Analysis: BIC
Bicriterial routing

BOI'02, day ??? *??.04.2002*

1 Solution

`prog/bic.cpp` Let K be the upper limit on the road toll. The cost of any route is then limited by nK.

The exemplary solution is a dynamic one. For each k from 0 to nK and for all vertexes one computes the minimal traveling time from s to the given vertex with the fee equal exactly k.

Initialization for $k = 0$ is just an application of the Dijkstra's algorithm for the graph limited to these edges with toll $c = 0$. For bigger k we first calculate the minimal time basing on the previously computed results and assuming that the last edge on the route has a positive toll (this is done for all edges). Next we take into account edges with toll $c = 0$, again using Dijkstra's algorithm.

Results calculated for vertex e give us the final result. This algorithm has time complexity $O((n + mlogn) * nK)$ — Dijkstra's algorithm is implemented using a heap. Memory complexity is $O(nK + m)$, since we can focus just on last $K + 1$ rows of the array of minimal traveling times (except for e).

2 Tests

File `bicgen.cpp` contains a generator of tests.

- `bic0.IN` (ϵ sek.) test from the problem text
- `bic1a.IN` (ϵ sek.) random, n=5, m=20
- `bic1b.IN` (0.1 sek.) not connected graph, n=10, m=50
- `bic2.IN` (0.1 sek.) random, n=2, m=300
- `bic3.IN` (0.3 sek.) large result, n=10, m=180
- `bic4.IN` (0.4 sek.) random, n=20, m=60
- `bic5.IN` (3.7 sek.) random, n=50, m=200
- `bic6.IN` (2.2 sek.) many routes (not fee-time pairs), n=50, m=147
- `bic7.IN` (8.0 sek.) random, n=75, m=240
- `bic8.IN` (10.3 sek.) random, n=88, m=230
- `bic9.IN` (9.1 sek.) large result, n=100, m=297
- `bic10a.IN` (13.8 sek.) random, n=98, m=300

- `bic10b.IN` (12.3 sek.) not connected graph, n=100, m=300

Pairs of tests 1a and 1b, 10a and 10b should be graded on a "conjunction" basis. It prevents from awarding points to solutions outputting just 0.