Das Schweizer System

Gruppenaufgabe Tag 1

Das Schweizer System ist ein Verfahren, um den Gewinner eines Turniers in einer Sportart zu bestimmen, bei der immer zwei Spieler gegeneinander antreten. Es ist das Standardverfahren für Schachturniere, wird aber z.B. auch bei Tischtennis- oder Tischfußballturnieren angewendet. Die Grundidee des Verfahrens ist, dass der Turniersieg in Runden ausgespielt wird. In jeder Runde gibt es einen neuen Gegner für jeden Spieler, und es spielen immer diejenigen Spieler gegeneinander, die (möglichst) die gleiche Punktzahl (bzw. die gleiche Anzahl an Siegen) erreicht haben.

Beim Schach erhält ein Spieler für einen Sieg einen Punkt, für ein Remis (Unentschieden) einen halben Punkt und für eine Niederlage keinen Punkt. Beim Schweizer System werden harte Regeln von weichen Regeln unterschieden. Harte Regeln müssen unbedingt eingehalten werden; weiche Regeln sollten möglichst eingehalten werden, aber müssen oder können eventuell sogar gar nicht eingehalten werden.

Harten Regeln des Schweizer Systems für Schach sind:

1. Niemand spielt gegen sich selbst.
2. Jeder Spieler spielt in jeder Runde gegen genau einen Gegner.
3. Kein Spieler spielt zweimal gegen den gleichen Spieler.
4. Niemand spielt mehr als zweimal aufeinander folgend mit den gleichen Steinen (schwarz oder weiß).

Weiche Regeln des Schweizer Systems für Schach sind:

1. Kein Spieler sollte die gleiche Farbe zweimal aufeinander folgend spielen.
2. Die Anzahl der Spieler, die gegen einen Gegner mit unterschiedlicher Punktzahl spielen, sollte minimiert werden.
3. Niemand sollte zweimal hintereinander gegen einen Gegner mit unterschiedlicher Punktzahl spielen.

Dabei ist die Erfüllung von 1. wichtiger als die Erfüllung von 2. und die wiederum wichtiger als die Erfüllung von 3.

Aufgabe 1: Ergänzungen zum System

Zu Beginn eines Turniers haben alle Spieler die gleiche Punktzahl. Unter der Voraussetzung, dass es keine Remis gibt: Wie viele Runden braucht man mindestens, bis es einen eindeutigen Gewinner, also einen Spieler mit einer eindeutigen maximalen Punktzahl gibt?

Die obigen Regeln haben noch ein Problem. Was tun Sie als Designer des Schweizer Systems, wenn die Gesamtzahl der Spieler ungerade ist? Erfordert Ihre Lösung neue harte und/oder weiche Regeln?

Exkurs: Aussagenlogik

Diese Aufgabe besteht darin eine vereinfachte Variante des Schweizer Systems für Schach durch eine Formel in Aussagenlogik zu modellieren. Die Regeln des Systems und der bisherige Turnierverlauf werden in einer aussagenlogischen Formel kodiert, so dass die gültigen Belegungen für die Formel die Paarungen der nächsten Turnierrunde beschreiben.

Aussagenlogische Formeln werden über eine Menge $\Sigma = \{P, Q, R, \ldots\}$ von aussagenlogischen Variablen gebildet, die entweder den Wahrheitswert 1 für „wahr“ oder 0 für „falsch“ annehmen können. Eine *Belegung* ist dann eine (partielle) Funktion $\mathscr{A} : \Sigma \mapsto \{0,1\}$. Eine Variable P ist wahr unter $\mathscr{A}$, falls $\mathscr{A}(P) = 1$, und falsch, falls $\mathscr{A}(P) = 0$. Aus den aussagenlogischen Variablen lassen sich dann mit Hilfe der Operatoren $\vee$ „oder“, $\neg$ „nicht“, $\wedge$ „und“, und $\rightarrow$ “impliziert”, komplizierte Formeln ϕ, ψ bilden. Deren Bedeutung läßt sich rekursiv über die Bedeutung ihrer Argumente (das sind dann Teilformeln) definieren

ϕ	ψ	$\phi \vee \psi$	$\neg\phi$	$\phi \wedge \psi$	$\phi \rightarrow \psi$
0	0	0	1	0	1
0	1	1	1	0	1
1	0	1	0	0	0
1	1	1	0	1	1

und $\mathscr{A}$ wird dann entsprechend obiger Tabelle auf Formeln erweitert. Eine Formel ϕ heißt dann *erfüllbar*, wenn es eine Belegung $\mathscr{A}$ gibt mit $\mathscr{A}(\phi) = 1$. Zum Beispiel ist die Formel $(P \vee Q) \rightarrow P$ erfüllbar für $\mathscr{A}_1(P) = \mathscr{A}_1(Q) = 0$. Die Belegung $\mathscr{A}_2(P) = 0, \mathscr{A}_2(Q) = 1$ erfüllt die Formel nicht: $\mathscr{A}_2((P \vee Q) \rightarrow P) = 0$.

Jede aussagenlogische Formel lässt sich in eine Menge von Teilformeln umwandeln, die per Konjunktion (also mit $\wedge$/und) verbunden sind. Im Folgenden betrachten wir nur noch implizit mit $\wedge$/und verbundene Mengen aussagenlogischer Formeln.

Das Problem der aussagenlogischen Erfüllbarkeit ist NP-vollständig, also schwierig. Aktuelle Algorithmen dafür haben (sehr) abstrakt folgende Struktur und im schlechtesten Fall immer eine exponentielle Laufzeit:

SAT(N)
Eingabe: Menge (Konjunktion) aussagenlogischer Formeln N
Ausgabe: 0 wenn es keine erfüllende Belegung gibt, sonst eine erfüllende Belegung
starte mit der leeren Belegung;
while (noch nicht alle Variablen aus den Formeln in N belegt) {
    if (Propagierung möglich) **then**
        propagiere vollständig;
    else
        rate die Belegung einer weiteren Variablen;
    if (Konflikt) **then** {
        if (keine geratene Variable in der Belegung) **then**
            return (0);
        nimm alle Belegungen inklusive der zuletzt geratenen Variable zurück;
        propagiere die inverse Belegung der zuletzt geratenen Variable;
        }
    }
return (erfüllende Belegung);

Propagierung ist die Ermittlung der Belegung von Variablen in polynomialer Zeit. Propagierte Variablen müssen auf einen bestimmten Wert gesetzt werden, damit eine Formel in der aktuellen Belegung erfüllbar bleibt. Zum Beispiel muss bei der Formel $P \wedge (Q \vee R)$ die Variable P auf 1 belegt werden. Oder falls die Variable P in $P \rightarrow Q$ schon auf 1 belegt ist, dann muss auch Q auf 1 belegt werden. Beide Situationen lassen sich sehr schnell, in linearer Zeit feststellen. Typischerweise sucht eine Propagierungsfunktion nach Formeln, bei denen alle bis auf eine Variable belegt sind und deren Wahrheitswert erst durch die Belegung dieser Variable festgelegt wird.

Ein Konflikt ist eine Situation, bei der die aktuelle Belegung eine Formel in N falsch macht. Auch das lässt sich sehr schnell berechnen. Zum Beispiel kann man für $N = \{P, \neg Q, \neg P \vee Q\}$ die Variablen P und Q beide propagieren, P wahr und Q falsch, aber dann wird die Formel $\neg P \vee Q$ falsch, und da wir keine Variablenbelegung geraten haben, würde der Algorithmus 0 zurückgeben; die Formel ist nicht erfüllbar.

Für $N = \{P \vee Q, \neg P \vee \neg Q, P \vee \neg Q\}$ lässt sich nichts propagieren. Der Algorithmus könnte dann z.B. Q als wahr raten. Mit der zweiten Formel ließe sich dann P als falsch propagieren und die dritte Formel würde dann einen Konflikt erzeugen. Da nur die Belegung von Q geraten wurde, nimmt der Algorithmus die Belegungen von P und Q zurück und propagiert dann Q als falsch. Dann würde jetzt P mit der ersten Formal als wahr propagiert, und der Algorithmus terminiert mit der erfüllenden Belegung P wahr und Q falsch.

Aufgabe 2: Modellierung der harten Regeln

Gegeben n Spieler $1, \ldots, n$, n gerade, eine Menge bereits gespielter Paarungen P_l bis einschließlich Runde l, wobei jedes Tripel $(i, j, k) \in P_l$ bedeutet, dass in Runde i der Spieler j mit den weißen Steinen gegen den Spieler k gespielt hat, und eine Funktion $p_l \colon \{1, \ldots, n\} \mapsto \mathbb{N} \cup \{0\}$, die jedem Spieler seinen aktuellen Punktestand nach Ausspielung von Runde l zuweist.

Gegeben P_l, p_l, $l > 2$, beschreiben Sie möglichst präzise eine Modellierung der harten Regeln für die nächste Runde $l + 1$ durch eine Menge von Formeln in Aussagenlogik.

Aufgabe 3: Minimale Belegungen

Manchmal ist man nicht nur an einer erfüllenden Belegung interessiert, sondern an einer *minimalen* erfüllenden Belegung. Dazu nimmt man typischerweise eine Gewichtsfunktion $\mu \colon \Sigma \mapsto \mathbb{N} \cup \{0\}$ an, die jeder Variable ein positives Gewicht oder 0 zuordnet. Man erhält dann das Gewicht einer Belegung, in dem man die Gewichte der wahren, also mit 1 belegten Variablen aufsummiert. Eine erfüllende Belegung ist minimal, wenn es keine andere erfüllende Belegung mit geringerem Gewicht gibt.

Erweitern Sie obigen Algorithmus **SAT** so, dass er nicht nur erfüllende, sondern minimale erfüllende Belegungen bezüglich einer positiven Gewichtsfunktion μ berechnet.

Aufgabe 4: Modellierung der weichen Regeln

Gegeben ein Algorithmus, der minimale erfüllende Belegungen berechnet. Wie können die weichen Regeln in Aussagenlogik und mittels einer von Ihnen gewählten Funktion μ modelliert werden?

Aufgabe 5:

Der Algorithmus **SAT** “lernt” nichts, er sucht nur systematisch nach einer erfüllenden Belegung. Welche Formeln könnten bei Identifikation eines Konflikts in obigem Algorithmus zu N hinzugefügt werden, ohne das Ergebnis des Algorithmus zu ändern? Was könnte bei der Version des Algorithmus, der die minimalen erfüllenden Belegungen berechnet, zusätzlich gelernt werden? Wie könnten die gelernten Formeln bei der Suche ausgenutzt werden, insbesondere wenn nicht notwendigerweise die zuletzt geratene Variable zurückgenommen wird?