한글 문자의 음소 및 음절 문자 특성의 구현 방안

변정용 강진곤
동국대학교 전자계산학과

An Implementation Method for The Phonemic and Syllabic Character Attributes of Hangul Character

Jeongyong Byun and Jingon Kang
Department of Computer Science, Dongguk University

요 약

훈민정음 해례에 따르면 한글문자는 음소 및 음절 문자 특성을 가지고 있다. 이러한 특성들을 컴퓨터 시스템에서 구현함에 있어서 야기되어 왔던 각종 문제를 분석한 다음 이들 문제들에 대하여 한글문자의 특성을 제약함이 없이 컴퓨터에 대한 기술을 개발함으로써 해결책을 모색한다. 본 논문은 훈민정음 해례에서 밝힌 한글 문자의 음소 및 음절 문자 특성에 따라서 기존의 코드 체계를 평가하며, 그리고 이들에 대한 구현 방안을 제시하고자 한다. 또한 이러한 특성을 반영한 한글 입출력틀인 '셔틀'을 개발하고 이에 대한 검증을 시도하였다.

1. 서론

컴퓨터는 인간의 정신 노동의 부담을 덜어 주는 대표적인 도구 가운데 하나이다. 그것은 한글 문서 작성기(워드 프로세서)를 가지고 인간의 정신 노동의 정체성을 어느 정도 해결해 주고 있기에 그렇다. 하지만 현재 컴퓨터에서 문서 작성은 문서 편집의 편리함을 제공하는 반면에 인간의 사고를 제한한다는 점이 문제이다. 다시 말해서 연필로 쓸 때는 자유롭지만 컴퓨터에서는 컴퓨터가 표현할 수 있는 한정된 문자만 사용해야 한다는 것이다. 이것은 도구의 문제라기 보다 도구를 만든 사람의 문제로 보아야 할 것이다.

컴퓨터에서 한글을 표현하려면 먼저 이들 글자 꼴을 대신할 수 있는 코드가 필요하다. 그래서 KS C 5601-1974를 제정하였지만 조합형 KS C 5601-1982를 거쳐서 오늘날 완성형 KS C 5601-1987에 이르고 있다[1]. 완성형은 프로그래밍의 편리성을 높인 반면에 한글의 특성인 음소 문자 특성을 표현하는 것이 불가능하고[2], 음절의 표현 범위를 크게 줄여 놓았다.

본 논문은 훈민정음 해례에서 밝힌 한글 문자의 음소 및 음절 문자 특성에 따라서 기존의 코드 체계를 평가하며, 그리고 이들에 대한 구현 방안을 제시하고자 한다. 또한 이러한 특성을 반영한 한글 입출력틀인 '셔틀'을 개발하고 이에 대한 검증을 시도하였다.

이 후의 논문 구성은 2장에서 음소 및 음절문자 특성과 기존 코드계의 평가를 하고, 3장에서 그 구현 방향을 제시하며, 4장에서 '셔틀'을 소개하고, 5장에서 결론을 맺는 순서로 되어 있다.

2. 한글문자의 개별적 특성

한글 문자는 문자로써의 일반성뿐만 아니라 개별성을 가지고 있다. 그 개별성으로서 훈민정음 해례[3,12]는 음소 및 음절문자 특성을 동시에 가짐을 분명하게 정의하고 있다. 이에 따르면 음소 글자의 집합을 원소나열법과 조건제시법으로 정의하고, 음절 글자의 집합은 조건제시법만으로 정의하고 있는 것으로 파악된다[4].

2.1 음소 문자 집합

음소 문자의 특성은 초성해에서 자음 17자를 정의하고, 중성해에서 모음 11자를 정의하였다. 또한 중성해는 초성해의 자음 17자와 같음을 '종성부용초성(終聲復用初聲)'이라 하였고, 이것은 등가관계로 표시함으로써 간략화한 것으로 본다[5,6]. 음소 문자의 특성은 합자해에서 계속되는 데 여기서 자음과 모음이 두 자, 석 자의 복자음 또는 복모음 집합으로 확장된다. 연서법(連書法)은 입술소리 (脣音) 글자인 ㅁ, ㅂ, ㅃ, ㅍ 아래 ㅇ 을 잇대어 쓰면 입술가벼운 소리(脣輕音)가 된다고 하고, 합용법(合用法)은 초성자 또는 중성자를 두 글자 또는 세 글자 함께 나란히 쓸 수 있다고 하였다. 여기서 초성자와 종성자의 집합은 각각 5219자와 1463자로 확장된다. 음소문자의 정의를 훈민정음 해례는 사실과 규칙을 이용하여 정의하고 있음을 알 수 있다.

2.2 음절문자 집합

훈민정음 해례의 합자해는 계속해서 음절문자 특성을 정의하고 있다. 먼저 부서법(附書法)은 앞에서 정의한 초성자, 중성자, 종성자들을 이차원의 기하구조를 가진 한 음절 글자로 구성하는 방법을 정의하고 있다. 그 내용은 중성자의 꼴이 가로꼴과 세로꼴의 형태를 취하고 있음에 따라서 가로꼴 중성자는 초성자 아래 쓰고, 세로꼴은 오른 녘에 쓰며, 종성자는 그 아래 잇대어 쓴다고 규정하고 있다. 그래서 한글은 음절 글자의 특성을 가지게 된다. 여기서 중요한 것은 이차원의 구조를 이룰 때 중성자 즉 모음이 세로꼴이냐 가로꼴이냐가 글자의 꼴을 다르게 결정한다는 사실이다. 그리고 합자해는 왜 이렇게 모아쓰느냐를 밝히고 있는 데 '범자필합이성음(凡字必合而成音)'이라 하여 초성자, 중성자, 종성자를 모아야 소리가 이루어진다고 해서 음절 글자를 구성하는 이유를 규정한 것이다.

2.3 가나 및 로마 문자와 비교

문자론적으로 한글 문자는 소리를 표기하는 표현문자에 속하며, 표현문자 가운데서도 초성자, 중성자, 종성자로 표현되어 있기 때문에 음소문자의 속성을 가졌으며, 또한 한 음절 글자가 이차원의 기하구조를 가지고 있기 때문에 음절문자의 속성을 동시에 가지고 있음을 보았다.

한글 문자의 개별적 속성은 가나 문자나 로마 문자와 비교하였을 때 더욱 분명하여 진다. 이들이 한글 문자와 마찬 가지로 표음문자라는 점에선 같다. 하지만 개별적 속성에서 가나 문자는 음소 문자 특성을 가지지 못한 음절문자 특성만을 가지고 있으며, 로마 문자는 음절 문자 특성을 가지지 못한 음소문자 특성만을 가지고 있다. 이들은 음소 및 음절 문자 속성을 함께 가지고 있는 한글 문자와 비교할 때 어느 한 쪽만 가지고 있으므로 반쪽 한글에 해당한다고 할 수 있다. 이 처럼 한글문자와 이들은 분명하게 그 속성이 다름을 알 수 있다.

3. 기존 코드계의 평가

한글문자의 속성인 음소 및 음절 문자 특성을 기존 코드계에서 어떻게 표현하고 있는 지를 초성자,중성자, 종성자의 표현여부, 그리고 음절 정보의 표현 방식에 대하여 평가한다.

3.1 자모형

최초의 정보교환용 한글 코드인 KS C 5601-1974[1]는 낱자형 또는 자모형이라 하여 한글의 자음과 모음을 코드화 대상으로 삼고 한글 오토마타를 통하여 음절을 인식하고 이들을 모아 음절 글자를 구성하였다. 자음의 집합은 초성 및 종성의 모든 복자음을 포함해서 30자로 구성되고, 모음의 집합은 모든 복자음을 포함해서 21자로 구성되었다. 따라서 음소 및 음절 문자 특성이 기본적으로 표현이 되어 있다고 본다.

그런데 문제는 훈민정음 해례의 종성부용초성에 따라서 종성자 코드를 별도로 구성하지 않고 자음을 초성자와 종성자의 도메인으로 삼은 것이다. 여기서 두 가지 사항이 문제로 지적될 수 있다. 첫째, 현대 맞춤법에 따를 때 초성에 올 수 있는 복자음의 종류와 종성에 올 수 있는 복자음의 종류가 다르기 때문에 음절 글자 속성과 관련해서 자음의 속성이 초성인지 종성인지를 판단해 주어야 한다는 것이며, 둘째, 복자음과 복모음에 대하여 하나의 코드를 부여함으로써 그들이 각각 어떠한 단자음과 단모음의 결합인지 알 수가 없다는 점이다. 예를 들어서 종성자 ㄺ은 코드 값이 십육진법으로는 0x41이다. 그러나 코드만을 통하여 'ㄹ'과 'ㄱ'으로 구성되어 있음을 알 수 없다.

3.2 조합형

조합형 KS C 5601-1982[1]은 현대 국어 맞춤법에 따라서 초성자 19자, 중성자 21자, 종성자 28자를 열여섯 비트에 5비트씩 세구획으로 나누어 각각을 표현한 방식이다. 여기서 음소문자 특성의 표현은 초성자, 중성자, 종성자로 분명하게 표현되어 있으며, 음절문자 특성의 표현도 음절단위로 표현하였다는 점에서 일차원의 명시적 표현으로 되어 있다. 여기서 문제점은 초성자, 종성자, 종성자의 개별 부분 코드가 역시 복자음과 복모음에 대하여 부여하고 있다는 점이

다. 그런데 조합형의 근본적인 문제는 ISO 646IRV와 같은 국제 표준 규격에 위배되기 때문에 코드로써의 보편성을 상실하고 있다는 점이다.

3.3 완성형

현행 표준인 완성형 KS C 5601-1987[1]은 2350음절 글자를 먼저 선정하여 이에 대하여 코드를 부여하고 있기 때문에 일단 음소 문자 특성은 배제시키고 있다. 그래서 '한'이라는 음절 글자가 두 바이트 코드 값 0x7d 0x75 만을 보고서 'ㅎ','ㅏ','ㄴ'으로 구성되어 있음을 알아 낼 수 있는 방법이 없다는 것이다. 뿐만 아니라 완성형은 현대 국어 맞춤법이 표현할 수 있는 음절 글자의 약 1/5에 해당하는 글자만 표현한다는 점도 중대한 결격 사유가 된다.

3.4 ISO 10646의 자소형

현행 국제문자 표준 코드계인 ISO 10646상에 훈민정음 창제 이래로 사용된 초성자, 중성자, 종성자 240자가 반영되어 있다[7,8,13]. 이것은 역시 음소문자를 일차적으로 표현하고 이후 모아쓰기 자동틀을 통하여 음절문자 특성을 표기하려는 방식이다. 자모형과 다른 점은 자음의 속성이 초성자와 종성자로 분류되어 있다는 점이다. 여기서 문제는 복자소에 대한 코드 부여와 아직도 추가로 자소가 발견될 수 있기 때문에 완전한 자소 집합이 아니며[16], 자소의 배열 방식에 있어서도 현대 한글과 옛 한글을 분리하여 배치한 점 그리고 단수 바이트 코드계에서 사용할 수 없다는 점등이다.

4. 구현 방안

4.1 개별적 속성

한글문자는 이원적 속성을 가지고 있기 때문에 이를 로마자나 가나문자의 입장에서 표현 방법을 찾는 것은 불가능하다. 완성형의 경우에는 가나문자의 입장에 있다. 왜냐면 음소문자적 특성을 완전히 상실하였기 때문이다. 만약 로마자 입장을 따른다면 풀어쓰기 한글을 구현하게 된다. 그러므로 한글의 개별적 속성을 가지고 있음이 입증된 이상 한글문자에 적합한 구현 방법을 개발해야 할 것이다.

먼저 한글문자에서 음소문자 속성과 음절문자 속성이 어떻게 존재하는 지를 보면 앞에서 밝혔듯이 음소문자 속성은 원소나열법과 조건제시법으로 정의하고, 음절문자 속성은 조건제시법으로 정의하고 있다. 전자는 사실(fact)이고, 후자는 규칙(rule)에 해당한다[4,6]. 따라서 사실은 직접적인 표현대상이 되고 규칙은 어떤 요구가 있을 때 사실에 대하여 새로운 정보를 생성해 낸다. 이러한 절차를 통하여 한글 문자의 이원적 속성을 구현함으로써 결국 훈민정음 해례의 원리를 그대로 따르는 것이 된다. 일차적으로 기본 28자를 정의하고 다시 그들을 확장한다. 그리고 나서 부서법과 성음법을 통하여 이차원의 기하구조를 가진 음절글자를 구성한다.

구현에서 사실인 기본 글자는 바로 코드화 대상이 된다. 여기서 구현 절차를 훈민정음 해례와 결부해서 설명해 본다. 사실이란 초성해와 중성해의 28자이다. 그런데 종성해에 대한 유권 해석의 여부에 따라서 17자를 추가로 사실화할 수 있다. 왜냐면 종성해는 초성해와 같은 내용을 반복해서 서술하는 것을 회피할 목적으로 '종성부용초성'이라 한 것으로 보기 때문이다. 그래서 45자를 코드화 대상으로 하고, 다음 단계로 복자소 확장 규칙을 적용한다. 복자소로 확장하는 것은 입력 과정에서 일차원의 문자열을 허용하는 것이며, 이들에 대한 글자꼴의 지원 여부가 허용의 여부로 이어진다. 일차원으로 표현된 문자열은 초성자, 중성자, 종성자에 대한 정보를 가지고 있거나 아니면 적어도 자음 또는 모음이라는 속성을 가지고 있다. 이들은 바로 음절 글자 정보이며 음절 글자의 표시가 요구될 때에만 적용된다. 어쨌든 훈민정음 해례에서 규정한 대로 법칙들은 최종 단계에서 적용되도록 해야 한다. 그래야 한글의 기본 문자 집단을 최소화할 수 있을 것이며, 고유한 특성 다시 말하면 다른 문자와 구별되는 개별적 속성을 유지할 수 있을 것이다.

내부 표현 : 일차원
-> 모아쓰기틀(Transducer)
-> 외부 표현 : 이차원

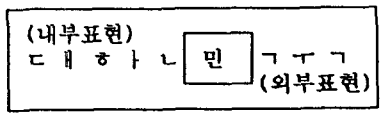


그림 4.1 음소 및 음절 문자의 구현

4.2 한글 코드와 정보의 표현

4.2.1 음절 글자 정보

한글의 음절 정보를 한글 코드에 표현하는 방법에는 크게 두 가지가 있다. 하나는 음절단위 표현이고, 다른 하나는 음절 글자 정보를 내포한 음소 글자 코드를 일차원으로 나열하는 것이다. 전자는 완성형과 조합형이 그 예이고, 후자는 자모형과 자소형이 그 예이다. 음절 글자 단위 표현에서 완성형은 음소 글자 정보가 없고, 조합형은 있다는 점이 다르다. 여기서 음소글자 단위 표현 방법을 좀 더 자세히 검토해 본다.

음절글자 단위 : 조합형, 완성형
음소글자 단위 : 자모형, 자소형(정음형)

음소글자 단위 표현법에는 훈민정음 해례의 종성해에서 '종성부용초성'의 해석에 따라서 두 가지 표현으로 나뉜다. 하나는 초성자, 중성자, 종성자로의 표현이고, 다른 하나는 자음글자와 모음글자로 구별한 표현이다. 첫번째 표현법은 한글의 한 음절 글자가 초성자, 중성자, 종성자로 이루어져 있다는 점을 반영한 표현이다. 두번째 표현법은 한글의 글자꼴이 28자라는 점에만 국한하고 그것의 음가나 한 음절에서 역할 정보는 담고 있지 않다. 다시 말하면 한 글자의 구성 요소가 홑소리냐 닿소리냐의 구별을 중심으로 표현한 것이다. 그래서 나중에 음절 글자 구성을 할 때에 자음자의 속성이 초성자인지 종성자인지 구별해야 하는 약간의 부담이 따른다. 이것은 현대 한글의 표현에서는 문제가 되지 않지만 옛 한글을 표현할 때는 문제가 된다. 두 표현법에서 공통적인 과제는 음절 글자 구성에 필요한 모음자의 꼴에 관한 정보로서 그것이 가로꼴인지 세로꼴인지를 알 필요가 있다.

4.2.2 가로꼴 또는 세로꼴 정보

이에 대한 정보는 33자 한글 코드[9]에서 제시된 바 있다. 기본 모음을 6,7열에 표현하면서 세로꼴은 6번열에 그리고 가로꼴은 7번열에 배치하였다. 단지 ㅣ 는 순서상 마지막이므로 어쩔 수 없이 7번열의 0x1100에 행에 배치하였으나 다른 가로꼴을 0x**00에 배치하지 않으므로써 쉽게 찾을 수 있도록 하였다. 이처럼 모음 또는 중성자의 코드상에 가로꼴 및 세로꼴에 관한 정보를 표현한다면 음절 글자 구성에 필요한 정보는 자소 정보와 함께 모두 가진 것으로 볼 수 있다. 다른 한 편으로 이에 대한 정보 표를 이용할 수도 있다. 처리 성능면에서 크게 차이가 나지 않기 때문에 코드에 표현하는 것이 이상적이지만 표를 이용하는 것도 별 문제가 없는 것으로 본다.

세로꼴: ㅏ ㅑ ㅓ ㅕ ㅐ ㅒ ㅔ ㅖ ㅣ
가로꼴: ㅗ ㅛ ㅜ ㅠ ㅡ

4.2.3 복자소

해례에서 연서법과 합용법은 기본자를 둘 또는 세 자씩 함께 이어 쓰거나 나란히 씀으로써 자소를 확장할 수 있다고 하였다. 그런데 이들 법칙을 미리 적용하여 이들 복자소에 대하여 코드를 부여한다면 복자소의 구성요소에 관한 정보를 잃어 버리는 결과에 이르게 된다. 여기서 복자소의 구성 요소 정보를 표를 가지고 해결할 수도 있겠지만 집합의 크기가 큼으로 기본 자소의 나열로써 표현한다면 구성에 관한 정보는 자연스럽게 표현할 수 있다.

완성형 : 한 - 0x7d 0x75
정음형 : 한 - 0xc6 0xd1 0xe3

4.3 음절 및 음소 연산

기존의 편집기들은 대부분 음절 연산만을 적용해왔다. 물론 일부는 입력과정에서 음소글자 단위로 지울 수 있으나 그것도 거꾸로 지우기가 고작이다. 그러다 보니 음소 글자에 대한 연산은 거의 생각하지 못하고 있다. 또한 그렇게 된 가장 큰 이유는 완성형 코드 때문에 그렇고, 단순하게 생각해서 한 음절 글자 가운데 특정 자소의 수정은 그 음절을 지우고 새로 입력하는 것으로 생각이 고정되어 있기 때문이다.

하지만 다른 문자와는 달리 한글문자는 음절 및 음소문자 특성을 함께 가지고 있기 때문에 음소 글자 단위 연산은 당연한 요구이다. 음소 연산이란 영문 편집기에서 한 글자 단위로 반디(커서)를 옮기 듯이 한 음절 글자의 구성 요소인 초성자, 중성자, 종성자에 대하여 옮겨서 가리킬 수 있거나, 또는 각 자소 마다 대치, 삽입, 삭제를 할 수 있어야 한다. 반디이동, 삽입,삭제는 기본 모드에서 가능하고 대치는 별도의 모드를 두도록 한다. 이들 연산에 대한 일반 연산 규칙을 고려하면 다음과 같으며 여기에 다른 구현 방법이 적용될 수도 있다.

(반디이동) 가로 이동은 기본 자소단위로 옮겨간다. 세로 이동은 현재 줄의 음절 위치와 같은 음절 위치로 이동하며 자소의 위치는 초성자이다.

(삽입) 현재 반디가 있는 곳의 자소의 위치 속성에 따라서 자음은 초성 또는 중성으로 결정된다. 그리고 반디가 놓여 있는 곳에 입력된다. 각 자소의 원소가 3를 초과하거나 글자꼴이 지원되지 않으면 입력되지 않거나 음절 글자 분기가 일어 난다.

(삭제) 현재 반디가 놓여 있는 기본 자소를 지운다. 이 때 종성자가 지워지면 문제가 없다. 중성자가 모두 지워지면 초성자와 종성자 사이에 중성 필코드+초성 및 중성 필코드를 채운다. 그래서 두 자소가 별개의 코드로 표시되게 한다. 초성자가 지워지면 초성 및 중성자 필코드를 채운다.

(대치) 대치는 일대일 대치만 허용한다. 그리고 현재 반디가 놓인 자소의 속성에 따라서 자음의 속성이 결정된다. 속성이 일치하지 않으면 입력이 되지 않는다.

삽입 또는 삭제를 할 때는 그로 인하여 한 음절 글자의 구성이 깨어 지게 된다면 분리해서 표시해야 한다. 예를 들어서 '말'에서 종성자에 반디를 두고'ㄱ'을 추가하면 '맑'이 되지만 만약 'ㄹ'를 추가한다면 '말ㄹ'이 된다. 또한 '말'에서 'ㅏ'를 삭제하였다면 'ㅁ ㄹ'로 표기한다. 다시 말해서 공간에 있어서 두

음절 글자식으로 표현한다.

대치 연산에서 예를 들면 '학'을 '락'으로 잘 못 쳤다고 할 때 음절 연산에서 처럼 음절 글자 '락'을 완전히 지우고 다시 '학'을 입력하지 않고, 'ㅎ'대신에 'ㄹ'만을 대치하면 된다.

4.4 코드와 글자꼴

모든 복자소는 기본 글자의 코드로 표현되기 때문에 훈민정음 해례의 합용병서법에 따라서 두 석자 조합이 가능함으로 여러가지 기본 글자 조합을 입력할 수 있다. 하지만 입력 가능한 글자의 제한은 글자꼴의 유무에 따라서 결정된다. 한 편으로 이것은 표현 범위를 제어할 수 있는 장치로 활용될 수 있다. 예를 들어서 현대 한글 11172 음절 글자를 표기 범위로 하였을 때 초성자 19자, 중성자 21자, 종성자 27자의 글자꼴을 준비하면 된다. 그리고 한글의 창제 이후에 시대별로 표현할 수 있는 글자의 범위의 제한은 이 처럼 글자꼴 집합의 조절을 통하여 할 수 있다.

글자꼴의 종류는 글자의 아름다움을 위하여 여러 벌 제작할 수 있다. 그러나 기본적으로 여덟 벌은 필요하다. 음소 글자 혼자 표기될 경우, 받침없는 가로꼴과 세로꼴 모음이 올 경우 그리고 받침있을 때 두 가지 경우, 복모음일 때 받침 있을 때와 없을 때등이다. 예를 들어서 옛 한글의 '쁘다'에서 'ㅱ'은 세 개의 코드로 표기된다. 하지만 하나의 글자꼴에 대응할 뿐만아니라 중성자의 꼴에 따라서 변하고 또한 받침의 유무에 따라서 그 모양이 또한 달라 진다.

여기서 한글 문자에 대한 새로운 인식이 필요하다. 로마자나 가나문자의 경우에는 코드와 글자꼴이 일대일로 대응된다. 그러나 한글문자의 경우에는 여러 개의 자소가 하나의 글자꼴과 대응된다. 물론 한글 문자의 경우에도 조합형 글자꼴인 경우 한 음절 글자를 이루는 자소 갯수 만큼 낱 자소 글자꼴이 대응되지만 이들은 하나의 글자꼴로 합쳐지게 되어 있다. 결과적으로 자소로 구성된 한 음절에 대하여 갖는 개별적 특성은 로마자 처럼 단순한 글자꼴에 더하여 모아쓰기 규칙이 더 있다. 따라서 한 음절 글자를 구성하는 자소의 문자열은 자소 글자꼴의 집합과 이를 이차원의 구조로 만드는 규칙의 집합에 대응을 이룬다.

. 로마자 코드 (ASCII, ISO 646IRV) -> 로마자 글자꼴
. 자소형(정음형,자모형) 코드 -> [모아쓰기 규칙 + 한글 글자꼴]

그래서 한글문자가 갖는 특성인 음소문자와 음절문자 특성은 내부적으로 표현할 때는 기본적으로 일차원의 음소문자적 특성을 나타내고, 화면에 표시될 때는 한글 글자꼴 앞부분에 있는 모아쓰기 규칙을 통하여 하나의 글자꼴에 대응된다. 따라서 한글 글자꼴는 지식베이스 처럼 사실인 자소 글자꼴와 규칙인 모아쓰기 규칙이 함께 있는 한글 글자꼴 지식베이스를 구성하고 있는 것으로 볼 수 있다[4].

5. 서블의 개발

서블은 음소 문자 및 음절 문자 특성을 함께 구현하는 실험용 기본 입출력 및 편집 도구이다. 이것은 피시 및 유닉스의 엑스 윈도우에서 C++ 언어로 구현하였다.

5.1 코드

서블의 코드는 자소형 또는 정음형을 채택한다. 여기서 현재의 한글 자판 KS C 5715를 존중하여 초성쌍자음과 모음의 ㅒ ㅖ ㅙ ㅞ를 추가하였다. 단수 바이트 코드 배치도는 그림 5.1과 같으며, 이들은 94범위 안에 충분하게 수용될 수 있으므로 한 바이트 코드 체계를 가지며 훈민정음 원리에 따라서 글자꼴이 제공되는 한 399억 음절 글자를 표현할 수 있다.

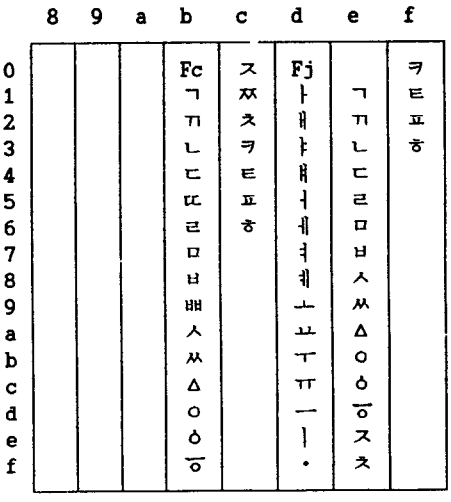


그림 5.1 정음형 코드[6,4]

5.2 입력 방식

현행 자모형(두벌식) 자판에서 간단하게 지금 쓰지 않는 옛 한글 자모 넉자 ㅿ ㆁ ㆆ . 를 그림 5.2와 같이 글쇠 위에 배열하고, 입술가벼운 소리는 ㅁ ㅂ ㅃ ㅍ에 ㅇ을 나열한 복자음으로 보며, ㅇㅇ ㆀ ㅱ, 소등도 마찬 가지로 본다. 자모형 자판에서 옛 한글을 입력

하려면 음절 글자를 구분 짓는 구별자가 필요해서 올림 L(¶)을 배정하였다. 이것은 자소형 자판일 경우에 필요하지 않다.

A	S	Dð	FΔ	Gō	H	J	K .	L ¶	:
ㅁ	ㄴ	ㅇ	ㄹ	ㅎ	ㅗ	ㅓ	ㅏ	ㅣ	;

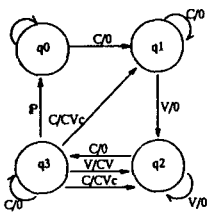
그림 5.2 옛 한글 자판 배열[10,11]

5.3 한글 자동틀

한글 자동틀(automata)은 통상적으로 자판등에서 일차원으로 입력되는 문자열로 부터 음절을 인식하고, 이들을 모아쓰기해 주는 두 가지 복합 기능을 말한다. 셔틀에서 이들은 실제 그 역할이 다르다는 점에서 따로 분리하여 구현하였다.

5.3.1 음절 접수기

접수기(accepter) 오토마톤은 일차원의 한글 자소 코드열이 들어 오면 이 들에 대하여 음절 글자 구성이 되는 지 여부를 판단한다. 접수기 오토마톤은 입력에 대하여 결과가 참(on) 또는 거짓(off) 두 가지만 출력한다. 그리고 코드에는 어떠한 변화도 일으키지 않는다. L은 옛한글 입력시 자모형 자판에서 편의상 음절 글자 구분 제어문자로 사용된다.



(관례) C:자음(초성자),V:모음(중성자),c(종성자)
그림 5.3 자모형 접수기-변환기

구현에서 입력 자판은 자모형 ks c 5715를 사용하고, 내부에 표현될 때는 자소형(정음형)으로 바꾸어 준다. 이 때 음절 구성 여부 인식에는 접수기 Haccepter()가 그리고 코드 변환에는 변화기 HCtransducer()가 함께 적용된다. 프로그래밍에서는 HCaccepter()은 Haccepter()에 포함되어 있어서 Haccepter()은 오토마타 성격을 띤다.

5.3.2 글자꼴 모아쓰기(transducer)

글자꼴 모아쓰기 오토마톤 또는 변환기 오토마톤은 일차원의 문자열에 대하여 한 음절 글자에 해당하는 이차원 구조로 구성하는 역할을 한다. 다시 말해서 입력되는 일차원 문자열을 이차원 구조의 음절 글자꼴로 바꾸어 주는 역할을 한다. 그래서 입력문자열은 이론적으로 출력문자열로 되면서 일종의 위상을 가진 이차원 형태로 변경된다.

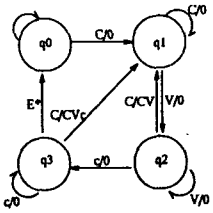


그림 5.4 자소형 변환기

5.4 음소 단위 연산

음소단위 연산은 한글 문자가 음소문자 특성을 가지고 있음을 나타내는 것으로 매우 편리한 기능이다. 물론 이들의 구현은 그래픽 화면 장치에서 가능하다. 음소 연산은 로마자 연산과 거의 같다. 편집기의 경우 내부 문서 반디는 자소 단위 즉 개별 낱자 코드 단위로 이동한다. 그리고 대응하는 화면 반디는 화면 위에서 이차원으로 구성되어 있는 한 음절 글자에서 해당 자소의 색상을 좀 짙게 하거나 다른 색상으로 하여 표시한다. 복자소인 경우 반디가 다 지날 때 까지 계속 머문다. 이것은 조합형 글자꼴일 때 쉽게 구현 가능한 데 초성 글자꼴을 다른 색상으로 빛 속성을 준 다음 나머지 글자꼴과 합성하도록 하였다. 이러한 표시 방법으로 4장에서 언급한 대치, 또는 삽입, 삭제등의 자소 단위 연산이 쉽게 이루어 질 수 있다.

또한 여기서 중요한 사실은 셔틀은 훈민정음 해례가 정의한 399억 글자를 생성할 수 있으며 현재 다 표현할 수 없는 것은 글자꼴 베이스가 충분하지 않기 때문이다. 다시 말해서 셔틀로 현대 한글만을 표현하려면 현대 한글의 글자꼴인 초성자 19자, 중성자 21자, 종성자 27자로 가능하다.

6. 결론

한글 문자는 문자론적 특성상 음소문자와 음절문자의 특성을 동시에 가지고 있음을 보았다. 이러한 특성들이 이제 까지 한글 코드를 제정하는 과정에서 어떻게 구현 또는 적용되어 왔는 지를 분석해 본 결과 이와 같은 인식의 부족으로 경우에 따라서 두 가지 속성 가운데 어느 한 쪽에 치우쳐 왔음이 판명된다. 음

절 문자 특성의 구현은 일단 한글을 표시한다는 목적에는 부합하지만 한국어 정보처리 전체를 감당하지 못함에 문제가 있다.

본 논문에서 주장하는 문자론적 개별 속성은 셔틀을 통하여 실험한 결과로 볼 때 훈민정음 해례에서 제시한 조건 제시법을 충실하게 구현함으로써 가능함을 검증할 수 있었다. 즉 일차원의 기본 자소 코드로써 내부 표현을 하고 화면에 보일 필요가 있을 때 해례의 합용법 및 부서법등을 적용하는 연산을 통하여 음절 문자 특성을 구현하였다. 여기서 규칙은 연산을 통하여 구현할 때 코드 집합의 크기를 작게 하며 음소 문자 특성을 원활하게 구현할 수 있다. 이것은 모두 훈민정음 해례의 충실한 구현에 기인한다고 결론 지을 수 있다. 다른 한 편으로 보면 본 연구의 핵심 알고리즘과 구현 원리는 세종임금과 집현전 학사들이 모두 만드셨고 본 연구는 그러한 사실을 이해하고 코딩을 하는 노력을 들인 것에 불과하다 감히 말할 수 있다.

과학이 정확한 만큼 공학은 단순화될 수 있을 것이다. 공학이 과학과 도구 사이에 존재한다고 본다면 아직도 우리는 훈민정음 원리의 깊은 뜻을 계속적인 연구를 통하여 규명하고 컴퓨터라는 도구의 속성과 결부하여 더욱 열심히 다듬어야 할 필요가 있다. 또한 과학과 공학의 힘으로 훈민정음의 원리가 적용되어 컴퓨터 위에서 현재 격고있는 사고의 제약을 완전하게 해결할 수 있도록 노력해야 할 것이다.

참고문헌

[1] 공업진흥청, KS C 5620-1977, KS C 5601-1974,1982,1987, KS C 5657-1989

[2] 변정용, "한국어정보처리를 위한 최적한 한글부호계", 제1회 한글 및 한국어 정보처리 학술 논문집, 한국정보과학회.한국인지과학회, 39~43, 1989

[3] 강신항, 훈민정음 연구, 성균관 대학교 출판부, 95~143, 1991

[4] 변정용, "훈민정음 원리의 공학화에 기반한 한글 코드의 발전방향", 한국정보과학회지, 제12권 8호, 72~88, 1994년 9월

[5] 정희성, "수학적 구조로 본 훈민정음의 창제 원리와 구조", 제1회 한글 및 한국어정보처리 학술 논문집, 174~177, 1989

[6] 변정용, "훈민정음 창제 원리와 한글코드 제정 원리", 제3회 한글 및 한국어 정보처리 학술 논문집, 155-158, 1991

[7] The Unicode Consortium,The Unicode Standard Worldwide Character Encoding, Ver. 1.0, Vol.1, 1991

[8] 이승호외 9인, 단일문자 표준 연구, 한국전산원, 1993

[9] 변정용, "한글 코드의 개선과 알고리즘의 개발: 낱자형 33자 코드", 동국대학교 논문집 제8집, 421~433, 1989

[10] 변정용, "정음형 코드의 저장효율 평가와 문자열 정렬 알고리즘", SC2/WG2, 1992

[11] 변정용.임해철, "한글자료의 표현과 세벌식 한글 낱자형 부호계 연구", '91 가을 학술 발표 논문집, 한국정보과학회, Vol.18, No.2, 829~832, 1991

[12] 서병국, 신강 훈민정음, 학문사, 1976

[13] 황종선, 임해창, 국제 표준 ISO/IEC DIS 10646-1.2:1992, 국제공용 글자판,공업진흥청, 1992

[14] 이균하, "한글의 열린 조합과 이를 위한 오토마타", 제3회 한글 및 한국어정보처리 학술발표논문집, 한국인지과학회.한국정보과학회, 159~167, 1991

[15] 정희성, "세종머쉰과 투링머쉰의 등가성에 관하여", 제1회 한글 및 한국어 정보처리 학술발표 논문집, 한국인지고학회.한국정보고학회, 21~28, 1989

[16] SC2 K055, DIS 10646에 대한 SC2의 공식 입장 보고, JTC1/K375, 1992