Algorithmic Problem Solving

Roland Backhouse

January 29, 2004

Outline

Goal Introduce principles of algorithm construction

Vehicle Fun problems (games, puzzles)

Chocolate-bar Problem

How many cuts are needed to cut a chocolate bar into all its individual pieces?

Assignment and Invariants

Let p be the number of pieces, and c be the number of cuts.

The process of cutting the bar is modelled by:

$$p\,,c \;:=\; p{+}1\,,c{+}1 \quad .$$

We observe that $(p{-}c)$ is an invariant. That is,

$$(p{-}c)[p\,,c \;:=\; p{+}1\,,c{+}1] \;=\; (p{+}1)-(c{+}1) \;=\; p{-}c$$

Initially, $p{-}c$ is 1. So, number of cuts is always one less than the number of pieces.

Hoare Triples

Eg. Jealous couples:

- Three couples Aa, Bb and Cc.
- One boat which can carry at most two people.
- Wives (a, b and c) may not be with a man (A, B and C) unless their husband is present.

Construct a sequence of actions S_0 satisfying

$$\{\, AaBbCc| \,\} \quad S_0 \quad \{\, |AaBbCc \,\} \ .$$

Problem Decomposition

- Exploit symmetry!

Decomposition into

$\{ \; AaBbCc | \; \}$

S_1

; $\{ \; ABC | abc \; \}$

S_2

; $\{ \; abc | ABC \; \}$

S_3

$\{ \; | AaBbCc \; \}$

(Impartial, Two-Person) Games

- Assume number of positions is finite.
- Assume game is guaranteed to terminate no matter how the players choose their moves.
- Game is lost when a player cannot move.

- A position is *losing* if *every* move is to a winning position.
- A position is *winning* if *there is* a move to a losing position.

Winning strategy is to maintain the invariant that one's opponent is always left in a losing position.

Winning Strategy

Maintain the invariant that one's opponent is always left in a losing position.

{ losing position, and not an end position }

make an arbitrary (legal) move

; { winning position, i.e. not a losing position }

apply winning strategy

{ losing position }

Example Winning Strategy

One pile of matches.

Move: remove one or two matches.

Winning strategy is to maintain the invariant that one's opponent is always left in a position where the number of matches is a multiple of 3.

$\{$ n is a multiple of 3, and $n \neq 0$ $\}$

$\text{if } 1 \leq n \rightarrow n := n-1 \ \square \ 2 \leq n \rightarrow n := n-2 \text{ fi}$

; $\{$ n is not a multiple of 3 $\}$

$n := n - (n \bmod 3)$

$\{$ n is a multiple of 3 $\}$

Sum Games

Given two games, each with its own rules for making a move, the *sum* of the games is the game described as follows.

For clarity, we call the two games the *left* and the *right* game.

A position in the sum game is the combination of a position in the left game, and a position in the right game.

A move in the sum game is a move in one of the games.

Sum Games (cont)

Define two functions L and R, say, on left and right positions, respectively, in such a way that a position (l,r) is a losing position exactly when $L.l = R.r$.

How do we specify the functions L and R?

Sum Games (Cont)

First: L and R have equal values on end positions.

Second:

$\{\ \mathsf{L}.\mathfrak{l} = \mathsf{R}.\mathfrak{r}\ \wedge\ (\mathfrak{l}$ is not an end position $\vee\ \mathfrak{r}$ is not an end position$)\ \}$

if $\mathfrak{l}$ is not an end position $\rightarrow$ change $\mathfrak{l}$

$\square$ $\mathfrak{r}$ is not an end position $\rightarrow$ change $\mathfrak{r}$

fi

$\{\ \mathsf{L}.\mathfrak{l} \neq \mathsf{R}.\mathfrak{r}\ \}$

Third,

$\{\ \mathsf{L}.\mathfrak{l} \neq \mathsf{R}.\mathfrak{r}\ \}$

apply winning strategy

$\{\ \mathsf{L}.\mathfrak{l} = \mathsf{R}.\mathfrak{r}\ \}$

Sum Games (cont)

Satisfying the first two requirements:

- For end positions l and r of the respective games, $L.l = 0 = R.r$.
- For every l' such that there is a move from l to l' in the left game, $L.l \neq L.l'$. Similarly, for every r' such that there is a move from r to r' in the right game, $R.r \neq R.r'$.

Winning strategy (third requirement):

$\{\ L.l \neq R.r\ \}$

if $L.l < R.r \rightarrow$ change r

$\square$ $R.r < L.l \rightarrow$ change l

fi

$\{\ L.l = R.r\ \}$.

Winning strategy (third requirement):

$$\{\ \mathsf{L.l} \neq \mathsf{R.r}\ \}$$

$$\textsf{if}\ \ \mathsf{L.l} < \mathsf{R.r} \ \rightarrow\ \text{change } \mathsf{r}$$

$$\Box\ \ \mathsf{R.r} < \mathsf{L.l} \ \rightarrow\ \text{change } \mathsf{l}$$

$$\textsf{fi}$$

$$\{\ \mathsf{L.l} = \mathsf{R.r}\ \}\ .$$

- For any number m less than $\mathsf{R.r}$, it is possible to move from r to a position r' such that $\mathsf{R.r}'=\mathsf{m}$. (Similarly, for left game.)

Summary of Requirements

Satisfying the first two requirements:

- For end positions l and r of the respective games, $L.l = 0 = R.r$.
- For every l' such that there is a move from l to l' in the left game, $L.l \neq L.l'$. Similarly, for every r' such that there is a move from r to r' in the right game, $R.r \neq R.r'$.
- For any number m less than $R.r$, it is possible to move from r to a position r' such that $R.r' = m$. (Similarly, for left game.)

MEX Function

Let p be a position in a game G. The mex value of p, denoted $mex_G.p$, is defined to be the smallest natural number, n, such that

- There is no legal move in the game G from p to a position q satisfying $mex_G.q = n$.
- For every natural number m less than n, there is a legal move in the game G from p to a position q satisfying $mex_G.q = m$.

Characterising Features

- Non-mathematical, easily explained problems (requiring mathematical solution)
- Minimal notation.
- Challenging problems.
- Simultaneous introduction of programming constructs and principles of program construction.