등산법을 이용한 한국어 맞춤법 교정기의 분석

윤근수*

The analysis of Korean Spelling Corrector using Hill-Climbing Method

Keun-Soo Yun*

요 약

최적 교정률을 보이는 모듈 순서를 찾는 것이 논문의 목적이다. 한국어 맞춤법 교정기의 성능을 분석하기 위하여 등산법을 실험에서 사용하였다. 주어진 오류어절 집합에 대하여 96.41%의 교정률을 나타낸 모듈순서열을 찾았다. 교정률이 상당히 높기때문에, 등산법이 맞춤법 교정기에 대하여 실용적인 방법임을 보였다.

ABSTRACT

To find the module sequence that makes correction rate optimal is the goal of this paper. The Hill-climbing algorithm was used in the experiment to analyze the performance of Korean Spelling Corrector. Given the wrong eojul set, We found the module sequence that shows correction rate of 96.41%. Because of the quite high correction rate, Hill-climbing is a practical method for our Spelling Corrector.

키워드

Korean spelling corrector, Module sequence, Hill-Climbing algorithm, Optimization
한국어철자교정기, 모듈순서, 등산법, 최적화

Ⅰ. 서 론

인터넷의 활성화와 함께 많은 정보가 문서를 통해 가상공간을 흘러 다니고 있으며, 신문사와 방송국을 비롯한 각종 언론 매체는 매일 엄청난 정보를 생산해 내고 있다. 이와 같은 문서들을 일일이 수작업으로 교정하기란 너무 많은 데이터로 인해 시간과 비용 측면에서 현실적으로 불가능하다. 이에 따라 교정의 자동화가 필요해 졌으며, 외국에서는 상당한 수준으로 발전하였다.

우리나라에서도 1980년대를 기점으로 한국어 정보처리가 상당히 진척되었다. 형태소분석기, 맞춤법 검사기, 맞춤법 교정기가 현재 개발되어 신문, 방송 등에 활용되고 있다[1][2][3][4]. 초기의 한국어 맞춤법 교정기는 교정성능이 낮았고 소규모였다. 근래에는 방대한 자료가 수집되어 고유명사 등을 제외한 대부분의 오류는 교정된다.

먼저 2장에서는 한국어 맞춤법 교정기와 사용된 오류어절 집합 및 개발환경에 대한 소개를 살펴본다. 3장에서는 등산법을 적용한 실험을 하며, 4장에서는 실험에 대한 분석을 하였으며, 5장에서는 결론을 제시하였다.

* 울산과학대학교 컴퓨터정보학부(ksyun@uc.ac.kr)
접수일자 : 2012. 07. 18 심사(수정)일자 : 2012. 07. 26 게재확정일자 : 2012. 08. 09

II. 맞춤법 교정기

2.1. 맞춤법 교정기 소개

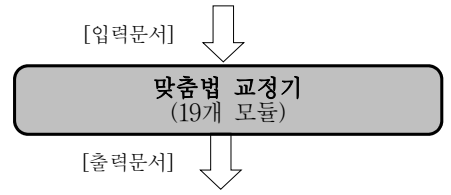
맞춤법교정기는 형태소 분석기로부터 출력을 받아 입력으로 사용한다. 입력되는 어절가운데 먼저 맞춤법 검사기가 작동되어 정확하다고 판단한 입력어절은 통과하게 되고, 오류가 발생하였다고 판단되는 어절은 선택되어 맞춤법 교정의 입력이 되어 교정작업을 거치게 된다. 그림 1에서 입력오류어절은 맞춤법 검사기에서 오류가 있다고 판정된 어절집합이다[5][6][7][8].

맞춤법 검사기는 어절을 구성하는 형태소의 품사 특성에 따라 모듈별로 구분되어 처리되며 명사, 대명사, 수사, 부사, 조사, 어미, 동사, 형용사, 준말, 단독 사용 어절부로 나누어져 있다[9]. 맞춤법 검사기는 형태소 오토마타를 따라 가면서 어절을 검증하는데 이것은 형태소 분석기의 역과정을 추적하는 것이다. 이 검증 과정에서는 사전 정보에 따라 한 어절 오토마타에서 수렴 여부를 시도해 보고 수렴되는 경로가 있으면 그 경로를 저장하고 종료한다. 형태소 오토마타에서 수렴이 되었지만 모호성을 가지는 특별한 오류는 오류 유형 추정 함수에서 별도로 오류 여부를 판정한다

오류 유형 추정을 위해서는 가능한 한 정확한 정보가 필요하므로 맞춤법 검사 과정에서 발생한 정보를 저장하여야 한다. 예를 들면, 띄어쓰기 후보 위치, 옳은 형태소라고 판단되는 위치 등을 오류 검사 과정에서 보관하여 교정기에게 넘겨주어야 한다.

맞춤법 교정기는 오류 어절을 입력으로 받아 오류 원인에 따라 띄어쓰기 오류, 철자 오류, 형태소간 결합 오류 등으로 나누어 앞 단계에서 넘어온 정보를 이용하여 교정한다. 맞춤법 교정기의 개요를 그림 1에 나타내었다.

입력오류어절	정답어절(수작업추가)
0%의시장점유율을	0%의 시장점유율을
0%의인구증가율을	0%의 인구증가율을
000가구등이	000가구 등이
000만워대	000만 워 대
•••••••	
성민엽처럼나	성민엽처럼 나
성바오로병원등	성바오로병원 등
성베드로병원이후	성베드로병원 이후
•••••••	
힘있을때	힘있을 때
힙합춤추기부터	힙합 춤추기부터
힝기스간	힝기스 간
힝기스누르고	힝기스 누르고



교정정보	교정어절
모듈c, 교정성공	0%의 시장점유율을
모듈c, 교정성공	0%의 인구증가율을
모듈c, 교정성공	000가구 등이
모듈0, 교정실패	
•••••••	
모듈l, 교정성공	성민엽처럼 나
모듈c, 교정성공	성바오로병원 등
모듈f, 교정성공	성베드로병원 이후
•••••••	
모듈k, 교정실패	힘 있을 때
모듈k, 교정성공	힙합 춤추기부터
모듈h, 교정성공	힝기스 간
모듈i, 교정성공	힝기스 누르고

그림 1. 맞춤법교정기 개요도
Fig. 1 Outline of spelling corrector

2.2. 입력과 출력의 구조

입력 문서의 엔트리 구성은

[오류어절 | 수작업으로 교정한 정답어절]

로 구성되어 있다. 여기서 정답어절은 교정기에 입력되지 않고 교정기가 제시하는 교정어절과 비교하기 위해 사용된다.

출력 문서의 엔트리 구성은

[교정정보 | 교정한 어절]

로 구성되어 있다. 교정기가 교정을 하지 못한 경우에는 [교정된 어절]부분을 비워두게 된다. 교정정보는 교정어절을 제시한 모듈 명과 교정에 성공하였는지 실패하였는지에 대한 정보를 보여준다. 예를 들어, 출력 문서 중에서 “모듈c,교정성공 |0%의 시장점유율을”는 모듈 M_c는 입력 오류어절에 대하여 교정 성공하였다는 정보를 제공하고 있다. 또한 “모듈k,교정실패 | 힘 있을 때”는 모듈 M_k에서 교정 어절을 제시하였으나 정답 어절과 다르므로 교정에 실패하였다는 의미이다. 그리고 입력어절 “000만워대”에 대한 출력어절 “모듈0, 교정실패 | 000만워대”인 경우는 모든 모듈에서 교정어절을 제시하지 못하여 교정실패로 분류된다.

III. 실 험

윈도우즈 XP환경에서 VC++.NET으로 실험하였으며 실험에 사용한 입력오류 어절집합은 신문사에서 다년간 발생한 오류어절을 수집한 것이다. 엔트리 수는 n(입력 오류 어절 집합) = 753,191개 어절이다. 모든 엔트리는 한 어절로 구성되어 있으며 한 어절에 대해 여러 번 띄어쓰기도 처리한다. 또한 한 오류어절에 대해 복수개의 정답어절인 경우도 포함되어 있다.

3.1. 모듈간 병렬처리 결과

모듈간의 임의 배치에 따른 최대예상 교정률과 최소예상 교정률을 구하여 교정률이 취할 수 있는 범위를 구하고자 한다. 표 1은 입력 오류 어절 집합에 대하여 각 모듈을 독립적으로 실행하여 얻은 결과를 나타낸다. 여기서 입력집합은 형태소 분석에 실패하여 맞춤법 검사기에서 오류 어절이라고 판단된 어절들로, 교정을 필요로 하는 전체 오류 어절 집합이다.

이 실험에서 정답어절 집합에서 중복되는 부분을 모두 제외하였을 때, 각 모듈이 정답을 제시한 O_i에 대한 각각의 합은 식 (1)과 같다.

$$\mathrm{n}(O) = \mathrm{n}\left(\bigcup_{i=1}^{N} O_i\right) = \mathrm{n}\left(\bigcup_{i=1}^{N-1} O_i\right) + \mathrm{n}(O_N) - \mathrm{n}\left(\left(\bigcup_{i=1}^{N-1} O_i\right) \bigcap O_N\right) = 733{,}696\text{개} \quad (1)$$

이 숫자의 의미는 입력집합 U(753,191개)에 대한 맞춤법 교정기가 교정할 수 있는 최대 교정 어절 개수는 733,696 개를 넘지 못한다는 것이다. 따라서 n(“)는 “최대예상 교정어절 수”를 나타낸다.

마찬가지로,

$$\mathrm{n}(X) = \mathrm{n}\left(\bigcup_{i=1}^{N} X_i\right) = \mathrm{n}\left(\bigcup_{i=1}^{N-1} X_i\right) + \mathrm{n}(X_N) - \mathrm{n}\left(\left(\bigcup_{i=1}^{N-1} X_i\right) \bigcap X_N\right) = 196{,}528 \quad (2)$$

식 (2)는 시스템의 모든 모듈에 의해서 한 번 이상 틀린 어절로 선택될 수 있는 수치이다.

표 1. 오류집합에 대한 모듈들의 독립 실행
Table 1. Independently module execution for wrong set

모듈 M_i	O_i(정답어절)	X_i(오답어절)	P_i(통과어절)
M_a	2,347 (82.55%)	496 (17.45%)	750,348 (99.62%)
M_b	27,839 (92.40%)	2,289 (7.60%)	723,063 (96.00%)
M_c	183,115 (96.46%)	6,713 (3.54%)	563,363 (74.80%)
M_d	0 (0.00%)	0 (0.00%)	753,191 (100.0%)
M_e	210 (28.04%)	539 (71.96%)	752,442 (99.90%)
M_f	19,999 (97.64%)	484 (2.36%)	732,708 (97.28%)
M_g	1,861 (97.13%)	55 (2.87%)	751,275 (99.75%)
M_h	125,661 (96.07%)	5,138 (3.93%)	622,392 (82.63%)
M_i	16,436 (89.40%)	1,949 (10.60%)	734,806 (97.56%)
M_j	117,944 (91.92%)	10,367 (8.08%)	624,880 (82.96%)
M_k	129,500 (89.39%)	15,367 (10.61%)	608,324 (80.77%)
M_l	61,661 (90.55%)	6,436 (9.45%)	685,094 (90.96%)
M_m	68 (0.53%)	12,745 (99.47%)	740,378 (98.30%)
M_n	82 (2.18%)	3,677 (97.82%)	749,432 (99.50%)
M_o	10 (0.03%)	35,038 (99.97%)	718,143 (95.35%)
M_p	498 (0.50%)	98,135 (99.50%)	654,558 (86.90%)
M_1	383,911 (98.60%)	5,434 (1.39%)	363,846 (48.30%)
M_2	9 (75.00%)	3 (25.00%)	753,179 (99.99%)
M_3	222,584 (88.91%)	27,758 (11.09%)	502,849 (66.76%)
	$\sum_{i=1}^{N} \mathrm{n}(O_i)$ =1,293,735	$\sum_{i=1}^{N} \mathrm{n}(X_i)$ =232,623	

다음으로 정답집합의 어절개수는 $\mathrm{n}(O)$ = 733,696 이며, 오답집합의 어절개수는 $\mathrm{n}(X)$ = 196,528이었다. 또한 정답집합 O와 오답집합 X의 교집합 $(O \cap X)$에 대한 어절개수는 $\mathrm{n}(O \cap X)$ = 178,409 이었다. 따라서 정답집합과 오답집합에 대한 차집합의 원소개수는 식 (3)과 같다.

$$\mathrm{n}(O-X) = \mathrm{n}(O - (O \cap X)) = \mathrm{n}(O) - \mathrm{n}(O \cap X) = \mathrm{n}\left(\bigcup_{i=1}^{N} O_i - \bigcup_{i=1}^{N} X_i\right) = \mathrm{n}\left(\bigcup_{i=1}^{N} O_i\right) - \mathrm{n}\left(\bigcup_{i=1}^{N} O_i \bigcap \bigcup_{i=1}^{N} X_i\right) = 733{,}696 - 178{,}409 = 555{,}287 \quad (3)$$

따라서 $\mathrm{n}(O - X)$은 맞춤법 교정기의 “최소교정 어절수”를 나타낸다.

위로부터 “최대예상 교정률 $\mathrm{CR}_{\mathrm{upper}}$”과 “최소예상 교정률 $\mathrm{CR}_{\mathrm{lower}}$”을 유도하면 식(4), 식(5)와 같다 :

CR_{upper} = n(O) / n(U) *100 %
= 733,696 / 753,191 *100 % = 97.41% (4)

CR_{lower} = n($O-X$) / n(U) *100 %
. = 555,287 / 753,191 *100 % = 73.72% (5)

식(4)와 식(5)로부터 교정률의 범위는 식(6)과 같음을 알 수 있다.

CR_{lower} <= CR <= CR_{upper} (6)

최대예상 교정률과 최소예상 교정률은 시스템의 모듈들을 가장 이상적으로 배치하였을 때의 상한과 최악의 모듈 배치를 하였을 때의 값인 하한에 해당한다. 이것은 모듈의 순서를 아무렇게나 배치하여도 교정률은 상한과 하한 사이에 존재함을 의미한다. 그림 2는 최대예상 교정과 최소예상 교정에 대한 개념을 표현한 것이며, 모듈 실행순서 조율을 통하여 n($O \cap X$) = 178,409를 최소화하여 교정성공률을 높여야 한다.

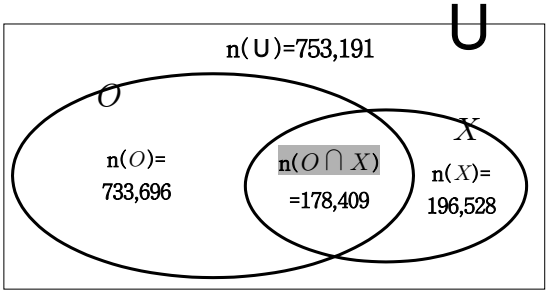


그림 2. 정답집합과 오답집합의 관련성
Fig. 2 Relationship of right answer set and wrong answer set

3.2. 등산법을 이용한 교정률 계산

앞 절에서처럼 모듈을 병렬 처리한 다음 각 모듈에서 나온 결과를 비교 검토하여 옳은 답을 선택하는 방법은 너무 많은 시스템 자원과 시간을 요구하므로 현실적으로 사용할 수 없기 때문에 그 대안으로 맞춤법 교정기의 모듈들을 순차 처리하는 방법을 사용한다. 그림 2에서 교정률에 영향을 미치는 교집합 **n($O \cap X$)** =178,409에 대해서 경험적 지식을 사용하는 Hill Climbing을 통하여 교정률을 개선시키는 실험을 하였다. 맞춤법 교정기를 구성하고 있는 모듈들 간의 실행순서에 의해 교정률이 다르게 나올 수 있으므로, 이들 모듈들 간에 서로 교정률에 영향을 미치는 부분을 분석하여 모듈 간 순서를 조정해 줄 필요가 있다.

표 2는 등산법(Hill-Climbing)탐색[10]을 시작하기 전 초기상태에서 정답집합과 오답집합 간의 가로채기 당하는 어절 개수를 보여준다. 매 단계마다 이 가로채기 당하는 어절 개수는 달라진다. 탐색 상태가 다음 상태로 바뀔 때마다 이와 같은 표를 재작성하는 과정이 필요하다.

그림 3, 그림 4, 그림 5, 그림 6은 표 2에서와 같이 각 상태에서 최대 교정률을 보이는 모듈을 선택해 나가는 과정을 보여준다. 표 2에서 진한 셀과 각 그림이 대응하고 있다. 전체 19개 단계 중에서 지면상 4단계만을 나타내었다.

표 2에서 휴리스틱 정보를 이용하여 근사 최적 교정률을 선택하는 과정으로 작업의 진행 방향은 상태 S_1 ~ 상태 S_{16}순서로 실행이 되며 각 상태(state)에서 선택된 모듈은 다음 상태에서 제외가 된다. 각 셀에는 3가지 숫자가 있으며, 첫 번째 숫자는 정답어절 개수를 나타내고, 두 번째 숫자는 오답어절 개수이고, 세 번째 숫자는 그 모듈의 교정률을 의미한다. 예를 들어 상태 S_1에서 모듈 M_a~M_p의 16개 모듈 중 모듈 M_g의 교정률이 제일 높다. 즉, $n(O_g)$=1,867, $n(X_g)$=58, f(M_g) = 96.99%임을 알 수 있다.

표 2. 등산법 탐색중 일부
Table 2. Part of hill-climbing search

state	S_1	S_2	S_3	S_4
입력어절	363834	361909	172163	158862
M_a	2427 448 84.42	2423 445 84.48	1753 254 87.34	1719 236 87.93
M_b	28998 2308 92.63	28998 2308 92.63	14642 1582 90.25	14043 1516 90.26
M_c	182414 8007 95.80	**181789** **7957** **95.81**		
M_d	0 0 0.00	0 0 0.00	0 0 0.00	0 0 0.00

M_e	217 592 26.82	217 587 26.99	214 521 29.12	214 502 29.89
M_f	20405 1097 94.90	20348 1095 94.89	**12420** **881** **93.38**	
M_g	**1867** **58** **96.99**			
M_h	124779 5539 95.75	124682 5518 95.76	40526 5100 88.82	**40112** **1089** **97.36**
M_i	16465 2243 88.01	16248 2234 87.91	11878 1634 87.91	10772 1545 87.46
M_j	97371 8463 92.00	97356 8422 92.04	32787 6835 82.75	31378 3329 90.41
M_k	105218 14511 87.88	104445 14406 87.88	61359 10617 85.25	54247 10076 84.34
M_l	232040 24099 90.59	230670 23846 90.63	102047 17239 85.55	94972 14966 86.39
M_m	72 12596 0.57	72 12573 0.57	59 9412 0.62	59 9082 0.65
M_n	97 4283 2.21	97 4264 2.22	87 2875 2.94	87 2535 3.32
M_o	10 34831 0.03	10 34822 0.03	10 6581 0.15	10 5230 0.19
M_p	511 98291 0.52	511 97595 0.52	495 48802 1.00	495 40780 1.20

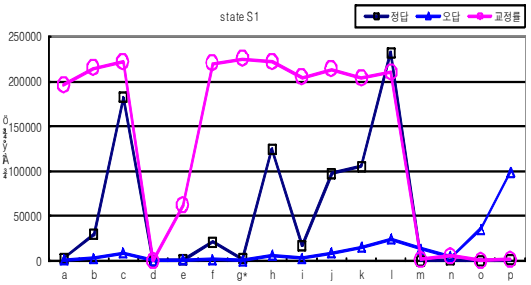


그림 3. 모듈 M_g를 선택
Fig. 3 Selection of module M_g

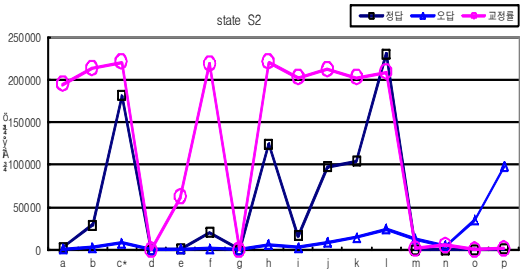


그림 4. 모듈 M_c를 선택
Fig. 4 Selection of module M_c

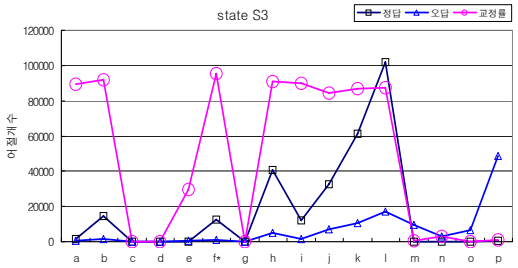


그림 5. 모듈 M_f를 선택
Fig. 5 Selection of module M_f

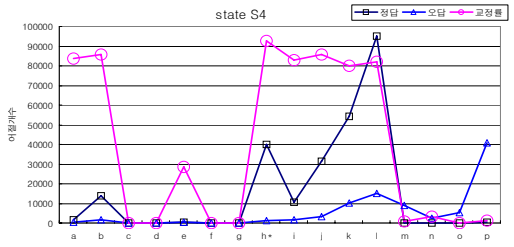


그림 6. 모듈 M_h를 선택
Fig. 6 Selection of module M_h

각상태 i에서 비용평가 함수는 식(7)을 사용하였다.

$$f(i)= \frac{n(O_i)}{n(O_i)+n(X_i)} \times 100\% \quad (7)$$

현재 상태에서 다음 상태로 나아갈 때 평가 함수는 최대 교정률을 사용하였다. 따라서 비용 평가 함수에 따라 다음과 같은 모듈 순서열을 찾을 수 있었다. 이 모듈들은 표 2에서 진한 글자로 되어 있으며 좌측에서 우측으로 진행하는 순서로 되어있다. 이렇게 구해진 모듈 순서열에 의해 전체 입력 어절 753,191 개 중

에 726,178 개를 올바르게 교정하였으며, 25,637 개 어절은 교정에 실패하였다. 따라서 등산법에서 구해진 맞춤법 교정기의 교정률은 726,178 / (726,178+25,637+1,376) * 100% = 96.41% 이었다. 등산법 탐색에 의해 구한 모듈순서는 식(8)과 같다:

$$M_g > M_c > M_f > M_h > M_j > M_i > M_k > M_a > M_l > M_b > M_e > M_n > M_m > M_p > M_o > M_d \quad (8)$$

한편, 한국어 맞춤법 교정기는 19개 모듈로 구성되어 있으며 그 중에서 다른 모듈에 영향을 거의 주지 않거나 교정 방법이 확실한 오류 교정 모듈은 미리 처리하거나 나중에 처리하는 것이 계산상 유리하므로 3개의 모듈은 등산법 대상에서 제외하였다.

모듈 M_1은 오용어를 교정하는 모듈로, '오얏 > 자두', '노가다 > 노동자', '까탈스럽다 > 까다롭다' 등처럼 교정 대상이 확실한 경우에도 사전에 저장을 해놓았다가 오류 어절을 만나면 대치어를 내보내게 되므로 다른 모듈에 영향을 주지 않는다.

마찬가지로 모듈 M_2는 실제 문서상에는 사용되지 않는 음절을 처리하는 모듈로, '것네 > 겠네', '읍니 > 습니' 등처럼 오류가 확실한 경우는 다른 모듈보다 먼저 처리하도록 하였다. 위의 두 모듈은 다른 모듈보다 앞서 처리하도록 하였다.

모듈 M_3은 다른 모든 모듈을 이용하여 교정해본 후 어떤 모듈에서도 교정을 하지 못 했을 경우 최종적으로 교정을 시도하게 했다. 이 모듈은 bi-gram 통계를 기반으로 띄어 쓸 확률이 매우 높은 음절 쌍 사이를 무조건 띄어쓰기 한다. 예를 들면, 오류 어절 '효율화는풀기어려운숙제로'이 주어졌을 때 '-는풀-'사이는 붙어서 사용되는 어절을 코퍼스에서 찾을 수 없으므로 띄어 쓸 확률이 매우 높다.

한편, 이렇게 별도로 배치하였을 경우 교정을 정확하게 한 어절 개수는 모듈 M_2는 383,911개, 모듈 M_1은 9개, 모듈 M_3은 3,073개인 반면에 교정을 하지 못한 어절 개수는 모듈 M_1는 5,434개, 모듈 M_2은 3개, 모듈 M_3은 698개였다. 교정을 못한 것 중에는 시스템 오류가 상당한 부분을 차지하였으므로 이들 3개 모듈에 대해서는 순서를 고정하는 것이 바람직하다.

먼저 처리되어야 할 2개 모듈(무조건 대치, 오용어)과 가장 나중에 처리되어야 하는 1개 모듈(무조건 띄어쓰기)을 제외하고 나면 고려하여야 할 모듈은 16개이므로 이들 모듈의 순열(16!)만 고려하면 되므로 문제 공간이 많이 줄어들었다.

Ⅳ. 결과 분석

한국어 맞춤법 교정기의 19개 모듈 모두가 1,376개 어절에 대해서 교정 어절을 제시하지 못하였으며 교정 어절을 제시하지 못한 원인은 대부분 다음과 같았다:

- 미등록어로 인한 교정 실패
  - "박은정이화여대법대교수는"와 같은 어절에서 사람이름 미등록어
- 오류 어절에 대한 정답 자체가 잘못 제공된 경우
  - 오류어절 "000만워대"의 정답어절이 "000만 워대"인 예
  - 오류어절 "1백메가디렘이"의 정답어절이 "1백 메가디 렘이"인 예
  - 오류어절 "아테나이오스는"의 정답어절이 "아테나이오 스는"인 예
- 형태소 분석 실패 오류 어절
  - 예) "는16대", "는386세대도", "들여해외여행을"
- 철자 교정 실패 오류 어절
  - 예) "말했7다", "100만뭔", "1종대형면서>1종 대형면허서", "10개경기장에스는> 10개 경기장에서는"

실험에 사용된 오류어절의 입력파일 크기는 20M 바이트 (753,191개 어절)이며, 각 단계의 모듈 실행순서를 파일에 미리 저장하여 한꺼번에 배치 처리하였다. 병렬처리단계에서 생성된 파일은 19(개/모듈)이며 각 파일의 크기는 33MB이므로 사용된 공간은 33M * 19개 모듈 = 627MB이였다. 이 단계에서 걸린 시간은 약 2시간 소요되었다.

등산법 처리단계에서 사용된 공간은 현재 상태에서 다음 상태로 진행할 때 비용 평가 함수를 구해야 하므로 각 상태에서 모든 모듈에 대해 계산을 하여야 한다. 각 상태에서 나오는 파일 크기는 평균 33M 바

이트였고, 136번의 평가 함수를 실행하였으므로 33M * 136회 = 4.22G 바이트가 사용되었다. 이 단계에서 걸린 시간은 1회당 평균 10분 정도였으며, (10분 * 136회) / 60분 = 약 22시간(1일) 소요되었다.

따라서 이 실험에 사용된 디스크 공간은 약 4.84G 바이트 (627M + 4.22G) 이었으며, 시간은 약 1일 (2+22 = 24시간) 정도가 소요되었다. 표 3은 실험에 사용된 시간과 공간을 요약한 것이다.

표 3. 실험에 사용된 시간과 공간
Table 3. Time and space used in experiment

구분	소요 시간	디스크 공간
병렬처리	2	627M
등산법	22	4.22G
계	24시간	4.84G

V. 결 론

본 논문에서는 주어진 오류어절 집합에 대하여 등산법 알고리즘을 사용한 맞춤법 교정기를 실험하였다. 실험결과 사용시간과 하드디스크 사용공간 측면에서 실용성이 있으며 입력오류어절 집합을 적절히 커버함을 보였다. 따라서 본 실험결과는 최상의 결과는 아니지만 빠른 응답이 필요한 경우에는 유용한 방법임을 알 수 있다.

실험에서 찾아낸 맞춤법 교정기의 교정률은 726,178 / (726,178+25,637+1,376) * 100% = 96.41% 이었으며 이 때의 모듈순서 열은 다음과 같다: M_g, M_c, M_f, M_h, M_j, M_i, M_k, M_a, M_l, M_b, M_e, M_n, M_m, M_p, M_o, M_d

현재 상태에서 선택할 수 있는 모듈들에 대해 교정률을 구한 후 최대값을 보인 모듈을 선택하는 평가 함수를 사용하였다. 실험에 사용된 교정기의 탐색 공간을 줄이기 위해 다른 모듈과 성격이 달라 절대 교정 기법으로 처리 할 수 있는 3개의 모듈은 제외하고, 나머지 16개 모듈에 대하여 등산법 탐색을 적용하였다. 실험에서 제외한 모듈들은 사전에 기반을 둔 오용어 교정 모듈, 문서에서 반드시 띄어 써야 하는 어절을 처리하는 모듈, 그리고 무조건 대치하는 모듈이었다.

문제점으로는 실험에 사용된 입력어절 집합에 따라 교정률이 달라질 수 있으며 채집한 데이터가 신문사 중심이라는 한계성이 있다. 또한 모듈 간 순차 처리는 입력 데이터에 대해 모듈 순서열을 다르게 하여 실행하기 때문에 교정률이 매번 달라질 수 있다[11].

참고 문헌

[1] 강재우, “접속정보를 이용한 한국어 철자 띄어쓰기 검사기의 설계 및 구현”, 한국과학기술원 전산학과 석사 학위 논문, 1990.

[2] 박종만, “효율적인 한국어 형태소분석기 및 철자 검사교정기의 구현,” 서울대학교 석사학위 논문, 1990.

[3] 심광섭, “음절 간 상호정보를 이용한 한국어 자동띄어쓰기”, 정보과학회논문지(B), 23권, 9호, pp. 991-1000, 1996.

[4] 강승식, 장병탁, “음절특성을 이용한 범용 한국어 형태소 분석기 및 맞춤법 검사기”, 정보과학회 논문지, 23권, 5호, pp. 530-539, 1996.

[5] 김덕봉, 최기선, 강재우, “한국어 형태소와 사전-접속 정보를 이용한 한글 철자 및 띄어쓰기 검사기”, 언어연구, 26권, 1호, pp. 87-113, 1990.

[6] 이병훈, 윤준태, 송만석, “말뭉치를 기반으로 한 한국어 철자교정기의 구현”, 한글 및 한국어 정보처리 학술발표논문집, pp. 285-293, 1993.

[7] 정한민, 이근배, 이종혁, “자판특성을 이용한 Neuro-Fuzzy 한국어 철자교정기의 구현”, 한글 및 한국어 정보처리 학술발표논문집, pp. 317-328, 1993.

[8] 이원일, 홍남희, 이종혁, 이근배, “Binary n-gram과 형태소 분석기를 이용한 한국어 철자 교정기”, ’93 KISS 학술발표 논문집, pp. 813-816, 1993.

[9] 이도길, 이상주, 임희석, 임해창, “한글 문장의 자동 띄어쓰기를 위한 두 가지 통계적 모델“, 정보과학회 논문지: 소프트웨어 및 응용, 30권 4호, pp. 358-370, 2003.

[10] Russell, Stuart J ; Nonvig, Peter, "Artificial Intelligence: A Modern Approach(2nd ed)", Upper Saddle River, New Jersey: Prentice Hall, pp. 111-114, 2003.

[11] 윤근수, “한국어 맞춤법 교정기의 문제점”, 한국전자통신학회논문지, 4권, 2호, pp. 405-408, 2010.

저자 소개

윤근수(Keun-Soo Yun)

1989년 2월 부산대 전산통계학과 학사
1991년 2월 부산대 전자계산학과 석사
2006년 8월 부산대 전자계산학과 박사
현재 울산과학대학교 컴퓨터정보학부 부교수
※ 관심분야 : 한국어정보처리, 패턴인식