

Paper:

Design and Implementation of Combinatorial Game Calculator

Kuo-Yuan Kao

Department of Information Management, National Penghu University, Penghu, Taiwan

E-mail: stone@npu.edu.tw

[Received April 20, 2007; accepted September 13, 2007]

The universal game calculator we developed based on a combinatorial game model features a general-purpose game analysis engine, unlike other game-specific programs. Users input game rules by defining game functions and input different game locations by describing game expressions, enabling the system to analyze optimal game strategies.

Keywords: combinatorial game theory, computer game, number theory, nimber, sumber

1. Introduction

Combinatorial game theory has been the basic common mathematical model for analyzing intelligent games since 1970s. Below we review some combinatorial game theory basics [1, 2].

1.1. Basic Definitions

Combinatorial game theory starts from by defining a game simply, i.e., as an ordered pair of sets of games. Conventionally, game G is denoted as:

$$G = \{G^L | G^R\} \dots \dots \dots (1)$$

where G^L and G^R are sets of games. When both G^L and G^R are empty sets (\emptyset), the game is special, and named 0.

$$0 = \{\emptyset | \emptyset\} \dots \dots \dots (2)$$

Each game has a negation; the negation of game G is defined as

$$-G = \{-G^R | -G^L\} \dots \dots \dots (3)$$

The sum of games G and H is defined as

$$G + H = \{G^L + H, G + H^L | G^R + H, G + H^R\} \dots \dots \dots (4)$$

A partial order is defined on the sets of games:

$$G \geq 0 \text{ if and only if no element exists in } G^R \leq 0 \dots \dots \dots (5)$$

$$G \leq 0 \text{ if and only if } -G \geq 0 \dots \dots \dots (6)$$

$$G \geq H \text{ if and only if } G - H \geq 0 \dots \dots \dots (7)$$

An equivalence relation may be further defined on the sets of games:

$$G \equiv H \text{ if and only if } G \geq H \text{ and } G \leq H \dots \dots \dots (8)$$

Equivalence classes form an algebraic group.

1.2. Outcome of Games

Mathematical games defined in the previous section are used to describe the locations of many real-life games, which generally have the following common properties:

- Two players, e.g. L and R , take turns moving.
- The game is a sum of locations, with each location having two sets of subsequent locations, one for each player.
- In a player's turn, the player can choose one location and move to one of its next locations.
- A player who cannot find a move loses.

Each game G has 4 possible outcomes, with the corresponding relations between G and 0 been as follows:

- $G = 0$: The first player cannot win the game.
- $G < 0$: L cannot win the game.
- $G > 0$: R cannot win the game.
- $G || 0$: The first player can win the game.

($G || 0$ shows that neither $G \geq 0$ nor $G \leq 0$, pronounced as G confused with 0.)

Several subgroups of games have had their addition and outcome well-studied, as detailed below.

1.3. Numbers

In the first subgroup, numbers, game G is a number if all elements in G^L and G^R are numbers and no element in G exceeds or equals to any element in G^R . Integers are defined as:

$$\begin{aligned} 1 &= \{0 | \emptyset\}, \\ 2 &= \{1 | \emptyset\}, \\ 3 &= \{2 | \emptyset\}, \\ &\dots \\ n &= \{n - 1 | \emptyset\} \dots \dots \dots (9) \end{aligned}$$

Rationales are defined as:

$$\begin{aligned} 1/2 &= \{0 | 1\}, \\ 1/4 &= \{0 | 1/2\}, \\ 3/4 &= \{1/2 | 1\}, \end{aligned}$$

$$1/8 = \{0|1/4\},$$

$$\dots$$

$$m/2^k = \{(m-1)/2^k|(m+1)/2^k\}. \quad \dots \quad (10)$$

These numbers, integers and rationales, are added in the usual ways. Numbers are well ordered, and their relationships to 0 are clear, so the outcome of any sum of numbers is determined easily.

1.4. Nimbers

In the subgroup, nimbers [3], game G is a number if all elements in G^L and G^R are numbers and $G^L = G^R$. Nimbers are defined as:

$$*1 = \{0|0\},$$

$$*2 = \{0,*1|0,*1\},$$

$$*3 = \{0,*1,*2|0,*1,*2\},$$

$$\dots$$

$$*n = \{0,*1,*2,\dots,*(n-1)|0,*1,*2,\dots,*(n-1)\}.$$

$$\dots \dots \dots (11)$$

The addition of nimbers involves simply: adding the nimbers as binary without carrying, e.g., $*2 + *6 = *4$. The nimber is closed under addition. For simplicity, $*1$ is denoted as $*$ and called a *star*. Each nimber other than 0 is confused with 0, so the outcome of any sum of nimbers is determined easily.

1.5. Sumbers

In the third subgroup, sumbers, each number d has a corresponding *up* defined as:

$$\uparrow(d) = \{\uparrow(d^L),*|\uparrow(d^R),*\}. \quad \dots \quad (12)$$

The negation of up is *down*:

$$\downarrow(d) = -\uparrow(d). \quad \dots \quad (13)$$

We use notation $m \cdot \uparrow(d)$ to denote the sum of m copies of $\uparrow(d)$. A *sumber* [5] S is a sum of ups, downs, and star($*$).

$$S = \sum_{k=1,n} a_k \cdot \uparrow(d_k) + a_0 \cdot * \quad \dots \quad (14)$$

where $a_k, 0 \leq k \leq n$, are integers. Sumbers are closed under addition. We use the notation $G \ll H$ to denote that the sum of any copies of G is less than H . The ups have the following properties, which are sufficient to determine the outcome of any sumber:

$$0 < \uparrow(d_1) < \uparrow(d_2), \quad 0 < d_1 < d_2 \quad \dots \quad (15)$$

$$0 < \uparrow(d_{n+1}) - \uparrow(d_n) < \uparrow(d_n) - \uparrow(d_{n-1}) \quad \dots \quad (16)$$

$$\uparrow(d_{n+1}) + \uparrow(d_{n+1}) - \uparrow(d_n) > * \quad \dots \quad (17)$$

2. Game Calculator Overview

Combinatorial games are generally analyzed in the following steps:

1. Determine the mathematical rules of a combinatorial game.

2. Describe the locations in the game.
3. Simplify each of the locations.
4. Calculate the outcome of the game (sum of locations).

These steps usually involve heavy calculation, requiring a game calculator for both combinatorial game researchers and users. Our goal is to design a calculator with the following functions:

1. Enable the user to input the mathematical rules of a combinatorial game.
2. Enable the user to input locations in the game.
3. Automatically simplify each of the locations.
4. Help the user calculate the outcome of the game.

The game calculator is a high-level language interpreter, and its users may want to write computer programs in the G language.

2.1. Game Notation

The basic data type of G language is the game. The calculator's three built-in games are: number, nimber, and up, e.g.,

```
numbers: 1, 2.011
nimbers: *, *(2)
sumbers: ^, ^(2.011)
```

Each number has an integer expressed conventionally and an optional rationale, which is a string of 0's and 1's led by a decimal point. The rationale is interpreted as binary, e.g., 2.011 is interpreted as $2 + 3/8 = 2.375$. Each nimber is expressed by a $*$ followed by a quoted integer, and $*(1)$ is abbreviated as $*$. Each up is expressed by an \uparrow followed by a quoted number, and $\uparrow(1)$ is abbreviated as \uparrow . An integer n followed by an up represents the sum of n copies of the up, e.g., $3 \uparrow(2)$ represents $\uparrow(2) + \uparrow(2) + \uparrow(2)$. Games are closed under addition and negation. The two operators are $+$ and $-$, e.g., $4 + 2.011 + *(3) + * - \uparrow + \uparrow(2)$.

A game may be an ordered pair of sets of games. Each set of games lists games separated by “,” e.g., $\{\}, \{0,*|0,\uparrow,\uparrow(2)\}$. The following is a partial grammar list:

```
<game>:: <number>
        | <nimber>
        | <up>
        | <game>
        | <game> + <game>
        | <game> - <game>
        | { <game-set> | <game-set> }
        | <function-name> ( <parameter> )
        ;
<game-set>:: <empty-set>
            | <game-set>, <game>
            | <game> : <range>
            ;
```