피연산자들의 트리구조 분석을 통한 게임공식 설계방법

장희동
호서대학교 게임공학과
dooly@hoseo.edu

A Design Method of Game Formulas by Analyzing the Tree Structure of The Operands

Hee-Dong Chang
Dept. of Game Engineering, Hoseo University

요 약

컴퓨터게임은 게임규칙들을 컴퓨터가 자동적으로 처리하기 때문에 게임규칙들이 수학적 함수들로 표현된 게임공식들이 필요하다. 게임공식들은 일반적으로 다변수 함수들이다. 게임공식을 설계하는 것은 관련 게임규칙을 만족하는 다변수 함수를 설계하는 것이기 때문에 복잡하고 어려운 문제이다. 본 논문에서는 게임공식을 체계적으로 설계하는 방법을 제안하였다. 제안하는 방법은 다변수 함수인 게임공식을 피연산자들의 트리구조로 분해하여 이 트리구조의 최하위레벨에는 단일변수 함수들로 구성된다. 그래서 게임공식의 트리구조 분해를 통해 복잡하고 어려운 다변수 함수 설계 문제를 단순하고 쉬운 단일변수 함수 설계 문제로 변경하여 설계하는 방법이다.

ABSTRACT

Computer games need game formulas which express game rules by mathematical functions because the game rules are automatically processed by computers. The game formulas are usually multi-variable functions. So the design of a game formula is a complex and difficult problem because it is the same problem of the design of a multi-variable function which should satisfy the related game rules. In this paper we propose a new method which can systematically design game formulas. The purposed method is the decomposing of tree structure for a game formula by analyzing the tree structure of the operands which have single-variable functions on the lowest levels. So the design method can change the complex and difficult problem of the design of a multi-variable function to the simple and easy problem of the design of the single-variable function which should satisfy the related game rules.

Keywords : Computer Game, Game Formula

접수일자 : 2011년 01월 31일 일차수정 : 2011년 03월 08일 심사완료 : 2011년 03월 22일

1. 서 론

게임 규칙은 게임의 구성요소로서, 게임 내부의 형식적인 구조를 이룬다[1]. 뿐만 아니라 게임규칙은 게임의 질을 결정한다[1].

특별히 컴퓨터게임의 경우, 게임규칙들은 컴퓨터로 처리되기 때문에, 수식으로 표현되는 것이 효율적이다. 게임규칙이 수학적인 함수나 수식으로 표현된 것을 게임공식이라 한다. 게임공식은 관련된 게임규칙들을 정확히 만족해야 한다.

보통 컴퓨터게임의 공식은 다변수 함수나 다변수들의 수식이다. 따라서 게임공식을 설계한다는 것은 관련 게임규칙들을 만족하는 다변수 함수를 설계한다는 의미이다. 다변수 함수의 경우는 그래프 모양을 직접 눈으로 확인하기 어렵기 때문에 게임규칙을 만족하는 함수 그래프를 찾는 것은 매우 어렵고 복잡한 문제이다.

실제 게임산업 현장에서는 게임공식의 설계는 게임디자이너의 경험에 의한 주관적인 방법으로 이루어지고 있다. 이 주관적인 방법은 만들어진 게임공식이 게임규칙을 만족하는 지를 보장하지 못하며 기획단계에서 게임공식의 적절성 여부를 확인하기 힘들다. 따라서 게임구현 후 플레이테스트를 통해 확인해야 하기 때문에 오류 수정의 비용이 높아진다. 또한 게임공식의 오류들은 게임의 재미와 밸런싱에 심각한 문제를 야기할 수 있다. 문헌조사에 의하면, 현재까지 게임공식의 설계방법에 대한 연구결과가 없는 상태이다.

본 연구는 컴퓨터게임의 디자인하는 단계에서 게임규칙을 만족하는 게임공식을 체계적으로 설계하는 방법을 제안한다. 제안하는 설계방법은 게임공식을 피연산자(operand)들의 트리구조로 분해하여 다변수 함수 설계문제를 보다 쉬운 단일변수 함수의 문제로 변환하여 설계하는 방법이다.

2장에서는 제안하는 게임공식의 설계방법을 설명하고 3장에서 설계방법의 연구결과에 대한 결론을 내린다.

2. 게임 공식의 설계 방법

게임규칙을 수식으로 표현한 게임공식은 일반적으로 여러 개의 입력변수들을 갖는 함수(multi-variable function)이다. 또한 입력변수들의 값과 함수의 값을 유저가 사용하는 경우는 정수(integer) 값을 사용한다. 왜냐하면 유저가 입력변수의 값을 선택하거나 유저에게 게임공식의 결과값을 보여주는 경우는 정수를 사용하여 값의 의미를 충분히 이해할 수 있도록 해야 하기 때문이다[2,3,4,5]. 대표적인 게임공식들은 데미지 계산 공식, 명중률 계산 규칙, 아이템 가격 산정 공식이다[2,3,4,5].

일반적으로 다변수 함수 형태인 게임공식은 다음과 같은 특징들을 갖고 있다.

(1) 게임공식 함수는 사용자가 사용하는 정보인 경우 입력변수 값 또는 함수 값을 정수 값으로 취한다.

(2) 게임공식 함수는 하나의 입력속성을 독립변수로 고려하면 증가함수 또는 감소함수의 그래프 모양을 갖는다. 왜냐하면 게임규칙적인 관점에서 보면 대응되는 속성(대응 수식)의 값은 입력되는 속성 값의 증가에 대하여 증가하거나 또는 감소하는 대응관계[2,3,4,5]를 갖기 때문이다.

(3) 게임공식은 속성들 사이의 증가 또는 감소의 관계성 규칙을 나타내는 공식이기 때문에 단일변수의 증가함수 또는 감소함수들과 랜덤함수, 비례함수, 누적함수, 차수함수, 지수함수들의 사칙연산(덧셈, 뺄셈, 곱셈, 나눗셈)들로 이루어진 수식으로 표현될 수 있다. 경우에 따라서는, 함수값을 정수값으로 변환하기 위해 반올림이나 정수자리 값만 취하는 연산을 추가한다.

게임공식은 다변수 함수이기 때문에 일반적으로 게임규칙들에 만족하는 함수를 설계하는 것은 어렵다.

본 논문에서 제안하는 게임공식의 설계방법의 핵심내용은 '게임공식은 단일변수 함수들의 사칙연산들로 구성되어 있다'는 특징을 사용하는 것이다.

즉, 설계하고자 하는 게임공식을 피연산자들의 트리구조 분석을 통해 하위레벨의 보다 단순한 피연산자들을 찾아내고 최하위 레벨에는 하나의 독립변수만을 사용하는 단일변수 함수들로 구성된 트리구조를 찾아낸다. 그리고 트리구조의 최하위 레벨의 단일변수 함수는 함수의 그래프 모양의 분석을 통해 찾아내고 이들 단일변수 함수들을 게임공식의 트리구조에 따라 사칙연산으로 상위레벨의 연산모듈로 통합되고 같은 방식들이 반복되어 최종적인 게임공식으로 통합되는 방법이다.

본 논문에서 제안하는 게임공식의 설계방법을 제시하기 위해, 먼저, 다음과 같은 내용들을 가정한다.

(1) 입력변수들의 속성들과 함수 값으로 대응하는 출력 속성들 사이의 관계성에 관련된 규칙들은 미리 준비되었다고 가정한다.

(2) 게임공식은 [그림 1]과 같이, 여러 개의 피연산자들과 사칙연산으로 구성되어 있고. 하나의 피연산자는 다시 여러 개의 피연산자들과 사칙연산으로 이루어진, 트리구조로 분해 할 수 있다. 트리구조의 최하위 레벨은 단일변수의 함수들로 구성되어 있다고 가정한다.

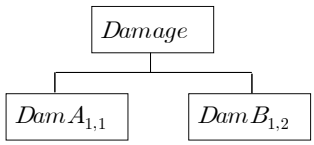
본 논문에서는, 제안하는 게임공식의 설계방법의 설명을 위해, Blizzard사의 게임 Diablo 2[3]의 데미지 규칙을 기반으로 한 게임공식설계에 대한 예를 함께 제시한다.

제안하는 게임공식 설계방법의 자세한 절차들은 다음과 같다.

(1) 게임공식에 관련된 규칙내용을 분석하여, [그림 1]과 같이, 게임공식의 결과 값을 결정하는 연산자와 피연산자들을 결정하여 최상위 트리구조를 정한다.

- 이때 트리구조는 규칙내용을 만족하는 피연산자들과 사칙연산자(+, - ,×,÷)의 종류들과 연산순서를 통해 결정된다. 즉 사칙연산이 먼저 이루어지는 피연산자일수록 하위레벨에 위치한다. 필요시 피연산자들의 값의 범위나 제한 조건들을 추가한다.

$Damage = DamA_{1,1} + DamB_{1,2}$
여기서,
$Damage$: 무기들의 공격에 의해 받게 되는 총 데미지
$DamA_{1,1}$: 주무기 공격에 의한 실제 데미지
$DamB_{1,2}$: 다른 무기들 공격에 의한 실제 데미지

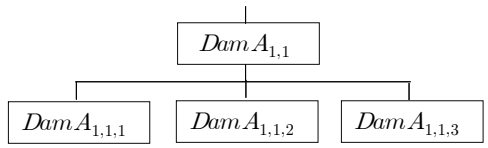


[그림 1] Diablo 2의 데미지 공식 설계를 위한 최상위 연산자와 피연산자들 그리고 관련된 트리구조

- 게임공식에 관련된 규칙내용을 만족하는 피연산자들과 사칙연산들의 결정은 관련규칙내용을 사칙연산의 의미로 재해석하여 이루어진다. 즉, 덧셈은 두 피연산자들의 양들을 합한다는 의미가 있고 또 다른 의미는 하나의 피연산자를 다른 피연산자의 양만큼 이동한다는 의미이다. 가중치 덧셈(weight sum)은 여러 피연산자들을 특정비율의 가중치들을 사용하여 합하여 통합한다는 의미이다. 뺄셈은 두 피연산자의 차이나 또는 하나의 피연산자의 양에 대해 다른 피연산자의 양만큼 뺀다는 의미이다. 곱셈은 하나의 피연산자에 대해 다른 피연산자의 양만큼의 배율로 스케일링한다는 의미이다. 나눗셈은 하나의 피연산자에 대해 다른 피연산자의 양만큼의 역배율로 스케일링한다는 의미이다.

(2) 최상위 피연산자들 중 하나를 선택하고 이 피연산자 값을 결정하는 연산자와 피연산자들을, [그림 2]와 같이 결정한다.

$DamA_{1,1} = [DamA_{1,1,1} * DamA_{1,1,2} * DamA_{1,1,3}]$
여기서,
$[X]$: 실수 X의 소수부분 버림(round down)
$DamA_{1,1,1}$: 주무기의 데미지
$DamA_{1,1,2}$: 공격자의 힘과 스킬의 효과로 인한 데미지 반영비율
$DamA_{1,1,3}$: 무기 마스터리에 의한 데미지 반영 비율

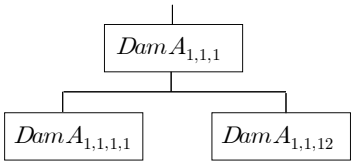


[그림 2] Diablo 2의 데미지 공식 설계를 위한 두번째 레벨의 연산자와 피연산자들 그리고 관련된 트리구조

- 동일한 방법으로 나머지 피연산자들 중에 하나를 선택하여 이 피연산자 값을 결정하는 연산자와 피연산자를 결정한다. 그래서 동일레벨의 모든 피연산자 값을 결정하는 연산자와 피연산자를 결정한다.

(3) 지금까지 구한 트리구조의 최하위에 위치한 피연산자에 대하여 [그림 3]과 같이 (2)와 동일한 방법으로 관련된 연산자와 피연산자를 구하고 트리구조를 갱신한다.

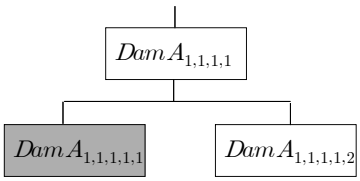
$DamA_{1,1,1} = Rand(DamA_{1,1,1,1}, DamA_{1,1,1,2})$
여기서,
$Rand(I,J)$:정수 I와 정수 J사이의 임의의 정수
$DamA_{1,1,1,1}$: 공격 무기의 최소 데미지
$DamA_{1,1,12}$: 공격 무기의 최대 데미지



[그림 3] Diablo 2의 데미지 공식 설계를 위한 세번째 레벨의 연산자와 피연산자들 그리고 관련된 트리구조

(4) 게임공식의 트리구조의 최하위에 위치한 피연산자가, [그림 4]의 피연산자 $DamA_{1,1,1,1,1}$ 와 같이, 단일독립변수의 함수가 될 때까지 (3)의 과정을 반복 적용한다.

$DamA_{1,1,1,1} = DamA_{1,1,1,1,1} + DamA_{1,1,1,1,2}$
여기서,
$DamA_{1,1,1,1,1}$: 공격 무기의 자체 최소 데미지
$DamA_{1,1,1,1,2}$: 공격 무기와 관련된 아이템들의 추가 최소 데미지



[그림 4] Diablo 2의 데미지 공식 설계를 위한 네번째 레벨의 연산자와 피연산자들 그리고 관련된 트리구조

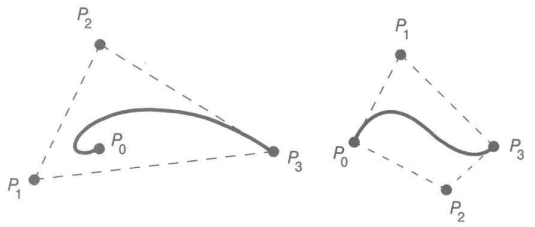
(5) 게임공식의 트리구조의 최하위레벨의 피연산자에 대한 단일변수의 함수를 결정한다.

- 단일변수 함수는 랜덤함수의 경우와 아닌 경우로 나누어진다. 랜덤함수(random function) [2,3,4,5]의 경우는 변수 값의 적절한 범위를 지정하여 결정한다.

- 랜덤함수가 아닌 경우는 단일변수 함수는 증가함수이거나 또는 감소함수이다. 함수의 결정은 2가지 방법으로 이루어진다. 첫 번째 방법은 정비례 함수, 실수형 차수 함수, 누적 합 함수, 지수형 함수, 또는 삼각함수의 그래프들[6]의 모양을 비교하여 적합한 증가모양 또는 감소 모양을 갖는 그래프의 함수들 정한다. 그리고는 게임규칙을 만족하는 함수 값의 범위를 만족하기 위해, 필요시, 스케일 상수 곱 연산과 평행이동을 위한 이동 상수의 합 연산을 추가하여 완성한다. 두 번째 방법은 첫 번째 방법에서 적합한 증가 또는 감소 모양의 그래프를 갖는 기본 함수가 없다고 판단될 경우는, [그림 5]와 같이 사용자가 임의로 증가 또는 감소 모양을 정할 수 있는 xy-평면상의 Bezier

Curve[7]를 통해 단일변수 함수의 그래프 곡선을 결정 한다. 하지만 이 Bezier Curve는 매개변수 t $(0 \le t \le 1)$에 관한 그래프 상의 점의 위치좌표 $(x,y,0)$를 출력하는 매개변수 t 의 함수이기 때문에 $y = f(x)$ 함수로 변환해야 한다. 이것은 $x = g_1(t), y = g_2(t)$ (여기서, $g_1(t), g_2(t)$는 Bezier Curve 수식에서 나온 t에 관한 3차 다항식[7]) 의 관계와 3차 다항식의 3가지 일반해의 공식들[8]을 이용하고 범위조건인 $0 \le t \le 1$를 적용해서 역함수 $t = g_1^{-1}(x)$를 구할 수 있다. 따라서 우리가 찾고자 하는 단일변수 함수는 $y = f(x) = g_2 \circ g_1^{-1}(x)$로 구한다.

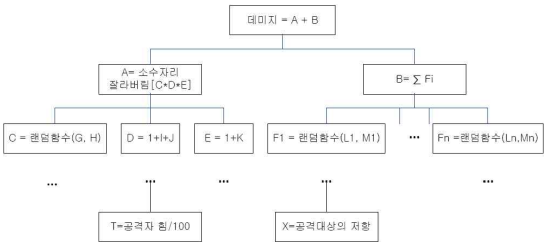
만약 함수 값이 정수값으로 출력되어야 하는 경우는 반올림, 또는 소수버림 연산을 추가한다.



[그림 5] 4개의 조정점들을 사용하여 자유로운 곡선을 만들 수 있는 Bezier Curve[7]

(6) [그림 6]과 같이, 완성된 게임공식의 트리구조를 통해 모든 피연산자들과 연산자들을 통합하여 최상위레벨의 게임공식을 [그림 7]과 같이 도출한다.

(7) 통합된 게임공식이 관련 게임규칙과 만족하는 지 여부를 검사하기 위해, 트리구조의 최하위레벨의 단일변수 함수의 입력변수의 샘플 입력 값들과 게임공식의 계산 값들의 관계를 검사하여 게임규칙의 내용을 만족하는 지 여부를 확인하고 만약 게임규칙의 내용을 만족하지 않는 오류들이 검출되면 게임공식의 트리구조의 중간레벨의 연산자와 피연산자들의 중간 결과 값들을 검사하여 연산자들을 수정하여 오류들을 제거한다.



[그림 6] Diablo 2 데미지 계산 공식 설계를 위한 전체 트리 구조도

$$Damage = \left[Rand(MinDam, MaxDam) * (1 + \frac{Str}{100} + \frac{SBs}{100}) * (1 + \frac{Mast}{100}) \right] + \sum_{i=1}^{i=} $$

$Damage$: 무기들의 공격에 의해 받게 되는 총 데미지
$[X]$: 실수 X의 소수부분 버림(round down)
$Rand(I,J)$: 정수 I와 정수 J사이의 임의의 정수
$MinDam$: 공격 무기의 최소 데미지와 아이템에 의한 추가 최소 데미지의 합
$MaxDam$: 공격 무기의 최대 데미지와 아이템에 의한 추가 최대 데미지의 합
Str : 공격자의 힘
SBs : 무기 마스터리(mastery)를 제외한 모든 스킬 보너스에 관련된 데미지들의 합
$Mast$: 바바리언 종족의 무기 마스터리에 대한 데미지 보너스
#EleDam : 그밖에 따로 계산되어야 하는 다른 공격무기와 관련된 다른 아이템에 대한 기본 데미지들의 수
$EleMin_i$: i번째의 따로 계산되어야 하는 다른 공격무기의 최소 데미지와 관련된 아이템의 최소 데미지 합
$EleMax_i$: i번째의 따로 계산되어야 하는 다른 공격무기의 최대 데미지와 관련된 아이템의 최대 데미지 합
$TarR_i$: i번째의 따로 계산되어야 하는 다른 공격무기에 대한 피공격자의 저항력

[그림 7] Diablo 2의 최종 데미지 계산 공식

3. 결 론

게임 규칙은, 게임의 구성요소로서, 게임 내부의 형식적인 구조를 이루고 게임의 질을 결정한다. 특히 컴퓨터게임의 경우는 게임 규칙을 수식으로 표현한 게임공식을 많이 사용한다. 이는 컴퓨터가 규칙들을 처리할 때 수식이 매우 효율적이기 때문이다. 예를 들면 데미지 계산 공식과 명중률 계산 공식이 대표적인 게임공식이다.

게임공식은, 수학적으로 보면, 다변수 함수이다. 따라서 규칙의 내용을 만족하는 게임공식을 설계하는 것은 다변수 함수를 설계하는 것이기 때문에 일반적으로 복잡하고 어려운 문제이다. 게임 산업체의 경우는, 검증되지 않은 주관적인 방법에 의존하여 게임공식을 설계하였다. 이것은 게임의 재미와 밸런싱에 심각한 문제를 야기할 수 있다.

본 논문에서 제안한 게임공식의 설계방법은 다음과 같다. 게임공식의 주요특징의 하나인 '게임공식은 단일변수의 증가함수, 감소함수, 랜덤함수, 또는 정수변환 함수의 사칙연산들로 이루어져 있고 이러한 사칙연산들은, 게임규칙의 내용을 연산적 의미로 재해석하여, 결정할 수 있다'는 것을 이용한 것이다. 즉, 게임공식을, [그림 6]과 같이, 피연산자들의 트리구조로 분해하여 다변수 함수 설계문제를 쉬운 단일변수 함수의 설계 문제로 변환하고 그리고 단일변수 함수의 설계 문제는 랜덤함수를 제외한 증가함수와 감소함수의 기본 함수들을 제안하고 이들 기본함수의 그래프 모양들을 비교하여 이들 중에 가장 적합한 함수를 찾고 적정한 함수값의 범위로 조정하기 위해 선택한 함수에 스케일링과 시프팅을 하여 설계를 완료하고 만약 적절한 기본 함수들이 없을 경우는 Bezier 곡선을 4개 조정점들을 통해 조정하여 원하는 곡선 그래프 모양을 직접 정하여 설계한다. 그 다음 피연산자들을, 트리구조에 따라 상위레벨 피연산자를 통합하여 최상위레벨 게임공식을 결정한다. 샘플 입력값들과 계산공식 결과값들의 관계를 비교하여 오류를 찾아내고 트리구조의 수정보완을 통해 오류를 해결하여 최종적인 게임공식의 설계를 완료한다.

본 연구결과는 컴퓨터게임 디자인에서 중요한 작업인 게임공식 설계를 위해 수학적으로 체계적인 방법이기 때문에, 정확한 게임공식의 설계를 보장할 수 있다.

참고문헌

[1] Katie Salen and Eric Zimmerman, "Rules of Play: Game Design Fundamentals", The MIT Press, 2004.

[2] TSR "Dungeons & Dragons", 2nd Ed. TSR, 1995.

[3] TSR, "Advance Dungeons & Dragons 한글판", 2nd Ed. TSR, 1997.

[4] Steve Jackson, 김성일 (편역), "GURPS 국문판", 도서출판 초여명, 1998.

[5] "Addiction's Diablo II Statistics Guide", http://addicted13.20m.com/index.html, 2001.

[6] William H. Beyer, "CRC Standard Mathematical Tables", 27th Ed. CRC Press, 1984.

[7] J. D. Foley, "Computer Graphics Principles and Practice", 2nd Ed., Addison-Wesley Pub., 1987.

[8] Cubic function, Wikipedia, http://en.wikipedia.org/wiki/Cubic_function

장 희 동 (Chang, Hee Dong)

1987-1997 한국전자통신연구원 영상통신연구실 선임연구원
1998-2002 숭의여자대학 컴퓨터게임과 조교수
2003-현재 호서대학교 게임공학과 부교수

관심분야 : 교육용게임 디자인, 디지털게임 디자인, 게임 메카닉스 디자인