



SNS COLLEGE OF ENGINEERING

Coimbatore-107



COURSE NAME: ANALYSIS OF ALGORITHM

II YEAR/ IV SEMESTER

UNIT – III

GREEDY TECHNIQUE

Topic

Greedy Technique: Kruskal's algorithm



Kruskal's Algorithm find MST.

Kruskal Algorithm

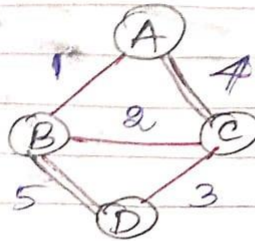
Steps to find MST:

- ① Sort all edges in increasing order by weight.
- ② Pick smallest edge. check any cycles are formed.
- ③ If cycle is formed, discard it.
- ④ If cycle is not formed, include that edge in MST.
- ⑤ Repeat until $(n-1)$ edges are formed.



Example 2

classmate
Date _____
Page _____



Finding MST (By Kruskal method)

Step 1: Sort the edges by their weight in Ascending order.

- * A-B : 1
- * B-C : 2
- * C-D : 3
- * A-C : 4
- * B-D : 5

Step 2: Add edges to MST that forms No cycle.

$A-B : 1$
 $B-C : 2$
 $C-D : 3$

4 vertices
 $edges = |V| - 1 = 4 - 1 = 3$
 Three edges with no cycles formed.

Step 3: Add the weight : $1 + 2 + 3 = 6$

MST for the graph = 6



Example : 2

Graph 'G' : Vertices A, B, C, D, E

Edges 'E' : A-B

(i) (A-B, 2)

(ii) (A-C, 3)

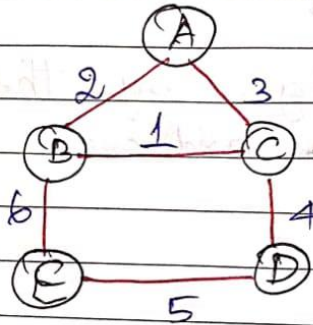
(iii) (B-C, 1)

(iv) (C-D, 4)

(v) (D-E, 5)

(vi) (B-E, 6)

Step 1: Form the Graph:



Step 2: Add Edges to MST one by one, that forms no cycle in Ascending order.

(i) B-C : 1

(ii) A-B : 2

(iii) A-C : 3

(iv) C-D : 4

(v) D-E : 5

(vi) B-E : 6

Step 3: Add Edges to MST one by one that forms no cycle.



classmate
Date _____
Page _____

(i) ✓ B-C : No cycle ; Add to MST
(ii) ✓ A-B : No cycle ; Add to MST
(iii) A-C : cycle formed i.e. A is already
X Connected to c through (A-B) &
(B-C). So skip this edge.
(iv) ✓ C-D : No cycle , Edge added to MST
(v) ✓ D-E : No cycle , Edge added to MST
(vi) B-E : cycle formed (B-A-C-D-E)
X So skip edge.

Final MST :

* Edges 'E' : (B-C ; 1), (A-B, 2), (C-D, 4)
(D-E, 5)

∴ Total weight = (1 + 2 + 4 + 5) = 12

5 vertices & 4 edges formed.
Thus MST for Graph is created.



Algorithm :

```
Algorithm kruskal ()
// problem description : Find MST using kruskal
// Input : Graph 'G'
// output : MST with its total cost.
count = 0
k = 0
sum = 0
for (i = 0 to totnodes)
parent [i] = i
while (count != totnodes - 1) do
{
pos = minimum ( totedges ); // Find min cost edge
if (pos == 1)
break;
v1 = G [ pos ]. v1
v2 = G [ pos ]. v2
i = find ( v1, parent )
j = find ( v2, parent )
if ( i != j ) then
{
tree [k][0] = v1 // stores edges of spanning tree
// (storing min edge in tree [])
tree [k][1] ← v2
k++
count ++;
sum + = G [ pos ]. cost // computing total cost of all the minimum distance.
```



accumulating total cost of MST
union (i, j, parent)

```
31 0  
32 1  
33 2  
34 3  
35 4  
36 5  
37 6  
38 7  
39 8  
40 9  
41 10  
42 11  
43 12  
44 13  
45 14  
46 15  
47 16  
48 17  
49 18  
50 19  
51 20  
52 21  
53 22  
54 23  
55 24  
56 25  
57 26  
58 27  
59 28  
60 29  
61 30  
62 31  
63 32  
64 33  
65 34  
66 35  
67 36  
68 37  
69 38  
70 39  
71 40  
72 41  
73 42  
74 43  
75 44  
76 45  
77 46  
78 47  
79 48  
80 49  
81 50  
82 51  
83 52  
84 53  
85 54  
86 55  
87 56  
88 57  
89 58  
90 59  
91 60  
92 61  
93 62  
94 63  
95 64  
96 65  
97 66  
98 67  
99 68  
100 69  
101 70  
102 71  
103 72  
104 73  
105 74  
106 75  
107 76  
108 77  
109 78  
110 79  
111 80  
112 81  
113 82  
114 83  
115 84  
116 85  
117 86  
118 87  
119 88  
120 89  
121 90  
122 91  
123 92  
124 93  
125 94  
126 95  
127 96  
128 97  
129 98  
130 99  
131 100  
132 101  
133 102  
134 103  
135 104  
136 105  
137 106  
138 107  
139 108  
140 109  
141 110  
142 111  
143 112  
144 113  
145 114  
146 115  
147 116  
148 117  
149 118  
150 119  
151 120  
152 121  
153 122  
154 123  
155 124  
156 125  
157 126  
158 127  
159 128  
160 129  
161 130  
162 131  
163 132  
164 133  
165 134  
166 135  
167 136  
168 137  
169 138  
170 139  
171 140  
172 141  
173 142  
174 143  
175 144  
176 145  
177 146  
178 147  
179 148  
180 149  
181 150  
182 151  
183 152  
184 153  
185 154  
186 155  
187 156  
188 157  
189 158  
190 159  
191 160  
192 161  
193 162  
194 163  
195 164  
196 165  
197 166  
198 167  
199 168  
200 169  
201 170  
202 171  
203 172  
204 173  
205 174  
206 175  
207 176  
208 177  
209 178  
210 179  
211 180  
212 181  
213 182  
214 183  
215 184  
216 185  
217 186  
218 187  
219 188  
220 189  
221 190  
222 191  
223 192  
224 193  
225 194  
226 195  
227 196  
228 197  
229 198  
230 199  
231 200  
232 201  
233 202  
234 203  
235 204  
236 205  
237 206  
238 207  
239 208  
240 209  
241 210  
242 211  
243 212  
244 213  
245 214  
246 215  
247 216  
248 217  
249 218  
250 219  
251 220  
252 221  
253 222  
254 223  
255 224  
256 225  
257 226  
258 227  
259 228  
260 229  
261 230  
262 231  
263 232  
264 233  
265 234  
266 235  
267 236  
268 237  
269 238  
270 239  
271 240  
272 241  
273 242  
274 243  
275 244  
276 245  
277 246  
278 247  
279 248  
280 249  
281 250  
282 251  
283 252  
284 253  
285 254  
286 255  
287 256  
288 257  
289 258  
290 259  
291 260  
292 261  
293 262  
294 263  
295 264  
296 265  
297 266  
298 267  
299 268  
300 269  
301 270  
302 271  
303 272  
304 273  
305 274  
306 275  
307 276  
308 277  
309 278  
310 279  
311 280  
312 281  
313 282  
314 283  
315 284  
316 285  
317 286  
318 287  
319 288  
320 289  
321 290  
322 291  
323 292  
324 293  
325 294  
326 295  
327 296  
328 297  
329 298  
330 299  
331 300  
332 301  
333 302  
334 303  
335 304  
336 305  
337 306  
338 307  
339 308  
340 309  
341 310  
342 311  
343 312  
344 313  
345 314  
346 315  
347 316  
348 317  
349 318  
350 319  
351 320  
352 321  
353 322  
354 323  
355 324  
356 325  
357 326  
358 327  
359 328  
360 329  
361 330  
362 331  
363 332  
364 333  
365 334  
366 335  
367 336  
368 337  
369 338  
370 339  
371 340  
372 341  
373 342  
374 343  
375 344  
376 345  
377 346  
378 347  
379 348  
380 349  
381 350  
382 351  
383 352  
384 353  
385 354  
386 355  
387 356  
388 357  
389 358  
390 359  
391 360  
392 361  
393 362  
394 363  
395 364  
396 365  
397 366  
398 367  
399 368  
400 369  
401 370  
402 371  
403 372  
404 373  
405 374  
406 375  
407 376  
408 377  
409 378  
410 379  
411 380  
412 381  
413 382  
414 383  
415 384  
416 385  
417 386  
418 387  
419 388  
420 389  
421 390  
422 391  
423 392  
424 393  
425 394  
426 395  
427 396  
428 397  
429 398  
430 399  
431 400  
432 401  
433 402  
434 403  
435 404  
436 405  
437 406  
438 407  
439 408  
440 409  
441 410  
442 411  
443 412  
444 413  
445 414  
446 415  
447 416  
448 417  
449 418  
450 419  
451 420  
452 421  
453 422  
454 423  
455 424  
456 425  
457 426  
458 427  
459 428  
460 429  
461 430  
462 431  
463 432  
464 433  
465 434  
466 435  
467 436  
468 437  
469 438  
470 439  
471 440  
472 441  
473 442  
474 443  
475 444  
476 445  
477 446  
478 447  
479 448  
480 449  
481 450  
482 451  
483 452  
484 453  
485 454  
486 455  
487 456  
488 457  
489 458  
490 459  
491 460  
492 461  
493 462  
494 463  
495 464  
496 465  
497 466  
498 467  
499 468  
500 469  
501 470  
502 471  
503 472  
504 473  
505 474  
506 475  
507 476  
508 477  
509 478  
510 479  
511 480  
512 481  
513 482  
514 483  
515 484  
516 485  
517 486  
518 487  
519 488  
520 489  
521 490  
522 491  
523 492  
524 493  
525 494  
526 495  
527 496  
528 497  
529 498  
530 499  
531 500  
532 501  
533 502  
534 503  
535 504  
536 505  
537 506  
538 507  
539 508  
540 509  
541 510  
542 511  
543 512  
544 513  
545 514  
546 515  
547 516  
548 517  
549 518  
550 519  
551 520  
552 521  
553 522  
554 523  
555 524  
556 525  
557 526  
558 527  
559 528  
560 529  
561 530  
562 531  
563 532  
564 533  
565 534  
566 535  
567 536  
568 537  
569 538  
570 539  
571 540  
572 541  
573 542  
574 543  
575 544  
576 545  
577 546  
578 547  
579 548  
580 549  
581 550  
582 551  
583 552  
584 553  
585 554  
586 555  
587 556  
588 557  
589 558  
590 559  
591 560  
592 561  
593 562  
594 563  
595 564  
596 565  
597 566  
598 567  
599 568  
600 569  
601 570  
602 571  
603 572  
604 573  
605 574  
606 575  
607 576  
608 577  
609 578  
610 579  
611 580  
612 581  
613 582  
614 583  
615 584  
616 585  
617 586  
618 587  
619 588  
620 589  
621 590  
622 591  
623 592  
624 593  
625 594  
626 595  
627 596  
628 597  
629 598  
630 599  
631 600  
632 601  
633 602  
634 603  
635 604  
636 605  
637 606  
638 607  
639 608  
640 609  
641 610  
642 611  
643 612  
644 613  
645 614  
646 615  
647 616  
648 617  
649 618  
650 619  
651 620  
652 621  
653 622  
654 623  
655 624  
656 625  
657 626  
658 627  
659 628  
660 629  
661 630  
662 631  
663 632  
664 633  
665 634  
666 635  
667 636  
668 637  
669 638  
670 639  
671 640  
672 641  
673 642  
674 643  
675 644  
676 645  
677 646  
678 647  
679 648  
680 649  
681 650  
682 651  
683 652  
684 653  
685 654  
686 655  
687 656  
688 657  
689 658  
690 659  
691 660  
692 661  
693 662  
694 663  
695 664  
696 665  
697 666  
698 667  
699 668  
700 669  
701 670  
702 671  
703 672  
704 673  
705 674  
706 675  
707 676  
708 677  
709 678  
710 679  
711 680  
712 681  
713 682  
714 683  
715 684  
716 685  
717 686  
718 687  
719 688  
720 689  
721 690  
722 691  
723 692  
724 693  
725 694  
726 695  
727 696  
728 697  
729 698  
730 699  
731 700  
732 701  
733 702  
734 703  
735 704  
736 705  
737 706  
738 707  
739 708  
740 709  
741 710  
742 711  
743 712  
744 713  
745 714  
746 715  
747 716  
748 717  
749 718  
750 719  
751 720  
752 721  
753 722  
754 723  
755 724  
756 725  
757 726  
758 727  
759 728  
760 729  
761 730  
762 731  
763 732  
764 733  
765 734  
766 735  
767 736  
768 737  
769 738  
770 739  
771 740  
772 741  
773 742  
774 743  
775 744  
776 745  
777 746  
778 747  
779 748  
780 749  
781 750  
782 751  
783 752  
784 753  
785 754  
786 755  
787 756  
788 757  
789 758  
790 759  
791 760  
792 761  
793 762  
794 763  
795 764  
796 765  
797 766  
798 767  
799 768  
800 769  
801 770  
802 771  
803 772  
804 773  
805 774  
806 775  
807 776  
808 777  
809 778  
810 779  
811 780  
812 781  
813 782  
814 783  
815 784  
816 785  
817 786  
818 787  
819 788  
820 789  
821 790  
822 791  
823 792  
824 793  
825 794  
826 795  
827 796  
828 797  
829 798  
830 799  
831 800  
832 801  
833 802  
834 803  
835 804  
836 805  
837 806  
838 807  
839 808  
840 809  
841 810  
842 811  
843 812  
844 813  
845 814  
846 815  
847 816  
848 817  
849 818  
850 819  
851 820  
852 821  
853 822  
854 823  
855 824  
856 825  
857 826  
858 827  
859 828  
860 829  
861 830  
862 831  
863 832  
864 833  
865 834  
866 835  
867 836  
868 837  
869 838  
870 839  
871 840  
872 841  
873 842  
874 843  
875 844  
876 845  
877 846  
878 847  
879 848  
880 849  
881 850  
882 851  
883 852  
884 853  
885 854  
886 855  
887 856  
888 857  
889 858  
890 859  
891 860  
892 861  
893 862  
894 863  
895 864  
896 865  
897 866  
898 867  
899 868  
900 869  
901 870  
902 871  
903 872  
904 873  
905 874  
906 875  
907 876  
908 877  
909 878  
910 879  
911 880  
912 881  
913 882  
914 883  
915 884  
916 885  
917 886  
918 887  
919 888  
920 889  
921 890  
922 891  
923 892  
924 893  
925 894  
926 895  
927 896  
928 897  
929 898  
930 899  
931 900  
932 901  
933 902  
934 903  
935 904  
936 905  
937 906  
938 907  
939 908  
940 909  
941 910  
942 911  
943 912  
944 913  
945 914  
946 915  
947 916  
948 917  
949 918  
950 919  
951 920  
952 921  
953 922  
954 923  
955 924  
956 925  
957 926  
958 927  
959 928  
960 929  
961 930  
962 931  
963 932  
964 933  
965 934  
966 935  
967 936  
968 937  
969 938  
970 939  
971 940  
972 941  
973 942  
974 943  
975 944  
976 945  
977 946  
978 947  
979 948  
980 949  
981 950  
982 951  
983 952  
984 953  
985 954  
986 955  
987 956  
988 957  
989 958  
990 959  
991 960  
992 961  
993 962  
994 963  
995 964  
996 965  
997 966  
998 967  
999 968  
1000 969
```

classmate
Date _____
Page _____

if (cost > INFINITY)

(count = tot nodes - 1) then

for (i = 0 to tot nodes - 1) then

{

write (tree [i] [0], tree [i] [1])

}

write ("cost of spanning Tree", sum)

}

Analysis: all Time complexity:

1. Sorting n Edges : $O(E \log E)$ times

2. Union - Find operations : $O(E \log V)$ times

3. Final complexity :

$O(E \log E) + E \log V = O(E \log E)$

$\therefore E \log E$ dominates $E \log V$.

Best & Average case :

$O(E \log E)$

Worst case :

$O(E \log V)$ (occurs when sorting is dominant)

Space Complexity:

1. store E edges $\rightarrow O(E)$

2. store V vertices $\rightarrow O(V)$

3. MST stores V-1 edges $\rightarrow O(V)$

Total space complexity = $O(V + E)$