20163170 컴퓨터공학부 최은주 정보검색과 데이터마이닝 클러스터링 알고리즘 구현

저는 이번 클러스터링 알고리즘 과제 그 중에서도 slink 알고리즘을 사용하여 구현하였습니다.

slink알고리즘을 통해 문서간의 클러스터를 만들기 위해 union-find 자료구조를 사용하였는데, union-find는 상호 배타적인 부분집합들로 나눠진 원소들을 조작하는 자료구조로서, 문서번호 간에는 공통원소가 없으며, 같은 클러스터에 속하는 문서들을 집합으로 표현하기에 알맞다고 생각하여 이 알고리즘을 사용하게 되었습니다.

주요 연산으로는 UNION 연산과 FIND연산이 있는데

union연산은 두 원소 혹은 두 원소가 속한 집합을 하나로 합치는 연산이고,

find연산은 주어진 원소가 속한 집합을 반환하는 연산입니다.

아래는 추가된 코드만 캡쳐하였습니다. (그 외 코드는 이전 과제와 동일)

```
 9  int parent[623];                    //find, merge를 통해 값을 바꿔주기 위해 전역변수로 각 문서가 속한 집합을 담을 배열을 선언
10  int find(int u){                    //주어진 원소가 속한 집합을 반환하는 find 연산
11      if(u == parent[u]) return u;    //만약 원소 u가 루트이면 자기 자신을 반환
12      else{
13          int y = find(parent[u]);    //그렇지 않다면 Y라는 변수에 u의 부모의 부모를 찾아
14          parent[u] = y;              //U의 부모를 y로 바꿔주고
15          return y;                   //y를 반환
16      }
17  }
18  void merge(int u, int v){           //두 원소 혹은 두 원소가 속한 집합을 하나로 합치는 union 연산
19      u = find(u); v = find(v);       //u와 v의 부모를 받아와
20      if(u!=v)    parent[v] = u;      //그 들의 부모가 같지 않다면(둘이 같은 집합이 아니면) v의 부모를 u로 설정
21  }

84      for(int i = 0; i < 623; i++){
85          parent[i] = i;
86      }                                           //623개의 문서번호로 초기화
87
88      ofstream cl("clus.txt");                    //출력결과를 보기 위한 파일
89      int max, maxI, maxJ;                        //순서대로 유사도 최대값, 최대값일 때의 문서번호 두개
90      for(int m = 0; m < 193753; m++){                //upper triangle matrix의 수만큼의 반복
91          max = -1;                               //한번 최대값을 찾은 후 다시 초기화
92          for(int i = 0; i < 623; i++){
93              for(int j = i+1; j < 623; j++){
94                  if(SIM[i][j] > max){                    //만약 matrix의 유사도값이 max값보다 클 경우
95                      if(find(i) == find(j))  continue;   //조건검사 : i와 j의 부모가 같다면 둘은 같은 집합이므로 넘어감
96
97                  else{                                   //두 문서번호가 같은 집합이 아닐 경우
98                      max = SIM[i][j];                    //최대값을 현재 유사도값으로 바꾸고
99                      maxI = i; maxJ = j;                 //최대값을 가지는 문서번호를 저장해둠
100                 }
101             }
102             }
103         }
104         merge(maxI, maxJ);                          //두 문서번호를 merge(같은 그룹으로 )
105     }
106
107     for(int i = 0; i < 623; i++){
108         cout << parent[i] << " ";
109         cl << parent[i] << " ";
110     }
111     cout << endl;
112     cl << endl;                                     //출력문
```

```
C:₩Users₩abcdo₩Desktop₩2018-2-ir₩문서유사도₩EXE>g++ forclustering.cpp

C:₩Users₩abcdo₩Desktop₩2018-2-ir₩문서유사도₩EXE>a.exe
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 301 21 22 23 24 25 26 27 28 29 30 8 32 33 34 5 26 37 3 39 40 301 42 40
 44 44 40 47 48 25 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 3 66 67 68 69 24 71 19 73 74 75 76 77 78 79 56 81 40 83
84 85 56 87 88 89 90 91 92 93 94 95 40 30 98 66 100 66 301 103 104 105 58 107 108 58 110 111 112 113 114 115 116 40 118
119 120 121 122 123 124 125 126 127 66 66 130 40 3 133 6 40 48 301 93 139 54 141 142 301 144 145 26 147 148 149 40 151 7
9 153 154 40 4 157 158 159 160 301 8 54 164 165 166 167 168 54 159 171 172 5 174 175 176 177 178 301 180 6 182 183 184 1
85 301 187 188 189 56 191 192 193 15 195 196 197 198 199 200 201 54 6 204 54 206 207 208 92 210 92 212 88 66 215 216 58
218 301 4 221 15 6 224 301 226 301 54 30 230 40 44 233 58 235 174 237 40 239 301 241 165 243 244 245 301 8 159 249 30 40
 58 253 254 12 256 301 104 58 260 192 262 263 56 265 266 3 268 301 270 271 272 125 274 301 165 277 265 58 280 281 282 28
3 284 285 286 287 301 289 290 40 292 293 294 294 296 297 301 299 1 506 44 303 304 305 306 307 308 309 310 301 312 313 40
 315 301 317 305 319 320 321 322 323 324 325 326 294 328 329 8 331 301 333 334 335 306 337 19 339 340 341 342 343 344 16
5 346 347 348 349 301 351 352 353 354 355 301 325 358 359 360 361 362 363 364 301 366 367 368 369 21 371 372 373 8 375 3
76 87 378 27 294 286 382 40 384 385 386 39 388 2 40 391 392 393 394 395 396 397 398 399 400 401 15 403 404 405 27 407 40
8 409 262 411 412 413 414 294 416 417 418 419 420 421 422 15 424 425 426 427 428 429 98 431 432 301 434 435 436 48 438 3
04 440 441 322 1 444 272 446 447 301 449 450 3 452 453 48 455 456 457 40 459 460 25 462 463 464 465 466 467 468 469 470
471 325 473 474 40 476 477 478 479 480 481 320 483 484 40 486 487 301 88 490 44 492 27 5 495 455 320 498 499 500 501 502
 503 498 505 506 104 508 509 510 506 464 19 514 27 516 517 518 287 520 521 8 523 524 294 301 40 528 78 530 531 532 533 5
34 535 536 537 478 539 540 541 542 40 322 294 546 547 548 549 550 551 552 44 554 8 294 401 558 506 560 335 562 563 564 5
65 566 567 568 44 320 15 572 573 574 575 44 577 40 579 25 581 582 583 465 585 294 56 588 40 590 591 592 160 301 595 596
384 598 599 600 601 54 177 87 605 606 294 608 609 610 611 612 353 506 301 301 301 301 301 620 301 301
```

출력결과입니다.

결과는 좀 더 중간과정이 잘 보이기 위해 upper triangle matrix의 총 개수인 193753보다 적게 반복하여 캡쳐하였습니다.

위 결과를 해석하자면 만약 162번과 8번이 같은 클러스터로 속할 경우 162의 부모를 8번으로 지정하여 둘이 같은 그룹임을 알 수 있습니다.

만약 330번이 162번과 같은 클러스터에 속하게 된다면 330번은 162번의 부모를 따라 8번으로 표시됩니다.