Computational Intelligence: Übungsblatt 1

Frank Mehne, János Sebestyén

Aufgabe 1.2: Tic Tac Toe: Konzept

Als Kodierung für eine *Spielsituation* verwenden wir eine 3x3-Matrix $S = (s_{i,j}) \epsilon \Sigma = \{0, 1, 2\}^3$. Dabei gibt $s_{i,j}$ die Belegung des Feldes in der i-ten Zeile und j-ten Spalte an, wobei 0 für ein leeres Feld, 1 für das Symbol des ersten Spielers (Kreis) und 2 für das Symbol des zweiten Spielers (Kreuz) steht. Beachtet man, dass immer zuerst der erste Spieler und dann der zweite Spieler abwechselnd einen Zug machen, ist die Menge $\boldsymbol{S}$ aller gültigen Spielsituationen gegeben durch

$$\boldsymbol{S} = \{S \in \Sigma :\ 0 \leq \ \#\{(i,j): s_{i,j} = 1\} - \#\{(i,j): s_{i,j} = 2\} \leq 1\},$$

wobei hier der Einfachheit halber auch solche Situationen mitgezählt werden, die eigentlich bereits für einen Spieler gewonnen sind. Die Mächtigkeit von $\boldsymbol{S}$ ist *#**S**=6.046*. Das heißt, es gibt 6.046 gültige Spielsituationen.

Auf $\boldsymbol{S}$ definieren wir die Relation $\sim$ in der folgenden Weise:

Zwei gültige Spielsituationen $S, T \in \boldsymbol{S}$ heißen äquivalent (geschrieben S~T) genau dann, wenn sich S durch eine der folgenden Operationen in T überführen lässt:

T=o_0(S)=S

T=o_1(S)=Rotation um 90°

T=o_2(S)=Rotation um 180°

T=o_3(S)=Rotation um 270°

T=o_4(S)=Rotation um 90° und Spiegelung an der x-Achse

T=o_5(S)=Rotation um 90° und Spiegelung an der y-Achse

T=o_6(S)=Rotation um 180° und Spiegelung an der x-Achse

T=o_7(S)=Rotation um 180° und Spiegelung an der y-Achse

Man kann leicht nachprüfen, dass $\sim$ eine Äquivalenzrelation ist (Reflexivität, Symmetrie, Transitivität). Damit wird eine Menge $\boldsymbol{S}/\sim$ von Äquivalenzklassen definiert, wobei jede gültige Spielsituation $S \in \boldsymbol{S}$ eindeutig ihrer Äquivalenzklasse [S] zugeordnet werden kann.

Durch folgende Vorschrift (Pseudocode) lässt sich eine Menge $\boldsymbol{R}$ von Repräsentanten für die Äquivalenzklassen und gleichzeitig die Zuordnung jeder Spielsituation zur ihr entsprechenden Repräsentanten *R(S)* konstruieren:

```
R_0= ()

i=0

for all S in S

      i++

      for all T in R_{i-1}

            if o_1(S)=T or o_1(S)=T or … or o_8(S)=T

                  R_i = R_{i-1}

                  R(S)=T

            else

                  R_i = R_{i-1}∪{S}

                  R(S)=S

            end if

      next T

next S

R=R_i
```

Ein *Tic Tac Toe Spieler P* ist durch jeweils eine deterministische Reaktion auf jede mögliche (gültige) Spielsituation gegeben.

Wie unter http://mathrec.org/old/2002jan/solutions.html nachzulesen ist, gibt es abhängig von der Nummer des Zuges folgende Anzahlen der möglichen (bis auf Äquivalenz verschiedenen) Situationen:

Zug	1	2	3	4	5	6	7	8	9
# Situationen =:s	1	3	12	38	108	174	204	153	57

Die Anzahl der möglichen Züge ist durch die Anzahl der noch freien Felder gegeben:

Zug	1	2	3	4	5	6	7	8	9
# mögliche Züge =:z	9	8	7	6	5	4	3	2	1

Dadurch bestehen je Zug die folgenden Anzahlen möglicher Reaktionsmuster:

Zug	1	2	3	4	5	6	7	8	9
# Reaktionsmuster = z^s	$9^1=9$	8^3	7^{12}	6^{38}	5^{108}	4^{174}	3^{204}	2^{153}	$1^{57}=1$

Der letzte Zug kann vernachlässigt werden, da er stets durch die Situation eindeutig bestimmt ist.

Das von uns gewählte Genom besitzt 8 Gene, wobei das i-te Gen das Reaktionsmuster des Spielers im i-ten Zug repräsentiert. Ein Reaktionsmuster beschreibt die Reaktion auf jede mögliche Situation in diesem Zug.

Durch die oben angegebenen Anzahlen möglicher Reaktionsmuster ergeben sich je Gen unterschiedliche benötigte Bitlängen:

Zug	1	2	3	4	5	6	7	8	**Summe**
z^s log 2	3,2	9,0	33,7	98,2	250,8	348,0	323,3	153,0	1219,1
benötigte Bitlänge	4	9	34	99	251	348	324	153	**1222**

Die gesamte Genomlänge ließe sich noch geringfügig verringern, wenn die Reaktionsmuster Zug-übergreifend als eine einzige Binärzahl codiert werden würden (wie an der Differenz der in der Tabelle angegebenen Summen erkennbar ist). Jedoch bestünde das Genom dann nur aus einem einzigen Gen, was die Möglichkeit der Rekombination von guten Teillösungen für einzelne Züge ausschließen würde.

Als Mutationsoperator wird ein zufälliger Bitflip für jedes Bit mit geringer Wahrscheinlichkeit gewählt. Als Rekombinationsoperator wählen wir Single-Point-Crossover, das nur an den Grenzen zwischen je zwei Genen möglich sein soll, um Rekombinationen von guten Teillösungen zu ermöglichen. Als Überlebenskriterium legen wir ein Altern von einer Generation fest. Die Eltern-Selektion erfolgt proportional zur Fitness (siehe unten).

Es wird sichergestellt, dass der Mutationsoperator nur Spieler erzeugt, die immer gültige Züge machen, indem jedes mögliche Genom als korrekter Spieler interpretiert wird. Dazu werden die folgenden Schritte durchgeführt:

- Decodiere die gegebene Binärzahl des Gens in eine Dezimalzahl *d*
- Ist die Anzahl der möglichen Reaktionsmuster = k, so reduziere d auf seinen Divisionsrest *d*'=*d* modulo *k*
- Bilde *d*' auf das entsprechende Reaktionsmuster ab, welches eine Zuordnung von jeder möglichen Spielsituation im betrachteten Zug auf einen konkreten Spielzug darstellt. Dies geschieht durch fortgesetzte ganzzahlige Division mit Rest:
    - Bezeichnet s wieder die Anzahl der möglichen Situationen und z die Anzahl der möglichen Züge je Situation, so ergibt
        - d' / z^s die Reaktion auf Situation 1,
        - d' / z^{s-1} die Reaktion auf Situation 2,
        - …
        - d' / z^1 die Reaktion auf Situation s-1, und
        - d' modulo z die Reaktion auf Situation s