LUA-Bot Coding

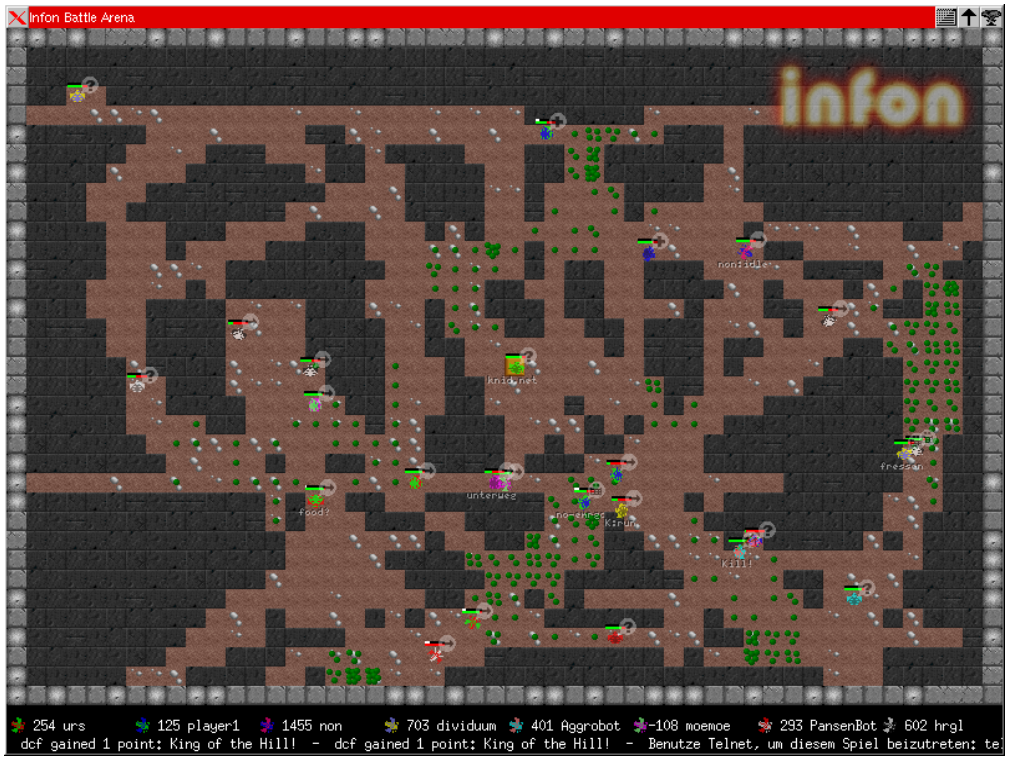
Schwarmtheorie in der Praxis

Fabian Haupt

Fachschaft Mathematik / Informatik
Institut für Mathematik und Informatik
Universität Würzburg

24. Oktober 2006

Eine Karte



Eine Standardkarte mit ein paar arbeitenden Bots.

Verschieden Feldertypen

- Es gibt nur ein paar wenige verschiedene Feldertypen:
  -  Futterfeld
  -  Normales Feld
  -  Nicht begehbares Feld

Verschieden Feldertypen

- Es gibt nur ein paar wenige verschiedene Feldertypen:
-  Futterfeld
-  Normales Feld
-  Nicht begehbares Feld

Verschieden Feldertypen

- Es gibt nur ein paar wenige verschiedene Feldertypen:
-  Futterfeld
-  Normales Feld
-  Nicht begehbares Feld

Verschieden Feldertypen

- Es gibt nur ein paar wenige verschiedene Feldertypen:
-  Futterfeld
-  Normales Feld
-  Nicht begehbares Feld

Kreature

Kreaturen

Am Anfang startet jeder Spieler mit zwei Worker-Kreaturen. Ziel ist es sich mit dieser Anfangspopulation gegen die anderen Völker durchzusetzen. Der Staat der als erstes eine bestimmte Punktezahl erreicht, gewinnt die Runde.

Die 2 Kreaturentypen im Detail

Die Typen von Kreaturen unterscheiden sich in ein paar wesentlichen Punkten.

Kreaturentypen

- ist wehrlos
- kann sich etwas schneller bewegen als ein Breeder
- kann sich in einen Breeder verwandeln (wenn genug Futter gesammelt wurde)

Breeder

Breeder

- kann sich verteidigen und auch angreifen
- kann bei genügend Futter neue Worker spawnen
- ist etwas langsamer beim Laufen und Futter sammeln

Die Regeln

- nur Breeder können angreifen
- Worker können zu Breedern mutieren
- Breeder können Worker spawnen
- essen, mutieren, spawnen und Gegner-Kreaturen töten bringt Punkte
- King-of-the-Hill(koth) Feld besetzt halten bringt Punkte
- das Volk das als erstes ein Punktelimit erreicht gewinnt

Die Regeln

- nur Breeder können angreifen
- Worker können zu Breedern mutieren
- Breeder können Worker spawnen
- essen, mutieren, spawnen und Gegner-Kreaturen töten bringt Punkte
- King-of-the-Hill(koth) Feld besetzt halten bringt Punkte
- das Volk das als erstes ein Punktelimit erreicht gewinnt

Die Regeln

- nur Breeder können angreifen
- Worker können zu Breedern mutieren
- Breeder können Worker spawnen
- essen, mutieren, spawnen und Gegner-Kreaturen töten bringt Punkte
- King-of-the-Hill(koth) Feld besetzt halten bringt Punkte
- das Volk das als erstes ein Punktelimit erreicht gewinnt

Die Regeln

- nur Breeder können angreifen
- Worker können zu Breedern mutieren
- Breeder können Worker spawnen
- essen, mutieren, spawnen und Gegner-Kreaturen töten bringt Punkte
- King-of-the-Hill(koth) Feld besetzt halten bringt Punkte
- das Volk das als erstes ein Punktelimit erreicht gewinnt

Die Regeln

- nur Breeder können angreifen
- Worker können zu Breedern mutieren
- Breeder können Worker spawnen
- essen, mutieren, spawnen und Gegner-Kreaturen töten bringt Punkte
- King-of-the-Hill(koth) Feld besetzt halten bringt Punkte
- das Volk das als erstes ein Punktelimit erreicht gewinnt

Die Regeln

- nur Breeder können angreifen
- Worker können zu Breedern mutieren
- Breeder können Worker spawnen
- essen, mutieren, spawnen und Gegner-Kreaturen töten bringt Punkte
- King-of-the-Hill(koth) Feld besetzt halten bringt Punkte
- das Volk das als erstes ein Punktelimit erreicht gewinnt

Das Futter

Futter

- Das Futter wächst an zufälligen Stellen auf der Karte.
- Die Orte sind während der Runden fest.
- Die Futterstellen haben unterschiedliche Radien.
- Das Futter wächst nacht.

Das Futter

Futter

- Das Futter wächst an zufälligen Stellen auf der Karte.
- Die Orte sind während der Runden fest.
- Die Futterstellen haben unterschiedliche Radien.
- Das Futter wächst nacht.

Das Futter

Futter

- Das Futter wächst an zufälligen Stellen auf der Karte.
- Die Orte sind während der Runden fest.
- Die Futterstellen haben unterschiedliche Radien.
- Das Futter wächst nacht.

Das Futter

Futter

- Das Futter wächst an zufälligen Stellen auf der Karte.
- Die Orte sind während der Runden fest.
- Die Futterstellen haben unterschiedliche Radien.
- Das Futter wächst nacht.

Welche Probleme stellen sich?

- effiziente Suche nach Futter?
- Wann mutieren bzw neue Kreaturen spawnen?
- angreifen bzw verteidigen oder weglaufen?
- Schwarmverhalten? Jede Kreatur einzeln laufen lassen oder zusammenbleiben?
- koth Feld ignorieren?

Welche Probleme stellen sich?

- effiziente Suche nach Futter?
- Wann mutieren bzw neue Kreaturen spawnen?
- angreifen bzw verteidigen oder weglaufen?
- Schwarmverhalten? Jede Kreatur einzeln laufen lassen oder zusammenbleiben?
- koth Feld ignorieren?

Welche Probleme stellen sich?

- effiziente Suche nach Futter?
- Wann mutieren bzw neue Kreaturen spawnen?
- angreifen bzw verteidigen oder weglaufen?
- Schwarmverhalten? Jede Kreatur einzeln laufen lassen oder zusammenbleiben?
- koth Feld ignorieren?

Welche Probleme stellen sich?

- effiziente Suche nach Futter?
- Wann mutieren bzw neue Kreaturen spawnen?
- angreifen bzw verteidigen oder weglaufen?
- Schwarmverhalten? Jede Kreatur einzeln laufen lassen oder zusammenbleiben?
- koth Feld ignorieren?

Welche Probleme stellen sich?

- effiziente Suche nach Futter?
- Wann mutieren bzw neue Kreaturen spawnen?
- angreifen bzw verteidigen oder weglaufen?
- Schwarmverhalten? Jede Kreatur einzeln laufen lassen oder zusammenbleiben?
- koth Feld ignorieren?

Strategie anfänglich

wenige Kreaturen

Wenn wir nur wenige Kreaturen haben bietet sich ein gewichteter Randomwalk an. Wann man allerdings dann die Worker mutiert, bzw neue Worker erstellt kann man z.B. von der Anzahl der Gegner abhängig machen.

Strategie im weiteren Verlauf

größerer Staat

Bei vielen Kreaturen können wir bereits Schwarmalgorithemen verwenden. Man kann z.B. Pheromonspuren implementieren, die dann recht schnell ein gutes Ergebniss liefern (siehe http://www.ameisenalgorithmus.de/).
Alternativ kann man auch systematisch alle Felder absuchen, ist aber weniger effektiv (aber einfacher :)) .

Arbeiterkreatur

Worker

Da Worker überhaupt keine Verteidungsmöglichkeit besitzen, bietet sich an, ihre höhere Geschwindigkeit auszunutzen und wegzulaufen. Ausbaufähig durch Abhauen z.B. zu einer größeren Ansammlung von Breedern.

Königinkreatur

Breeder

Entscheiden ab welchem Abstand Gegner angegriffen werden sollen.
Verfolgen von gegenerischen Workern?
Eigenen angegriffenen Kreaturen in der Nähe helfen?

Möglichkeiten das gesamte Schwarmverhalten zu steuern:

- gar nicht beachten, jede Kreatur für sich selber arbeiten lassen
- straffes Einhalten einer Formation
- an ein paar Nachbarn orientieren, und versuchen nicht zu weit wegzukommen
- am kompletten Schwarm orientieren

Strategien Fazit:

KISS - Keep it simple, stupid! funktioniert sehr gut gegen viele bots, hat allerdings gegen eine ordentlich durchdachte und implementierte Schwarmstrategie keine Chance.

LUA-Intro

LUA

Es folgt nun eine kurze Einführung in die Skripsprache Lua.
Lua findet unter anderem auch in Spielen wie Enemy-Territory, Farcry, Crisis, etc Verwendung.
Dementsprechend viel Dokumentation gibt es auch in Google.

Variablentypen

Typen in LUA

- nil
- boolean
- number
- string
- userdata (um auf C Funktionen zuzugreifen)
- function (ähnlich einem Funktionspointer in C)
- thread (threading kann lua native)
- table

Verwendung von Variablen

Variablen müssen nicht deklariert werden. Aber jede Zuweisung kann den Typ einer Variablen ändern.

```
> a = 3.14159265
> print(a,type(a))
3.14159265      number
> a = true
> print(a,type(a))
true    boolean
>
```

table-type

Vielleicht ein etwas besonderer Typ in LUA ist die *table*. Vergleichbar mit einer *map* aus der *STL*, mit dem Unterschied, dass die Tabelle unterschiedliche Typen fassen kann.

```
> t={}
> t[5] = 42
> t["text"] = true
> print(t[5]) --> 42
42
> print(t["text"])
true
> print(t[100])
nil
>
```

Gültigkeitsbereiche von Variablen

Es gibt in LUA nur *globale* und *lokale* Variablen. Es gibt also nichts das vergleichbar zu private oder anderen scopes wäre. Globale Variablen im BotCode können von allen Kreaturen aus verändert und ausgelesen werden. Lokale Variablen in Funktionen (*locale i = 0*) sind nur innerhalb des Blocks verwendbar.

Operatoren

Es gibt in Lua die üblichen Operatoren:

```
+,-,*,/,^
<,>,<=,>=,==,~=
and, or
..
```

Operatorprioritäten

Die Prioritäten der Operatoren sind wie folgt:

```
^
not
* /
+ -
..
< > <= >= ~= ==
and
or
```

if...then...else

```
if a < b then
return a
elseif a==b then
return b
else
  return b+a
end
```

while, repeat

```
c = 10
while c~=0 do
c = c - 1
print(c)
end

repeat
  print("Hallo")
until false
```

for-Schleife

```
for i=a,b,c do
   print(i)
end

a = {10,5,1,3}
for i,v in ipairs(a) do
   print(i,v)
end
```

Funktionsdeklaration

Eine allgemeine Funktionsdeklaration:

```
function functionname (paramter1, parameter2)
    ...
    return value -- optional
end
```

und ein Beispiel:

```
function addvectors (a, b)
    result = {0,0,0}
    for i=0,3 do
        result[i] = a[i] + b[i]
    end
    return result
end
```

Verwenden von netcat

netcat aufruf

nc dresden 2323

```
[...]
------------------------------------------------------------
                  Woche der Informatik-Version
     Fachschaft Mathematik / Informatik - Fabian Haupt
------------------------------------------------------------
enter '?' for help
>
```

Die GUI

Server vs GUI

Grundsätzlich ist bei der Simulation Server und Client getrennt. Wer eine lokale Version der Beamer-Ausgabe haben möchte muss den Client aufrufen.

Aufruf

```
cd /HOME/scratch/infon
./infon dresden:2323
```

Eigener Code

upload

Der eigene BotCode kann grundsätzlich mit Copy&Paste in den Server geladen werde. Nach dem Joinen einfach 'b' drücken und den Code hineinpasten. Die Eingabe mit '.' beenden. Fertig.
Für häufigeres Uploaden bietet sich die Vereinfachung des Vorganges mit Skripten an (Makefile,ruby,bash...), siehe dazu auch *http://erde/infon*

Library

Es gibt eine umfangreiche Library an vorgefertigten High-Level-Funktionen für die Kreaturen.
Eine Übersicht gibt es auf *http://erde/infon*.

Credits

Thanks to

- Florian 'dividuum' Wesch - Original Author
- my friends from #geeks - help with some latex-beamer woes
- Fachschaft Mathematik / Informatik - for all their work
- noneira - correcting many typos

go, go, go

Let the coding begin!!