Acro-Nim

Bart Massey
`bart@cs.pdx.edu`

May 1, 2002

1 The Game Of Acro-Nim

Acro-Nim (from the Latin *acro*, heights, hence “High Nim”) is a Nim variant with some interesting rules.

The basic game is 1-3-5-7 Nim, with the winner taking the last stone. To review the Nim rules: Nim is a game played with piles of stones. Players alternate in taking one or more stones from a single pile and discarding them. The player that takes the last stone from the board wins. 1-3-5-7 Nim is the variant where there are four initial piles, of size 1, 3, 5 and 7.

Acro-Nim has four specific rules that alter the basic Nim game.

Pass Rule: Anytime a player takes three or more stones from a pile, the player receives a *pass token*. On any turn, a player with a pass token may turn in a pass token rather than taking any stones. If both players pass in succession, the game is terminated and scored a draw.

Poison Stone: One of the stones in the initial pile of seven stones is the *poison stone*. Any player taking the poison stone, without taking at least one other stone from the pile, immediately loses the game (even if the poison stone is the last available stone).

Equalizer Rule: Once per game, a player may on their turn choose to *equalize* the piles rather than taking any stones. To equalize the piles, the player arranges the piles as evenly as possible. If the poison stone is present, and the arrangement is uneven, the poison stone should be part of one of the larger piles.

Split Rule: Once per game, a player may on their turn choose to *split* a pile of two or more stones rather than taking any stones. To split a pile, it is split into two piles, with the stones divided as evenly as possible. A pile containing the poison stone may not be split.

2 A Z Specification Of Acro-Nim

The Z formal specification notation is described elsewhere. In this section, it will be used to formalize the rules of Acro-Nim.

The state of a game of Acro-Nim consists of the state of the board and of two players. We will call these players North and South, and let South move first.

The Acro-Nim board consists of piles of stones. We will model these piles using schema. There are no empty piles: a pile ceases to exist when its last stone is removed. Our board design urges to keep track of whether a pile contains poison stones or not as part of the pile state. While the rules given above mention only one poison stone, it is convenient to generalize the specification to handle other initial positions.

$$
\begin{array}{|l}
\textit{Pile} \\
\hline
\textit{stones} : \mathbb{N}_1 \\
\textit{poisoned} : \mathbb{N} \\
\hline
\textit{poisoned} \leq \textit{stones} \\
\hline
\end{array}
$$

The board is a set of piles.

$$
\begin{array}{|l}
\textit{Board} \\
\hline
\textit{piles} : \mathbb{P}\ \textit{Pile} \\
\hline
\end{array}
$$

The initial board is the 1-3-5-7 board with the 7 pile containing one poisoned stone, described above. A helper function is useful to generate the piles.

$$
\begin{array}{|l}
\textit{make\_pile} : \mathbb{N}_1 \times \mathbb{N}_1 \to \textit{Pile} \\
\hline
\forall\, n, np : \mathbb{N}_1;\ p : \textit{Pile} \bullet \\
\qquad \textit{make\_pile}(n, np) = p \Leftrightarrow \\
\qquad\qquad p.\textit{stones} = n \land p.\textit{poisoned} = np
\end{array}
$$

$$
\begin{array}{|l}
\textit{InitBoard} \\
\hline
\textit{Board} \\
\hline
\textit{piles} = \{\textit{make\_pile}(1, 0), \textit{make\_pile}(3, 0), \\
\qquad \textit{make\_pile}(5, 0), \textit{make\_pile}(7, 1)\} \\
\hline
\end{array}
$$

The player state consists of the set of resources available to the player. While the game as described above only allows a single equalizer or split, it is convenient to generalize the specification to allow other initial states and/or rules.

$$
\begin{array}{|l}
\textit{Player} \\
\hline
\textit{equalizes}, \textit{splits}, \textit{passes} : \mathbb{N} \\
\hline
\end{array}
$$

The initial player state is as described.

$$
\begin{array}{|l}
\textit{InitPlayer} \\
\hline
\textit{Player} \\
\hline
\textit{equalizes} = 1 \\
\textit{splits} = 1 \\
\textit{passes} = 0 \\
\hline
\end{array}
$$

It will be necessary to have a record of move types, and of the state of play. It is also necessary to name the players.

$$
\begin{aligned}
&MOVE ::= move\_take\langle\langle Pile \times \mathbb{N}_1 \times \mathbb{N}\rangle\rangle \mid \\
&\quad move\_pass \mid move\_split\langle\langle Pile\rangle\rangle \mid \\
&\quad move\_equalize \mid move\_none \\
&PLAYER ::= north \mid south \\
&OPPONENT == \{north \mapsto south, south \mapsto north\} \\
&PLAY ::= game\_continues \mid game\_won\langle\langle PLAYER\rangle\rangle \mid game\_drawn
\end{aligned}
$$

The game state consists of the board state, the state of each player, a record of the last move type (for the draw rule), and an indication of the state of play.

GameState

$$
\begin{array}{|l}
Board \\
player : PLAYER \rightarrow Player \\
to\_move : PLAYER \\
last\_move : MOVE \\
play : PLAY \\
\hline
\end{array}
$$

The initial state is as described above.

InitGameState

$$
\begin{array}{|l}
GameState \\
\hline
InitBoard \\
\operatorname{ran} player = InitPlayer \\
to\_move = south \\
last\_move = move\_none \\
play = game\_continues \\
\hline
\end{array}
$$

The moves in Acro-Nim are all transformations on the game state. The player on move alternates.

Move

$$
\begin{array}{|l}
\Delta GameState \\
\hline
play = game\_continues \\
to\_move' = OPPONENT\ to\_move \\
\hline
\end{array}
$$

There are four distinct types of move.

To conveniently annotate the various types of moves, it is useful to formalize the notion of a player state adjustment.

$$
\begin{array}{|l}
adjust\_player : Player \rightarrow (\mathbb{N} \times \mathbb{N} \times \mathbb{N}) \rightarrow Player \\
\hline
\forall\, p, p' : Player;\ ne, ns, np : \mathbb{N} \bullet \\
\quad p' = adjust\_player\ p\ (ne, ns, np) \Leftrightarrow \\
\qquad (p'.equalizes = p.equalizes + ne\ \wedge \\
\qquad p'.splits = p.splits + ns\ \wedge \\
\qquad p'.passes = p.passes + np)
\end{array}
$$

A *take* move takes a selected number of (poisoned and/or non-poisoned) stones from a selected pile according to the rules of Nim. A pile disappears when its last stone is taken. The poison stone has to be handled carefully.

$$
\begin{array}{|l}
\textit{MoveTake} \\
\hline
\textit{Move} \\
p? : \textit{Pile} \\
n?, np? : \mathbb{N} \\
\hline
p?.\textit{stones} \geq n? \geq 1 \\
p?.\textit{poisoned} \geq np? \geq n? - (p?.\textit{stones} - p?.\textit{poisoned}) \\
\textit{player}'\ \textit{to\_move}' = \textit{player}\ \textit{to\_move}' \\
\textit{player}'\ \textit{to\_move} = \mathbf{if}\ n? \geq 3 \\
\quad \mathbf{then}\ \textit{adjust\_player}\ (\textit{player}\ \textit{to\_move})\ (0,0,1) \\
\quad \mathbf{else}\ \textit{player}\ \textit{to\_move} \\
\exists\, \textit{rest} : \mathbb{P}\, \textit{Pile} \bullet \\
\quad \textit{piles} = \textit{rest} \cup \{p?\} \land \\
\quad \textit{piles}' = \textit{rest} \cup \\
\qquad (\mathbf{if}\ p?.\textit{stones} = n? \\
\qquad\quad \mathbf{then}\ \varnothing \\
\qquad\quad \mathbf{else}\ \{\textit{make\_pile}(p?.\textit{stones} - n?, p?.\textit{poisoned} - np?)\}) \\
\textit{play}' = \mathbf{if}\ n? = np? \\
\quad \mathbf{then}\ \textit{game\_won}\ \textit{to\_move}' \\
\quad \mathbf{else\ if}\ \textit{piles}' = \varnothing \\
\qquad \mathbf{then}\ \textit{game\_won}\ \textit{to\_move} \\
\qquad \mathbf{else}\ \textit{game\_continues} \\
\textit{last\_move}' = \textit{move\_take}\ (p?, n?, np?) \\
\hline
\end{array}
$$

If the player has a pass token, they may pass. If the previous move was a pass, then the game is drawn. Otherwise, play continues in the same board state.

$$
\begin{array}{|l}
\textit{MovePass} \\
\hline
\textit{Move} \\
\hline
\textit{piles}' = \textit{piles} \\
\textit{player}'\ \textit{to\_move}' = \textit{player}\ \textit{to\_move}' \\
\textit{player}'\ \textit{to\_move} = \textit{adjust\_player}\ (\textit{player}\ \textit{to\_move})\ (0,0,-1) \\
\textit{play}' = \mathbf{if}\ \textit{last\_move} = \textit{move\_pass} \\
\quad \mathbf{then}\ \textit{game\_drawn} \\
\qquad \mathbf{else}\ \textit{game\_continues} \\
\textit{last\_move}' = \textit{move\_pass} \\
\hline
\end{array}
$$

A sensible way to handle split and equalizer moves is to construct a function that levels a set of piles. This function proceeds in two phases. First, the set of piles is combined into a single big pile containing all the stones.

$$
\begin{array}{|l}
\mathit{pile\_up} : \mathbb{P}\,\mathit{Pile} \to \mathit{Pile} \\
\hline
\forall\, p : \mathit{Pile} \bullet \\
\quad \mathit{pile\_up}\,\{p\} = p \\
\forall\, p, p', q : \mathit{Pile};\ \mathit{ps} : \mathbb{P}\,\mathit{Pile} \bullet \\
\quad p' = \mathit{pile\_up}\,(\{p\} \cup \mathit{ps}) \Leftrightarrow \\
\qquad (q = \mathit{pile\_up}\,\mathit{ps} \land \\
\qquad p'.\mathit{stones} = p.\mathit{stones} + q.\mathit{stones} \land \\
\qquad p'.\mathit{poisoned} = p.\mathit{poisoned} + q.\mathit{poisoned})
\end{array}
$$

Next, the big pile is split into the requested number of smaller piles. The piles are split as evenly as possible, as are the poison stones: the poison stones are placed in the larger piles if possible.

$$
\begin{array}{|l}
\mathit{level\_piles} : \mathbb{N}_1 \to \mathbb{P}\,\mathit{Pile} \to \mathbb{P}\,\mathit{Pile} \\
\hline
\forall\, \mathit{ps} : \mathbb{P}\,\mathit{Pile} \bullet \\
\quad \mathit{level\_piles}\,1\,\mathit{ps} = \{\mathit{pile\_up}\,\mathit{ps}\} \\
\forall\, p, p', q : \mathit{Pile};\ n, \mathit{ns}, \mathit{np} : \mathbb{N};\ \mathit{ps}, \mathit{ps}' : \mathbb{P}\,\mathit{Pile} \bullet \\
\quad \mathit{ps}' = \mathit{level\_piles}\,n\,\mathit{ps} \Leftrightarrow \\
\qquad (n = \#\mathit{ps} \geq 2 \land \\
\qquad p = \mathit{pile\_up}\,\mathit{ps} \land \\
\qquad p'.\mathit{stones} = p.\mathit{stones}\ \mathsf{div}\ n \land \\
\qquad p'.\mathit{poisoned} = p.\mathit{poisoned}\ \mathsf{div}\ n \land \\
\qquad q.\mathit{stones} = p.\mathit{stones} - p'.\mathit{stones} \land \\
\qquad q.\mathit{poisoned} = p.\mathit{poisoned} - p'.\mathit{poisoned} \land \\
\qquad \mathit{ps}' = \{p'\} \cup \mathit{level\_piles}\,(n-1)\,\{q\})
\end{array}
$$

If the player has not consumed all of their splits, they may select a pile of two or more stones containing no poison stones and split it as evenly as possible.

$$
\begin{array}{|l}
\textit{MoveSplit} \\
\hline
\mathit{Move} \\
p? : \mathit{Pile} \\
\hline
p?.\mathit{stones} \geq 2 \\
p?.\mathit{poisoned} = 0 \\
\exists\, \mathit{rest} : \mathbb{P}\,\mathit{Pile} \bullet \\
\quad \mathit{piles} = \mathit{rest} \cup \{p?\} \land \\
\quad \mathit{piles}' = \mathit{rest} \cup \mathit{level\_piles}\,2\,\{p?\} \\
\mathit{player}'\,\mathit{to\_move}' = \mathit{player}\,\mathit{to\_move}' \\
(\mathit{player}\,\mathit{to\_move}).\mathit{splits} > 0 \\
\mathit{player}'\,\mathit{to\_move} = \mathit{adjust\_player}\,(\mathit{player}\,\mathit{to\_move})\,(0, -1, 0) \\
\mathit{last\_move}' = \mathit{move\_split}\,p? \\
\hline
\end{array}
$$

Finally, if the player has not consumed all of their equalizes, they may pile up all the stones and split them as evenly as possible into the same number of piles they started with.

$$
\begin{array}{|l}
\multicolumn{1}{l}{\textit{MoveEqualize}} \\
\hline
Move \\
\hline
piles' = level\_piles\ (\#piles)\ piles \\
player'\ to\_move' = player\ to\_move' \\
(player\ to\_move).equalizes > 0 \\
player'\ to\_move = adjust\_player\ (player\ to\_move)\ (-1, 0, 0) \\
last\_move' = move\_equalize \\
\hline
\end{array}
$$

A legal game of Acro-Nim consists simply of those moves described above that are legal in the given state.

$$
\begin{aligned}
&InitGame \mathrel{\widehat{=}} InitGameState \\
&Game \mathrel{\widehat{=}} MoveTake \lor MovePass \lor \\
&\qquad MoveSplit \lor MoveEqualize
\end{aligned}
$$

The terminal states are those which have no successor because the game is over.