Heuristik.

Fortgeschrittene Methoden: Planung

Wenn Ihr Problem für einen groben Spielbaum zu schwierig ist, dann stehen Sie vor einer schwer zu überwindenen Hürde. Wenn Sie zu diesem Punkt kommen, dann fehlt Ihnen entweder Rechenzeit oder Speicherplatz oder. OK, selbstverständlich sind Rechenzeit und Speicherplatz für den Computer getrennte Kategorien, aber wir wissen, dass uns das nicht viel hilft, wenn wir ein Spielbrett von 640x480 Pixeln haben, anstelle der 8x8 Felder eines Schachbretts.

Ein anschauliches Beispiel ist das japanische Spiel Go. Es wird auf einem Brett mit 19x19 Spielfeldern gespielt, und das mit identischen Spielsteinen. Und Sie können NICHT mehr als 2 oder 3 Spielzüge voraussehen, weil Sie nämlich nach 3 Spielzügen 391*391*391 = 59 776 471 mögliche Spielkombinationen haben. Zusätzlich hängt das Resultat infolge der Natur des Spiels stark von den Anfangszügen ab, und normalerweise braucht man zwischen 200 und 350 Züge, um ein Spiel zu beenden. Bei diesen Dimensionen kommen Sie wahrscheinlich zu dem Dchluss, daß es unmöglich ist, für dieses Spiel auf einem Mikrocomputer einen Computergegner zu programmieren. Glücklicherweise ist dies nicht so.

Bei solch schwierigen Problemen können wir nicht mehr auf niedrigen Stufen wie Spielbäumen denken. Wir verwenden hochentwickelte KI-Techniken, wie in diesem Fall Planung. Planung ist der Prozeß des Teilens einer Aufgabe in kleinere Aufgaben, d.h. dass sie entsprechend ihrer Natur und Abhängigkeiten geordnet und in dieser Reihenfolge ausgeführt werden. In unserem Spiel Go heisst das vielleicht, dass Sie entscheiden ob man z.B. die Ecken in Angriff nimmt oder die Mitte beherr-schen möchte. Ich nehme an, daß Sie das Spiel nicht kennen, aber beachten Sie den Unterschied zwischen dieser Entscheidung und der, ob man den Ritter oder den Bishop bewegt. Wenn Sie ein wenig darüber nachdenken, werden Sie bemerken, daß Entscheidungen im Tatsächlichen Leben wie

das Befehlen einer Armee oder Beschüsse über Bankbeteiligungen häufig von ähnlicher Natur sind. Das ist auch die unbewusst angewendete Methode mit der Sie ihre eigenen Spiele spielen.

Wenn Sie sich hinsetzen, mit dem Programmieren beginnen und diese Methoden einführen, werden Sie entdecken wie Sie intelligentere Methoden erhalten und Ihr Code in zunehmendem Maße umfangreich und schwieriger zu handhaben wird. Außerdem sind Spiel-Bäume zum Beispiel leistungsfähiger als Planung beim Schachspielen gewesen.

Dieses ist nicht sehr überraschend, wenn Sie sich daran erinnern, daß Sie intelligenter als Ihr Computer sind, aber der Computer dafür schneller rechnen kann. So verwenden Sie einfachere Methoden, wenn Sie können.

Codieren

Obgleich es noch viel mehr über KI gibt, worüber ich sprechen möchte, beschränke ich mich auf das Programmieren von Spielen, und ich nehme an, dass alles sollte euch Programmierer zum Weitermachen bewegen. Das Codieren von etwas Intelligentem wird Ihnen ein ungeahntes Vergnügen bereiten. Sei es Tic-Tac-Toe oder Warcraft XXI, versuchen Sie es. Lesen Sieweitere Tutorials und Bücher über KI, lesen Sie den Code von Beispielprgrammen und codieren Sie. Diese drei Schritte zahlen sich immer aus.